



**PENGEMBANGAN ROBOT ARM BERBASIS IMAGE
PROCESSING UNTUK PEMROSESAN GESTURE JARI
TANGAN SECARA REAL-TIME DI LABORATORIUM FTI
UNIVERSITAS DINAMIKA**



Program Studi

S1 Teknik Komputer

Oleh:

Achmad Bayu Setiawan

22410200002

**FAKULTAS TEKNOLOGI DAN INFORMATIKA
UNIVERSITAS DINAMIKA
2025**

**Pengembangan Robot Arm Berbasis Image Processing
untuk Pemrosesan Gesture Tangan Secara Real-Time di
Laboratorium FTI Universitas Dinamika**

Diajukan sebagai salah satu syarat untuk menyelesaikan

Program Strata Satu (S1)

Disusun Oleh :



Nama : Achmad Bayu Setiawan
Nim : 22410200002
Program : S1 (Strata Satu)
Jurusan : Teknik Komputer

FAKULTAS TEKNOLOGI DAN INFORMATIKA

UNIVERSITAS DINAMIKA

2025

LEMBAR PENGESAHAN

**PENGEMBANGAN ROBOT ARM BERBASIS IMAGE PROCESSING
UNTUK PEMROSESAN GESTURE JARI TANGAN SECARA REAL-TIME
DI LABORATORIUM FTI UNIVERSITAS DINAMIKA**

Laporan Kerja Praktik oleh

Nama : Achmad Bayu Setiawan

NIM : 22410200002

Telah diperiksa, diuji, dan disetujui

Surabaya, 7 Juli 2024

Disetujui oleh:

Pembimbing

Penyelia



Heri Pratikno, M.T.
NIDN. 0716117302



Teguh Sutanto, M.Kom.
NIDN. 0713027801

Mengetahui,

Ketua Program Studi S1 Teknik Komputer



Fakultas Teknologi dan Informatika
UNIVERSITAS
Dinamika



Pauladie Susanto, S.Kom., M.T.
NIDN. 0729047501



UNIVERSITAS
“Just do it”
Dinamika

PERNYATAAN

PERSETUJUAN PUBLIKASI DAN KEASLIAN KARYA ILMIAH

Sebagai mahasiswa Universitas Dinamika, Saya :

Nama : Achmad Bayu Setiawan

NIM : 22410200002

Program Studi : SI Teknik Komputer

Fakultas : Fakultas Teknologi dan Informatika

Jenis Karya : Laporan Kerja Praktik

Judul Karya : PENGEMBANGAN ROBOT ARM BERBASIS IMAGE
PROCESSING UNTUK PEMROSESAN GESTURE
JARI TANGAN SECARA REAL-TIME DI
LABORATORIUM FTI UNIVERSITAS DINAMIKA

Menyatakan dengan sesungguhnya bahwa :

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

Demikian surat pernyataan ini Saya buat dengan sebenar-benarnya.

Surabaya, 7 Juli 2025



Achmad Bayu Setiawan
NIM : 22410200002

ABSTRAK

Laporan ini membahas mengenai perancangan dan pengembangan sistem kendali robot arm yang berbasis image processing, dengan fokus pada pengenalan gesture tangan secara real-time. Sistem ini dikembangkan menggunakan kombinasi teknologi modern yaitu MediaPipe untuk pendeteksian titik-titik tangan (hand landmarks), OpenCV untuk pemrosesan citra, serta Jetson Nano sebagai platform komputasi utama. Tujuan dari proyek ini adalah menciptakan sistem interaksi antara manusia dan mesin tanpa kontak fisik, di mana gerakan tangan dapat diterjemahkan sebagai perintah untuk menggerakkan lengan robot (robot arm). Sistem bekerja dengan cara menangkap gambar secara langsung dari kamera, kemudian mengekstraksi fitur-fitur penting dari tangan seperti posisi ibu jari, telunjuk, dan jari lainnya. Berdasarkan posisi titik-titik tersebut, sistem mengklasifikasikan pola gesture seperti "tangan terbuka", "mengepal", dan "menunjuk", lalu menerjemahkannya menjadi instruksi digital untuk mengendalikan motor servo pada robot arm. Pengujian dilakukan dalam lingkungan laboratorium dengan berbagai variasi posisi dan jarak tangan. Hasil menunjukkan bahwa sistem mampu mengenali gesture dengan akurasi yang cukup tinggi dan waktu respon cepat, selama pencahayaan dan sudut pandang kamera sesuai. Implementasi ini membuktikan bahwa pengolahan citra real-time dapat diaplikasikan secara efektif dalam sistem kendali fisik seperti robot, sekaligus membuka peluang untuk aplikasi lain seperti asistensi robotik, alat bantu disabilitas, atau sistem interaktif tanpa sentuhan.

Kata Kunci: Image Processing, Gesture Recognition, MediaPipe, Robot Arm, Jetson Nano

KATA PENGANTAR

Puji syukur saya panjatkan kepada Tuhan Yang Maha Esa atas segala rahmat yang telah diberikan-Nya, sehingga penulis dapat menyelesaikan laporan kerja praktek ini sebagai salah satu persyaratan untuk menyelesaikan program sarjana pada program studi S1 Teknik Komputer Universitas Dinamika Surabaya.

Laporan kerja praktik di Laboratorium IOT Univesitas Dinamika Surabaya, disusun berdasarkan hasil kerja yang telah dilaksanakan di bulan Februari sampai bulan maret. Dan hasil dari pelaksaan kerja praktik ini, bertujuan untuk mengembangkan dan membuat Robot Arm Berbasis Image Processing untuk Pemrosesan Gesture Tangan Secara Real-Time

Dalam penyusunan laporan ini, penulis menyadari masih banyak kekurangan baik dari segi susunan serta cara penulisan laporan ini, oleh karena itu harapan penulis agar bisa mendapatkan saran dan kritik yang sifatnya membangun demi kesempurnaan laporan ini. Semoga laporan ini bisa bermanfaat bagi para pembaca pada umumnya dan juga bermanfaat bagi penyusun.

Surabaya, 12 Mei 2025

Achmad Bayu Setiawan

DAFTAR ISI

ABSTRAK.....	vi
KATA PENGANTAR.....	vii
DAFTAR ISI.....	viii
DAFTAR GAMBAR.....	x
DAFTAR LAMPIRAN.....	xi
BAB I PENDAHULUAN	1
1.1 Latar Belakang.....	1
1.2 Rumusan Masalah	2
1.3 Batasan Masalah.....	2
1.4 Tujuan.....	3
1.5 Manfaat.....	3
BAB II GAMBARAN UMUM PERUSAHAAN.....	4
2.1 Sejarah Singkat Perusahaan.....	4
2.1.1. Visi dan Misi	5
2.1.2. Tujuan.....	5
2.1.3. Struktur Organisasi.....	6
2.1.4. Informasi Kontak.....	6
2.2.1. Program Studi S1 Teknik Komputer.....	7
2.2.2. Visi dan Misi	7
2.2.3. Tujuan.....	8
2.2.4. Profesi Lulusan.....	8
BAB III LANDASAN TEORI.....	10
3.1 Pengolahan Citra Digital (Image Processing).....	10
3.2 Pengenalan Gesture Tangan.....	10
3.3 PyCharm	11
3.4 OpenCV (Open Source Computer Vision Library).....	12
3.5 MediaPipe.....	13
3.6 PCA9685 dan Library Adafruit.....	14
3.7 Real-Time System	15
BAB IV DESKRIPSI PEKERJAAN	16

4.1	Gambaran Umum Proyek.....	16
4.2	Implementasi Pengolahan Citra Digital	17
4.3	Deteksi Gesture Tangan dan Interpretasi	20
4.4	Kendali Servo dengan PCA9685.....	25
4.5	Kendali Servo Berdasarkan Gesture dan Posisi.....	27
4.6	Antarmuka Visual dan Monitoring Sistem	29
4.7	Evaluasi Respons Sistem dan Akurasi Gesture	31
4.8	Diagram Alur Sistem dan Logika Kendali Gesture.....	32
BAB V PENUTUP		34
5.1	Kesimpulan.....	34
5.2	Saran.....	35
DAFTAR PUSTAKA.....		36



UNIVERSITAS
Dinamika

DAFTAR GAMBAR

	Halaman
Gambar 2. 1 Struktur Organisasi.....	6
Gambar 3. 1 Hand Landmark (Sumber : https://bit.ly/4kNELhM)	11
Gambar 3. 2 Logo Pycharm (Sumber : https://intellij-support.jetbrains.com/)	12
Gambar 3. 3 Tampilan Logo OpenCV (Sumber : https://logodix.com/opencv)	13
Gambar 3. 4 Logo MediaPipe (Sumber : https://chuoling.github.io/mediapipe/)	14
Gambar 4.1 Kode inisialisasi Pustaka.....	16
Gambar 4. 2 Konversi BGR ke RGB.....	17
Gambar 4. 3 Contoh Hand Landmark (Sumber : https://www-analyticsvidhya-com.translate.goog/blog/2022/03/hand-landmarks-detection-on-an-image-using-mediapipe/?_x_tr_sl=en&_x_tr_tl=id&_x_tr_hl=id&_x_tr_pto=imgs)	18
Gambar 4. 4 Kode video capture.....	19
Gambar 4. 5 Kode tampilkan hasil gambar	19
Gambar 4. 6 Konversi BGR ke RGB.....	20
Gambar 4. 7 Pose Open/Move	21
Gambar 4. 8 Pose Mengepal	21
Gambar 4. 9 Pose OK.....	22
Gambar 4. 10 Pose Metal	22
Gambar 4. 11 Kode insialisasi mp dan deteksi gestur	24
Gambar 4. 12 Lanjutan kode deteksi gestur.....	24
Gambar 4. 13 Kode deteksi berbagai gestur jari tangan.....	24
Gambar 4. 14 Lanjutan kode gestur jari tangan.....	25
Gambar 4.15 Konversi derajat servo ke pwm.....	26
Gambar 4.16 Pergerakan servo untuk arm robot	26
Gambar 4. 17 Kode kontrol servo dengan hand landmark	27
Gambar 4. 18 Lanjutan kode kontrol servo.....	27
Gambar 4. 19 Kode perhitungan sudut rotasi dan Gerak servo.....	28
Gambar 4. 20 Kode detect servo	30
Gambar 4. 21 Hasil percobaan gesture lain	31
Gambar 4.22 Flowchart program	33

DAFTAR LAMPIRAN

	Halaman
Lampiran 1. Letter of Acceptance (LoA)	37
Lampiran 2. Acuan Kerja.....	38
Lampiran 3. Logbook Harian.....	40
Lampiran 4. Kehadiran Kerja Praktik.....	42
Lampiran 5. Kartu Bimbingan Kerja Praktik	43
Lampiran 6. Biodata Penulis.....	44



UNIVERSITAS
Dinamika

BAB I

PENDAHULUAN

1.1 Latar Belakang

Perkembangan teknologi di bidang kecerdasan buatan dan pengolahan citra telah membuka berbagai peluang baru dalam penerapan sistem kendali berbasis gesture. Salah satu implementasi yang menarik adalah kendali robot menggunakan gerakan tangan secara real time, yang dapat meningkatkan interaksi manusia-mesin secara lebih alami. Teknologi seperti MediaPipe dari google, yang mampu mendeteksi titik-titik tangan (landmarks) secara akurat dan cepat, menjadi komponen penting dalam sistem ini.

Laboratorium Fakultas Teknologi Industri (FTI) Universitas Dinamika sebagai pusat pembelajaran dan pengembangan teknologi, mendukung kegiatan penelitian dan kerja praktik yang berkaitan dengan sistem berbasis kecerdasan buatan dan robotika. Melalui proyek ini, mahasiswa ditantang untuk merancang dan mengembangkan sistem robot arm yang mampu merespon gesture tangan secara real time menggunakan kamera dan pemrosesan citra digital.

Implementasi sistem ini diharapkan dapat menjadi salah satu kontribusi nyata dalam pemanfaatan teknologi komputer visi dan kecerdasan buatan di lingkungan akademik dan industri, khususnya dalam bidang otomatisasi dan human interaction.

1.2 Rumusan Masalah

Berdasarkan latar belakang tersebut, rumusan masalah dalam kerja praktik ini sebagai berikut:

1. Bagaimana merancang sistem pengolahan citra untuk mengenali gesture tangan secara real-time?
2. Bagaimana cara menghubungkan hasil pengenalan gesture dengan sistem kendali robot arm?
3. Bagaimana tingkat akurasi dan kecepatan respons sistem dalam lingkungan laboratorium?

1.3 Batasan Masalah

Batasan masalah pada pelaksanaan kerja praktik ini adalah berikut :

1. Modul sistem ini hanya mengenali gesture tangan menggunakan teknologi MediaPipe berbasis kamera secara real-time tanpa dukungan sensor tambahan seperti IMU atau sensor jarak.
2. Jenis gesture tangan yang dikenali hanya dibatasi pada empat pola utama, yaitu reset (pose awal), start(pose ok), tangan terbuka (open hand), dan mengepal (fist).
3. Pengolahan citra gesture dilakukan dengan pendekatan deteksi landmark dari tangan dan logika posisi koordinat jari, tanpa menggunakan metode klasifikasi berbasis machine learning atau deep learning.
4. Pengujian sistem hanya dilakukan dalam kondisi pencahayaan laboratorium standar dengan latar belakang yang minim gangguan, sehingga performa sistem belum diuji dalam kondisi pencahayaan rendah atau area ramai.

1.4 Tujuan

Tujuan Kerja praktik di Laboratorium Laboratorium IOT Univesitas Dinamika Surabaya adalah sebagai berikut:

1. Mengembangkan sistem pengenalan gesture tangan berbasis image processing secara real-time
2. Mengintegrasikan sistem gesture recognition dengan kendali robot arm.
3. Menguji performa sistem dalam skenario laboratorium.

1.5 Manfaat

Sistem robot arm berbasis image processing untuk pengenalan gesture tangan yang dikembangkan di Laboratorium FTI Universitas Dinamika memiliki berbagai manfaat, baik bagi mahasiswa maupun lingkungan laboratorium, antara lain:

1. Sistem ini membantu mahasiswa memahami penerapan konsep pengolahan citra digital dan pengenalan gesture tangan secara langsung, sehingga memperkuat pemahaman terhadap teori yang telah dipelajari di kelas, khususnya dalam bidang computer vision dan interaksi manusia-komputer (HCI).
2. Robot arm ini berfungsi sebagai media pembelajaran yang interaktif di laboratorium, di mana mahasiswa dapat melakukan eksperimen secara real-time untuk mengendalikan gerakan robot hanya melalui isyarat tangan, tanpa alat input konvensional seperti joystick atau tombol.
3. Dengan melibatkan teknologi seperti MediaPipe dan Jetson Orin Nano, mahasiswa memperoleh keterampilan praktis dalam mengembangkan sistem berbasis AI dan mengintegrasikan komputasi visual dengan aktuator seperti motor servo, yang berguna dalam dunia industri dan penelitian berbasis otomasi.

BAB II

GAMBARAN UMUM PERUSAHAAN

2.1 Sejarah Singkat Perusahaan

Pada 30 April 1983 terjadi pengembangan teknologi dan informasi menjadi hal penting dalam pembangunan dan pengembangan nasional. Kedua hal tersebut juga harus diiringi dengan di bidang ekonomi dan bisnis untuk bisa bersaing di era yang terus berkembang. Seni dan budaya harus tetap dipertahankan agar identitas bangsa tidak musnah. Melalui empat (4) hal utama, yaitu kritis, kreatif, kolaborasi, dan komunikasi, para pendiri terdiri dari laksda. TNI (Purn) Mardiono, Ir. Andrian A.T, Ir. Handoko A.T, Dra. Rosy Merianti, Ak. Dalam bidang teknologi informasi dengan nama AKIS (Akademi Komputer dan Informatika Surabaya).

Pada 10 Maret 1984 ijin operasional penyelenggara program Diploma III Manajemen Informatika diberikan kepada AKIS melalui SK Kopertis Wilayah VII Jawa Timur.

Pada 19 Juni 1984 AKIS yang berlokasi di Ketintang Surabaya memperoleh status terdaftar dari DIKTI.

Pada 20 Maret 1986 terus meningkatnya kebutuhan pendidikan, Yayasan Putra Bhakti memutuskan untuk merubah Akademi menjadi Sekolah Tinggi. AKIS (Akademi Komputer dan Informatika Surabaya) berubah menjadi Sekolah Tinggi Manajemen Informatika dan Teknik Komputer Surabaya, yang lebih dikenal dengan STIKOM Surabaya.

Pada 11 Desember 1987 STIKOM Surabaya membangun kampus pertama yang berlokasi di Jalan Kutisari No.66 Surabaya, yang diresmikan oleh Letnan Jendral TNI Wahono selaku Gubernur Jawa Timur pada saat itu.

Pada 28 Oktober 1997 Awal pemasangan tiang pancang pertama STIKOM Surabaya di Jalan Raya Kedung Baruk No.98 Surabaya bersamaan dengan Hari Sumpah Pemuda.

Pada 04 September Seiring dengan perubahan zaman serta kebutuhan masyarakat, STIKOM Surabaya resmi berubah menjadi Institut dengan nama Institut Bisnis dan Informatika STIKOM Surabaya yang memiliki 2 fakultas dengan

9 program studi.

Pada 29 Juli 2019 Melalui Surat Keputusan Riset Dikti, Institut Bisnis dan Informatika STIKOM Surabaya resmi berubah menjadi Universitas Dinamika yang memiliki 2 fakultas dengan 9 program studi, yakni Fakultas Teknologi dan Informatika, Prodi S1 Teknik Komputer, Prodi S1 Desain Komunikasi Visual, Prodi S1 Desain Produk, Prodi D4 Produksi Film dan Televisi, dan Prodi D3 Sistem Informasi. Serta Fakultas Ekonomi dan Bisnis (FEB) dengan Prodi S1 Manajemen, Prodi S1 Akuntansi, dan Prodi S1 Administrasi Perkantoran.

Pada 31 Mei 2021 Melalui Surat Keputusan Rektor, Universitas Dinamika melakukan perubahan struktur organisasi dengan membentuk fakultas baru, yakni Fakultas Desain dan Industri Kreatif (FDIK) dengan 3 program studi, yaitu Prodi S1 Desain Produk, Prodi S1 Desain Komunikasi Visual, dan D4 Produksi Film dan Televisi yang sebelumnya berada dibawah naungan Fakultas Teknologi dan Informatika (FTI).

2.1.1. Visi dan Misi

Visi

Menjadi *smart entrepreneurial university* berskala global yang produktif dalam berinovasi.

Misi

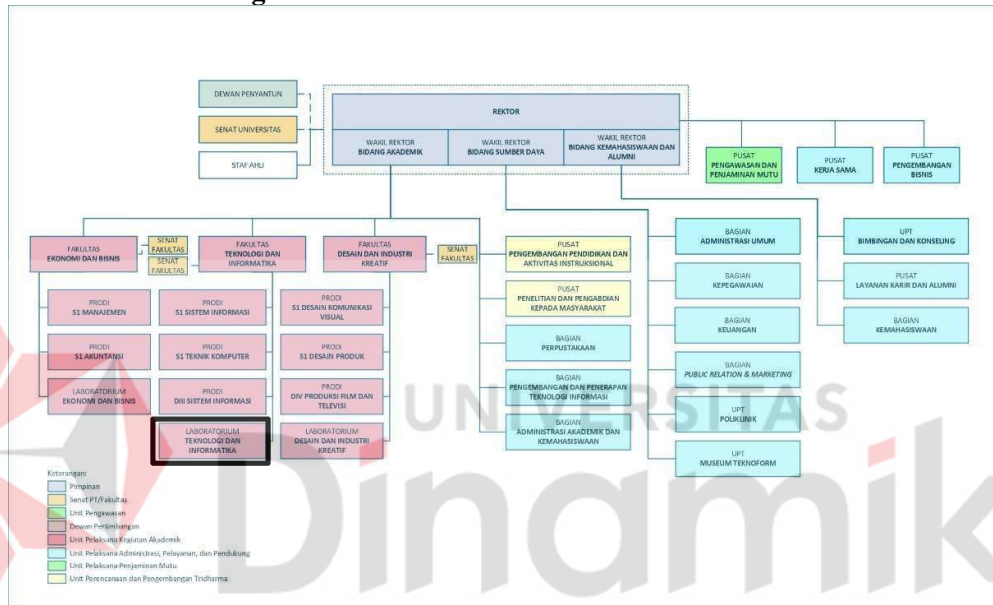
1. Menyelenggarakan dan mengembangkan pendidikan berbasis teknologi informasi yang bermutu dan berdaya saing global.
2. Melaksanakan penelitian yang berfokus pada pengembangan inovasi untuk mewujudkan entrepreneurial university.
3. Melakukan pengabdian untuk menyebar luaskan ipteks dan hasil inovasi bagi kesejahteraan masyarakat.
4. Melaksanakan kemitraan berskala global.
5. Mengembangkan bisnis dan kewirausahaan secara otonom yang akuntabel dan transparan.

2.1.2. Tujuan

1. Menyelenggarakan pendidikan yang berkualitas, inovatif, dan futuristik.
2. Menciptakan SDM berdaya saing global dan berjiwa *entrepreneur*.

3. Menghasilkan penelitian berkualitas dan berskala global.
4. Menghasilkan inovasi yang bernilai jual dan bermanfaat bagi masyarakat.
5. Melaksanakan diseminasi ipteks dan/atau hasil inovasi untuk meningkatkan kesejahteraan masyarakat.
6. Mewujudkan kemitraan berskala global.
7. Menjamin keberlanjutan Perguruan Tinggi.

2.1.3. Struktur Organisasi



Gambar 2. 1 Struktur Organisasi

Pada bagian yang di tandai dengan kotak yang berwarna hitam disitulah tempat saya melakukan kerja pratik tersebut.

2.1.4. Informasi Kontak

Tempat : Jl. Raya Kedung Baruk 98 Surabaya

Email : official@dinamika.ac.id

Website : dinamika.ac.id

No telfon : +62(31)8721731

Sosial Media

Facebook : Universitas Dinamika

Youtube : Universitas Dinamika

Instagram : @universitasdinamika

2.2.1. Program Studi S1 Teknik Komputer

S1 Teknik Komputer mulai beroperasi mulai tahun 1991. Program Studi S1 Teknik Komputer bertujuan mencetak lulusan yang kompeten dalam bidang sistem komputer dan otomasi industri. Mahasiswa dibekali dengan berbagai mata kuliah terkait, seperti **Microcontroller, Programmable Logic Controller (PLC), Sistem Digital, Jaringan Komputer, dan Robotika.**

Lulusan program ini diharapkan mampu menganalisis, merancang, dan mengembangkan sistem otomasi industri berbasis komputer. Profesi yang dapat ditempuh antara lain **Computer System Engineer, Embedded System Developer, dan Automation Engineer.**

Program ini juga menekankan pada pemahaman teori dan praktik melalui berbagai kegiatan akademik, seperti **praktikum, penelitian, dan proyek akhir.** Selain itu, mahasiswa dibekali dengan keterampilan dalam pemrograman tingkat tinggi, pengolahan data, serta pengembangan sistem berbasis kecerdasan buatan. Dengan kurikulum yang terus disesuaikan dengan perkembangan industri, lulusan S1 Teknik Komputer Universitas Dinamika diharapkan dapat bersaing di dunia kerja maupun melanjutkan studi ke jenjang yang lebih tinggi.

Selain itu, mahasiswa juga diberikan kesempatan untuk mengikuti **sertifikasi industri** seperti Cisco Certified Network Associate (CCNA) dan sertifikasi pemrograman embedded system, guna meningkatkan daya saing di dunia kerja. Dengan adanya kerja praktik, mahasiswa dapat mengasah keterampilan yang relevan dengan kebutuhan industri dan memiliki pengalaman nyata sebelum terjun ke dunia profesional.

2.2.2. Visi dan Misi

Visi:

Mengembangkan keilmuan di bidang IoT yang didukung oleh kecerdasan artifial, dan diintegrasikan dengan konsep *technopreneurship*, sehingga mampu menciptakan inovasi yang bermanfaat bagi masyarakat dan industri berskala global.

Misi:

1. Mengembangkan pendidikan dan pengajaran di bidang Teknik Komputer yang bermutu, berwawasan global, dan mengarah pada *technopreneurship*.
2. Melaksanakan penelitian di bidang Teknik Komputer yang inovatif dan

solutif bagi masyarakat dan industri berskala global.

3. Melaksanakan pengabdian atau penerapan hasil inovasi di bidang Teknik Komputer yang bermanfaat bagi masyarakat dan industri.

2.2.3. Tujuan

Tujuan Program Studi S1 Teknik Komputer adalah sebagai berikut :

1. Lulusan memiliki kemampuan menganalisis permasalahan sistem komputer khususnya pada aspek perangkat keras untuk menghasilkan solusi bagi organisasi.
2. Lulusan memiliki kemampuan menganalisis perangkat lunak (meliputi pemrograman antarmuka, pemrograman *real-time*) dan perangkat keras (meliputi pemantauan, pengendalian) sistem komputer sebagai solusi bagi permasalahan organisasi. Lulusan memiliki kemampuan menganalisis dan merancang sistem komputer dengan menerapkan sistem tertanam, *Internet of Things*(IoT), kecerdasan artifisial, dan atau jaringan komputer untuk menghasilkan solusi bagi organisasi.
3. Lulusan yang memiliki kemampuan dalam merumuskan keputusan yang tepat berdasarkan analisis informasi dan data, beretika, dan bertanggung jawab pada pekerjaan dalam lingkup tugasnya.

2.2.4. Profesi Lulusan

Profesi Lulusan Program Studi

1. **IoT Engineer** : Menyediakan produk dan atau solusi IoT sesuai dengan kebutuhan pengguna.
2. **Artificial Intelligent Engineer** : Membangun solusi berbasis kecerdasan artifisial (Artificial Intelligence/AI)
3. **Network Designer** : Melaksanakan penyediaan desain instalasi jaringan dan infrastruktur meliputi kegiatan pemetaan kebutuhan, monitoring dan pengawasan dampak design pembangunan dan pengembangan instalasi jaringan dan infrastruktur yang dibutuhkan oleh user sejalan dengan rencana dan pengembangan organisasi.
4. **Digital Computer Technology Advisor** : Memecahkan masalah teknis, memberikan saran tentang perangkat keras dan perangkat lunak yang tepat,

serta mengoptimalkan penggunaan teknologi dalam bisnis atau kehidupan sehari-hari.

5. **Industrial Automation** : Meningkatkan efisiensi dan produktivitas sistem produksi di industri



UNIVERSITAS
Dinamika

BAB III

LANDASAN TEORI

3.1 Pengolahan Citra Digital (Image Processing)

Pengolahan citra digital merupakan salah satu cabang dari ilmu komputer yang berfokus pada pemrosesan gambar berbasis piksel secara numerik menggunakan perangkat komputer. Berbeda dengan pengolahan citra analog, proses ini bekerja pada representasi digital dari suatu gambar dua dimensi, memungkinkan analisis, perbaikan kualitas, dan ekstraksi informasi visual secara sistematis.

Tahapan dasar dalam pengolahan citra mencakup beberapa proses penting. Dimulai dari akuisisi citra, yaitu tahap pengambilan gambar melalui sensor atau kamera digital. Selanjutnya, gambar akan melalui proses prapemrosesan, seperti penghilangan noise, transformasi warna (misalnya dari RGB ke *grayscale*), dan peningkatan kontras, agar kualitas citra menjadi lebih baik untuk dianalisis. Setelah itu dilakukan segmentasi, yaitu pemisahan objek utama (seperti tangan) dari latar belakang. Proses berikutnya adalah ekstraksi fitur, di mana sistem mengambil elemen-elemen penting dari gambar seperti bentuk kontur atau titik-titik kunci. Terakhir, sistem melakukan klasifikasi, yaitu menentukan kategori dari objek berdasarkan fitur yang telah diperoleh.

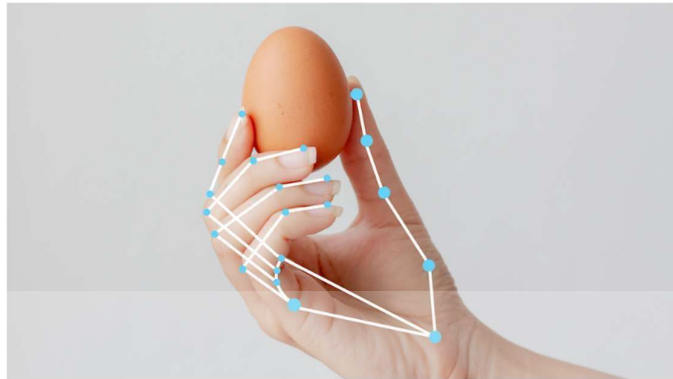
Teknik pengolahan citra sangat relevan untuk sistem yang mengandalkan penglihatan komputer, seperti pengenalan gestur. Kemajuan teknologi dalam bidang ini banyak dibantu oleh *framework* modern seperti OpenCV, *TensorFlow*, dan MediaPipe yang mempermudah pengembangan aplikasi visual secara *real-time* (Gonzalez & Woods, 2018).

3.2 Pengenalan Gesture Tangan

Gesture tangan adalah bentuk komunikasi visual yang lazim digunakan manusia untuk mengekspresikan maksud tertentu tanpa menggunakan kata-kata. Dalam bidang interaksi manusia-komputer (*Human-Computer Interaction*), gesture digunakan sebagai metode alternatif untuk mengendalikan sistem digital secara

alami dan efisien, tanpa perangkat input fisik seperti mouse atau keyboard.

Deteksi gesture berbasis visi mengandalkan kamera untuk merekam posisi dan bentuk tangan. Tahapan yang dilalui antara lain deteksi tangan, yaitu mengenali keberadaan tangan dalam gambar, diikuti oleh pelacakan tangan agar pergerakan dapat terus diikuti secara berkelanjutan dari frame ke frame. Setelah itu dilakukan identifikasi fitur, misalnya melalui landmark atau kontur tangan, yang akan diolah lebih lanjut dalam proses klasifikasi gesture.



Gambar 3. 1 Hand Landmark
(Sumber : <https://bit.ly/4kNELhM>)

Gesture dapat dibedakan menjadi dua jenis, yaitu statik, yang merujuk pada posisi tangan tetap seperti simbol “OK”, dan dinamis, yang melibatkan gerakan seperti melambaikan tangan. Dalam sistem robot berbasis gesture, hasil klasifikasi gesture ini digunakan sebagai sinyal kendali untuk menjalankan perintah tertentu pada aktuator robot. Oleh karena itu, akurasi dalam pengenalan gesture sangat memengaruhi respons keseluruhan sistem (Mitchell et al., 2020).

3.3 PyCharm

PyCharm adalah sebuah perangkat lunak Integrated Development Environment (IDE) yang didesain khusus untuk pengembangan aplikasi berbasis bahasa pemrograman Python. Dikembangkan oleh JetBrains, PyCharm banyak digunakan oleh para profesional di berbagai bidang seperti data science, machine learning, otomasi sistem, hingga pengolahan citra digital. IDE ini menawarkan sebuah solusi komprehensif yang tidak hanya menyediakan editor kode, tetapi juga fitur manajemen pustaka, kontrol versi, dan fasilitas debugging yang sangat membantu.



Gambar 3. 2 Logo Pycharm
(Sumber : <https://intellij-support.jetbrains.com/>)

Dalam proyek kerja praktik ini, PyCharm difungsikan sebagai lingkungan pengembangan utama untuk menulis, menjalankan, serta menguji program Python yang menggunakan berbagai pustaka seperti OpenCV, MediaPipe, dan Adafruit PCA9685. Keunggulan PyCharm terletak pada fitur-fitur seperti penyorotan sintaks (syntax highlighting), pelengkapan kode otomatis (auto-completion), pemeriksaan kesalahan secara langsung (real-time error checking), serta dukungan pengelolaan environment virtual yang memudahkan dalam proses pengembangan aplikasi dan mengurangi potensi kesalahan pemrograman.

Penggunaan IDE ini menjadi sangat penting dalam mendukung efektivitas penulisan logika pengolahan citra dan pengendalian servo. Dengan memanfaatkan PyCharm, pengembangan sistem berbasis gestur jari tangan dapat dilakukan secara lebih terstruktur, modular, serta memudahkan pengujian dan pengembangan berkelanjutan (JetBrains, 2023).

3.4 OpenCV (Open Source Computer Vision Library)

OpenCV, singkatan dari Open Source Computer Vision Library, adalah pustaka pemrograman bersifat open-source yang dirancang untuk memenuhi berbagai kebutuhan dalam pengolahan citra dan visi komputer. Pustaka ini pertama kali dikembangkan oleh Intel dan kini telah menjadi standar utama dalam pembuatan aplikasi berbasis pengolahan citra, berkat kelengkapan fungsi serta efisiensi dalam pemrosesan gambar secara real-time.

Dalam konteks proyek kerja praktik ini, OpenCV berperan dalam mendukung akuisisi gambar dari kamera, pengolahan warna, serta penyajian hasil deteksi gestur tangan pada tampilan antarmuka grafis. Fungsi-fungsi penting seperti VideoCapture digunakan untuk membuka aliran video, cvtColor untuk mengubah

ruang warna, serta imshow dan putText untuk menampilkan hasil deteksi secara langsung pada layar. Kecepatan OpenCV dalam mengelola frame video membuatnya sangat sesuai untuk aplikasi pengenalan gestur secara real-time.



Gambar 3. 3 Tampilan Logo OpenCV
(Sumber : <https://logodix.com/opencv>)

Selain kecepatan dan keluwesan penggunaannya, OpenCV juga dapat berjalan lintas platform dan kompatibel dengan pustaka lain seperti NumPy dan MediaPipe. Dalam proyek ini, OpenCV tidak hanya berperan dalam tahap awal pengolahan citra, tetapi juga sebagai penghubung antara input visual dengan sistem kontrol servo yang menggerakkan lengan robot (Bradski & Kaehler, 2008).

3.5 MediaPipe

MediaPipe adalah sebuah platform pemrosesan media yang dikembangkan oleh Google, dirancang untuk mendukung aplikasi berbasis pembelajaran mesin dan visi komputer. Salah satu komponen utamanya adalah MediaPipe Hands, yakni modul untuk mendeteksi dan melacak tangan secara real-time menggunakan input video dari kamera biasa.

Pendekatan yang digunakan MediaPipe terdiri dari dua tahap: pertama, sistem melakukan deteksi lokasi tangan dalam citra, lalu melanjutkan ke tahap estimasi landmark, yaitu pemetaan posisi 21 titik kunci pada tangan, mencakup ujung-ujung jari dan pergelangan. Titik-titik ini digunakan untuk mengenali bentuk tangan dan klasifikasinya sebagai gesture tertentu.



Gambar 3. 4 Logo MediaPipe
(Sumber : <https://chuoling.github.io/mediapipe/>)

Keunggulan MediaPipe terletak pada kemampuannya untuk dijalankan secara efisien bahkan di perangkat dengan sumber daya terbatas seperti smartphone atau modul Jetson Orin Nano. Platform ini tidak memerlukan kamera kedalaman, cukup kamera RGB biasa, namun mampu memberikan hasil yang sangat presisi. Dengan kinerja yang cepat, ringan, dan mudah diintegrasikan, MediaPipe menjadi pilihan utama dalam pengembangan aplikasi gesture tracking berbasis citra (Lugaresi et al., 2019; Zhang et al., 2020).

3.6 PCA9685 dan Library Adafruit

Dalam proyek ini, pustaka `adafruit-circuitpython-pca9685` digunakan untuk mempermudah pengaturan sinyal PWM. Fungsi seperti `set_servo_angle(channel, angle)` digunakan untuk mengatur posisi servo berdasarkan hasil perhitungan dari gesture tangan. Penggunaan PCA9685 juga membantu mengurangi beban pemrosesan pada Jetson Nano karena sinyal PWM dihasilkan oleh modul secara independen.

Pustaka `adafruit-circuitpython-pca9685` merupakan sebuah library Python yang dibuat oleh Adafruit guna memudahkan pengguna dalam mengoperasikan modul PCA9685 pada platform CircuitPython dan MicroPython. Library ini menyediakan interface pemrograman yang mudah digunakan untuk melakukan inisialisasi modul, mengatur frekuensi PWM, serta mengubah nilai duty cycle pada setiap kanal tanpa harus menangani komunikasi I2C secara langsung.

Penggunaan pustaka ini sangat bermanfaat dalam pengembangan sistem embedded berbasis Python, seperti pada perangkat Raspberry Pi, Jetson Nano, maupun mikrokontroler lain yang kompatibel. Hal ini mempercepat proses pengembangan prototipe dan implementasi aplikasi robotik, khususnya yang melibatkan pengaturan sudut servo motor pada lengan robot secara real-time.

Dengan memanfaatkan PCA9685 dan pustaka ini, beban pemrosesan pada mikrokontroler dapat diminimalkan karena pengaturan sinyal PWM dilakukan oleh modul eksternal. Oleh karena itu, sistem dapat mengendalikan lebih banyak channel

PWM dengan konsumsi daya yang rendah dan stabilitas sinyal yang tinggi (NXP Semiconductors, 2019; Adafruit, 2023).

3.7 Real-Time System

Sistem real-time adalah sistem yang dapat merespons input dalam waktu yang sangat singkat. Dalam pengembangan robot arm berbasis gesture ini, kemampuan sistem untuk merespons pergerakan tangan secara langsung dan akurat sangat penting untuk menghasilkan interaksi yang alami dan responsif antara manusia dan robot.

Sistem waktu nyata (*real-time system*) merujuk pada sistem komputasi yang dirancang untuk merespons input atau kejadian dalam batas waktu tertentu. Keandalan sistem ini tidak hanya ditentukan dari ketepatan hasil, tetapi juga dari kecepatan respons terhadap stimulus. Hal ini sangat penting dalam sistem kendali seperti robot, di mana keterlambatan dalam eksekusi dapat menyebabkan kesalahan atau kerusakan.

Secara umum, sistem waktu nyata dibagi menjadi dua kategori. Hard real-time system adalah sistem yang tidak boleh melewati batas waktu yang telah ditetapkan, biasanya digunakan dalam sistem kritis seperti alat medis atau penerbangan. Sebaliknya, soft real-time system masih dapat mentoleransi keterlambatan dalam jumlah kecil, namun tetap mengutamakan kecepatan reaksi, contohnya pada sistem kendali robot.

Dalam proyek ini, sistem yang digunakan termasuk kategori soft real-time, di mana pengenalan gesture tangan harus segera direspon agar robot dapat bergerak secara sinkron dengan perintah pengguna. Untuk memenuhi kebutuhan ini, diperlukan perangkat keras seperti Jetson Orin Nano yang mendukung pemrosesan AI dan kendali servo secara bersamaan. Dengan kemampuan tersebut, sistem mampu menjalankan berbagai proses, mulai dari penangkapan gambar, analisis gesture, hingga pengiriman sinyal kontrol ke motor servo, dalam waktu yang sangat singkat (Tanenbaum & Bos, 2015).

BAB IV

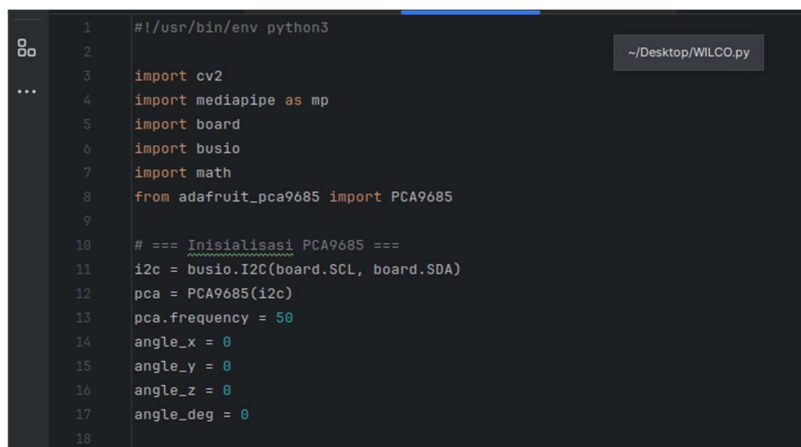
DESKRIPSI PEKERJAAN

4.1 Gambaran Umum Proyek

Kerja praktik yang dilaksanakan di Laboratorium IoT Universitas Dinamika ini bertujuan untuk merancang dan mengimplementasikan sebuah sistem interaktif yang mampu mengendalikan gerakan lengan robot (*robot arm*) berdasarkan gestur tangan manusia secara waktu nyata (*real-time*). Sistem ini memanfaatkan pendekatan *vision-based interaction*, yaitu pengenalan gestur tangan menggunakan citra video dari kamera, tanpa bantuan perangkat keras tambahan seperti sensor IMU atau sensor ultrasonik.

Inovasi utama dari sistem ini terletak pada kemampuannya menerjemahkan pergerakan alami tangan manusia menjadi sinyal digital yang dapat digunakan untuk menggerakkan motor servo pada lengan robot. Dengan teknologi ini, manusia dapat mengendalikan robot hanya melalui isyarat tangan sederhana seperti membuka, mengepal, memberi isyarat “OK”, atau gerakan simbolik lainnya.

Implementasi sistem dilakukan melalui integrasi antara perangkat keras seperti kamera, board Jetson Orin Nano, motor servo, dan modul PWM PCA9685, serta perangkat lunak berbasis Python yang memanfaatkan pustaka OpenCV dan MediaPipe. Seluruh sistem dirancang agar mampu bekerja secara efisien dan stabil dalam lingkungan laboratorium yang memiliki keterbatasan pencahayaan dan ruang.



```
1  #!/usr/bin/env python3
2
3  import cv2
4  import mediapipe as mp
5  import board
6  import busio
7  import math
8  from adafruit_pca9685 import PCA9685
9
10 # === Inisialisasi PCA9685 ===
11 i2c = busio.I2C(board.SCL, board.SDA)
12 pca = PCA9685(i2c)
13 pca.frequency = 50
14 angle_x = 0
15 angle_y = 0
16 angle_z = 0
17 angle_deg = 0
18
```

Gambar 4.1 Kode inisialisasi Pustaka

Gambar 4.1 menampilkan potongan kode awal program yang bertugas untuk mengimpor pustaka-pustaka eksternal yang diperlukan serta melakukan inisialisasi awal sistem. Pustaka cv2 digunakan untuk menangani proses pengambilan dan pengolahan citra dari kamera, sedangkan mediapipe dimanfaatkan dalam mendeteksi titik-titik landmark tangan secara real-time. Selain itu, pustaka board dan busio digunakan untuk konfigurasi antarmuka I2C yang diperlukan oleh modul PWM PCA9685.

Setelah itu, dilakukan proses inisialisasi terhadap modul PCA9685 dengan menyambungkan pin SCL dan SDA melalui objek I2C, lalu menetapkan frekuensi PWM sebesar 50 Hz yang umum digunakan untuk mengontrol motor servo. Variabel angle_x, angle_y, angle_z, dan angle_deg juga didefinisikan sebagai variabel awal untuk menyimpan nilai sudut rotasi yang akan dihitung berdasarkan input gesture tangan.

4.2 Implementasi Pengolahan Citra Digital

Pengolahan citra digital memegang peranan sentral dalam keseluruhan sistem pengenalan gestur berbasis visual yang dikembangkan dalam proyek ini. Proses ini dimulai dengan pengambilan gambar secara real-time menggunakan kamera USB Logitech C270 yang dihubungkan langsung ke perangkat pemroses utama, yaitu Jetson Orin Nano. Kamera tersebut berfungsi sebagai sensor visual untuk menangkap gerakan tangan pengguna secara kontinu dalam bentuk rangkaian gambar atau frame video yang berkelanjutan. Setiap frame yang diperoleh dari kamera akan segera diteruskan ke sistem pemrosesan untuk menjalani serangkaian tahapan transformasi dan analisis, sebelum akhirnya diproses menjadi perintah kontrol terhadap aktuator.

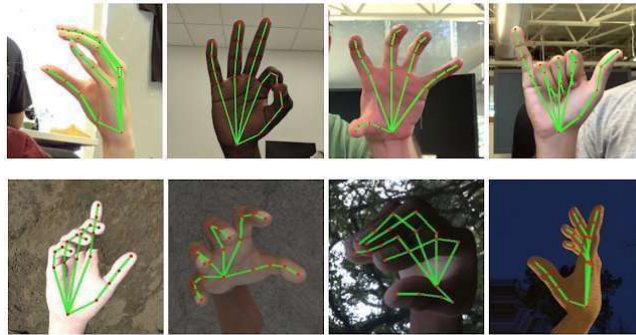
```

99         frame = cv2.flip(frame, [flipCode: 1])
100         h, w, _ = frame.shape
101         rgb = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
102         results = hands.process(rgb)
103
104         gesture_text = "Waiting..."
105     
```

Gambar 4. 2 Konversi BGR ke RGB

Secara default, OpenCV membaca gambar dalam format BGR (Blue-Green-Red), yang merupakan standar internal pustaka tersebut. Namun, pustaka MediaPipe yang digunakan untuk mendeteksi dan melacak tangan, hanya dapat

memproses input dalam format RGB (Red-Green-Blue). Oleh karena itu, setiap frame yang ditangkap harus terlebih dahulu diubah ruang warnanya dari BGR ke RGB. Transformasi warna ini dilakukan melalui fungsi `cv2.cvtColor()` dan menjadi bagian dari tahap pra-pemrosesan (preprocessing) yang sangat krusial. Jika proses ini dilewati atau tidak dilakukan dengan benar, maka model deteksi pada MediaPipe dapat gagal mengenali struktur tangan karena ketidaksesuaian format input.



Gambar 4. 3 Contoh Hand Landmark

(Sumber : https://www-analyticsvidhya-com.translate.goog/blog/2022/03/hand-landmarks-detection-on-an-image-using-mediapipe/?_x_tr_sl=en&_x_tr_tl=id&_x_tr_hl=id&_x_tr_pto=imgs)

Tahapan pra-pemrosesan tidak berhenti sampai di situ. Gambar yang telah dikonversi warnanya seperti pada Gambar 4.3 diatas kemudian dibersihkan dari noise atau gangguan visual lainnya. Proses denoising ini penting untuk meningkatkan kualitas gambar dengan menghilangkan elemen acak yang dapat memengaruhi akurasi pendeteksian. Selain itu, dilakukan juga normalisasi nilai intensitas piksel agar distribusi warna menjadi lebih merata, serta penyesuaian ukuran gambar agar semua frame memiliki dimensi yang seragam. Penyeragaman ukuran frame sangat penting untuk menjaga kestabilan sistem dalam membaca pergerakan tangan pengguna.

MediaPipe secara internal mampu melakukan estimasi bounding box serta deteksi landmark pada tangan, yang secara tidak langsung berfungsi sebagai segmentasi area tangan dari latar. Dengan demikian, sistem tetap dapat fokus pada bagian tangan meskipun lingkungan visual di sekitar tangan tidak sepenuhnya dihilangkan.

```

89 # === Buka Kamera ===
90 cap = cv2.VideoCapture(0)
91

```

Gambar 4. 4 Kode video capture

```

...
156 # Tampilkan hasil
157 cv2.putText(frame, gesture_text, img (10, 30),
158             cv2.FONT_HERSHEY_SIMPLEX, fontScale 0.5, color (0, 255, 0), thickness 2)
159
160 cv2.putText(frame, "H: f'Servo X: (angle_x)\"", img (10, 50),
161             cv2.FONT_HERSHEY_SIMPLEX, fontScale 0.5, color (0, 255, 0), thickness 2)
162 cv2.putText(frame, "H: f'Servo Y: (angle_y)\"", img (10, 70),
163             cv2.FONT_HERSHEY_SIMPLEX, fontScale 0.5, color (0, 255, 0), thickness 2)
164 cv2.putText(frame, "H: f'Servo Z: (angle_z)\"", img (10, 90),
165             cv2.FONT_HERSHEY_SIMPLEX, fontScale 0.5, color (0, 255, 0), thickness 2)
166 cv2.putText(frame, "H: f'Pergeseran Putar: (angle_deg)\"", img (10, 110),
167             cv2.FONT_HERSHEY_SIMPLEX, fontScale 0.5, color (0, 255, 0), thickness 2)
168 cv2.imshow('Window: Hand Gesture Servo Control', frame)
169 if cv2.waitKey(1) & 0xFF == 27:
170     break

```

Gambar 4. 5 Kode tampilkan hasil gambar

Pustaka OpenCV memiliki peran ganda dalam sistem ini, yakni sebagai alat bantu dalam pemrosesan citra sekaligus sebagai sarana visualisasi dan monitoring. OpenCV menyediakan sejumlah fungsi yang sangat berguna selama pengembangan dan operasional sistem. Fungsi `cv2.VideoCapture()` seperti pada Gambar 4.4 digunakan untuk membuka koneksi dengan kamera dan mengambil aliran video, sedangkan `cv2.imshow()` pada Gambar 4.5 menampilkan hasil gambar ke layar secara real-time. Untuk memberikan informasi tambahan kepada pengguna, digunakan pula fungsi seperti `cv2.putText()` pada Gambar 4.5 untuk menampilkan label gesture dan nilai sudut servo, serta `cv2.line()` untuk menggambarkan koneksi antar titik-titik landmark pada tangan. Visualisasi semacam ini memberikan umpan balik instan kepada pengguna mengenai gesture apa yang dikenali oleh sistem, serta bagaimana sistem merespons input visual yang diberikan.

Salah satu tantangan besar dalam pengolahan citra digital adalah ketergantungan terhadap kondisi pencahayaan dan kualitas gambar yang diterima. Pencahayaan yang tidak stabil, bayangan yang muncul di area tangan, atau latar belakang yang kompleks dapat menyebabkan kesalahan dalam proses deteksi landmark. Selain itu, keterbatasan resolusi kamera juga dapat mengurangi kejelasan bentuk jari, terutama saat tangan berada jauh dari kamera. Untuk meminimalkan risiko ini, pengambilan citra dilakukan di laboratorium dengan kondisi yang terkontrol sehingga pencahayaan dibuat stabil, dan menggunakan latar belakang dengan warna-warna netral seperti putih atau abu-abu.

Secara keseluruhan, pengolahan citra digital bukan hanya menjadi tahap awal dari sistem ini, tetapi juga menjadi landasan utama yang menentukan

efektivitas dan keberhasilan sistem secara keseluruhan. Tanpa input visual yang bersih dan akurat, tahapan selanjutnya seperti deteksi gestur dan kendali servo tidak akan dapat bekerja dengan optimal. Dengan memastikan bahwa citra yang diterima oleh sistem telah melewati serangkaian proses pra-pemrosesan yang baik, maka pengenalan gesture dapat dilakukan secara presisi, dan perintah gerakan robot yang dihasilkan pun menjadi lebih responsif serta sesuai dengan instruksi visual yang diberikan oleh pengguna.

```

99     frame = cv2.flip(frame, flipCode=1)
100     h, w, _ = frame.shape
101     rgb = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
102     results = hands.process(rgb)
103
104     gesture_text = "Waiting..."
105

```

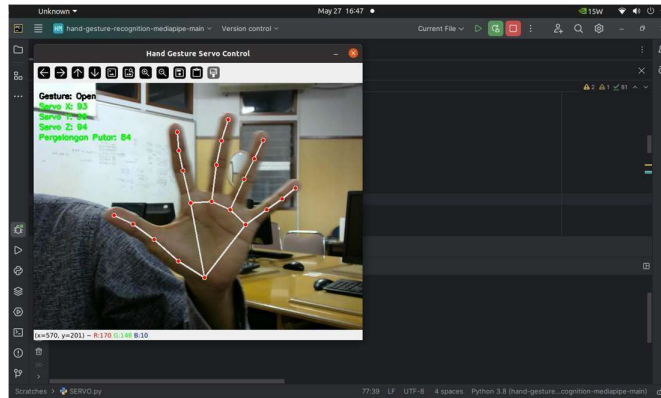
Gambar 4. 6 Konversi BGR ke RGB

Pada bagian Gambar 4.6 tersebut, digunakan pustaka OpenCV untuk menangkap aliran video dari kamera menggunakan `cv2.VideoCapture(0)`. Setiap frame yang diperoleh akan diproses secara real-time. Format warna gambar yang ditangkap dikonversi dari BGR ke RGB menggunakan `cv2.cvtColor()` karena MediaPipe hanya menerima input dalam format RGB. Selain itu, digunakan pula fungsi `cv2.flip()` untuk membalik citra agar lebih sesuai dengan tampilan pengguna, dan `cv2.imshow()` untuk menampilkan citra secara langsung pada jendela antarmuka. Fungsi-fungsi ini mendukung visualisasi gestur dan memberikan umpan balik kepada pengguna saat gestur dikenali.

4.3 Deteksi Gesture Tangan dan Interpretasi

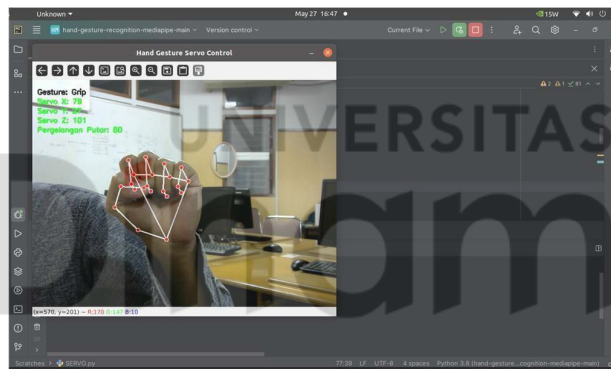
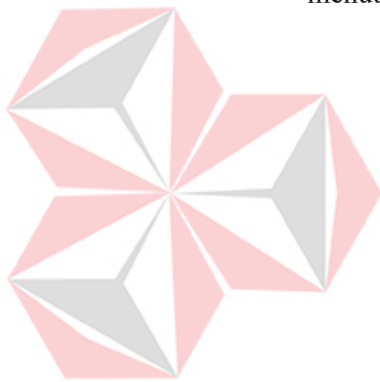
Hasil ekstraksi landmark dari MediaPipe tidak secara langsung menyatakan jenis gesture tangan. Oleh karena itu, sistem harus melakukan interpretasi berdasarkan posisi relatif antara titik-titik tersebut. Proses ini dilakukan oleh fungsi `detect_gesture()` yang memiliki logika klasifikasi sebagai berikut:

- Gesture "Open/Move": Diidentifikasi saat keempat jari (telunjuk, tengah, manis, dan kelingking) memiliki posisi y (vertikal) yang lebih kecil daripada buku-buku jari masing-masing. Ini menunjukkan bahwa tangan dalam kondisi terbuka, biasanya digunakan sebagai perintah membuka gripper atau melepas objek yang terlihat seperti pada Gambar 4.6 dibawah.



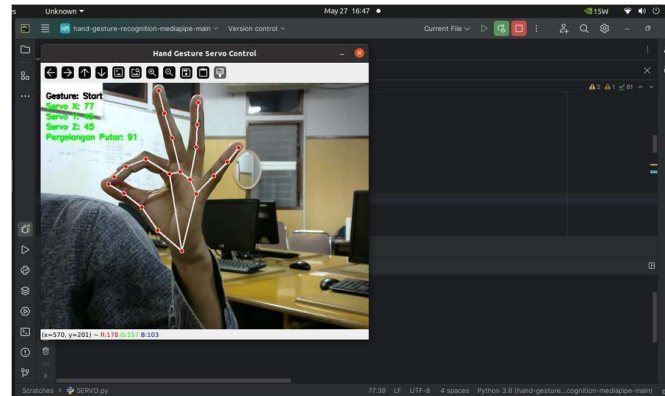
Gambar 4. 7 Pose Open/Move

- Gesture "Grip" (Mengepal): Kebalikannya dari "Open", semua ujung jari memiliki posisi y lebih besar daripada pangkalnya. Ini menandakan tangan dalam posisi menggenggam dan akan diterjemahkan sebagai perintah untuk menutup gripper yang terlihat seperti pada Gambar 4.8 dibawah ini.



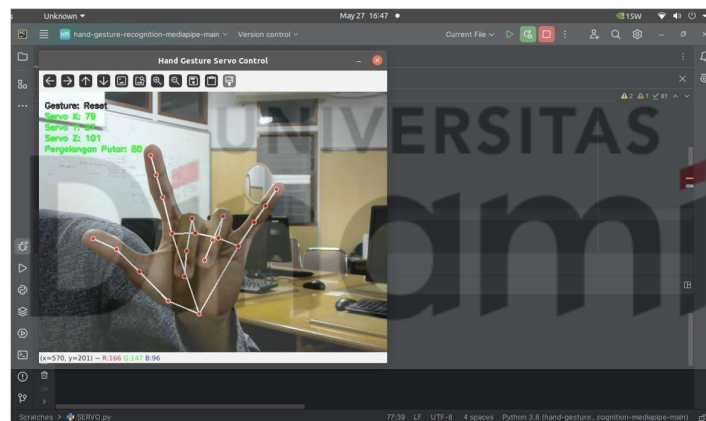
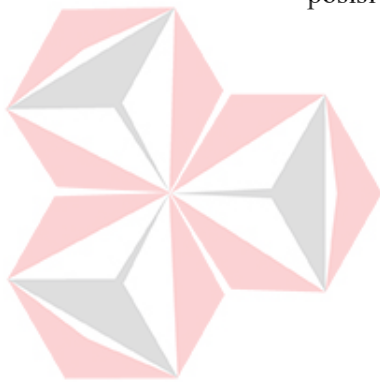
Gambar 4. 8 Pose Mengepal

- Gesture "Start" (OK Gesture): Terjadi ketika ujung ibu jari dan ujung telunjuk sangat dekat (jarak < 0.05) dan ketiga jari lainnya berada dalam posisi terbuka. Gesture ini digunakan sebagai isyarat awal untuk mengaktifkan kendali robot seperti pada Gambar 4.9 dibawah.



Gambar 4. 9 Pose OK

- Gestur "Reset" (Metal): Terdeteksi saat hanya telunjuk dan kelingking dalam posisi terbuka, sedangkan jari tengah dan manis menutup. Gestur ini berfungsi sebagai *safety trigger* untuk mengembalikan semua servo ke posisi awal atau default seperti pada Gambar 4.10.



Gambar 4. 10 Pose Metal

Proses ini melibatkan perhitungan jarak Euclidean antara titik-titik dan perbandingan nilai y dari setiap landmark. Akurasi deteksi gesture sangat dipengaruhi oleh stabilitas posisi tangan, pencahayaan, serta kejernihan gambar.

Sebagai contoh, gestur “Start” atau dikenal sebagai gesture OK dikenali apabila ujung ibu jari dan ujung jari telunjuk memiliki jarak *euclidean* yang sangat kecil, menandakan kedua jari tersebut saling menyentuh, sementara tiga jari lainnya dalam posisi terbuka.

Gestur *Open Hand* diidentifikasi ketika semua jari (telunjuk, jari tengah, jari manis, dan kelingking) berada dalam posisi terbuka, yang dapat dikenali dengan

melihat posisi ujung jari lebih tinggi (lebih kecil pada koordinat y) dibandingkan dengan titik buku jarinya. Sebaliknya, gestur mengepal dikenali ketika seluruh ujung jari berada di bawah posisi buku jarinya, menandakan jari-jari dalam kondisi menekuk. Sedangkan gestur *Reset*, yang menyerupai simbol “metal”, dikenali ketika hanya jari telunjuk dan kelingking dalam posisi terbuka, sedangkan dua jari tengah lainnya menekuk, dan ibu jari tidak menjadi parameter utama.

Seluruh proses ini terjadi secara berulang pada setiap frame video dalam loop utama sistem, sehingga mampu memberikan respons secara waktu nyata (real-time). Efisiensi MediaPipe dalam melakukan deteksi dan pelacakan ini memungkinkan sistem dapat berjalan secara lancar meskipun pada perangkat dengan sumber daya komputasi yang terbatas seperti Jetson Orin Nano. Hal ini menjadikan MediaPipe sangat ideal digunakan dalam proyek kerja praktik ini sebagai landasan utama dalam pengenalan gesture tangan berbasis citra digital.

Citra yang diproses ditampilkan secara langsung ke pengguna menggunakan antarmuka grafis yang dibuat dengan bantuan OpenCV. Informasi seperti garis koneksi antar landmark, label gesture, dan status servo juga ditampilkan agar pengguna dapat memantau sistem secara visual.

Selain visualisasi, OpenCV juga digunakan untuk debugging selama pengembangan, sehingga setiap perubahan gesture dapat terlihat dan diverifikasi secara langsung melalui jendela video.

Sistem mendeteksi gesture tangan berdasarkan posisi landmark jari yang diperoleh melalui MediaPipe. Setiap titik landmark memiliki indeks tertentu, di mana ujung ibu jari berada pada indeks 4, telunjuk pada 8, dan seterusnya. Cuplikan kode berikut menunjukkan bagian awal dari fungsi `detect_gesture()`, yaitu pengambilan titik-titik koordinat penting pada jari, serta perhitungan jarak antara ibu jari dan telunjuk yang digunakan untuk mendeteksi gesture “Start” (OK).

```

38 # === Inisialisasi MediaPipe ===
39 mp_hands = mp.solutions.hands
40 hands = mp_hands.Hands(max_num_hands=1, min_detection_confidence=0.7)
41 mp_draw = mp.solutions.drawing_utils
42
43 # Variabel global (di luar loop)
44 prev_angle_z = 60 # Sudut awal servo
45 alpha = 0.1 # Semakin kecil -> semakin halus
46
47 # Fungsi deteksi gesture
48 def detect_gesture(hand_landmarks):
49     lm = hand_landmarks.landmark
50     thumb_tip = lm[4]
51     index_tip = lm[8]
52     middle_tip = lm[12]
53     ring_tip = lm[16]
54     pinky_tip = lm[20]
55     index_mcp = lm[5]
56     middle_mcp = lm[9]

```

Gambar 4. 11 Kode inisialisasi mp dan deteksi gestur

```

57     ring_mcp = lm[13]
58     pinky_mcp = lm[17]
59
60     # Jarak ibu jari & telunjuk
61     thumb_index_dist = ((thumb_tip.x - index_tip.x) ** 2 + (thumb_tip.y - index_tip.y) ** 2) ** 0.5
62

```

Gambar 4. 12 Lanjutan kode deteksi gestur

Titik-titik landmark ini menjadi dasar untuk menganalisis gesture karena mereka menggambarkan bentuk dan posisi jari tangan secara relatif terhadap sendi utama masing-masing jari. Setelah titik landmark diperoleh, sistem melakukan analisis logis terhadap posisi jari tangan untuk menentukan gesture yang sedang dilakukan pengguna. Gesture seperti "*Open Hand*", "*Grip*", "*Start(OK)*", dan "*Reset*" dikenali berdasarkan kondisi vertikal jari (y-axis) terhadap sendinya masing-masing. Logika rule-based ini digunakan karena lebih ringan dan cocok untuk pemrosesan real-time tanpa memerlukan model machine learning.

```

63 # Gesture: Open Hand
64 if (index_tip.y < index_mcp.y and middle_tip.y < middle_mcp.y and
65     ring_tip.y < ring_mcp.y and pinky_tip.y < pinky_mcp.y):
66     return "Open"
67
68 # Gesture: Grip (Fist)
69 if (index_tip.y > index_mcp.y and middle_tip.y > middle_mcp.y and
70     ring_tip.y > ring_mcp.y and pinky_tip.y > pinky_mcp.y):
71     return "Grip"
72
73 # OK
74 if (thumb_index_dist < 0.05 and
75     middle_tip.y < middle_mcp.y and
76     ring_tip.y < ring_mcp.y and
77     pinky_tip.y < pinky_mcp.y and mulai == 0):
78     return "Start"
79

```

Gambar 4. 13 Kode deteksi berbagai gestur jari tangan

```

80  # Reset (metal sign 🖐️)
... 81  if (index_tip.y < index_mcp.y and
82      pinky_tip.y < pinky_mcp.y and
83      middle_tip.y > middle_mcp.y and
84      ring_tip.y > ring_mcp.y):
85      return "Reset"
86
87  return "Unknown"
88

```

Gambar 4. 14 Lanjutan kode gestur jari tangan

Gambar 4.13 tersebut merupakan potongan kode fungsi `detect_gesture()` yang berisi logika klasifikasi gesture berdasarkan posisi vertikal dan jarak antar titik-titik landmark tangan. Jika tidak ada kondisi yang terpenuhi, sistem akan mengembalikan nilai “Unknown” sebagai fallback, yang menunjukkan bahwa gesture tangan pengguna belum dikenali atau tidak valid.

4.4 Kendali Servo dengan PCA9685

Setelah gestur tangan berhasil dikenali melalui proses klasifikasi berbasis koordinat landmark, sistem kemudian melanjutkan ke tahap konversi gestur menjadi sinyal kendali motor. Dalam konteks ini, gesture yang terdeteksi tidak langsung menggerakkan aktuator, melainkan terlebih dahulu diterjemahkan menjadi nilai sudut tertentu yang sesuai dengan karakteristik pergerakan motor servo. Masing-masing gesture yang dikenali seperti “Open”, “Grip”, “Start”, dan “Reset” telah dipetakan ke dalam nilai sudut spesifik yang merepresentasikan posisi akhir dari motor servo yang dikendalikan.

Untuk menghasilkan sinyal *PWM* (*Pulse Width Modulation*) yang dibutuhkan oleh motor servo, sistem memanfaatkan modul PCA9685, yakni sebuah kontroler PWM 16-channel berbasis I2C yang umum digunakan dalam sistem robotika. Modul ini mampu menghasilkan sinyal PWM 12-bit dengan tingkat resolusi hingga 4096 langkah (0–4095), yang menjadikannya sangat ideal untuk mengatur posisi sudut servo secara presisi. Komunikasi antara Jetson Orin Nano sebagai pusat pemrosesan dengan modul PCA9685 dilakukan melalui jalur I2C (Inter-Integrated Circuit), yang memungkinkan pengiriman sinyal digital antar perangkat dengan efisiensi tinggi dan latensi rendah.

```

19
20 # Fungsi konversi derajat ke PWM
21 def set_servo_angle(channel, angle):
22     pulse_min = 500
23     pulse_max = 2500
24     pulse = pulse_min + (angle / 180.0) * (pulse_max - pulse_min)
25     duty_cycle = int(pulse * 65535 / 20000)
26     pca.channels[channel].duty_cycle = duty_cycle
27

```

Gambar 4.15 Konversi derajat servo ke pwm

```

28 ## Pergerakan Servo Awal
29 set_servo_angle(channel: 2, angle: 90) # Servo 5 Pergelangan Putar
30 set_servo_angle(channel: 4, angle: 180) # Servo 4 Pergelangan Naik Turun
31 set_servo_angle(channel: 8, angle: 45) # Servo 2 Z Jauh Dekat
32 set_servo_angle(channel: 6, angle: 30) # Servo 3 Y Naik Turun
33 set_servo_angle(channel: 10, angle: 90) # Servo 1 X Kanan Kiri
34

```

Gambar 4.16 Pergerakan servo untuk arm robot

Penggunaan modul ini dimediasi oleh pustaka `adafruit-circuitpython-pca9685`, yaitu pustaka Python resmi dari Adafruit yang dirancang untuk menyederhanakan proses konfigurasi dan pengendalian modul PCA9685. Dengan pustaka ini, pengaturan frekuensi PWM, penentuan nilai duty cycle, serta alokasi kanal PWM untuk masing-masing motor dapat dilakukan secara efisien melalui skrip Python tanpa harus mengakses level register secara manual.

Dalam implementasi sistem, gesture yang dikenali akan dipetakan ke dalam sudut tertentu (misalnya 30° untuk “Open” dan 100° untuk “Grip” pada motor gripper), lalu nilai sudut tersebut dikonversi menjadi sinyal PWM menggunakan fungsi `set_servo_angle()`. Fungsi ini bertugas menghitung nilai pulsa (dalam satuan mikrodetik) dari sudut derajat, kemudian menerjemahkannya ke dalam nilai 16-bit duty cycle yang sesuai dengan spesifikasi PCA9685. Konversi ini biasanya dilakukan dalam rentang pulsa 500–2500 mikrodetik, yang secara umum merepresentasikan rentang pergerakan 0° hingga 180° pada servo standar.

Secara teknis, setiap kanal PWM pada PCA9685 dikonfigurasi untuk mengendalikan motor servo tertentu yang mewakili satu sumbu gerakan pada robot arm. Misalnya, kanal 15 digunakan untuk motor gripper, kanal 10 untuk gerakan horizontal (sumbu X), dan kanal 6 untuk sumbu Z. Ketika gesture tertentu dikenali, sistem akan secara otomatis mengirimkan perintah PWM ke kanal yang bersangkutan, sehingga motor yang dituju bergerak sesuai dengan nilai sudut hasil interpretasi gesture.

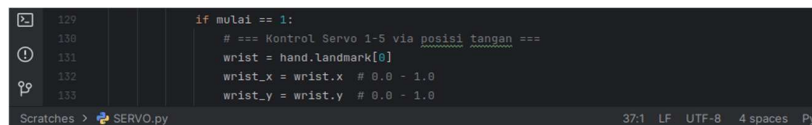
Keunggulan penggunaan PCA9685 dalam proyek ini adalah

kemampuannya untuk mengendalikan banyak motor secara paralel dengan presisi tinggi tanpa membebani sumber daya pemrosesan utama pada Jetson Orin Nano. Dengan demikian, kendali robot arm dapat dilakukan secara real-time dan responsif, seiring dengan gesture yang diberikan oleh pengguna.

4.5 Kendali Servo Berdasarkan Gesture dan Posisi

Pada sistem kendali robot arm berbasis gesture ini, pengenalan gesture saja belum cukup untuk menghasilkan kontrol pergerakan yang dinamis dan fleksibel. Oleh karena itu, sistem tidak hanya mengklasifikasikan gesture statis (seperti Open atau Grip), tetapi juga mengevaluasi posisi tangan secara kontinu berdasarkan koordinat landmark dari MediaPipe. Ketika sistem berada dalam kondisi aktif (ditandai dengan variabel mulai == 1), maka setiap pergerakan tangan pengguna akan langsung diterjemahkan ke dalam gerakan servo secara real-time. Implementasi ini memungkinkan robot arm untuk mengikuti arah dan orientasi tangan pengguna dengan tingkat presisi yang cukup tinggi.

Terdapat beberapa parameter posisi yang diekstrak dari landmark tangan, khususnya titik wrist (pergelangan tangan) dan titik jari tengah (landmark[9]). Dari titik-titik tersebut, sistem menghitung orientasi dan posisi tangan pengguna dalam tiga sumbu ruang utama: X (kanan-kiri), Y (atas-bawah), dan Z (maju-mundur), serta rotasi pergelangan tangan seperti pada Gambar 4.17, Gambar 4.18, dan Gambar 4.19 di bawah.



```

129 if mulai == 1:
130     # === Kontrol Servo 1-5 via posisi tangan ===
131     wrist = hand.landmark[0]
132     wrist_x = wrist.x # 0.0 - 1.0
133     wrist_y = wrist.y # 0.0 - 1.0
  
```

Scratches > SERVO.py 37:1 LF UTF-8 4 spaces Py

Gambar 4. 17 Kode kontrol servo dengan hand landmark



```

134 # Menghitung t
135 r_wrist = wrist_x
136 wrist_x = r_wrist * 1000000 # konversi ke nilai positif kecil
137
138 # min = 1.0
139 # max = 4.0
140 # t = 0.0
141
142 if wrist_x > x_max:
143     t = x_max
144 if wrist_x < x_min:
145     t = x_min
146 if (wrist_x > x_max) & (wrist_x < x_min):
147     t = wrist_x
148
149 target_angle = int((r_wrist - x_min) / (x_max - x_min) * 180)
150 target_angle = max(0, min(180, target_angle))
151
152 # Smoothing: low-pass filter
153 angle_x = int(alpha * target_angle + (1 - alpha) * prev_angle_x)
154 prev_angle_x = angle_x
155
156 # Pergerakan pergelangan
157 # Gerakan ke dalam
158 x1, y1 = int(wrist_x * w), int(wrist_y * h)
159 x2, y2 = int(hand_landmark[9].x * w), int(hand_landmark[9].y * h)
  
```

Scratches > SERVO.py 37:1 LF UTF-8 4 spaces Py

Gambar 4. 18 Lanjutan kode kontrol servo

```

161
162 # Hitung sudut rotasi telapak tangan
163 dx = x2 - x1
164 dy = y2 - y1
165 angle_rad = math.atan2(dy, dx)
166 angle_deg = int(math.degrees(angle_rad))
167 angle_deg = angle_deg * -1
168 if angle_deg < 0 :
169     angle_deg = prev_angle_deg
170 prev_angle_deg = angle_deg
171
172 angle_x = int(wrist.x * 180)
173 angle_y = int((1 - wrist.y) * 80) + 20
174 angle_z = int(wrist.z * 180)
175
176 # Pergerakan Servo
177 set_servo_angle(channel=2, angle_deg) # Servo 5 Pergelangan Putar
178 set_servo_angle(channel=4, angle_z) # Servo 4 Pergelangan Naik Turun
179
180 set_servo_angle(channel=6, angle_z) # Servo 2 Z Jauh Dekat // Belum Sempurna
181 # set_servo_angle(8, 90) # Servo 2 Z Jauh Dekat // Sementara
182
183 set_servo_angle(channel=8, angle_y) # Servo 3 Y Naik Turun
184 set_servo_angle(channel=10, angle_x) # Servo 1 X Kanan Kiri
185
186 # Tampilkan hasil

```

Gambar 4. 19 Kode perhitungan sudut rotasi dan Gerak servo

Pertama, pada sumbu X (horizontal), sistem membaca nilai `wrist.x` yang berkisar antara 0.0 hingga 1.0, di mana 0.0 menunjukkan posisi tangan paling kiri dalam frame kamera, dan 1.0 menunjukkan posisi paling kanan. Nilai ini dikalibrasikan ke dalam rentang sudut servo, yaitu antara 0 hingga 180 derajat. Semakin ke kanan tangan pengguna bergerak dalam citra, maka semakin besar nilai sudut yang dikirimkan ke motor servo kanal X.

Kedua, untuk sumbu Y (vertikal), sistem memanfaatkan nilai `wrist.y`, yang juga memiliki rentang antara 0.0 hingga 1.0. Nilai ini kemudian dikonversi menjadi sudut servo dengan skala yang disesuaikan agar tetap berada dalam rentang operasional servo, umumnya antara 20 hingga 100 derajat. Pergerakan tangan ke atas akan menyebabkan sudut servo mengecil, sedangkan pergerakan ke bawah meningkatkan sudut, sesuai dengan arah perubahan posisi vertikal tangan dalam koordinat citra.

Ketiga, pada sumbu Z (kedalaman), digunakan nilai `wrist.z` dari landmark tangan yang menunjukkan jarak relatif tangan terhadap kamera. Nilai ini bersifat negatif atau sangat kecil, sehingga sistem mengalikannya dengan faktor pengali tertentu agar menjadi nilai positif dalam skala yang dapat diinterpretasikan. Setelah dikalibrasi, nilai tersebut dikonversi ke dalam bentuk sudut dengan pendekatan linier, yaitu memetakan rentang jarak ke dalam rentang sudut servo (misalnya 45–110 derajat). Untuk menghindari perubahan sudut yang tiba-tiba akibat fluktuasi nilai landmark, digunakan teknik penyaringan menggunakan *low-pass filter*, di

mana nilai sudut servo dihitung sebagai gabungan antara nilai sebelumnya dan nilai baru dengan bobot tertentu. Hal ini menghasilkan pergerakan servo yang lebih halus dan stabil.

Terakhir, untuk rotasi pergelangan tangan, sistem melakukan penghitungan sudut rotasi berdasarkan perbandingan antara posisi titik wrist dengan titik landmark jari tengah (landmark[9]). Dengan menggunakan fungsi trigonometri atan2 , diperoleh sudut arah antara dua titik tersebut dalam satuan radian, yang kemudian dikonversi ke derajat. Sudut ini mencerminkan orientasi tangan pengguna (misalnya posisi tangan memutar ke kiri atau ke kanan), dan digunakan untuk mengatur rotasi pada sumbu tertentu dari lengan robot.

Seluruh perhitungan ini dilakukan secara real-time pada setiap frame video, sehingga sistem dapat menangkap gerakan tangan yang halus maupun cepat. Integrasi antara gesture dan posisi memungkinkan robot arm tidak hanya melakukan perintah diskret (seperti buka/tutup gripper), tetapi juga pergerakan kontinu mengikuti arah dan orientasi tangan, yang menjadikan sistem ini lebih interaktif dan intuitif digunakan dalam skenario laboratorium atau demonstrasi.

4.6 Antarmuka Visual dan Monitoring Sistem

Dalam sistem interaktif berbasis visi komputer, antarmuka visual memainkan peran yang sangat penting dalam memberikan umpan balik langsung kepada pengguna, sekaligus sebagai alat bantu dalam proses pengembangan dan debugging. Dalam proyek ini, pustaka OpenCV (Open Source Computer Vision Library) tidak hanya dimanfaatkan untuk mengakuisisi dan memproses citra dari kamera, tetapi juga berfungsi sebagai sarana visualisasi output sistem secara waktu nyata (*real-time visualization interface*).

Antarmuka yang dikembangkan dirancang untuk menampilkan berbagai informasi yang relevan dalam satu jendela video yang terus diperbarui secara dinamis. Pertama, sistem menampilkan citra kamera secara langsung dalam bentuk video streaming, di mana tangan pengguna yang menjadi objek utama dapat diamati secara visual. Pada citra ini, landmark tangan yang dihasilkan oleh MediaPipe ditampilkan dalam bentuk titik-titik kecil yang saling terhubung oleh garis, menggambarkan struktur tulang dan sendi tangan. Visualisasi ini membantu pengguna memahami bagian-bagian tangan yang sedang dianalisis oleh sistem.


```

109
110 # == Baca Gesture ==
111 gesture = detect_gesture(hand)
112 gesture_text = f"Gesture: {gesture}"
113
114 # == Kontrol Servo 0 via Gesture ==
115 if gesture == "Grip" and mulai == 1:
116     set_servo_angle(channel: 15, angle: 100)
117 elif gesture == "Open" and mulai == 1:
118     set_servo_angle(channel: 15, angle: 30)
119 elif gesture == "Reset":
120     set_servo_angle(channel: 2, angle: 90) # Servo 5 Pergelangan Putar
121     set_servo_angle(channel: 4, angle: 180) # Servo 4 Pergelangan Naik Turun
122     set_servo_angle(channel: 6, angle: 45) # Servo 2 Z Jauh Dekat
123     set_servo_angle(channel: 8, angle: 30) # Servo 3 Y Naik Turun
124     set_servo_angle(channel: 10, angle: 90) # Servo 1 X Kanan Kiri
125     mulai = 0
126 elif gesture == "Start":
127     mulai = 1
128

```

Gambar 4. 20 Kode detect servo

Kedua, sistem juga memberikan label teks yang menunjukkan jenis gesture yang sedang dikenali, misalnya “Open”, “Grip”, “Start”, atau “Reset”. Label ini ditampilkan secara overlay di atas citra video, sehingga pengguna dapat mengetahui secara instan apakah gesture yang mereka tunjukkan telah dikenali dengan benar oleh sistem. Umpan balik semacam ini sangat penting untuk menjaga keterlibatan pengguna serta memberikan rasa kontrol terhadap robot arm yang sedang dikendalikan.

Selain label gesture, antarmuka visual juga menyertakan informasi numerik terkait sudut servo, yang ditampilkan sebagai teks overlay pada bagian bawah atau samping layar. Informasi ini mencakup nilai-nilai aktual sudut servo pada masing-masing sumbu (X, Y, Z), termasuk rotasi pergelangan dan posisi gripper. Data ini dihasilkan dari hasil kalkulasi gesture dan posisi tangan, lalu dikirim ke motor servo melalui PCA9685. Dengan demikian, pengguna dapat melihat hubungan langsung antara posisi tangan dan gerakan servo, baik dalam bentuk visual (gerakan fisik) maupun digital (angka sudut servo).

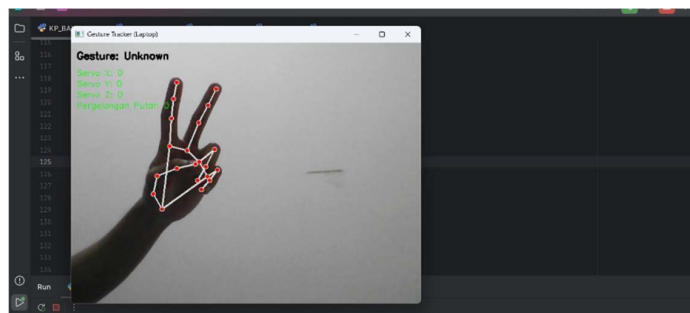
Fungsi monitoring ini tidak hanya berguna bagi pengguna akhir, tetapi juga sangat membantu dalam proses pengembangan dan pengujian sistem. Selama proses debugging, pengembang dapat mengamati apakah gesture dikenali dengan benar, apakah nilai sudut servo masuk dalam batas yang diharapkan, dan apakah ada anomali dalam pergerakan motor. Dengan demikian, antarmuka visual berbasis OpenCV tidak hanya bersifat kosmetik, melainkan juga memiliki nilai fungsional tinggi sebagai alat pengawasan, validasi, dan pelatihan pengguna terhadap sistem gesture-to-motion ini.

4.7 Evaluasi Respons Sistem dan Akurasi Gesture

Untuk menilai kinerja dan keandalan sistem kendali robot arm berbasis gesture tangan yang dikembangkan, dilakukan serangkaian pengujian yang terstruktur dan sistematis. Evaluasi ini bertujuan untuk mengukur responsivitas sistem terhadap gestur pengguna, tingkat akurasi dalam mengenali gesture tertentu, serta kestabilan dalam pengendalian motor servo yang menjadi aktuator utama dari robot arm.

Pengujian dilakukan di lingkungan laboratorium dengan pengaturan kondisi yang terkendali, yaitu pencahayaan buatan yang stabil, latar belakang berwarna netral (biasanya putih atau abu-abu), serta tanpa keberadaan objek bergerak di sekitar tangan pengguna. Hal ini dilakukan untuk meminimalisir gangguan visual yang dapat mempengaruhi proses deteksi landmark tangan dan akurasi gesture recognition.

Salah satu parameter penting yang diukur adalah waktu respons sistem, yaitu selang waktu antara saat gesture dikenali oleh algoritma deteksi hingga motor servo merespons dengan mengubah posisinya sesuai perintah. Berdasarkan hasil pengamatan, waktu respons rata-rata sistem berada dalam rentang 70 hingga 90 milidetik, tergantung pada jenis gesture yang diberikan serta tingkat pencahayaan saat pengujian berlangsung. Gesture sederhana seperti “Open” dan “Grip” cenderung menghasilkan waktu respons yang lebih cepat, sementara gesture yang lebih kompleks seperti “Reset” memerlukan pemrosesan sedikit lebih lama akibat konfigurasi jari yang lebih sulit dikenali secara tepat.



Gambar 4. 21 Hasil percobaan gesture lain

Selain respons, aspek akurasi pengenalan gesture juga menjadi fokus utama evaluasi. Sistem menunjukkan performa yang sangat baik untuk gesture dengan struktur tangan yang jelas dan konsisten, seperti “Open” (semua jari terbuka) dan “Grip” (semua jari tertutup), dengan tingkat akurasi pengenalan melebihi 90%

dalam 50 percobaan terpisah untuk masing-masing gesture. Sementara itu, gesture seperti “Start” (OK gesture) dan “Reset” (metal gesture) memperlihatkan penurunan akurasi, terutama karena ketergantungan pada posisi ibu jari dan kelingking yang lebih sulit dipetakan secara konsisten. Gesture “Reset”, misalnya, rentan terhadap kesalahan karena beberapa pengguna cenderung tidak menjaga kelingking dalam posisi terbuka sepenuhnya, atau terjadi tumpang tindih antara jari.

Dari sisi kendali aktuator, sistem menunjukkan stabilitas gerakan servo yang memadai, terutama setelah penerapan teknik penyaringan menggunakan *low-pass filter* pada nilai sudut. Filter ini berguna untuk meredam lonjakan atau fluktuasi sinyal PWM yang disebabkan oleh perubahan kecil pada posisi landmark, yang jika tidak disaring dapat menyebabkan gerakan servo yang tidak stabil atau terlalu sensitif. Dengan adanya filter ini, perubahan sudut servo menjadi lebih halus dan mengikuti pergerakan tangan secara lebih natural.

Secara keseluruhan, hasil evaluasi membuktikan bahwa sistem mampu bekerja secara real-time dengan tingkat keakuratan dan kestabilan yang tinggi untuk gesture umum, dan masih menunjukkan performa yang dapat diterima untuk gesture kompleks. Evaluasi ini menunjukkan bahwa sistem memiliki potensi yang besar untuk diterapkan pada skenario interaktif yang membutuhkan respons cepat dan pengenalan gesture yang andal, seperti dalam bidang edukasi robotik, asistensi pengguna disabilitas, dan antarmuka kendali mesin yang bersifat nirsentuh.

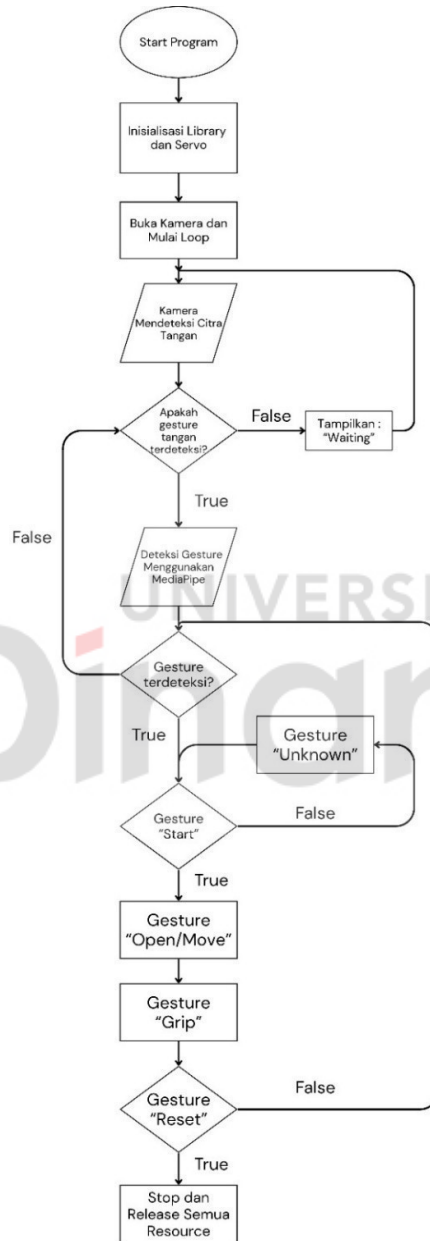
4.8 Diagram Alur Sistem dan Logika Kendali Gesture

Untuk memperjelas proses kerja sistem kendali robot arm berbasis gesture tangan, berikut disajikan diagram alur (flowchart) yang menggambarkan urutan logika program mulai dari akuisisi citra, pengenalan gesture, hingga eksekusi perintah kendali motor servo. Diagram ini berfungsi sebagai representasi visual dari skrip Python utama yang dijalankan pada Jetson Orin Nano, dan memberikan gambaran menyeluruh mengenai struktur pemrosesan sistem.

Program diawali dengan inisialisasi library seperti cv2, mediapipe, dan adafruit_pca9685, serta mengatur posisi awal servo motor. Setelah kamera dibuka dan aliran video dimulai, sistem akan terus membaca frame dalam loop.

Jika terdeteksi tangan oleh MediaPipe, sistem akan menjalankan fungsi

detect_gesture() untuk mengenali jenis gesture yang ditunjukkan. Jika gesture adalah “Start”, maka sistem masuk ke mode aktif (mulai = 1). Gesture “Reset” akan mengembalikan semua servo ke posisi default. Sedangkan “Grip” dan “Open” hanya dieksekusi jika status mulai = 1.



Gambar 4.22 Flowchart program

BAB V

PENUTUP

5.1 Kesimpulan

Berdasarkan hasil kerja praktik yang dilakukan di Laboratorium IoT Universitas Dinamika serta analisis terhadap implementasi sistem kendali robot *arm* berbasis *image processing* untuk pengenalan *gesture* tangan secara *real-time*, maka dapat diambil beberapa kesimpulan sebagai berikut:

1. Sistem pengolahan citra digital berhasil dirancang dan diimplementasikan untuk mengenali gestur tangan secara *real-time*. Dengan menggunakan kombinasi pustaka OpenCV dan MediaPipe, sistem mampu mendeteksi dan melacak *landmark* tangan secara akurat. Proses deteksi gestur seperti "*Start(OK)*", "*Open*", "*Grip*", dan "*Reset (metal)*" berhasil dilakukan secara konsisten dalam lingkungan laboratorium dengan pencahayaan terkendali.
2. Integrasi *gesture recognition* dengan sistem kendali robot arm berhasil dilakukan secara fungsional. Hasil deteksi gestur yang dilakukan sistem digunakan untuk mengendalikan pergerakan motor servo melalui modul PCA9685. Setiap gestur dikonversi menjadi sinyal PWM yang sesuai dengan gerakan robot arm, seperti membuka dan menutup gripper atau mengatur posisi motor pada sumbu X, Y, dan Z.
3. Sistem menunjukkan respons yang cukup cepat dan akurasi cukup tinggi pada gestur yang sederhana. Waktu respons rata-rata sistem berkisar antara 70–90 milidetik. Gestur dengan konfigurasi tangan yang jelas seperti "*Open*" dan "*Grip*" memiliki akurasi lebih dari 90%, sementara gestur kompleks seperti "*Reset*" masih menghadapi beberapa tantangan dari sisi konsistensi pengguna.
4. Tujuan kerja praktik telah tercapai sesuai dengan rumusan masalah. Sistem dapat mengenali *gesture* tangan secara *real-time*, menghubungkan hasil deteksi dengan kendali fisik pada robot arm, serta memberikan hasil uji coba yang menunjukkan performa baik dalam hal akurasi dan kecepatan respon.

5.2 Saran

Adapun saran untuk pengembangan sistem dan penelitian selanjutnya adalah sebagai berikut:

1. Penambahan dukungan sensor tambahan seperti IMU atau sensor jarak dapat meningkatkan akurasi dalam memahami orientasi dan jarak tangan, terutama untuk penggunaan di lingkungan luar laboratorium yang tidak memiliki pencahayaan stabil.
2. Penerapan machine learning atau deep learning untuk klasifikasi gesture dapat meningkatkan akurasi dan fleksibilitas pengenalan gesture tangan yang lebih kompleks dan bervariasi.
3. Optimalisasi performa sistem pada perangkat berdaya rendah seperti Jetson Nano sebaiknya terus ditingkatkan dengan memperhatikan efisiensi pemrosesan citra, agar sistem dapat berjalan lancar meskipun diimplementasikan dalam platform embedded yang terbatas.
4. Penerapan sistem dalam skenario nyata, seperti kontrol perangkat rumah tangga, asisten robotik, atau alat bantu bagi penyandang disabilitas, dapat menjadi pengembangan lanjutan yang bermanfaat secara sosial dan praktis.
5. Perluasan jenis gesture yang dikenali, misalnya dengan mengenali gesture dinamis (gerakan tangan seperti menggeser, melambaikan tangan, dsb.) akan menjadikan sistem lebih interaktif dan sesuai dengan kebutuhan aplikasi masa depan.

Dengan adanya sistem yang telah dikembangkan ini, diharapkan dapat menjadi dasar bagi pengembangan teknologi interaksi manusia dan mesin yang lebih alami dan efisien, serta sebagai referensi pembelajaran lanjutan dalam bidang pengolahan citra dan robotika.

DAFTAR PUSTAKA

- Adafruit. (2023). *Adafruit CircuitPython PCA9685 Library*. Retrieved from https://github.com/adafruit/Adafruit_CircuitPython_PCA9685
- Bradski, G., & Kaehler, A. (2008). *Learning OpenCV: Computer vision with the OpenCV library*. O'Reilly Media.
- Gonzalez, R. C., & Woods, R. E. (2018). *Digital image processing* (4th ed.). Pearson.
- JetBrains. (2023). *PyCharm: Python IDE for Professional Developers*. Retrieved from <https://www.jetbrains.com/pycharm/>
- Lugaresi, C., Tang, J., Nash, H., McClanahan, C., Ceze, L., & Taylor, J. (2019). MediaPipe: A framework for building perception pipelines. *arXiv preprint*, arXiv:1906.08172. <https://doi.org/10.48550/arXiv.1906.08172>
- Mitchell, T., Bianchi-Berthouze, N., & Williams, A. (2020). Recognizing and responding to human emotions in interactive systems. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 27(2), 1–34. <https://doi.org/10.1145/3377141>
- NXP Semiconductors. (2019). *PCA9685 16-channel, 12-bit PWM Fm+ I2C-bus LED controller*. Retrieved from <https://www.nxp.com/docs/en/data-sheet/PCA9685.pdf>
- Tanenbaum, A. S., & Bos, H. (2015). *Modern operating systems* (4th ed.). Pearson.
- Zhang, F., Bazarevsky, V., Vakunov, A., Tkachenka, A., Sung, G., Chang, C. L., & Grundmann, M. (2020). MediaPipe Hands: On-device real-time hand tracking. *arXiv preprint*, arXiv:2006.10214. <https://doi.org/10.48550/arXiv.2006.10214>