

**PENERAPAN MEDIAPIPE DAN OPENCV UNTUK MENDETEKSI
KESALAHAN GERAKAN PADA AKTIVITAS LATIHAN DI GYM
BERBASIS STREAMLIT**

(Studi Kasus : V-Squat Machine)

Skripsi ini Diajukan Sebagai Syarat Melaksanakan Kewajiban Studi Strata Satu (S1)
Program Studi Teknik Informatika



Dosen Pembimbing :

Fitri Mintarsih , M.Kom.
197406242007101001

Disusun Oleh:

Royan Sabila Rosyad Wahyudi

11210910000055

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS SAINS DAN TEKNOLOGI

UNIVERSITAS ISLAM NEGERI SYARIF HIDAYATULLAH JAKARTA

1446 H / 2025 M

LEMBAR PERSETUJUAN

PENERAPAN MEDIAPIPE DAN OPENCV UNTUK MENDETEKSI KESALAHAN GERAKAN PADA AKTIVITAS LATIHAN DI GYM BERBASIS STREAMLIT

Skripsi

Sebagai salah satu syarat untuk memperoleh gelar sarjana Komputer
(S.Kom)

Oleh:

Royan Sabila Rosyad Wahyudi

NIM: 11210910000055

Menyetujui,

Dosen Pembimbing 1



Fitri Mintarsih, M.Kom.
NIP. 197212232007102004

Dosen Pembimbing 2



Anif Hanifa Setianingrum, M.Sc.
NIP. 196410102024212001

PERNYATAAN ORISINALITAS

Dengan ini saya menyatakan bahwa:

1. Saya menyatakan bahwa skripsi ini merupakan hasil karya orisinal yang saya susun sendiri sebagai salah satu syarat untuk memperoleh gelar Sarjana Strata 1 (S1) di UIN Syarif Hidayatullah Jakarta.
2. Semua sumber yang saya gunakan dalam penulisan ini telah saya cantumkan sesuai dengan ketentuan akademik yang berlaku di UIN Syarif Hidayatullah Jakarta.
3. Jika di kemudian hari terbukti bahwa karya ini bukan hasil karya asli saya atau merupakan hasil jiplakan dari karya orang lain, maka saya bersedia menerima sanksi yang berlaku di UIN Syarif Hidayatullah Jakarta.

Tangerang, 17 Juni 2025



Royan Sabila Rosyad Wahyudi
11210910000055

ABSTRAK

Nama : Royan Sabilia Rosyad Wahyudi
Program Studi : Teknik Informatika
Judul : Penerapan Mediapipe dan OpenCV Untuk Mendeteksi Kesalahan Gerakan Pada Aktivitas Latihan di Gym
(Studi Kasus: V-Squat Machine)

Meningkatnya tren popularitas olahraga angkat beban di kalangan Gen Z yang sebagian dipengaruhi oleh media sosial seperti Instagram, TikTok, dan YouTube telah menciptakan fenomena baru dalam dunia kebugaran. Namun, tingginya minat masyarakat terhadap aktivitas gym tidak diimbangi dengan pemahaman yang memadai mengenai teknik gerakan yang benar, yang dapat menyebabkan cedera hingga 27% peserta gym dalam 6 bulan terakhir. Studi lain menunjukkan bahwa cedera paling sering terjadi pada bahu (25,2%), lutut (20,2%), dan punggung bawah (17,7%) pada aktivitas gym yang melibatkan bodybuilding, weightlifting, dan powerlifting. Penelitian ini mengembangkan sistem deteksi kesalahan gerakan squat pada V-Squat Machine menggunakan MediaPipe dan OpenCV yang diimplementasikan dalam aplikasi web Streamlit. Hasil penelitian pada empat subjek pengujii yang terdiri dari tiga instruktur ahli dan satu non-instruktur menunjukkan bahwa sistem mampu mendeteksi kesalahan gerakan dengan tingkat akurasi 100% pada kondisi pencahayaan optimal dan memberikan *feedback real-time* kepada pengguna. Evaluasi performa menggunakan metrik MPJPE (*Mean Per Joint Position Error*) menunjukkan performa sangat baik dengan nilai rata-rata 10.31-10.83 piksel pada kondisi terang. Penelitian ini berhasil mengembangkan solusi teknologi yang efektif untuk membantu pengguna gym dalam memperbaiki teknik latihan dan mengurangi risiko cedera melalui *feedback real-time* yang akurat dan mudah digunakan.

Kata kunci: Computer Vision, MediaPipe, OpenCV, Deteksi Gerakan, Squat, V-Squat Machine, MPJPE, Streamlit, Pose Estimation

ABSTRACT

Nama : Royan Sabila Rosyad Wahyudi
Program Studi : Teknik Informatika
Judul : Penerapan Mediapipe dan OpenCV Untuk Mendeteksi Kesalahan Gerakan Pada Aktivitas Latihan di Gym (Studi Kasus: V-Squat Machine)

The increasing trend of weightlifting popularity among Gen Z, partly influenced by social media platforms such as Instagram, TikTok, and YouTube, has created a new phenomenon in the fitness world. However, the high public interest in gym activities is not balanced with adequate understanding of proper movement techniques, which can cause injuries in up to 27% of gym participants in the last 6 months. Other studies show that injuries most commonly occur in the shoulder (25.2%), knee (20.2%), and lower back (17.7%) in gym activities involving bodybuilding, weightlifting, and powerlifting. This research develops a squat movement error detection system on V-Squat Machine using MediaPipe and OpenCV implemented in a Streamlit web application. The research results on four test subjects consisting of three expert instructors and one non-instructor show that the system can detect movement errors with 100% accuracy under optimal lighting conditions and provide real-time feedback to users. Performance evaluation using MPJPE (Mean Per Joint Position Error) metrics demonstrates excellent performance with average values of 10.31-10.83 pixels under bright conditions. This research successfully develops an effective technological solution to help gym users improve their exercise techniques and reduce injury risk through accurate and user-friendly real-time feedback.

Keywords: Computer Vision, MediaPipe, OpenCV, Movement Detection, Squat, V-Squat Machine, MPJPE, Streamlit, Pose Estimation

KATA PENGANTAR

Segala puji dan syukur kehadirat Allah SWT yang telah memberikan rahmat dan hidayah-Nya sehingga penulis dapat menyelesaikan skripsi dengan judul “**Penerapan Mediapipe dan OpenCV untuk Mendeteksi Kesalahan Gerakan Pada Aktivitas Latihan di Gym Berbasis Streamlit**”. Dalam penyusunan skripsi ini, penulis mendapatkan bantuan dari berbagai pihak. Oleh karena itu, penulis ingin mengucapkan terima kasih yang sebesar-besarnya kepada :

1. Allah SWT, atas segala rahmat, kasih sayang, dan petunjuk yang tak pernah putus, menghadirkan kekuatan di saat lemah dan harapan di setiap kekhawatiran.
2. Kedua orang tua dan keluarga yang selalu menjadi sumber inspirasi terbesar. Doa, dukungan, dan cinta kasih yang tulus dari kalian adalah energi utama yang

menguatkan penulis untuk terus melangkah, meski jalan seringkali terasa berat dan berliku.

3. Bapak Husni Teja Sukmana, S.T., M.Sc., Ph.D selaku dekan Fakultas Sains dan Teknologi UIN Syarif Hidayatullah Jakarta
4. Ibu Dr. Dewi Khairani M.Sc., selaku Ketua Program Studi Teknik Informatika dan Bapak Saepul Aripiyanto, M.Kom selaku Sekretaris Program Studi Teknik Informatika yang telah memberikan arahan dalam pelaksanaan studi.
5. Ibu Fitri Mintarsih, M.Kom. selaku Dosen Pembimbing Akademik (DPA) dan Dosen Pembimbing I yang telah dengan sabar membimbing, memberi masukan, dan membuka wawasan baru bagi penulis dalam menyusun skripsi ini.
6. Ibu Anif Hanifa Setianingrum, M.Sc. selaku Dosen Pembimbing 2 yang telah memberikan bimbingan dan arahan dalam pengerjaan skripsi ini.
7. Wildan Ihsan Wahyudi, selaku instruktor yang telah membimbing dan mengarahkan untuk memulai olahraga gym, selalu setia menemani setiap sesi latihan, menjadi teman diskusi, serta pendukung utama dalam proses pengambilan data dan studi kasus gerakan.
8. Diri saya sendiri, untuk segala keberanian, ketekunan, dan konsistensi dalam menghadapi setiap tantangan. Terima kasih telah percaya pada diri sendiri, bersedia berproses, dan tidak menyerah, bahkan ketika harus membagi waktu antara tugas akademik, bootcamp, kegiatan pengembangan diri, dan proyek freelance. Perjalanan ini mengajarkan bahwa setiap pencapaian besar selalu lahir dari keberanian untuk melangkah keluar dari zona nyaman.
9. Rekan-rekan seperjuangan, sahabat, dan semua pihak yang tidak dapat disebutkan satu per satu, yang telah memberikan dukungan moral, bantuan, dan semangat, baik secara langsung maupun tidak langsung. Kebersamaan dan dukungan kalian adalah anugerah yang sangat berarti.

Penulis sadar sepenuhnya bahwa skripsi ini masih jauh dari kata sempurna. Proses pembelajaran yang panjang ini membuat penulis semakin memahami arti penting kritik dan saran yang membangun. Penulis sangat mengharapkan masukan demi perbaikan karya ini ke depannya. Besar harapan penulis, semoga skripsi ini dapat memberikan manfaat, menjadi sumber inspirasi, dan kontribusi dalam pengembangan teknologi, khususnya pada bidang deteksi gerakan untuk aktivitas kebugaran di Indonesia.

Depok, 21 Januari 2025



Royan Sabila Rosyad Wahyudi

11210910000055

DAFTAR ISI

LEMBAR PERSETUJUAN.....	2
PERNYATAAN ORISINALITAS.....	3
ABSTRAK	4
ABSTRACT	5
KATA PENGANTAR.....	6
DAFTAR ISI.....	9
DAFTAR GAMBAR	12
DAFTAR TABEL	13
BAB I PENDAHULUAN	14
1.1 Latar Belakang	14
1.2 Identifikasi Masalah	15
1.3 Rumusan Masalah	15
1.4 Batasan Masalah	16
1.5 Tujuan Penelitian	16
1.6 Manfaat Penelitian	17
1.7 Metodologi Penelitian.....	17
1.7.1 Metode Pengumpulan Data.....	17
1.7.2 Metode Implementasi.....	18
1.8 Sistematika Penulisan.....	18
BAB II LANDASAN TEORI.....	20
2.1 Computer Vision.....	20
2.2 Pose Estimation	21
2.3 Mediapipe	22
2.3.1 BlazePose Landmark	23
2.4 OpenCV.....	24
2.5 Streamlit.....	25
2.6 Aktivitas Latihan di Gym	26
2.6.1 Pengertian Latihan Beban	26
2.6.2 Squat	27
2.6.3 Teknik dan Tahapan Gerakan V-Squat yang Benar.....	28
2.6.4 Rujukan Ilmiah sebagai Landasan Penetapan Threshold Feedback Squat	30
2.7 Angle Calculation.....	32
2.8 Mean Per Joint Position Error (MPJPE)	34

2.9 Studi Literatur Sejenis	36
BAB III METODOLOGI PENELITIAN	42
3.1. Metode Pengumpulan Data	43
3.2 Perangkat Penelitian.....	43
3.2. Metode Pengembangan Sistem	44
3.3.1 Analysis	44
3.3.2 Design.....	44
3.3.3 Implementation.....	45
3.3.4 Testing	46
3.4. Kerangka Penelitian.....	46
BAB IV IMPLEMENTASI.....	47
4.1 Analysis	47
4.1.1 State Diagram.....	48
4.1.2 Perhitungan Manual Sudut dan Thresholds	49
4.1.3 Perhitungan Manual Metrik <i>Mean Per Joint Position Error (MPJPE)</i>	55
4.2 Design	58
4.2.1 Desain Arsitektur Workflow Sistem.....	58
4.2.2 Flowchart Fitness Vision	59
4.2.3 Use Case Diagram	61
4.2.4 Activity Diagram.....	62
4.3 Implementasi	63
4.3.1 Penggunaan Model Mediapipe Pose.....	63
4.3.2 Penggunaan Landmark yang dibutuhkan.....	66
4.3.3 Perhitungan Sudut antar Landmark dan State Sequence Squat.....	67
4.3.4 Implementasi Thresholds Transisi Gerakan	69
4.3.5 Perhitungan Repetisi dan Feedback Visual.....	71
4.3.6 Mekanisme Inaktivitas (Reset Counter)	73
4.3.7 User Interface Streamlit	74
4.4 Testing	82
4.4.1 Pengujian Mode Upload Video (Analisis dari Video Rekaman)	82
4.4.2 Pengujian pada Mode Beginner & Pro.....	83
4.4.3 Skenario Pengujian pada Berbagai Kondisi	85
4.5 Evaluasi.....	88
BAB V HASIL DAN PEMBAHASAN	89
5.1 Hasil Uji Coba Sistem.....	89
5.2 Pengujian <i>Blackbox</i>	92

5.3 Evaluasi dengan MPJPE	95
BAB VI PENUTUP	99
6.1 Kesimpulan	99
6.2 Saran.....	99
DAFTAR PUSTAKA.....	101
LAMPIRAN	105

DAFTAR GAMBAR

Gambar 2. 1 BlazePose Landmarks	24
Gambar 2. 2 Grip Seated Cable Row	Error! Bookmark not defined.
Gambar 2. 3 Ilustrasi otot yang terlibat selama latihan Seated Cable Row ..	Error! Bookmark not defined.
Gambar 2. 4 Kontraksi otot fase s1 ke s2 (pull, trans)	Error! Bookmark not defined.
Gambar 2. 5 Exercise Return to Starting Position and.....	Error! Bookmark not defined.
Gambar 2. 6 BlazePose Landmarks on Face and Shoulders with Angle Calculation.....	34
Gambar 3. 1 Metode Waterfall.....	44
Gambar 3. 2 Alur Penelitian	47
Gambar 4. 1 State Transition Diagram pada Proses Deteksi Gerakan.....	48
Gambar 4. 2 Arsitektur Sistem Deteksi dan Analisis Gerakan Squat	58
Gambar 4. 3 Flowchart Fitness Vision	60
Gambar 4. 4 Use Case Diagram Fitness Vision.....	62
Gambar 4. 5 Activity Diagram Fitness Vision	62
Gambar 4. 6 Tampilan Homepage Navigasi Fitness Vision	76
Gambar 4. 7 Interface LIve Detection Page	77
Gambar 4. 8 User Interface Upload Video Fitness Vision	82
Gambar 4. 9 Preview Pengujian pada menu Upload Video	83
Gambar 4. 10 Preview Pengujian pada mode Beginner.....	84
Gambar 4. 11 Preview Pengujian pada Mode Pro	85

DAFTAR TABEL

Tabel 2.1 Studi Literatur	36
Tabel 2. 2 Perbandingan Penelitian Terdahulu	40
Tabel 4. 1 Tabel Skenario untuk Pengujian Blackbox Testing.....	86
Tabel 4. 2 Tabel Skenario Pengujian pada Instruktur Ahli.....	87

BAB I

PENDAHULUAN

1.1 Latar Belakang

Pada masa kini, perkembangan teknologi informasi telah mengalami kemajuan pesat, terutama dalam kemudahan akses informasi melalui media sosial. Platform seperti Instagram, YouTube, dan TikTok memainkan peran besar dalam membentuk gaya hidup sehat di kalangan Gen-Z (Likai Liu, 2024). Fenomena ini diperkuat dengan algoritma rekomendasi konten di media sosial, yang menghadirkan video terkait kebugaran di beranda pengguna. Pengaruh dari para influencer kebugaran yang membagikan aktivitas latihan mereka di media sosial semakin memperkuat tren ini, menjadikan gym tidak hanya sebagai sarana berolahraga, tetapi juga sebagai bagian dari gaya hidup urban.

Namun, meskipun minat terhadap olahraga di gym terus meningkat, masih banyak individu yang belum memahami teknik atau form gerakan yang benar saat berlatih. Kesalahan dalam melakukan form latihan dapat berdampak serius terhadap efektivitas latihan maupun kesehatan tubuh. Form yang kurang tepat dapat mengurangi efektivitas kerja otot yang ditargetkan dan, dalam jangka panjang, berpotensi menimbulkan cedera. Menurut (Ramirez et al., 2022) menegaskan bahwa kesalahan teknik dalam latihan dapat menyebabkan cedera seperti strain otot, cedera sendi, hingga gangguan pada tulang belakang.

Fakta di lapangan menunjukkan bahwa insiden cedera akibat latihan di gym bukanlah hal yang jarang terjadi. Menurut penelitian yang dilakukan oleh (Bukhary et al., 2023) melaporkan bahwa sekitar 27% peserta gym mengalami cedera terkait latihan beban dalam 6 bulan terakhir, dengan bagian tubuh paling umum yang terkena dampaknya adalah bahu (7,4%), lutut (4,6%), dan pergelangan tangan. Cedera-cedera ini sering kali disebabkan oleh teknik yang salah, repetisi berlebihan, atau kurangnya panduan saat berlatih. Studi lain yang lebih spesifik pada aktivitas gym yang melibatkan bodybuilding, weightlifting, dan powerlifting, menunjukkan bahwa cedera paling sering terjadi pada bahu (25,2%) , punggung bawah (17,7%) , dan lutut (20,2%) (Hetaimish et al., 2024). Data ini menunjukkan pentingnya pemantauan teknik latihan yang tepat guna meminimalisir risiko cedera.

Sebagai upaya untuk mengatasi persoalan tersebut, dibutuhkan solusi berbasis teknologi yang dapat membantu para pengguna gym dalam memantau sekaligus memperbaiki teknik latihan. Penelitian ini menawarkan metode computer vision untuk mendeteksi kesalahan pada form gerakan latihan gym, khususnya pada studi kasus gerakan V-Squat Machine. Dengan pemanfaatan OpenCV dan MediaPipe, sistem yang dirancang mampu menganalisis video input

maupun deteksi secara langsung (live detection) untuk mengidentifikasi kesalahan pada form gerakan. Selain itu, sistem juga dapat secara otomatis menghitung jumlah repetisi yang dilakukan dengan form yang benar.

Dalam implementasinya, MediaPipe digunakan untuk mendeteksi pose tubuh secara real-time, sedangkan OpenCV digunakan untuk menganalisis pergerakan. Pada studi kasus latihan Squat, sistem akan mendeteksi sudut gerakan tubuh pengguna dan memberikan *feedback* apabila ditemukan kesalahan. Dengan dukungan aplikasi berbasis Streamlit, sistem ini dapat diakses secara interaktif melalui perangkat pengguna, sehingga diharapkan dapat membantu memperbaiki teknik latihan, menurunkan risiko cedera, serta meningkatkan hasil latihan di gym.

1.2 Identifikasi Masalah

Berdasarkan latar belakang di atas, maka penulis ingin mengidentifikasi masalah sebagai berikut :

1. Ketidaktahuan tentang Form Latihan yang Benar

Banyak individu yang baru memulai aktivitas latihan di gym tidak mengetahui apakah gerakan yang mereka lakukan sudah sesuai dengan standar atau tidak. Ketidaktahuan ini sering kali disebabkan oleh kurangnya panduan langsung dari pelatih atau informasi yang memadai mengenai teknik latihan yang benar.

2. Risiko Cedera Akibat Kesalahan Gerakan

Kesalahan dalam form latihan dapat meningkatkan risiko cedera serius, seperti strain otot, dislokasi sendi, atau bahkan cedera tulang belakang. Hal ini dapat berdampak buruk pada keberlanjutan program latihan individu dan mengurangi efektivitas latihan.

3. Kurangnya Sistem Pemantauan Otomatis dalam Latihan

Saat ini, sebagian besar pemantauan latihan di gym masih dilakukan secara manual melalui observasi pribadi dengan melihat kaca (jika ada) pada saat latihan. Hal ini tidak selalu akurat dan dapat menyebabkan kelalaian dalam mendeteksi kesalahan gerakan.

1.3 Rumusan Masalah

Dari hasil identifikasi masalah yang telah dijelaskan oleh penulis, dapat disimpulkan rumusan masalah sebagai berikut:

1. Bagaimana merancang sistem berbasis computer vision menggunakan MediaPipe dan OpenCV untuk mendeteksi kesalahan gerakan pada latihan gym, khususnya pada latihan *Squat*?
2. Bagaimana sistem ini dapat menghitung jumlah repetisi gerakan yang dilakukan dengan form yang benar secara otomatis?
3. Bagaimana sistem ini dapat memberikan feedback real-time kepada pengguna untuk memperbaiki teknik latihan?

1.4 Batasan Masalah

Adapun batasan penelitian yang dilakukan dalam penelitian ini, yaitu :

1. Penelitian ini hanya berfokus pada pendekripsi kesalahan gerakan berdasarkan deviasi sudut sendi dari rentang ideal
2. Penelitian ini mengasumsikan kondisi ideal dalam pengambilan video, seperti pencahayaan yang cukup, latar belakang yang minim gangguan, dan subjek yang mengenakan pakaian yang kontras dengan latar belakang
3. Penelitian ini tidak menggunakan dataset besar untuk melatih model. Analisis kesalahan gerakan didasarkan pada satu sampel video per jenis latihan.
4. Hasil output dari penelitian ini adalah sebuah website berbasis streamlit yang mampu mendekripsi mendekripsi kesalahan gerakan pada latihan *squat*.
5. Cakupan subjek penelitian ini ditujukan untuk laki2 dengan berat badan ideal rentang umur 18-50 tahun

1.5 Tujuan Penelitian

Berdasarkan permasalahan masalah yang telah disebutkan sebelumnya, maka tujuan penelitian ini sebagai berikut :

1. Membangun sistem berbasis website pendekripsi kesalahan gerakan pada aktivitas latihan di gym menggunakan MediaPipe dan OpenCV yang diimplementasikan pada platform Streamlit.
2. Mendekripsi kesalahan gerakan pada studi kasus latihan seated cable row berdasarkan analisis sudut sendi menggunakan MediaPipe.

3. Menampilkan visualisasi hasil deteksi kesalahan gerakan secara real-time pada platform Streamlit, termasuk overlay kerangka tubuh (skeleton) dan informasi sudut sendi.

1.6 Manfaat Penelitian

Manfaat yang diperoleh dari penelitian ini, yaitu :

1. Memberikan *feedback* visual secara *real-time* mengenai kesalahan gerakan saat melakukan latihan, sehingga dapat membantu memperbaiki postur dan teknik latihan
2. Meningkatkan efektivitas latihan dan meminimalisir risiko cedera akibat kesalahan gerakan
3. Membuka peluang pengembangan sistem yang lebih canggih dengan menambahkan fitur-fitur seperti analisis kecepatan gerakan, penghitungan repetisi, dan personalisasi program latihan

1.7 Metodologi Penelitian

Untuk menyelesaikan penelitian ini penulis menggunakan metodologi sebagai berikut:

1.7.1 Metode Pengumpulan Data

Pendekatan yang digunakan dalam penelitian ini berbeda dari metode umum dalam computer vision yang biasanya mengandalkan dataset besar untuk melatih model. Penelitian ini menerapkan strategi pengumpulan data yang lebih terarah dan spesifik. Alih-alih menggunakan dataset eksternal, penelitian ini memanfaatkan rekaman video yang diambil langsung (primary data). Setiap rekaman video merepresentasikan satu sampel untuk tiap jenis latihan yang dianalisis. Pengambilan video dilakukan dengan kamera smartphone dalam kondisi pencahayaan yang optimal dan latar belakang yang sederhana untuk mengurangi gangguan selama proses deteksi. Subjek penelitian diminta melakukan gerakan latihan sesuai instruksi, dan video direkam dari sudut pandang terbaik agar pergerakan sendi yang relevan dapat terdokumentasi dengan jelas.

Penelitian ini lebih menitikberatkan pada kualitas dan relevansi data daripada jumlahnya. Data yang dikumpulkan tidak bertujuan untuk melatih model machine learning, melainkan dianalisis secara langsung. Kesalahan dalam

gerakan dievaluasi bukan berdasarkan pengenalan pola dari dataset besar, melainkan dengan menghitung deviasi sudut sendi yang terdeteksi dibandingkan dengan rentang nilai ideal yang telah ditetapkan untuk tiap fase gerakan dalam latihan. Pendekatan ini memungkinkan analisis yang lebih mendalam dan spesifik terkait biomekanika gerakan tanpa harus bergantung pada dataset yang luas dan beragam.

1.7.2 Metode Implementasi

Dalam pengembangan sistem, penelitian ini menggunakan metode Waterfall sebagai panduan. Metode ini terdiri dari empat tahapan utama: *Analysis*, *Design*, *Implementation*, dan *Deployment*. Tahap *Analysis* bertujuan untuk mengidentifikasi kebutuhan sistem, kemudian dilanjutkan dengan tahap *Design* yang berfokus pada perancangan arsitektur serta antarmuka aplikasi. Selanjutnya, tahap *Implementation* dilakukan dengan membangun sistem menggunakan MediaPipe, OpenCV, dan Streamlit. Terakhir, tahap *Deployment* melibatkan penerapan sistem ke Digital Ocean Cloud agar dapat diakses secara luas.

1.8 Sistematika Penulisan

Sistematika penulisan yang dilakukan dalam penelitian ini terdiri dari beberapa bagian, yaitu:

BAB I PENDAHULUAN

Pada bab ini membahas tentang hal umum dalam penelitian, seperti latar belakang dan identifikasi masalah ditemukan, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, metodologi penelitian, dan sistematika penulisan.

BAB II LANDASAN TEORI

Bab ini berisi penjelasan mengenai teori dan konsep yang mendasari penelitian serta materi yang diperlukan untuk mendukung pelaksanaan penelitian.

BAB III METODE PENELITIAN

Bab ini menguraikan metode yang digunakan untuk memperoleh data, teknik yang diterapkan dalam pengembangan sistem, serta kerangka berpikir yang digunakan dalam penyusunan penelitian ini.

BAB IV IMPLEMENTASI

Bab ini menjelaskan proses penerapan metode yang telah dipilih dalam menyelesaikan permasalahan yang menjadi fokus penelitian.

BAB V HASIL DAN PEMBAHASAN

Bab ini membahas hasil yang diperoleh dari proses implementasi dan eksperimen yang telah dilakukan pada bab sebelumnya, serta analisis terhadap hasil yang diperoleh.

BAB VI PENUTUP

Bab ini berisi kesimpulan dari hasil penelitian yang telah diperoleh serta jawaban atas permasalahan yang dirumuskan. Selain itu, bab ini juga memuat saran-saran yang dapat dijadikan referensi untuk penelitian selanjutnya

BAB II

LANDASAN TEORI

2.1 Computer Vision

Computer vision, atau penglihatan komputer, merupakan cabang multidisiplin dalam ilmu komputer yang berfokus pada pengembangan model dan algoritma yang memungkinkan komputer untuk meniru kemampuan manusia dalam memahami serta menginterpretasikan informasi visual dari dunia nyata (Ghazali Sulong & Martin Randles, 2023). Tujuan utama dari bidang ini adalah mengotomatisasi berbagai tugas yang biasanya dilakukan oleh manusia dalam mengolah citra dan video, seperti identifikasi objek, pengenalan pola, deteksi peristiwa, serta pemahaman adegan kompleks. Penerapan computer vision telah banyak dimanfaatkan dalam berbagai bidang, seperti sistem pengawasan keamanan, analisis citra medis, kendaraan otonom, robotika, interaksi manusia-komputer, hingga analisis konten multimedia (Gongjin Lan et al., 2023).

Library pendukung dalam bidang computer vision, seperti OpenCV, EmguCV, SimpleCV, dan AForge.NET, mempermudah proses pengembangan sistem berbasis pengolahan citra. Menurut Thierry (2018), secara umum proses Computer Vision terdiri dari empat tahap utama, yaitu *Image Acquisition*, *Image Processing*, *Image Analysis*, dan *Image Understanding*. Tahap awal adalah *Image Acquisition*, yaitu proses pengambilan gambar digital menggunakan sensor, seperti kamera. Setelah citra diperoleh, dilakukan tahap *Image Processing* yang bertujuan untuk meningkatkan kualitas citra, mengurangi gangguan (noise), serta memperjelas fitur-fitur penting guna mendukung analisis lebih lanjut. Langkah berikutnya adalah *Image Analysis*, di mana algoritma computer vision digunakan untuk mengenali dan mengukur karakteristik utama dari citra, seperti tepi, sudut, tekstur, dan warna. Data yang telah diekstraksi kemudian digunakan dalam tahap deteksi dan pengenalan objek untuk mengidentifikasi serta menentukan lokasi suatu objek dalam citra. Tahap terakhir adalah *Image Understanding*, di mana informasi visual yang telah dianalisis digunakan untuk mendapatkan pemahaman lebih mendalam terkait isi citra dan memungkinkan komputer untuk mengambil keputusan atau melakukan tindakan yang sesuai, tahap ini umumnya menerapkan metode kecerdasan buatan, seperti Artificial Neural Network (termasuk Multi-Layer Perceptron Network dan Radial Basis Function), Fuzzy System, Fuzzy Neural Network, Support Vector Machine (SVM), Principal Component Analysis (PCA), Evolutionary Algorithm (EA), serta metode terbaru seperti Deep Learning Network, yang mencakup Convolutional Neural Network (CNN) dan Deep Belief Network (DBN) (Thierry Bouwmans et al., 2018).

Dalam pengembangan aplikasi computer vision, berbagai library dan framework telah tersedia untuk mempermudah implementasi. OpenCV (Open Source Computer Vision Library) merupakan salah satu pustaka yang paling banyak digunakan karena kelengkapan algoritmanya serta dukungan komunitas yang luas. Selain itu, terdapat pustaka lain seperti Pillow (sebelumnya dikenal sebagai PIL) dan Scikit-image, serta framework deep learning seperti TensorFlow dan PyTorch yang memberikan kapabilitas lebih lanjut dalam tugas computer vision (Venkata Mahesh Babu Batta, 2024). Kemajuan deep learning dalam beberapa tahun terakhir telah meningkatkan performa sistem computer vision secara signifikan, terutama dalam tugas-tugas seperti klasifikasi gambar, segmentasi objek, dan deteksi objek secara real-time.

2.2 Pose Estimation

Pose estimation adalah cabang spesifik dalam computer vision yang berfokus pada deteksi dan lokalisasi bagian-bagian tubuh manusia (atau objek artikulasi lainnya) dalam citra atau video (Isha Chaudhary et al., 2023). Konsep utama dari *pose estimation* adalah untuk mengidentifikasi dan melacak *keypoints* atau titik-titik penting pada tubuh, seperti sendi-sendi utama (misalnya pergelangan tangan, siku, bahu, lutut, pergelangan kaki), wajah, dan bagian tubuh lainnya, untuk merekonstruksi "pose" atau konfigurasi tubuh dalam ruang 2D atau 3D (Meet Shah et al., 2023). Informasi pose yang diperoleh sangat berharga untuk berbagai aplikasi, termasuk analisis gerakan manusia, pengenalan aksi, interaksi manusia-komputer berbasis gestur, animasi karakter virtual, dan dalam konteks skripsi ini, deteksi kesalahan gerakan dalam aktivitas latihan fisik.

Pendekatan modern untuk pose estimation didominasi oleh metode berbasis deep learning, yang telah mencapai kemajuan signifikan dalam akurasi dan robustitas. Algoritma deep learning untuk pose estimation umumnya mengikuti dua tahapan: deteksi orang (person detection) dan lokalisasi keypoint (keypoint localization). Berdasarkan urutan tahapan ini, pendekatan pose estimation dapat diklasifikasikan menjadi top-down dan bottom-up (Ruijie Li, 2023). Pendekatan top-down pertama kali mendekripsi individu dalam citra, kemudian melakukan estimasi pose untuk setiap orang yang terdeteksi secara individual. Metode ini cenderung lebih akurat, terutama dalam skenario dengan orang yang tidak terlalu berdekatan, namun kompleksitas komputasinya meningkat seiring jumlah orang. Sebaliknya, pendekatan bottom-up pertama-tama mendekripsi semua keypoint tubuh yang mungkin ada dalam citra, tanpa mengidentifikasi individu terlebih dahulu, kemudian mengelompokkan keypoint yang

terdeteksi untuk membentuk pose individu. Menurut Jiangyao Wang (2022) pendekatan bottom-up lebih efisien secara komputasi dan lebih cocok untuk skenario dengan banyak orang atau kerumunan, meskipun terkadang kurang akurat dibandingkan top-down dalam kondisi tertentu.

MediaPipe Pose, *framework* yang akan digunakan dalam skripsi ini, mengimplementasikan solusi real-time pose estimation berbasis deep learning. Framework ini mampu mendeteksi dan melacak 33 landmark 2D tubuh manusia dengan kecepatan dan akurasi yang tinggi, bahkan pada perangkat mobile (Meet Shah et al., 2023). Landmark ini mencakup titik-titik kunci pada wajah, tangan, dan seluruh tubuh, membentuk representasi kerangka tubuh yang rinci. Kemampuan tracking kerangka tubuh secara real-time menjadikan MediaPipe Pose alat yang sangat efektif untuk aplikasi yang membutuhkan analisis gerakan tubuh secara dinamis, seperti deteksi kesalahan gerakan latihan, analisis performa olahraga, dan interaksi berbasis gestur.

2.3 Mediapipe

MediaPipe adalah sebuah *framework* sumber terbuka yang dikembangkan oleh Google, dirancang khusus untuk membangun *pipeline* pemrosesan data multimedia secara *end-to-end* (Google, 2023). *Framework* ini menekankan pada kecepatan dan efisiensi, sehingga sangat ideal untuk aplikasi yang membutuhkan pemrosesan data *real-time* pada berbagai platform, mulai dari perangkat *mobile*, *web*, hingga *desktop*. *MediaPipe* menawarkan solusi komprehensif untuk berbagai tugas *machine learning* dan pemrosesan media, termasuk di antaranya deteksi objek, segmentasi citra, pengenalan gestur, estimasi pose (*pose estimation*) manusia. Keunggulan utama *MediaPipe* terletak pada modularitasnya, di mana *pipeline* dibangun dari komponen-komponen modular yang disebut *calculators*. *Calculators* ini merupakan unit pemrosesan independen yang dapat dikombinasikan dan dikonfigurasi untuk membentuk alur kerja yang kompleks namun terstruktur dengan baik.

MediaPipe dikenal sebagai *framework* yang bersifat multi-modal, yang berarti mampu diterapkan secara fleksibel pada berbagai jenis data multimedia, baik berupa foto tunggal maupun rangkaian video. Pengembangan *MediaPipe* oleh Google didasarkan pada TensorFlow Lite, sebuah pustaka *machine learning* ringan yang memungkinkan kemudahan adaptasi dan modifikasi data melalui penggunaan grafik komputasi. *Framework* ini menyediakan beragam model deteksi dan pelacakan tubuh manusia yang andal, yang dilatih menggunakan dataset Google yang sangat besar dan representatif.

Secara fundamental, arsitektur MediaPipe dibangun atas tiga pilar utama: evaluasi kinerja (*performance evaluation*) yang terukur, *framework* terintegrasi untuk mengakuisisi data sensor dari berbagai sumber, serta kumpulan komponen yang dapat digunakan kembali (*reusable components*) yang disebut kalkulator. Konfigurasi sistem dalam MediaPipe dilakukan melalui *pipeline configuration*, yang mendefinisikan grafik komputasi (*computation graph*) yang menghubungkan kalkulator-kalkulator yang berbeda melalui aliran data (*data streams*). Fleksibilitas arsitektur ini memungkinkan pengembang untuk memodifikasi kalkulator yang sudah ada atau bahkan menciptakan kalkulator baru yang sepenuhnya disesuaikan dengan kebutuhan spesifik aplikasi yang mereka bangun, sehingga memberikan tingkat kustomisasi yang tinggi dalam pemrosesan data multimedia.

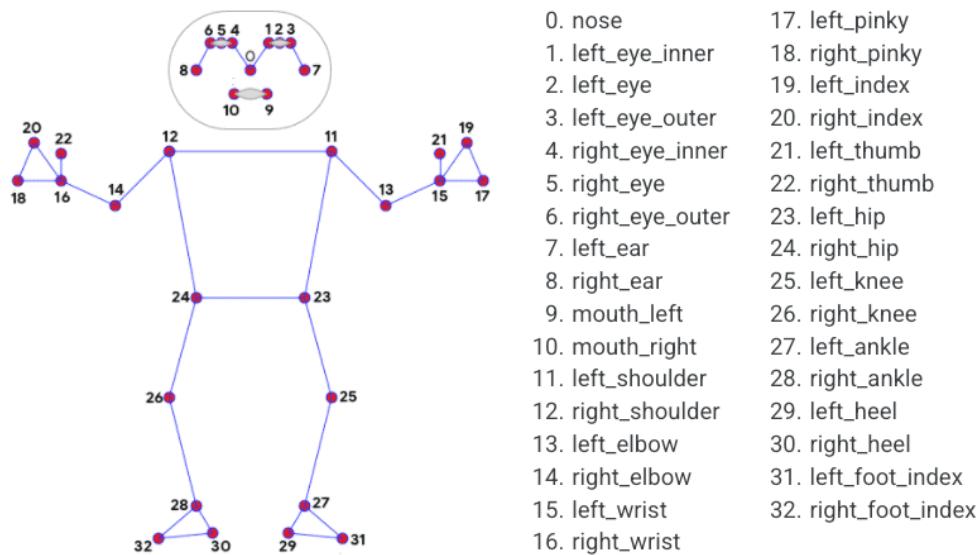
2.3.1 BlazePose Landmark

BlazePose Landmarks merupakan salah satu model *machine learning* (ML) yang disediakan oleh MediaPipe, dirancang secara khusus untuk melakukan pelacakan pose tubuh (*body pose tracking*) dengan tingkat ketelitian tinggi (*high-fidelity*). Model ini menghasilkan output berupa 33 *landmark* 3D yang mengidentifikasi titik-titik kunci pada tubuh manusia, serta masker segmentasi latar belakang (*background segmentation mask*) dari tubuh yang terdeteksi dalam *frame* video RGB. Arsitektur BlazePose dibangun sebagai *superset* dari topologi-topologi model computer vision yang telah teruji kinerjanya, yaitu COCO (Common Objects in Context), BlazeFace, dan BlazePalm. Integrasi topologi ini memungkinkan BlazePose untuk mencapai performa yang komprehensif dalam estimasi pose tubuh, menggabungkan kemampuan deteksi objek umum (COCO), deteksi wajah yang efisien (BlazeFace), dan pelacakan tangan yang responsif (BlazePalm) dalam satu model terpadu.

Untuk mencapai kinerja *real-time*, BlazePose mengimplementasikan *pipeline* deteksi-pelacakan dua langkah (*two-step detection-tracking pipeline*), sebuah pendekatan yang juga diterapkan pada model MediaPipe Hands dan MediaPipe Face Mesh. *Pipeline* ini dimulai dengan tahap deteksi, di mana model menggunakan detektor untuk menemukan *region-of-interest* (ROI) yang berpotensi mengandung pose tubuh manusia dalam *frame*. Selanjutnya, pada tahap pelacakan (*tracking*), model menggunakan *tracker* yang menerima *frame* terpotong ROI (*ROI-cropped frame*) sebagai input untuk memprediksi 33 *landmark* pose tubuh dan masker segmentasi di dalam area ROI tersebut. Pendekatan ini memungkinkan BlazePose untuk

memfokuskan sumber daya komputasi pada area yang relevan, sehingga meningkatkan efisiensi dan kecepatan pemrosesan tanpa mengorbankan akurasi estimasi pose.

Pada gambar 2.1 mengilustrasikan visualisasi 33 *landmark* yang dihasilkan oleh model BlazePose Landmarks. Setiap *landmark* direpresentasikan sebagai titik berwarna pada tubuh manusia, dengan penomoran ID (0 hingga 32) dan nama deskriptif yang sesuai dengan bagian tubuh yang diwakilinya. Sebagai contoh, *landmark* ID 0 diberi label "nose" (hidung), *landmark* ID 1 adalah "left_eye_inner" (sudut dalam mata kiri),



Gambar 2. 1 BlazePose Landmarks

landmark ID 11 merepresentasikan "left_shoulder" (bahu kiri), dan *landmark* ID 25 adalah "left_knee" (lutut kiri). Garis-garis yang menghubungkan antar *landmark* pada gambar menunjukkan koneksi kerangka tubuh (*skeleton*) yang direkonstruksi oleh model. Dengan memanfaatkan 33 titik *landmark* yang tersebar di seluruh tubuh, BlazePose Landmarks menyediakan representasi pose tubuh yang kaya dan informatif, yang dapat dimanfaatkan untuk berbagai aplikasi analisis gerakan manusia, termasuk deteksi kesalahan gerakan dalam konteks latihan fisik yang menjadi fokus utama penelitian ini.

2.4 OpenCV

OpenCV (Open Source Computer Vision Library) adalah sebuah *library* open source yang sangat populer dan luas digunakan dalam bidang *computer vision*, pengolahan citra digital, dan *machine learning*. Dikembangkan sejak tahun 1999 oleh Intel dan kini dikelola

oleh komunitas pengembang global, *OpenCV* menyediakan lebih dari 2500 algoritma yang dioptimalkan, mencakup berbagai aspek pengolahan citra dan *computer vision*. *Library* ini ditulis dalam bahasa C++ namun menyediakan *interface* yang nyaman untuk bahasa pemrograman lain seperti Python, Java, dan MATLAB, menjadikannya sangat fleksibel dan mudah diintegrasikan ke dalam berbagai proyek pengembangan perangkat lunak. *OpenCV* dirancang untuk efisiensi komputasi dan kinerja *real-time*, sehingga sangat cocok untuk aplikasi yang membutuhkan pemrosesan citra dan video secara cepat, seperti analisis video, deteksi objek, pengenalan wajah, *camera calibration*, *feature extraction*, dan pelacakan objek.

OpenCV menyediakan fungsionalitas yang kaya untuk berbagai tahapan pemrosesan citra. Pada tahap pra-pemrosesan, *OpenCV* menawarkan berbagai fungsi untuk *image filtering* (seperti *blurring*, *sharpening*, *noise reduction*), transformasi warna, operasi morfologi, dan segmentasi citra. Untuk ekstraksi fitur, *OpenCV* menyediakan implementasi algoritma klasik seperti *Canny edge detection*, *Harris corner detection*, *SIFT (Scale-Invariant Feature Transform)*, dan *SURF (Speeded Up Robust Features)*, serta deskriptor fitur modern seperti *ORB (Oriented FAST and Rotated BRIEF)*. Dalam hal deteksi objek dan pengenalan pola, *OpenCV* menyediakan algoritma *machine learning* klasik seperti *Haar cascades* untuk deteksi objek cepat, serta dukungan untuk model *deep learning* melalui integrasi dengan *framework* seperti *TensorFlow* dan *PyTorch*. Untuk proyek skripsi ini, *OpenCV* akan dimanfaatkan terutama untuk tugas-tugas pengolahan *frame* video yang diperoleh dari *MediaPipe*, manipulasi citra, perhitungan sudut antar *landmark* tubuh (*angle calculation*), dan implementasi logika deteksi kesalahan gerakan berdasarkan analisis geometri dan temporal dari data *pose estimation*.

2.5 Streamlit

Streamlit adalah sebuah *framework* Python sumber terbuka yang dirancang untuk memudahkan data scientist dan machine learning engineer dalam membangun aplikasi *web* interaktif dari script Python. Streamlit menekankan pada kesederhanaan dan kecepatan pengembangan, memungkinkan pengguna untuk membuat aplikasi web yang powerful dan *user-friendly* hanya dengan menulis beberapa baris kode Python, tanpa memerlukan pengetahuan mendalam tentang pengembangan tampilan antarmuka. Keunggulan utama Streamlit adalah *live-reloading*, di mana setiap perubahan pada kode Python akan secara otomatis diperbarui pada aplikasi web yang sedang berjalan, mempercepat proses iterasi dan *prototyping*. Streamlit juga menyediakan berbagai elemen *user interface (widgets)* siap pakai,

seperti *buttons*, *sliders*, *text inputs*, *charts*, dan *maps*, yang dapat dengan mudah ditambahkan ke aplikasi untuk menciptakan *interface* yang interaktif dan responsif.

Dalam konteks skripsi ini, Streamlit dipilih sebagai *framework* untuk membangun aplikasi web yang akan mendemonstrasikan sistem deteksi kesalahan gerakan latihan Seated Cable Row. Streamlit sangat ideal untuk tujuan ini karena kemampuannya dalam menyajikan hasil pemrosesan computer vision secara visual dan interaktif kepada pengguna akhir. Aplikasi Streamlit akan digunakan untuk menampilkan feed video latihan *real-time* yang dianalisis oleh sistem, memvisualisasikan *pose estimation* dengan *overlay* kerangka tubuh pada video, dan memberikan feedback kepada pengguna mengenai potensi kesalahan gerakan yang terdeteksi. Kemudahan penggunaan *Streamlit* dan integrasinya yang mulus dengan library Python lainnya, seperti OpenCV dan MediaPipe, menjadikannya platform yang efektif untuk memvalidasi dan mendemonstrasikan hasil penelitian skripsi dalam format aplikasi web yang mudah diakses dan dipahami.

2.6 Aktivitas Latihan di Gym

2.6.1 Pengertian Latihan Beban

Latihan beban (*resistance training*) adalah bentuk latihan fisik yang melibatkan penggunaan resistensi eksternal, seperti *barbell*, *dumbbell*, mesin gym, atau berat badan sendiri untuk menstimulasi kontraksi otot. Tujuan utamanya adalah meningkatkan kekuatan otot, daya tahan, hipertrofi (*peningkatan massa otot*), serta adaptasi neurologis yang mendukung performa fisik. Latihan ini bekerja dengan prinsip *overload*, di mana otot diberi beban melebihi kapasitas normal untuk memicu adaptasi fisiologis, seperti peningkatan sintesis protein dan efisiensi sistem saraf.

Menurut penelitian oleh Westcott (2012), latihan beban secara teratur tidak hanya meningkatkan komposisi tubuh dengan mengurangi persentase lemak dan meningkatkan massa otot, tetapi juga berperan dalam meningkatkan kepadatan tulang, metabolisme basal, serta mengurangi risiko penyakit kronis seperti diabetes tipe 2. Selain itu, ACSM (2009) menekankan bahwa latihan ini harus dilakukan dengan prinsip *progressiveness*, *variation*, dan *specificity* untuk memaksimalkan hasil dan meminimalkan risiko cedera.

2.6.2 Squat

Squat merupakan salah satu bentuk latihan dasar yang melibatkan aktivitas beberapa sendi secara bersamaan (latihan compound multi-joint) dan berfungsi untuk meningkatkan kekuatan serta hipertrofi otot pada tubuh bagian bawah. Gerakan ini secara dominan mengaktifkan otot quadriceps femoris (paha depan), gluteus maximus (bokong), dan hamstring (paha belakang), di samping peranan penting dari otot inti (core muscles) sebagai penstabil gerakan. Secara umum, rangkaian gerakan squat terdiri atas fase eksentrik, yaitu menurunkan pinggul, dan fase konsentrik, yaitu mengangkat pinggul kembali ke posisi semula, dengan posisi akhir paha idealnya sejajar dengan lantai (parallel) atau lebih rendah.

Pada penggunaan V-Squat Machine, prinsip dasar squat tetap dipertahankan, namun pelaksanaannya dibantu oleh mesin yang memandu jalur gerak pengguna. V-Squat Machine didesain untuk mereplikasi pola gerakan squat, namun pada lintasan yang tetap (fixed path) atau semi-tetap yang biasanya berbentuk busur (arc). Hal ini membedakan V-Squat Machine dari squat konvensional, seperti barbell back squat, yang mengharuskan atlet mengendalikan stabilitas barbel dan tubuh di berbagai bidang gerak. Dengan adanya jalur yang telah ditentukan oleh mesin, kebutuhan stabilisasi dari otot-otot sinergis dan inti menjadi lebih rendah, sehingga pengguna dapat lebih terfokus pada aktivasi otot target utama.

Menurut (Escamilla et al., 2002), squat dikategorikan sebagai latihan fungsional karena meniru pola gerak sehari-hari seperti duduk, berdiri, dan mengangkat beban. Manfaat squat tidak hanya terbatas pada peningkatan kekuatan dan massa otot, tetapi juga berkontribusi pada peningkatan koordinasi neuromuskular, keseimbangan, serta fleksibilitas atau rentang gerak sendi panggul, lutut, dan pergelangan kaki. Studi yang dilakukan oleh (Schoenfeld et al., 2016) juga menguatkan bahwa berbagai variasi squat sangat efektif dalam menstimulasi otot quadriceps dan gluteus maximus. Pada latihan menggunakan V-Squat, walaupun pola aktivasi otot serupa, penggunaan mesin memungkinkan latihan dengan beban lebih berat sambil mengurangi risiko kesalahan teknik terkait keseimbangan.

Oleh sebab itu, squat pada V-Squat Machine dapat didefinisikan sebagai bentuk latihan squat berbantuan alat, yang dirancang untuk mengoptimalkan latihan otot utama tubuh bagian bawah, khususnya quadriceps dan gluteus, dalam jalur gerakan yang terkontrol, sehingga memberikan stabilitas lebih dan memungkinkan fokus maksimal pada otot target.

2.6.3 Teknik dan Tahapan Gerakan Squat pada V-Squat Machine yang Benar

Gerakan V-Squat yang benar dapat dibagi menjadi tiga tahapan utama: posisi awal (persiapan), fase menurun (eksentrik), dan fase mendorong (konsentris), berikut penjelasan dari masing-masing tahapan gerakan tersebut:

Posisi Awal:

Tahap ini adalah fondasi untuk melakukan gerakan yang aman dan efektif. Penempatan posisi yang tepat akan memastikan otot yang ditargetkan bekerja secara optimal dan mencegah terjadinya *miss* gerakan yang dapat menyebabkan cedera.

- Tempatkan kaki di plat selebar bahu, ujung kaki boleh mengarah ke luar atau lurus ke depan sesuai kenyamanan, dan jaga punggung tetap lurus dan netral, dada sedikit terangkat.
- Pegang handle pada sisi mesin agar tubuh tetap seimbang.
- Dengan posisi tubuh yang sudah siap, luruskan sedikit lutut untuk mendorong beban ke atas dan melepaskan kunci pengaman mesin. Ini adalah posisi awal untuk memulai repetisi dengan aman.

Fase Menurun (Eksentrik):

Fase eksentrik merupakan komponen penting dalam latihan squat yang berkontribusi signifikan terhadap adaptasi kekuatan dan hipertrofi otot. Penelitian yang dilakukan oleh (Nicolas Da Silva Pereira et al., 2024) menunjukkan bahwa kontrol yang baik selama fase eksentrik dapat meningkatkan aktivasi otot dan mencegah cedera. Berikut tahapan dari fase eksentrik:

- Tarik napas, tekuk lutut dan pinggul bersamaan untuk menurunkan tubuh. Gerakan mirip saat duduk di kursi.

- Turunkan tubuh secara perlahan dan terkontrol dengan kecepatan yang konsisten. Penelitian menunjukkan bahwa kontrol eksentrik yang baik dapat meningkatkan aktivasi otot hingga 20-30% dibandingkan dengan gerakan yang terlalu cepat (Cong Xie et al., 2024). Jaga agar punggung tetap lurus dan inti tubuh (core) tetap kencang untuk melindungi tulang belakang.
- Lanjutkan gerakan menurun hingga paha berada dalam posisi sejajar (paralel) dengan *footplate*, atau sedikit lebih rendah jika fleksibilitas memungkinkan. Menurut penelitian (Kathy E. O'Neill et al., 2021) mengatakan bahwa kedalaman squat yang optimal (90-125 derajat fleksi lutut) dapat menghasilkan aktivasi otot quadriceps dan gluteus yang maksimal.

Fase Mendorong (Konsentrifis):

Fase konsentrifis merupakan fase akhir di mana tubuh didorong kembali ke posisi awal. Pada fase ini, otot bekerja secara maksimal untuk menghasilkan tenaga dan kecepatan gerakan. Penerapan teknik yang benar sangat penting agar kinerja latihan menjadi optimal sekaligus mencegah risiko cedera. Berikut tahapan dari fase konsentrifis:

- Hembuskan napas saat mendorong tubuh ke atas dengan menekan tumit dan seluruh telapak kaki ke *footplate*. Fokuskan tenaga pada paha depan dan *glutes*.
- Pastikan punggung tetap lurus dan tidak melengkung saat mendorong beban. Studi yang dilakukan oleh (Neil Welch et al., 2015) mengatakan bahwa mempertahankan postur netral punggung selama fase konsentrifis dapat mengurangi stress pada diskus intervertebralis hingga 40%. Hindari menggunakan momentum atau menghentak beban yang dapat meningkatkan risiko cedera.
- Kemudian luruskan kaki hampir sepenuhnya tanpa mengunci lutut, pertahankan sedikit fleksi untuk menjaga ketegangan otot dan melindungi sendi. Ulangi gerakan sesuai kebutuhan.

2.6.4 Rujukan Ilmiah sebagai Landasan Penetapan Threshold Feedback Squat

Pemahaman biomekanika pada squat sangat diperlukan untuk menekan risiko cedera sekaligus meningkatkan efektivitas latihan. (Weng et al., 2024) mengungkapkan bahwa pemahaman tentang sudut sendi pada saat squat dapat membantu menemukan pola gerakan yang berbahaya dan memfasilitasi intervensi korektif yang tepat. (Lisna Wijayanti et al., 2024) juga menegaskan bahwa teknik squat yang baik akan mengurangi tekanan pada sendi utama, terutama pada lutut, pinggul, serta tulang belakang.

Setiap kesalahan posisi saat squat berdampak langsung pada aspek biomekanik. Tubuh yang terlalu miring ke belakang ($\text{hip_vertical_angle} > 50^\circ$), seperti yang dijelaskan (Bayattork et al., 2024), dapat memicu ketidakstabilan postural dan penurunan kontrol neuromuskular. Sebaliknya, posisi badan yang terlalu maju ke depan ($\text{hip_vertical_angle} < 10^\circ$) meningkatkan risiko cedera tulang belakang karena beban kompresi dan geser yang tinggi pada area lumbal, sebagaimana dipaparkan (Yavuz & Erdag, 2017). Jika lutut melewati jari kaki secara berlebihan dapat menyebabkan tekanan pada sendi lutut meningkat dan berpotensi menimbulkan cedera ligamen, sebagaimana diungkapkan (Wallace et al., 2002). Sedangkan squat yang terlalu dalam ($\text{knee_vertical_angle} > 125^\circ$) memang meningkatkan beban pada sendi lutut, namun masih dianggap aman untuk orang sehat tanpa riwayat cedera menurut (Cooke, 2025) dan (Rojas-Jaramillo et al., 2024).

Kemudian hasil penelitian lain yang dipaparkan oleh (Kathy E. O'Neill et al., 2021) mengatakan bahwa squat dengan kedalaman ($\text{knee_vertical_angle} 90^\circ$) dapat memberikan aktivasi otot yang lebih tinggi dibanding squat yang lebih dangkal (125°). Pada penelitian tersebut dijelaskan *vastus lateralis* (otot paha depan) dapat meningkat drastis di kedalaman squat 90° , dengan rata-rata hingga 1.4 kali lebih tinggi dan puncaknya mencapai 1.87 kali dibandingkan kedalaman squat 125° . Hal ini masuk akal karena *vastus lateralis* bekerja paling intens di posisi fleksi lutut maksimum yakni saat lutut paling menekuk.

Namun, mengingat aspek keamanan sendi lutut dan mempertimbangkan rekomendasi dari penelitian terbaru yang menyatakan squat dalam tetap aman bagi

individu sehat tanpa riwayat cedera, dalam penelitian ini saya memilih mengambil batas tengah kedalaman squat, yaitu pada sudut sekitar 105° fleksi lutut. Keputusan ini diambil agar tetap memperoleh manfaat aktivasi otot yang signifikan, namun pada saat yang sama tetap menjaga keamanan dan kenyamanan peserta uji, sehingga hasil yang diperoleh dapat diaplikasikan secara lebih luas dan relevan bagi populasi umum.

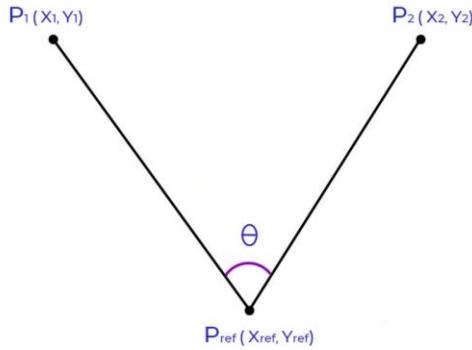
Berdasarkan hasil-hasil tersebut, penggunaan ambang batas sudut tertentu sangat dianjurkan sebagai dasar sistem feedback otomatis pada analisis squat berbasis pose estimation. Penerapan threshold ini terbukti membantu mendeteksi serta mengoreksi pola gerak yang berisiko secara real-time, sehingga manfaat latihan dapat ditingkatkan sekaligus menurunkan kemungkinan terjadinya cedera. Berikut rincian hasil dari tabel penetapan thresholds untuk feedback gerakan squat:

No.	Feedback Kesalahan	Sudut Relevan	Threshold Angle ($^\circ$)	Sumber Literatur Ilmiah	Kesimpulan Ilmiah
1.	Badan Terlalu ke Belakang	hip_vertical_angle	> 50	(Bayattork et al., 2024)	Risiko ketidakstabilan postural, gangguan keseimbangan, dan kompensasi postur
2.	Badan Terlalu ke Depan	hip_vertical_angle	< 10	(Yavuz & Erdag, 2017)	Risiko excessive forward lean, tekanan pada punggung

					bawah, teknik tidak optimal
3.	Lutut Melewati Jari Kaki	ankle_vertical_angle	> 45	(Wallace et al., 2002)	Risiko tekanan berlebih pada lutut, kompensasi ankle dorsiflexion terbatas
4.	Squat Terlalu Dalam	knee_vertical_angle	> 105	(Cooke, 2025; Rojas-Jaramillo et al., 2024), (Kathy E. O'Neill et al., 2021)	Umumnya aman untuk populasi sehat, kecuali ada riwayat cedera/kelainan khusus

2.7 Angle Calculation

Angle Calculation dalam sistem koordinat dua dimensi dapat dilakukan menggunakan operasi dot product (perkalian titik vektor) dan fungsi arccosine. Dot product merupakan operasi aljabar vektor yang menghitung persamaan arah dua vektor, sementara arccosine berfungsi sebagai invers dari fungsi cosinus untuk mengonversi nilai cosinus ke sudut sebenarnya. Dalam penelitian ini digunakan untuk menghitung sudut *offset* antara titik hitung (*nose*) dan bahu (*shoulders*), dengan hidung sebagai titik referensi P_{ref} (X_{ref}, Y_{ref}), serta bahu kiri P_1 (X_1, Y_1) dan bahu kanan P_2 (X_2, Y_2).



Dot product dari dua vektor $a = \begin{pmatrix} a_x \\ a_y \end{pmatrix}$ dan $b = \begin{pmatrix} b_x \\ b_y \end{pmatrix}$ didefinisikan sebagai:

$$a \cdot b = a_x \cdot b_x + a_y \cdot b_y$$

Dimana a_x dan a_y merupakan komponen vektor a, sedangkan b_x dan b_y merupakan komponen vektor b. Nilai ini merepresentasikan geometri satu vektor terhadap vektor lainnya. Kemudian hubungan antara dot product dan sudut θ antara kedua vektor dinyatakan melalui persamaan berikut:

$$a \cdot b = |a| \cdot |b| \cdot \cos \theta$$

Dengan $|a|$ dan $|b|$ adalah panjang vektor, dihitung menggunakan rumus Euclidean Distance:

$$|a| = \sqrt{a_x^2 + a_y^2}, \quad |b| = \sqrt{b_x^2 + b_y^2}$$

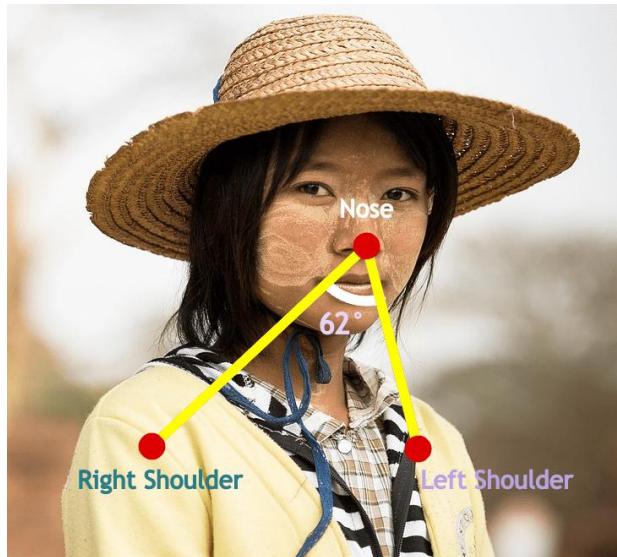
Kemudian untuk memperoleh sudut θ , nilai cosinus di invers menggunakan fungsi arccosinus:

$$\theta = \arccos\left(\frac{a \cdot b}{|a| \cdot |b|}\right)$$

Keterangan:

θ = Sudut antara vektor a dan b (rentang 0° hingga 180°)

\arccos = Fungsi invers cosinus yang mengonversi rasio ke sudut



Gambar 2. 2 BlazePose Landmarks on Face and Shoulders with Angle Calculation

Sudut offset antara hidung (*nose*) dan bahu (*shoulders*) dihitung dengan mengambil titik hidung sebagai referensi $P_{ref}(X_{ref}, Y_{ref})$. Vektor dibentuk dari P_{ref} ke bahu kiri $P_1(X_1, Y_1)$ dan bahu kanan $P_2(X_2, Y_2)$:

$$\theta = \arccos \left(\frac{\overrightarrow{P1_{ref}} \cdot \overrightarrow{P2_{ref}}}{|P1_{ref}| \cdot |P2_{ref}|} \right)$$

Jika sudut θ yang dihasilkan melebihi batas ambang (*Offset_Thresh*), sistem mengasumsikan individu menghadap ke depan kamera, dan pesan peringatan akan ditampilkan.

2.8 Mean Per Joint Position Error (MPJPE)

Mean Per Joint Position Error (MPJPE) adalah salah satu metrik yang sering digunakan untuk menilai tingkat akurasi sistem estimasi pose manusia, baik pada bidang 2D maupun 3D. Metrik ini menghitung rata-rata selisih posisi pada setiap sendi antara hasil prediksi model dengan data ground truth (Yuchen Shi et al., 2024). Pada estimasi pose 2D, MPJPE mengukur seberapa besar perbedaan koordinat piksel tiap titik sendi hasil prediksi dibandingkan dengan posisi aslinya.

Menurut (Asuka Ishii & Hiroo Ikeda, 2024), MPJPE memberikan nilai evaluasi yang objektif sehingga dapat digunakan untuk melakukan perbandingan performa berbagai

algoritma estimasi pose secara kuantitatif. Penggunaan MPJPE sangat penting dalam penelitian estimasi pose karena metrik ini memberikan gambaran yang jelas terkait tingkat kesalahan posisi setiap sendi yang dihasilkan oleh model. Dengan adanya nilai MPJPE, peneliti dapat mengetahui seberapa akurat model dalam memprediksi posisi tubuh manusia. Selain itu, MPJPE juga memudahkan proses evaluasi dan perbandingan antara berbagai metode atau algoritma yang dikembangkan, sehingga dapat membantu dalam menentukan pendekatan mana yang paling efektif untuk studi kasus tertentu.

Secara matematis, MPJPE diperoleh dengan menghitung rata-rata selisih posisi setiap sendi pada satuan piksel (untuk 2D) atau milimeter (untuk 3D), menggunakan *Euclidean Distance* sebagai acuannya. Untuk estimasi pose 2D, perhitungan MPJPE dapat dirumuskan sebagai berikut:

$$\text{MPJPE} = \frac{1}{N \cdot J} \sum_{i=1}^N \sum_{j=1}^J \|\hat{p}_{ij} - p_{ij}\|$$

- N: jumlah sample frame yang di evaluasi
- J: jumlah sendi (landmark tubuh) yang dideteksi
- \hat{p}_{ij} : koordinat prediksi sendi ke-*j* pada frame ke-*i*
- p_{ij} : posisi ground truth sendi ke-*j* pada frame ke-*i*

Resolusi video yang digunakan dalam penelitian ini memiliki resolusi 640x480 pixel , sehingga perlu dilakukan penyesuaian skala dari nilai referensi yang berasal dari literatur ilmiah dengan resolusi berbeda. Menurut penelitian yang dilakukan (Calabrese et al., 2019) mencatat nilai MPJPE sekitar ~7 pixel yang dapat dijadikan acuan untuk performa excellent pada kategori 2D *Pose Estimation*. Sementara itu, berdasarkan penelitian yang dilakukan (Lino et al., 2025) dataset BlendMimic3D dengan resolusi 1000x1002 pixel memberikan nilai MPJPE yang bervariasi, yaitu sekitar 56–120 pixel , tergantung pada kondisi visibilitas sendi (*visible, occluded, all-occluded*).

Dalam proses evaluasi sistem deteksi pose tubuh pada dataset beresolusi 640x480 piksel, kategori MPJPE (Mean Per Joint Position Error) telah ditetapkan secara khusus untuk memberikan pemahaman yang lebih terperinci mengenai kinerja sistem dalam mengidentifikasi posisi landmark tubuh. Berdasarkan hasil analisis literatur serta penyesuaian

skala yang disesuaikan dengan resolusi data, klasifikasi evaluasi MPJPE dapat dirumuskan sebagai berikut:

- Excellent Performance (≤ 10 pixel) : Nilai MPJPE di bawah atau sama dengan 10 piksel menunjukkan performa yang sangat baik, dengan kesalahan minimal dalam deteksi posisi landmark tubuh.
- Good Performance (11–25 pixel): Nilai MPJPE yang berada pada rentang 11 hingga 25 piksel dikategorikan sebagai performa yang baik, meskipun masih terdapat sedikit kesalahan prediksi
- Poor Performance (> 25 pixel) : Jika nilai MPJPE melebihi 25 piksel , sistem dianggap memiliki performa yang kurang memadai. Kategori ini mencakup kasus-kasus di mana deteksi landmark tubuh tidak cukup akurat, sehingga umpan balik yang diberikan kepada pengguna mungkin tidak tepat.

Untuk penelitian ini, metrik MPJPE diimplementasikan untuk menilai sejauh mana sistem mampu mengenali dan melacak posisi sendi tubuh secara akurat selama aktivitas gerakan V-Squat. Nilai MPJPE yang diperoleh kemudian dianalisis lebih lanjut melalui visualisasi grafik dan statistik deskriptif guna memberikan gambaran objektif mengenai performa sistem dalam berbagai kondisi uji.

2.9 Studi Literatur Sejenis

Berikut merupakan tabel literatur review yang berisi 10 penelitian sebelumnya berupa jurnal yang membahas topik terkait computer vision dan penerapan human pose.

Tabel 2.1 Studi Literatur

No.	Peneliti (Tahun)	Judul	Metode	Kelebihan	Kekurangan
1.	(Saleem et al., 2023)	TrainERAI - Live Gym Tracker using Artificial Intelligence	MediaPipe, OpenCV, CVZone, rep counter dengan	Deteksi postur real-time, penghitung kalori, integrasi	Tidak mendukung variasi latihan kompleks, fokus hanya pada latihan

			threshold sudut sendi	permainan interaktif, dan sistem insentif keuangan	dasar (pull, push, leg)
2.	(Dedhia et al., 2023)	Pose Estimation and Virtual Gym Assistant using MediaPipe and Machine Learning	MediaPipe BlazePose, algoritma machine learning (SVM, ANN)	Klasifikasi postur akurat (99% dengan Logistic Regression), analisis sudut sendi dinamis, dan feedback spesifik untuk latihan compound	Dataset terbatas, belum diuji untuk latihan dengan beban tinggi
3.	(Chaithra V S et al., 2024)	Posture Assessment Using Pose Detection in Python: A Real-time Approach with MediaPipe and OpenCV	MediaPipe Pose, OpenCV	Fokus pada analisis postur ergonomis, antarmuka pengguna sederhana, dan kompatibilitas multi-perangkat	Hanya fokus pada postur statis (duduk/berdiri), belum mencakup gerakan dinamis
4.	(Ko et al., 2024)	Real-Time AI Posture Correction for Powerlifting Exercises	YOLOv5, MediaPipe	Klasifikasi fase kontraksi (eksentrrik & konsentrrik), feedback	Memerlukan dataset besar, belum diuji untuk latihan dengan variasi

		Using YOLOv5 and MediaPipe		detail, dan integrasi dengan aplikasi web	gerakan kompleks
5.	(Radhakrishnan & Vijayaraghavan, n.d.)	Real-Time 3D Human Pose Estimation Using Deep Learning Model for Ergonomics	YOLOv8, MediaPipe	Deteksi 3D real-time, fokus pada ergonomi pekerja kantor, dan integrasi alert otomatis	Tidak menyediakan feedback korektif spesifik, hanya notifikasi umum
6.	(Meet Shah et al., 2023)	Computer Vision & Deep Learning based Realtime and Pre-Recorded Human Pose Estimationx	BlazePose GHUM 3D	Dapat melakukan estimasi pose manusia secara real-time dan akurat dengan visibilitas mencapai 96.9%	Akurasi dapat dipengaruhi oleh resolusi gambar atau video, jumlah orang dalam scene, dan keberadaan oklusi
7.	(Dudekula et al., 2024)	Physiotherapy Assistance for Patients Using Human Pose Estimation With Raspberry Pi	MediaPipe dan OpenCV	Sistem dapat memberikan panduan fisioterapi yang akurat dan real-time kepada pasien melalui umpan balik visual dan audio	Terbatas pada latihan fisioterapi tertentu dan akurasi dapat dipengaruhi oleh faktor lingkungan seperti kondisi pencahayaan, latar belakang,

					dan sudut kamera
8.	(Chakole et al., 2024)	Real-Time Full-Body Detection Using Computer Vision Leveraging OpenCV and MediaPipe	MediaPipe Holistic	Sistem dapat mendeteksi pose tubuh manusia secara real-time dengan akurasi tinggi, mencapai 95% untuk deteksi wajah dan 92% untuk deteksi pose tubuh	Dapat mengalami kesulitan dalam kondisi cahaya redup atau sudut kamera ekstrim, serta ketika bagian tubuh terhalang (oklusi)
9.	(Isha Chaudhary et al., 2023)	Real-Time Yoga Pose Detection Using OpenCV and MediaPipe	MediaPipe dan OpenCV	Sistem dapat mendeteksi pose yoga secara akurat dan memberikan umpan balik real-time kepada pengguna	Dapat mengalami kesulitan dalam membedakan pose yoga yang serupa dan akurasi dapat dipengaruhi oleh variasi kondisi, pakaian, dan bentuk tubuh
10.	(Bhamidipati et al., 2023)	Robust Intelligent Posture Estimation for an AI Gym Trainer	MediaPipe dan OpenCV	Sistem dapat melakukan estimasi pose pengguna secara akurat dalam	Belum ada penelitian tentang penerapan sistem ini pada

		using Mediapipe and OpenCV		berbagai kondisi pencahayaan dan kuat terhadap oklusi dan latar belakang	latihan selain bicep curls
11.	(Kim et al., 2023)	Human Pose Estimation Using MediaPipe Pose and Optimization Method Based on a Humanoid Model	MediaPipe Pose dan uDEAS	Sistem dapat melakukan estimasi pose 3D dari gambar 2D dengan akurasi yang tinggi	Membutuhkan waktu sekitar 0.33 detik untuk memproses satu frame gambar, yang mungkin menjadi kendala untuk aplikasi real-time tertentu

Tabel 2. 2 Perbandingan Penelitian Terdahulu

No.	Peneliti (Tahun)	Judul	Metode	OpenCV	Mediapipe	Streamlit
1.	(Saleem et al., 2023)	TrainERA - Live Gym Tracker using Artificial Intelligence	BlazePose GHUM 3D	✓	✓	-
2.	(Dedhia et al., 2023)	Pose Estimation and Virtual Gym Assistant using MediaPipe and Machine Learning	MediaPipe dan OpenCV	✓	✓	-

3.	(Chaithra V S et al., 2024)	Posture Assessment Using Pose Detection in Python: A Real-time Approach with MediaPipe and OpenCV	Mediapipe dan OpenCV	✓	✓	✓
4.	(Ko et al., 2024)	Real-Time AI Posture Correction for Powerlifting Exercises Using YOLOv5 and MediaPipe	YOLOv5, Mediapipe	✓	✓	-
5.	(Radhakrishnan & Vijayaraghavan, n.d.)	Real-Time 3D Human Pose Estimation Using Deep Learning Model for Ergonomics	YOLOv8, Mediapipe	✓	✓	-
6.	(Meet Shah et al., 2023)	Computer Vision & Deep Learning based Realtime and Pre-Recorded Human Pose Estimation	BlazePose GHUM 3D	✓	✓	-
7.	(Dudekula et al., 2024)	Physiotherapy Assistance for Patients Using Human Pose Estimation With Raspberry Pi	Mediapipe dan OpenCV	✓	✓	-
8.	(Chakole et al., 2024)	Real-Time Full-Body Detection Using Computer Vision Leveraging OpenCV and MediaPipe	Mediapipe dan OpenCV	✓	✓	-
9.	(Isha Chaudhary et al., 2023)	Real-Time Yoga Pose Detection	Mediapipe dan OpenCV	✓	✓	-

		Using OpenCV and MediaPipe				
10.	(Bhamidipati et al., 2023)	Robust Intelligent Posture Estimation for an AI Gym Trainer using Mediapipe and OpenCV	MediaPipe dan OpenCV	✓	✓	-
11.	(Kim et al., 2023)	Human Pose Estimation Using MediaPipe Pose and Optimization Method Based on a Humanoid Model	MediaPipe Pose dan uDEAS	-	✓	-

BAB III

METODOLOGI PENELITIAN

Pembahasan pada bab 3 ini menjelaskan terkait proses pengumpulan data yang menggunakan metode studi literatur dan pustaka. Setelah tahap studi literatur dan pustaka dilakukan untuk mendapatkan informasi yang diperlukan untuk proses penelitian, selanjutnya penulis menggunakan metode pengembangan sistem yaitu waterfall

3.1. Metode Pengumpulan Data

Dalam penelitian ini, metode pengumpulan data dilakukan melalui studi literatur serta observasi langsung di lokasi penelitian. Studi literatur diterapkan dengan menelusuri, mengkaji, dan menganalisis berbagai sumber referensi yang relevan dengan objek serta metode yang digunakan dalam penelitian ini. Referensi diperoleh melalui dua cara, yaitu secara langsung dari literatur yang tersedia di perpustakaan serta secara daring melalui sumber-sumber terpercaya di internet. Hasil dari studi literatur ini digunakan sebagai dasar dalam penyusunan landasan teori, perumusan metodologi penelitian, serta dalam proses perancangan dan pengembangan sistem yang dikaji.

Selain studi literatur, observasi juga dilakukan secara langsung di Ideal Gym, Matoa, Jakarta Selatan, sebagai lokasi utama penelitian. Observasi ini bertujuan untuk menganalisis teknik dan tahapan gerakan seated cable row berdasarkan ketersediaan fasilitas latihan yang digunakan dalam studi kasus. Dengan mengombinasikan pendekatan studi literatur dan observasi, penelitian ini diharapkan dapat memberikan analisis yang akurat terkait teknik pelaksanaan gerakan serta efektivitasnya dalam mengoreksi gerakan.

3.2 Perangkat Penelitian

Pada proses penelitian ini, penulis menggunakan *tools* sebagai berikut :

Laptop Dell Latitude 5420, dengan spesifikasi:

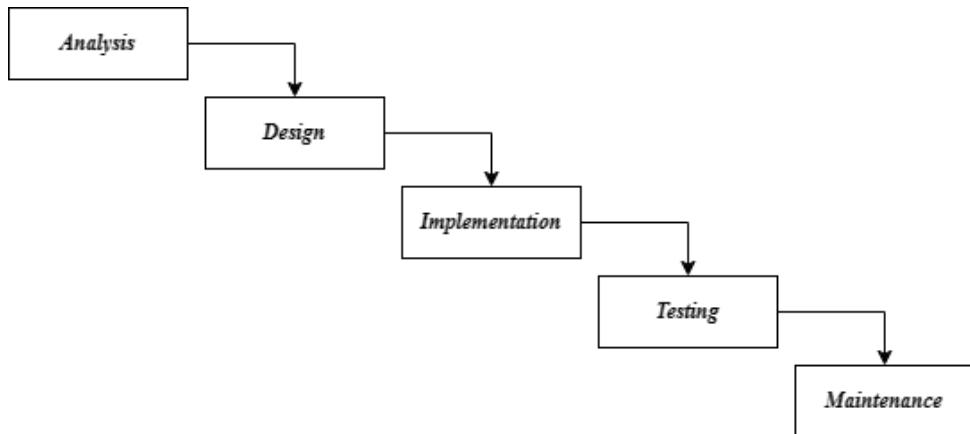
1. Sistem Operasi : Windows 11 Pro
2. Prosesor : Intel Core i7-1165G7 @ 2.8 GHz (8CPU)
3. Kartu Grafis : Intel® Iris Xe Graphic
4. Ram : 16 GB LPDDR4 *dual channel*
5. Kamera : Webcam Camtech CT-30

Software yang digunakan:

1. Visual Studio Code
2. Python versi 3.11
3. Library Mediapipe dan OpenCV

4. Pengembangan User Interface menggunakan Streamlit
5. Digital Ocean Cloud

3.2. Metode Pengembangan Sistem



Gambar 3. 1 Metode Waterfall

Di dalam penelitian ini metode pengembangan sistem yang digunakan adalah metode Waterfall. Metode ini dipilih karena lebih sistematis dan terstruktur, sehingga memudahkan penulis dalam membuat tahapan pengembangan dengan terstruktur. Metode waterfall terdiri dari empat tahap utama, yaitu *Analysis*, *Design*, *Implementation*, dan *Testing*. Berikut penjelasan lengkap terkait tahapan metode tersebut:

3.3.1 Analysis

Pada tahap analisis penulis mengidentifikasi kebutuhan sistem terhadap hal-hal yang dibutuhkan dalam pengembangan sistem deteksi kesalahan gerakan untuk studi kasus *seated cable row* menggunakan *blazepose landmark model* seperti metode dan library yang dibutuhkan untuk sistem, cara *processing frame* menggunakan opencv, serta perhitungan *angle calculation* berdasarkan studi literatur sudah dilakukan di tahap sebelumnya.

3.3.2 Design

Tahap perancangan dalam penelitian ini berfokus pada desain antarmuka pengguna dengan menggunakan framework Streamlit. Tujuan utama dari tahap ini adalah memastikan

bahwa interaksi antara sistem dan pengguna menjadi mudah dipahami dan bahwa penggunaan sistem menjadi lebih terstruktur. Antarmuka yang dirancang mencakup beberapa komponen utama, seperti tampilan video real-time, visualisasi landmark tubuh hasil menggunakan mediapipe, serta koreksi gerakan berbasis teks yang memberikan *feedback* terhadap kesalahan gerakan pengguna.

Selain perancangan antarmuka, tahap ini juga mencakup penyusunan diagram UML, desain arsitektur sistem, serta flowchart. Komponen-komponen ini disusun untuk memvisualisasikan alur kerja sistem secara menyeluruh, sehingga dapat mempermudah pemahaman mengenai mekanisme kerja sistem yang dikembangkan dalam penelitian ini.

3.3.3 Implementation

Tahap implementasi merupakan step yang penting dalam merancang sistem deteksi kesalahan gerakan *squat*. Pada tahap ini, kode program dikembangkan menggunakan bahasa pemrograman Python dengan dukungan berbagai *library* yang relevan. Berikut merupakan proses tahapan dari implementasi:

1. Input Frame dengan Kamera (Handphone)

Proses diawali dengan pengambilan gambar dari video latihan *Seated Cable Row* yang direkam menggunakan kamera *handphone*. Sistem dirancang untuk mengolah video tersebut secara *frame by frame*. Sebelum diproses lebih lanjut, setiap *frame* dikonversi dari format BGR menjadi RGB agar dapat dianalisis menggunakan mediapipe.

2. Landmark Detection

Setelah gambar siap untuk dianalisis, sistem memanfaatkan MediaPipe untuk mendeteksi dan mengidentifikasi keypoint tubuh dalam setiap *frame*. Model BlazePose Landmarks dari MediaPipe digunakan untuk mendeteksi delapan titik tubuh yang diperlukan dalam analisis gerakan *Seated Cable Row*. Selain itu, sistem juga mencatat koordinat posisi masing-masing titik untuk keperluan evaluasi gerakan.

3. Perhitungan Sudut Gerakan

Berdasarkan hasil deteksi *landmark*, sistem menghitung sudut-sudut pada persendian yang berperan dalam menilai akurasi gerakan *Seated Cable Row*. Perhitungan ini bertujuan untuk mengevaluasi apakah posisi tubuh saat latihan telah sesuai dengan teknik yang benar. Proses ini dilakukan dengan menerapkan

perhitungan matematika *angle calculation* guna memperoleh nilai sudut antar sendi tubuh.

4. Analisis Gerakan dan Penentuan Feedback

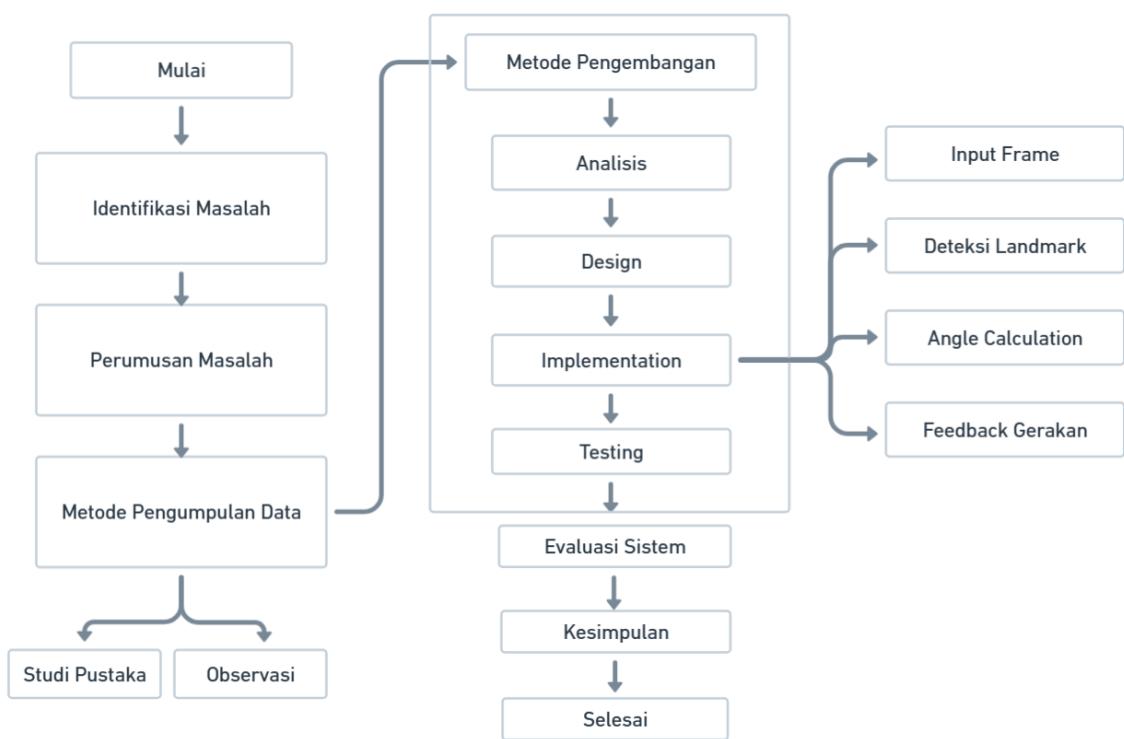
Tahap akhir dari implementasi adalah analisis hasil perhitungan sudut untuk menentukan apakah terdapat kesalahan dalam gerakan yang dilakukan pengguna. Jika ditemukan ketidaksesuaian, sistem akan memberikan *feedback* berupa koreksi atau saran perbaikan yang ditampilkan kepada pengguna berupa *pop up text* di dalam video tersebut. *Feedback* ini dirancang agar pengguna dapat memperbaiki teknik latihannya secara mandiri dan lebih efektif.

3.3.4 Testing

Pada tahap ini, dilakukan pengujian terhadap sistem deteksi kesalahan gerakan *Seated Cable Row* untuk memastikan seluruh fungsi berjalan sesuai dengan spesifikasi yang telah dirancang. Pengujian dilakukan menggunakan metode *Blackbox Testing*, yang berfokus pada validasi keluaran sistem berdasarkan input yang diberikan. Pengujian ini bertujuan untuk mengevaluasi akurasi *feedback* sistem serta ketepatan perhitungan *Row_Count* dalam mendeteksi gerakan yang benar dan salah. Sistem diuji dengan berbagai skenario, termasuk variasi posisi tubuh, sudut gerakan, dan kecepatan eksekusi latihan. Hasil dari pengujian ini akan digunakan untuk memastikan bahwa sistem mampu memberikan umpan balik yang akurat kepada pengguna serta mendeteksi dan menghitung jumlah repetisi latihan dengan benar.

3.4. Kerangka Penelitian

Kerangka penelitian ini menyajikan alur sistematis dalam menyelesaikan permasalahan penelitian, dimulai dari identifikasi masalah hingga kesimpulan. Diagram alur pada gambar 3.2 menggambarkan tahapan utama yang meliputi analisis, desain, implementasi, dan pengujian, serta proses implementasi seperti pendekripsi keyframe, konversi warna frame, dan analisis gerakan. Dengan visualisasi ini, penelitian dapat dilakukan secara terstruktur dan terarah.



Gambar 3. 2 Alur Penelitian

BAB IV IMPLEMENTASI

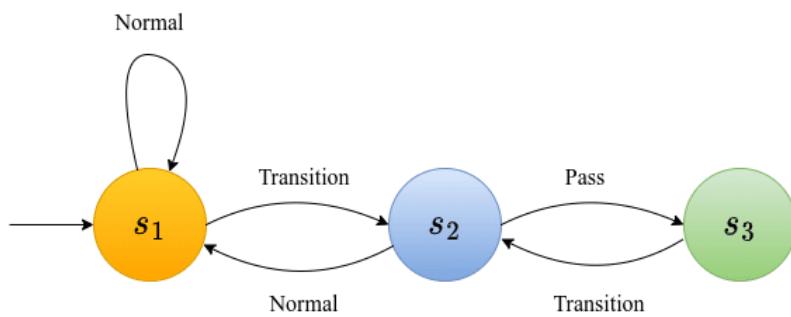
4.1 Analysis

Pada tahap analisis, penelitian ini berfokus pada pemahaman pemanfaatan MediaPipe dan OpenCV secara optimal dalam mendeteksi kesalahan gerakan saat melakukan latihan squat di gym. MediaPipe, sebagai framework yang menawarkan solusi pemrosesan pose tubuh secara real-time, memungkinkan pengambilan data berupa titik-titik penting tubuh seperti lutut, pinggul, bahu, dan pergelangan kaki. Data landmark tersebut kemudian dianalisis lebih lanjut menggunakan OpenCV, misalnya untuk menghitung sudut antara beberapa titik tubuh. Melalui

metode ini, sistem dapat memberikan umpan balik langsung mengenai kesalahan postur atau teknik yang sering terjadi, seperti posisi lutut yang terlalu maju melewati ujung kaki atau punggung yang tidak lurus saat squat. Hasil analisis memperlihatkan bahwa kombinasi kedua teknologi ini berpotensi besar meningkatkan ketepatan deteksi kesalahan gerakan dibandingkan metode manual yang biasanya mengandalkan pengamatan pelatih atau instruktur.

Selain itu, penelitian ini juga mengevaluasi implementasi aplikasi berbasis Streamlit sebagai platform interaktif untuk menampilkan hasil deteksi. Streamlit dipilih karena kemampuannya mengintegrasikan kode Python langsung ke dalam antarmuka web yang ramah pengguna. Melalui aplikasi ini, pengguna dapat memantau video atau tangkapan kamera secara real-time yang kemudian dianalisis oleh model deteksi. Hasil analisis, seperti indikator kesalahan gerakan dan rekomendasi perbaikan, disajikan secara visual dalam bentuk teks, grafik, atau overlay pada video. Diharapkan kedepannya aplikasi ini dapat responsif dan mampu memberikan umpan balik secara cepat, meskipun terdapat beberapa keterbatasan pada akurasi deteksi, terutama pada kondisi pencahayaan yang kurang baik atau saat pengguna mengenakan pakaian dengan warna serupa latar belakang.

4.1.1 State Diagram



Gambar 4. 1 State Transition Diagram pada Proses Deteksi Gerakan

Diagram state transition di atas menggambarkan alur perubahan status dalam proses analisis gerakan squat. Terdapat tiga state utama, yaitu S_1 (Normal), S_2 (Transition), dan S_3 (Pass). Proses dimulai dari state S_1 , yang merepresentasikan kondisi normal. Pada state ini, sistem akan tetap berada di S_1 selama mendeteksi gerakan normal. Jika terdeteksi adanya perubahan atau peralihan gerakan, sistem akan berpindah ke state S_2 (Transition). Dari S_2 ,

sistem dapat kembali ke S_1 jika gerakan kembali normal, atau melanjutkan ke S_3 (Pass) apabila gerakan telah melewati fase transisi dengan benar. Selain itu, dari S_3 , sistem dapat kembali ke S_2 jika terjadi transisi ulang. Diagram ini membantu memvisualisasikan bagaimana sistem mendeteksi dan mengklasifikasikan setiap fase gerakan squat secara berurutan dan dinamis.

4.1.2 Perhitungan Manual Sudut dan Thresholds

Analisis postur pada sistem ini dilakukan dengan cara mengukur sudut-sudut utama pada tubuh yang dibentuk dari susunan kerangka landmark. Cara ini memungkinkan penilaian pergerakan secara objektif dan terukur. Penghitungan sudut didasarkan pada koordinat (x, y) dari setiap landmark sendi yang diperoleh dari model MediaPipe Pose. Sudut-sudut yang dihasilkan kemudian dibandingkan dengan rentang nilai threshold yang telah ditentukan untuk memutuskan apakah suatu gerakan tergolong "benar" atau "salah".

Fungsi inti yang digunakan adalah $find_angle(p1, p2, ref_pt)$. Fungsi ini menghitung sudut antara dua vektor:

1. Vektor \vec{A} , yang membentang dari titik referensi (ref_pt) ke titik pertama ($p1$).
2. Vektor \vec{B} , yang membentang dari titik referensi (ref_pt) ke titik kedua ($p2$).

Rumus utama untuk menghitung sudut antara dua vektor adalah:

$$\theta = \arccos\left(\frac{\vec{A} \cdot \vec{B}}{|\vec{A}| |\vec{B}|}\right)$$

Dimana:

- $\vec{A} = p1 - ref_pt$
- $\vec{B} = p2 - ref_pt$
- \vec{A} dan \vec{B} merupakan sebuah vektor yang sudut nya akan dihitung
- $\vec{A} \cdot \vec{B}$ merupakan *dot product* dari kedua vektor
- $|\vec{A}|$ dan $|\vec{B}|$ adalah magnitudo atau panjang dari masing-masing vektor.

Berikut dijelaskan terkait simulasi perhitungan manual untuk setiap feedback kesalahan gerakan yang dihasilkan oleh sistem, beserta mekanisme perhitungan untuk squat yang dinilai benar. Perhitungan ini mengacu pada ambang batas (thresholds) pada mode beginner.

Skenario: Badan Terlalu Ke Belakang

Untuk feedback tersebut dipicu ketika sudut kemiringan punggung terhadap garis vertikal melebihi batas atas, yang menandakan pengguna terlalu bersandar dan condong ke belakang.

- Sudut Relevan: $hip_vertical_angle$
- Nilai Thresholds: HIP_THRESH adalah [10, 50], sehingga kondisi pemicu adalah jika sudut $> 50^\circ$

Contoh Perhitungan:

1. Koordinat Landmark (Hipotetis):

- Bahu ($shldr_coord$): (280, 400)
- Pinggul (hip_coord): (350, 410)

2. Perhitungan Vektor:

- \vec{A} (Vektor Pinggul ke Bahu): $(280 - 350, 400 - 410) = (-70, -10)$
- \vec{B} (Vektor Vertikal dari Pinggul): $(350 - 350, 0 - 410) = (0, -410)$

3. Perhitungan Dot Product:

- $\vec{A} \cdot \vec{B} = (-70 \times 0) + (-10 \times -410) = 4100$

4. Perhitungan Magnitudo:

- $|\vec{A}| = \sqrt{(-70)^2 + (-10)^2} = \sqrt{4900 + 100} = \sqrt{5000} \approx 70.71$
- $|\vec{B}| = \sqrt{0^2 + (-410)^2} = 410$

5. Perhitungan Sudut Akhir:

- $\theta = \arccos\left(\frac{4100}{70.71 \times 410}\right) = \arccos\left(\frac{4100}{28991.1}\right) = \arccos(0.1414) \approx 81.8^\circ$

Hasilnya sudut yang dihitung adalah 81.8° . Karena $81.8^\circ > 50^\circ$, kondisi terpenuhi dan sistem akan menampilkan umpan balik "BADAN TERLALU KE BELAKANG"

Skenario: Badan Terlalu Ke Depan

Feedback tersebut dipicu ketika sudut kemiringan tubuh terhadap garis vertikal berada di bawah batas minimum, menandakan punggung pengguna terlalu bungkuk ke depan.

- Sudut relevan: *hip_vertical_angle*
- Nilai Thresholds: *HIP_THRESH* adalah [10, 50], sehingga kondisi pemicu adalah jika sudut < 10°.

Contoh Perhitungan:

1. Koordinat Landmark (Hipotetis):

- Bahu (*shldr_coord*): (355, 250)
- Pinggul (*hip_coord*): (350, 410)

2. Pembentukan Vektor:

- $\vec{A} = (355 - 350, 250 - 410) = (5, -160)$
- $\vec{B} = (0, -410)$

3. Perhitungan Dot Product:

- $\vec{A} \cdot \vec{B} = (5 \times 0) + (-160 \times -410) = 65600$

4. Perhitungan Magnitudo:

- $|\vec{A}| = \sqrt{5^2 + (-160)^2} = \sqrt{25 + 25600} = \sqrt{25625} \approx 160.08$
- $|\vec{B}| = 410$

5. Perhitungan Sudut Akhir:

$$\theta = \arccos\left(\frac{65600}{160.08 \times 410}\right) = \arccos\left(\frac{65600}{65632.8}\right) = \arccos(0.9995) \approx 1.8^\circ$$

Hasilnya sudut yang dihitung adalah 1.8°. Karena 1.8° < 10°, kondisi terpenuhi dan sistem akan menampilkan feedback "BADAN TERLALU KE DEPAN".

Skenario: Lutut Melewati Jari Kaki

Feedback ini dipicu ketika sudut kemiringan tulang kering (tibia) terhadap garis vertikal terlalu besar.

- Sudut relevan: *ankle_vertical_angle*

- Nilai Thresholds: *ANKLE_THRESH* adalah 45, sehingga kondisi pemicu adalah jika sudut $> 45^\circ$.

Contoh Perhitungan:

1. Koordinat Landmark (Hipotetis):

- Lutut ('knee_coord'): '(440, 490)'
- Pergelangan Kaki ('ankle_coord'): '(360, 560)'

2. Perhitungan Vektor (diukur dengan referensi pada pergelangan kaki):

- \vec{A} (Vektor Pergelangan Kaki ke Lutut): $(440 - 360, 490 - 560) = (80, -70)$
- \vec{B} (Vektor Vertikal dari Pergelangan Kaki): $(360 - 360, 0 - 560) = (0, -560)$

3. Perhitungan Dot Product:

- $\vec{A} \cdot \vec{B} = (80 \times 0) + (-70 \times -560) = 39200$

4. Perhitungan Magnitudo:

- $|\vec{A}| = \sqrt{80^2 + (-70)^2} = \sqrt{6400 + 4900} = \sqrt{11300} \approx 106.3$
- $|\vec{B}| = 560$

5. Perhitungan Sudut Akhir:

- $\theta = \arccos\left(\frac{39200}{106.3 \times 560}\right) = \arccos\left(\frac{39200}{59528}\right) = \arccos(0.6585) \approx 48.8^\circ$

Didapatkan simulasi hasil sudut yang dihitung adalah 48.8° . Karena $48.8^\circ > 45^\circ$, kondisi terpenuhi dan sistem akan menampilkan feedback "LUTUT MELEWATI JARI KAKI".

Skenario: Squat Terlalu Dalam

Feedback ini muncul ketika pengguna turun melebihi batas kedalaman squat yang ideal, yang diukur dari sudut kemiringan paha dan juga pinggul.

- Sudut relevan: *knee_vertical_angle*
- Nilai Thresholds: *KNEE_THRESH* adalah [50, 70, 100], sehingga kondisi pemicu adalah jika sudut > 100°.

Contoh Perhitungan:

1. Koordinat Landmark (Hipotetis):

- Pinggul (*hip_coor*): (300, 510)
- Lutut (*knee_coord*): (420, 490)

2. Pembentukan Vektor: Sudut diukur dengan referensi pada lutut ('*knee_coord*').

- \vec{A} (Vektor Lutut ke Pinggul): $(300 - 420, 510 - 490) = (-120, 20)$
- \vec{B} (Vektor Vertikal dari Lutut): $(420 - 420, 0 - 490) = (0, -490)$

3. Perhitungan Dot Product:

- $\vec{A} \cdot \vec{B} = (-120 \times 0) + (20 \times -490) = -9800$

4. Perhitungan Magnitudo:

- $|\vec{A}| = \sqrt{(-120)^2 + 20^2} = \sqrt{14400 + 400} = \sqrt{14800} \approx 121.65$
- $|\vec{B}| = 490$

5. Perhitungan Sudut Akhir:

- $\theta = \arccos\left(\frac{-9800}{121.65 \times 490}\right) = \arccos\left(\frac{-9800}{59608.5}\right) = \arccos(-0.1644) \approx 99.45^\circ$

Berdasarkan hasil perhitungan tersebut, sudut yang terbentuk di lutut adalah sekitar 99.45°. Nilai ini mendekati batas bawah threshold yang ditetapkan, misalnya 100°. Jika pengguna melakukan gerakan squat sedikit lebih dalam, misalkan koordinat pinggul berubah menjadi (290, 515), maka sudut yang dihasilkan akan melebihi 100°. Pada kondisi tersebut, sistem akan memberikan umpan balik berupa peringatan "SQUAT TERLALU DALAM" kepada pengguna.

Skenario: Squat Benar (Correct Squat)

Pada kondisi squat yang benar, seluruh sudut utama tubuh berada dalam rentang yang dapat diterima pada tiap fase gerakan (state s1 hingga state s3 atau PASS). Sistem akan mengidentifikasi repetisi squat sebagai valid jika seluruh parameter sudut berada dalam batas yang telah ditentukan.

- Kondisi:
 - *hip_vertical_angle* berada di antara 10° dan 50° .
 - *ankle_vertical_angle* tidak melebihi 45° .
 - *knee_vertical_angle* berada dalam rentang PASS, yaitu antara 70° dan 100° .

Contoh Perhitungan:

1. Koordinat Landmark (Hipotetis):

- Bahu (*shldr_coord*): (400, 390)
- Pinggul (*hip_coord*): (320, 490)
- Lutut (*knee_coord*): (420, 500)
- Pergelangan Kaki (*ankle_coord*): (400, 570)

2. Perhitungan 'hip_vertical_angle':

- Vektor \vec{A} (Pinggul ke Bahu): (80, -100)
- Vektor \vec{B} (Vertikal Pinggul): (0, -490)
- $\theta_{\text{hip}} = \arccos\left(\frac{49000}{\sqrt{16400 \times 4900}}\right) = \arccos(0.781) \approx 38.6^\circ$

Interpretasi: 38.6° berada dalam rentang $(10^\circ, 50^\circ)$, sehingga dinyatakan benar.

3. Perhitungan *ankle_vertical_angle*:

- Vektor \vec{A} (Ankle ke Lutut): (20, -70)
- Vektor \vec{B} (Vertikal Ankle): (0, -570)
- $\theta_{\text{ankle}} = \arccos\left(\frac{39900}{\sqrt{5300 \times 5700}}\right) = \arccos(0.961) \approx 16.0^\circ$

Interpretasi: $16.0^\circ < 45^\circ$, sehingga dinyatakan benar.

4. Perhitungan *knee_vertical_angle*:

- Vektor \vec{A} (Lutut ke Pinggul): (-100, -10)
- Vektor \vec{B} (Vertikal Lutut): (0, -500)
- $\theta_{\text{knee}} = \arccos\left(\frac{5000}{\sqrt{10100} \times 500}\right) = \arccos(0.0995) \approx 84.3^\circ$

Interpretasi: 84.3° berada dalam rentang *PASS* ($70^\circ, 100^\circ$), sehingga dinyatakan benar.

Karena semua sudut berada dalam rentang yang telah ditentukan, sistem menganggap gerakan ini sebagai repetisi squat yang valid dan akan menambah jumlah hitungan squat (*Squat_Count*) ketika pengguna kembali ke posisi awal (*state s1*).

4.1.3 Perhitungan Manual Metrik *Mean Per Joint Position Error* (MPJPE)

Dalam evaluasi performa model estimasi postur, diperlukan sebuah metrik kuantitatif untuk mengukur tingkat akurasi. Salah satu metrik yang sangat umum digunakan baik dalam 2D/3D *Pose Estimation* adalah *Mean Per Joint Position Error* (MPJPE). Pada bagian ini akan dijelaskan secara rinci metodologi perhitungan MPJPE, mulai dari definisi matematis hingga simulasi perhitungan manual pada skenario dengan tingkat akurasi tinggi.

Mean Per Joint Position Error (MPJPE) didefinisikan sebagai rata-rata *Euclidean Distance* antara koordinat persendian (landmark) yang diprediksi oleh model dan koordinat persendian referensi (ground truth). Metrik ini memberikan ukuran kesalahan (error) rata-rata dalam satuan piksel (px). Nilai MPJPE yang lebih rendah mengindikasikan akurasi model yang lebih tinggi.

Perhitungan MPJPE didasarkan pada dua rumus utama berikut:

- *Euclidean Distance (Per Joint Position Error - PJPE)*

Untuk mengetahui seberapa akurat model mendekripsi posisi sendi, pertama-tama dihitung jarak lurus antara titik prediksi $P_p = (x_p, y_p)$ dan titik *ground truth* $P_{gt} = (x_{gt}, y_{gt})$. Di dalam code python rumus ini diimplementasikan dalam kode menggunakan fungsi `np.linalg.norm()`

$$E_{\text{joint}} = \sqrt{(x_p - x_{gt})^2 + (y_p - y_{gt})^2}$$

- *Mean Per Joint Position Error (MPJPE)*

Setelah memperoleh nilai error untuk setiap sendi, langkah berikutnya adalah menghitung rata-rata dari semua error tersebut. Nilai rata-rata inilah yang disebut sebagai MPJPE. Secara matematis, MPJPE dirumuskan sebagai berikut:

$$\text{MPJPE} = \frac{\sum_{i=1}^N E_{\text{joint}_i}}{N_{\text{joints}}}$$

Di mana N_{joints} adalah jumlah total sendi yang dievaluasi. Dengan kata lain, MPJPE adalah total error seluruh sendi dibagi dengan jumlah sendi. Nilai MPJPE yang lebih kecil menunjukkan prediksi model yang semakin mendekati data referensi (ground truth), sehingga akurasi model semakin baik.

Untuk memberikan gambaran yang lebih jelas, berikut adalah contoh simulasi perhitungan manual MPJPE pada satu frame, dengan asumsi bahwa perbedaan antara prediksi dan ground truth relatif kecil (menandakan akurasi tinggi):

Penetapan Koordinat Hipotetis:

Misalkan sistem memproses sebuah *frame* dan menghasilkan set koordinat berikut, di mana selisih antara prediksi dan *ground truth* terlihat seperti tabel dibawah ini:

Sendi (Joint)	Koordinat Prediksi $P_p = (x_p, y_p)$	Koordinat Ground Truth $P_{gt} = (x_{gt}, y_{gt})$
Shoulder	(150, 210)	(154, 214.5)
Elbow	(180, 280)	(186, 285.3)
Wrist	(200, 350)	(202, 352.2)
Hip	(140, 340)	(147, 345.6)
Knee	(130, 450)	(135, 456.2)
Ankle	(120, 550)	(128, 554.3)
Foot	(140, 580)	(145, 583.3)

Perhitungan Per Joint Position Error (PJPE):

Dengan menggunakan rumus *Euclidean Distance*, error untuk setiap sendi dihitung sebagai berikut:

- Shoulder:

$$\begin{aligned} E_{\text{shoulder}} &= \sqrt{(150 - 154)^2 + (210 - 214.5)^2} = \sqrt{(-4)^2 + (-4.5)^2} \\ &= \sqrt{16 + 20.25} = \sqrt{36.25} \approx 6.02 \text{ px} \end{aligned}$$

- Elbow:

$$\begin{aligned} E_{\text{elbow}} &= \sqrt{(180 - 186)^2 + (280 - 285.3)^2} = \sqrt{(-6)^2 + (-5.3)^2} \\ &= \sqrt{36 + 28.09} = \sqrt{64.09} \approx 8.01 \text{ px} \end{aligned}$$

- Wrist:

$$\begin{aligned} E_{\text{wrist}} &= \sqrt{(200 - 202)^2 + (350 - 352.2)^2} = \sqrt{(-2)^2 + (-2.2)^2} \\ &= \sqrt{4 + 4.84} = \sqrt{8.84} \approx 2.97 \text{ px} \end{aligned}$$

(Nilai error yang sangat kecil ini menunjukkan akurasi yang sangat tinggi untuk deteksi pergelangan tangan pada frame ini)

- Hip:

$$\begin{aligned} E_{\text{hip}} &= \sqrt{(140 - 147)^2 + (340 - 345.6)^2} = \sqrt{(-7)^2 + (-5.6)^2} \\ &= \sqrt{49 + 31.36} = \sqrt{80.36} \approx 8.96 \text{ px} \end{aligned}$$

- Knee:

$$\begin{aligned} E_{\text{knee}} &= \sqrt{(130 - 135)^2 + (450 - 456.2)^2} = \sqrt{(-5)^2 + (-6.2)^2} \\ &= \sqrt{25 + 38.44} = \sqrt{63.44} \approx 7.96 \text{ px} \end{aligned}$$

- Ankle:

$$\begin{aligned} E_{\text{ankle}} &= \sqrt{(120 - 128)^2 + (550 - 554.3)^2} = \sqrt{(-8)^2 + (-4.3)^2} \\ &= \sqrt{64 + 18.49} = \sqrt{82.49} \approx 9.08 \text{ px} \end{aligned}$$

- Foot:

$$\begin{aligned} E_{\text{foot}} &= \sqrt{(140 - 145)^2 + (580 - 583.3)^2} = \sqrt{(-5)^2 + (-3.3)^2} \\ &= \sqrt{25 + 10.89} = \sqrt{35.89} \approx 5.99 \text{ px} \end{aligned}$$

Perhitungan Nilai Akhir MPJPE:

Nilai MPJPE dihitung dengan merata-ratakan semua nilai PJPE yang diperoleh pada step sebelumnya

- Total Error (Jumlah PJPE):

$$\sum E_{\text{joint}} = 6.02 + 8.01 + 2.97 + 8.96 + 7.96 + 9.08 + 5.99 = 48.99$$

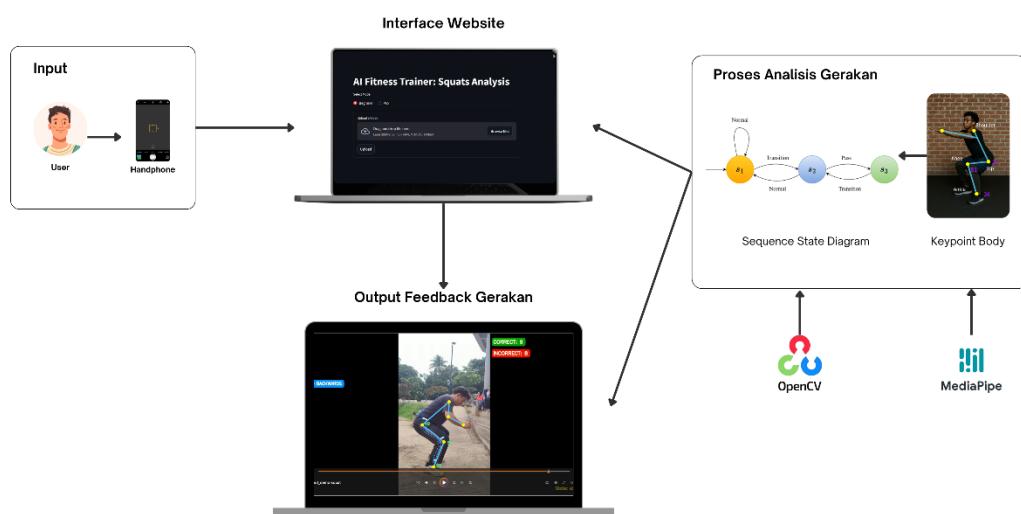
- MPJPE (Rata-rata):

$$\text{MPJPE} = \frac{48.99}{7} \approx 7.00 \text{ px}$$

Nilai MPJPE sekitar 7 piksel ini menunjukkan bahwa rata-rata prediksi posisi sendi model hanya berselisih sekitar 7 piksel dari *ground truth*. Hal ini mengindikasikan bahwa evaluasi analisis video tersebut memiliki tingkat akurasi yang sangat baik pada skenario tersebut.

4.2 Design

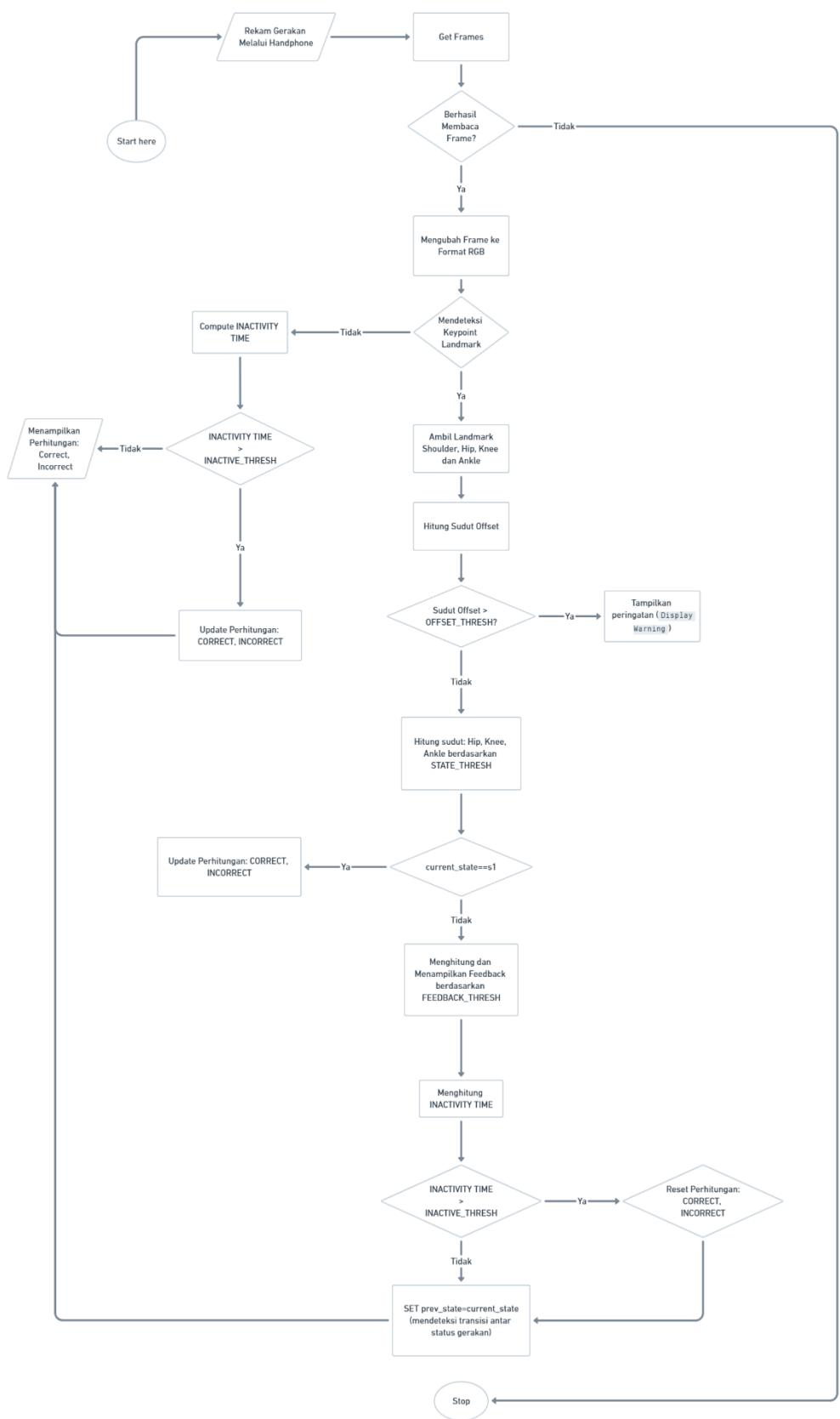
4.2.1 Desain Arsitektur Workflow Sistem



Gambar 4. 2 Arsitektur Sistem Deteksi dan Analisis Gerakan Squat

Desain arsitektur fitness vision memiliki 3 bagian yaitu: input, proses analisis gerakan squat, dan output. Tahapan awal dimulai ketika pengguna merekam aktivitas squat menggunakan kamera, lalu mengunggah video tersebut ke aplikasi web. Setelah proses unggah selesai, sistem memanfaatkan MediaPipe untuk mendeteksi pose tubuh dan menandai titik-titik penting seperti lutut, pinggul, bahu, serta pergelangan kaki. Landmark yang dihasilkan kemudian dianalisis oleh OpenCV untuk menghitung sudut-sudut dan menentukan fase gerakan sesuai dengan state diagram yang telah dibuat (S_1 : Normal, S_2 : Transition, S_3 : Pass). Seluruh hasil analisis ini kemudian divisualisasikan di dalam aplikasi, sehingga pengguna dapat memperoleh umpan balik berupa jumlah gerakan yang benar maupun salah, serta tampilan visual pose tubuh secara real-time. Dengan alur kerja seperti ini, sistem dapat memberikan feedback yang interaktif dan informatif mengenai kualitas gerakan squat yang dilakukan oleh pengguna.

4.2.2 Flowchart Fitness Vision



Gambar 4. 3 Flowchart Fitness Vision

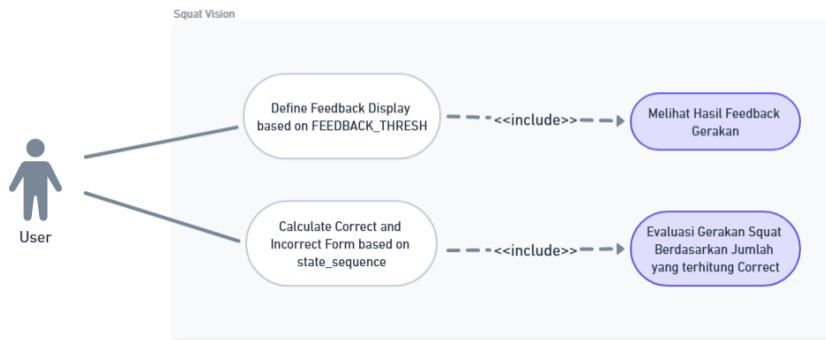
Flowchart Fitness Vision diawali dengan proses pengambilan frame melalui kamera handphone, yang kemudian dievaluasi apakah berhasil membaca frame atau tidak. Jika berhasil, frame akan dikonversi ke format RGB untuk mempermudah pemrosesan. Selanjutnya, sistem melakukan pendekripsi keypoint landmark pada tubuh, seperti bahu (shoulder), pinggul (hip), lutut (knee), dan pergelangan kaki (ankle). Jika deteksi landmark berhasil, sistem menghitung sudut offset antara titik-titik tersebut. Jika nilai sudut offset melebihi threshold (OFFSET_THRESH), maka tampilan peringatan akan ditampilkan kepada pengguna.

Setelah itu, sistem menghitung sudut-sudut kunci berdasarkan posisi hip, knee, dan ankle, serta mengevaluasi apakah gerakan squat sesuai dengan kondisi yang telah ditentukan dalam STATE_THRESH. Berdasarkan hasil evaluasi ini, sistem memberikan feedback secara real-time, baik dalam bentuk peringatan kesalahan maupun konfirmasi jika gerakan dilakukan dengan benar. Proses selanjutnya termasuk penghitungan waktu inaktivitas (INACTIVITY_TIME) untuk memantau apakah pengguna tetap aktif dalam latihan. Jika waktu inaktivitas melebihi batas tertentu (INACTIVE_THRESH), sistem akan mereset perhitungan status gerakan. Terakhir, sistem juga mencatat transisi antar status gerakan untuk analisis lebih lanjut. Proses ini berlangsung secara kontinu hingga pengguna menghentikan aplikasi.

4.2.3 Use Case Diagram

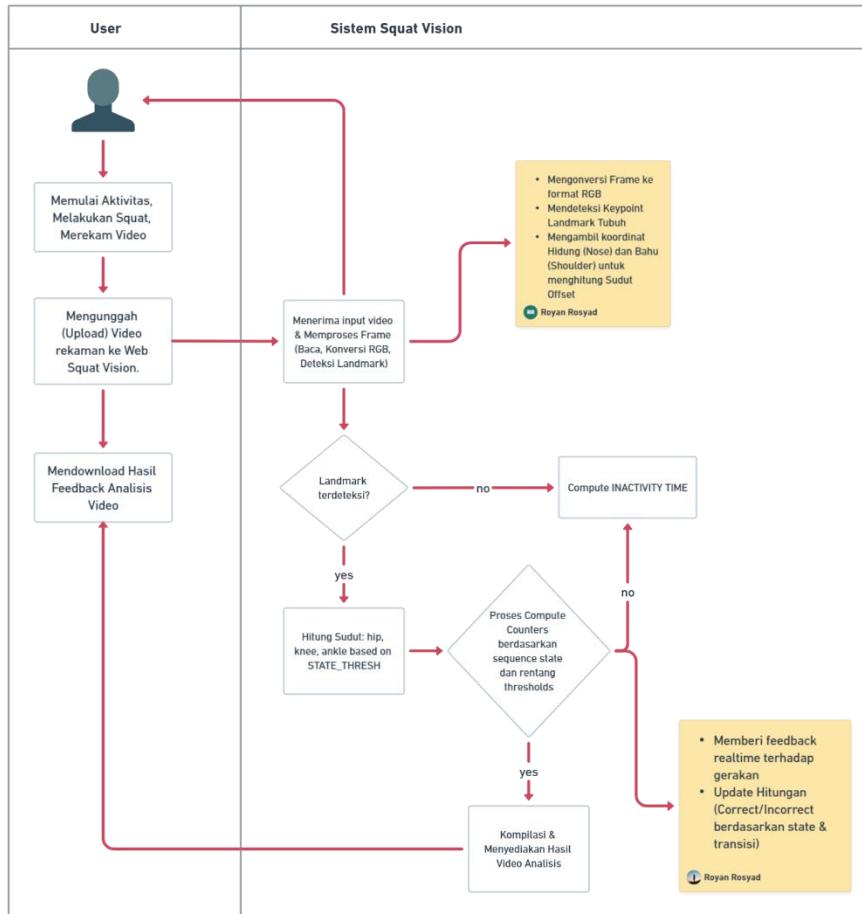
Diagram Use Case ini menggambarkan bagaimana pengguna berinteraksi dengan sistem Fitness Vision, sekaligus menampilkan fungsi-fungsi utama yang tersedia dalam mendekripsi kesalahan gerakan squat. Pengguna berpartisipasi dalam dua use case utama: *Define Feedback Display based on FEEDBACK_THRESH* dan *Calculate Correct and Incorrect Form based on state_sequence*. Pada use case pertama, pengguna dapat menerima umpan balik gerakan secara real-time, berupa peringatan atau konfirmasi yang menunjukkan apakah posisi tubuh saat squat sudah benar atau masih salah. Sementara itu, use case kedua memberikan evaluasi terhadap gerakan squat berdasarkan jumlah repetisi yang dilakukan dengan tepat. Diagram ini juga memperlihatkan hubungan antara kedua use case tersebut, yang

saling melengkapi untuk memberikan gambaran menyeluruh tentang performa latihan pengguna.



Gambar 4. 4 Use Case Diagram Fitness Vision

4.2.4 Activity Diagram



Gambar 4. 5 Activity Diagram Fitness Vision

Activity Diagram pada Gambar 4.5 menggambarkan alur aktivitas antara pengguna dan sistem Fitness Vision dalam proses analisis gerakan squat. Diagram ini dimulai dari pengguna yang melakukan aktivitas squat dan merekam video gerakan, kemudian mengunggah hasil rekaman ke sistem. Sistem menerima input video, mendeteksi pose melalui landmark, menghitung squat, serta membedakan gerakan yang benar dan salah berdasarkan *state_sequence*. Jika terjadi ketidakaktifan, sistem menghitung waktu inaktivitas untuk evaluasi lebih lanjut. Setelah proses selesai, sistem memberikan umpan balik terhadap gerakan pengguna dan menampilkan hasil analisis video yang dapat diunduh oleh pengguna.

4.3 Implementasi

Pada tahap ini proses penulisan code dilakukan menggunakan IDE Visual Studio Code. Bagian ini menjelaskan tahapan implementasi sistem deteksi kesalahan squat, mulai dari penentuan threshold, ekstraksi fitur pose, hingga logika deteksi dan feedback pada pengguna.

4.3.1 Penggunaan Model Mediapipe Pose

Langkah pertama dalam implementasi sistem deteksi kesalahan squat adalah inisialisasi model MediaPipe Pose. MediaPipe Pose merupakan solusi deteksi pose yang dikembangkan oleh Google yang mampu mengestimasi 33 titik landmark pada tubuh manusia. Dalam sistem ini, model MediaPipe Pose menjadi komponen inti yang bertanggung jawab untuk mendeteksi posisi dan orientasi tubuh pengguna saat melakukan gerakan squat.

```
import cv2
import mediapipe as mp
import numpy as np

def draw_rounded_rect(img, rect_start, rect_end, corner_width,
box_color):

    x1, y1 = rect_start
    x2, y2 = rect_end
    w = corner_width

    # draw filled rectangles
    cv2.rectangle(img, (x1 + w, y1), (x2 - w, y1 + w),
box_color, -1)
    cv2.rectangle(img, (x1 + w, y2 - w), (x2 - w, y2),
box_color, -1)
    cv2.rectangle(img, (x1, y1 + w), (x1 + w, y2 - w),
box_color, -1)
    cv2.rectangle(img, (x2 - w, y1 + w), (x2, y2 - w),
box_color, -1)
```

```

        cv2.rectangle(img, (x1 + w, y1 + w), (x2 - w, y2 - w),
box_color, -1)

    # draw filled ellipses
    cv2.ellipse(img, (x1 + w, y1 + w), (w, w),
                angle = 0, startAngle = -90, endAngle = -180,
color = box_color, thickness = -1)

    cv2.ellipse(img, (x2 - w, y1 + w), (w, w),
                angle = 0, startAngle = 0, endAngle = -90, color
= box_color, thickness = -1)

    cv2.ellipse(img, (x1 + w, y2 - w), (w, w),
                angle = 0, startAngle = 90, endAngle = 180,
color = box_color, thickness = -1)

    cv2.ellipse(img, (x2 - w, y2 - w), (w, w),
                angle = 0, startAngle = 0, endAngle = 90, color
= box_color, thickness = -1)

    return img

def draw_dotted_line(frame, lm_coord, start, end, line_color):
pix_step = 0

for i in range(start, end+1, 8):
    cv2.circle(frame, (lm_coord[0], i+pix_step), 2,
line_color, -1, lineType=cv2.LINE_AA)

return frame

def draw_text(
    img,
    msg,
    width = 8,
    font=cv2.FONT_HERSHEY_SIMPLEX,
    pos=(0, 0),
    font_scale=1,
    font_thickness=2,
    text_color=(0, 255, 0),
    text_color_bg=(0, 0, 0),
    box_offset=(20, 10),
):
    offset = box_offset
    x, y = pos
    text_size, _ = cv2.getTextSize(msg, font, font_scale,
font_thickness)
    text_w, text_h = text_size
    rec_start = tuple(p - o for p, o in zip(pos, offset))
    rec_end = tuple(m + n - o for m, n, o in zip((x + text_w, y
+ text_h), offset, (25, 0)))

```

```

        img = draw_rounded_rect(img, rec_start, rec_end, width,
text_color_bg)

        cv2.putText(
            img,
            msg,
            (int(rec_start[0] + 6), int(y + text_h + font_scale -
1)),
            font,
            font_scale,
            text_color,
            font_thickness,
            cv2.LINE_AA,
        )

    return text_size

def find_angle(p1, p2, ref_pt = np.array([0,0])):
    p1_ref = p1 - ref_pt
    p2_ref = p2 - ref_pt

    cos_theta = (np.dot(p1_ref,p2_ref)) / (1.0 *
np.linalg.norm(p1_ref) * np.linalg.norm(p2_ref))
    theta = np.arccos(np.clip(cos_theta, -1.0, 1.0))

    degree = int(180 / np.pi) * theta

    return int(degree)

def get_landmark_array(pose_landmark, key, frame_width,
frame_height):
    denorm_x = int(pose_landmark[key].x * frame_width)
    denorm_y = int(pose_landmark[key].y * frame_height)

    return np.array([denorm_x, denorm_y])

def get_landmark_features(kp_results, dict_features, feature,
frame_width, frame_height):

    if feature == 'nose':
        return get_landmark_array(kp_results,
dict_features[feature], frame_width, frame_height)

    elif feature == 'left' or 'right':
        shldr_coord = get_landmark_array(kp_results,
dict_features[feature]['shoulder'], frame_width, frame_height)
        elbow_coord = get_landmark_array(kp_results,
dict_features[feature]['elbow'], frame_width, frame_height)
        wrist_coord = get_landmark_array(kp_results,
dict_features[feature]['wrist'], frame_width, frame_height)

```

```

        hip_coord    = get_landmark_array(kp_results,
dict_features[feature]['hip'], frame_width, frame_height)
        knee_coord   = get_landmark_array(kp_results,
dict_features[feature]['knee'], frame_width, frame_height)
        ankle_coord  = get_landmark_array(kp_results,
dict_features[feature]['ankle'], frame_width, frame_height)
        foot_coord   = get_landmark_array(kp_results,
dict_features[feature]['foot'], frame_width, frame_height)

        return shldr_coord, elbow_coord, wrist_coord, hip_coord,
knee_coord, ankle_coord, foot_coord

    else:
        raise ValueError("feature needs to be either 'nose',
'left' or 'right')

def get_mediapipe_pose(
    static_image_mode = False,
    model_complexity = 1,
    smooth_landmarks = True,
    min_detection_confidence = 0.5,
    min_tracking_confidence = 0.5

):
    pose = mp.solutions.pose.Pose(
        static_image_mode =
        static_image_mode,
        model_complexity =
        model_complexity,
        smooth_landmarks =
        smooth_landmarks,
        min_detection_confidence =
        min_detection_confidence,
        min_tracking_confidence =
        min_tracking_confidence
    )
    return pose

```

4.3.2 Penggunaan Landmark yang dibutuhkan

Setelah model MediaPipe Pose berhasil diinisialisasi dan digunakan untuk mendekripsi pose pada setiap frame, langkah berikutnya adalah mengekstraksi titik-titik kunci (landmark) tubuh yang relevan untuk analisis gerakan squat. Proses ini sangat penting karena setiap landmark mewakili posisi anatomi tertentu pada tubuh, seperti bahu, siku, pinggul, lutut, pergelangan kaki, dan kaki.

Untuk memudahkan proses ekstraksi dan pemetaan, sistem menggunakan struktur dictionary yang mengelompokkan indeks landmark berdasarkan sisi tubuh (kiri dan kanan) serta titik pusat (nose). Dengan pendekatan ini, pengambilan koordinat landmark dari hasil deteksi MediaPipe menjadi lebih terstruktur dan efisien, sehingga dapat langsung digunakan untuk perhitungan sudut maupun analisis postur.

Berikut adalah kode yang digunakan untuk mendefinisikan dan mengelompokkan indeks landmark utama:

```
# Dictionary to maintain the various landmark features.
self.dict_features = {}
self.left_features = {
    'shoulder': 11,
    'elbow' : 13,
    'wrist' : 15,
    'hip' : 23,
    'knee' : 25,
    'ankle' : 27,
    'foot' : 31
}

self.right_features = {
    'shoulder': 12,
    'elbow' : 14,
    'wrist' : 16,
    'hip' : 24,
    'knee' : 26,
    'ankle' : 28,
    'foot' : 32
}

self.dict_features['left'] = self.left_features
self.dict_features['right'] = self.right_features
self.dict_features['nose'] = 0
```

4.3.3 Perhitungan Sudut antar Landmark dan State Sequence Squat

Pada tahap ini, sistem menghitung sudut antar titik kunci (landmark) tubuh seperti bahu, pinggul, lutut, dan pergelangan kaki. Sudut-sudut ini digunakan untuk menentukan fase gerakan squat dan melakukan state tracking.

Angle Calculation:

```
def find_angle(p1, p2, ref_pt = np.array([0, 0])):
```

```

p1_ref = p1 - ref_pt
p2_ref = p2 - ref_pt

cos_theta = (np.dot(p1_ref,p2_ref)) / (1.0 *
np.linalg.norm(p1_ref) * np.linalg.norm(p2_ref))
theta = np.arccos(np.clip(cos_theta, -1.0, 1.0))

degree = int(180 / np.pi) * theta

return int(degree)

```

Urutan State:

```

def _get_state(self, knee_angle):
    knee = None
    if self.thresholds['HIP_KNEE_VERT']['NORMAL'][0] <=
knee_angle <= self.thresholds['HIP_KNEE_VERT']['NORMAL'][1]:
        knee = 1
    elif self.thresholds['HIP_KNEE_VERT']['TRANS'][0] <=
knee_angle <= self.thresholds['HIP_KNEE_VERT']['TRANS'][1]:
        knee = 2
    elif self.thresholds['HIP_KNEE_VERT']['PASS'][0] <=
knee_angle <= self.thresholds['HIP_KNEE_VERT']['PASS'][1]:
        knee = 3
    return f's{knee}' if knee else None

def _update_state_sequence(self, state):
    if state == 's2':
        if (('s3' not in self.state_tracker['state_seq']) and
(self.state_tracker['state_seq'].count('s2'))==0) or \
            (('s3' in self.state_tracker['state_seq']) and
(self.state_tracker['state_seq'].count('s2'))==1)):
            self.state_tracker['state_seq'].append(state)
    elif state == 's3':
        if (state not in self.state_tracker['state_seq']) and
's2' in self.state_tracker['state_seq']:
            self.state_tracker['state_seq'].append(state)

```

Fungsi `find_angle()` menggunakan trigonometri dan aljabar vektor untuk menghitung sudut antara dua titik landmark relatif terhadap titik referensi. Proses perhitungannya dapat diuraikan dalam beberapa tahap:

6. Menghitung Vektor Relatif:

Fungsi pertama-tama menghitung vektor relatif dari kedua titik terhadap titik referensi. Misalnya untuk menghitung sudut lutut

- Hip = [100, 150]
- Knee = [100, 250]

- Titik Vertikal = [100, 0]

Sehingga Vektor Relatif nya:

- $p1_ref = \text{Hip} - \text{Knee} = [100, 150] - [100, 250] = [0, -100]$
- $p2_ref = \text{Titik_vertikal} - \text{Knee} = [100, 0] - [100, 250] = [0, -250]$

7. Menghitung Dot Product

Dot product dari kedua vektor dihitung menggunakan rumus:

- $p1_ref \cdot p2_ref = (p1_ref_x \times p2_ref_x) + (p1_ref_y \times p2_ref_y)$
- $\text{dot_product} = (0 \times 0) + (-100 \times -250) = 25000$

3. Menghitung Panjang Vektor

Panjang (magnitudo) setiap vektor dihitung menggunakan rumus Pythagoras:

$$|v| = \sqrt{(v_x^2 + v_y^2)}$$

Untuk $p1_ref$:

$$|p1_ref| = \sqrt{(0^2 + (-100)^2)} = 100$$

Untuk $p2_ref$:

$$|p2_ref| = \sqrt{(0^2 + (-250)^2)} = 250$$

4. Perhitungan Sudut Cosinus

Perhitungan nilai kosinus dari sudut (θ) antara dua vektor dapat diperoleh menggunakan persamaan berikut:

$$\cos(\theta) = (p1_ref \cdot p2_ref) / (|p1_ref| \times |p2_ref|)$$

Sebagai ilustrasi perhitungan:

$$\cos(\theta) = 25000 / (100 \times 250) = 1,0$$

4.3.4 Implementasi Thresholds Transisi Gerakan

Threshold digunakan untuk membedakan antara gerakan squat yang benar dan salah berdasarkan sudut-sudut yang telah dihitung. Setiap threshold

diatur sesuai mode (Beginner/Pro) dan digunakan dalam logika deteksi kesalahan.

```
# Get thresholds for beginner mode
def get_thresholds_beginner():

    _ANGLE_HIP_KNEE_VERT = {
        'NORMAL' : (0, 32),
        'TRANS' : (35, 65),
        'PASS' : (70, 95)
    }

    thresholds = {
        'HIP_KNEE_VERT': _ANGLE_HIP_KNEE_VERT,
        'HIP_THRESH' : [10, 50],
        'ANKLE_THRESH' : 45,
        'KNEE_THRESH' : [50, 70, 95],
        'OFFSET_THRESH' : 35.0,
        'INACTIVE_THRESH' : 15.0,
        'CNT_FRAME_THRESH' : 50,
        # MPJPE visualization settings
        'VISUALIZE_MPJPE_COMPARISON': False
    }

    return thresholds


# Get thresholds for beginner mode
def get_thresholds_pro():

    _ANGLE_HIP_KNEE_VERT = {
        'NORMAL' : (0, 32),
        'TRANS' : (35, 65),
        'PASS' : (80, 95)
    }

    thresholds = {
        'HIP_KNEE_VERT': _ANGLE_HIP_KNEE_VERT,
        'HIP_THRESH' : [15, 50],
        'ANKLE_THRESH' : 30,
        'KNEE_THRESH' : [50, 80, 95],
        'OFFSET_THRESH' : 35.0,
        'INACTIVE_THRESH' : 15.0,
        'CNT_FRAME_THRESH' : 50,
```

```

        # MPJPE visualization settings
        'VISUALIZE_MPJPE_COMPARISON': False
    }

return thresholds

```

Penggunaan Threshold dalam Deteksi:

```

else:
    if hip_vertical_angle > self.thresholds['HIP_THRESH'][1]:
        self.state_tracker['DISPLAY_TEXT'][0] = True

    elif hip_vertical_angle < self.thresholds['HIP_THRESH'][0]
        and \
            self.state_tracker['state_seq'].count('s2') == 1:
                self.state_tracker['DISPLAY_TEXT'][1] = True

    if self.thresholds['KNEE_THRESH'][0] < knee_vertical_angle <
        self.thresholds['KNEE_THRESH'][1] and \
            self.state_tracker['state_seq'].count('s2') == 1:
                self.state_tracker['LOWER_HIPS'] = True

    elif knee_vertical_angle >
        self.thresholds['KNEE_THRESH'][2]:
        self.state_tracker['DISPLAY_TEXT'][3] = True
        self.state_tracker['INCORRECT_POSTURE'] = True

    if (ankle_vertical_angle > self.thresholds['ANKLE_THRESH']):
        self.state_tracker['DISPLAY_TEXT'][2] = True
        self.state_tracker['INCORRECT_POSTURE'] = True

```

4.3.5 Perhitungan Repetisi dan Feedback Visual

Sistem menghitung jumlah repetisi squat yang benar dan salah berdasarkan urutan state yang terdeteksi. Selain itu, feedback visual diberikan secara real-time pada frame video untuk membantu pengguna memperbaiki postur.

```

if current_state == 's1':
    if len(self.state_tracker['state_seq']) == 3 and not
        self.state_tracker['INCORRECT_POSTURE']:

```

```

        self.state_tracker['SQUAT_COUNT']+=1
        play_sound = str(self.state_tracker['SQUAT_COUNT'])

    elif 's2' in self.state_tracker['state_seq'] and
len(self.state_tracker['state_seq'])==1:
        self.state_tracker['IMPROPER_SQUAT']+=1
        play_sound = 'incorrect'

    elif self.state_tracker['INCORRECT_POSTURE']:
        self.state_tracker['IMPROPER_SQUAT']+=1
        play_sound = 'incorrect'

    self.state_tracker['state_seq'] = []
    self.state_tracker['INCORRECT_POSTURE'] = False

```

Kode Feedback Visual:

```

def _show_feedback(self, frame, c_frame, dict_maps,
lower_hips_disp):
    if lower_hips_disp:
        draw_text(
            frame,
            'LOWER YOUR HIPS',
            pos=(30, 80),
            text_color=(0, 0, 0),
            font_scale=0.6,
            text_color_bg=(255, 255, 0)
        )
    for idx in np.where(c_frame)[0]:
        draw_text(
            frame,
            dict_maps[idx][0],
            pos=(30, dict_maps[idx][1]),
            text_color=(255, 255, 230),
            font_scale=0.6,
            text_color_bg=dict_maps[idx][2]
        )
    return frame

# Pemanggilan Feedback dan Counter pada Frame
draw_text(
    frame,
    "CORRECT: " + str(self.state_tracker['SQUAT_COUNT']),
    pos=(int(frame_width*0.68), 30),
    text_color=(255, 255, 230),
    font_scale=0.7,
    text_color_bg=(18, 185, 0)
)

draw_text(
    frame,

```

```

    "INCORRECT: " + str(self.state_tracker['IMPROPER_SQUAT']),
    pos=(int(frame_width*0.68), 80),
    text_color=(255, 255, 230),
    font_scale=0.7,
    text_color_bg=(221, 0, 0),
)

# Feedback kesalahan gerakan

self.FEEDBACK_ID_MAP = {
    0: ('BADAN TERLALU KE BELAKANG', 215, (0, 153, 255)),
    1: ('BADAN TERLALU KE DEPAN', 215, (0, 153, 255)),
    2: ('LUTUT MELEWATI JARI KAKI', 170, (255, 80, 80)),
    3: ('SQUAT TERLALU DALAM', 125, (255, 80, 80))
}

```

4.3.6 Mekanisme Inaktivitas (Reset Counter)

Setelah sistem melakukan perhitungan repetisi squat dan memberikan feedback visual secara real-time, langkah berikutnya adalah mengantisipasi kondisi di mana pengguna berhenti bergerak atau tidak melakukan squat dalam jangka waktu tertentu. Untuk menjaga akurasi penghitungan dan mencegah akumulasi data yang tidak relevan, sistem menerapkan mekanisme reset counter berbasis inaktivitas.

Mekanisme ini bekerja dengan memantau perubahan state gerakan squat. Jika state tidak berubah dalam periode waktu tertentu (berdasarkan threshold INACTIVE_THRESH), maka sistem akan menganggap pengguna sedang tidak aktif dan otomatis mereset penghitung repetisi squat (SQUAT_COUNT) dan squat tidak benar (IMPROPER_SQUAT). Selain itu, sistem juga memberikan notifikasi visual pada frame untuk menginformasikan bahwa counter telah di-reset akibat inaktivitas.

```

# Compute Inactivity

if self.state_tracker['curr_state'] ==
self.state_tracker['prev_state']:
    end_time = time.perf_counter()
    self.state_tracker['INACTIVE_TIME'] += end_time -
self.state_tracker['start_inactive_time']
    self.state_tracker['start_inactive_time'] = end_time

```

```

        if self.state_tracker['INACTIVE_TIME'] >=
self.thresholds['INACTIVE_THRESH']:
    self.state_tracker['SQUAT_COUNT'] = 0
    self.state_tracker['IMPROPER_SQUAT'] = 0
    display_inactivity = True

else:
    self.state_tracker['start_inactive_time'] =
time.perf_counter()
    self.state_tracker['INACTIVE_TIME'] = 0.0

# Menampilkan notifikasi jika counter di-reset karena
inaktivitas

if display_inactivity:
    play_sound = 'reset_counters'
    self.state_tracker['start_inactive_time'] =
time.perf_counter()
    self.state_tracker['INACTIVE_TIME'] = 0.0

```

4.3.7 User Interface Streamlit

Pada sistem ini, antarmuka pengguna dibangun menggunakan Streamlit untuk memudahkan interaksi dan visualisasi hasil analisis squat. Salah satu fitur utama adalah halaman *live analysis* dan upload video, di mana pengguna dapat mengunggah video latihan squat untuk dianalisis secara otomatis.

Tampilan Home:

```

import streamlit as st

st.set_page_config(page_title="Fitness Vision", page_icon="🏋️",
layout="centered")

st.title('Fitness Vision: Analyze Your Squat Technique')
st.markdown("---")
st.markdown("""
### Overview
Fitness Vision adalah aplikasi AI yang membantu Anda
menganalisis teknik squat secara real-time maupun dari video
rekaman. Dengan teknologi pose estimation (MediaPipe), aplikasi
ini memberikan feedback langsung mengenai postur squat Anda dan
mengevaluasi akurasi deteksi pose menggunakan metrik MPJPE (Mean
Per Joint Position Error).

**Fitur utama:**
- Analisis squat secara real-time melalui webcam
- Upload video untuk analisis postur squat
- Evaluasi akurasi pose estimation dengan MPJPE
- Mode Beginner & Pro sesuai kebutuhan
- Visualisasi perbedaan prediksi dan ground truth
""")

```

```
st.markdown("""  
### Cara Menggunakan  
1. Pilih menu **Live Stream** untuk analisis squat secara langsung menggunakan webcam.  
2. Pilih menu **Upload Video** untuk menganalisis video squat yang sudah direkam.  
3. Aktifkan fitur MPJPE untuk melihat evaluasi akurasi pose estimation.  
4. Kunjungi halaman **MPJPE Analysis** untuk memahami lebih lanjut tentang metrik evaluasi dan interpretasinya.  
""")  
  
st.markdown("""  
## Quick Navigation  
""")  
col1, col2, col3 = st.columns(3)  
with col1:  
    if st.button('📷 Live Stream'):  
        st.switch_page('pages/1_📷_Live_Stream.py')  
with col2:  
    if st.button('📁 Upload Video'):  
        st.switch_page('pages/2_📁_Upload_Video.py')  
with col3:  
    if st.button('📊 MPJPE Analysis'):  
        st.switch_page('pages/3_📊_MPJPE_Analysis.py')  
  
st.markdown("""  
---  
  
<sub>Fitness Vision is developed using Mediapipe and OpenCV.</sub>  
""", unsafe_allow_html=True)
```

The screenshot shows the homepage of the Squat Vision application. At the top right, there are 'Deploy' and three-dot menu icons. The main title is 'Squat Vision: Analyze Your Squat Technique'. Below it is a horizontal line. Under the title, there's a section titled 'Overview' with a brief description: 'Squat Vision adalah aplikasi AI yang membantu Anda menganalisis teknik squat secara real-time maupun dari video rekaman. Dengan teknologi pose estimation (MediaPipe), aplikasi ini memberikan feedback langsung mengenai postur squat Anda dan mengevaluasi akurasi deteksi pose menggunakan metrik MPJPE (Mean Per Joint Position Error).'. A 'Fitur utama:' heading lists features: 'Analisis squat secara real-time melalui webcam', 'Upload video untuk analisis postur squat', 'Evaluasi akurasi pose estimation dengan MPJPE', 'Mode Beginner & Pro sesuai kebutuhan', and 'Visualisasi perbedaan prediksi dan ground truth'. Another section, 'Cara Menggunakan', provides instructions: 1. Pilih menu Live Stream untuk analisis squat secara langsung menggunakan webcam. 2. Pilih menu Upload Video untuk menganalisis video squat yang sudah direkam. 3. Aktifkan fitur MPJPE untuk melihat evaluasi akurasi pose estimation. 4. Kunjungi halaman MPJPE Analysis untuk memahami lebih lanjut tentang metrik evaluasi dan interpretasinya.

Gambar 4. 6 Tampilan Homepage Navigasi Fitness Vision

Halaman Live Analysis:

```

import streamlit as st
from streamlit_webrtc import VideoHTMLAttributes,
webRTCStreamer
from aiortc.contrib.media import MediaRecorder
from utils import get_mediapipe_pose
from process_frame import ProcessFrame
from thresholds import get_thresholds_beginner,
get_thresholds_pro

st.title('AI Fitness Trainer: Squats Analysis')

col1, col2 = st.columns(2)
with col1:
    mode = st.radio('Select Mode', ['Beginner', 'Pro'],
horizontal=True)
with col2:
    enable_mpjpe = st.checkbox('Enable MPJPE Evaluation',
value=False)

if mode == 'Beginner':
    thresholds = get_thresholds_beginner()
else:
    thresholds = get_thresholds_pro()

live_process_frame = ProcessFrame(thresholds=thresholds,
flip_frame=True,
evaluate_mpjpe=enable_mpjpe)
pose = get_mediapipe_pose()

def video_frame_callback(frame):
    frame = frame.to_ndarray(format="rgb24")

```

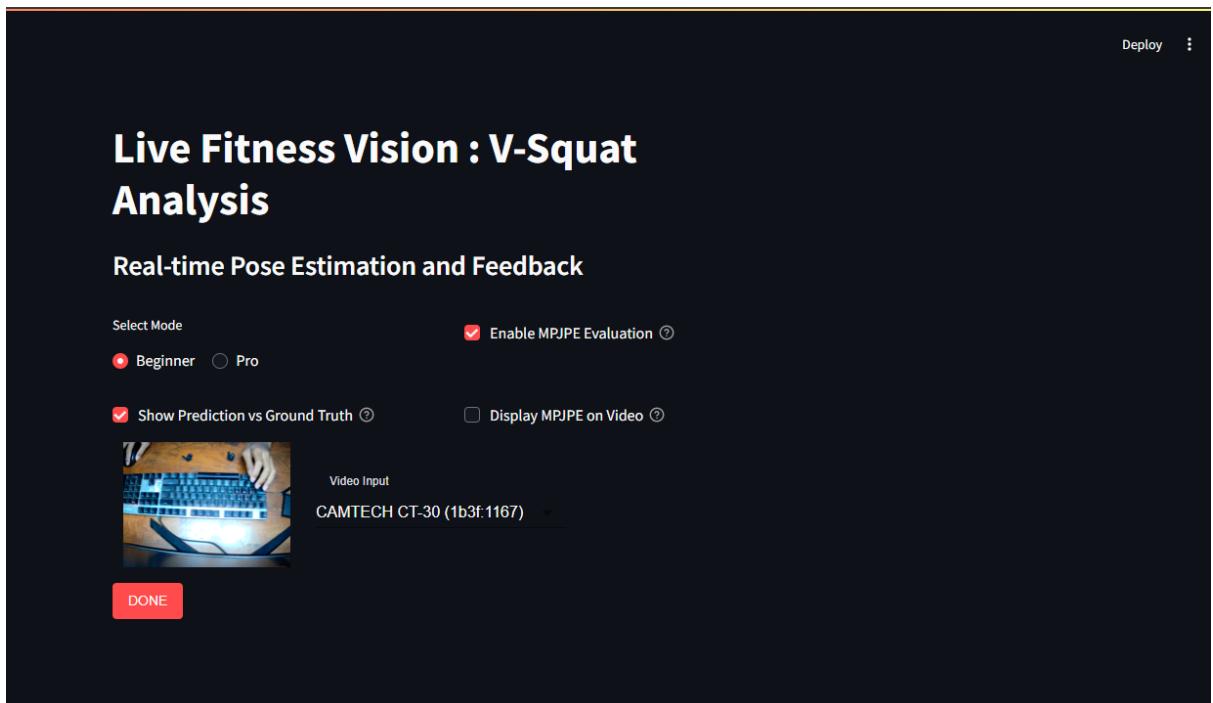
```

frame, _ = live_process_frame.process(frame, pose)
return av.VideoFrame.from_ndarray(frame, format="rgb24")

ctx = webrtc_streamer(
    key="Squats-pose-analysis",
    video_frame_callback=video_frame_callback,
    rtc_configuration={"iceServers": [{"urls":
        ["stun:stun.l.google.com:19302"]}]},
    media_stream_constraints={"video": {"width": {"min":480,
'iideal':480}}, "audio": False},
    video_html_attrs=VideoHTMLAttributes(autoPlay=True,
controls=False, muted=False)
)

# Menampilkan hasil MPJPE jika diaktifkan
if ctx.state.playing and enable_mpjpe:
    st.markdown("### MPJPE Over Time")
    if len(live_process_frame.mpjpe_values) > 0:
        st.line_chart(live_process_frame.mpjpe_values,
use_container_width=True)

```



Gambar 4. 7 Interface Live Detection Page

Halaman Upload Video:

```

import os
import sys
import streamlit as st
import cv2
import tempfile
import numpy as np

BASE_DIR = os.path.abspath(os.path.join(__file__, '../..'))
sys.path.append(BASE_DIR)

from utils import get_mediapipe_pose
from process_frame import ProcessFrame
from thresholds import get_thresholds_beginner,
get_thresholds_pro

st.title('AI Fitness Trainer: Squats Analysis')

col1, col2 = st.columns(2)
with col1:
    mode = st.radio('Select Mode', ['Beginner', 'Pro'],
horizontal=True)
with col2:
    enable_mpjpe = st.checkbox('Enable MPJPE Evaluation',
value=False,
help="Mean Per Joint Position
Error - Evaluates pose estimation accuracy")

# MPJPE visualization options if MPJPE is enabled
if enable_mpjpe:
    col1_mpjpe, col2_mpjpe = st.columns(2)
    with col1_mpjpe:
        show_comparison = st.checkbox('Show Prediction vs Ground
Truth', value=False,
help="Visualize the
difference between predicted and ground truth landmarks")
    with col2_mpjpe:
        display_mpjpe = st.checkbox('Display MPJPE on Video',
value=False,
help="Show MPJPE values
directly on the video frame")
else:
    show_comparison = False
    display_mpjpe = False

thresholds = None

if mode == 'Beginner':
    thresholds = get_thresholds_beginner()

elif mode == 'Pro':
    thresholds = get_thresholds_pro()

```

```

upload_process_frame = ProcessFrame(thresholds=thresholds,
evaluate_mpjpe=enable_mpjpe,
                                         display_mpjpe=display_mpjpe)

# Initialize face mesh solution
pose = get_mediapipe_pose()

download = None

if 'download' not in st.session_state:
    st.session_state['download'] = False

output_video_file = f'output_recorded.mp4'

if os.path.exists(output_video_file):
    os.remove(output_video_file)

with st.form('Upload', clear_on_submit=True):
    up_file = st.file_uploader("Upload a Video", ['mp4', 'mov',
'avi'])
    uploaded = st.form_submit_button("Upload")

stframe = st.empty()

ip_vid_str = '<p style="font-family:Helvetica; font-weight:
bold; font-size: 16px;">Input Video</p>'
warning_str = '<p style="font-family:Helvetica; font-weight:
bold; color: Red; font-size: 17px;">Please Upload a Video
first!!!</p>'

warn = st.empty()

download_button = st.empty()

if up_file and uploaded:

    download_button.empty()
    tfile = tempfile.NamedTemporaryFile(delete=False)

    try:
        warn.empty()
        tfile.write(up_file.read())

        vf = cv2.VideoCapture(tfile.name)

        # ----- Write the processed video
frame. -----
        fps = int(vf.get(cv2.CAP_PROP_FPS))
        width = int(vf.get(cv2.CAP_PROP_FRAME_WIDTH))
        height = int(vf.get(cv2.CAP_PROP_FRAME_HEIGHT))

```

```

        frame_size = (width, height)
        fourcc = cv2.VideoWriter_fourcc(*'mp4v')
        video_output = cv2.VideoWriter(output_video_file,
fourcc, fps, frame_size)
        #
-----



        txt = st.sidebar.markdown(ip_vid_str,
unsafe_allow_html=True)
        ip_video = st.sidebar.video(tfile.name)

        while vf.isOpened():
            ret, frame = vf.read()
            if not ret:
                break

            # convert frame from BGR to RGB before processing
            it.
            frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
            out_frame, _ = upload_process_frame.process(frame,
pose)
            stframe.image(out_frame)
            video_output.write(out_frame[..., ::-1])

        vf.release()
        video_output.release()

        # Show MPJPE statistics if evaluation was enabled
        if enable_mpjpe and upload_process_frame.mpjpe_values:
            st.markdown("### MPJPE Over Time")

            # Create a line chart with improved styling
            st.line_chart(
                upload_process_frame.mpjpe_values,
                use_container_width=True
            )

            # Display metrics below the chart
            col1, col2, col3 = st.columns(3)

            # Calculate statistics
            avg_mpjpe =
np.mean(upload_process_frame.mpjpe_values)
            min_mpjpe = min(upload_process_frame.mpjpe_values)
            if upload_process_frame.mpjpe_values else 0
            max_mpjpe = max(upload_process_frame.mpjpe_values)
            if upload_process_frame.mpjpe_values else 0

            with col1:
                st.metric("Average MPJPE (px)",
f"{avg_mpjpe:.2f}")
            with col2:
                st.metric("Min MPJPE (px)", f"{min_mpjpe:.2f}")
            with col3:
                st.metric("Max MPJPE (px)", f"{max_mpjpe:.2f}")

```

```

# Add styling for metrics to match the desired look
st.markdown("""
<style>
    .stMetric {
        background-color: #262730;
        padding: 15px;
        border-radius: 5px;
        margin-bottom: 20px;
    }
    .stMetric label {
        color: #FAFAFA;
    }
    .stMetric .css-1wivap2 {
        font-size: 42px;
        color: #FFFFFF;
        font-weight: bold;
    }
</style>
""", unsafe_allow_html=True)

stframe.empty()
ip_video.empty()
txt.empty()
tfile.close()

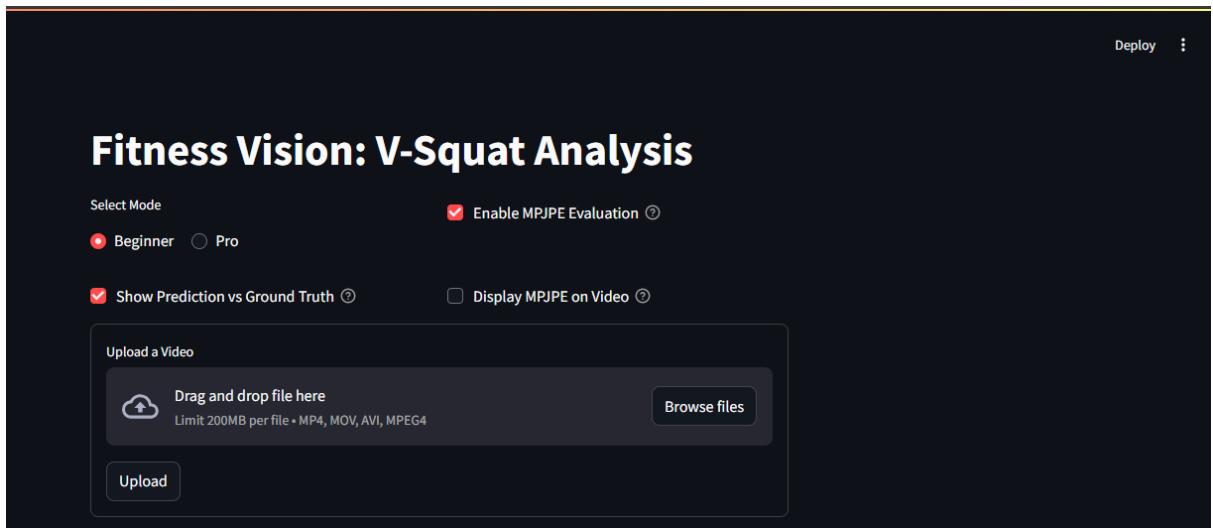
except AttributeError:
    warn.markdown(warning_str, unsafe_allow_html=True)

if os.path.exists(output_video_file):
    with open(output_video_file, 'rb') as op_vid:
        download = download_button.download_button('Download Video', data = op_vid, file_name='output_recorded.mp4')

    if download:
        st.session_state['download'] = True

if os.path.exists(output_video_file) and
st.session_state['download']:
    os.remove(output_video_file)
    st.session_state['download'] = False
    download_button.empty()

```



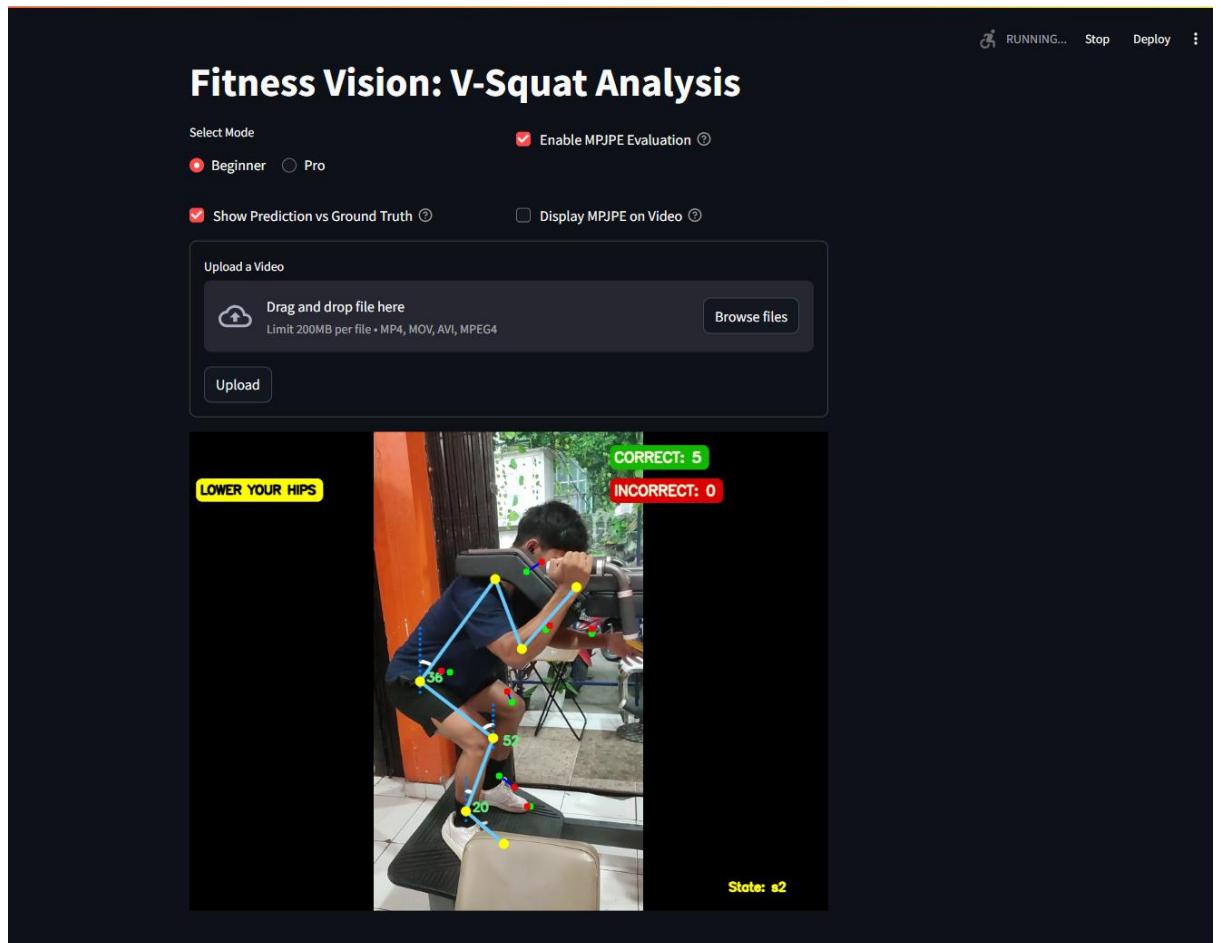
Gambar 4. 8 User Interface Upload Video Fitness Vision

4.4 Testing

Pada tahap ini, dilakukan pengujian menyeluruh terhadap sistem deteksi kesalahan squat yang telah dikembangkan. Pengujian bertujuan untuk memastikan seluruh fitur berjalan dengan baik, memberikan feedback yang akurat, serta mampu beradaptasi pada berbagai kondisi penggunaan.

4.4.1 Pengujian Mode Upload Video (Analisis dari Video Rekaman)

Pengujian ini bertujuan untuk memastikan bahwa sistem mampu menangani video latihan squat yang diunggah oleh pengguna. Setiap frame dalam video akan dianalisis oleh sistem guna mendeteksi posisi tubuh, menghitung total repetisi, serta menampilkan umpan balik visual mengenai teknik pelaksanaan gerakan.



Gambar 4. 9 Preview Pengujian pada menu Upload Video

Pengujian mode upload video menunjukkan bahwa sistem mampu memproses video rekaman dengan efektif, memberikan feedback yang relevan, dan menghasilkan analisis yang akurat. Fitur seperti deteksi pose, perhitungan repetisi, serta umpan balik visual bekerja dengan baik.

4.4.2 Pengujian pada Mode Beginner & Pro

Pengujian dilakukan pada dua mode, yakni Beginner dan Pro, yang masing-masing memiliki *threshold* deteksi berbeda. Mode Beginner dirancang agar lebih toleran terhadap kesalahan postur, sementara mode Pro menetapkan standar yang lebih tin

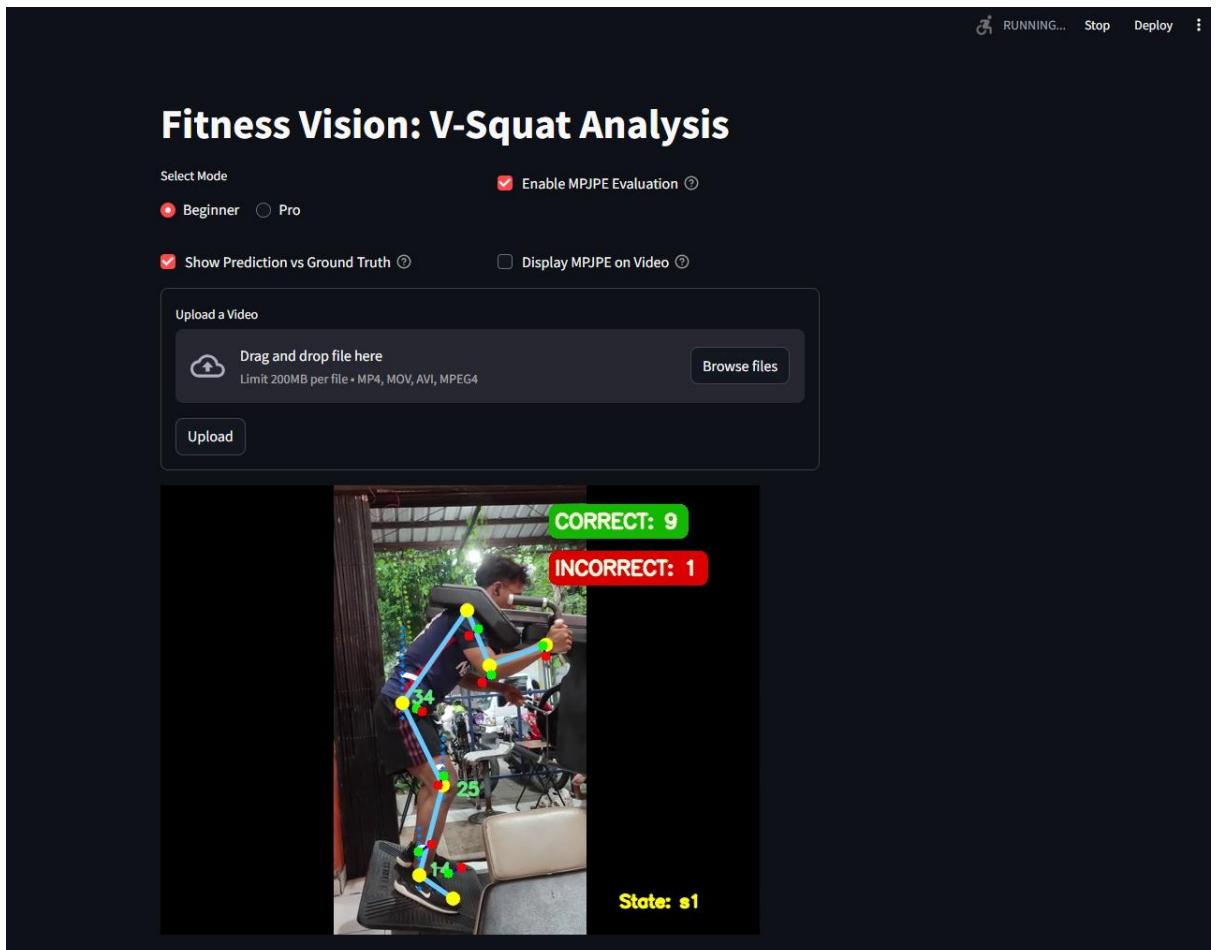
ggi dan ketat. Melalui pengujian ini, dipastikan bahwa sistem mampu menyesuaikan tingkat sensitivitas deteksi serta memberikan umpan balik yang relevan sesuai dengan kemampuan pengguna. Pada tahap ini dilakukan pengujian mode Beginner dan Pro pada satu sample video yang sama.

```
if mode == 'Beginner':
    thresholds = get_thresholds_beginner()
elif mode == 'Pro':
    thresholds = get_thresholds_pro()

upload_process_frame = ProcessFrame(thresholds=thresholds, ...)
```

Kode tersebut menunjukkan pemilihan threshold berdasarkan mode yang dipilih pengguna, sehingga logika deteksi dan feedback akan menyesuaikan.

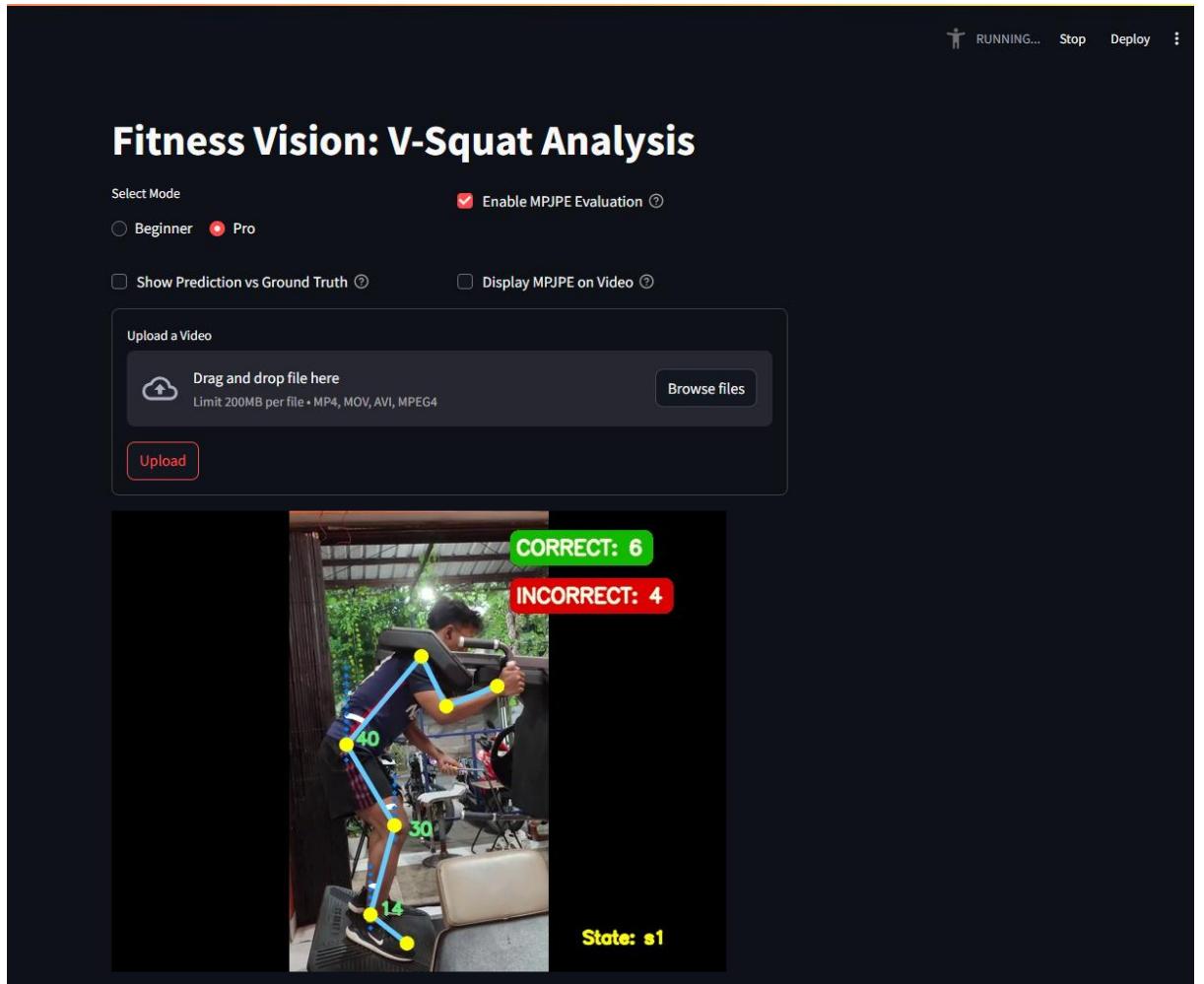
Pengujian Mode Beginner :



Gambar 4. 10 Preview Pengujian pada mode Beginner

Dari pengujian mode beginner tersebut didapatkan hasil jumlah gerakan yang benar sejumlah sembilan, dan gerakan yang salah sejumlah satu.

Pengujian Mode Pro:



Gambar 4. 11 Preview Pengujian pada Mode Pro

Kemudian dari hasil pengujian mode pro terdapat sebuah perbedaan hasil yakni untuk jumlah gerakan benar yang terhitung sejumlah enam, dan gerakan salah yang terdeteksi sejumlah empat. Hal tersebut menandakan befungsinya perbedaan penilaian gerakan antara kedua mode tersebut.

4.4.3 Skenario Pengujian pada Berbagai Kondisi

Pengujian dilakukan pada berbagai kondisi lingkungan dan variasi teknik squat untuk memastikan sistem tetap adaptif dan akurat. Berikut rincian variabel yang diuji meliputi pencahayaan, sudut kamera, postur subjek, serta teknik squat benar dan salah

Tabel 4. 1 Tabel Skenario untuk Pengujian Blackbox Testing

No.	Skenario	Feedback Gerakan	Jumlah Pengujian	Ekspektasi
1.	Pendeteksian dengan kondisi cahaya terang dan pakaian tidak kontras dengan background	“Badan Terlalu ke Belakang”	2 Repetisi	Sistem memberikan feedback sesuai kondisi tersebut jika sudut hip-vertical > 45°
		“Badan Terlalu ke Depan”	2 Repetisi	Sistem memberikan feedback sesuai kondisi tersebut jika sudut hip-vertical < 20°
		“Lutut Melewati Jari Kaki”	5 Repetisi	Sistem memberikan feedback sesuai kondisi tersebut jika sudut knee-ankle line > 30°
		“Squat Terlalu Dalam”	5 Repetisi	Sistem memberikan feedback sesuai kondisi tersebut jika sudut hip-knee line > 95°
2.	Pendeteksian dengan kondisi cahaya gelap dan pakaian kontras dengan background	“Badan Terlalu ke Belakang”	2 Repetisi	Sistem memberikan feedback sesuai kondisi tersebut jika sudut hip-vertical > 45°
		“Badan Terlalu ke Depan”	2 Repetisi	Sistem memberikan feedback sesuai kondisi tersebut jika sudut hip-vertical < 20°

		“Lutut Melewati Jari Kaki”	5 Repetisi	Sistem memberikan feedback sesuai kondisi tersebut jika sudut knee-ankle line $> 30^\circ$
		“Squat Terlalu Dalam”	5 Repetisi	Sistem memberikan feedback sesuai kondisi tersebut jika sudut hip-knee line $> 95^\circ$

Dengan skenario pengujian ini, diharapkan sistem mampu memberikan feedback yang tepat dan konsisten terhadap empat jenis kesalahan teknik squat pada kedua mode skenario (*Beginner* dan *Pro*). Hasil dari pengujian ini akan menjadi dasar evaluasi akurasi sistem dalam mendekripsi dan mengklasifikasikan setiap jenis kesalahan gerakan.

Tabel 4. 2 Tabel Skenario Pengujian pada Instruktur Ahli

Skenario	Subjek Pengujian	Feedback Gerakan	Jumlah Pengujian	Ekspektasi
Terang	Laki-laki usia 26 tahun	Perfect Squat	10 Repetisi	Dapat menghitung gerakan Correct dan Incorrect dengan baik
	Perempuan usia 24 tahun			
Gelap	Laki-laki usia 23 tahun	Perfect Squat	10 Repetisi	Masih dapat menghitung gerakan Correct dan Incorrect dengan baik
	Perempuan usia 45 tahun			

Berdasarkan tabel skenario pengujian di atas, akan dilakukan pengujian sebanyak 40 kali yang terbagi menjadi 20 kali pengujian untuk masing-masing kondisi pencahayaan terang dan gelap. Pengujian ini melibatkan dua subjek dengan karakteristik usia yang berbeda, yaitu laki-laki usia 25 - 50 tahun dan Perempuan usia 45 tahun, dimana masing-masing akan

melakukan 10 repetisi gerakan squat menggunakan *V-Squat Machine*. Pengujian ini bertujuan untuk memvalidasi kehandalan sistem dalam mendeteksi dan memberikan feedback yang akurat pada berbagai kondisi pencahayaan dan karakteristik pengguna yang berbeda.

4.5 Evaluasi

Pada tahap evaluasi, sistem mendeteksi kesalahan pada gerakan squat diuji untuk menilai tingkat akurasi serta konsistensi dalam memberikan *feedback* kepada pengguna. Pengujian dilakukan dengan metode *blackbox testing*, di mana penilaian sistem didasarkan pada respons yang dihasilkan terhadap berbagai skenario gerakan squat yang benar maupun salah. Setiap hasil pengujian dicatat sebagai sesuai atau tidak sesuai dengan harapan, dan tingkat akurasi sistem dihitung berdasarkan persentase keberhasilan dari seluruh skenario yang diuji.

Selain itu, hasil video gerakan juga dianalisis menggunakan metrik Mean Per Joint Position Error (MPJPE) guna menilai ketepatan deteksi pose tubuh. Metrik ini digunakan untuk mengevaluasi seberapa akurat sistem mendeteksi posisi landmark tubuh seperti bahu, pinggul, lutut, dan pergelangan kaki dibandingkan dengan posisi sebenarnya.

BAB V

HASIL DAN PEMBAHASAN

5.1 Hasil Uji Coba Sistem

Pengujian fungsionalitas sistem bertujuan untuk mengevaluasi kemampuan sistem dalam mendeteksi gerakan squat, menghitung jumlah repetisi, serta mengidentifikasi apakah formulasi gerakan sudah sesuai atau terdapat kesalahan. Pengujian dilakukan berdasarkan skenario yang sudah dilakukan pada tabel 4.3, yaitu pada kondisi pencahayaan terang dan gelap. Berikut adalah ringkasan hasil uji coba:

Tabel 5. 1 Hasil Uji Coba Fungsional – Kondisi Terang

No.	Subjek Pengujian	Total Gerakan	Form Correct	Form Incorrect
1.	Instruktur laki-laki usia 26 tahun (penguji 1) ucup	10 Repetisi	10	-
2.	Instruktur laki-laki usia 24 tahun (penguji 2) ican	10 Repetisi	10	-
3.	Instruktur laki-laki usia 49 tahun (penguji 3) om rudi	10 Repetisi	10	-
4.	Non-Instruktur laki-laki usia 19 tahun (penguji 4) bang ori	10 Repetisi	-	10

Berdasarkan hasil Tabel 5.1, dapat dilihat bahwa tiga penguji yang merupakan instruktur berpengalaman (dengan rentang usia 24–49 tahun) mampu melakukan seluruh repetisi squat dengan teknik yang benar. Sistem berhasil mendeteksi semua gerakan mereka sebagai “*Form Correct*”, tanpa ada satu pun kesalahan teridentifikasi. Hal ini menunjukkan bahwa pada kondisi pencahayaan terang, sistem memiliki tingkat akurasi yang sangat tinggi dalam mengenali dan mengklasifikasikan teknik squat yang sesuai standar.

Penambahan penguji keempat, yaitu non-instruktur laki-laki berusia 19 tahun yang baru mulai latihan gym, memberikan gambaran lebih luas terkait kemampuan sistem dalam mendeteksi kesalahan pada pengguna pemula. Pada subjek ini, seluruh 10 repetisi squat dideteksi sebagai “*Form Incorrect*”, kemudian sistem secara konsisten memberikan *feedback*

“Badan terlalu ke Belakang”. Hasil ini membuktikan bahwa sistem tidak hanya mampu mengidentifikasi gerakan yang benar pada instruktur, tetapi juga efektif dalam mendeteksi dan memberikan feedback gerakan pada pemula yang melakukan kesalahan teknik.

Tabel 5. 2 Hasil Uji Coba Fungsional – Kondisi Gelap

No.	Subjek Pengujian	Total Gerakan	Form Correct	Form Incorrect	Keterangan
1.	Non-Instruktur laki-laki usia 21 tahun (penguji 5) naylor	10 Repetisi	3	7	Pada saat pengujian titik landmark <i>knee</i> dan <i>ankle</i> sering kali bergeser yang mengakibatkan perhitungan untuk kedalaman squat tidak optimal seperti yang dilakukan pada pengujian tersebut, serta sering kali terdapat error “ <i>Camera not aligne properly</i> ” karena kondisi cahaya yang minim
2.	Instruktor laki-laki usia 24 tahun (penguji 2) ican	10 Repetisi	2	8	Hasil dari pengujian tersebut sering kali skeleton landmark tiba2 hilang dan muncul yang mengakibatkan perhitungannya menjadi tidak

					optimal, serta menampilkan feedback kesalahan yang seharusnya bukan dihitung sebagai kesalahan.
3.	Non-Instruktur laki-laki usia 21 tahun (penguji 6) deden	11 Repetisi	8	3	Hasil pengujiannya lebih optimal karena memang dari segi pencahayaan nya tidak segelap dari pengujian sebelumnya, namun beberapa feedback kesalahan tidak tertampilkan secara realtime.
4.	Non-Instruktur laki-laki usia 23 tahun (penguji 7) alden	10 Repetisi			

Berdasarkan hasil pada Tabel 5.2, terlihat bahwa kondisi pencahayaan yang minim secara signifikan memengaruhi akurasi sistem dalam mendekripsi dan mengklasifikasikan gerakan squat. Pada dua pengujian dengan cahaya sangat redup, titik landmark pada knee dan ankle sering bergeser, bahkan skeleton landmark terkadang hilang-muncul secara tiba-tiba. Hal ini menyebabkan sistem gagal menghitung kedalaman squat secara optimal, serta sering memunculkan error “*Camera not aligned properly*”. Akibatnya, feedback yang diberikan pun tidak selalu sesuai dengan kesalahan sebenarnya, bahkan beberapa gerakan yang benar juga terdeteksi sebagai salah.

Namun, pada pengujian dengan pencahayaan yang sedikit lebih baik, sistem menunjukkan performa yang lebih optimal dengan jumlah deteksi benar yang lebih tinggi dan feedback kesalahan yang lebih relevan, meskipun masih terdapat keterlambatan dalam menampilkan *feedback* secara *realtime*. Hal ini menandakan pentingnya pencahayaan yang memadai agar akurasi deteksi postur dan klasifikasi kesalahan dapat berjalan dengan optimal.

5.2 Pengujian *Blackbox*

Kondisi Cahaya	Feedback Gerakan	Jumlah Pengujian	Hasil	Keterangan
Terang	“Badan Terlalu ke Belakang”	3 Repetisi	Valid	Sistem berhasil mendeteksi ketika badan pengguna terlalu condong ke belakang (sudut hip-vertical $> 50^\circ$). Dalam tiga repetisi yang diuji, sistem memberikan feedback realtime sesuai dengan kondisi kesalahan gerakan
	“Badan Terlalu ke Depan”	3 Repetisi	Valid	Sistem mampu mengenali kondisi di mana badan pengguna terlalu condong ke depan (sudut hip-vertical $< 10^\circ$). Pada tiga repetisi pengujian, feedback yang diberikan sesuai dengan ekspektasi
	“Lutut Melewati Jari Kaki”	5 Repetisi	Valid	Sistem berhasil mendeteksi kasus di mana lutut pengguna melewati jari kaki (sudut knee-ankle line $> 45^\circ$)

				dalam semua repetisi yang diuji.
	“Squat Terlalu Dalam”	5 Repetisi	Valid	Sistem dengan baik mendeteksi situasi di mana squat dilakukan terlalu dalam (sudut hip-knee line $> 105^\circ$)

Kondisi Cahaya	Feedback Gerakan	Jumlah Pengujian	Hasil	Keterangan
Gelap	“Badan Terlalu ke Belakang”	3 Repetisi	Tidak Valid	Sistem gagal memberikan <i>feedback</i> secara konsisten karena sering terjadi gangguan pada deteksi landmark akibat pencahayaan minim.
	“Badan Terlalu ke Depan”	3 Repetisi	Tidak Valid	Deteksi pose tidak optimal, skeleton landmark sering hilang, sehingga <i>feedback</i> sering tidak muncul atau tidak sesuai kondisi gerakan sebenarnya.
	“Lutut Melewati Jari Kaki”	3 Repetisi	Tidak Valid	Sistem tidak dapat mengenali

				secara tepat posisi lutut terhadap jari kaki karena titik landmark sering bergeser atau hilang akibat pencahayaan rendah.
“Squat Terlalu Dalam”	3 Repetisi	Tidak Valid	Feedback tidak muncul secara realtime karena deteksi sudut menjadi tidak akurat, sehingga sistem gagal memberikan <i>feedback</i> yang sesuai pada setiap percobaan.	

Hasil pengujian pada kondisi cahaya terang menunjukkan tingkat keberhasilan 100% valid dalam mendeteksi kesalahan gerakan pengguna. Hal tersebut menandakan bahwa dalam kondisi pencahayaan yang ideal, model mediapipe yang digunakan mampu menentukan posisi sendi titik landmark secara akurat. Dengan demikian, perhitungan sudut deviasi gerakan dapat dilakukan dengan sangat presisi, sehingga sistem berhasil memberikan feedback yang tepat dan sesuai dengan kesalahan gerakan yang disimulasikan. Hasil ini membuktikan bahwa mekanisme logika dan nilai *threshold* untuk setiap *feedback* telah dirancang dan diterapkan dengan efektif.

Namun dalam pengujian yang dilakukan pada kondisi pencahayaan minim (gelap), seluruh skenario feedback menghasilkan status "Tidak Valid". Hal ini menunjukkan bahwa

pencahayaan yang buruk secara signifikan menurunkan proses analisis yang dilakukan sistem, yakni memengaruhi kemampuan model mediapipe dalam mengenali titik landmark tubuh secara signifikan. Akibatnya, sistem tidak mampu menghitung sudut deviasi gerakan dengan presisi dan akurat. Bahkan ketika pengguna benar-benar melakukan kesalahan gerakan, sistem tidak mampu mendeteksi kesalahan tersebut karena nilai deviasi yang dihitung tidak mencapai *threshold* yang telah ditentukan. Hasil ini sejalan dengan batasan masalah penelitian, yang mengasumsikan bahwa sistem dirancang untuk beroperasi pada kondisi pencahayaan yang ideal.

5.3 Evaluasi dengan MPJPE

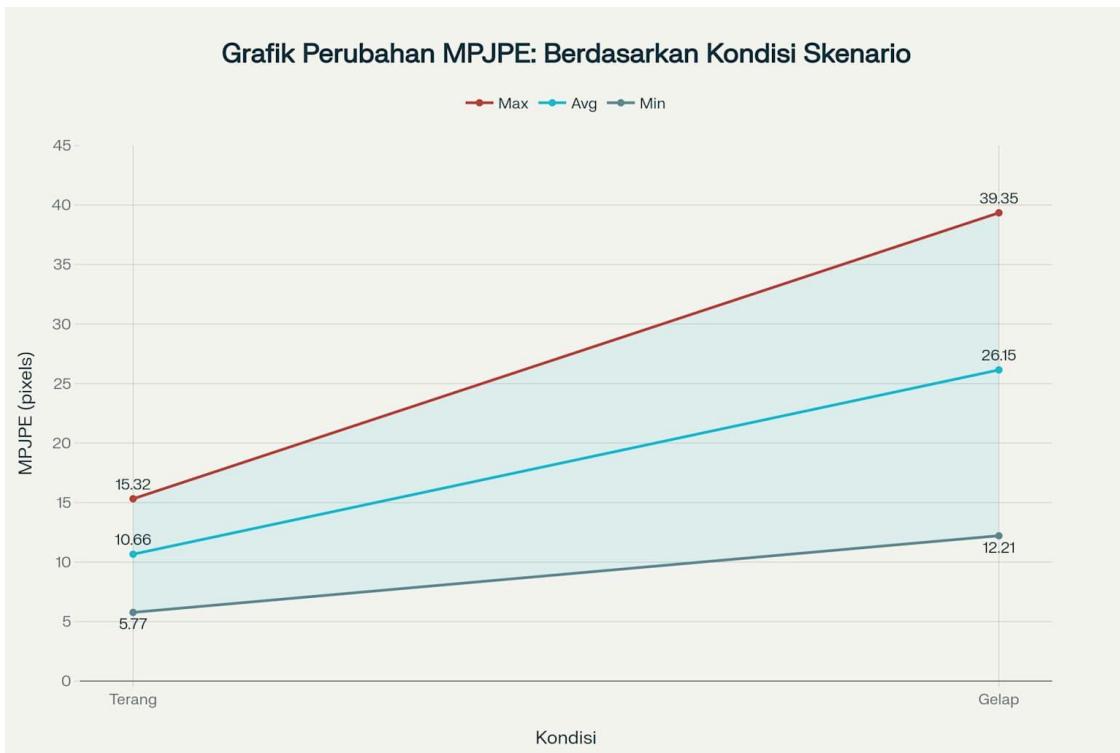
Pada tahap evaluasi dengan metrik *Mean Per Joint Position Error* (MPJPE), sistem diuji menggunakan empat subjek pengujian dengan kondisi yang berbeda, yaitu dua subjek dalam kondisi cahaya terang dan dua subjek dalam kondisi cahaya gelap. Berikut adalah hasil evaluasi MPJPE untuk masing-masing subjek:

Skenario Pengujian	ID Penguji	Deskripsi Penguji	Hasil Repetisi (Benar/Salah)	MPJPE Min	MPJPE Max	MPJPE Rata-rata
Terang	Penguji 1	Instruktur laki-laki usia 26 tahun (penguji 1) ucup	Benar: 10 Salah: 0	5.76 px	15.40 px	10.83 px
	Penguji 2	Instruktur laki-laki usia 24 tahun (penguji 2) ican	Benar: 10 Salah: 0	6.29 px	14.50 px	10.31px
	Penguji 3	Instruktur laki-laki usia 49	Benar: 10 Salah: 0	5.69 px	15.32 px	10.79 px

		tahun (penguji 3) om rudi				
	Penguji 4	Non-Instruktor laki-laki usia 19 tahun (penguji 4) bang ori	Benar: 0 Salah: 10	5.34 px	16.08	10.73 px

Skenario Pengujian	ID Penguji	Deskripsi Penguji	Hasil Repetisi (Benar/Salah)	MPJPE Min	MPJPE Max	MPJPE Rata-rata
Gelap	Penguji 5	Non-Instruktor laki-laki usia 21 tahun (penguji 5) naylor	10 Repetisi	10.62 px	29.73 px	19.69 px
	Pengujii 2	Instruktor laki-laki usia 24 tahun (penguji 2) ican	10 Repetisi	16.78 px	43.18 px	29.44 px
	Penguji 6	Non-Instruktor laki-laki usia 24 tahun	10 Repetisi	9.24 px	45.15 px	29.31 px

	(penguji 6) deden				
Penguji 7	Non-Instruktur laki-laki usia 23 tahun (penguji 7) alden	10 Repetisi			



Hasil evaluasi pada kondisi pencahayaan terang menunjukkan konsistensi performa yang sangat baik dengan nilai MPJPE rata-rata berkisar antara 10.31-10.83 piksel. Keempat subjek pengujian, yang terdiri dari tiga instruktor (P1, P2, P3) dan satu non-instruktor (P4), menunjukkan nilai MPJPE yang berada dalam kategori Good Performance (11-25 piksel), bahkan mendekati batas Excellent Performance (≤ 10 piksel).

Variasi nilai MPJPE pada kondisi terang tergolong sangat rendah dengan standar deviasi hanya 0.24 piksel, yang menandakan kestabilan dalam mendeteksi landmark tubuh.

Nilai MPJPE minimum berkisar 5.34-6.29 piksel, sementara nilai maksimum berada pada rentang 14.50-16.08 piksel. Hasil ini menegaskan bahwa MediaPipe mampu mempertahankan akurasi optimal dalam mendeteksi posisi titik-titik sendi selama pelaksanaan squat pada kondisi pencahayaan yang optimal.

Namun pada kondisi pencahayaan gelap menunjukkan degradasi performa yang dramatis dengan nilai MPJPE rata-rata meningkat menjadi 19.69-29.44 piksel. Dari tiga subjek yang diuji (P5, P2, P6), hanya satu subjek (P5) yang masuk kategori Good Performance dengan MPJPE 19.69 piksel, sementara dua subjek lainnya (P2 dan P6) masuk kategori Poor Performance dengan MPJPE masing-masing 29.44 dan 29.31 piksel.

Pada kondisi ini, variasi nilai MPJPE juga naik drastis, yang terlihat dari standar deviasi sebesar 5.59 piksel dan menunjukkan sistem menjadi jauh lebih tidak stabil. Selain itu, rentang nilai MPJPE menjadi sangat lebar, mulai dari nilai terendah 9.24 piksel hingga yang tertinggi mencapai 45.15 piksel pada subjek P6. Nilai yang sangat bervariasi ini menjadi penyebab utama skeleton landmark sering tidak terdeteksi dengan baik dan posisinya bisa bergeser-geser apabila pencahayaannya gelap.

BAB VI

PENUTUP

6.1 Kesimpulan

Penelitian ini berhasil mengembangkan sistem deteksi kesalahan gerakan squat berbasis computer vision menggunakan MediaPipe dan OpenCV yang diimplementasikan dalam aplikasi web Streamlit. Sistem mampu mendeteksi empat jenis kesalahan utama dalam gerakan squat pada V-Squat Machine, yaitu "Badan Terlalu ke Belakang", "Badan Terlalu ke Depan", "Lutut Melewati Jari Kaki", dan "Squat Terlalu Dalam" dengan tingkat akurasi 100% pada kondisi pencahayaan yang optimal. Evaluasi menggunakan metrik *Mean Per Joint Position Error* (MPJPE) menunjukkan performa yang sangat baik dengan nilai rata-rata 10.31-10.83 piksel pada kondisi terang, yang masuk dalam kategori *Good Performance* dan mendekati *Excellent Performance*.

Sistem yang dikembangkan juga memiliki kemampuan memberikan *feedback real-time* kepada pengguna dan menghitung jumlah repetisi gerakan yang benar dan salah secara otomatis. Pengujian pada berbagai subjek menunjukkan bahwa sistem dapat membedakan dengan baik antara teknik yang benar (pada instruktur berpengalaman) dan teknik yang salah (pada pemula), dengan fitur mode Beginner dan Pro yang memiliki tingkat sensitivitas threshold berbeda. Namun, penelitian ini juga mengidentifikasi keterbatasan signifikan pada kondisi pencahayaan yang kurang baik, di mana nilai MPJPE meningkat drastis menjadi 19.69-29.44 piksel dan menyebabkan penurunan akurasi deteksi pose secara keseluruhan.

6.2 Saran

Berdasarkan hasil penelitian yang telah dilakukan, sistem deteksi kesalahan squat masih dapat dioptimalkan melalui beberapa pengembangan. Adapun saran-saran berikut diharapkan dapat menjadi acuan dalam meningkatkan kualitas serta fungsi sistem ke depannya:

1. Integrasi data anatomi pengguna secara detail, seperti tinggi badan dan proporsi tubuh, dapat membantu mempersonalisasi analisis gerakan pada sistem.

2. Diperlukan fitur yang mampu menilai kecepatan dan durasi tiap fase gerakan untuk mendeteksi kesalahan akibat tempo gerak yang terlalu cepat atau penggunaan momentum secara berlebihan.
3. Eksplorasi kemungkinan integrasi dengan sensor tambahan pada alat gym untuk mengurangi ketergantungan pada kondisi pencahayaan yang kurang memadai, serta meningkatkan akurasi deteksi pose dan stabilitas sistem secara keseluruhan.
4. Menggunakan pendekatan hybrid CNN-LSTM untuk meningkatkan akurasi sistem dan mengurangi biaya komputasi di skala *production*. Peneliti mengungkapkan untuk sistem ini memerlukan biaya komputasi yang relatif mahal dengan spesifikasi vCPU dan Ram yang cukup tinggi agar bisa berjalan dengan optimal.
5. Mengembangkan analisis studi kasus pada gerakan latihan beban lain yang fokus pada tubuh bagian atas, seperti Lateral Raises, Pec Deck, Seated Cable Row, Lat Pull Down, dan sebagainya. Setiap jenis latihan memiliki karakteristik biomekanika dan risiko kesalahan teknik yang berbeda, sehingga perlu dilakukan penyesuaian metode deteksi, perhitungan sudut, serta threshold feedback sesuai dengan anatomi dan tujuan latihan.

Dengan mempertimbangkan berbagai saran tersebut, diharapkan penelitian ini dapat menjadi landasan bagi studi-studi selanjutnya dalam mengembangkan sistem deteksi kesalahan gerakan latihan beban yang lebih komprehensif, tidak hanya terbatas pada latihan lower body seperti squat, tetapi juga mencakup analisis otomatis pada berbagai latihan upper body. Pengembangan ke depan diharapkan mampu menghadirkan solusi berbasis teknologi yang semakin presisi, adaptif terhadap karakteristik anatomi pengguna, serta dapat diintegrasikan dengan sensor hardware dan model deep learning untuk mendukung keamanan dan meningkatkan stabilitas sistem secara keseluruhan.

DAFTAR PUSTAKA

- Asuka Ishii & Hiroo Ikeda. (2024). 3D Pose Estimation from Monocular Video with Camera-Bone Angle Regularization on the Image Feature. *IEEE International Conference on Acoustics, Speech, and Signal Processing*. <https://doi.org/10.1109/icassp48485.2024.10446350>
- Bayattork, M., Yaghoubitajani, Z., & Bettany-Saltikov, J. (2024). Do adolescents with different types and degrees of idiopathic scoliosis curves differ in postural control compared to their healthy peers? A cross-sectional study. *BMC Musculoskeletal Disorders*, 25(1), 1071. <https://doi.org/10.1186/s12891-024-08210-6>
- Bhamidipati, V. S. P., Saxena, I., Saisanthyia, D., & Retnadhas, M. (2023). Robust Intelligent Posture Estimation for an AI Gym Trainer using Mediapipe and OpenCV. *2023 International Conference on Networking and Communications (ICNWC)*, 1–7. <https://doi.org/10.1109/ICNWC57852.2023.10127264>
- Bukhary, H. A., Basha, N. A., Dobel, A. A., Alsufyani, R. M., Alotaibi, R. A., & Almadani, S. H. (2023). Prevalence and Pattern of Injuries Across the Weight-Training Sports. *Cureus*. <https://doi.org/10.7759/cureus.49759>
- Calabrese, E., Taverni, G., Easthope, C. A., Skriabine, S., Corradi, F., Longinotti, L., Eng, K., & Delbruck, T. (2019). DHP19: Dynamic Vision Sensor 3D Human Pose Dataset. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 1695–1704. <https://doi.org/10.1109/CVPRW.2019.00217>
- Chaitra V S, Vikas M S, Pankaja B, & Sowmya B. (2024). Posture Assessment Using Pose Detection in Python: A Real-time Approach with MediaPipe and OpenCV. *International Journal For Multidisciplinary Research*. <https://doi.org/10.36948/ijfmr.2024.v06i06.33753>
- Chakole, M., Sontakke, S., Umredkar, L., Rathod, V., Umate, R., & Dorle, S. (2024). Real-Time Full-Body Detection Using Computer Vision: Leveraging OpenCV and MediaPipe. *International Journal of Electrical and Electronics Engineering*, 11(11), 231–240. <https://doi.org/10.14445/23488379/IJEEE-V11I11P123>
- Cong Xie, Xinchuan Guo, & Botao Qie. (2024). Model Based Quadriceps Activation Strategy in Squat Training with Exhaustion Method Under Different Loads. *International Symposium on Computer Science and Intelligent Control*. <https://doi.org/10.1109/iscsic64297.2024.00102>
- Cooke, J. (2025). Protective Mechanisms Deep Squat. *Science in Sports & Exercise*. <https://jcfitness.co.uk/blog/should-you-avoid-full-squatting/>
- Dedhia, U., Bhoir, P., Ranka, P., & Kanani, P. (2023). Pose Estimation and Virtual Gym Assistant Using MediaPipe and Machine Learning. *2023 International Conference on Network, Multimedia and Information Technology (NMITCON)*, 1–7. <https://doi.org/10.1109/NMITCON58196.2023.10275938>
- Dudekula, K. V., Mukkoti, M. V. C., Yellapragada, V. P. K., Kasaraneni, P. P., Challa, P. R., Gangishetty, D., Solanki, M., & Singhu, R. (2024). Physiotherapy Assistance for Patients Using Human Pose Estimation With Raspberry Pi. *ASEAN Journal of Scientific and Technological Reports*, 27(4), e251096. <https://doi.org/10.55164/ajstr.v27i4.251096>

- Escamilla, R. F., Fleisig, G. S., Zheng, N., Lander, J. E., Barrentine, S. W., Andrews, J. R., Bergemann, B. W., & Moorman, C. T. (2002). Effects of technique variations on knee biomechanics during the squat and leg press: *Medicine & Science in Sports & Exercise*, 33(9), 1552–1566. <https://doi.org/10.1097/00005768-200109000-00020>
- Ghazali Sulong & Martin Randles. (2023). Computer Vision Using Pose Estimation. *Wasit Journal of Computer and Mathematics Science*, 2(1), 85–92. <https://doi.org/10.31185/wjcm.111>
- Gongjin Lan, Yu Wu, Fei Hu, & Qi Hao. (2023). Vision-Based Human Pose Estimation via Deep Learning: A Survey. *IEEE Transactions on Human-Machine Systems*, 53(1), 253–268. <https://doi.org/10.1109/thms.2022.3219242>
- Hetaimish, B., Ahmed, H., Otayn, A., Alzahrani, A. M., Almasoudi, E., Elaiw, M., Alzwaihri, A. S., & Samargandi, R. (2024). Prevalence, and types of overuse injuries in gym centers: A cross-sectional study in Saudi Arabia. *Medicine*, 103(28), e38830. <https://doi.org/10.1097/MD.00000000000038830>
- Isha Chaudhary, Nongmeikapam Thoiba Singh, Milind Chaudhary, & Komal Yadav. (2023). Real-Time Yoga Pose Detection Using OpenCV and MediaPipe. *2023 4th International Conference for Emerging Technology (INCET)*. <https://doi.org/10.1109/inct57972.2023.10170485>
- Jingyao Wang & Naigong Yu. (2022). Top-Down Meets Bottom-Up for Multi-Person Pose Estimation. *Chinese Control and Decision Conference*. <https://doi.org/10.1109/ccdc55256.2022.10033584>
- Kathy E. O'Neill, O'Neill, K. E., Stelios G. Pscharakis, & Pscharakis, S. G. (2021). The effect of back squat depth and load on lower body muscle activity in group exercise participants. *Sports Biomechanics*, 1–12. <https://doi.org/10.1080/14763141.2021.1875034>
- Kim, J.-W., Choi, J.-Y., Ha, E.-J., & Choi, J.-H. (2023). Human Pose Estimation Using MediaPipe Pose and Optimization Method Based on a Humanoid Model. *Applied Sciences*, 13(4), 2700. <https://doi.org/10.3390/app13042700>
- Ko, Y.-M., Nasridinov, A., & Park, S.-H. (2024). Real-Time AI Posture Correction for Powerlifting Exercises Using YOLOv5 and MediaPipe. *IEEE Access*, 12, 195830–195853. <https://doi.org/10.1109/ACCESS.2024.3516723>
- Likai Liu. (2024). Application of Communication Effectiveness Theory to Fitness Bloggers' Social Media Communication—A Study of Audience Changes in Health Perceptions and Behaviour. *Journal of Education, Humanities and Social Sciences*. <https://doi.org/10.54097/xy6dw403>
- Lino, F., Santiago, C., & Marques, M. (2025). *Benchmarking 3D Human Pose Estimation Models Under Occlusions* (No. arXiv:2504.10350). arXiv. <https://doi.org/10.48550/arXiv.2504.10350>
- Lisna Wijayanti, Nelly Anggraini, Deden Akbar Izzudin, & Qorry Armen Gemael. (2024). Biomechanical Analysis of Movement in Basic Squat Exercise: Impact on Efficiency and Injury Risk. *COMPETITOR: Jurnal Pendidikan Kepelatihan Olahraga*. <https://doi.org/10.26858/cjpk.v16i3.68734>
- Meet Shah, Kinjal Gandhi, Bhagyesha Manishkumar Pandhi, Priyanka Padhiyar, & Sheshang Degadwala. (2023). Computer Vision & Deep Learning based Realtime and Pre-Recorded Human Pose Estimation. *2023 2nd International Conference on Applied Artificial Intelligence and Computing (ICAAIC)*. <https://doi.org/10.1109/icaaic56838.2023.10141279>

Neil Welch, Welch, N., Kieran Moran, Moran, K., Joseph Antony, Antony, J., Chris Richter, Richter, C., Brendan Marshall, Marshall, B., Joe Coyle, Coyle, J., Éanna Falvey, Falvey, E., Andrew Franklyn-Miller, & Franklyn-Miller, A. (2015). The effects of a free-weight-based resistance training intervention on pain, squat biomechanics and MRI-defined lumbar fat infiltration and functional cross-sectional area in those with chronic low back. *BMJ Open Sport and Exercise Medicine*, 1(1). <https://doi.org/10.1136/bmjsem-2015-000050>

Nicolas Da Silva Pereira, Luiza P. Chaffe, Matheus Iglesias Marques, Rodrigo Freire Guimarães, J. M. Geremia, Marco A. Vaz, B. Baroni, & R. Rodrigues. (2024). Reverse Nordic Curl Does Not Generate Superior Eccentric Activation of the Quadriceps Muscle Than Bodyweight Squat-Based Exercises. *Journal of Sport Rehabilitation*. <https://doi.org/10.1123/jsr.2023-0431>

Progression Models in Resistance Training for Healthy Adults. (2009). *Medicine & Science in Sports & Exercise*, 41(3), 687–708. <https://doi.org/10.1249/MSS.0b013e3181915670>

Radhakrishnan, J., & Vijayaraghavan, A. (n.d.). *Real-Time 3D Human Pose Estimation Using Deep Learning Model for Ergonomics*.

Ramirez, V. J., Bazrgari, B., Gao, F., & Samaan, M. (2022). Low Back Biomechanics during Repetitive Deadlifts: A Narrative Review. *IIE Transactions on Occupational Ergonomics and Human Factors*, 10(1), 34–46. <https://doi.org/10.1080/24725838.2021.2015642>

Rojas-Jaramillo, A., Cuervo-Arango, D. A., Quintero, J. D., Ascuntar-Viteri, J. D., Acosta-Arroyave, N., Ribas-Serna, J., González-Badillo, J. J., & Rodríguez-Rosell, D. (2024). Impact of the deep squat on articular knee joint structures, friend or enemy? A scoping review. *Frontiers in Sports and Active Living*, 6, 1477796. <https://doi.org/10.3389/fspor.2024.1477796>

Ruijie Li. (2023). The comparison of top-down and bottom-up methods in multi-person pose estimation. *Applied and Computational Engineering*, 13(1), 177–182.
<https://doi.org/10.54254/2755-2721/13/20230728>

Saleem, S., Nunes, J., & M, A. (2023). TrainERA1 - Live Gym Tracker using Artificial Intelligence. *SSRN Electronic Journal*. <https://doi.org/10.2139/ssrn.4383182>

Schoenfeld, B. J., Contreras, B., Vigotsky, A. D., & Peterson, M. (2016). *Differential Effects of Heavy versus Moderate Loads on Measures of Strength and Hypertrophy in Resistance-Trained Men*.

Thierry Bouwmans, Bouwmans, T., Sajid Javed, Javed, S., Hongyang Zhang, Zhang, H., Zhouchen Lin, Lin, Z., Ricardo Otazo, & Otazo, R. (2018). On the Applications of Robust PCA in Image and Video Processing. *Proceedings of the IEEE*, 106(8), 1427–1457.
<https://doi.org/10.1109/jproc.2018.2853589>

Venkata Mahesh Babu Batta. (2024). Image Processing using Python. *International Journal of Advanced Research in Science, Communication and Technology*, 575–579.
<https://doi.org/10.48175/IJARSCT-17499>

Wallace, D. A., Salem, G. J., Salinas, R., & Powers, C. M. (2002). Patellofemoral Joint Kinetics While Squatting with and without an External Load. *Journal of Orthopaedic & Sports Physical Therapy*, 32(4), 141–148. <https://doi.org/10.2519/jospt.2002.32.4.141>

Weng, J., Jiang, X., & Chen, Y. (2024). Real-time Squat Pose Assessment and Injury Risk Prediction Based on Enhanced Temporal Convolutional Neural Networks. *International Journal*

of Medical and All Body Health Research, 5(1), 53–62.

<https://doi.org/10.54660/IJMBHR.2024.5.1.53-62>

Westcott, W. L. (2012). *Resistance Training is Medicine: Effects of Strength Training on Health*.

Yavuz, H. U., & Erdag, D. (2017). Kinematic and Electromyographic Activity Changes during Back

Squat with Submaximal and Maximal Loading. *Applied Bionics and Biomechanics*, 2017, 1–8.

<https://doi.org/10.1155/2017/9084725>

LAMPIRAN