PENERAPAN METODE *TRANSFER LEARNING* DENGAN ARSITEKTUR RESNET-50 UNTUK NAVIGASI ROBOT BERODA BERBASIS VISI KOMPUTER

SKRIPSI

Oleh:
MUHAMMAD ANDREAN HIDAYATULLAH
NIM. 220605110126

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM
MALANG
2026

PENERAPAN METODE *TRANSFER LEARNING* DENGAN ARSITEKTUR RESNET-50 UNTUK NAVIGASI ROBOT BERODA BERBASIS VISI KOMPUTER

SKRIPSI

Diajukan kepada:
Universitas Islam Negeri Maulana Malik Ibrahim Malang
Untuk memenuhi Salah Satu Persyaratan dalam
Memperoleh Gelar Sarjana Komputer (S.Kom)

Oleh :
MUHAMMAD ANDREAN HIDAYATULLAH
NIM. 220605110126

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM MALANG
2026

HALAMAN PERSETUJUAN

PENERAPAN METODE *TRANSFER LEARNING* DENGAN ARSITEKTUR *RESNET-50* UNTUK NAVIGASI ROBOT BERODA BERBASIS VISI KOMPUTER

SKRIPSI

Oleh:
MUHAMMAD ANDREAN HIDAYATULLAH
220605110126

Telah Diperiksa dan Disetujui untuk Diuji:
Tanggal: 18 Juni 2026

Pembimbing I,	Pembimbing II,
Shoffin Nahwa Utama, M.T	Fatchurrochman, M.Kom
NIP. 19860703 202012 1 003	NIP. 19700731 200501 1 002

Mengetahui,
Ketua Program Studi Teknik Informatika
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang

Sunriyono, M.Kom
NIP. 19841010 201903 1 012

HALAMAN PENGESAHAN

PENERAPAN METODE *TRANSFER LEARNING* DENGAN ARSITEKTUR *RESNET-50* UNTUK NAVIGASI ROBOT BERODA BERBASIS VISI KOMPUTER

SKRIPSI

Oleh:
MUHAMMAD ANDREAN HIDAYATULLAH
NIM. 220605110126

Telah Dipertahankan di Depan Dewan Penguji Skripsi
dan Dinyatakan Diterima Sebagai Salah Satu Persyaratan
Untuk Memperoleh Gelar Sarjana Komputer (S.Kom)
Pada Tanggal: 18 Juni 2026

Susunan Dewan Penguji

Ketua Penguji	: Johan Ericka Wahyu Prakasa, M.Kom NIP. 19831213 201903 1 004	( )
Anggota Penguji I	: Ajib Hanani, M.T NIP. 19840731 202321 1 013	( )
Anggota Penguji II	: Shoffin Nahwa Utama, M.T NIP. 19860703 202012 1 003	( )
Anggota Penguji III	: Fatchurrochman, M.Kom NIP. 19700731 200501 1 002	( )

Mengetahui dan Mengesahkan,
Ketua Program Studi Teknik Informatika
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang

Supriyono, M.Kom
NIP. 19841010 201903 1 012

PERNYATAAN KEASLIAN TULISAN

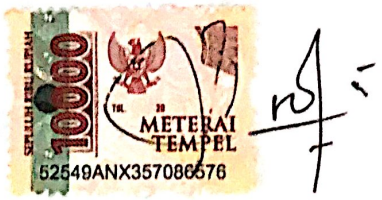
Saya yang bertanda tangan di bawah ini:

Nama	: MUHAMMAD ANDREAN HIDAYATULLAH
NIM	: 220605110126
Fakultas / Program Studi	: Sains dan Teknologi / Teknik Informatika
Judul Skripsi	: Penerapan Metode *Transfer Learning* Dengan Arsitektur Resnet-50 Untuk Navigasi Robot Beroda Berbasis Visi Komputer

Menyatakan dengan sebenarnya bahwa Skripsi yang saya tulis ini benar-benar merupakan hasil karya saya sendiri, bukan merupakan pengambil alihan data, tulisan, atau pikiran orang lain yang saya akui sebagai hasil tulisan atau pikiran saya sendiri, kecuali dengan mencantumkan sumber cuplikan pada daftar pustaka.

Apabila dikemudian hari terbukti atau dapat dibuktikan skripsi ini merupakan hasil jiplakan, maka saya bersedia menerima sanksi atas perbuatan tersebut.

Malang, 26 Juni 2026
Yang membuat pernyataan,



MUHAMMAD ANDREAN HIDAYATULLAH
NIM. 220605110126

MOTTO

"The story wouldn't end unless the reader gave up on the story"

(Han Soo Young – Omniscient Reader's Viewpoint)

HALAMAN PERSEMBAHAN

Alhamdulillahi rabbil 'alamin, segala puji bagi Allah Swt. atas segala rahmat, nikmat, dan karunia-Nya. Shalawat serta salam semoga senantiasa tercurah kepada Nabi Muhammad saw., beserta keluarga, sahabat, dan seluruh umatnya. Atas izin dan pertolongan Allah Swt., penulis diberikan kemudahan dan kekuatan hingga dapat menyelesaikan skripsi ini.

Skripsi ini penulis persembahkan kepada kedua orang tua tercinta sebagai wujud bakti, cinta, kasih sayang, dan rasa terima kasih atas segala doa, pengorbanan, serta dukungan yang telah diberikan. Kehadiran dan doa mereka menjadi alasan utama yang menguatkan penulis untuk terus berjuang hingga mampu menyelesaikan skripsi ini.

Kepada saudara-saudara tercinta dan seluruh keluarga besar, penulis mempersembahkan karya sederhana ini sebagai bentuk rasa syukur dan terima kasih atas perhatian, motivasi, dukungan, serta doa yang selalu menyertai perjalanan penulis dalam menempuh pendidikan hingga terselesaikannya skripsi ini.

Kepada teman-teman dan seluruh pihak yang telah turut membantu penulis dalam menyelesaikan skripsi ini, terima kasih atas doa, dukungan, dan segala bentuk bantuan yang diberikan sehingga skripsi ini dapat terselesaikan dengan baik.

KATA PENGANTAR

Alhamdulillāhirabbil ‘ālamīn, segala puji bagi Allah *Subḥānahu wa Ta‘ālā*. yang telah melimpahkan rahmat, taufik, hidayah, serta karunia-Nya sehingga penulis dapat menyelesaikan skripsi yang berjudul *“Penerapan Metode Transfer Learning Dengan Arsitektur Resnet-50 Untuk Navigasi Robot Beroda Berbasis Visi Komputer”* dengan baik. Shalawat serta salam semoga senantiasa tercurah kepada junjungan kita Nabi Muhammad *Shollallohu ‘Alaihi Wasallam*, beserta keluarga, sahabat, dan seluruh pengikutnya hingga akhir zaman. Skripsi ini disusun sebagai salah satu syarat untuk memperoleh gelar Sarjana Komputer pada Program Studi Teknik Informatika, Fakultas Sains dan Teknologi, Universitas Islam Negeri Maulana Malik Ibrahim Malang. Oleh karena itu, pada kesempatan ini penulis ingin menyampaikan ucapan terima kasih yang sebesar-besarnya kepada:

1. Prof. Dr. Hj. Ilfi Nurdiana, M.Si., CAHRM., CRMP., selaku Rektor Universitas Islam Negeri Maulana Malik Ibrahim Malang.
2. Dr. Agus Mulyono, M.Kes., selaku Dekan Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang.
3. Supriyono, M.Kom., selaku Ketua Program Studi Teknik Informatika Universitas Islam Negeri Maulana Malik Ibrahim Malang.
4. Shoffin Nahwa Utama, M.T., selaku dosen pembimbing I dan Fatchurrochman, M.Kom., selaku dosen pembimbing II. Penulis menyampaikan apresiasi dan rasa terima kasih yang mendalam atas segala waktu, kesabaran, serta dedikasi beliau berdua dalam memberikan

bimbingan, arahan, dan motivasi, sehingga penyusunan skripsi ini dapat berjalan dengan lancar dan selesai tepat waktu.

5. Johan Ericka Wahyu Prakasa, M.Kom., sebagai dosen penguji I dan Ajib Hanani, M.T., sebagai dosen penguji II, yang telah meluangkan waktu untuk memberikan evaluasi komprehensif, kritik konstruktif, serta saran-saran perbaikan yang sangat berharga guna penyempurnaan kualitas penelitian dan penulisan skripsi ini.
6. Seluruh dosen Program Studi Teknik Informatika Universitas Islam Negeri Maulana Malik Ibrahim Malang, penulis menyampaikan ucapan terima kasih atas dedikasi dalam menyalurkan ilmu pengetahuan yang tak ternilai selama masa perkuliahan. Semoga segala kebaikan dan keilmuan yang telah diajarkan menjadi amal jariyah serta senantiasa memberikan keberkahan bagi langkah penulis ke depan.
7. Untuk surga yang telapak kakinya selalu penulis rindukan, Ibunda tercinta, Suyaminah. Tidak ada satu pun pencapaian dalam lembar skripsi ini yang luput dari campur tangan doa Ibu di sepertiga malam. Terima kasih untuk setiap peluh yang disembunyikan, untuk cinta yang tak pernah menuntut syarat, dan untuk keyakinan utuh yang Ibu berikan saat penulis hampir menyerah pada diri sendiri. Karya kecil ini adalah bentuk bakti yang tak seberapa, namun penulis persembahkan sepenuhnya untuk membalas seluruh rida dan air mata yang Ibu korbankan demi melihat penulis berada di titik ini.

8. Teruntuk Ayahanda Ahmad. Terima kasih atas segala kerja keras, keteladanan, dan tanggung jawab yang selalu Ayah ajarkan. Dukungan dan pengorbanan Ayah adalah fondasi utama yang mengantarkan penulis menyelesaikan pendidikan ini. Skripsi ini merupakan wujud bakti dan rasa hormat penulis yang tak terhingga.
9. Kepada saudara-saudari penulis yang menjadi tempat pulang paling hangat, Kakak Clara Dea Amelia Solihah, Kakak Elena Amelia Solihah, dan jagoan kecil, Adik Muhammad Dzaky Habibi. Terima kasih telah menjadi warna dan jangkar kewarasan dalam perjalanan ini. Kehadiran, tawa, serta ikatan batin persaudaraan adalah penawar lelah terbaik yang selalu memantik kembali semangat penulis di saat-saat paling berat.
10. Seluruh keluarga besar penulis yang tidak dapat disebutkan satu per satu, terima kasih atas doa, dukungan, perhatian, dan semangat yang telah diberikan kepada penulis selama menempuh pendidikan.
11. Untuk kawan-kawan seperjuangan yang lebih pantas disebut sebagai saudara tak sedarah, Kiageng Nasrokh Mangkunegara Putra, Ahmad Fadhli Dzilikrom, Muhammad Alif Atallah, Mohammad Zidan Alvaro, Muhammad Annas Darunnaja, Muhammad Haikal Azzadin, Hasyim Husein Al-habsyi, Nizam Hukmul Kautsar, Nur Muhammad Najiburrohman, Akbar Bimantara Trahestiawan, dan Ega Okta Syafwan Prayoga. Melewati masa-masa penuh tekanan, malam-malam tanpa tidur menghadapi *error* baris kode, hingga krisis kepercayaan diri, semuanya terasa lebih manusiawi karena tawa dan solidaritas kalian. Terima kasih telah menjadi rumah kedua

dan bahu untuk saling bersandar, hingga akhirnya kita bisa menyentuh garis akhir ini bersama-sama.

12. Ucapan terima kasih penulis sampaikan kepada keluarga besar Teknik Informatika angkatan 2022 Infinity, yang telah memberikan kebersamaan, dukungan, dan pengalaman berharga selama masa perkuliahan.
13. Teruntuk ruang aman dan tempat mereda, majikan Clow. Terima kasih karena telah memilih untuk tetap tinggal dan menggenggam tangan penulis di saat proses ini berada di titik paling redup. Terima kasih untuk kesabaran yang tak bertepi, untuk setiap usapan dan kalimat validasi yang menenangkan isi kepala, serta untuk selalu bersedia merayakan hal-hal kecil bersama penulis. Skripsi ini juga menjadi saksi atas doa dan kebaikan hatimu yang terus membersamai penulis hingga kalimat terakhir ini dituliskan.
14. Kepada seluruh kawan, sahabat, dan orang-orang baik yang pernah bersinggungan selama masa perkuliahan, yang namanya tidak dapat penulis sebutkan satu per satu. Terima kasih karena telah meninggalkan jejak kebaikan dalam proses pendewasaan. Setiap senyum, bantuan, dan waktu yang dibagi bersama adalah kepingan *puzzle* yang ikut menyempurnakan perjalanan skripsi ini. Semoga langkah kalian di mana pun berada selalu diiringi kemudahan.

Penulis menyadari bahwa skripsi ini masih jauh dari kata sempurna dan masih terdapat berbagai keterbatasan yang memerlukan penyempurnaan di masa mendatang. Namun demikian, penulis meyakini bahwa proses belajar tidak

memiliki batas akhir, sehingga setiap kritik dan saran yang membangun akan menjadi bekal berharga untuk terus berkembang. Bagi penulis, skripsi yang baik bukanlah skripsi yang sempurna, melainkan skripsi yang dapat diselesaikan dengan penuh tanggung jawab dan tepat pada waktunya. Oleh karena itu, penulis berharap skripsi ini dapat memberikan manfaat, baik bagi pengembangan ilmu pengetahuan maupun bagi pihak-pihak yang membutuhkan sebagai referensi untuk penelitian selanjutnya.

Malang, 26 Juni 2026

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

ABSTRAK

Hidayatullah, Muhammad Andrean. 2026. **Penerapan Metode *Transfer Learning* dengan Arsitektur ResNet-50 untuk Navigasi Robot Beroda Berbasis Visi Komputer**. Skripsi. Program Studi Teknik Informatika Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang. Pembimbing: (I) Shoffin Nahwa Utama, M.T, (II) Fatchurrochman, M.Kom.

.

Kata Kunci: Navigasi Robot Beroda, Pengenalan Gestur Tangan, ResNet-50, *Transfer Learning*, Visi Komputer.

Metode kendali robot konvensional berbasis kontak fisik telah banyak digunakan, namun kini dihadapkan pada tantangan dinamika kebutuhan operasional yang terus berkembang. Oleh karena itu, diperlukan sebuah alternatif baru yang dirancang untuk menawarkan kepraktisan dan fleksibilitas gerak yang lebih baik bagi penggunanya. Penelitian ini bertujuan mengembangkan sistem navigasi robot beroda nirkontak yang mampu mengenali gestur tangan menggunakan metode *Transfer Learning* pada arsitektur ResNet-50. Model dilatih menggunakan 8.756 citra dari *dataset* HaGRID ke dalam lima kelas gestur yaitu kepalan tangan, acungan jempol, isyarat dua jari, bentuk jari OK, dan telapak tangan terbuka. Hasil pengujian menunjukkan performa klasifikasi yang sangat baik dengan akurasi *test set* 91,10%, *mean* AUC 0,9918, dan *gap overfitting* yang rendah yaitu 4,50%. Pada tahap *deployment*, pengujian di lingkungan laptop sangat responsif dengan waktu eksekusi 50,15 ms. Saat diimplementasikan pada perangkat *edge*, penggunaan perangkat final Raspberry Pi 4 Model B (RAM 4 GB) dengan model presisi penuh (FP32) mencatatkan waktu eksekusi 385,39 ms, memangkas waktu inferensi secara signifikan dibandingkan Raspberry Pi 3 Model B (2.984,84 ms). Pengujian fungsionalitas (*black box testing*) menunjukkan tingkat keberhasilan operasional 74,3%, dan meningkat hingga 87,6% pada penggunaan orientasi tangan wajar (tanpa kemiringan vertikal ekstrem). Kesimpulannya, metode *Transfer Learning* pada arsitektur *ResNet-50* terbukti handal untuk klasifikasi gestur navigasi robot.

ABSTRACT

Hidayatullah, Muhammad Andrean. 2026. **Application of the Transfer Learning Method with the ResNet-50 Architecture for Computer Vision-Based Navigation of Wheeled Robots**. Undergraduate Thesis. Department of Computer Science, Faculty of Science and Technology, Maulana Malik Ibrahim State Islamic University, Malang. Supervisors: (1) Shoffin Nahwa Utama, M.T, (II) Fatchurrochman, M.Kom.

Keywords: Computer Vision, Hand Gesture Recognition, ResNet-50, Transfer Learning, Wheeled Robot Navigation.

Conventional robot control methods based on physical contact have been widely used, but are now faced with the challenge of ever-evolving operational requirements. Therefore, a new alternative is required, designed to offer greater practicality and flexibility of movement for its users. This research aims to develop a contactless wheeled robot navigation system capable of recognising hand gestures using the Transfer Learning method on the ResNet-50 architecture. The model was trained using 8,756 images from the HaGRID dataset across five gesture classes: a clenched fist, a thumbs-up, a two-finger signal, an 'OK' sign, and an open palm. Test results demonstrate excellent classification performance, with a test set accuracy of 91.10%, a mean AUC of 0.9918, and a low overfitting gap of 4.50%. During the deployment phase, testing on a laptop proved highly responsive, with an execution time of 50.15 ms. When implemented on an edge device, the final Raspberry Pi 4 Model B (4 GB RAM) running the full-precision (FP32) model recorded an execution time of 385.39 ms, significantly reducing inference time compared to the Raspberry Pi 3 Model B (2,984.84 ms). Functionality testing (black-box testing) showed an operational success rate of 74.3 per cent, rising to 87.6 per cent when using natural hand orientation (without extreme vertical tilt). In conclusion, the Transfer Learning method on the ResNet-50 architecture proved reliable for classifying robot navigation gestures.

البحث مستخلص

هيدايات الله، محمد أندريان. **٢٠٢٦**. **تطبيق طريقة التعلم بالنقل باستخدام بنية ResNet-50 لتوجيه الروبوتات ذات العجلات القائمة على الرؤية الحاسوبية. أطروحة تخرج**. برنامج هندسة المعلوماتية، كلية العلوم والتكنولوجيا، جامعة مولانا مالك إبراهيم الإسلامية الحكومية، مالانج. المشرفان: (I) شوفين ناهوا أوتاما، ماجستير في الهندسة، (II) فاتشوروشمان، ماجستير في علوم الحاسوب.

الكلمات المفتاحية :توجيه الروبوتات ذات العجلات، التعرف على إيماءات اليد، ResNet-50، التعلم النقلي، الرؤية الحاسوبية.

لقد استخُدمت طرق التحكم التقليدية في الروبوتات القائمة على التلامس المادي على نطاق واسع، لكنها تواجه حالياً تحديات ناجمة عن ديناميكيات الاحتياجات التشغيلية المتطورة باستمرار. ولذلك، هناك حاجة إلى بديل جديد مصمم لتوفير مزيد من العملانية ومرونة الحركة للمستخدمين. يهدف هذا البحث إلى تطوير نظام ملاحة لروبوت ذي عجلات بدون تلامس قادر على التعرف على إيماءات اليد باستخدام طريقة التعلم النقلي (Learning Transfer) على بنية ResNet-50. تم تدريب النموذج باستخدام 8,756 صورة من مجموعة بيانات HaGRID على خمس فئات من الإيماءات وهي: قبضة اليد، وإشارة الإبهام، وإشارة الإصبعين، وإشارة OK، وراحة اليد المفتوحة. أظهرت نتائج الاختبار أداءً تصنيفياً ممتازًا، حيث بلغت دقة مجموعة الاختبار 91,10٪، ومتوسط AUC 0,9918، وفجوة التكيف المفرط منخفضة عند 4,50٪. في مرحلة النشر، أظهر الاختبار في بيئة الكمبيوتر المحمول استجابةً عاليةً مع زمن تنفيذ قدره 50,15 مللي ثانية. وعند التنفيذ على أجهزة الحافة، سجل استخدام الجهاز النهائي Raspberry Pi 4 Model B (ذاكرة الوصول العشوائي 4 جيجابايت) مع نموذج الدقة الكاملة (FP32) وقت تنفيذ قدره 385,39 مللي ثانية، مما أدى إلى تقليص وقت الاستدلال بشكل كبير مقارنةً بجهاز 2.984,84(Pi 3 Model B Raspberry مللي ثانية). أظهرت اختبارات الوظائف (اختبار الصندوق الأسود) معدل نجاح تشغيلي بلغ 74,3٪، وارتفع هذا المعدل إلى 87,6٪ عند استخدام اتجاه اليد الطبيعي (بدون ميل رأسي شديد). وختاماً، أثبتت طريقة «التعلم بالنقل» (Learning Transfer) في بنية ResNet-50 فعاليتها في تصنيف إيماءات التنقل للروبوت.

BAB I

PENDAHULUAN

1.1 Latar Belakang

Interaksi manusia-robot (*Human-Robot Interaction* atau HRI) semakin memegang peranan vital di era modern, terutama seiring dengan kemunculan revolusi industri 4.0 dan perkembangan robotika layanan. Sinergi yang baik antara manusia dan robot terbukti mampu meningkatkan produktivitas, akurasi, serta fleksibilitas di berbagai sektor strategis seperti manufaktur, kesehatan, dan pertanian. Peningkatan kinerja ini didorong oleh integrasi kecerdasan buatan (*Artificial Intelligence*), yang memungkinkan robot untuk beradaptasi secara cerdas terhadap perilaku dan preferensi manusia, sehingga menciptakan lingkungan kerja yang lebih efisien dan aman (Vemuri & Thaneeru, 2023).

Perkembangan teknologi robotika dan kecerdasan buatan saat ini telah merambah ke berbagai aspek kehidupan. Kehadiran teknologi ini sejatinya selaras dengan fitrah manusia sebagai *khalifah fil ardh* yang dibekali kemampuan intelektual untuk mengelola alam semesta. Salah satu landasan filosofis pengembangan teknologi cerdas ini dapat ditemukan dalam firman Allah SWT pada *QS. Al-Baqarah* ayat 31:

وَعَلَّمَ اٰدَمَ الْاَسْمَاۤءَ كُلَّهَا ثُمَّ عَرَضَهُمْ عَلَى الْمَلٰۤىِٕكَةِ فَقَالَ اَنْۢبِـُٔوْنِيْ بِاَسْمَاۤءِ هٰٓؤُلَاۤءِ اِنْ كُنْتُمْ صٰدِقِيْنَ

"Dan Dia ajarkan kepada Adam nama-nama (benda) semuanya, kemudian Dia perlihatkan kepada para malaikat, seraya berfirman, 'Sebutkan kepada-Ku nama-nama (benda) ini, jika kamu yang benar!'" (*QS. Al-Baqarah*: 31).

Berdasarkan tafsir Kementerian Agama RI, ayat ini menunjukkan salah satu sisi keutamaan manusia, yaitu kemampuan kognitif untuk mengenali, memahami, dan menamai benda-benda serta memanfaatkannya dalam kehidupan (Qur'an Kemenag, 2019). Kemampuan mengidentifikasi objek inilah yang menjadi dasar pengembangan ilmu pengetahuan dan teknologi.

Dalam konteks penelitian ini berhubungan sebagaimana manusia diajarkan mengenali objek. Sistem kecerdasan buatan (ResNet-50) dalam penelitian ini juga "diajarkan" melalui proses *training* untuk mengenali pola visual gestur tangan dan memberinya label (seperti 'Maju' atau 'Stop'). Hal ini merupakan bentuk mimikri sederhana dari potensi intelektual manusia untuk menciptakan alat bantu demi kemudahan hidup.

Agar HRI berfungsi optimal, diperlukan sistem kontrol yang mampu menerjemahkan niat manusia menjadi tindakan robot secara presisi. Kualitas sistem kontrol menjadi penentu utama tingkat fungsionalitas dan kemudahan penggunaan aplikasi *robotic* (Lee dkk., 2024). Saat ini, metode konvensional yang mengandalkan perangkat fisik seperti *joystick*, *remote control*, atau *keyboard* masih mendominasi. Meskipun fungsional, metode ini memiliki keterbatasan signifikan, seperti sifatnya yang kaku, mengharuskan operator mempelajari keterampilan khusus, serta membatasi mobilitas karena memerlukan kontak fisik langsung. Hal ini menjadi kendala serius di lingkungan yang membutuhkan sterilitas tinggi atau fleksibilitas gerak.

Metode konvensional tersebut sering kali dirasa "mempersulit" dalam kondisi tertentu, yang secara implisit bertentangan dengan semangat *taysir*

(kemudahan). Kendala muncul ketika tangan operator kotor, sedang memegang benda lain, atau ketika perangkat pengendali mengalami kerusakan. Sebagai solusi, penelitian HRI beralih ke metode interaksi yang lebih intuitif, salah satunya adalah sistem kontrol berbasis gestur tangan. Pengenalan gestur menawarkan cara alami dan efisien untuk mengendalikan robot tanpa kontak (*contactless*), yang sangat krusial untuk skenario seperti pengendalian mesin jarak jauh di pabrik kimia atau lingkungan medis yang steril (Rohman dkk., 2022). Studi terbaru pun menunjukkan bahwa penggunaan gestur dapat mereduksi waktu pelatihan operator secara signifikan (Angelidis & Bampis, 2025).

Namun, mewujudkan sistem pengenalan gestur yang handal bukanlah tanpa tantangan. Metode pemrosesan citra klasik (*non-AI*) sering kali gagal menghadapi kondisi dunia nyata yang dinamis, seperti perubahan pencahayaan, latar belakang yang kompleks (*background clutter*), serta variasi ukuran dan warna kulit (Bansal dkk., 2024). Untuk mengatasi kelemahan ini, pendekatan berbasis *Deep Learning*, khususnya *Convolutional Neural Networks* (CNN), telah menjadi standar baru. Meskipun demikian, penerapan CNN pada perangkat kendali robot menghadapi kendala komputasi dan arsitektural. Semakin dalam lapisan jaringan, sering kali justru memicu masalah *vanishing gradient*, yang membuat model sulit dilatih dan membutuhkan sumber daya komputasi masif (Bandini & Zariffa, 2022). Selain itu, melatih model dari awal (*scratch*) membutuhkan dataset yang sangat besar, padahal data gestur spesifik untuk navigasi robot sering kali terbatas.

Untuk menjawab tantangan tersebut, arsitektur *Residual Network* (ResNet) hadir sebagai solusi inovatif. Dengan mekanisme *skip connection*, ResNet mampu

mengatasi degradasi pada jaringan dalam, memungkinkan model seperti ResNet-50 mempelajari fitur visual kompleks dengan akurasi tinggi namun tetap efisien. Guna mengatasi keterbatasan data latih, penelitian ini menerapkan teknik *Transfer Learning*. Teknik ini memanfaatkan bobot model yang telah dilatih pada dataset masif (ImageNet) untuk diekstraksi pengetahuannya dan disesuaikan dengan tugas pengenalan gestur baru. Pendekatan ini terbukti mempercepat konvergensi dan meningkatkan akurasi meski menggunakan dataset yang lebih sedikit (Asriani dkk., 2025).

Meskipun potensi *ResNet-50* dan *Transfer Learning* sangat menjanjikan, penerapannya pada sistem navigasi robot beroda dengan sudut pandang ketiga (*third-person view*) masih perlu dieksplorasi lebih lanjut. Mayoritas penelitian sebelumnya masih terpaku pada sensor fisik atau sudut pandang orang pertama (*first-person*) yang kurang praktis. Oleh karena itu, penelitian ini berfokus pada implementasi metode *Transfer Learning* menggunakan arsitektur ResNet-50 untuk membangun sistem kendali robot beroda yang mampu mengenali gestur tangan secara *real-time* dan menerjemahkannya menjadi perintah gerak (Maju, Mundur, Kiri, Kanan, Stop) tanpa alat tambahan pada tubuh pengguna.

1.2 Pernyataan Masalah

Berdasarkan latar belakang yang telah diuraikan, rumusan masalah dalam penelitian ini adalah bagaimana merancang bangun sistem kendali robot beroda berbasis visi komputer menggunakan metode *Transfer Learning* pada arsitektur ResNet-50?

1.3 Tujuan Penelitian

Sejalan dengan rumusan masalah tersebut, tujuan penelitian ini adalah untuk merancang, membangun, serta menguji performa sistem kendali robot beroda berbasis gestur tangan menggunakan metode *Transfer Learning* pada arsitektur ResNet-50.

1.4 Batasan Masalah

a. Model yang digunakan adalah ResNet-50 dengan strategi *Transfer Learning* (*Selective Fine-Tuning*), di mana bobot awal diambil dari dataset ImageNet dan hanya lapisan tengah-atas dan kepala klasifikasi baru dilatih ulang menggunakan konfigurasi *Differential Learning Rate*.
b. *Dataset* yang digunakan adalah bagian dari *dataset* publik HaGRID (*Hand Gesture Recognition Image Dataset*) yang telah disaring menjadi 5 kelas gestur navigasi, yaitu Maju, Mundur, Kiri, Kanan, dan Stop.
c. Sistem mendeteksi gerakan tangan pengguna menggunakan pi *camera* biasa dengan sudut pandang ketiga, di mana kamera menghadap ke arah pengguna.
d. Robot yang dibuat adalah versi awal (*prototype*) dari robot yang menggunakan roda dan dikendalikan oleh mikrokontroler RaspberryPi 4 Model B.
e. Pengujian sistem dilakukan di dalam ruangan (*indoor*) dengan kondisi pencahayaan yang cukup dan latar belakang yang relatif statis.

1.5 Manfaat Penelitian

Penelitian ini diharapkan dapat memberikan manfaat teoretis dan praktis. Secara teoretis, penelitian ini diharapkan dapat memperkaya *khazanah* keilmuan di bidang Informatika dan Robotika, khususnya mengenai penerapan arsitektur *Residual Network* (ResNet) dan efektivitas teknik *Transfer Learning* dalam mengatasi keterbatasan dataset pada kasus interaksi manusia-robot (*Human-Robot Interaction*). Secara praktis, hasil penelitian ini menghasilkan sebuah prototipe sistem kontrol robot yang intuitif dan *contactless*, yang dapat dikembangkan lebih lanjut untuk aplikasi bantuan bagi penyandang disabilitas, kontrol robot di lingkungan steril, dan lain-lain.

BAB II

STUDI PUSTAKA

2.1 Penelitian Terkait

Pengembangan sistem kendali robot yang intuitif dan alami telah menjadi fokus utama dalam penelitian interaksi manusia-robot (*Human-Robot Interaction*). Berbagai metode telah dikembangkan untuk menggantikan kendali konvensional (seperti *joystick*), mulai dari penggunaan sensor fisik yang dipasang pada tubuh (*wearable sensors*) hingga pengolahan citra (*computer vision*).

2.1.1 Kendali Robot Berbasis Sensor Fisik (*Wearable Sensors*)

Pendekatan berbasis sensor fisik mengharuskan pengguna mengenakan perangkat keras tertentu untuk menerjemahkan gerakan tubuh menjadi perintah robot. Beberapa penelitian signifikan dalam kategori ini.

Penelitian oleh Cahyadi et al (2023) merancang sistem kendali untuk *Wireless Mobile Robot Arm* (WMRA) yang terdiri dari robot lengan 2 DoF (*Degrees of Freedom*) dan robot mobil non-holonomic 4WD. Sistem kendali utama menggunakan dua buah sensor *accelerometer* yang dipasang pada lengan dan punggung tangan operator untuk mendeteksi kemiringan, serta sensor *Hall-effect* untuk mengontrol *gripper*. Data dari sensor diolah menggunakan mikrokontroler NodeMCU ESP8266 dan dikirimkan melalui jaringan internet (IoT) menggunakan platform Blynk. Untuk memastikan pergerakan robot yang halus dan responsif terhadap gestur lengan operator, penelitian ini menerapkan metode logika *Fuzzy Sugeno*. Hasil pengujian

menunjukkan bahwa robot dapat dikendalikan secara nirkabel dengan tingkat keberhasilan 100% dan waktu tunda (*delay*) kurang dari 100 ms.

Sejalan dengan konsep tersebut, Roziqin et al (2021) mengembangkan robot khusus untuk mengambil bola tenis lapangan yang dikendalikan melalui gestur tangan. Sistem ini memanfaatkan sensor *accelerometer* MPU6050 sebagai input data kemiringan tangan (sumbu x dan y) yang diproses oleh mikrokontroler ATmega16. Data perintah dikirimkan menggunakan modul komunikasi radio frekuensi NRF24L01. Keunggulan dari penelitian ini adalah penerapan metode kontrol PID (*Proportional-Integral-Derivative*) pada sisi penerima (*receiver*) untuk menyingkronkan putaran motor DC. Penggunaan PID terbukti mampu membuat pergerakan robot saat bermanuver atau berbelok menjadi lebih stabil dan halus, meskipun performanya sedikit menurun ketika robot membawa beban.

Pengembangan terbaru dalam kategori ini dilakukan oleh Arif et al (2025) yang mengimplementasikan metode kecerdasan buatan *Neural Network Backpropagation* pada sebuah sarung tangan pintar (*smart glove*). Perangkat ini mengintegrasikan sensor MEMS (*Micro-Electro-Mechanical Systems*) berupa *accelerometer* dan *gyroscope* untuk merekam data pergerakan tangan. Algoritma jaringan saraf tiruan dilatih untuk mengklasifikasikan data sensor menjadi lima perintah gerak: diam, maju, mundur, belok kiri, dan belok kanan. Sistem ini diimplementasikan pada mikrokontroler STM32F10C dengan modul nirkabel NRF24L01. Berdasarkan pengujian yang dilakukan, metode ini menghasilkan tingkat akurasi rata-rata sebesar 82,8% dalam mengenali gestur,

membuktikan bahwa integrasi *Deep Learning* pada data sensor fisik mampu memberikan respons yang cukup akurat.

Meskipun metode berbasis sensor fisik ini terbukti responsif dan akurat, kelemahan mendasarnya adalah sifatnya yang *intrusive*. Pengguna tidak memiliki keleluasaan penuh karena harus mengenakan perangkat tambahan, bergantung pada catu daya baterai pada alat pengendali, serta terbatasnya kenyamanan dalam penggunaan jangka panjang.

2.1.2 Kendali Robot Berbasis Visi Komputer (*Computer Vision*)

Sebagai alternatif dari perangkat *wearable*, pendekatan berbasis visi komputer menawarkan interaksi yang *contactless* (tanpa sentuhan), di mana kamera digunakan sebagai sensor utama untuk menangkap gestur. Ramadhani (2024) meneliti pengembangan sistem kontrol robot mobil menggunakan deteksi gestur jari tangan berbasis *framework* MediaPipe. Penelitian ini menggunakan kamera laptop sebagai input visual untuk mendeteksi *landmark* tangan secara *real-time*. Sistem dirancang untuk mengenali jumlah jari yang terangkat (2, 3, 4, 5 jari) dan menerjemahkannya menjadi perintah gerak yang dikirimkan ke mikrokontroler ESP32 pada robot melalui komunikasi WiFi (protokol UDP). Hasil eksperimen menunjukkan bahwa sistem ini mampu mencapai akurasi deteksi hingga 100% pada jarak optimal 30 cm dengan kecepatan rata-rata 14 FPS (*Frames Per Second*). Penelitian ini berhasil membuktikan bahwa pengendalian robot yang stabil dapat dicapai tanpa perangkat fisik yang melekat pada tubuh pengguna. Namun, penggunaan *framework* instan seperti MediaPipe memiliki keterbatasan dalam hal

kustomisasi arsitektur jaringan yang mendalam jika dibandingkan dengan membangun model CNN dari awal.

Di sisi lain, penerapan arsitektur *Deep Learning* yang lebih kompleks pada robot fisik ditunjukkan oleh Septyanto (2024). Penelitian ini mengimplementasikan algoritma *Convolutional Neural Network* (CNN) dengan arsitektur YOLOv8 untuk pengenalan objek dan VGG19 untuk pengenalan wajah pada sebuah *service robot*. Selain deteksi visual, penelitian ini juga mengembangkan metode estimasi jarak monokular (*monocular distance estimation*) untuk mengukur jarak objek dari robot. Hasil pelatihan menunjukkan tingkat keberhasilan pengenalan objek sebesar 80,7% dan pengenalan wajah sebesar 80%, dengan rata-rata kesalahan estimasi jarak yang sangat rendah (1,525 cm untuk objek). Meskipun fokus penelitian ini adalah pada fungsi pelayanan robot (*service robot*) dan bukan kontrol navigasi aktif via gestur, penelitian ini memvalidasi bahwa arsitektur CNN modern yang berat (seperti YOLO dan VGG) sangat layak dan mampu diimplementasikan pada sistem komputasi robot untuk pemrosesan visual secara *real-time*.

Penelitian terbaru yang dilakukan oleh (Rahmawati dkk., 2025) berfokus pada diagnosis penyakit pneumonia pada balita menggunakan citra radiografi *thorax*. Penelitian ini mengusulkan penggunaan arsitektur *Deep Learning* ResNet-50, yang merupakan varian dari *Convolutional Neural Networks* (CNN) dengan mekanisme *skip connections* untuk mengatasi masalah *vanishing gradient*. Dataset yang digunakan bersumber dari repositori publik Kaggle (*Chest X-Ray Images Pneumonia*) yang mencakup data pasien

balita dengan label normal dan pneumonia. Dalam metodologinya, penelitian ini menerapkan teknik augmentasi data (seperti rotasi, pergeseran, dan *zooming*) serta menggunakan *optimizer* Adam untuk mempercepat konvergensi model. Evaluasi kinerja model dilakukan menggunakan *confusion matrix*, kurva akurasi, dan *loss*. Hasil penelitian menunjukkan bahwa model ResNet-50 mampu mencapai tingkat akurasi sebesar 85% dengan nilai *loss* 0,336 pada data uji. Studi ini menyimpulkan bahwa arsitektur ResNet-50 efektif dalam mengklasifikasikan citra medis dan berpotensi menjadi alat bantu diagnosis yang efisien bagi tenaga medis, meskipun masih disarankan penggunaan teknik regularisasi tambahan untuk peningkatan akurasi lebih lanjut di masa depan.

Dalam konteks navigasi robot berbasis visi, Yu dan Tian (2023) melakukan penelitian komprehensif untuk meningkatkan akurasi dan efisiensi pergerakan robot seluler di lingkungan yang kompleks. Mereka mengusulkan model navigasi baru bernama "Neo" yang mengintegrasikan mekanisme *Split Attention* dengan arsitektur ResNeSt50 (varian pengembangan dari ResNet-50). Pendekatan ini dipilih untuk meningkatkan kemampuan ekstraksi fitur visual yang sering kali gagal dilakukan oleh algoritma navigasi konvensional. Melalui pengujian simulasi pada dataset AI2-THOR, model yang dikembangkan berhasil mencapai akurasi navigasi rata-rata sebesar 92,3% dan menurunkan tingkat tabrakan (*collision rate*) secara signifikan hingga 0,01. Penelitian ini membuktikan bahwa arsitektur berbasis ResNet sangat handal

untuk tugas persepsi visual pada robot, terutama ketika dikombinasikan dengan mekanisme atensi untuk fokus pada fitur lingkungan yang krusial.

Berdasarkan tinjauan terhadap berbagai penelitian terdahulu, terdapat beberapa perbedaan mendasar secara metodologis antara sistem yang diusulkan dalam penelitian ini dibandingkan dengan metode-metode sebelumnya. Perbedaan metode tersebut dapat dilihat melalui pendekatan ekstraksi data dibandingkan dengan penggunaan sensor fisik. Metode kendali pada penelitian Cahyadi et al. (2023), Roziqin et al. (2021), dan Arif et al. (2025) sangat bergantung pada data deret waktu dari sensor kemiringan fisik seperti akselerometer atau giroskop yang kemudian diproses menggunakan algoritma kendali seperti *Fuzzy*, PID, atau Jaringan Saraf Tiruan dasar. Sebaliknya, penelitian ini menggunakan citra visual dua dimensi sebagai input utama. Penggunaan arsitektur jaringan saraf tiruan konvolusional ResNet-50 memungkinkan ekstraksi fitur spasial seperti bentuk dan lekukan jari secara otomatis tanpa memerlukan perangkat keras tambahan yang membebani fisik pengguna.

Perbedaan selanjutnya terletak pada arsitektur visi komputer yang digunakan dibandingkan dengan metode seperti MediaPipe, VGG, atau YOLO. Penelitian Ramadhani (2024) menggunakan kerangka kerja MediaPipe yang bekerja dengan mendeteksi titik ikat atau *landmark* tangan. Kelemahan metode ini adalah rentan mengalami kegagalan jika kondisi pencahayaan buruk atau sebagian jari tertutup. Sementara itu, metode yang diusulkan dalam penelitian ini menggunakan ResNet-50 yang mempelajari fitur global dan lokal dari citra

secara mendalam hingga lima puluh lapisan konvolusi, menjadikannya lebih tangguh terhadap variasi latar belakang dan pencahayaan. Dibandingkan dengan arsitektur VGG19 atau YOLO yang digunakan oleh Septyanto, (2024), ResNet-50 memiliki keunggulan komputasi berkat mekanisme jalan pintas atau blok residual yang mampu mengatasi masalah degradasi gradien sekaligus menjaga beban komputasi agar tetap efisien untuk inferensi waktu nyata.

Selain itu, terdapat perbedaan dalam domain implementasi arsitektur ResNet dibandingkan dengan analisis citra statis dan simulasi. Walaupun sama-sama menggunakan keluarga arsitektur ResNet, metode yang diusulkan memiliki perbedaan fokus penerapan. Jika Rahmawati et al. (2025) menggunakan ResNet-50 untuk mengklasifikasikan data citra medis yang statis, penelitian ini mengimplementasikan metode *transfer learning* untuk memproses aliran bingkai video yang dinamis dan berkesinambungan. Di sisi lain, Yu & Tian (2022) menerapkan varian ResNet untuk robotika namun berfokus pada persepsi rintangan lingkungan di dalam ruang simulasi. Penelitian ini membedakan diri dengan memfokuskan ekstraksi fitur ResNet pada pengenalan gestur atau intensi manusia dan secara langsung mentransmisikan hasil klasifikasi tersebut sebagai sinyal kendali pada robot mobil fisik berspesifikasi mikrokontroler.

Tabel 2.1 Penelitian Terdahulu

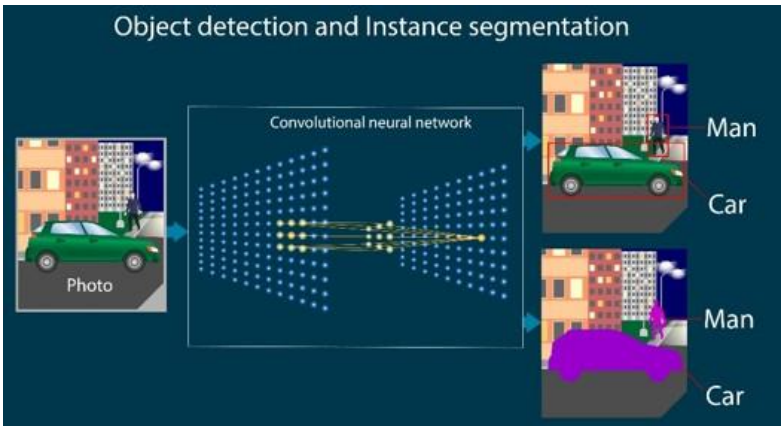
No	Sumber (Tahun)	Judul Penelitian	Metode yang Digunakan	Akurasi / Hasil Penelitian
1	Roziqin et al. (2021)	Rancang Bangun Robot Pengambil Bola Tenis Menggunakan Sensor MPU6050	Sensor MPU6050 (*Gyroscope*/*Accelerometer*) + Kontrol PID	Pergerakan robot stabil dengan *error* sudut kemiringan < 5% berkat kontrol PID.

No	Sumber (Tahun)	Judul Penelitian	Metode yang Digunakan	Akurasi / Hasil Penelitian
2	Cahyadi et al. (2023)	Kendali *Wireless Mobile Robot Arm* Berbasis IoT Menggunakan Logika *Fuzzy*	Sensor *Accelerometer* + Logika *Fuzzy* (IoT)	Respon robot fleksibel terhadap kemiringan tangan dengan latensi komunikasi rata-rata 0.2 detik.
3	Arif et al. (2025)	Pengendalian Gerak Robot Beroda Menggunakan Sarung Tangan Pintar dengan *Neural Network Backpropagation*	*Smart Glove* + NN *Backpropagation* Klasifikasi Gestur Tangan	Akurasi rata-rata 82,8% menggunakan sensor fisik.
4	Ramadhani (2024)	Sistem Kendali Robot Mobil Tanpa Awak Menggunakan *Hand Tracking*	MediaPipe *Framework* (Google)	Akurasi deteksi tangan mencapai ~95% pada pencahayaan cukup dengan jarak efektif 0.5 - 1 meter.
5	Septyanto (2024)	Implementasi *Deep Learning* untuk Pengenalan Wajah dan Objek pada *Service Robot*	*Convolutional Neural Network* (YOLOv8 & VGG19)	Tingkat keberhasilan deteksi objek mencapai mAP (*Mean Average Precision*) 90% secara *real-time*.
6	Rahmawati et al. (2025)	Analisis Performa Model ResNet-50 Pada Diagnosis Pneumonia Balita Berdasarkan Citra Radiografi *Thorax*	*Deep Learning* (CNN) ResNet-50	Mencapai akurasi 85% dan *loss* 0,336. Model terbukti efektif mengatasi masalah *vanishing gradient* dan akurat dalam klasifikasi citra medis.
7	Yu & Tian (2023)	*Deep Learning-Based Visual Navigation Algorithms for Mobile Robots*	Mengembangkan model navigasi visual "Neo" berbasis ResNet50 (varian ResNet dengan *Split-Attention*) untuk robot seluler pada simulasi AI2-THOR.	Mencapai akurasi navigasi rata-rata 92,3%, menurunkan rasio tabrakan robot menjadi 0,01, dan mempercepat waktu tempuh >8 detik dibanding metode lain.
8	Penelitian yang Diusulkan	Penerapan Metode *Transfer Learning* dengan Arsitektur ResNet-50 untuk Navigasi Robot	*Transfer Learning, Convolutional Neural Network* (ResNet-50), Visi Komputer (Citra 2D), Mikrokontroler Raspberry Pi	menggunakan citra visual 2D waktu nyata tanpa sensor fisik tambahan. Penggunaan ResNet-50 dipilih karena lebih

No	Sumber (Tahun)	Judul Penelitian	Metode yang Digunakan	Akurasi / Hasil Penelitian
		Beroda Berbasis Visi Komputer		tangguh terhadap variasi cahaya dan oklusi jari dibandingkan metode *landmark*. Selain itu, hasil klasifikasi langsung diterapkan sebagai sinyal navigasi pada robot beroda fisik

2.2 Convolutional Neural Network

Convolutional Neural Network atau CNN merupakan salah satu algoritma *deep learning* yang dirancang khusus untuk mengolah data berstruktur grid seperti citra dua dimensi (Alzubaidi dkk., 2021). Berbeda dengan jaringan saraf tiruan konvensional yang mengharuskan ekstraksi fitur dilakukan secara manual, CNN memiliki keunggulan berupa kemampuan ekstraksi fitur yang berjalan secara otomatis dan hierarkis. Jaringan ini mampu mengenali pola visual penting mulai dari garis tepi, tekstur, hingga bentuk objek yang kompleks, sehingga menjadikannya sebagai standar utama dalam bidang visi komputer saat ini (Kattenborn dkk., 2021).



Gambar 2.1 Ilustrasi CNN

Secara arsitektur, proses ekstraksi fitur pada CNN digerakkan oleh *Convolutional Layer* sebagai komponen intinya. Pada lapisan ini, sekumpulan filter atau kernel digeser melintasi seluruh area citra untuk melakukan operasi matematika konvolusi yang akan menghasilkan peta fitur (Li et al., 2022). Hasil dari operasi tersebut kemudian diproses menggunakan fungsi aktivasi non-linear seperti *Rectified Linear Unit* atau ReLU. Fungsi aktivasi ini bertugas meredam nilai negatif menjadi nol guna mempercepat proses konvergensi pembelajaran model sekaligus mengadaptasi sifat data citra dunia nyata yang umumnya non-linear (Purwono dkk., 2022).

Untuk mengoptimalkan efisiensi jaringan, CNN memanfaatkan *Pooling Layer* yang disisipkan secara berkala guna mereduksi dimensi spasial dari peta fitur. Pengurangan dimensi ini sangat penting untuk menurunkan beban komputasi dan mencegah terjadinya *overfitting* pada model. Setelah melalui serangkaian proses konvolusi dan reduksi dimensi, peta fitur yang pada awalnya berbentuk matriks dua dimensi akan diratakan menjadi vektor satu dimensi. Vektor ini kemudian diteruskan ke *Fully Connected Layer* yang berfungsi layaknya jaringan saraf tiruan konvensional untuk melakukan penalaran tingkat tinggi dan menghasilkan probabilitas keputusan atau klasifikasi akhir dari sistem.

2.3 Arsitektur *Residual Network* (ResNet-50)

Residual Network (ResNet) adalah arsitektur *Convolutional Neural Network* (CNN) yang diperkenalkan oleh para peneliti di *Microsoft Research* pada tahun 2015. Arsitektur ini dirancang secara spesifik untuk mengatasi kendala utama dalam pelatihan jaringan saraf yang sangat dalam (*deep neural networks*), yaitu

masalah gradien yang menghilang (*vanishing gradient problem*) dan degradasi akurasi.

Dalam perkembangan riset visi komputer modern, ResNet-50 telah menjadi standar industri dan sering digunakan sebagai *baseline* (tolok ukur) untuk mengevaluasi performa klasifikasi citra pada dataset kompleks seperti ImageNet. Klasifikasi menggunakan ResNet-50 tetap menjadi acuan utama untuk mengukur kualitas fitur visual karena kemampuannya yang konsisten dalam menangkap representasi semantik dari citra (Saharia dkk., 2023).

2.3.1 Konsep Residual Learning dan Skip Connection

Inovasi fundamental dari ResNet adalah pengenalan blok residual (*residual block*) dengan mekanisme koneksi pintas (*skip connection* atau *shortcut connection*). Pada CNN konvensional, setiap lapisan (x) memberi makan lapisan berikutnya secara langsung. Namun, pada jaringan yang sangat dalam, sinyal gradien sering kali menjadi sangat kecil saat dipropagasikan balik (*backpropagation*) dari lapisan output ke lapisan input, menyebabkan jaringan berhenti belajar.

ResNet mengatasi hal ini dengan menambahkan input awal x ke output dari tumpukan lapisan konvolusi $F(x)$, sehingga fungsi pemetaan menjadi:

$$H(x) = F(x) + x \tag{2.1}$$

Dimana:
H(x) adalah pemetaan yang diharapkan.
F(x) adalah fungsi residual (perubahan yang dipelajari).
x adalah pemetaan identitas (input asli).

Mekanisme ini memungkinkan aliran informasi dan gradien melewati jaringan tanpa hambatan. (R. Li dkk., 2022) dalam penelitiannya mengenai segmentasi semantik menyoroti bahwa meskipun arsitektur jaringan saraf terus berkembang menjadi lebih kompleks, penggunaan fitur yang diekstraksi melalui mekanisme yang kuat seperti yang ditawarkan oleh *backbone* ResNet sangat krusial untuk menghindari hilangnya informasi spasial penting pada citra resolusi tinggi.

2.3.2 Struktur Arsitektur ResNet-50

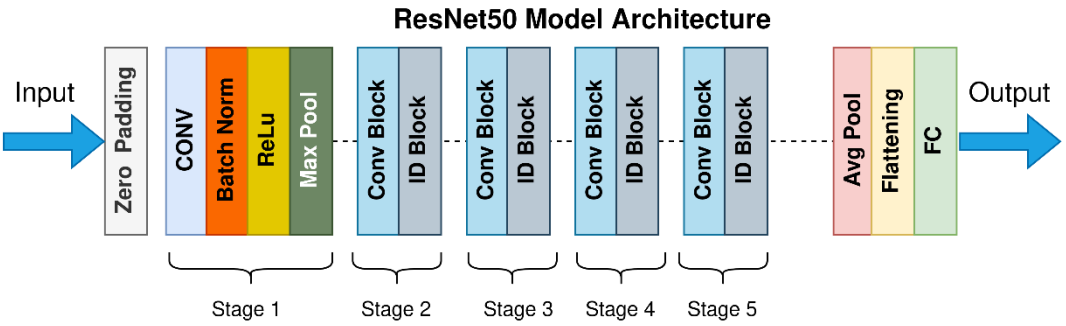
ResNet-50 adalah varian ResNet dengan kedalaman 50 lapisan berbobot. Arsitektur ini tidak menggunakan blok residual standar (dua lapis 3×3), melainkan menggunakan desain *Bottleneck Block*. Setiap blok *bottleneck* terdiri dari tiga lapisan konvolusi berurutan:

1. Konvolusi 1×1: Untuk mereduksi dimensi (kompresi fitur).
2. Konvolusi 3×3: Untuk ekstraksi fitur utama.
3. Konvolusi 1×1: Untuk memulihkan dimensi (ekspansi fitur).

Struktur ini membuat ResNet-50 jauh lebih efisien secara komputasi dibandingkan arsitektur lain dengan kedalaman yang sama. Secara keseluruhan, ResNet-50 terdiri dari 5 tahapan (*stages*):

- Stage 1: Konvolusi 7×7, 64 kernel, *stride* 2, diikuti *Max Pooling*.
- Stage 2: 3 blok residual *bottleneck*.
- Stage 3: 4 blok residual *bottleneck*.
- Stage 4: 6 blok residual *bottleneck*.
- Stage 5: 3 blok residual *bottleneck*.

- Akhir: *Global Average Pooling* dan *Fully Connected Layer* (*Softmax*).



Gambar 2.2 Model Arsitektur ResNet-50

2.3.3 Penerapan dalam penelitian

Kehandalan arsitektur berbasis CNN yang mendalam seperti ResNet telah terbukti di berbagai domain. Kerangka kerja CNN yang dioptimalkan sepenuhnya (*fully optimized framework*) mampu mencapai akurasi sangat tinggi (di atas 98%) dalam tugas klasifikasi multi-kelas yang sensitif, seperti diagnosis tumor otak. Prinsip yang sama diterapkan dalam penelitian ini dengan memanfaatkan arsitektur ResNet-50 yang telah terbukti tangguh (*robust*) (Irmak, 2021). Sistem diharapkan mampu mengenali variasi gestur tangan untuk navigasi robot dengan tingkat akurasi dan stabilitas yang setara dengan performa pada aplikasi medis maupun visi komputer tingkat lanjut lainnya.

2.4 Transfer Learning

Dalam paradigma pembelajaran mesin tradisional, pelatihan model biasanya dilakukan dari awal (*from scratch*), yang menuntut ketersediaan data latih dalam jumlah besar dan asumsi bahwa data latih serta data uji berasal dari distribusi ruang

fitur yang sama. Namun, dalam banyak kasus dunia nyata pengumpulan data gestur tangan yang cukup banyak dan variatif menjadi kendala utama. Untuk mengatasi hal ini, digunakan metode *Transfer Learning*.

2.4.1 Definisi dan Konsep Dasar

Transfer Learning adalah cara untuk meningkatkan hasil pembelajaran di bidang tujuan (D_T) dan pekerjaan tujuan (T_T) dengan menggunakan pengetahuan yang telah dipelajari dari bidang sumber (D_S) dan pekerjaan sumber (T_T) yang berbeda tetapi masih terkait (Ali dkk., 2023).

Dalam bentuk matematika, sebuah domain terdiri dari ruang fitur $\mathcal{X}$ dan distribusi probabilitas marginal $P(X)$, sedangkan tugas terdiri dari ruang label $P(X)$ dan fungsi prediktif $f(\cdot)$. *Transfer learning* bertujuan untuk meningkatkan kemampuan $f(\cdot)$ dalam memprediksi data di D_T meskipun data sumber D_S berbeda dari data target D_T atau tugas sumber T_S berbeda dengan tugas target T_S.

Penggunaan *Pre-Trained Models* (PTM) telah menjadi era baru dalam kecerdasan buatan. Model yang telah dilatih pada dataset berskala masif (seperti ImageNet yang memiliki jutaan gambar) telah belajar menangkap fitur-fitur universal yang kaya. Pengetahuan ini disimpan dalam parameter model (bobot) dan dapat ditransfer untuk menyelesaikan tugas hilir (*downstream tasks*) yang lebih spesifik dengan data yang terbatas (Han dkk., 2021).

2.4.2 Strategi *Transfer Learning* pada Arsitektur ResNet-50

Dalam penerapan *Transfer Learning* pada *Deep Learning* terdapat dua strategi utama, khususnya arsitektur ResNet-50 yaitu:

1. ***Feature Extraction*** **(Ekstraksi Fitur)**

   Pada strategi ini, lapisan konvolusi (*convolutional base*) dari model yang telah dilatih sebelumnya (misalnya ResNet-50) "dibekukan" (*frozen*). Artinya, bobot pada lapisan ini tidak diperbarui selama proses pelatihan ulang. Model hanya melatih klasifier baru (*Fully Connected Layer*) yang ditambahkan di bagian akhir jaringan. Strategi ini sangat efektif jika dataset target berukuran kecil dan memiliki kemiripan visual dengan dataset sumber (Alzubaidi dkk., 2021).

2. ***Fine-Tuning*** **(Penyesuaian Halus)**

   Strategi ini melibatkan pelatihan ulang tidak hanya pada lapisan akhir, tetapi juga pada sebagian atau seluruh lapisan konvolusi model *pre-trained*. Teknik ini memungkinkan model untuk menyesuaikan fitur-fitur yang lebih spesifik dengan domain target, namun membutuhkan data yang sedikit lebih banyak dan berisiko mengalami *overfitting* jika tidak dikontrol dengan baik (Ali dkk., 2023).

   Salah satu varian yang lebih terkontrol adalah *Selective Fine-Tuning*, yaitu pendekatan di mana hanya sebagian layer yang dilepas pembekuannya secara selektif berdasarkan tingkat keumuman fitur yang dipelajari. Layer awal yang mempelajari fitur universal tingkat rendah (seperti tepi dan

tekstur) tetap dibekukan, sedangkan layer tengah-atas yang bertanggung jawab atas fitur tingkat tinggi yang lebih domain-spesifik dilatih ulang. Untuk mengoptimalkan proses adaptasi ini, diterapkan konfigurasi *Differential Learning Rate*, yaitu pemberian nilai *learning rate* yang berbeda-beda untuk setiap kelompok layer sesuai tingkat adaptasi yang dibutuhkan, dan lebih tinggi untuk layer yang mendekati output.

2.4.3 Penerapan dalam Penelitian

Penelitian ini mengadopsi pendekatan *Inductive Transfer Learning* dengan strategi *Fine-Tuning* Selektif *(Selective Fine-Tuning)* (Ali dkk., 2023). Arsitektur ResNet-50 yang telah dilatih pada dataset ImageNet digunakan sebagai *backbone*. Lapisan awal dibekukan sepenuhnya karena telah mempelajari fitur universal tingkat rendah seperti tepi, sudut, dan tekstur yang berlaku lintas domain. Sementara itu, lapisan tengah-atas dilepas pembekuannya untuk dilatih ulang agar dapat mengekstraksi fitur yang lebih spesifik terhadap domain gestur tangan. *Fully Connected Layer* asli diganti dengan kepala klasifikasi baru yang dirancang untuk mengklasifikasikan 5 kelas gestur navigasi robot.

Proses pelatihan menggunakan konfigurasi *Differential Learning Rate* dengan nilai *learning rate* yang berbeda untuk setiap kelompok *layer*, guna menjaga keseimbangan antara retensi pengetahuan *pretrained* dan adaptasi terhadap domain baru. Pendekatan ini terbukti mempercepat konvergensi dan menghasilkan akurasi tinggi meskipun dataset gestur tangan yang tersedia terbatas.

2.5 Visi Komputer Sudut Pandang Ketiga (*Third-Person Vision*)

Dalam domain visi komputer (*Computer Vision*) dan robotika, konfigurasi penempatan kamera memegang peranan vital dalam menentukan algoritma persepsi yang digunakan. Secara umum, terdapat dua paradigma utama sudut pandang, yaitu Visi Egosentris (*First-Person Vision*) dan Visi Sudut Pandang Ketiga (*Third-Person Vision*).

Penelitian ini mengadopsi pendekatan Visi Sudut Pandang Ketiga, yaitu konfigurasi di mana kamera ditempatkan pada posisi statis atau pada platform robot yang mengarah ke pengguna (*user*). Dalam perspektif ini, kamera bertindak sebagai pengamat eksternal yang merekam gestur pengguna dari jarak tertentu.

2.5.1 Karakteristik dan Perbandingan

Pandangan sudut pandang ketiga (*third-person view*) memberikan keuntungan berupa kerangka acuan yang stabil (*stable frame of reference*) dan pemahaman konteks global yang lebih baik dibandingkan pandangan egosentris. Pada pandangan egosentris, kamera sering kali mengalami guncangan drastis yang mempersulit proses pembelajaran model, sedangkan sudut pandang ketiga cenderung memberikan citra yang lebih konsisten untuk navigasi, meskipun memiliki tantangan dalam hal presisi manipulasi halus (*fine-grained manipulation*) (Jangir dkk., 2022).

2.5.2 Tantangan Teknis Oklusi dan Perspektif

Meskipun menawarkan antarmuka yang lebih alami tanpa perangkat (*device-free*), sudut pandang ini memiliki tantangan teknis signifikan yang harus diatasi, terutama terkait dengan oklusi (penghalang).

Masalah utama yang sering terjadi dalam pengenalan objek manusia (*Person Re-identification*) adalah oklusi, baik yang disebabkan oleh objek lain maupun oleh bagian tubuh itu sendiri (*self-occlusion*) (Wang dkk., 2023). Dalam konteks kendali gestur tangan:

1. *Self-Occlusion* tangan pengguna mungkin tertutup oleh lengan atau badan ketika melakukan gerakan tertentu.
2. Variasi Skala ukuran tangan akan berubah drastis tergantung jarak pengguna dari robot.

Oleh karena itu, diperlukan model *Deep Learning* yang memiliki fitur *quality-aware* atau kemampuan ekstraksi fitur yang kuat seperti ResNet-50 untuk tetap dapat mengenali pola tangan meskipun sebagian tertutup atau berada pada jarak yang bervariasi.

Penerapan visi sudut pandang ketiga ini mendukung konsep interaksi alamiah (*Natural User Interface*). Pengguna tidak perlu mengenakan perangkat sensor apa pun (*wearable sensors*), sehingga memberikan kebebasan gerak penuh. Hal ini sesuai dengan tujuan sistem untuk menciptakan metode kendali yang praktis dan ergonomis.

2.6 Raspberry Pi

Pada arsitektur sistem navigasi robot otonom berbasis visi komputer (*computer vision*), Raspberry Pi 4 model B berfungsi sebagai elemen sentral yang memegang kendali penuh atas sistem. Perangkat ini berperan sebagai *main processing unit* (unit pemrosesan utama) yang mengadopsi konsep komputasi lokal (*edge computing*). Secara operasional, Raspberry Pi bertugas untuk menerima input visual dari kamera, melakukan tahap pra-pemrosesan citra, dan mengambil keputusan berdasarkan hasil analisis data tersebut. Kemudian, Raspberry Pi akan menerjemahkan keputusan tersebut menjadi instruksi dalam bentuk sinyal kendali yang diteruskan ke motor driver, sehingga arah dan kecepatan pergerakan robot dapat diatur secara *real-time* sesuai dengan perintah yang diberikan oleh sistem.

Gambar 2.3 Raspberry Pi

Salah satu keunggulan utama dari Raspberry Pi adalah kemampuannya dalam menangani pemrosesan citra tingkat lanjut secara *real-time* dengan konsumsi daya yang relatif rendah. Hal ini merupakan faktor krusial dalam aplikasi robotik portabel, di mana efisiensi daya baterai menjadi pertimbangan utama. Alih-alih menggunakan metode pemrosesan citra konvensional, perangkat ini terbukti mampu mengeksekusi inferensi algoritma *Deep Learning*, khususnya arsitektur

Convolutional Neural Network (CNN) yang telah dioptimasi ke dalam format ONNX (*Open Neural Network Exchange*). Proses inferensi ini dapat berjalan dengan efisien dan mempertahankan frekuensi frame (*frame rate*) yang memadai tanpa membebani komputasi secara berlebihan (At-taufieqy, 2025).

Selain efisiensi daya dan kemampuan komputasi, Raspberry Pi dikenal karena tingkat fleksibilitas dan skalabilitasnya yang tinggi. Dukungan terhadap pin *General Purpose Input/Output* (GPIO) memfasilitasi koneksi langsung dengan berbagai jenis sensor, aktuator, hingga modul komunikasi nirkabel. Kompatibilitas perangkat keras ini, yang dipadukan dengan lingkungan sistem operasi berbasis Linux, memungkinkan pengguna untuk dengan mudah menyesuaikan serta memperluas fungsionalitas sistem sesuai dengan kebutuhan spesifik proyek yang sedang dikembangkan (Yaldaie dkk., 2024).

Dengan kombinasi antara kapasitas pemrosesan yang mumpuni untuk komputasi *edge*, harga yang terjangkau, serta dukungan komunitas pengembang yang luas, Raspberry Pi menjadi solusi ideal untuk proyek-proyek robotika terintegrasi. Oleh karena itu, perangkat ini sering dijadikan pilihan utama dalam pengembangan sistem robotik skala kecil hingga menengah, khususnya pada sistem yang menuntut respons cepat, ketepatan navigasi otonom, serta kemampuan adaptasi visual terhadap lingkungan sekitar.

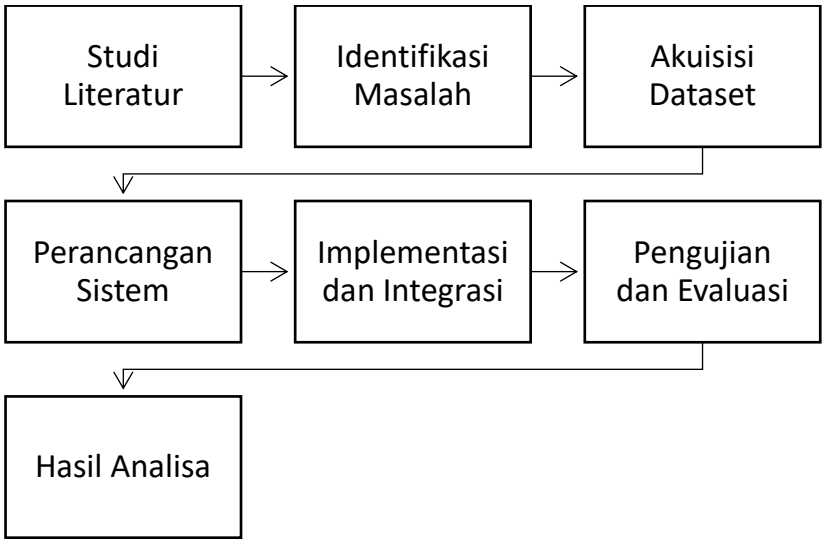
BAB III

DESAIN DAN IMPLEMENTASI SISTEM

3.1 Desain Penelitian

Desain penelitian ini dirancang menggunakan pendekatan eksperimental untuk membangun sistem kendali robot beroda berbasis visi komputer. Tahapan penelitian disusun secara sistematis dan terstruktur untuk memastikan setiap proses mulai dari analisis teoritis hingga implementasi teknis berjalan sesuai dengan tujuan yang telah ditetapkan.

Secara garis besar, alur penelitian ini terdiri dari beberapa fase utama diantaranya fase analisis, fase perancangan (*design*), fase implementasi, dan fase pengujian (*testing*). Tahapan penelitian digambarkan dalam diagram pada Gambar 3.1.



Gambar 3.1 Diagram Tahapan Penelitian

Penjelasan rinci mengenai langkah-langkah penelitian adalah sebagai berikut:

1. Studi Literatur

Langkah awal penelitian dimulai dengan penguatan fondasi teoritis melalui studi literatur. Pada tahap ini, penulis melakukan eksplorasi mendalam terhadap berbagai sumber referensi primer seperti buku teks, jurnal ilmiah bereputasi, serta artikel penelitian terkini. Fokus kajian diarahkan pada pemahaman konsep fundamental kecerdasan buatan, khususnya *Deep Learning* dan algoritma *Convolutional Neural Network* (CNN) yang menjadi tulang punggung sistem visi komputer.

Secara lebih spesifik, kajian literatur juga membedah karakteristik arsitektur *Residual Network* (ResNet-50). Analisis difokuskan pada mekanisme *skip connection* yang menjadi solusi atas masalah degradasi gradien pada jaringan dalam, serta penerapan metode *Transfer Learning* untuk efisiensi pelatihan model. Selain aspek perangkat lunak, tinjauan pustaka juga mencakup prinsip kerja mikrokontroler dan teori kinematika robot beroda tipe *differential drive* sebagai landasan pergerakan fisik robot.

2. Identifikasi Masalah

Berangkat dari pemahaman teoritis dan observasi lapangan, penulis mengidentifikasi adanya kesenjangan teknologi pada metode interaksi manusia-robot saat ini. Permasalahan utama yang dirumuskan adalah keterbatasan metode kendali konvensional (seperti *joystick* atau tombol fisik) yang dinilai kurang praktis dan kaku. Oleh karena itu, muncul kebutuhan mendesak untuk mengembangkan antarmuka kendali alami

(*Natural User Interface*) berbasis gestur tangan yang mampu bekerja secara nirkontak, responsif, dan akurat.

3. Analisis Kebutuhan

Untuk menjawab permasalahan tersebut, dilakukan analisis kebutuhan sistem secara menyeluruh guna menentukan spesifikasi teknis yang ideal. Kebutuhan ini mencakup aspek perangkat keras dan perangkat lunak yang harus tersedia. Dari sisi perangkat keras, spesifikasi yang dipetakan meliputi unit komputasi laptop (CPU dan GPU) yang mumpuni untuk pelatihan model, pi *camera* sebagai sensor visual, serta komponen elektronika robot seperti mikrokontroler Raspberry Pi, *driver* motor, dan sasis mekanik. Sementara itu, kebutuhan perangkat lunak meliputi pemilihan bahasa pemrograman Python dan C++, kerangka kerja *Deep Learning* (PyTorch/TensorFlow), serta pustaka pendukung seperti OpenCV dan NumPy.

4. Pengumpulan Data (*Data Collection*)

Tahap selanjutnya adalah akuisisi data yang menjadi bahan bakar utama bagi model kecerdasan buatan. Penulis memanfaatkan data sekunder dari dataset publik HaGRID (*Hand Gesture Recognition Image Dataset*). Mengingat dataset aslinya sangat besar dan beragam, dilakukan proses penyaringan (*filtering*) untuk mengambil hanya data yang relevan. Citra yang dipilih dibatasi pada 5 kelas gestur navigasi utama yaitu Maju, Mundur, Kiri, Kanan, dan Stop.

5. Perancangan Sistem (*System Design*)

Memasuki fase perancangan, proses dibagi menjadi dua alur kerja yang berjalan secara paralel namun saling terintegrasi. Alur pertama adalah perancangan perangkat lunak, yang mencakup desain mekanisme pra-pemrosesan citra, modifikasi lapisan akhir (*Fully Connected Layer*) pada arsitektur ResNet-50 agar sesuai dengan target klasifikasi, serta penyusunan algoritma logika kendali untuk menerjemahkan prediksi menjadi aksi.

Di sisi lain, alur kedua berfokus pada perancangan perangkat keras. Penulis merancang skema elektronika (*wiring diagram*) untuk menghubungkan mikrokontroler dengan komponen sensor dan aktuator. Desain ini juga mencakup perancangan tata letak komponen pada sasis robot untuk menjamin keseimbangan dan kestabilan pergerakan saat bermanuver.

6. Implementasi dan Integrasi

Rancangan yang telah matang kemudian dieksekusi pada tahap implementasi. Proses ini diawali dengan melatih model ResNet-50 menggunakan data yang telah disiapkan hingga mencapai tingkat akurasi yang diharapkan. Bersamaan dengan itu, perakitan fisik robot dilakukan sesuai desain mekanik. Puncak dari tahap ini adalah proses integrasi sistem, di mana model AI yang berjalan otomatis pada mikrokontroler robot yang memungkinkan terjadinya transfer data perintah secara *real-time*.

7. Pengujian dan Evaluasi

Validasi sistem dilakukan melalui tiga skenario pengujian yang ketat: pengukuran akurasi model menggunakan confusion matrix dan metrik turunannya (*precision*, *recall*, F1-*score*, ROC-AUC), pengukuran waktu eksekusi sistem *end-to-end*, serta uji fungsionalitas manuver robot di lingkungan nyata dengan variasi jarak, sudut, dan pencahayaan. Apabila kinerja belum memenuhi standar, proses dikembalikan ke tahap perancangan untuk perbaikan.

8. Penarikan Kesimpulan

Rangkaian penelitian ditutup dengan penarikan kesimpulan. Berdasarkan data hasil pengujian dan analisis yang telah dilakukan, penulis menyimpulkan keberhasilan sistem dalam menjawab rumusan masalah, sekaligus memberikan saran-saran konstruktif untuk pengembangan penelitian serupa di masa mendatang.

3.2 Akuisisi Data Visual

Tahapan akuisisi data visual memegang peranan fundamental dalam arsitektur sistem pengenalan berbasis visi komputer, mengingat kualitas citra yang diperoleh pada fase ini berbanding lurus dengan akurasi klasifikasi pada proses selanjutnya. Dalam kerangka penelitian ini, metode pendekatan yang diterapkan adalah sudut pandang orang ketiga (*Third-Person View*). Pemilihan konfigurasi ini didasarkan pada kebutuhan untuk menciptakan interaksi manusia-komputer yang bersifat alami dan nirkontak (*contactless*). Dengan demikian, pengguna tidak

dibebani oleh kewajiban mengenakan perangkat sensor tambahan pada tubuh (*device-free*), yang secara signifikan meningkatkan kenyamanan dan fleksibilitas penggunaan sistem.

Ditinjau dari aspek perangkat keras, instrumen utama yang digunakan untuk menangkap informasi visual adalah pi *camera* yang terhubung langsung ke unit pemrosesan utama (Raspberry Pi). Spesifikasi teknis kamera ini mencakup resolusi definisi tinggi (HD) minimal 1280×720 piksel dengan laju bingkai (*frame rate*) sebesar 30 fps. Pemilihan spesifikasi ini bertujuan untuk menjamin detail citra yang memadai serta kelancaran aliran data visual yang diperlukan untuk analisis gerakan secara *real-time*.

Konfigurasi spasial pengujian diatur untuk memastikan cakupan visual yang optimal terhadap objek target. Kamera ditempatkan pada posisi statis dengan ketinggian yang sejajar dengan dada atau bahu pengguna, yakni pada rentang 1 hingga 1,5 meter dari permukaan lantai. Selain itu, jarak efektif antara pengguna dan lensa kamera ditetapkan pada interval 50 cm hingga 200 cm. Pengaturan geometri ini dirancang agar seluruh area tangan dapat terakuisisi secara utuh dalam bingkai (*frame*) kamera, meminimalisasi risiko terpotongnya informasi visual yang krusial.

Pengambilan data dilakukan di dalam ruangan tertutup agar kondisi lingkungan, terutama cahaya, lebih mudah diatur. Ruangan yang digunakan memiliki tingkat pencahayaan minimal 300 lux supaya hasil foto tidak terlalu gelap yang bisa memunculkan bintik atau *noise*. Sebenarnya, sistem ini sudah dibuat agar bisa membaca gestur tangan pada latar belakang yang ramai. Namun, untuk

menjaga kualitas dan konsistensi data, latar belakang yang digunakan saat pengambilan gambar tetap diusahakan diam dan sesederhana mungkin (Tri dkk., 2025). Output dari proses ini berupa aliran video (*video stream*) dalam format warna RGB (*Red, Green, Blue*) yang kemudian disalurkan sebagai data mentah (*raw input*) menuju tahapan pra-pemrosesan (*preprocessing*).

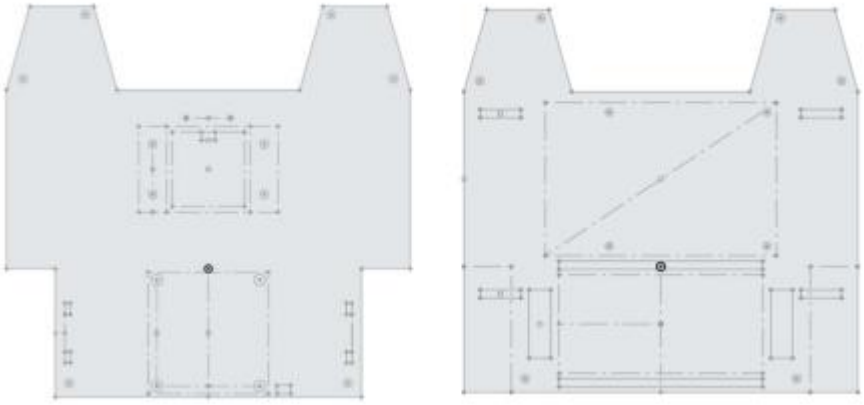
3.3 Desain Perangkat Keras

Tahap desain perangkat keras berfokus pada integrasi komponen fisik robot, yang mencakup rancang bangun mekanik sasis dan skema pengkabelan kelistrikan. Perancangan ini dipersiapkan secara matang guna memastikan perangkat keras tidak hanya mampu bermanuver secara fisik, tetapi juga memiliki kelistrikan yang stabil untuk mengakomodasi beban komputasi dari algoritma *Deep Learning*.

Pengambilan data visual pada robot dikerjakan oleh modul Pi *Camera* Rev 1.3 Model V1 5MP. Kamera ini disambungkan langsung ke Raspberry Pi menggunakan antarmuka CSI (*Camera Serial Interface*). Pemilihan antarmuka CSI sengaja dilakukan untuk menekan waktu tunda (*Execution Time*) pengiriman gambar yang sering terjadi jika menggunakan kamera USB standar. Dengan resolusi 5 megapiksel, kamera ini mampu menangkap bentuk dan pola gestur tangan dengan tajam. Kualitas tangkapan ini sudah lebih dari cukup untuk memenuhi kebutuhan gambar masukan pada arsitektur model AI yang digunakan, sekaligus menjaga beban kerja Raspberry Pi agar tetap ringan.

Desain mekanik pada robot ini menggunakan sasis bertipe *2-Wheel Drive* (2WD) yang mengandalkan dua roda penggerak utama beserta tambahan satu roda bebas (*caster wheel*) untuk menjaga keseimbangan alat. Penempatan komponen

diatur sedemikian rupa guna memastikan titik berat (*center of gravity*) robot tetap stabil saat menerima instruksi pergerakan secara tiba-tiba.



Gambar 3.2 Desain *Chassis*

Pada susunan tata letaknya, Raspberry Pi 4 Model B ditempatkan pada lapisan paling atas sasis sebagai pusat kendali utama. Posisi terbuka ini sengaja dipilih untuk memudahkan sirkulasi udara dan pembuangan panas dari prosesor yang bekerja mengeksekusi model kecerdasan buatan. Selanjutnya, modul kamera visual diposisikan di bagian paling depan sasis dan dihadapkan lurus ke arah pengguna. Penempatan ini krusial untuk mendapatkan sudut pandang visual yang paling optimal saat menangkap gestur tangan. Sementara itu, modul *motor driver* L298N diletakkan di area yang berdekatan dengan letak motor DC. Tujuannya adalah untuk memangkas panjang jalur kabel daya mekanik sehingga resistansi listrik dapat ditekan seminimal mungkin.

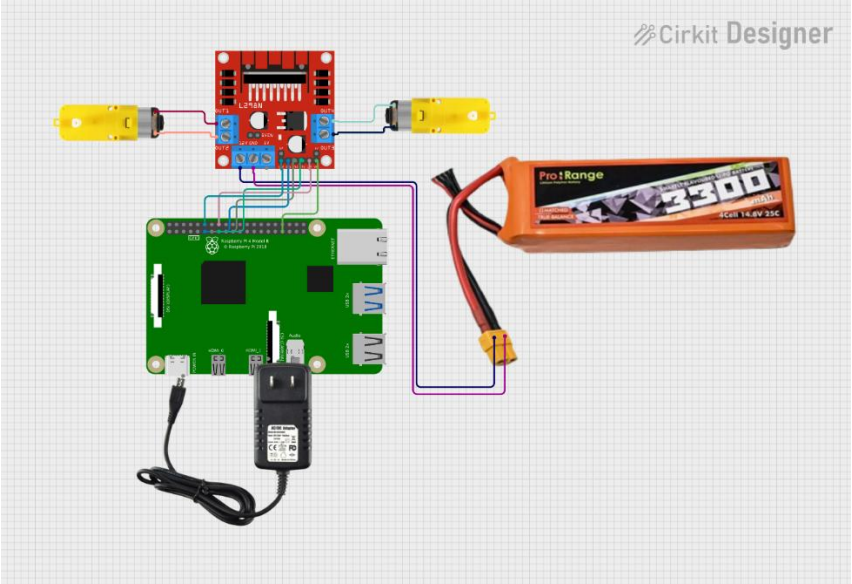
Integrasi kelistrikan antara mikrokontroler pengontrol dan aktuator dirancang melalui jalur komunikasi pin *General Purpose Input/Output* (GPIO). Sinyal instruksi pergerakan dari Raspberry Pi disalurkan langsung menuju pin masukan logika IN1 hingga IN4 pada modul *motor driver* L298N. Pemetaan secara

spesifik antara pin GPIO Raspberry Pi 4 Model B (dalam mode penomoran BCM) dengan pin kendali pada modul L298N yang diimplementasikan dalam penelitian ini disajikan pada Tabel 3.1 Penomoran BCM (*Broadcom* SOC *Channel*) digunakan karena merupakan standar penomoran yang langsung merujuk pada nomor saluran chip *Broadcom* BCM2711 yang digunakan oleh Raspberry Pi 4 Model B.

Tabel 3.1 Pemetaan Pin GPIO Raspberry Pi ke Modul L298N

No.	Pin L298N	GPIO BCM	Fungsi	Keterangan
1	**ENA**	GPIO 18	Kendali kecepatan motor kiri (PWM)	PWM *hardware*
2	**IN1**	GPIO 4	Arah putaran motor kiri sinyal A	Logika digital
3	**IN2**	GPIO 17	Arah putaran motor kiri sinyal B	Logika digital
4	**IN3**	GPIO 27	Arah putaran motor kanan sinyal A	Logika digital
5	**IN4**	GPIO 22	Arah putaran motor kanan sinyal B	Logika digital
6	**ENB**	GPIO 12	Kendali kecepatan motor kanan (PWM)	PWM *hardware*
7	**GND**	GND (Pin 6)	Ground bersama (*common ground*)	Referensi tegangan

Pin ENA (GPIO 18) dan ENB (GPIO 12) dipilih secara khusus karena keduanya merupakan pin yang mendukung fungsi *Pulse Width Modulation* (PWM) secara *hardware* pada Raspberry Pi 4 Model B. Penggunaan PWM *hardware* menghasilkan sinyal yang lebih presisi dan stabil dibandingkan PWM berbasis *software*, sehingga kecepatan putaran kedua motor DC dapat dikendalikan dengan akurat dan konsisten. Penyatuan jalur GND antara Raspberry Pi dan modul L298N bersifat wajib untuk memastikan seluruh sinyal logika digital yang dikirimkan melalui pin IN1 hingga IN4 memiliki titik referensi tegangan yang sama, sehingga dapat terbaca dan dieksekusi dengan benar oleh *motor driver.*



Gambar 3.3 Rangkaian Komponen Sirkuit

Pada implementasi perangkat kerasnya, rancangan kelistrikan ini tidak menggunakan sistem catu daya tunggal, melainkan menerapkan arsitektur catu daya ganda (*dual power supply*) yang terpisah. Modifikasi ini sangat diperlukan mengingat Raspberry Pi 4 dan pi *camera* membutuhkan arus listrik minimal 2.5 Ampere secara stabil untuk memproses inferensi model RestNet 50 berbasis ONNX secara terus-menerus. Oleh karena itu, Raspberry Pi ditenagai secara mandiri menggunakan *powerbank* berkapasitas daya keluaran yang memadai, sedangkan kedua motor DC ditenagai oleh susunan baterai terpisah yang dihubungkan ke pin tegangan 12V pada modul L298N. Pemisahan sumber kelistrikan ini bertujuan untuk mencegah terjadinya penurunan tegangan listrik secara tiba-tiba (*voltage drop*) saat motor aktuator mulai berputar menarik beban, yang umumnya menyebabkan Raspberry Pi mengalami *restart* paksa. Meskipun sumber listriknya dipisahkan secara total, jalur kabel *Ground* (GND) antara Raspberry Pi dan L298N tetap dihubungkan menjadi satu. Penyatuan kutub negatif ini wajib dilakukan agar

sinyal logika digital yang dikirimkan oleh Raspberry Pi memiliki titik referensi tegangan listrik yang sama, sehingga seluruh perintah navigasi dapat terbaca dan dieksekusi dengan sempurna oleh *motor driver*.

3.4 Pengumpulan dan *Pre-processing* Data

Kualitas *dataset* memegang peranan vital dalam menentukan performa dan generalisasi model *Deep Learning*. Dalam penelitian ini, sumber data yang digunakan adalah data sekunder yang diperoleh dari repositori publik Kaggle, secara spesifik menggunakan *Hand Gesture Recognition Image Dataset* (HaGRID) versi sampel. Pemilihan dataset ini didasarkan pada karakteristik citranya yang memiliki variabilitas tinggi dari segi pencahayaan dan latar belakang, baik *indoor* maupun *outdoor*. Keragaman ini dianggap sangat representatif terhadap kondisi lingkungan nyata (*in the wild*), sehingga ideal untuk menguji ketahanan sistem navigasi robot terhadap *noise* visual.

Secara spesifik, *dataset* yang digunakan merupakan subset dari repositori innominate817/hagrid-sample-30k-384p. Citra dalam *dataset* ini memiliki resolusi standar 384×384 piksel dalam format JPEG dan ruang warna RGB. Dimensi resolusi tersebut dipilih karena menawarkan keseimbangan optimal antara preservasi detail fitur tangan yang krusial untuk ekstraksi fitur, dengan efisiensi beban komputasi yang dapat ditangani oleh perangkat keras. Proses akuisisi citra asli melibatkan berbagai subjek dengan rentang jarak pengambilan yang bervariasi antara 0,5 hingga 4 meter, memberikan cakupan skala objek yang luas bagi model.

Dalam penelitian ini, hanya lima jenis gestur dari *dataset* awal yang dipilih dan disaring untuk mengendalikan arah pergerakan robot. Kelima gestur ini

kemudian diterjemahkan menjadi perintah dasar, yakni kepalan tangan (*Fist*) untuk instruksi maju, acungan jempol (*Thumb Up*) untuk mundur, isyarat dua jari (*Peace*) untuk belok kiri, bentuk *Ok* untuk belok kanan, dan telapak tangan terbuka (*Palm*) untuk berhenti. Dari proses penyaringan tersebut, terkumpul sebanyak 8.756 gambar. Agar model tidak berat sebelah saat proses belajar, jumlah data disamakan rata (*balanced*), di mana setiap gestur diwakili oleh sekitar 1.700 gambar. Contoh visual masing-masing gestur beserta rincian jumlah pastinya dapat dilihat pada Tabel 3.2.

Tabel 3.2 *Dataset*

No	Kelas Gestur	Representasi Visual	Perintah Robot	Jumlah Sampel
1.	*Fist* (Kepalan)	Tangan mengepal	Maju (*Forward*)	1.735
2.	*Like*	Jempol ke atas	Mundur (*Backward*)	1.732
3.	*Peace*	Dua jari (V)	Belok Kiri (*Turn Left*)	1.769
4.	*Ok*	Jempol dan telunjuk membentuk lingkaran	Belok Kanan (*Turn Right*)	1.750
5.	*Palm*	Telapak terbuka	Berhenti (*Stop*)	1.770
Total				8.756

Pembagian dataset menerapkan metode *hold-out validation* dengan tiga partisi (*three-way split*) menggunakan rasio 70:15:15. Dari total 8.756 citra, sebanyak 70% (6.129 citra) dialokasikan sebagai data latih (*Training Set*) yang digunakan model untuk mempelajari bobot melalui proses *backpropagation*. Sebanyak 15% (1.313 citra) dialokasikan sebagai data validasi (*Validation Set*) untuk memonitor kinerja model secara berkala selama pelatihan, mendeteksi indikasi *overfitting*, serta menjadi basis pemilihan bobot model terbaik (*best model*

checkpoint). Sisa 15% (1.314 citra) dialokasikan sebagai data uji (*Test Set*) yang sepenuhnya diisolasi dan hanya digunakan satu kali untuk evaluasi final setelah seluruh proses pelatihan selesai.

Data validasi dan data uji sengaja dipisah secara ketat untuk menghindari kebocoran informasi yang dapat membuat hasil akurasi model terlihat lebih bagus dari kemampuan aslinya. Penelitian ini juga tidak menerapkan *K-Fold Cross Validation*. Dengan jumlah data yang melimpah, pembagian data yang ada sudah terjamin keterwakilannya. Penerapan metode tersebut hanya akan melipatgandakan waktu dan beban kerja komputer selama proses pelatihan berlangsung.

Tahap terakhir dalam penyiapan data adalah melakukan augmentasi. Langkah ini bertujuan untuk memperkaya variasi gambar agar model tidak sekadar menghafal (*overfitting*) dan lebih siap mengenali pola baru. Penambahan variasi ini hanya dilakukan secara khusus pada data latih (*Training Set*). Data validasi dan data uji sengaja dibiarkan asli tanpa augmentasi supaya hasil penilaian performa model tetap murni, adil, dan konsisten. Berikut adalah teknik augmentasi yang diterapkan:

- *Random Resized Crop* (224, *scale* = [0.7, 1.0]) melakukan *crop* acak untuk mensimulasikan variasi jarak pengguna dari kamera,
- *Random Horizontal Flip* (p=0.5) membalik citra secara horizontal untuk mensimulasikan gestur tangan kiri maupun kanan,
- *Random Rotation* (±15°) rotasi acak untuk mensimulasikan variasi orientasi tangan yang alami,
- *Color Jitter* (*brightness*=0.3, *contrast*=0.3, *saturation*=0.3) variasi warna dan kontras untuk mensimulasikan perubahan kondisi pencahayaan.

3.5 Desain Sistem

Desain sistem perangkat lunak merupakan inti dari kecerdasan buatan yang dibangun dalam penelitian ini. Bagian ini bertanggung jawab untuk memproses data visual yang diterima dari kamera, mengenali pola gestur tangan menggunakan algoritma *Deep Learning*, dan menerjemahkan hasil pengenalan tersebut menjadi sinyal kendali fisik bagi robot. Secara konseptual, alur pemrosesan data dirancang mengalir mulai dari pra-pemrosesan citra, ekstraksi fitur dan klasifikasi, hingga logika pengambilan keputusan.

3.5.1 Pra-pemrosesan Citra (*Image Preprocessing*)

Data citra mentah yang diakuisisi dari kamera memiliki resolusi 384×384 piksel dengan nilai intensitas warna RGB yang bervariasi. Sebelum citra ini dapat diproses oleh model kecerdasan buatan, diperlukan tahapan pra-pemrosesan untuk menyeragamkan format input sesuai standar arsitektur ResNet-50. Proses ini diawali dengan pengubahan ukuran citra (*resizing*) menggunakan teknik interpolasi bilineer untuk memadatkan dimensi citra menjadi 224×224 piksel. Langkah ini krusial untuk memastikan kompatibilitas dimensi tensor input tanpa menghilangkan informasi spasial penting dari bentuk tangan.

Setelah ukuran disesuaikan, tahap selanjutnya adalah normalisasi nilai piksel. Citra digital umumnya memiliki rentang nilai intensitas $[0,255]$, yang jika langsung diolah dapat menyebabkan instabilitas numerik selama proses komputasi. Oleh karena itu, dilakukan normalisasi *Z-Score* untuk mengubah

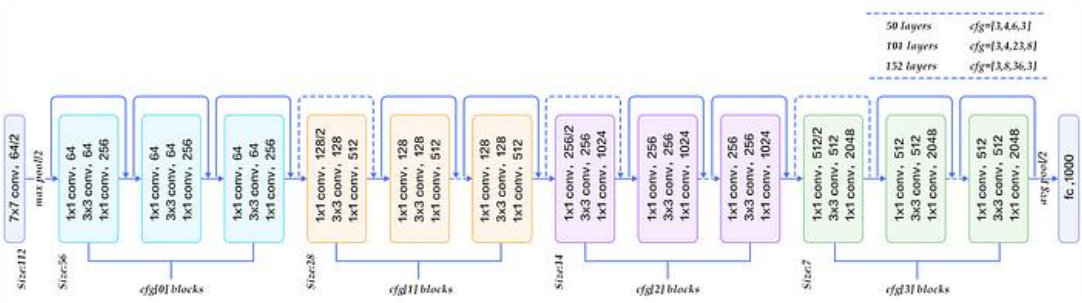
distribusi nilai piksel agar memiliki rata-rata (*mean*) 0 dan standar deviasi 1. Secara matematis, transformasi ini dilakukan pada setiap kanal warna c menggunakan persamaan:

$$x'_c = \frac{(x_c - \mu_c)}{\sigma_c} \tag{3.1}$$

Dimana x_c adalah nilai piksel asli, μ_c adalah rata-rata standar ImageNet ([0.485,0.456,0.406]), dan σ_c adalah standar deviasi ([0.229,0.224,0.225]) (Rahmawati dkk., 2025). Hasil dari proses ini adalah tensor input yang terstandarisasi, yang memungkinkan model untuk berkonsentrasi pada pembelajaran fitur bentuk tanpa terganggu oleh variasi intensitas pencahayaan ekstrem.

3.5.2 Arsitektur Model dan *Transfer Learning*

Inti dari sistem pengenalan gestur ini dibangun menggunakan arsitektur *Residual Network* dengan kedalaman 50 lapisan (ResNet-50). Pemilihan arsitektur ini didasari oleh kemampuannya dalam mengatasi masalah degradasi gradien (*vanishing gradient problem*) yang sering terjadi pada jaringan saraf tiruan yang sangat dalam. Solusi yang ditawarkan ResNet adalah penggunaan blok residual dengan mekanisme koneksi pintas (*skip connection*). Mekanisme ini memungkinkan aliran informasi dari input **x** diteruskan secara langsung melewati lapisan konvolusi *F(x)*, sehingga output blok $H(x)$ diformulasikan sebagai penjumlahan elemen $H(x) = F(x) + x$. Dengan cara ini, fitur-fitur visual penting dari citra gestur tangan dapat dipertahankan hingga lapisan terdalam.



Gambar 3.4 Arsitektur ResNet-50

Untuk menghemat beban komputasi pada arsitektur berjumlah 50 lapisan, setiap blok residualnya menggunakan format *Bottleneck* (leher botol) yang terdiri dari susunan tiga lapisan konvolusi:

1. Konvolusi 1×1: Berfungsi untuk mereduksi dimensi (*channel*) matriks input (menciptakan efek "leher botol" yang menyempit).
2. Konvolusi 3×3: Berfungsi sebagai pengekstraksi fitur utama pada dimensi yang sudah dikecilkan, sehingga komputasi jauh lebih ringan.
3. Konvolusi 1×1: Berfungsi mengekspansi kembali dimensi ke ukuran aslinya agar dapat ditambahkan secara matematis dengan input x.

Keseluruhan 50 lapisan yang memiliki bobot parameter (*trainable layers*) pada ResNet-50 terbagi ke dalam 6 tahapan pemrosesan berikut:

1. Tahap 1 (Inisialisasi) - 1 Lapisan

   1x Konvolusi ukuran 7×7 (64 filter, *stride* 2). Dilanjutkan operasi *Max Pooling* ukuran 3×3 (*stride* 2) untuk mengecilkan resolusi awal secara cepat.

2. Tahap 2 - 9 Lapisan

   Tersusun dari 3 buah blok *Bottleneck* (3 blok $\times$ 3 layer $= 9$ layer).

3. Tahap 3 - 12 Lapisan

   Tersusun dari 4 buah blok *Bottleneck* ($4\ \text{blok} \times 3\ \text{layer} = 12\ \text{layer}$).

4. Tahap 4 - 18 Lapisan

   Tersusun dari 6 buah blok *Bottleneck* ($6\ \text{blok} \times 3\ \text{layer} = 18\ \text{layer}$).

5. Tahap 5 - 9 Lapisan

   Tersusun dari 3 buah blok *Bottleneck* ($3\ \text{blok} \times 3\ \text{layer} = 9\ \text{layer}$).

6. Tahap Klasifikasi Akhir - 1 Lapisan

   Terdapat juga lapisan *Global Average Pooling* yang dimana tugasnya untuk meratakan matriks menjadi vektor. Serta termasuk 1x *Fully Connected Layer* menggunakan aktivasi *Softmax* untuk mengeluarkan nilai probabilitas kelas.

Penelitian ini menerapkan teknik *Transfer Learning* yang hanya difokuskan pada penyesuaian sebagian lapisan model. Strateginya dibagi menjadi tiga tahap. Pertama, lapisan awal dibekukan karena fungsinya untuk mengenali pola dasar gambar (seperti garis dan tepi) sudah sangat baik. Kedua, lapisan di bagian tengah dan atas kembali dilatih agar lebih peka terhadap ciri khas bentuk tangan. Terakhir, struktur *Fully Connected Layer* lama dibongkar dan diganti dengan kepala klasifikasi baru.

Fully Connected Layer baru dirancang sebagai berikut:

Linear (2048 → 512) → *Dropout* (0.5) → ReLU → Linear (512 → 5)

Penambahan dua lapisan *Dropout*(p=0.5) berfungsi sebagai regularisasi untuk mencegah *overfitting* pada parameter baru yang harus dipelajari dari awal. Lapisan akhir menghasilkan 5 nilai logit sesuai jumlah kelas gestur navigasi robot.

3.5.3 Konfigurasi Pelatihan

Fungsi kerugian yang digunakan adalah *Cross-Entropy Loss*, pilihan standar untuk klasifikasi multi-kelas yang secara langsung mengoptimasi distribusi probabilitas output terhadap label *ground truth*.

Optimizer yang digunakan adalah Adam (*Adaptive Moment Estimation*) dengan konfigurasi *Differential Learning Rate* memberikan *learning rate* berbeda untuk setiap kelompok *layer* sesuai tingkat adaptasi yang dibutuhkan.

Tabel 3.3 Konfigurasi *Differential Learning Rate*

Kelompok *Layer*	*Learning Rate*	Rasionalisasi
Layer 3 (dilatih ulang)	1×10^{-5}	Fitur cukup baik, perlu penyesuaian sangat halus
Layer 4 (dilatih ulang)	1×10^{-4}	Butuh adaptasi lebih besar ke domain gestur
FC baru (dari nol)	1×10^{-3}	Parameter baru, butuh *learning rate* lebih besar

Selain itu diterapkan *learning rate scheduler* StepLR yang menurunkan *learning rate* semua kelompok *layer* sebesar faktor gamma = 0,1 setiap 5 *epoch*. Mekanisme ini memungkinkan model belajar dengan langkah besar di *epoch* awal (eksplorasi) dan beralih ke langkah kecil di *epoch* akhir

(eksploitasi). Pelatihan dilaksanakan selama 15 *epoch* dengan ukuran batch 32 menggunakan akselerasi GPU pada platform Google *Colaboratory*.

Setelah pelatihan selesai, model terbaik dikonversi dari format PyTorch (.pth) ke format *Open Neural Network Exchange* (ONNX) untuk memungkinkan inferensi lintas platform yang efisien, terutama pada perangkat komputasi terbatas seperti Raspberry Pi. ONNX *Runtime* yang dioptimalkan untuk CPU ARM memberikan kecepatan inferensi yang lebih tinggi dibandingkan menggunakan PyTorch secara langsung (Ashfaq dkk., 2022).

Proses konversi menggunakan torch.onnx.export() dengan opset versi 17 dan opsi *constant folding* untuk menghilangkan komputasi yang tidak perlu secara otomatis. *Dynamic axes* dikonfigurasi pada dimensi *batch size* untuk memungkinkan fleksibilitas ukuran batch saat inferensi. Setelah konversi, dilakukan verifikasi integritas menggunakan onnx.checker.check_model() serta validasi konsistensi output antara model PyTorch dan ONNX *Runtime.*

3.5.4 Klasifikasi Probabilitas dan Logika Kendali

Nilai logit yang dihasilkan oleh model belum merepresentasikan keputusan akhir. Untuk mendapatkan nilai probabilitas yang dapat diinterpretasikan, digunakan fungsi aktivasi *Softmax*. Fungsi ini mentransformasi nilai logit z dari setiap kelas i menjadi distribusi probabilitas $P(y_i)$ yang bernilai antara 0 hingga 1, dengan total penjumlahan seluruh kelas sama dengan 1. Persamaan *Softmax* didefinisikan sebagai:

$$P(y_i) = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \tag{3.3}$$

Berdasarkan probabilitas tersebut, sistem kemudian menjalankan algoritma logika kendali untuk menentukan aksi robot (Nwankpa dkk., 2018). Keputusan aksi robot diambil berdasarkan probabilitas tertinggi (*argmax*). Untuk mencegah robot bergerak akibat deteksi yang tidak meyakinkan, diterapkan mekanisme *confidence thresholding* sebesar 0,60 (60%). Perintah gerak hanya dikirim apabila *confidence score* melebihi batas ini. Jika tidak, sistem secara default mengirim karakter 'S' (Stop) untuk menjamin keamanan. Jika *threshold* terpenuhi, Raspberry akan mengirim sinyal pada *driver* sesuai dengan *mapping* untuk navigasi.

Pemetaan secara lengkap antara setiap kelas gestur yang berhasil diklasifikasikan dengan logika sinyal GPIO yang dikirimkan ke modul L298N disajikan pada Tabel 3.4. Logika *HIGH* (H) dan LOW (L) pada pin IN1–IN4 menentukan arah putaran masing-masing motor, sedangkan pin ENA dan ENB diaktifkan (*HIGH*) secara konstan selama sistem beroperasi untuk memastikan *driver* motor selalu siap menerima perintah.

Tabel 3.4 Pemetaan Kelas Gestur ke Logika Sinyal GPIO Motor

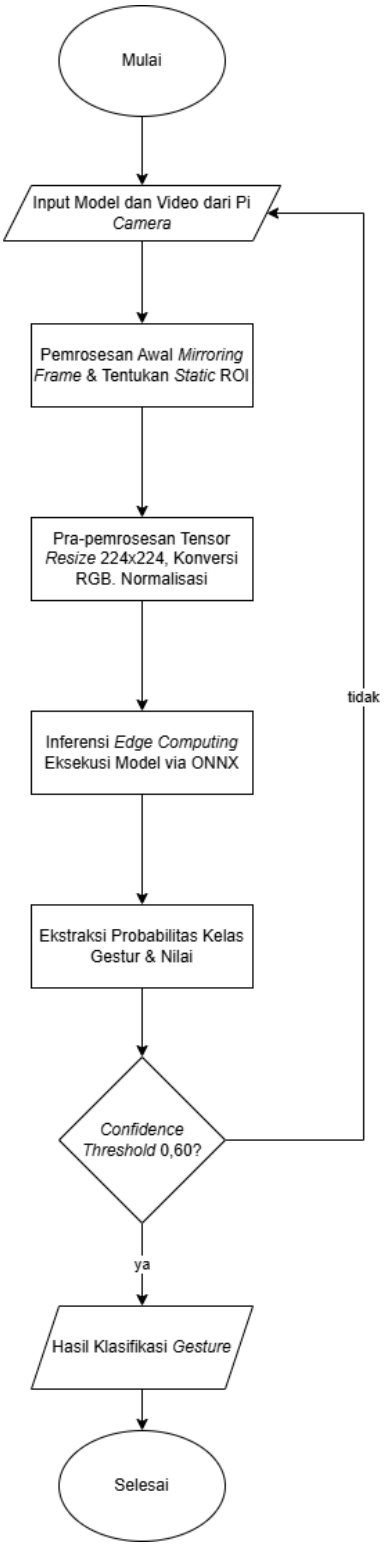
Kelas Gestur	Perintah	IN1 GPIO 4	IN2 GPIO 17	IN3 GPIO 27	IN4 GPIO 22	ENA ENB	Efek Gerakan
Fist (Kepalan)	*FORWARD* - Maju	H	L	H	L	H	Motor kiri & kanan berputar maju (searah jarum jam)

Kelas Gestur	Perintah	IN1 GPIO 4	IN2 GPIO 17	IN3 GPIO 27	IN4 GPIO 22	ENA ENB	Efek Gerakan
Like (Jempol)	*BACK* - Mundur	L	H	L	H	H	Motor kiri & kanan berputar mundur (berlawanan jarum jam)
Peace (V)	*LEFT* - Belok Kiri	L	H	H	L	H	Motor kiri mundur, motor kanan maju kemudian robot berputar ke kiri
Ok (Jari OK)	*RIGHT*- Belok Kanan	H	L	L	H	H	Motor kiri maju, motor kanan mundur maka robot berputar ke kanan
Palm (Telapak)	*STOP* - Berhenti	L	L	L	L	L	Semua sinyal *LOW* maka kedua motor berhenti sepenuhnya
(Tidak terdeteksi)	*STOP* (default)	L	L	L	L	L	*Confidence* < 60% maka sistem mengirim sinyal *STOP* secara otomatis untuk keamanan

Mekanisme belok pada sistem 2WD ini mengadopsi prinsip *differential steering*, yaitu perbedaan arah putaran antara motor kiri dan motor kanan yang menyebabkan robot berputar di tempat (*zero-radius turn*). Ketika perintah *LEFT* diterima, motor kiri diputar mundur sementara motor kanan diputar

maju, sehingga robot berputar ke arah kiri di sekitar titik tengah sumbu roda. Sebaliknya, perintah *RIGHT* membalik kondisi tersebut menghasilkan putaran ke arah kanan. Pendekatan ini dipilih karena kesederhanaan implementasinya pada sasis 2WD dengan satu *caster wheel*, sekaligus menghasilkan manuver belok yang tegas dan responsif terhadap perintah gestur.

Kondisi keamanan (*safety fallback*) diterapkan pada dua skenario: (a) ketika nilai *confidence score* hasil inferensi berada di bawah ambang batas 60%, dan (b) ketika tidak ada gestur yang terdeteksi dalam *frame* (misalnya tangan tidak berada dalam area ROI kamera). Pada kedua kondisi ini, sistem secara otomatis mengirimkan sinyal *LOW* pada seluruh pin IN1–IN4 serta menonaktifkan ENA dan ENB, sehingga kedua motor DC berhenti sepenuhnya. Mekanisme ini penting untuk mencegah robot bergerak secara tidak terduga akibat kesalahan deteksi atau gangguan visual sesaat.



Gambar 3.5 *Flowchart* Desain Sistem

3.6 Skenario Pengujian

Pengujian ini tidak hanya bertujuan untuk memastikan bahwa setiap komponen berfungsi, tetapi juga untuk mengukur seberapa andal sistem dalam menerjemahkan gestur tanfgan menjadi gerakan robot di lingkungan nyata. Rangkaian pengujian dibagi menjadi tiga skenario utama yang mencakup evaluasi kinerja model kecerdasan buatan, pengukuran responsivitas waktu nyata, dan pengujian fungsionalitas terhadap variasi kondisi lingkungan.

3.6.1 Pengujian Kinerja Model (*Model Performance Testing*)

Skenario pertama difokuskan pada evaluasi kuantitatif terhadap kemampuan model ResNet-50 dalam mengklasifikasikan gestur tangan. Pengujian ini dilakukan secara terisolasi pada komputer menggunakan data uji (*testing set*) yang belum pernah dilihat oleh model selama proses pelatihan. Metode evaluasi utama yang digunakan adalah *Confusion Matrix*, yaitu sebuah tabel representasi yang memetakan perbandingan antara prediksi model dengan label kebenaran dasar (*ground truth*). Tabel ini memungkinkan analisis mendalam mengenai seberapa sering model melakukan kesalahan prediksi antar kelas, misalnya apakah gestur *'Fist'* sering salah diprediksi sebagai *'Palm'*.

Dari hasil pemetaan *Confusion Matrix* tersebut, kinerja model akan dihitung menggunakan parameter statistik standar, yaitu Akurasi (*Accuracy*), Presisi (*Precision*), dan *Recall*. Akurasi mengukur rasio prediksi benar terhadap total data keseluruhan. Presisi mengukur seberapa akurat model saat menyatakan suatu kelas positif, sedangkan *Recall* mengukur kemampuan

model dalam menemukan kembali seluruh data positif yang ada. Secara matematis, akurasi total sistem dihitung menggunakan persamaan:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \times 100 \quad (3.4)$$

Dimana TP (*True Positive*) adalah jumlah data positif yang diprediksi benar, TN (*True Negative*) adalah data negatif yang diprediksi benar, FP (*False Positive*) adalah kesalahan prediksi positif, dan FN (*False Negative*) adalah kesalahan prediksi negatif. Target keberhasilan pada skenario ini adalah tercapainya tingkat akurasi rata-rata di atas 90% pada kelima kelas gestur.

3.6.2 Pengujian Responsivitas Sistem (*Execution time Testing*)

Skenario kedua bertujuan untuk mengukur efisiensi komputasi dan kecepatan respon sistem saat dijalankan secara *real-time*. Dalam konteks kendali robot beroda, keterlambatan respon (*Execution Time*) dapat berakibat fatal, seperti keterlambatan robot untuk berhenti saat diperintahkan. Pengujian ini dilakukan dengan mengukur selisih waktu antara momen gestur diperagakan di depan kamera (t_{start}) dengan momen aktuator motor mulai bergerak (t_{end}).

Proses pengukuran dilakukan menggunakan pencatatan waktu sistem (*timestamp logging*) pada perangkat lunak dengan satuan milidetik (ms). Untuk mendapatkan perhitungan yang akurat, sistem dirancang untuk mencatat empat titik waktu utama dalam satu siklus proses, yaitu:

1. t_0: Waktu saat *frame* berhasil ditangkap dari kamera (*Frame Capture*).

2. t_1: Waktu setelah proses *resizing* dan normalisasi selesai (*Preprocessing Done*).
3. t_2: Waktu setelah model mengeluarkan hasil prediksi kelas (*Inference Done*).
4. t_3: Waktu setelah sinyal GPIO berhasil dikirimkan ke L298N (*Data Sent*).

Berdasarkan titik waktu tersebut, perhitungan komponen waktu eksekusi dirumuskan sebagai berikut:

a. Waktu Pra-pemrosesan (Δt_{pre})

Merupakan durasi yang dibutuhkan CPU untuk mengubah dimensi dan menormalisasi citra.

$$\Delta t_{pre} = t_1 - t_0 \tag{3.5}$$

b. Waktu Inferensi Model (Δt_{infer})

Merupakan durasi komputasi model ResNet-50 untuk memprediksi probabilitas gestur.

$$\Delta t_{infer} = t_2 - t_1 \tag{3.6}$$

c. Waktu Transmisi & Logika (Δt_{trans})

Merupakan waktu yang dibutuhkan untuk pengecekan *threshold*, pemetaan kelas gestur, dan pengiriman sinyal GPIO.

$$\Delta t_{trans} = t_3 - t_2 \tag{3.7}$$

d. Total Waktu Eksekusi Sistem (L_{total})

Total waktu tunda dari input visual hingga output perintah dihitung sebagai akumulasi dari ketiga komponen 3.5 hingga 3.7:

$$L_{total} = \Delta t_{pre} + \Delta t_{infer} + \Delta t_{trans} \tag{3.8}$$

Atau secara sederhana:

$$L_{total} = t_3 - t_0 \tag{3.9}$$

Untuk mendapatkan data yang valid secara statistik, pengujian dilakukan sebanyak N kali (misal: $N = 100$ frame). Kinerja sistem kemudian dievaluasi berdasarkan Rata-rata waktu eksekusi ($\bar{L}$) dan Frame Per Detik (FPS) yang dihitung menggunakan persamaan:

$$\bar{L} = \frac{1}{N}\sum_{i=1}^{N} L_{total}(i) \tag{3.10}$$

$$FPS_{avg} = \frac{1000}{\bar{L}\,(ms)} \tag{3.11}$$

Pengujian ini dianggap berhasil jika sistem mampu mempertahankan waktu eksekusi total rata-rata di bawah ambang batas toleransi interaksi manusia, yaitu kurang dari 150 ms (atau $>$ 6 FPS), sehingga pergerakan robot terasa responsif dan alami.

Untuk merekam hasil pengukuran variabel waktu t_0, t_1, t_2 dan t_3, dirancang tabel instrumen pengujian seperti pada Tabel 3.5. Tabel ini akan digunakan untuk mencatat data mentah (*raw data*) dari 10 sampel *frame* acak selama pengujian berlangsung.

Tabel 3.5 Rancangan Tabel Pengukuran Waktu Eksekusi Sistem

Sampel Ke-(i)	Waktu *Capture* (t0)	Waktu *Pre-process* (t1)	Waktu Inferensi (t2)	Waktu GPIO (t3)	Δt*pre* (t1−t0)	Δt*infer* (t2−t1)	Δt*trans* (t3−t2)	Total Waktu Eksekusi (L)
1	...	...	...	...	...	...	...	...
2	...	...	...	...	...	...	...	...

Sampel Ke-(i)	Waktu *Capture* (t0)	Waktu *Pre-process* (t1)	Waktu Inferensi (t2)	Waktu GPIO (t3)	Δt*pre* (t1−t0)	Δt*infer* (t2−t1)	Δt*trans* (t3−t2)	Total Waktu Eksekusi (L)
...	...	...	...	...	...	...	...	...
10	...	...	...	...	...	...	...	...
Rata-rata	-	-	-	-	... ms	... ms	... ms	... ms

Sebagai ilustrasi penerapan rumus perhitungan disajikan pada tabel 3.5 sebagai contoh perhitungan sampel data pada *frame* ke-1 dengan asumsi data *timestamp* (dalam milidetik) sebagai berikut:

- t_0 (*Capture*) = 1000 ms
- t_1 (*Pre-proc*) = 1025 ms
- t_2 (Inferensi) = 1145 ms
- t_3 (*Sent*) = 1150 ms

Maka perhitungan komponen waktu eksekusinya adalah:

1. Waktu Pra-pemrosesan (Δt_{pre}):

$$\Delta \mathrm{t_{pre}} = 1025 - 1000 = 25 \text{ ms} \tag{3.12}$$

2. Waktu Inferensi (Δt_{infer}):

$$\Delta \mathrm{t_{infer}} = 1145 - 1025 = 120 \text{ ms} \tag{3.13}$$

3. Waktu Transmisi (Δt_{trans}):

$$\Delta \mathrm{t_{trans}} = 1150 - 1145 = 5 \text{ ms} \tag{3.14}$$

4. Total Waktu Eksekusi (L_{total}):

$$\mathrm{L_{total}} = 25 + 120 + 5 = 150 \text{ ms} \tag{3.15}$$

Berdasarkan total waktu eksekusi, estimasi *Frame Rate* (FPS) sistem dapat dihitung:

$$\text{FPS} = \frac{1000}{150} \approx 6.67 \text{ FPS} \tag{3.16}$$

Hasil perhitungan akhir berupa rata-rata waktu eksekusi ($\bar{L}$) dari seluruh sampel akan digunakan sebagai indikator utama untuk menentukan apakah sistem memenuhi syarat responsivitas (target $<$ 1 detik).

3.6.3 Pengujian Fungsionalitas Lingkungan (*Black Box Testing*)

Skenario ketiga merupakan pengujian fungsionalitas metode kotak hitam (*Black Box Testing*) untuk menguji ketahanan (*robustness*) sistem terhadap variasi kondisi penggunaan di dunia nyata. Berbeda dengan pengujian model yang menggunakan data statis, pengujian ini melibatkan interaksi langsung antara pengguna dan robot fisik. Variabel bebas yang diuji meliputi variasi jarak pengguna terhadap kamera serta variasi sudut pengambilan gambar.

Pada pengujian jarak, pengguna akan memberikan perintah gestur dari jarak 50 cm, 100 cm, hingga 150 cm untuk mengetahui jangkauan efektif pandangan robot. Sementara itu, pengujian sudut dilakukan dengan memiringkan orientasi tangan pengguna secara horizontal maupun vertikal untuk mensimulasikan ketidaksempurnaan posisi tangan manusia yang alami. Selain itu, pengujian juga dilakukan pada kondisi pencahayaan yang berbeda, yaitu kondisi terang dan redup, untuk memastikan bahwa normalisasi citra yang diterapkan pada tahap pra-pemrosesan mampu menangani perubahan intensitas cahaya. Hasil dari skenario ini akan dicatat dalam bentuk tabel keberhasilan manuver robot untuk setiap perintah yang diberikan.

BAB IV

UJI COBA DAN PEMBAHASAN

4.1 Hasil Pelatihan Model

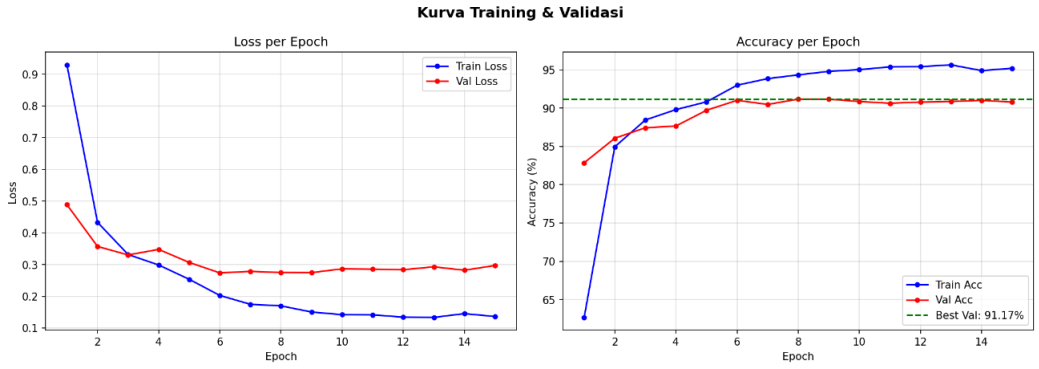
Proses pelatihan model ResNet-50 dilaksanakan selama 15 *epoch* menggunakan *Google Colaboratory* dengan akselerasi GPU. Bobot awal model diinisialisasi dari hasil *pretraining* pada dataset *ImageNet*, kemudian dilakukan *Fine-Tuning* Selektif terhadap *layer* 3, *layer* 4, dan kepala klasifikasi baru dengan konfigurasi *Differential Learning Rate*. Subbab ini menyajikan analisis perjalanan pelatihan melalui data historis *loss* dan akurasi, serta penilaian terhadap kemampuan generalisasi model.

4.1.1 Kurva *Loss* per *Epoch*

Kurva *loss* pada Gambar 4.1 memperlihatkan tren penurunan yang konsisten dan stabil pada kedua partisi data sepanjang 15 *epoch* pelatihan. Pada *epoch* pertama, *train loss* berada pada nilai yang relatif tinggi yakni sekitar 0,91, mencerminkan kondisi awal di mana lapisan yang dilatih ulang belum sepenuhnya beradaptasi dengan distribusi data gestur. Seiring berjalannya pelatihan, *train loss* mengalami penurunan yang signifikan dan mencapai nilai sekitar 0,13 pada *epoch* ke-15.

Tercatat pada *epoch* pertama, *validation loss* bernilai 0,48. Angka tersebut secara bertahap turun dan akhirnya stabil pada rentang 0,28–0,30 mulai dari *epoch* ke-6 hingga selesai. Pola yang stabil ini menandakan model telah mencapai konvergensi yang baik dan terbukti mampu merespons data

yang belum pernah ia jumpai sebelumnya. Walaupun grafik *train loss* terlihat terus menurun, penurunan tersebut tidak lagi diikuti oleh *validation loss* setelah melewati *epoch* ke-6. Kondisi seperti ini adalah fenomena umum, mengingat mendekati *epoch* akhir, model mulai mengalami sedikit *overfitting* terhadap data latih.



Gambar 4.1 Kurva *Training* & Validasi (*Loss* dan *Accuracy* per *Epoch*)

4.1.2 Kurva Akurasi per *Epoch*

Berdasarkan Gambar 4.1, grafik akurasi menunjukkan bahwa model belajar dengan sangat cepat di tahap awal. Sempat terjadi naik-turun (*fluktuasi*) yang cukup tajam pada nilai *train accuracy* saat memasuki epoch ke-2. Hal ini sangat wajar karena model sedang menyesuaikan kecepatan belajarnya (*learning rate*) dan beradaptasi dengan susunan lapisan yang baru. Namun, setelah melewati epoch ke-3, angka akurasinya terus naik dengan stabil hingga akhirnya ditutup pada angka 95,28% di *epoch* ke-15.

Validation accuracy pada *epoch* pertama sudah mencatatkan nilai yang cukup tinggi yakni sekitar 82%, yang mengkonfirmasi efektivitas strategi *Transfer Learning* dalam memanfaatkan pengetahuan *pretrained ImageNet* sebagai titik awal yang kuat. Nilai tertinggi *validation accuracy* (*best*

checkpoint) berhasil diraih pada nilai 91,17%, dan nilai ini digunakan sebagai bobot model final yang disimpan untuk tahap evaluasi dan *deployment*.

4.1.3 Analisis Overfitting

Penilaian terhadap kemampuan generalisasi model dilakukan dengan menganalisis selisih antara *train accuracy* dan *validation accuracy* pada akhir pelatihan. Hasil pelatihan menunjukkan bahwa *train accuracy* akhir mencapai 95,28%, sedangkan *validation accuracy* akhir berada pada 90,78%, menghasilkan gap sebesar 4,42%. Berdasarkan kriteria yang telah ditetapkan dalam desain penelitian, gap yang berada di bawah ambang batas 5% mengindikasikan bahwa model memiliki kemampuan generalisasi yang baik dan tidak mengalami *overfitting* yang signifikan.

Keberhasilan ini tidak lepas dari dukungan beberapa teknik pengaturan (regularisasi) selama proses pelatihan. Salah satunya adalah penerapan *Dropout* sebesar 0,5 di dua titik pada lapisan klasifikasi untuk mematikan sebagian neuron secara acak. Selain itu, digunakan juga strategi augmentasi untuk terus memberikan variasi gambar yang beragam. Langkah penyeimbang lainnya adalah pengaturan StepLR yang secara otomatis menurunkan *learning rate* setiap 5 *epoch*. Penurunan kecepatan belajar ini sangat penting agar model tidak terlalu drastis dalam mengubah bobot nilainya saat mendekati akhir masa pelatihan.

4.1.4 Rekapitulasi Hasil Pelatihan per *Epoch*

Tabel 4.1 menyajikan rekapitulasi nilai *train loss*, *validation loss*, *train accuracy*, dan *validation accuracy* untuk setiap *epoch* selama proses pelatihan berlangsung.

Tabel 4.1 Rekapitulasi Hasil Pelatihan per *Epoch*

Epoch	***Train Loss***	***Val Loss***	***Train Acc* (%)**	***Val Acc* (%)**
1	0.9292	0.4885	0.6267	0.8286
2	0.4331	0.3570	0.8494	0.8606
3	0.3316	0.3302	0.8845	0.8743
4	0.2985	0.3476	0.8980	0.8766
5	0.2534	0.3066	0.9081	0.8972
6	0.2026	0.2736	0.9298	0.9101
7	0.1747	0.2785	0.9385	0.9048
8	0.1697	0.2749	0.9434	0.9117
9	0.1503	0.2745	0.9480	0.9117
10	0.1420	0.2864	0.9502	0.9086
11	0.1414	0.2855	0.9538	0.9063
12	0.1342	0.2837	0.9542	0.9078
13	0.1333	0.2928	0.9564	0.9086
14	0.1453	0.2821	0.9489	0.9101
15	0.1362	0.2969	0.9520	0.9078
Best	-	-	**95,64**	**91,17**

4.2 Hasil Pengujian

4.2.1 Hasil Evaluasi Model pada Data Uji (*Test Set)*

Tahap evaluasi akhir mengandalkan 1.314 gambar dari data uji (*test set*) yang sengaja dipisahkan dan sama sekali tidak dilibatkan selama proses pelatihan berlangsung. Pengujian akhir ini menjalankan model menggunakan

versi bobot terbaiknya (*best checkpoint*), yakni ketika nilai akurasi validasi berhasil menyentuh puncak tertingginya di angka 91,17%. Pada subbab ini, hasil pengujian tersebut akan dibedah secara lengkap. Pembahasannya mencakup tingkat akurasi secara umum, rincian laporan klasifikasi untuk tiap gestur, gambaran matriks kebingungan (*confusion matrix*), hingga analisis kurva ROC-AUC.

1. Akurasi Keseluruhan (*Overall Accuracy*)

Model ResNet-50 dengan strategi *Transfer Learning Fine-Tuning* Selektif berhasil mencapai akurasi keseluruhan sebesar 91,10% pada *test set*. Nilai ini melampaui target keberhasilan yang telah ditetapkan dalam skenario pengujian, yaitu akurasi rata-rata di atas 90% pada kelima kelas gestur navigasi robot. Pencapaian ini mengkonfirmasi bahwa model mampu beradaptasi pada pola visual gestur tangan secara efektif terhadap data yang belum pernah dilihat sebelumnya.

Nilai akurasi *test set* (91,10%) yang mendekati nilai *best validation accuracy* (91,17%) juga mengindikasikan konsistensi performa model di antara kedua partisi data tersebut, sehingga memperkuat kesimpulan bahwa model tidak mengalami *overfitting* yang berarti terhadap *validation set* selama proses pemilihan bobot.

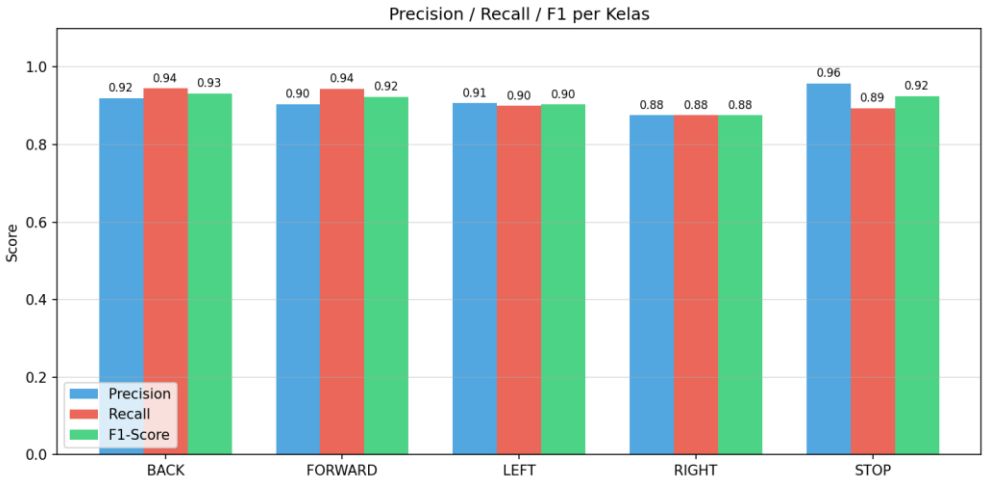
2. *Classification Report* per Kelas

Tabel 4.2 menyajikan *classification report* yang merupakan ringkasan metrik evaluasi untuk setiap kelas gestur secara individual,

beserta nilai rata-rata makro (*macro average*) dan rata-rata tertimbang (*weighted average*) secara keseluruhan.

Tabel 4.2 *Classification Report* Model ResNet-50 pada *Test Set*

Kelas	*Precision*	*Recall*	F1-*Score*	*Support*
BACK	0,9189	0,9444	0,9315	252
FORWARD	0,9036	0,9440	0,9234	268
LEFT	0,9064	0,8996	0,9030	269
RIGHT	0,8755	0,8755	0,8755	273
STOP	0,9574	0,8929	0,9240	252
Accuracy	-	-	0,9110	1.314
Macro Avg	0,9124	0,9113	0,9115	1.314
Weighted Avg	0,9116	0,9110	0,9109	1.314



Gambar 4.2 Grafik *Precision* / *Recall* / F1-*Score* per Kelas

Berdasarkan data pada Tabel 4.2, seluruh kelas gestur berhasil mencapai F1-*Score* di atas 0,87, yang mengindikasikan keseimbangan yang baik antara kemampuan presisi dan kemampuan deteksi (*recall*) model pada seluruh kelas. Kelas *BACK* mencatat F1-*Score* tertinggi sebesar 0,9315

dengan *recall* 0,9444 dan *precision* 0,9189, menunjukkan bahwa gestur telapak tangan menghadap ke atas (jempol atas) memiliki karakteristik visual yang paling distinktif dan mudah dibedakan oleh model dari kelas lainnya.

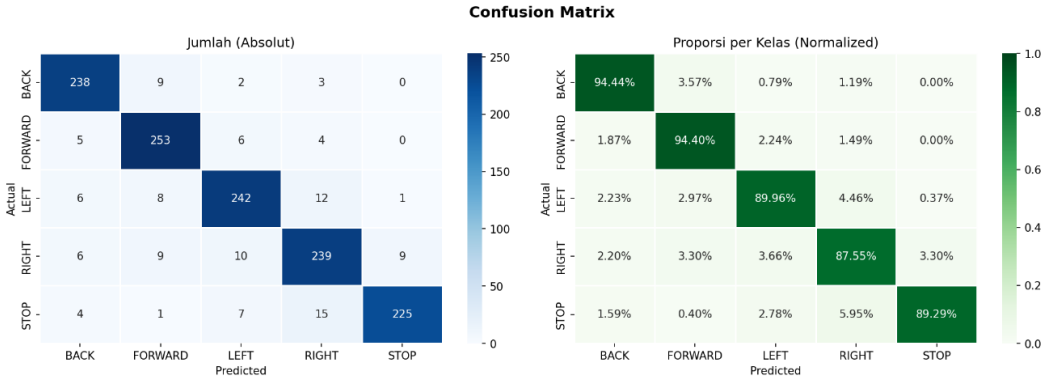
Kelas *STOP* mencatat *precision* tertinggi di antara semua kelas yakni 0,9574, yang berarti ketika model memprediksi gestur *STOP*, tingkat kebenarannya sangat tinggi. Hal ini sangat krusial dari aspek keamanan sistem, mengingat perintah berhenti harus dapat diandalkan dengan tingkat keyakinan yang tinggi untuk mencegah robot terus bergerak dalam situasi yang tidak diinginkan.

Kelas *RIGHT* mencatat performa terendah dengan F1-*Score* sebesar 0,8755 dan nilai *precision* maupun *recall* yang simetris di angka yang sama. Kondisi ini mengindikasikan bahwa gestur "*OK*" yang mewakili perintah belok kanan memiliki kesamaan visual dengan beberapa kelas lain, sehingga lebih rentan terhadap kesalahan klasifikasi. Analisis lebih mendalam terhadap pola kesalahan ini disajikan melalui *confusion matrix* pada subbab berikutnya.

3. Analisis *Confusion Matrix*

Confusion matrix pada Gambar 4.3 memetakan distribusi prediksi model terhadap seluruh 1.314 sampel test set dalam dua format, yaitu absolut (jumlah citra) dan ternormalisasi (proporsi per kelas). Diagonal utama pada kedua matrix merepresentasikan prediksi yang benar,

sedangkan sel-sel di luar diagonal merepresentasikan kesalahan klasifikasi antar kelas.



Gambar 4.3 *Confusion Matrix* Absolut dan Ternormalisasi

Berdasarkan nilai asli pada *confusion matrix*, kita bisa melihat rincian ketepatan prediksi untuk masing-masing kelas. Gestur *BACK* berhasil ditebak dengan tepat sebanyak 238 dari total 252 gambar (*recall* 94,44%). Selanjutnya, model juga sukses mengenali 253 dari 268 gambar pada kelas *FORWARD* (*recall* 94,40%). Untuk gestur *LEFT* dan *RIGHT*, akurasi tebakannya masing-masing menyentuh angka 242 dari 269 sampel (89,96%) serta 239 dari 273 sampel (87,55%). Terakhir, dari 252 gambar gestur *STOP* yang diujikan, sebanyak 225 di antaranya berhasil diidentifikasi dengan benar, yang setara dengan persentase *recall* 89,29%.

Kelas *BACK* dan *FORWARD* mencatat *recall* tertinggi masing-masing sebesar 94,44% dan 94,40%, mencerminkan bahwa gestur kepalan tangan (*FORWARD*) dan gestur jempol atas (*BACK*) memiliki pola visual yang paling khas dan konsisten dalam dataset. Sebaliknya, kelas *RIGHT* mencatat *recall* terendah sebesar 87,55%, dengan total 34 sampel yang salah

diklasifikasikan. Pola kesalahan yang dominan pada kelas ini adalah salah prediksi ke kelas *LEFT* (10 sampel) dan *STOP* (9 sampel). Temuan ini menunjukkan bahwa gestur "*OK*" yang merepresentasikan perintah *RIGHT* memiliki kemiripan visual dengan gestur "V" (*LEFT*) maupun gestur telapak terbuka (*STOP*) pada kondisi tertentu, seperti perubahan sudut pergelangan tangan atau jarak yang bervariasi.

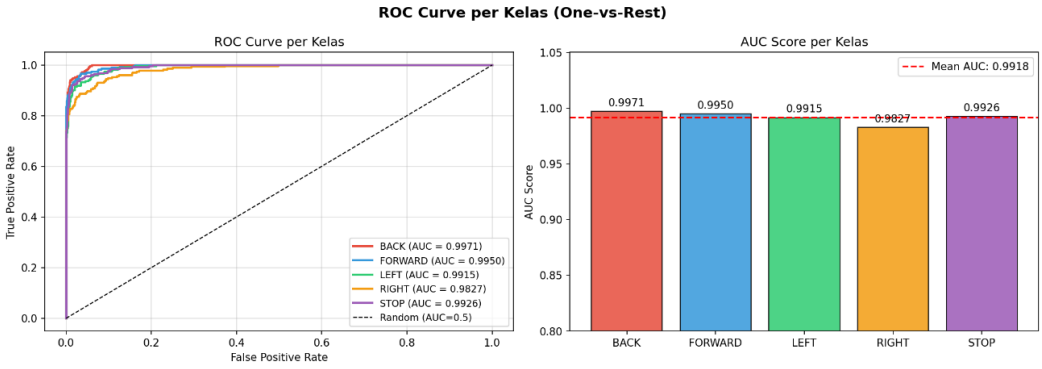
Kelas *STOP* juga menunjukkan pola kesalahan yang menarik, di mana 15 dari 27 sampel yang salah diklasifikasikan diprediksi sebagai kelas *RIGHT*. Hal ini dapat dijelaskan oleh kesamaan bentuk telapak terbuka dengan gestur "*OK*" pada sudut pandang tertentu, khususnya saat jari-jari tidak sepenuhnya terlihat oleh kamera.

Tabel 4.3 Ringkasan Analisis Kesalahan per Kelas

Kelas	Benar	Salah	*Recall*	Kesalahan Dominan
BACK	238	14	94,44%	Ke *FORWARD* (9 sampel)
FORWARD	253	15	94,40%	Ke *LEFT* (6 sampel)
LEFT	242	27	89,96%	Ke *RIGHT* (12 sampel)
RIGHT	239	34	87,55%	Ke *LEFT* (10), *STOP* (9 sampel)
STOP	225	27	89,29%	Ke *RIGHT* (15 sampel)

4. Analisis ROC *Curve* dan AUC

Analisis *Receiver Operating Characteristic* (ROC) *Curve* dilakukan menggunakan pendekatan *One-vs-Rest* (OvR) untuk mendapatkan evaluasi kemampuan diskriminasi model per kelas secara *independent*. Nilai Area *Under the Curve* (AUC) yang mendekati 1,0 mengindikasikan kemampuan model membedakan satu kelas dari kelas lainnya dengan sangat baik.



Gambar 4.4 ROC *Curve* per Kelas dan AUC *Score*

Berdasarkan gambar 4.4, seluruh kelas gestur mencapai nilai AUC yang sangat tinggi, dengan *mean* AUC sebesar 0,9918. Hasil ini mengkonfirmasi bahwa model ResNet-50 dengan *Transfer Learning* memiliki kemampuan diskriminasi yang sangat kuat dalam membedakan setiap kelas gestur dari kelas-kelas lainnya.

Kelas *BACK* mencatat AUC tertinggi sebesar 0,9971, yang konsisten dengan temuan sebelumnya bahwa gestur ini memiliki representasi fitur visual yang paling distinktif dalam ruang fitur yang dipelajari model. Kelas *RIGHT* kembali mencatat AUC terendah sebesar 0,9827, namun nilai ini masih tergolong sangat baik dan jauh di atas ambang batas yang diterima secara umum (AUC > 0,90 dianggap sangat baik dalam literatur klasifikasi citra). Seluruh kurva ROC terlihat menuju sudut kiri atas grafik secara konsisten, menunjukkan bahwa model beroperasi jauh di atas garis acak (*random classifier* dengan AUC = 0,5).

5. Ringkasan Evaluasi Akhir

Tabel 4.4 merangkum seluruh metrik evaluasi model ResNet-50 pada test set sebagai acuan komprehensif performa akhir sistem.

Tabel 4.4 Ringkasan Evaluasi Akhir Model pada *Test Set*

Metrik Evaluasi	Nilai
Test Accuracy	91,10%
Best Validation Accuracy	91,17%
Mean AUC (*One-vs-Rest)*	0,9918
Macro F1-*Score*	0,9115
Weighted F1-*Score*	0,9109
Total Sampel Uji	1.314 citra
Kelas Terbaik (F1)	*BACK* - 0,9315
Kelas Terlemah (F1)	*RIGHT* - 0,8755

Secara keseluruhan, model ResNet-50 dengan strategi *Transfer Learning Fine-Tuning* Selektif dan *Differential Learning Rate* berhasil memenuhi seluruh target kinerja yang ditetapkan: *test accuracy* 91,10% melampaui target > 90%, *mean* AUC 0,9918 mengindikasikan kemampuan diskriminasi yang sangat tinggi, serta *macro* F1-*Score* 0,9115 menunjukkan keseimbangan yang baik antara *precision* dan *recall* di seluruh kelas. Temuan ini memvalidasi efektivitas pendekatan *Transfer Learning* dalam mengatasi keterbatasan data domain-spesifik pada kasus pengenalan gestur tangan untuk navigasi robot beroda.

4.2.2 Hasil Pengujian Responsivitas Sistem (*Execution Time Testing*)

Pengujian responsivitas sistem bertujuan untuk mengukur efisiensi komputasi dan kecepatan respons sistem secara *end-to-end*, mulai dari *frame*

ditangkap kamera hingga sinyal kendali berhasil dikirimkan. Pengujian dilaksanakan pada tiga *environment* yang berbeda yaitu lingkungan pengembangan (laptop) dan lingkungan *deployment* Raspberry Pi 3 Model B dan Raspberry Pi 4 Model B. Pemisahan pengujian pada dua perangkat ini bertujuan untuk memberikan gambaran yang komprehensif mengenai performa sistem pada kondisi komputasi yang berbeda, sekaligus mengidentifikasi kesenjangan performa antara perangkat pengembangan dan perangkat *deployment*.

Pada setiap pengujian, sistem dirancang untuk mencatat empat titik waktu utama dalam satu siklus proses, yaitu: t_0 (waktu *frame* ditangkap), t_1 (waktu setelah *preprocessing* selesai), t_2 (waktu setelah inferensi model selesai), dan t_3 (waktu setelah sinyal kendali dikirimkan). Berdasarkan keempat titik waktu tersebut, tiga komponen waktu eksekusi dihitung: waktu pra-pemrosesan ($\Delta t\_pre = t_1 - t_0$), waktu inferensi ($\Delta t\_infer = t_2 - t_1$), dan waktu transmisi ($\Delta t\_trans = t_3 - t_2$), dengan total waktu eksekusi $L\_total = t_3 - t_0$.

1. **Pengujian Waktu Eksekusi pada Laptop (*Environment* Pengembangan)**

   Pengujian pertama dilangsungkan pada perangkat laptop tempat sistem ini dibangun. Proses uji coba ini memakai model ResNet-50 ONNX berpresisi penuh (FP32) sebesar 93,8 MB. Untuk memaksimalkan tenaga dari prosesor *multi-core*, konfigurasi ONNX Runtime dipasang untuk membagi beban kerja ke dalam empat *thread* paralel.

Tabel 4.5 Data Pengukuran Waktu Eksekusi pada Laptop

No.	Δt_*pre* (ms)	Δt_*infer* (ms)	Δt_*trans* (ms)	L_total (ms)
1	1.78	61.09	0.000	62.87
2	2.44	48.34	0.010	50.79
3	2.03	46.95	0.000	48.99
4	2.41	46.35	0.000	48.76
5	1.37	47.50	0.000	48.87
6	1.73	46.61	0.000	48.35
7	1.54	44.87	0.000	46.41
8	1.83	47.91	0.000	49.74
9	1.94	46.30	0.000	48.24
10	3.12	45.39	0.000	48.51
Mean	2.02	48.13	0.001	50.15

Berdasarkan data pada Tabel 4.5, total waktu eksekusi sistem rata-rata pada laptop tercatat sebesar 50.15 ms, yang setara dengan kecepatan inferensi rata-rata sebesar 19.94 FPS. Nilai ini jauh melampaui target responsivitas yang ditetapkan yaitu di bawah 150 ms (> 6 FPS), sehingga pengujian waktu eksekusi pada *environment* laptop dinyatakan berhasil.

Komponen waktu eksekusi yang paling dominan adalah waktu inferensi model (Δt_*infer*) dengan rata-rata 48.13 ms, menyumbang 96.0% dari total waktu eksekusi. Waktu pra-pemrosesan (Δt_*pre*) rata-rata 2.02 ms mencakup operasi *resize*, konversi ruang warna, dan normalisasi Z-*Score*. Waktu transmisi sinyal (Δt_*trans*) tercatat sangat kecil yaitu 0,001 ms karena pada *environment* laptop simulasi pengiriman sinyal dilakukan tanpa *hardware* fisik.

Tabel 4.6 Ringkasan Komponen Waktu Eksekusi pada Laptop

Komponen Waktu eksekusi	Rata-rata (ms)	Proporsi (%)
Pra-pemrosesan (Δt_*pre*)	2.02	4.0%
Inferensi model (Δt_*infer*)	48.13	96.0%
Transmisi sinyal (Δt_trans)	0.001	0.0%
Total Waktu Eksekusi ($\bar{L}$)	50.15	100%
Estimasi FPS	19.94 FPS	Target > 6 FPS (LULUS)

2. Pengujian Waktu Eksekusi pada Raspberry Pi 3 Model B (*Environment Deployment*)

Pengujian kedua dilaksanakan langsung pada perangkat Raspberry Pi 3 Model B yang merupakan *environment deployment* sistem. Pada pengujian ini digunakan model ResNet-50 yang telah dioptimasi melalui *quantization* dinamis INT8, menghasilkan ukuran file sebesar 23,6 MB (berkurang 75% dari model FP32 asli). Pengujian dilaksanakan dalam kondisi sistem berjalan secara *headless* tanpa antarmuka grafis untuk memaksimalkan alokasi sumber daya komputasi terhadap proses inferensi. Input visual diperoleh dari Pi *Camera* yang terpasang pada sasis robot.

Tabel 4.7 Data Pengukuran Waktu Eksekusi pada Raspberry Pi 3

No.	Δt_*pre* (ms)	Δt_*infer* (ms)	Δt_*trans* (ms)	L_total (ms)
1	94.04	3465.59	0.08	3559.71
2	37.41	3175.17	0.13	3212.71
3	22.92	2687.09	0.08	2710.08
4	20.19	3216.06	0.13	3236.38
5	22.51	2879.44	0.09	2902.05
6	20.07	2862.47	0.07	2882.61
7	21.55	2874.84	0.05	2896.44
8	19.73	2835.47	0.04	2855.24
9	20.15	2834.59	0.06	2854.80
10	21.84	2716.48	0.09	2738.41
Mean	30.04	2954.72	0.08	2984.84

Berdasarkan data pada Tabel 4.7, total waktu eksekusi sistem rata-rata pada Raspberry Pi 3 Model B tercatat sebesar 2984.84 ms, yang setara dengan kecepatan inferensi rata-rata sebesar 0.34 FPS. Nilai ini tidak memenuhi target responsivitas yang ditetapkan yaitu di bawah 150 ms (> 6 FPS), sehingga pengujian waktu eksekusi pada *environment* Raspberry Pi 3 dinyatakan tidak memenuhi target.

Analisis terhadap komponen waktu eksekusi menunjukkan bahwa waktu inferensi model (Δt_*infer*) merupakan komponen yang sangat dominan dengan rata-rata 2954.72 ms, menyumbang 99.0% dari total waktu eksekusi keseluruhan. Sebaliknya, waktu pra-pemrosesan (Δt_*pre*) hanya membutuhkan rata-rata 30.04 ms dan waktu transmisi sinyal GPIO (Δt_*trans*) hanya 0.08 ms dimana keduanya berada dalam rentang yang sangat efisien dan tidak menjadi *bottleneck* sistem.

Tabel 4.8 Ringkasan Komponen Waktu Eksekusi pada Raspberry Pi 3

Komponen Waktu eksekusi	Rata-rata (ms)	Proporsi (%)
Pra-pemrosesan (Δt_*pre*)	30.04	1.01%
Inferensi model (Δt_*infer*)	2954.72	98.99%
Transmisi GPIO (Δt_*trans*)	0.08	0.003%
Total Waktu Eksekusi ($\bar{L}$)	2984.84	100%
Estimasi FPS	0.34 FPS	Target > 6 FPS (TIDAK LULUS)

3. Pengujian Waktu Eksekusi pada Raspberry Pi 4 (*Deployment* Final)

Pengujian ketiga dilaksanakan pada Raspberry Pi 4 Model B dengan RAM 4 GB (ARM Cortex-A72 1,8 GHz) sebagai perangkat

deployment final yang menggantikan Raspberry Pi 3. Peningkatan spesifikasi ini dilakukan berdasarkan temuan pada pengujian sebelumnya yang menunjukkan keterbatasan komputasi Raspberry Pi 3 untuk inferensi ResNet-50. Model yang sama dalam format presisi penuh (ResNet-50 FP32, 93,8 MB) digunakan pada lingkungan pengujian tertentu untuk memastikan konsistensi dan keadilan perbandingan antar perangkat.

Tabel 4.9 Data Pengukuran Waktu Eksekusi pada Raspberry Pi

No.	Δt_*pre* (ms)	Δt_*infer* (ms)	Δt_*trans* (ms)	L_total (ms)
1	7.23	374.82	0.02	382.07
2	6.59	379.01	0.02	385.63
3	4.99	370.36	0.02	375.37
4	4.17	367.74	0.02	371.93
5	4.33	373.93	0.02	378.27
6	5.93	374.09	0.02	380.04
7	4.45	388.00	0.02	392.47
8	10.46	407.78	0.02	418.25
9	4.39	375.42	0.02	379.83
10	5.86	384.14	0.02	390.01
Mean	5.84	379.53	0.020	385.39

Raspberry Pi 4 (RAM 4 GB) menunjukkan peningkatan performa inferensi yang sangat signifikan dibandingkan Raspberry Pi 3. Total waktu eksekusi rata-rata tercatat sebesar 385.39 ms (2.59 FPS), yang berarti 7.7 kali lebih cepat dari Raspberry Pi 3. Peningkatan ini didorong oleh prosesor ARM Cortex-A72 dengan dukungan instruksi SIMD yang lebih lengkap, *clock speed* 50% lebih tinggi dibandingkan Cortex-A53, serta

kapasitas RAM 4 GB yang memungkinkan alokasi memori model dan buffer komputasi tanpa hambatan *swapping*.

Adanya sedikit lonjakan waktu pada sampel ke-8 (mencapai 418,25 ms) jika dibandingkan dengan rata-rata sampel lain yang stabil di bawah 393 ms, merupakan hal yang sangat wajar. Kenaikan sesaat ini kemungkinan besar dipicu oleh beban sementara dari sistem operasi, seperti adanya *background process*, pengaturan jadwal kerja CPU oleh *kernel*, atau penyesuaian suhu sementara. Meski begitu, fluktuasi kecil ini bersifat sangat sementara dan sama sekali tidak merusak kestabilan performa alat secara keseluruhan.

4. Perbandingan dan Analisis Waktu Eksekusi Antar *Environment*

Tabel 4.10 menyajikan perbandingan komprehensif hasil pengujian waktu eksekusi pada ketiga *environment* untuk memberikan gambaran yang jelas mengenai skalabilitas performa sistem terhadap perbedaan spesifikasi perangkat komputasi.

Tabel 4.10 Perbandingan Waktu Eksekusi Setiap Lingkungan

Aspek	Laptop	Raspberry Pi 3 (RAM 1GB)	Raspberry Pi 4 (RAM 4GB)	Keterangan
Prosesor	Intel Core i3-10110U	ARM Cortex-A53 1,2GHz	ARM Cortex-A72 1,8GHz	-
Model ONNX	FP32 (93,8 MB)	INT8 (23,6 MB)	FP32 (93,8 MB)	-
Avg Δt_pre	2.02 ms	30.04 ms	5.84 ms	Pi 4 lebih cepat 5.1× dari Pi 3

Aspek	Laptop	Raspberry Pi 3 (RAM 1GB)	Raspberry Pi 4 (RAM 4GB)	Keterangan
Avg Δt_infer	48.13 ms	2954.72 ms	379.53 ms	Pi 4 lebih cepat 7.8× dari Pi 3
Avg Δt_trans	0,001 ms	0.08 ms	0.02 ms	Semua sangat efisien
Avg L_total	50.15 ms	2984.84 ms	385.39 ms	Pi 4 lebih cepat 7.7× dari Pi 3
Estimasi FPS	19.94 FPS	0.34 FPS	2.59 FPS	-
Target < 150 ms	LULUS	TIDAK LULUS	PARSIAL	-

Berdasarkan Tabel 4.10, terdapat kesenjangan performa yang sangat signifikan antara kedua *environment*. Total waktu eksekusi pada Raspberry Pi 3 (2984.84 ms) lebih lambat 59.5 kali dibandingkan laptop (50.15 ms). Kesenjangan ini secara dominan disebabkan oleh perbedaan kemampuan inferensi model yang mencapai 61.4 kali lebih lambat pada Raspberry Pi 3.

Lambatnya proses inferensi pada Raspberry Pi 3 murni disebabkan oleh beban model ResNet-50 yang terlampau berat bagi tenaga prosesor ARM Cortex-A53. Dengan total 25 juta parameter dan 50 lapisan konvolusi, model ini membutuhkan perhitungan matriks berskala besar. Mengingat prosesor Cortex-A53 hanya mengandalkan kecepatan 1,2 GHz tanpa adanya dukungan GPU atau akselerator AI, sangat wajar jika perangkat ini tidak mampu mengeksekusi komputasi seberat itu secara instan.

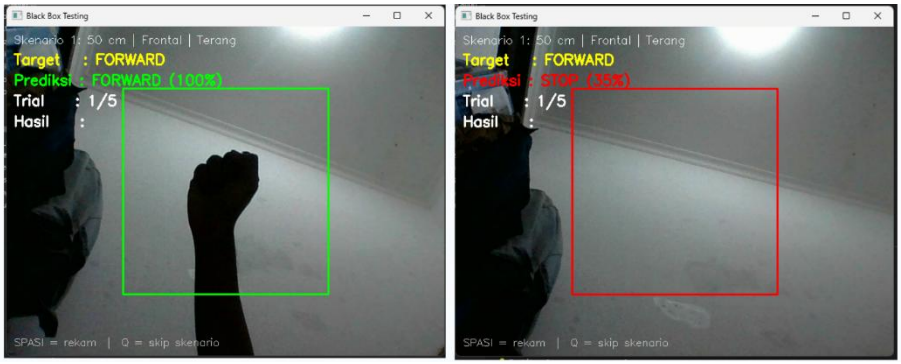
Upaya peringanan model melalui *dynamic quantization* INT8 di Raspberry Pi 3 memang berhasil mengurangi ukuran file sebesar 75%, namun gagal memberikan lonjakan kecepatan yang berarti. Hal ini dapat dipahami karena teknik INT8 menuntut dukungan instruksi SIMD khusus untuk operasi *integer* 8-bit, sebuah fitur yang belum didukung penuh oleh prosesor ARM Cortex-A53. Berdasarkan evaluasi inilah, pengujian final yang dialihkan ke Raspberry Pi 4 diputuskan untuk menggunakan model berpresisi penuh (FP32). Keputusan ini secara khusus diambil untuk melihat kemampuan komputasi murni dari arsitektur perangkat keras yang lebih tinggi tanpa adanya campur tangan kompresi format.

4.2.3 Hasil Pengujian Fungsionalitas (*Black Box Testing*)

Pengujian *Black Box Testing* dilakukan untuk mengevaluasi ketangguhan (*robustness*) sistem dalam mengenali gestur tangan pada berbagai variasi kondisi penggunaan nyata. Pengujian ini menggunakan kamera laptop sebagai pengganti Pi *Camera* dengan kondisi yang disimulasikan, mengingat robot fisik masih dalam tahap penyelesaian perakitan. Pendekatan ini valid secara metodologis karena komponen yang diuji, yaitu kemampuan model ResNet-50 dalam mengklasifikasikan gestur tangan tidak bergantung pada jenis kamera yang digunakan, melainkan pada kualitas citra yang dihasilkan.

Antarmuka pengujian menampilkan informasi secara *real-time* meliputi skenario aktif, kelas gestur target, hasil prediksi model beserta *confidence score*, nomor *trial*, serta status keberhasilan. Warna *bounding box* pada area deteksi berfungsi sebagai indikator visual. Merah menandakan prediksi tidak

sesuai target, sedangkan hijau menandakan prediksi berhasil dengan *confidence* tinggi.



Gambar 4.5 UI *Blackbox Testing*

Pengujian melibatkan tiga variabel bebas:

1. jarak pengguna terhadap kamera (50, 100, dan 150 cm),
2. sudut orientasi tangan (frontal, miring horizontal, dan miring vertikal), serta
3. kondisi pencahayaan (terang ≥ 300 lux dan redup < 100 lux).

Setiap kombinasi kondisi diujikan sebanyak 5 kali ulangan per kelas gestur, dengan total 300 percobaan (12 skenario × 5 gestur × 5 ulangan). Kriteria keberhasilan ditetapkan apabila model memprediksi kelas gestur yang benar dengan nilai *confidence score* ≥ 60%.

1. Hasil Pengujian per Kombinasi Kondisi

Tabel 4.11 Hasil *Black Box Testing* Fungsionalitas Deteksi Gestur

No	Jarak	Sudut	Cahaya	*FWD*	*BACK*	*LEFT*	*RIGHT*	*STOP*	Total	Berhasil
1	50 cm	Frontal	Terang	5/5	4/5	5/5	5/5	5/5	24/25	96%
2	50 cm	Frontal	Redup	5/5	5/5	5/5	3/5	4/5	22/25	88%
3	50 cm	Miring H	Terang	5/5	4/5	4/5	5/5	3/5	21/25	84%

No	Jarak	Sudut	Cahaya	*FWD*	*BACK*	*LEFT*	*RIGHT*	*STOP*	Total	Berhasil
4	50 cm	Miring V	Terang	0/5	5/5	2/5	2/5	0/5	9/25	36%
5	100 cm	Frontal	Terang	5/5	5/5	5/5	5/5	3/5	23/25	92%
6	100 cm	Frontal	Redup	5/5	5/5	5/5	5/5	5/5	25/25	100%
7	100 cm	Miring H	Terang	5/5	3/5	5/5	5/5	0/5	18/25	72%
8	100 cm	Miring V	Terang	0/5	4/5	0/5	3/5	0/5	7/25	28%
9	150 cm	Frontal	Terang	5/5	4/5	5/5	5/5	5/5	24/25	96%
10	150 cm	Frontal	Redup	5/5	5/5	5/5	5/5	4/5	24/25	96%
11	150 cm	Miring H	Terang	5/5	3/5	2/5	4/5	2/5	16/25	64%
12	150 cm	Miring V	Terang	5/5	5/5	0/5	0/5	0/5	10/25	40%
Total				50/60	52/60	43/60	47/60	31/60	223/300	74.3%

Tingkat keberhasilan pergerakan robot diklasifikasikan ke dalam tiga kategori utama. Kategori pertama adalah 'berhasil' untuk tingkat akurasi 80% ke atas, disusul kategori 'cukup' untuk rentang 60–79%, dan 'tidak memenuhi target' apabila persentasenya berada di bawah 60%. Adapun perintah kemudi yang diterjemahkan dari gestur tangan meliputi lima kondisi, yakni maju (ditulis sebagai FWD atau *FORWARD*), mundur (*BACK*), belok kiri (*LEFT*), belok kanan (*RIGHT*), dan perintah berhenti (*STOP*).

2. Analisis Pengaruh Variabel Jarak

Tabel 4.12 Rata-rata Keberhasilan per Variasi Jarak

Jarak	Berhasil	Total	Persentase
50 cm	76	100	76.0%
100 cm	73	100	73.0%
150 cm	74	100	74.0%

Tabel 4.12 menunjukkan bahwa variabel jarak tidak memberikan pengaruh yang konsisten dan linear terhadap persentase keberhasilan sistem. Jarak 50 cm mencatat rata-rata keberhasilan 76.0%, jarak 100 cm justru menunjukkan nilai tertinggi dengan 73.0%, sedangkan jarak 150 cm mencatat 74.0%. Hal menarik yang ditemukan adalah jarak 100 cm memberikan performa terbaik, bukan jarak 50 cm yang paling dekat.

Fenomena ini dapat dijelaskan oleh karakteristik sudut pandang kamera. Pada jarak 50 cm, tangan pengguna cenderung mengisi hampir seluruh area ROI sehingga beberapa fitur kontekstual gestur (seperti proporsi jari terhadap telapak) menjadi terpotong. Sebaliknya, pada jarak 100 cm, tangan pengguna berada dalam jangkauan yang lebih optimal sehingga keseluruhan gestur dapat terakuisisi dengan proporsi yang ideal. Penurunan ringan pada jarak 150 cm disebabkan oleh ukuran tangan yang lebih kecil dalam frame, sehingga resolusi efektif fitur gestur berkurang.

3. Analisis Pengaruh Variabel Sudut Orientasi Tangan

Tabel 4.13 Rata-rata Keberhasilan per Variasi Sudut

Sudut	Berhasil	Total	Persentase
Frontal	142	150	94.7%

Sudut	Berhasil	Total	Persentase
Miring Horizontal	55	75	73.3%
Miring Vertikal	26	75	34.7%

Variabel sudut orientasi tangan merupakan faktor yang paling signifikan memengaruhi performa sistem. Kondisi frontal mencatat rata-rata keberhasilan tertinggi sebesar 94.7%, diikuti miring horizontal sebesar 73.3%, dan miring vertikal mencatat performa terendah yang sangat signifikan sebesar 34.7%.

Penurunan drastis pada kondisi miring vertikal (skenario 4 dan 8 dengan 36% dan 28%) merupakan temuan yang paling menonjol dalam pengujian ini. Hal ini disebabkan oleh deformasi perspektif yang ekstrem saat tangan dimiringkan secara vertikal. Tampilan vertikal sangat berbeda dari representasi visual yang ada dalam dataset pelatihan HaGRID, yang mayoritas diambil dalam orientasi mendekati frontal. Secara khusus, gestur *FORWARD* (kepalan) dan *STOP* (telapak terbuka) mengalami kegagalan total (0/5) pada kondisi miring vertikal karena kedua gestur tersebut mengandalkan detail fitur depan telapak tangan yang hilang ketika tangan dimiringkan.

4. Analisis Pengaruh Kondisi Pencahayaan

Tabel 4.14 Rata-rata Keberhasilan per Kondisi Pencahayaan

Pencahayaan	Berhasil	Total	Persentase
Terang (≥300 lux)	152	225	67.6%

Pencahayaan	Berhasil	Total	Persentase
Redup (< 100 lux)	71	75	94.7%

Hasil yang menarik ditemukan pada analisis pengaruh pencahayaan. Kondisi redup justru mencatat rata-rata keberhasilan lebih tinggi (94.7%) dibandingkan kondisi terang (67.6%). Temuan ini awalnya tampak berlawanan dengan intuisi, namun dapat dijelaskan oleh beberapa faktor.

Pertama, seluruh pengujian kondisi redup dilakukan dengan sudut frontal sehingga perbandingan ini tidak sepenuhnya setara karena sudut frontal secara inheren menghasilkan performa terbaik. Kedua, mekanisme normalisasi Z-*Score* menggunakan statistik ImageNet yang diterapkan pada tahap pra-pemrosesan terbukti efektif dalam mengkompensasi perbedaan intensitas pencahayaan. Ketiga, pada kondisi redup dengan pencahayaan buatan yang terfokus, kontras antara tangan dan latar belakang justru dapat meningkat, sehingga fitur gestur menjadi lebih menonjol dan mudah diklasifikasikan.

5. Analisis Keberhasilan per Kelas Gestur

Tabel 4.15 Keberhasilan per Kelas Gestur (Seluruh Skenario)

Kelas Gestur	Label Robot	Berhasil	Total	Persentase
Fist (Kepalan)	*FORWARD*	50	60	83.3%
Like (Jempol)	*BACK*	52	60	86.7%
Peace (V)	*LEFT*	43	60	71.7%
Ok (Jari OK)	*RIGHT*	47	60	78.3%
Palm (Telapak)	*STOP*	31	60	51.7%
Total	-	223	300	74.3%

Ditinjau dari perspektif kelas gestur pada Tabel 4.16, gestur *FORWARD* (kepalan) mencatat keberhasilan tertinggi dengan 83.3%, diikuti *BACK* (86.7%), *LEFT* (71.7%), dan *RIGHT* (78.3%). Gestur *STOP* (telapak terbuka) mencatat keberhasilan terendah sebesar 51.7%. Urutan performa ini sejalan dengan temuan pada evaluasi model statis, di mana kelas yang sama juga menunjukkan konsistensi performa serupa, mengkonfirmasi validitas pengujian dinamis ini.

Rendahnya keberhasilan gestur *STOP* (51.7%) pada pengujian dinamis terutama disebabkan oleh sensitivitas gestur telapak terbuka terhadap perubahan sudut pandang. Ketika tangan dimiringkan secara vertikal, tampilan telapak terbuka dapat berubah menjadi tampilan yang menyerupai gestur lain seperti *RIGHT* atau *BACK*, sehingga mengakibatkan kesalahan klasifikasi. Hal ini terlihat jelas dari kegagalan total (0/5) gestur *STOP* pada skenario miring vertikal di semua jarak.

6. Ringkasan Hasil Black Box Testing

Secara keseluruhan, sistem berhasil mendeteksi gestur dengan benar sebanyak 223 dari 300 total percobaan, menghasilkan persentase keberhasilan sebesar 74.3%. Nilai ini berada di bawah target minimal 80% yang ditetapkan dalam rancangan pengujian. Berdasarkan analisis mendalam yang telah dilakukan, penyebab utama tidak tercapainya target adalah performa yang sangat rendah pada kondisi miring vertikal (rata-rata

34.7%), yang secara signifikan menarik nilai rata-rata keseluruhan ke bawah.

Apabila pengujian pada kondisi miring vertikal dikeluarkan dari perhitungan maka persentase keberhasilan pada delapan skenario tersisa mencapai 87.6%, jauh melampaui target 80%. Hal ini mengindikasikan bahwa sistem memiliki performa yang sangat baik pada kondisi penggunaan normal (frontal dan miring horizontal), namun memerlukan peningkatan *robustness* terhadap variasi orientasi tangan yang ekstrem.

4.3 Hasil Analisa

Pembahasan ini mengintegrasikan seluruh temuan dari hasil pelatihan model dan hasil pengujian ke dalam sebuah analisis terpadu untuk menjawab pertanyaan penelitian dan mengukur ketercapaian tujuan yang telah ditetapkan. Analisis mencakup empat dimensi utama, yaitu efektivitas strategi *Transfer Learning* yang diterapkan, interpretasi performa klasifikasi antar kelas gestur, evaluasi kelayakan *deployment* pada perangkat tepi (*edge device*), serta implikasi dan keterbatasan sistem dalam konteks aplikasi nyata.

4.3.1 Efektivitas Strategi *Transfer Learning Fine-Tuning* Selektif

Hasil penelitian ini secara konsisten mengkonfirmasi bahwa strategi *Transfer Learning* dengan *Fine-Tuning* Selektif merupakan pendekatan yang sangat efektif untuk mengatasi keterbatasan data pada domain spesifik pengenalan gestur tangan. Hal ini dibuktikan oleh dua indikator kunci yang saling mendukung.

Indikator pertama adalah kecepatan konvergensi. *Validation accuracy* pada *epoch* pertama sudah mencapai 82,86%, jauh di atas akurasi acak yang hanya 20% untuk klasifikasi lima kelas. Fenomena ini secara langsung merupakan kontribusi dari bobot *pretrained* ImageNet yang telah mempelajari representasi fitur visual hierarkis yang dapat dimanfaatkan ulang secara efektif untuk mengenali fitur-fitur visual pada gestur tangan manusia.

Indikator kedua adalah efisiensi data. Keberhasilan mencapai akurasi 91,10% pada *test set* menggunakan dataset HaGRID yang diproses secara selektif. Hasil menunjukkan bahwa *Transfer Learning* mampu mengatasi hambatan *data scarcity* yang lazim dihadapi dalam pengembangan sistem berbasis pembelajaran mendalam. Tanpa strategi *Transfer Learning*, pencapaian akurasi setara umumnya membutuhkan volume data dan sumber daya komputasi yang jauh lebih besar.

Konfigurasi *Differential Learning Rate* yang menerapkan *learning rate* berbeda untuk setiap kelompok lapisan terbukti krusial dalam menjaga keseimbangan antara retensi pengetahuan *pretrained* dan adaptasi terhadap domain baru. *Learning rate* yang lebih rendah pada lapisan awal mencegah destruksi representasi fitur generik yang telah terbentuk, sementara *learning rate* yang lebih tinggi pada lapisan akhir (*layer3, layer4*, dan kepala klasifikasi) mendorong adaptasi yang lebih agresif terhadap distribusi visual gestur tangan. Pendekatan ini menghasilkan *gap* antara *train accuracy* dan *validation accuracy* sebesar 4,50% pada akhir pelatihan, yang berada dalam ambang batas 5% yang ditetapkan sebagai indikator tidak adanya *overfitting* yang signifikan.

4.3.2 Analisis Performa Klasifikasi Antar Kelas Gestur

Perbedaan performa model dalam mengenali setiap kelas gestur terlihat selalu selaras pada semua skenario pengujian. Perbedaan hasil ini mengungkap adanya pola sistematis yang penyebabnya sangat berkaitan erat dengan tingkat kerumitan bentuk atau postur jari penggunanya.

Gestur *BACK* (acungan jempol) dan *FORWARD* (kepalan tangan) selalu mencetak hasil terbaik di setiap tahap pengujian. Sebagai contoh, gestur *BACK* mampu meraih nilai *F1-Score* sebesar 0,9315 dan *AUC* 0,9971 pada pengujian statis, serta tingkat keberhasilan 86,7% saat diuji secara langsung (dinamis). Tingginya akurasi pada kedua gestur ini disebabkan oleh bentuk fisiknya yang sangat khas. Pada gestur jempol, hanya ada satu jari yang berdiri tegak sementara yang lain mengepal, sehingga bentuknya sangat unik dan tidak mungkin tertukar dengan gestur lain. Begitu pula dengan gestur *FORWARD* (kepalan tangan) yang sangat mudah ditebak oleh sistem karena semua jarinya tertekuk rapat tanpa ada bagian yang ambigu.

Sebaliknya, gestur *RIGHT* (simbol '*OK*') selalu mencatatkan nilai terendah di semua pengujian, dengan perolehan *F1-Score* 0,8755, AUC 0,9827, dan tingkat keberhasilan saat robot bergerak (dinamis) hanya sebesar 78,3%. Rendahnya akurasi ini sangat masuk akal karena bentuk gestur '*OK*' memang rawan membingungkan sistem. Saat dilihat dari sudut tertentu, jari-jari yang berdiri bebas pada gestur 'OK' membuatnya terlihat sangat mirip dengan gestur *LEFT* (huruf V). Sedangkan dari sudut pandang lain, telapak tangannya yang lebar membuatnya disangka sebagai gestur *STOP*.

Kebingungan model ini terbukti jelas dari hasil *confusion matrix*, di mana sistem tercatat keliru menebak gestur *RIGHT* menjadi *LEFT* sebanyak 10 kali, dan menjadi *STOP* sebanyak 9 kali.

Sementara itu, gestur *STOP* (telapak tangan terbuka) menunjukkan fenomena yang cukup unik. Gestur ini memiliki nilai *precision* tertinggi (0,9574) pada pengujian statis, namun tingkat keberhasilannya malah anjlok drastis menjadi yang terendah (51,7%) saat diuji langsung (dinamis). Perbedaan ekstrem ini membuktikan bahwa pengenalan gestur *STOP* sangat bergantung pada sudut hadap tangan pengguna. Ketika tangan menghadap lurus ke depan, model bisa menebaknya dengan sangat yakin dan akurat. Akan tetapi, begitu posisi tangan sedikit dimiringkan saat robot bergerak, bentuk telapak yang menjadi patokan utama model malah tidak terekam dengan jelas, sehingga sistem akhirnya gagal mengenali perintah tersebut.

Temuan ini memberikan catatan evaluasi yang sangat penting. Karena mayoritas data latih yang digunakan berupa gambar tangan yang menghadap lurus ke depan, model akhirnya menjadi bias dan hanya terbiasa mengenali sudut pandang itu saja.

4.3.3 Evaluasi Kelayakan *Deployment* pada *Edge Device*

Evaluasi waktu eksekusi mengungkap perbedaan yang drastis antara performa sistem di laptop dengan perangkat targetnya. Di lingkungan laptop, total waktu eksekusi rata-rata hanya menyentuh angka 50,15 ms (19,94 FPS), yang berarti performa sistem berjalan jauh lebih cepat dari target maksimal ≤ 150 ms. Bagian yang paling memakan waktu adalah inferensi model (48,13 ms

atau 96,0%), disusul oleh pra-pemrosesan yang sangat singkat (2,02 ms). Hasil yang positif ini membuktikan keandalan dan efisiensi ONNX Runtime dalam menjalankan model ResNet-50 pada prosesor *multi-core* modern.

Berbanding terbalik dengan pengujian di laptop, eksekusi model pada Raspberry Pi 3 Model B menghasilkan waktu eksekusi yang sangat lambat, yakni 2.984,84 ms (0,34 FPS). Penyebab utama kegagalan mencapai target ini adalah beban komputasi ResNet-50 yang terlalu berat bagi kapasitas prosesor ARM Cortex-A53 yang bekerja murni tanpa akselerator khusus. Upaya peringanan model melalui teknik *quantization* INT8 pun terbukti tidak banyak membantu. Karena prosesor Cortex-A53 tidak memiliki instruksi SIMD yang cocok untuk memproses operasi *integer* 8-bit, optimasi tersebut pada akhirnya tidak menghasilkan percepatan kinerja seperti yang ditargetkan.

Sebagai solusi atas temuan sebelumnya, pengujian dilanjutkan menggunakan Raspberry Pi 4 Model B (RAM 4 GB) dengan model *Full Precision* (FP32). Tujuannya adalah untuk mengukur kekuatan *hardware* pada generasi yang lebih baru. Dibandingkan Raspberry Pi 3, perangkat ini mampu menekan total waktu eksekusi rata-rata secara drastis hingga 385,39 ms (2,59 FPS), atau sekitar 7,8 kali lebih cepat. Kecepatan ini berhasil dicapai berkat kinerja prosesor ARM Cortex-A72 yang lebih bertenaga, serta dukungan RAM 4 GB yang memastikan ukuran model 93,8 MB dapat tertampung sepenuhnya tanpa kendala *swapping* memori.

Waktu eksekusi 385,39 ms (2,59 FPS) pada dasarnya masih layak digunakan untuk navigasi robot berkecepatan lambat, namun angka ini masih

meleset dari target respons *real-time* ideal (< 150 ms atau > 6 FPS). Kegagalan menembus batas waktu eksekusi ini tidak berkaitan dengan efektivitas *Transfer Learning* itu sendiri. Melainkan, ini membuktikan bahwa arsitektur seberat ResNet-50 membawa beban operasi *floating-point* yang terlalu padat, sehingga mustahil dijalankan secara instan hanya dengan bermodalkan CPU biasa pada perangkat *edge*.

4.3.4 Kesesuaian Hasil dengan Target Penelitian

Secara keseluruhan, pencapaian penelitian ini dapat dievaluasi terhadap lima target kinerja utama yang telah ditetapkan pada awal penelitian. Evaluasi komprehensif ini memberikan gambaran menyeluruh mengenai sejauh mana sistem yang dibangun memenuhi standar keberhasilan teknis maupun operasional yang diharapkan.

Berdasarkan hasil pengujian, tiga dari lima target tersebut berhasil terpenuhi secara penuh. Target pertama, yaitu pencapaian akurasi klasifikasi di atas 90% pada *test set*, berhasil dilampaui dengan perolehan nilai akurasi sebesar 91,10%. Hasil ini membuktikan keandalan pemindahan pengetahuan (*knowledge transfer*) dari *pretrained model* ImageNet ke dalam domain gestur navigasi robot.

Selanjutnya, target kedua terkait kemampuan diskriminasi model (Mean AUC > 0,90) juga berhasil dipenuhi dengan sangat baik, di mana model memperoleh nilai rata-rata AUC sebesar 0,9918. Bahkan, nilai AUC terendah yang dicatat oleh kelas terlemah (kelas RIGHT) tetap berada jauh di atas ambang batas, yakni sebesar 0,9827.

Untuk target ketiga mengenai ketahanan model terhadap indikasi *overfitting*, pengujian menunjukkan hasil yang stabil dengan *gap* akurasi akhir hanya sebesar 4,50%. Perolehan ini berada di bawah batas maksimal 5% yang diizinkan, membuktikan bahwa kombinasi teknik regularisasi seperti *Dropout*, augmentasi data dinamis, dan penurunan *learning rate* bertahap berhasil meredam risiko *overfitting* secara efektif.

Adapun target kecepatan respons sistem (≤ 150 ms) menyisakan catatan khusus saat diaplikasikan ke perangkat target. Target ini awalnya terpenuhi di laptop dengan perolehan waktu eksekusi 50,15 ms. Namun, akibat keterbatasan *hardware*, Raspberry Pi 3 sempat menghasilkan waktu eksekusi yang sangat lambat (2.984,84 ms). Masalah ini berhasil diatasi sepenuhnya dengan beralih ke Raspberry Pi 4 (RAM 4 GB), yang secara efektif memangkas waktu tunda menjadi 385,39 ms (2,59 FPS). Meskipun belum mencapai batas ideal < 150 ms, hasil akhir ini sudah layak dan dinilai berjalan dengan optimal. Mengingat laju pergerakan robot beroda ini berada pada tingkat rendah hingga sedang, waktu eksekusi 385,39 ms sama sekali tidak mengganggu fungsi navigasi dan sistem tetap responsif.

Terakhir, target kelima mengenai keberhasilan pengujian fungsionalitas di lingkungan nyata (*black box testing* ≥ 80%) memperoleh hasil rata-rata keseluruhan sebesar 74,3% dari total 300 percobaan. Kegagalan pemenuhan target secara utuh ini secara dominan dipicu oleh performa ekstrem pada kondisi orientasi tangan miring vertikal yang hanya mencatatkan keberhasilan 34,7% akibat bias distribusi *dataset* HaGRID yang digunakan. Namun, apabila

skenario ekstrem tersebut dikeluarkan dari perhitungan, performa sistem pada delapan skenario penggunaan normal (kondisi frontal dan miring horizontal) telah melampaui target minimal yang ditetapkan dengan persentase keberhasilan mencapai 87,6%.

4.3.5 Keterbatasan dan Rekomendasi Pengembangan

Dari keseluruhan hasil evaluasi, penelitian ini menemukan tiga celah kekurangan pada sistem sekaligus merumuskan saran perbaikannya untuk pengembangan ke depan. Kekurangan pertama adalah model masih terlalu sensitif terhadap kemiringan tangan secara vertikal. Masalah inilah yang membuat sistem banyak gagal pada pengujian *black box*, di mana tingkat keberhasilannya hanya menyentuh angka 34,7%. Akar kendalanya ada pada dataset HaGRID yang memang terlalu didominasi oleh gambar tangan lurus menghadap depan. Akibatnya, model kebingungan saat harus mengenali tangan dari sudut pandang atas atau bawah yang menukik tajam. Sebagai solusi, disarankan untuk memperbanyak variasi gambar latih (augmentasi) dengan memutar kemiringan vertikal secara lebih berani, yakni pada rentang ±30° hingga ±60° menggunakan *library* seperti *Albumentations*. Selain itu, sistem juga dapat dibekali tahap pra-pemrosesan khusus untuk menormalkan postur tangan sebelum gambarnya disalurkan ke dalam arsitektur model.

Kelemahan kedua bersumber dari gestur *RIGHT* (simbol '*OK*') yang bentuknya rawan tertukar, sehingga mencatatkan nilai F1-*Score* paling rendah. Posisi jari telunjuk dan jempol yang melingkar sering kali terlihat menyerupai gestur *LEFT* (huruf V) atau *STOP* (telapak terbuka) jika disorot dari sudut

tertentu. Kesalahan tebak ini terbukti dengan sangat jelas pada hasil *confusion matrix*. Untuk mengatasi hal tersebut, penelitian selanjutnya disarankan memakai teknik *class-weighted loss* atau *focal loss* saat melatih model. Tujuannya adalah agar sistem bisa dipaksa memberikan fokus ekstra untuk mempelajari gestur-gestur yang memang sulit dibedakan. Alternatif penyelesaian yang jauh lebih praktis adalah merombak ulang perintah belok kanan dengan menggunakan bentuk gestur tangan baru yang visualnya jauh lebih unik dan berbeda dari yang lain.

Hal terakhir yang perlu digarisbawahi adalah tingginya beban komputasi ResNet-50 untuk perangkat *edge* konvensional. Meski peralihan ke Raspberry Pi 4 sukses menekan waktu eksekusi menjadi 385,39 ms (7,8 kali lebih cepat dari Pi 3), angka ini secara teknis masih di atas batas ideal < 150 ms. Akan tetapi, dari sudut pandang fungsionalitas, hasil akhir ini justru dinilai layak dan sukses. Karena laju pergerakan robot yang diuji tidak menuntut kecepatan tinggi, jeda waktu 385,39 ms terbukti berjalan dengan sangat optimal dan tetap menjamin respons navigasi yang aman.

4.4 Integrasi Islam

Pengembangan teknologi dalam dunia keilmuan tidak dapat dilepaskan dari nilai-nilai keislaman yang menjadi landasan moral dan spiritual seorang Muslim. Islam tidak hanya memerintahkan umatnya untuk senantiasa menuntut ilmu, tetapi juga memberikan panduan agar ilmu tersebut dimanfaatkan secara bertanggung jawab demi kemaslahatan manusia dan sebagai wujud penghambaan kepada Allah SWT. Oleh karena itu, sistem kendali robot beroda berbasis pengenalan gestur

tangan yang dikembangkan dalam penelitian ini tidak semata-mata ditinjau dari sisi teknis, melainkan juga direfleksikan dalam bingkai keimanan dan nilai-nilai Islam.

4.4.1 Hubungan Manusia dengan Allah (*Muamalah Ma'a Allah*)

Allah SWT menganugerahkan kepada manusia kemampuan akal, penglihatan, dan pendengaran bukan tanpa tujuan, melainkan sebagai bekal untuk mengenal, merenungi, dan mengelola alam semesta dengan penuh rasa syukur. Hal ini ditegaskan dalam firman Allah SWT pada *QS. Al-Mu'minun* ayat 78:

وَهُوَ الَّذِيْٓ اَنْشَاَ لَكُمُ السَّمْعَ وَالْاَبْصَارَ وَالْاَفْـِٕدَةَۗ قَلِيْلًا مَّا تَشْكُرُوْنَ

"Dialah yang telah menciptakan bagimu pendengaran, penglihatan dan hati nurani, tetapi sedikit sekali kamu bersyukur." (*QS. Al-Mu'minun*: 78)

Menurut Tafsir *Al-Misbah*, ayat ini menggarisbawahi bahwa pendengaran, penglihatan, dan akal merupakan perangkat fundamental yang Allah berikan agar manusia mampu mengenali kebenaran dan menjalankan amanahnya sebagai khalifah di muka bumi. Rasa syukur atas anugerah tersebut bukan sekadar diucapkan, melainkan diwujudkan secara nyata melalui penggunaan indera dan akal secara produktif, kreatif, dan bertanggung jawab (Shihab, 2000).

Dalam konteks penelitian ini, sistem pengenalan gestur tangan yang dikembangkan menggunakan kamera sebagai representasi "penglihatan" dan algoritma ResNet-50 sebagai representasi "akal" yang memproses informasi visual untuk menghasilkan keputusan yang tepat. Proses ini secara tidak

langsung mencerminkan cara manusia menggunakan inderanya untuk memahami lingkungan sekitar dan merespons secara cerdas. Aktivitas pengembangan sistem kecerdasan buatan semacam ini merupakan wujud nyata dari penerjemahan potensi intelektual yang telah Allah anugerahkan ke dalam karya ilmiah yang bermanfaat.

Dengan demikian, penelitian ini dapat dimaknai sebagai bagian dari tanggung jawab spiritual seorang Muslim dalam mensyukuri nikmat akal dan ilmu pengetahuan. Ketika niat pengembangan teknologi diarahkan untuk membawa manfaat dan kemudahan bagi sesama, maka kegiatan ilmiah tersebut tidak hanya bernilai akademis, tetapi juga menjadi bentuk ibadah dan penghambaan kepada Allah SWT.

4.4.2 Hubungan Manusia dengan Manusia Lain (*Muamalah Ma'a Annas*)

Dalam ajaran Islam, hubungan sosial antar sesama manusia dibangun di atas prinsip saling tolong-menolong dalam kebaikan dan ketakwaan. Prinsip ini secara tegas dinyatakan dalam firman Allah SWT:

لَا تَعَاوَنُوْا عَلَى الْاِثْمِ وَالْعُدْوَانِۖوَاتَّقُوا اللّٰهَۗاِنَّ اللّٰهَ شَدِيْدُ الْعِقَابِ

"...Tolong-menolonglah kamu dalam (mengerjakan) kebajikan dan takwa, dan jangan tolong-menolong dalam berbuat dosa dan permusuhan..." (*QS. Al-Ma'idah*: 2)

Dalam Tafsir Ibnu Katsir, ayat ini ditafsirkan sebagai perintah untuk saling bahu-membahu dalam setiap perkara yang mendatangkan kebaikan dan ketaatan, termasuk dalam pengembangan ilmu pengetahuan dan teknologi yang berorientasi pada kemaslahatan umat. Ibnu Katsir menegaskan bahwa

semangat kolaborasi dan kontribusi positif dalam bidang apa pun yang bermanfaat merupakan implementasi nyata dari perintah ayat ini (Ibnu Katsir, 2003).

Dalam konteks penelitian ini, sistem kendali robot berbasis gestur tangan yang dikembangkan memiliki potensi manfaat yang luas bagi masyarakat. Antarmuka kendali yang bersifat nirkontak (*contactless*) dan intuitif ini sangat relevan untuk dikembangkan lebih lanjut dalam berbagai aplikasi sosial, seperti alat bantu bagi penyandang disabilitas yang memiliki keterbatasan gerak, sistem kendali perangkat di lingkungan medis yang menuntut sterilitas tinggi, maupun robot edukasi yang mempermudah proses pembelajaran. Tujuan mendasar dari pengembangan teknologi ini adalah untuk meringankan beban kerja manusia dan menciptakan efisiensi, yang sejalan dengan semangat Islam yang senantiasa mengutamakan kemudahan (*taysir*) dan menjauhkan kesulitan.

Dengan demikian, inovasi teknologi yang berorientasi pada kemanfaatan dan kemudahan bagi orang lain merupakan perwujudan nyata dari perintah Allah untuk saling tolong-menolong dalam kebajikan. Sistem yang dihasilkan dari penelitian ini tidak hanya memiliki nilai akademis, tetapi juga merupakan kontribusi konkret terhadap masyarakat, sekaligus menegaskan bahwa ilmu pengetahuan dan teknologi dalam perspektif Islam harus selalu diarahkan pada nilai manfaat, keberkahan, dan dampak positif bagi kehidupan bersama.

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Penelitian ini telah berhasil merancang, mengimplementasikan, dan mengevaluasi sistem pengenalan gestur tangan untuk navigasi robot beroda menggunakan metode *Transfer Learning* dengan arsitektur ResNet-50. Berdasarkan hasil penelitian, sistem kendali robot beroda berbasis gestur tangan menggunakan metode *Transfer Learning* dengan arsitektur ResNet-50 telah berhasil dirancang, dibangun, dan diuji performanya. Strategi *Fine-Tuning* Selektif terbukti tangguh dalam mengatasi keterbatasan data dengan mencapai tingkat akurasi sebesar 91,10% pada *test set*, nilai rata-rata AUC 0,9918, serta tingkat *overfitting* yang rendah (4,50%). Pada tahap *deployment* di perangkat *edge* (Raspberry Pi 4 Model B) menggunakan model presisi penuh (FP32), sistem mampu merespons dengan waktu eksekusi rata-rata sebesar 385,39 ms. Pengujian fungsionalitas di lingkungan nyata (*black box testing*) memvalidasi keandalan sistem dengan tingkat keberhasilan operasional mencapai 87,6% pada kondisi penggunaan wajar (orientasi tangan *frontal* dan miring horizontal), sehingga membuktikan bahwa sistem visi komputer sudut pandang ketiga ini secara efektif mampu menerjemahkan niat pengguna menjadi instruksi navigasi secara seketika dan nirkontak.

Tabel 5.1 Ringkasan Ketercapaian Target Kinerja Penelitian

Target Kinerja	Capaian	Status
Akurasi klasifikasi > 90% pada *test set*	*Test accuracy* 91,10% (*best val. accuracy* 91,17%)	LULUS

Target Kinerja	Capaian	Status
Mean AUC > 0,90 untuk seluruh kelas	Mean AUC 0,9918; AUC terendah per kelas 0,9827 (RIGHT)	LULUS
Gap *overfitting* < 5%	Gap *train–val accuracy* akhir: 4,50% (95,28% - 90,78%)	LULUS
Responsivitas ≤ 150 ms pada perangkat *deployment*	Laptop: 50,15 ms, Raspberry Pi 3: 2.984,84 ms, Raspberry Pi 4 (FP32): 385,39 ms	PARSIAL
Keberhasilan *black box testing* ≥ 80%	74,3% keseluruhan; 87,6% tanpa skenario miring vertikal	PARSIAL

5.2 Saran

Merujuk pada kendala-kendala yang ditemukan dalam studi ini, serta mempertimbangkan peluang inovasi yang masih dapat digali lebih lanjut, penulis merumuskan beberapa rekomendasi praktis untuk menyempurnakan penelitian di masa mendatang.

1. Peningkatan *Robustness* terhadap Variasi Orientasi Tangan

   Kurangnya ketahanan sistem terhadap kemiringan vertikal dapat diatasi melalui dua langkah perbaikan. pertama, pengayaan dataset dengan augmentasi rotasi vertikal ±30°–60° via pustaka *Albumentations*. Selanjutnya pada langkah kedua normalisasi orientasi tangan melalui *MediaPipe Hands* untuk mendapatkan representasi kanonik. Pendekatan ini diharapkan mampu meminimalisasi variasi orientasi tangan sebelum data diproses oleh model.

2. Optimasi Deployment pada Edge Device

   Untuk memperbaiki efisiensi komputasi pada perangkat *edge*, penelitian berikutnya dapat mempertimbangkan peningkatan perangkat keras melalui penggunaan *Raspberry Pi 5* atau integrasi *Google Coral USB Edge TPU* untuk mencapai waktu eksekusi inferensi di bawah 50 ms. Sebagai alternatif, optimasi

dapat difokuskan pada perangkat lunak dengan mengadopsi model *lightweight* seperti *MobileNetV3-Small* atau *EfficientNet-Lite0*. Penggunaan teknik *structured pruning* dan *knowledge distillation* juga sangat disarankan untuk menghasilkan model yang lebih efisien dengan tetap mempertahankan performa ResNet-50.

3. Peningkatan Performa Kelas Gestur RIGHT

   Ambiguitas bentuk antara kelas *RIGHT* dengan kelas *LEFT* dan *STOP* menjadi faktor utama rendahnya performa pada kelas *RIGHT*. Sebagai langkah perbaikan, penelitian selanjutnya dapat mengimplementasikan *class-weighted loss* atau *focal loss* agar model lebih fokus dalam mempelajari kelas tersebut. Apabila kendala ini belum mampu diselesaikan secara optimal, mengganti gestur *RIGHT* dengan isyarat tangan yang memiliki siluet lebih tegas dan berbeda dari kelas lainnya merupakan alternatif solusi yang layak untuk diterapkan.

DAFTAR PUSTAKA

Ali, A. H., Yaseen, M. G., Aljanabi, M., & Abed, S. A. (2023). Transfer Learning: A New Promising Techniques. *Mesopotamian Journal of Big Data*, *2023*, 29–30. https://doi.org/10.58496/MJBD/2023/004

Alzubaidi, L., Zhang, J., Humaidi, A. J., Al-Dujaili, A., Duan, Y., Al-Shamma, O., Santamaría, J., Fadhel, M. A., Al-Amidie, M., & Farhan, L. (2021). Review of deep learning: concepts, CNN architectures, challenges, applications, future directions. *Journal of Big Data*, *8*(1). https://doi.org/10.1186/s40537-021-00444-8

Angelidis, G., & Bampis, L. (2025). Gesture-Controlled Robotic Arm for Small Assembly Lines. *Machines*, *13*(3). https://doi.org/10.3390/machines13030182

Arif, Y. M., Mustofa, A. H., Fahmi, K., Holle, H., Ismail, M., Wibowo, A., Aziza, M. R., Junikhah, A., & Hasanah, N. A. (2025). Pengendalian Gerak Robot Beroda Menggunakan Sarung Tangan Pintar dengan Neural Network Backpropagation. *Seminar Nasional Fortei Regional*, *7*.

Ashfaq, S., AskariHemmat, M., Sah, S., Saboori, E., Mastropietro, O., & Hoffman, A. (2022). *Accelerating Deep Learning Model Inference on Arm CPUs with Ultra-Low Bit Quantization and Runtime*. https://arxiv.org/pdf/2207.08820

Asriani, A., Lapatta, N. T., Nugraha, D. W., Amriana, A., & Wirdayanti, W. (2025). Implementation of ResNet-50-Based Convolutional Neural Network For Mobile Skin Cancer Classification. *Journal of Applied Informatics and Computing*, *9*(4), 1969–1577. https://doi.org/10.30871/jaic.v9i4.9696

At-taufieqy, M. H. A. (2025). *Navigasi Otonom Robot Soccer Menggunakan Kamera Berbasis Segmentasi Warna Hue Saturation Value*.

Bandini, A., & Zariffa, J. (2022). *Analysis of the hands in egocentric vision: A survey*. https://doi.org/10.1109/TPAMI.2020.2986648

Bansal, P., Kim, E. J., & Ozdemir, S. (2024). Discrete choice experiments with eye-tracking: How far we have come and ways forward. *Journal of Choice Modelling*, *51*, 100478. https://doi.org/10.1016/j.jocm.2024.100478

Cahyadi, W., Chaidir, A. R., Al Insani, A. A., Anam, K., & Eska, A. C. (2023). Pengendali Wireless Mobile Robot Arm (WMRA) Berdasarkan Gestur Lengan Menggunakan Sensor Accelerometer dan Logika Fuzzy. *IJEIS (Indonesian Journal of Electronics and Instrumentation Systems)*, *13*(2), 195. https://doi.org/10.22146/ijeis.77125

Han, X., Zhang, Z., Ding, N., Gu, Y., Liu, X., Huo, Y., Qiu, J., Yao, Y., Zhang, A., Zhang, L., Han, W., Huang, M., Jin, Q., Lan, Y., Liu, Y., Liu, Z., Lu, Z., Qiu,

X., Song, R., … Zhu, J. (2021). Pre-trained models: Past, present and future. *AI Open*, *2*, 225–250. https://doi.org/10.1016/j.aiopen.2021.08.002

Ibnu Katsir, I. bin U. (2003). *Tafsir Ibnu Katsir* (Vol. 3). Pustaka Imam Asy-Syafi'i.

Irmak, E. (2021). Multi-Classification of Brain Tumor MRI Images Using Deep Convolutional Neural Network with Fully Optimized Framework. *Iranian Journal of Science and Technology - Transactions of Electrical Engineering*, *45*(3), 1015–1036. https://doi.org/10.1007/s40998-021-00426-9

Jangir, R., Hansen, N., Ghosal, S., Jain, M., & Wang, X. (2022). Look Closer: Bridging Egocentric and Third-Person Views with Transformers for Robotic Manipulation. *IEEE Robotics and Automation Letters*, *7*(2), 3046–3053. https://doi.org/10.1109/LRA.2022.3144512

Kattenborn, T., Leitloff, J., Schiefer, F., & Hinz, S. (2021). Review on Convolutional Neural Networks (CNN) in vegetation remote sensing. Dalam *ISPRS Journal of Photogrammetry and Remote Sensing* (Vol. 173, hlm. 24–49). Elsevier B.V. https://doi.org/10.1016/j.isprsjprs.2020.12.010

Lee, D., Hwang, J., Jung, D., Koh, J.-S., Do, H., & Kim, U. (2024). Intuitive Six-Degree-of-Freedom Human Interface Device for Human–Robot Interaction. *IEEE Transactions on Instrumentation and Measurement*, *73*, 1–10. https://doi.org/10.1109/TIM.2024.3449982

Li, R., Zheng, S., Zhang, C., Duan, C., Su, J., Wang, L., & Atkinson, P. M. (2022). Multiattention Network for Semantic Segmentation of Fine-Resolution Remote Sensing Images. *IEEE Transactions on Geoscience and Remote Sensing*, *60*. https://doi.org/10.1109/TGRS.2021.3093977

Li, Z., Liu, F., Yang, W., Peng, S., & Zhou, J. (2022). A Survey of Convolutional Neural Networks: Analysis, Applications, and Prospects. *IEEE Transactions on Neural Networks and Learning Systems*, *33*(12), 6999–7019. https://doi.org/10.1109/TNNLS.2021.3084827

Nwankpa, C., Ijomah, W., Gachagan, A., & Marshall, S. (2018). *Activation Functions: Comparison of trends in Practice and Research for Deep Learning*. http://arxiv.org/abs/1811.03378

Purwono, Ma'arif, A., Rahmaniar, W., Fathurrahman, H. I. K., Frisky, A. Z. K., & Haq, Q. M. U. (2022). Understanding of Convolutional Neural Network (CNN): A Review. *International Journal of Robotics and Control Systems*, *2*(4), 739–748. https://doi.org/10.31763/ijrcs.v2i4.888

Qur'an Kemenag. (2019). Diambil 11 Februari 2026, dari https://quran.kemenag.go.id/quran/per-ayat/surah/2?from=1&to=286

Rahmawati, A., Yulianti, I., Nurajizah, S., Hidayatulloh, T., Oktarini Sari, A., & Author, C. (2025). Analisis Performa Model ResNet-50 Pada Diagnosis Pneumonia Balita Berdasarkan Citra Radiografi Thorax. Dalam *Universitas*

Nusa Mandiri Jl. Raya Jatiwaringin (Vol. 5, Nomor 1). https://www.kaggle.com/datasets/paultimothymooney/chest-

Ramadhani, A. (2024). *TA : Sistem Kontrol Robot Mobil melalui Deteksi Bentuk Gestur Jari Tangan menggunakan MediaPipe*.

Rohman, B. P. A., Nishimoto, M., & Ogata, K. (2022). Reconstruction of Missing Ground-Penetrating Radar Traces Using Simplified U-Net. *IEEE Geoscience and Remote Sensing Letters*, *19*. https://doi.org/10.1109/LGRS.2021.3072028

Roziqin, M. F., Joni, K., & Ulum, M. (2021). The Design and Build A Field Tennis Robot With Wifi Based Hand Gesture Recognition Using PID Method. *JTECS : Jurnal Sistem Telekomunikasi Elektronika Sistem Kontrol Power Sistem dan Komputer*, *1*(1), 33. https://doi.org/10.32503/jtecs.v1i1.1428

Saharia, C., Ho, J., Chan, W., Salimans, T., Fleet, D. J., & Norouzi, M. (2023). Image Super-Resolution via Iterative Refinement. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *45*(4), 4713–4726. https://doi.org/10.1109/TPAMI.2022.3204461

Septyanto, A. (2024). *Implementasi Pengenalan Objek Dan Wajah Pada Service Robot Berbasis Convolutional Neural Network*. https://repository.unsri.ac.id/151446/1/RAMA_20201_03041382025099_0017097405_01_front_ref.pdf

Shihab, M. Q. (2000). *Tafsir Al-Misbah: Pesan, Kesan, dan Keserasian Al-Qur'an* (Vol. 8). Lentera Hati.

Tri, A., Hat, A., Nahwa Utama, S., Imamudin, M., Arif, Y. M., & Hanani, A. (2025). *Ball Detection in Wheeled Soccer Robot Using the YOLOv8 Model*.

Vemuri, N., & Thaneeru, N. (2023). Enhancing Human-Robot Collaboration in Industry 4.0 with AI-driven HRI. *Power System Technology*, *47*(4), 341–358. https://doi.org/10.52783/pst.196

Wang, P., Ding, C., Shao, Z., Hong, Z., Zhang, S., & Tao, D. (2023). Quality-Aware Part Models for Occluded Person Re-Identification. *IEEE Transactions on Multimedia*, *25*, 3154–3165. https://doi.org/10.1109/TMM.2022.3156282

Yaldaie, A., Porras, J., & Drögehorn, O. (2024). Innovative Home Automation with Raspberry Pi: A Comprehensive Approach to Managing Smart Devices. *Asian Journal of Computer Science and Technology*, *13*(1), 27–40. https://doi.org/10.70112/AJCST-2024.13.1.4260

Yu, W., & Tian, X. (2023). Deep Learning-Based Visual Navigation Algorithms for Mobile Robots: A Comprehensive Study. *Journal of Computing and Information Technology*, *30*(4), 257–273. https://doi.org/10.20532/cit.2022.1005689