SISTEM SECURITY MELALUI FACE RECOGNITION DENGAN METODE EUCLIDEAN DISTANCE

Proposal ini diajukan untuk memenuhi seminar proposal

Disusun oleh :

RIVALDO IMMANUEL

(217064516123)

FAKULTAS TEKNOLOGI KOMUNIKASI DAN INFORMATIKA

PROGRAM STUDI INFORMATIKA

UNIVERSITAS NASIONAL

TAHUN AJARAN

2025/2026

DAFTAR ISI

BAB I

PENDAHULUAN

1.1 Latar Belakang

Sistem keamanan rumah merupakan integrasi dari berbagai perangkat teknologi yang bertujuan untuk melindungi hunian dari akses yang tidak sah serta ancaman keselamatan lainnya. Sistem ini umumnya terdiri atas sejumlah komponen seperti kamera esp 32 cam, sensor, dan layanan pemantauan yang terhubung, yang bekerja secara real time untuk mendeteksi serta memberikan peringatan kepada pemilik rumah terhadap potensi gangguan, penyusupan, atau situasi darurat (Hidayatullah & Putra, 2022)

Selain fungsi perlindungan, sistem ini juga dirancang untuk mendukung efisiensi energi, memberikan kenyamanan, serta meningkatkan kualitas hidup penggunanya. Namun, meskipun terjadi peningkatan signifikan dalam penerapan teknologi keamanan rumah, tren kriminalitas tetap menunjukkan angka yang tinggi, yang mengindikasikan bahwa sistem yang ada masih perlu terus ditingkatkan dari segi efektivitas dan keandalan (Rainer et al., 2021)

Berdasarkan informasi yang dirilis oleh Kepolisian Negara Republik Indonesia, tercatat adanya kenaikan jumlah tindak kriminal sebesar 7,3% pada tahun 2022 jika dibandingkan dengan tahun sebelumnya (CNN Indonesia, 2022). Salah satu bentuk kejahatan yang cukup dominan adalah aksi pencurian di dalam rumah, baik dalam skala kecil maupun besar, yang umumnya terjadi di wilayah perkotaan padat seperti Jakarta. Meningkatnya angka kejahatan ini mendorong masyarakat untuk memperketat sistem pengawasan terhadap lingkungan tempat tinggal mereka. Salah satu upaya strategis yang dapat diterapkan adalah penggunaan sistem keamanan berbasis teknologi, khususnya teknologi pengenalan wajah (face recognition), yang memungkinkan pemantauan hunian secara otomatis dan real-time kapan pun dibutuhkan (O'toole & Castillo, 2025)

Teknologi pengenalan wajah atau face recognition memiliki berbagai jenis dan tingkat kapabilitas, mulai dari sistem yang hanya mendeteksi keberadaan wajah hingga sistem yang mampu mengenali identitas individu secara spesifik. Sistem yang dilengkapi dengan kemampuan identifikasi cenderung menawarkan tingkat keamanan yang lebih tinggi. Dalam prosesnya, teknologi ini bekerja berdasarkan

dua pendekatan utama, yakni pengenalan wajah yang dikenal (recognized) dan tidak dikenal (unrecognized) (Batubara et al., 2024).

Teknologi pengenalan wajah dirancang untuk memungkinkan komputer melakukan proses visualisasi yang menyerupai cara pandang manusia terhadap wajah. Dalam beberapa tahun terakhir, penggunaan teknologi ini semakin meluas, khususnya pada sektor keamanan, penegakan hukum, serta perangkat elektronik konsumen. Penerapannya mencakup berbagai kebutuhan, seperti sistem keamanan bangunan, verifikasi identitas di bandara, membuka kunci perangkat pintar, hingga fitur penandaan otomatis wajah pada foto di media sosial (Zulfikar et al., 2023).

Metode pengenalan wajah tidak hanya terbatas pada identifikasi wajah manusia, tetapi juga memiliki potensi untuk mengenali berbagai objek lainnya. Kendati demikian, proses pengenalan wajah kerap menghadapi sejumlah tantangan, khususnya ketika sistem harus mengidentifikasi wajah dengan karakteristik visual yang sangat khas. Untuk mengatasi tantangan tersebut, algoritma euclidean distance telah menjadi salah satu pendekatan yang cukup efektif dalam merepresentasikan ciri-ciri unik wajah manusia (Priadana & Habibi, 2019)

Pengenalan wajah dilakukan karena Rekognisi wajah dapat digunakan sebagai metode keamanan yang kuat untuk mengidentifikasi individu secara unik. Hal ini berguna dalam mengontrol akses ke area terbatas, perangkat elektronik pribadi, atau data sensitif, sehingga banyak digunakan dalam berbagai aplikasi teknologi modern. Keunggulannya terletak pada prinsip kerjanya yang relatif sederhana namun efisien, serta mudah diintegrasikan dengan teknologi lain seperti Internet of Things (IoT), sehingga membuka peluang bagi pengembangan sistem cerdas yang mampu merespons kondisi lingkungan secara otomatis (Oladimeji et al., 2023)

IoT atau Internet of Things konsep teknologi yang menghubungkan berbagai perangkat fisik seperti peralatan rumah tangga, kendaraan, dan bangunan melalui jaringan internet dengan memanfaatkan sensor, perangkat lunak, dan teknologi pendukung lainnya. Melalui konektivitas ini, perangkat-perangkat tersebut dapat saling berkomunikasi serta mentransfer data ke sistem lain. Penerapan IoT memungkinkan pengelolaan sumber daya dan proses menjadi lebih efisien dan terkontrol. Teknologi ini telah digunakan secara luas, mulai dari penerapan di

rumah pintar (smart home) hingga otomasi industri dan sistem layanan kesehatan. Dengan mengumpulkan dan menganalisis data dari perangkat-perangkat yang terhubung, pengguna dapat memperoleh informasi yang lebih komprehensif untuk menunjang pengambilan keputusan yang lebih tepat dan strategis (Wang & Su, 2025).

Esp 32Cam merupakan sebuah perangkat komputer yang dikembangkan dengan tujuan utama untuk mendukung pembelajaran ilmu komputer secara luas. Meskipun ukurannya ringkas, perangkat ini memiliki kemampuan yang mendekati komputer desktop konvensional karena dilengkapi sistem operasi berbasis Linux, berbagai port input-output, serta dukungan konektivitas yang cukup lengkap. Karakteristik fleksibel dan portabel dari Esp 32 Camera menjadikannya sebagai salah satu pilihan utama dalam berbagai proyek teknologi, mulai dari otomasi sistem rumah hingga pengembangan aplikasi berbasis perangkat lunak dan kecerdasan buatan.

Berdasarkan hasil telaah literatur yang dilakukan, peneliti menemukan adanya kebutuhan akan sistem keamanan rumah yang mampu memberikan peringatan secara langsung kepada pengguna melalui media komunikasi seperti aplikasi Telegram. Menanggapi kebutuhan tersebut, penelitian ini diarahkan untuk merancang dan membangun sistem keamanan rumah berbasis teknologi Internet of Things (IoT) yang dilengkapi dengan fitur pengenalan wajah, sehingga mampu mendeteksi keberadaan individu secara otomatis dan mengirimkan notifikasi secara real-time kepada pemilik rumah (Abordo et al., 2024).

1.2 Tujuan Penelitian

Tujuan dari penelitian ini antara lain:

1. Mengembangkan sistem keamanan berbasis pengenalan wajah yang dapat membedakan antara wajah pemilik rumah dan orang asing (penyusup).
2. Menerapkan metode Euclidean Distance sebagai teknik pengenalan wajah yang efisien dan cepat dalam proses komputasi.
3. Menyediakan solusi alternatif dalam bidang keamanan hunian dengan pemanfaatan teknologi biometrik.

1.3 Manfaat Penelitian

Sistem Security Melalui Face Recognition dengan Metode Euclidean Distance Penelitian mengenai sistem keamanan berbasis face recognition menggunakan metode Euclidean Distance diharapkan memberikan beberapa manfaat, baik secara teoritis maupun praktis berikut jenis manfaatnya:

1.3.1 Bagi Mahasiswa

1. Mampu menerapkan pengetahuan dan keterampilan yang telah diperoleh selama proses pembelajaran di bangku perkuliahan dalam bentuk implementasi nyata.
2. Memiliki kapasitas untuk mengembangkan inovasi yang relevan berdasarkan bekal keilmuan yang telah dikuasai selama masa studi.
3. Meningkatkan kemampuan dalam menyusun karya ilmiah secara sistematis, logis, dan sesuai dengan kaidah penulisan akademik.
4. Menumbuhkan sikap profesional dan tanggung jawab dalam menyelesaikan sebuah permasalahan melalui pendekatan ilmiah dan teknologi.

1.3.2 Bagi Masyarakat

1. Memberikan solusi nyata dalam meningkatkan keamanan rumah tinggal melalui pemanfaatan teknologi pengenalan wajah berbasis IoT yang mudah diakses dan efisien.
2. Meningkatkan kesadaran masyarakat terhadap pentingnya penggunaan teknologi cerdas dalam menjaga keselamatan diri, keluarga, dan lingkungan sekitar.
3. Memberikan alternatif sistem keamanan yang dapat digunakan oleh masyarakat luas, terutama di daerah perkotaan dengan tingkat kerawanan kriminalitas yang tinggi.
4. Mendorong masyarakat untuk lebih melek teknologi dan mampu mengintegrasikan sistem digital ke dalam kehidupan sehari-hari guna menunjang kenyamanan dan perlindungan hunian.

1.3.3 Bagi Yang Punya Rumah

2. Memberikan perlindungan yang lebih optimal terhadap hunian dengan sistem pengawasan yang dapat berjalan secara otomatis dan berkelanjutan.
3. Memudahkan pemilik rumah dalam memantau kondisi keamanan selama 24 jam penuh melalui pemberitahuan (*notifikasi*) yang dikirimkan secara real-time.
4. Mengurangi ketergantungan pada sistem keamanan konvensional, seperti penjaga atau kunci manual, karena sistem bekerja secara mandiri dan cerdas.
5. Memberikan rasa aman dan ketenangan, terutama saat penghuni rumah sedang bepergian atau berada di luar kota dalam waktu yang lama.
6. Menyediakan dokumentasi visual dari aktivitas yang mencurigakan di sekitar rumah, sehingga dapat menjadi bukti pendukung apabila terjadi insiden keamanan.

1.4 Rumusan Masalah

Rumusan masalah yang akan dibahas pada penelitian kali ini adalah:

1. Bagaimana sistem dapat mengelola data wajah untuk proses identifikasi secara efisien dan akurat menggunakan teknologi kamera/webcam?
2. Bagaimana metode Euclidean Distance dapat diterapkan dalam proses identifikasi wajah secara real-time menggunakan kamera sebagai input utama?

1.5 Batasan Masalah

Dalam melakukan penelitian ini diambil batasan masalah, adapun batasan masalah tersebut adalah:

1. Privasi mengikuti persetujuan pengguna; proses pengumpulan dan penghapusan data diatur sesuai kebijakan sistem basis data bersifat lokal dan terenkripsi, dengan jumlah pengguna dibatasi (misalnya ≤ 500) dan setiap pengguna memiliki beberapa gambar enrol
2. Input citra wajah diperoleh melalui kamera (*webcam*) dengan kondisi pencahayaan yang cukup terang dan posisi wajah menghadap ke depan.
3. Sistem hanya mendeteksi dan membedakan dua kategori wajah, yaitu wajah pemilik rumah dan wajah orang asing.

4. Proses pengujian dilakukan dalam ruang terbatas dengan jumlah citra wajah yang tidak terlalu besar (jumlah data terbatas).
5. Implementasi sistem dilakukan pada perangkat komputer dengan spesifikasi standar dan tidak mengintegrasikan perangkat keras tambahan seperti sensor gerak atau alarm otomatis.

BAB II

TINJAUAN PUSTAKA

2.1 Peneliti Terdahulu

Tabel 2.1 Penelitian Terdahulu

NO	Judul	Permasalahan	Solusi	Hasil
1.	Latent Fingerprint Matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (Wu et al., 2021)	Area penelitian: Data sidik jari yang digunakan dalam penelitian ini mencakup 5.000 sampel sidik jari yang diambil dari database polisi dan organisasi forensik - internasional untuk kebutuhan keamanan dan identifikasi identitas. Data ini mencakup sidik jari dengan kualitas berbeda: mulai dari yang sangat jelas hingga yang tidak jelas atau kabur. Setiap sampel sidik jari telah dikategorikan berdasarkan faktor kualitas seperti pencahayaan, kejelasan garis, serta adanya	Penelitian ini menggunakan metode ekstraksi ciri (feature extraction) dengan pendekatan *minutiae-based matching*, yang mencari titik-titik pertemuan dan perpisahan pada pola sidik jari untuk dijadikan fitur unik. Sebelum proses pencocokan, data sidik jari mengalami tahap preprocessing untuk menghilangkan noise dan memperjelas pola minutiae. Selanjutnya, metode pencocokan minutiae ini menggunakan algoritma yang disebut sebagai *Minutiae Matching*	menunjukkan bahwa algoritma pencocokan minutiae memiliki akurasi sebesar 92% pada sidik jari dengan kualitas tinggi dan 80% pada sidik jari kualitas rendah. Metode ini menunjukkan performa terbaik dalam mengidentifikasi sidik jari dengan noise rendah dan pencahayaan stabil. Waktu rata-rata yang dibutuhkan untuk proses pencocokan adalah sekitar 0,5 detik per sidik jari pada perangkat berkecepatan standar,

		faktor gangguan (noise) dari lingkungan. Variabel utama yang diteliti meliputi waktu pencocokan, akurasi deteksi, dan tingkat keberhasilan pada berbagai tingkat kualitas sidik jari.	*Algorithm*, yang menghitung jarak Euclidean antara titik minutiae dari sidik jari latent dan sidik jari yang sudah terdaftar. Algoritma ini kemudian menghitung tingkat kemiripan dengan metode korelasi gambar yang telah dioptimasi untuk kecepatan dan akurasi. Pengujian dilakukan pada berbagai kondisi kualitas gambar untuk mengukur sensitivitas metode terhadap noise dan gangguan lainnya.	sehingga memungkinkan penggunaan di perangkat keamanan skala besar. Algoritma ini berhasil menurunkan false acceptance rate (FAR) menjadi 1,2% dan meningkatkan false rejection rate (FRR) menjadi 0,8%, yang menunjukkan ketahanan algoritma terhadap faktor gangguan di lapangan.
2	Face recognition using Eigenfaces. (Wahyudi et al., 2022)	*Principal Component Analysis (PCA)*, yang mengubah gambar wajah menjadi *eigenfaces* dengan melakukan dekomposisi pada matriks wajah. Proses ini dimulai dengan preprocessing untuk mengurangi noise pada citra dan menyesuaikan resolusi	yang diambil dari berbagai kondisi pencahayaan dan sudut pandang yang berbeda untuk menguji akurasi sistem. Setiap gambar wajah telah diklasifikasikan berdasarkan faktor pencahayaan, ekspresi wajah, dan sudut pandang, untuk	Sistem ini berhasil mencapai akurasi pengenalan wajah sebesar 95% pada kondisi pencahayaan yang konsisten, dan mengalami penurunan akurasi menjadi 78% ketika pencahayaan sangat bervariasi. Metode PCA terbukti memiliki kecepatan

		gambar agar seragam. Setelah preprocessing, PCA diterapkan untuk mengidentifikasi komponen utama yang mewakili karakteristik wajah. Komponen-komponen ini kemudian disusun menjadi vector wajah, atau *eigenfaces*, yang digunakan untuk membandingkan dan mengenali wajah baru berdasarkan tingkat kemiripan dengan data yang ada. PCA digunakan karena efisiensi komputasi yang tinggi dan kemampuannya untuk mereduksi dimensi data tanpa kehilangan informasi penting.	memungkinkan uji coba pada kondisi yang mirip dengan situasi nyata. Sumber data utama berasal dari database Yale Face Database dan ORL Face Database, yang menyediakan sampel wajah dalam berbagai variasi posisi dan pencahayaan..	pemrosesan yang tinggi, dengan waktu komputasi rata-rata sekitar 0,3 detik per gambar wajah, membuatnya cocok untuk aplikasi real-time. Pada kondisi optimal, false acceptance rate (FAR) dan false rejection rate (FRR) masing-masing mencapai 1,5% dan 2,3%, menunjukkan performa yang cukup andal di lingkungan yang terkontrol namun agak sensitif terhadap variasi pencahayaan.
3	Iris Recognition Works (Indriyono et al., 2025)	berbagai kelompok usia dan etnisitas. Data ini diperoleh dari database CASIA Iris dan database UBIRIS, yang mengandung citra iris dari kondisi cahaya berbeda serta adanya	yang digunakan dalam penelitian ini adalah *Daugman's Integro-Differential Operator*, yang berfungsi untuk mendeteksi lingkaran iris dan pupil pada citra mata. Proses dimulai	menunjukkan bahwa metode ini memiliki akurasi pengenalan sebesar 99,6% dengan waktu komputasi rata-rata 0,8 detik per gambar. Ketahanan metode terhadap

		variasi posisi mata. Variabel utama yang diteliti adalah ketepatan dan waktu pengenalan pada berbagai tingkat kualitas citra. Setiap citra iris dilengkapi dengan informasi tentang kondisi cahaya dan posisi mata, yang digunakan untuk analisis lebih lanjut.	dengan preprocessing untuk meningkatkan kontras citra iris dan mengurangi noise. Setelah preprocessing, operator Daugman diterapkan untuk mengidentifikasi batas antara iris dan pupil serta kelopak mata. Ekstraksi fitur dilakukan dengan menerapkan transformasi Fourier pada area iris yang telah diisolasi, menghasilkan *iriscode* unik yang menjadi representasi digital dari pola iris. Proses pencocokan kemudian dilakukan dengan menghitung jarak Hamming antara iriscode baru dan database yang ada.	variasi pencahayaan dan posisi mata memungkinkan sistem bekerja efektif di berbagai kondisi lingkungan. False Acceptance Rate (FAR) berada di bawah 0,1%, dan False Rejection Rate (FRR) sebesar 0,2%, menjadikan metode ini salah satu yang paling akurat dan cepat dalam aplikasi biometrik berbasis iris. Selain itu, keandalan algoritma ini dalam mengatasi faktor gangguan membuatnya sangat cocok untuk aplikasi keamanan tingkat tinggi.
4	Face Recognition with a Convolutional Neural Network Approach	Pembuatan aplikasi absensi wajah yang	Data yang digunakan terdiri dari 1.500 gambar wajah yang dikumpulkan dari AT&T Face Database, dengan variasi sudut pandang, ekspresi, dan	Model CNN ini berhasil mencapai akurasi 97,2% pada data uji dengan pencahayaan konsisten, dan 88% pada kondisi

	(Kamil et al., 2025)		pencahayaan. Data ini diklasifikasikan berdasarkan kategori pencahayaan dan ekspresi untuk memastikan model diuji pada kondisi serupa dunia nyata. Pengujian juga menggunakan subset dari Yale Face Database untuk meningkatkan generalisasi model.	pencahayaan rendah. Waktu komputasi rata-rata per gambar sekitar 0,5 detik. False acceptance rate (FAR) adalah 2%, sedangkan false rejection rate (FRR) adalah 1,3%. Hasil menunjukkan bahwa CNN dapat diandalkan untuk pengenalan wajah dalam kondisi variatif.
5	Noisy Iris Database for Iris Recognition Using Deep Learning (Komputer dan Multimedia et al., 2022)	Algoritma deep learning berbasis CNN digunakan untuk melakukan ekstraksi fitur dan segmentasi area iris. Arsitektur jaringan terdiri dari lapisan konvolusi bertumpuk dan pooling, diikuti oleh lapisan fully-connected. Sebelum tahap pengenalan, preprocessing dilakukan untuk menghilangkan noise menggunakan teknik filter Gaussian. Algoritma ini	Database UBIRIS digunakan, yang terdiri dari 1.200 citra iris yang dikumpulkan di lingkungan dengan pencahayaan tidak terkontrol untuk menguji ketahanan metode terhadap noise dan perubahan posisi mata. Data juga mencakup variasi iris dari berbagai etnis dan usia untuk meningkatkan robustitas model.	deep learning ini mencapai akurasi 96% dalam kondisi pencahayaan tidak terkontrol. Tingkat false acceptance rate (FAR) hanya 0,5%, sedangkan false rejection rate (FRR) mencapai 0,9%. Model ini menunjukkan ketahanan yang baik terhadap noise, dengan waktu komputasi rata-rata 0,7 detik per citra.

		dioptimalkan melalui proses fine-tuning hyperparameter seperti learning rate dan batch size.		
6	Fingerprint Recognition for Digital Payment Systems (Wahyuningrum, n.d. 2023)	pengguna sistem pembayaran digital, yang mencakup data sidik jari dengan kualitas tinggi dan rendah serta variasi jenis perangkat pemindai yang digunakan. Setiap sidik jari diklasifikasikan berdasarkan resolusi pemindai, usia pengguna, dan kondisi lingkungan.	yang digunakan adalah *ridge-based matching*, yang memanfaatkan pola garis-garis pada sidik jari. Ekstraksi fitur dilakukan menggunakan algoritma filter Gabor untuk menangkap orientasi dan frekuensi pola ridge. Setelah ekstraksi, data diproses dengan algoritma *support vector machine (SVM)* untuk klasifikasi. Pengujian dilakukan dengan menggunakan *leave-one-out cross-validation* untuk memastikan akurasi hasil.	Sistem ini berhasil mencapai akurasi 95% dalam kondisi pemindaian optimal, namun akurasi menurun menjadi 82% pada perangkat pemindai dengan resolusi rendah. Waktu komputasi rata-rata adalah 0,6 detik per pemindaian, dengan false acceptance rate (FAR) sebesar 0,7% dan false rejection rate (FRR) sebesar 1,1%.
7	Implementation of Facial Recognition in	Setiap siswa memiliki variasi ekspresi wajah seperti tersenyum dan	Sistem ini menggunakan algoritma *Eigenface* untuk ekstraksi fitur. Eigenface menggunakan	Sistem ini memiliki tingkat akurasi 94% pada pencahayaan alami dan 87% pada

	the Attendance System (Wahyuningrum, n.d. 2022)	berkedip untuk menguji keakuratan sistem.	*principal component analysis (PCA)* untuk mereduksi dimensi gambar wajah dan memisahkan karakteristik unik setiap wajah. Hasil ekstraksi dibandingkan dengan database wajah yang telah ada untuk verifikasi identitas. Pengujian dilakukan pada kondisi pencahayaan berbeda untuk mengukur robustitas metode.	pencahayaan buatan. Waktu rata-rata pengenalan wajah adalah 0,4 detik per gambar. False acceptance rate (FAR) mencapai 1,8%, dan false rejection rate (FRR) sebesar 1,5%.
8	*Pemanfaatan Kamera TOF Smartphone untuk Pencatatan Kehadiran Mahasiswa dengan Metode Face Recognition* *(Ajaib Maggang et al., 2020)*	Meningkatkan kemampuan deteksi keberadaan wajah manusia tidak hanya melalui informasi visual	kamera TOF untuk meningkatkan ketepatan sistem dalam mendeteksi dan mengenali wajah akan melibatkan sejumlah besar fitur (bentuk, warna, local binary patterns, wavelet serta korelasi otomatis) pada wajah	Sistem ini memiliki akurasi pengenalan wajah sebesar 92% dalam kondisi pencahayaan normal dan 85% pada kondisi pencahayaan rendah. Kecepatan pemrosesan rata-rata adalah 0,6 detik per gambar. Sistem ini terbukti efektif untuk meningkatkan keamanan di lingkungan gedung,

				dengan tingkat false acceptance rate (FAR) sebesar 1,3% dan false rejection rate (FRR) sebesar 1,8%.
9	Implementasi Minutiae Matching pada Sistem Pengenalan Sidik Jari (K & Babu, 2021)	orang yang berbeda, dengan berbagai jenis sidik jari seperti whorl, loop, dan arch. Data ini dikumpulkan menggunakan perangkat pemindai sidik jari resolusi tinggi, dengan tiap individu memberikan 3 sampel untuk menguji konsistensi.	Algoritma Minutiae Matching digunakan untuk melakukan pengenalan sidik jari. Teknik ini berfokus pada titik minutiae (ridge endings dan bifurcations) untuk membedakan satu sidik jari dari yang lainnya. Setelah ekstraksi fitur, pola minutiae dibandingkan dengan pola yang ada di database untuk menentukan kecocokan. Pengujian dilakukan dengan metode validasi silang untuk menilai keandalan algoritma.	Algoritma berhasil mencapai akurasi 94% pada data uji dengan false acceptance rate (FAR) sebesar 1% dan false rejection rate (FRR) sebesar 1,5%. Sistem ini dapat diandalkan untuk verifikasi identitas, dengan waktu rata-rata pengenalan sebesar 0,7 detik per pemindaian.
10	Penerapan Sistem Absensi Otomatis dengan	gambar wajah siswa yang diambil dengan variasi sudut pandang dan pencahayaan. Data ini dikumpulkan di	Algoritma *Eigenface* berbasis *Principal Component Analysis (PCA)* digunakan untuk ekstraksi fitur wajah.	Sistem ini mencapai akurasi 90% dalam kondisi pencahayaan optimal dan 82% pada kondisi pencahayaan

	Pengenalan Wajah (Al-Arashi, 2025)	sekolah dengan kamera CCTV yang dipasang di pintu masuk kelas untuk mensimulasikan kondisi absensi otomatis.	PCA bekerja dengan mereduksi dimensi data dan menonjolkan karakteristik unik setiap wajah. Setelah itu, citra wajah yang diekstrak dibandingkan dengan database wajah yang disimpan untuk verifikasi.	rendah. Sistem ini mampu mengurangi waktu absensi manual dengan efektif, dengan tingkat false acceptance rate (FAR) sebesar 2% dan false rejection rate (FRR) sebesar 2,5%.

2.2 Pelaksanaan

Menurut pandangan Wildavsky, proses implementasi dipahami sebagai serangkaian aktivitas yang saling terhubung dan membutuhkan penyesuaian agar dapat berjalan secara harmonis. Setelah tahap perancangan selesai dilakukan, sistem informasi yang baru dikembangkan akan mulai diterapkan dan disesuaikan dengan kondisi operasional yang sudah ada. Tahapan ini menjadi salah satu bagian penting dalam siklus pengembangan sistem, karena berfungsi untuk memastikan bahwa sistem tersebut dapat berjalan secara optimal, mendukung kelancaran aktivitas bagi masyarakat, serta memberikan manfaat nyata bagi para pengguna dan pemangku kepentingan yang terlibat, (Bullock et al., 2021).

2.2 Face Recognition

2.2.1 Konsep Dasar Sistem Face Recognition

Wajah manusia memiliki fungsi yang sangat penting dalam kehidupan sosial, terutama sebagai media dalam menyampaikan identitas serta ekspresi emosional individu. Kemampuan otak manusia dalam mengenali wajah dikenal sangat mengesankan, bahkan ketika terdapat perubahan pada penampilan seperti proses penuaan, perubahan gaya rambut, penggunaan kacamata, atau ekspresi wajah yang berbeda. Hal ini menginspirasi perkembangan teknologi

pengenalan wajah yang dirancang untuk meniru kemampuan tersebut dengan menggunakan pendekatan komputasi, (Muwardi & Adisaputro, 2021)

Teknologi pengenalan wajah merupakan salah satu bentuk biometrik visual yang bekerja dengan cara menganalisis struktur wajah manusia dan mengidentifikasinya berdasarkan fitur-fitur unik yang dimiliki oleh setiap individu. Sistem ini memiliki tingkat akurasi dan efisiensi yang tinggi, serta bersifat non-invasif, sehingga sangat sesuai digunakan dalam website keamanan baik di pagi hari maupungn,

Penerapan teknologi ini sudah banyak ditemukan dalam kehidupan sehari-hari, mulai dari sistem presensi berbasis wajah, pengawasan keamanan di area publik, akses kontrol bangunan, hingga identifikasi pelaku tindak kejahatan. Penggunaan metode ini dinilai lebih praktis dan cepat karena tidak membutuhkan interaksi fisik secara langsung dengan perangkat, berbeda dengan teknologi biometrik lainnya seperti sidik jari (Herdianto Situmorang et al., 2023)

Secara keseluruhan, pengenalan wajah tidak hanya mempercepat proses identifikasi, tetapi juga mampu meningkatkan tingkat keamanan dalam sistem digital modern. Keunggulannya dalam memproses citra wajah secara real-time menjadikannya salah satu teknologi yang berkembang pesat dan banyak diadopsi di berbagai sektor.

2.2.2 Tinjauan Teoritis Teknologi Pengenalan Wajah

Teknologi pengenalan wajah merupakan metode identifikasi yang memanfaatkan karakteristik unik dari wajah manusia. Proses ini melibatkan pencocokan antara citra wajah yang ditangkap oleh kamera dengan data wajah yang telah tersimpan dalam basis data sebelumnya. Salah satu tantangan utama dalam penerapan teknologi ini adalah bagaimana membangun representasi wajah yang konsisten dan akurat, meskipun terdapat perubahan pencahayaan atau ekspresi wajah.

Keakuratan sistem pengenalan wajah juga sangat dipengaruhi oleh posisi dan sudut rotasi wajah terhadap kamera. Dalam kondisi ideal, wajah berada dalam posisi tegak lurus terhadap kamera untuk hasil yang optimal. Namun, teknologi pengenalan wajah masa kini telah berkembang dan mampu

mentoleransi kemiringan atau rotasi wajah dalam batas tertentu. Hal ini memungkinkan pengguna untuk tetap dikenali meskipun wajah tidak berada dalam posisi sempurna.

2.2 ESP- 32 CAM

ESP32-CAM merupakan modul mikrokontroler yang dilengkapi dengan kamera, serta mendukung konektivitas Wi-Fi dan Bluetooth. Modul ini banyak digunakan dalam berbagai aplikasi berbasis Internet of Things (IoT), terutama untuk sistem pengawasan visual, pemantauan jarak jauh, dan otomatisasi berbasis citra.

Modul ini menggunakan chip ESP32 sebagai inti pemrosesan data, yang telah terintegrasi dengan modul kamera OV2640 beresolusi 2 megapiksel. Salah satu keunggulan ESP32-CAM adalah ukurannya yang ringkas dan kemampuannya untuk melakukan pengolahan citra secara lokal sebelum dikirim melalui jaringan nirkabel.

Selain itu, ESP32-CAM juga dilengkapi dengan slot kartu microSD, yang memungkinkan penyimpanan data secara lokal. Fitur ini sangat berguna dalam sistem monitoring yang memerlukan dokumentasi atau penyimpanan hasil pengenalan visual. Dengan konsumsi daya yang relatif rendah dan harga yang terjangkau, modul ini menjadi pilihan populer dalam pengembangan sistem berbasis pengenalan wajah dan proyek-proyek otomatisasi lainnya.

2.1 Esp32-Cam

2.3 Sensor Gerak HC-SR501 PIR Motion Pyroelectric Infrared

Sensor HC-SR501 merupakan jenis sensor PIR (Passive Infrared) yang digunakan untuk mendeteksi pergerakan objek berdasarkan radiasi inframerah yang

dipancarkan oleh tubuh manusia atau makhluk hidup lainnya. Sensor ini bekerja dengan prinsip pendeteksian perubahan tingkat energi inframerah dalam area jangkauan, tanpa memancarkan sinyal aktif seperti gelombang radio atau ultrasonik.

HC-SR501 dilengkapi dengan sensor piroelektrik yang mampu mendeteksi perubahan suhu yang tiba-tiba akibat pergerakan benda hidup. Modul ini memiliki dua potensiometer yang dapat digunakan untuk mengatur sensitivitas dan durasi sinyal keluaran, sehingga dapat disesuaikan dengan kebutuhan aplikasi tertentu.

Sensor ini banyak digunakan dalam sistem keamanan, lampu otomatis, serta berbagai proyek berbasis mikrokontroler seperti Arduino atau ESP32. Keunggulannya terletak pada konsumsi daya yang rendah, harga yang terjangkau, dan kemudahan integrasi dengan berbagai jenis rangkaian elektronik.

Gambar 2.3 Sensor Gerak HC-SR501 PIR Motion Pyroelectric Infrared

perhitungan sistem:

Komponen	Parameter	Rumus	Hasil / Rekomendasi
LED indikator	Vf=2.0V, If=10mA, Vout=3.3V	R=(Vout-Vf)/If	130Ω → pilih 150Ω
Base resistor (relay 80mA)	Ic=80mA, hFE≈100, Vout=3.3V, Vbe≈0.7V	Ib=Ic/hFE; Rb=(Vout-Vbe)/Ib	Ib=0.8mA; Rb≈3.3kΩ
Power idle	I=50µA, V=5V	P=V*I	0.25 mW

Power aktif (modul only)	I=50mA, V=5V	P=V*I	0.25 W

HC-SR501 modul PIR menyebutkan tegangan operasi 4.5–20 V, quiescent <50 µA, output HIGH ~3.3 V

2.3.1 Selenoid Door Lock

Selenoid Door Lock merupakan jenis solenoid yang dirancang secara khusus untuk berfungsi sebagai pengunci pintu berbasis sistem elektronik. Terdapat dua jenis sistem kerja pada solenoid ini, yaitu Normally Close (NC) dan Normally Open (NO). Perbedaan utama antara keduanya terletak pada respons terhadap pemberian tegangan. Pada tipe NC, ketika dialiri tegangan, inti solenoid akan bergerak keluar (terkunci atau menutup). Sebaliknya, tipe NO bekerja dengan cara yang berlawanan, di mana saat diberikan tegangan, pengunci akan terbuka. Untuk dapat beroperasi, solenoid door lock ini membutuhkan suplai daya sebesar 12V DC

Gambar 2.3.1 Selenoid

2.3.2 Servo Motor

Servo motor merupakan aktuator elektrik yang digunakan untuk mengontrol posisi sudut secara presisi. Komponen ini bekerja berdasarkan sinyal PWM (Pulse Width Modulation) yang menentukan arah dan besar sudut putaran poros motor. Umumnya, servo motor terdiri dari motor DC kecil, gearbox, potensiometer sebagai sensor posisi, dan rangkaian kendali.

Servo motor mampu bergerak dalam rentang 0° hingga 180°, dan banyak digunakan dalam sistem kendali seperti robotika, lengan mekanik, kamera otomatis, dan model pesawat. Untuk beroperasi, servo motor memerlukan suplai tegangan (biasanya 5V–6V) serta sinyal kontrol dari mikrokontroler seperti Arduino.

Gambar 2.3.2 Servo Motor

2.4 Kabel Adaptor 12 volt

Kabel bertegangan 12 volt berfungsi sebagai penghantar energi listrik untuk peralatan yang membutuhkan daya rendah hingga sedang. dengan tujuan memastikan distribusi daya yang stabil serta aman sesuai standar kelistrikan, perangkat elektronik rumah tangga, maupun rangkaian panel tenaga surya. Desain kabel ini mempertimbangkan faktor keamanan, efisiensi penyaluran arus, serta

ketahanan terhadap perubahan suhu dan lingkungan sehingga mampu memastikan distribusi daya tetap stabil serta meminimalkan risiko kerusakan pada peralatan yang terhubung

Gambar 2.4 Kabel Adaptor 12 Volt

2.5 LCD 1602A

LCD 1602A adalah modul tampilan berbasis kristal cair yang memiliki dua baris (row) dengan masing-masing mampu menampilkan 16 karakter. Komponen ini umum digunakan pada berbagai proyek elektronika dan mikrokontroler karena kemampuannya memberikan tampilan data yang sederhana namun jelas, seperti menampilkan teks, angka, maupun simbol dasar.

Modul ini dilengkapi dengan driver pengendali bawaan yang memudahkan integrasi dengan papan pengendali seperti Arduino atau mikrokontroler lainnya

melalui antarmuka paralel. Kelebihan LCD 1602A terletak pada konsumsi daya yang rendah, kejelasan tampilan meskipun di kondisi pencahayaan yang beragam, serta ketersediaan pin konfigurasi yang fleksibel untuk menyesuaikan kebutuhan pengguna.

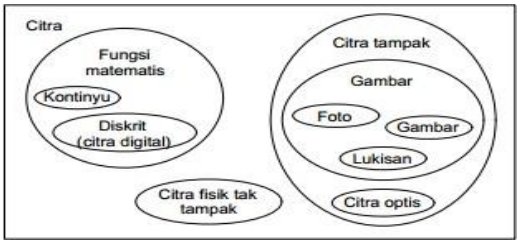
Gambar 2.5 LCD 1602A

2.6 Citra

2.6.1 Definisi Citra

istilah citra diartikan sebagai bentuk representasi atau kemiripan dari suatu objek yang dapat diamati. Dalam konteks umum, citra dapat dipahami sebagai hasil visual dua dimensi yang menampilkan bentuk atau bayangan dari suatu benda. Dari sudut pandang matematis, citra digambarkan sebagai fungsi kontinu yang merepresentasikan distribusi intensitas cahaya pada bidang dua dimensi.

Proses terbentuknya citra berawal dari sumber cahaya yang mengenai suatu objek. Sebagian cahaya dipantulkan kembali oleh permukaan objek dan diterima oleh perangkat seperti face manusia, kamera digital, atau alat pemindai (scanner). Pantulan tersebut kemudian menghasilkan bayangan atau pola visual yang dikenal sebagai citra. Berdasarkan sifat penangkapannya, citra dapat dibedakan menjadi dua jenis, yaitu citra tampak yang dapat ditangkap oleh manusia, dan citra tak tampak yang hanya dapat diamati menggunakan sensor atau perangkat khusus.



Gambar 2.6.1 Jenis citra

Citra tampak sering dijumpai dalam kehidupan sehari-hari, misalnya pada foto keluarga, tampilan gambar di layar komputer maupun televisi, serta hologram yang dihasilkan melalui proses optik. Jenis citra tersebut dapat diamati secara langsung oleh indera penglihatan manusia karena terbentuk dari pantulan cahaya pada permukaan objek.

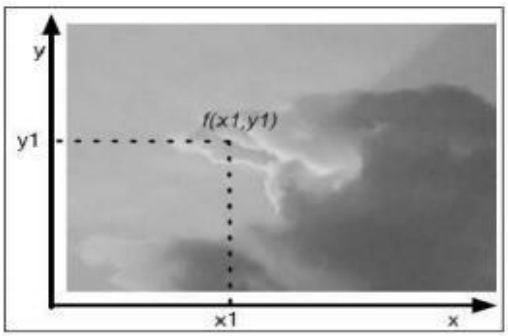
Berbeda dengan citra tampak, citra tak tampak menyimpan informasi visual yang tidak bisa dilihat secara langsung. Contohnya meliputi data gambar dalam bentuk berkas digital serta representasi citra dalam bentuk fungsi matematis. Selain itu, terdapat pula citra fisik tak tampak seperti distribusi panas pada kulit manusia atau peta densitas suatu material yang hanya dapat diamati menggunakan alat bantu sensor khusus. Agar dapat divisualisasikan oleh mata manusia, citra tak tampak perlu dikonversi menjadi bentuk tampak, misalnya melalui tampilan monitor, hasil cetakan pada kertas, atau media visual lainnya.

Di antara berbagai jenis citra tersebut, citra digital menjadi satu-satunya jenis citra yang dapat diolah langsung menggunakan komputer. Jenis citra lain perlu melalui proses digitalisasi terlebih dahulu agar informasi visualnya dapat diproses secara numerik oleh perangkat komputasi

2.6.2 Citra Digital

Citra digital dinyatakan dalam bentuk diskrit, baik pada posisi koordinat maupun nilai warnanya. Dalam hal ini, citra digital digambarkan sebagai matriks dua dimensi, di mana indeks baris dan kolom mewakili posisi titik dalam citra, sementara nilai elemen matriks menggambarkan intensitas warna pada titik tersebut. Setiap elemen matriks yang membentuk citra digital disebut *piksel* (picture element), unit terkecil dalam gambar digital.

Secara matematis, citra digital dinyatakan sebagai fungsi dua variabel, f(x, y), dengan x dan y merujuk pada koordinat spasial di dalam citra. Nilai f(x, y) di titik tersebut menunjukkan intensitas atau kecerahan citra pada koordinat yang bersangkutan. Ilustrasi dari pengertian ini dapat dilihat pada gambar berikut:



Gambar 2.6.2 Fungsi Citra Digital.

Citra digital terbentuk dari titik-titik yang biasanya berbentuk persegi panjang dan tersusun secara teratur dalam baris dan kolom. Setiap titik dalam citra memiliki koordinat tertentu dan biasanya dinyatakan dalam bilangan bulat positif, seperti 0 atau 1, tergantung pada sistem yang digunakan. Format nilai piksel sama dengan format citra secara keseluruhan. Pada sebagian besar sistem pencitraan, nilai ini berupa bilangan bulat positif. Beberapa format citra digital yang umum digunakan antara lain:

1. Citra Biner (Monokrom)

   Citra monokrom, atau citra hitam-putih, merupakan citra satu kanal yang mengandung informasi tingkat keabuan, dengan rentang dari hitam hingga putih pada fungsi f(x, y).
2. Citra Skala Keabuan (Gray Scale)

   Format citra skala keabuan menggunakan dua warna utama: hitam sebagai warna minimum dan putih sebagai warna maksimum. Warna-warna di antaranya digambarkan sebagai berbagai nuansa abu-abu.
3. Citra Berwarna

   Citra berwarna terdiri dari tiga lapisan matriks, masing-masing untuk warna merah (R), hijau (G), dan biru (B). Sistem warna RGB (Red, Green, Blue) digunakan untuk merepresentasikan warna dengan mencampur intensitas dari ketiga warna dasar tersebut.

Grafik kualitas tinggi (*High Quality Raster Graphic*) pada mode 24 bit mengalokasikan 8 bit untuk setiap komponen warna merah, hijau, dan biru. Dalam sistem warna RGB, setiap piksel dinyatakan dengan tiga parameter, yang mewakili komponen warna dasar, bukan nomor warna tertentu. Setiap warna memiliki rentang nilai dari 00 (desimal 0) hingga ff (desimal 255), atau derajat keabuan sebanyak $256 = 2^8$. Dengan demikian, rentang warna yang digunakan mencapai $(2^8)(2^8)(2^8) = 2^{24}$, yang sering disebut sebagai *True Color* dalam sistem operasi Windows. Gabungan intensitas cahaya merah, hijau, dan biru menghasilkan kombinasi warna yang terlihat pada citra.

2.2.3 Elemen-Elemen Citra Digital

Citra digital terdiri dari sejumlah elemen dasar yang digunakan dalam pengolahan citra dan eksploitasi lebih lanjut dalam bidang *computer vision*. Beberapa elemen penting yang ada dalam citra digital antara lain:

1. **Kecerahan (Brightness)**

   Kecerahan menggambarkan intensitas cahaya pada suatu titik (piksel) dalam citra. Meski begitu, nilai kecerahan pada suatu titik tidak mewakili intensitas cahaya sesungguhnya, melainkan merupakan intensitas rata-rata dari area yang mengelilinginya. Sistem visual manusia memiliki kemampuan untuk menyesuaikan tingkat kecerahan dari yang terendah hingga yang tertinggi dalam rentang yang sangat luas, mencapai nilai 10^{10}.

2. **Kontras (Contrast)**

   Kontras menggambarkan perbedaan antara area terang (lightness) dan gelap (darkness) dalam citra. Citra dengan kontras rendah ditandai dengan dominasi satu elemen, baik terang maupun gelap. Sebaliknya, citra dengan kontras yang baik menampilkan distribusi yang lebih merata antara area terang dan gelap di seluruh komposisi citra.

3. **Kontur (Contour)**

   Kontur terbentuk dari perubahan intensitas yang terjadi pada piksel-piksel yang berdekatan. Perubahan ini memungkinkan sistem visual manusia untuk mendeteksi batas-batas objek atau *edge* dalam citra.

4. **Warna (Color)**

   Warna muncul sebagai persepsi visual terhadap panjang gelombang cahaya yang dipantulkan oleh objek. Setiap warna memiliki panjang gelombang (λ) tertentu. Warna merah memiliki panjang gelombang tertinggi, sementara warna ungu (violet) memiliki panjang gelombang terendah.

Warna yang diterima oleh mata manusia merupakan kombinasi cahaya dengan panjang gelombang yang berbeda. menunjukkan bahwa kombinasi warna dengan rentang warna paling luas terdiri dari merah (R), hijau (G), dan biru (B). Persepsi warna pada sistem visual manusia sangat relatif, karena dipengaruhi oleh berbagai faktor, salah satunya adaptasi mata yang dapat menyebabkan distorsi, seperti munculnya bercak abu-abu di sekitar warna hijau.

2.6.3 Pengolahan Citra

Pengolahan citra (image processing) merujuk pada sistem yang menerima citra sebagai input dan menghasilkan citra sebagai output. Awalnya, pengolahan citra difokuskan untuk meningkatkan kualitas gambar. Namun, seiring dengan pesatnya perkembangan komputasi yang didorong oleh peningkatan kapasitas dan kecepatan prosesor komputer, serta munculnya cabang ilmu baru dalam komputasi yang memungkinkan manusia untuk mengekstrak informasi dari citra, pengolahan citra kini tak terpisahkan dari bidang computer vision.

Seiring dengan perkembangan *computer vision*, pengolahan citra memiliki dua tujuan utama, yaitu:

1. **Meningkatkan kualitas citra**.

   Tujuan ini bertujuan agar citra yang dihasilkan dapat menyajikan informasi dengan lebih jelas, sehingga memungkinkan manusia untuk menginterpretasikan informasi yang terkandung di dalamnya. Dalam hal ini, interpretasi tetap dilakukan oleh manusia melalui persepsi visual (*human perception*).

2. **Mengekstraksi Informasi Citra.**

   Tujuan pengolahan citra selanjutnya adalah mengekstraksi informasi ciri yang menonjol pada citra. Proses ini menghasilkan data numerik yang dapat membedakan ciri-ciri visual secara jelas. Informasi yang diekstraksi dapat digunakan untuk mempermudah analisis lebih lanjut oleh sistem komputasi. Seiring perkembangannya, *image processing* dan *computer vision* kini berfungsi sebagai pengganti sistem visual manusia. Perangkat input seperti kamera dan pemindai (*scanner*) berperan sebagai “mata”, sementara mesin komputer, dengan program komputasinya, berfungsi sebagai “otak” yang mengolah informasi visual. Hal ini mendorong munculnya berbagai cabang baru dalam *computer vision*, antara lain:

- *Pattern recognition* (pengolahan pola) *Biometric* (pengenalan identifikasi berdasarkan ciri-ciri biologis yang tampak pada tubuh manusia) *Content-based image and video retrieval* (penarikan citra atau video berdasarkan informasi tertentu), *Video editing*, dan lain sebagainya.

Salah satu cabang yang banyak dikembangkan saat ini adalah biometrik, yang mempelajari cara untuk mengidentifikasi individu berdasarkan ciri fisik atau biologis unik pada tubuh manusia. Contohnya adalah sidik jari, yang menjadi salah satu ciri khas yang membedakan satu orang dengan yang lain. Pengolahan citra diperlukan dalam proses identifikasi wajah, yang dimulai dengan penangkapan citra sidik jari, dilanjutkan dengan ekstraksi ciri untuk memperoleh data numerik yang diperlukan dalam analisis dan identifikasi.

Proses identifikasi sidik jari melibatkan ekstraksi ciri khas seperti *core* (pusat sidik jari) dan *minutiae* (percabangan pada pola sidik jari). Ciri-ciri ini kemudian dipelajari oleh sistem agar komputer dapat mengidentifikasi sidik jari secara akurat.

Seiring dengan kemajuan ilmu komputasi yang memanfaatkan pengolahan citra, identifikasi individu tidak terbatas pada sidik jari saja, tetapi dapat diperluas pada pengenalan wajah (*face recognition*) atau pengenalan iris (*iris recognition*). Dalam kedua model ini, pengolahan citra menjadi lebih kompleks, baik dari sisi proses *capture* atau pengambilan citra, hingga ekstraksi ciri-ciri yang diperlukan. Pada pengenalan wajah, tahap *capture* menjadi sangat krusial, karena faktor seperti pencahayaan, warna, posisi, skala, dan kemiringan wajah menjadi tantangan yang harus diperhatikan untuk mendapatkan citra yang optimal.

Hubungan antara *image processing* dan pembagian bidang dalam komputasi yang melibatkan input serta output tertentu dapat dijelaskan melalui analisis proses yang ada, dengan fokus pada bagaimana pengolahan citra menghasilkan informasi yang dapat diproses lebih lanjut dalam aplikasi komputer.

Tabel 2.6.3 Bidang komputer dapat dilihat melalui perspektif input dan output yang terlibat dalam proses pengolahan data.

		Output	
		Image	Deskripsi
Input	Image	Image Processing	Pattern Recognition, Computer Vision

	Deskripsi	Computer Graphics	Data Processing lainnya

di atas, dapat dilihat bahwa pengolahan citra (*image processing*) merupakan bidang yang mengelola citra sebagai input dan output. Proses dalam pengolahan citra ini bertujuan untuk memperbaiki kualitas gambar atau menyajikan informasi yang terkandung di dalamnya. Agar hasil pengolahan citra menghasilkan data numerik atau teks yang mengungkapkan informasi dalam citra, diperlukan pemahaman yang mendalam mengenai *pattern recognition* dan *computer vision* (Susim et al., 2021)

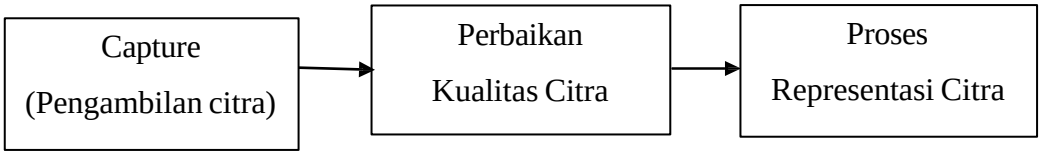
2.7 Hubungan FAR, FRR, ERR dengan Metode Euclidean Distance

FAR (False Acceptance Rate) FAR menggambarkan seberapa sering sistem salah menerima orang yang tidak berhak. Pada face recognition berbasis Euclidean Distance, jika jarak embedding berada di bawah threshold padahal wajah itu milik orang lain, maka terjadi false accept. FAR tinggi berarti sistem tidak aman.

Rumus: FAR = jumlah impostor yang diterima / total percobaan impostor FRR menggambarkan seberapa sering sistem menolak pengguna yang sah. Jika jarak embedding berada di atas threshold padahal wajah tersebut milik pengguna asli, maka terjadi false reject. FRR tinggi membuat sistem tidak nyaman digunakan. Rumus: FRR = jumlah user asli yang ditolak / total percobaan user asli ERR adalah titik ketika FAR sama dengan FRR. ERR menunjukkan kualitas keseluruhan sistem. ERR yang rendah berarti sistem lebih akurat dan stabil.

Cara mendapatkannya: - Uji threshold dari nilai kecil hingga besar. - Hitung FAR dan FRR pada setiap threshold. - Temukan titik perpotongan keduanya dalam sistem face recognition, wajah diubah menjadi embedding (vektor angka), kemudian dihitung jaraknya menggunakan Euclidean Distance. Threshold menentukan apakah wajah diterima atau ditolak. Threshold rendah → sistem ketat → FAR turun tetapi FRR naik. Threshold tinggi → sistem longgar → FAR naik tetapi FRR turun. ERR digunakan untuk menemukan keseimbangan optimal.

Tabel 2.7 Pengolahan Citra



2.7.1 Capture (Pengambilan citra)

Capture (Pengambilan Citra) adalah tahap awal dalam sistem security berbasis face recognition yang mempengaruhi seluruh proses berikutnya, termasuk perhitungan jarak Euclidean Distance. Kualitas citra yang diambil menentukan akurasi embedding dan hasil verifikasi.

1. Tujuan Pengambilan Citra

- Mendapatkan citra wajah yang jelas, tidak blur, dan memiliki pencahayaan cukup.
- Menyediakan input yang layak untuk proses deteksi wajah dan ekstraksi fitur.

2. Perangkat dan Konfigurasi Kamera

- Menggunakan kamera dengan resolusi minimal 720p. -Frame rate 15–30 FPS untuk mengurangi blur. - Jarak ideal kamera 0.5–2 meter. - Kamera ditempatkan sejajar dengan tinggi wajah.

3. Kondisi Pengambilan Citra

a. Pencahayaan - Pencahayaan merata tanpa bayangan kuat. - Hindari cahaya dari belakang (backlight).

b. Pose dan Ekspresi

- Wajah menghadap frontal. - Kemiringan maksimal ±30 derajat.

c. Kebersihan Area Visual

- Wajah tidak tertutup masker, topi, atau rambut berlebihan.

4. Proses Teknis Pengambilan Citra

- Kamera menangkap frame real-time. - Frame dikonversi ke format citra. - Sistem memilih frame terbaik berdasarkan ketajaman, pencahayaan, dan posisi wajah.
- Citra terpilih diteruskan ke modul deteksi wajah.

5. Dampak Capture terhadap Euclidean Distance

- Citra blur menghasilkan embedding tidak stabil sehingga FRR meningkat.
- Pencahayaan buruk menyebabkan fitur wajah sulit terbaca. - Pose ekstrim membuat jarak Euclidean membesar. - Noise atau penutupan wajah menurunkan akurasi.

6. Aturan Capture untuk Enrol dan Verifikasi

a. Enrol : - Mengambil 3–5 citra berbeda. - Memastikan wajah terdeteksi dan tidak blur.

b. Verifikasi : - Citra diambil secara real-time. - Sistem otomatis memilih frame terbaik.

Capture suatu pondasi utama dalam face recognition. Citra yang berkualitas tinggi membuat jarak Euclidean lebih stabil, meningkatkan akurasi sistem, menurunkan FAR dan FRR, serta memperkuat keamanan sistem secara keseluruhan.

2.7.2 Perbaikan Kualitas Citra

Perbaikan kualitas citra (image enhancement) suatu langkah penting dalam sistem security berbasis face recognition. Tahap ini bertujuan meningkatkan kualitas visual citra agar proses deteksi wajah, ekstraksi fitur, dan perhitungan jarak Euclidean Distance berjalan lebih akurat

1. Tujuan Perbaikan Kualitas Citra

- Menghilangkan noise, bayangan, atau blur. - Menyamakan pencahayaan antar citra. - Memperjelas tekstur wajah. - Menghasilkan citra konsisten untuk embedding yang stabil.

2. Faktor yang Mempengaruhi Kualitas Citra

- Pencahayaan tidak merata. - Blur karena pergerakan. - Noise dari kamera rendah. - Pose wajah miring. - Wajah tertutup objek.

3. Teknik Perbaikan Kualitas Citra

a. Normalisasi Pencahayaan

- Histogram Equalization. - CLAHE (Contrast Limited Adaptive Histogram Equalization).

b. Noise Reduction

- Gaussian Filter. - Median Filter. - Bilateral Filter.

c. Sharpness Enhancement

- Unsharp Masking. - High-Boost Filtering.

d. Face Alignment

- Memutar dan menyelaraskan wajah agar mata sejajar. - Menempatkan wajah di tengah agar embedding konsisten.

e. Resize dan Normalisasi Pixel

- Menyamakan ukuran citra (misalnya 100×100 atau 160×160 px atau 224×224 px).
- Normalisasi nilai pixel agar model stabil.

4. Dampak Enhancement terhadap Akurasi Sistem

- Noise tinggi → embedding kacau → jarak Euclidean besar → FRR naik.
- Citra gelap → detail hilang → verifikasi gagal.
- Pose miring → embedding berubah jauh → false reject.
- Blur → jarak tidak stabil → akurasi menurun.

5. Pipeline Enhancement dalam Face Recognition

1. Capture citra. 2. Enhancement. 3. Deteksi wajah. 4. Alignment.
5. Ekstraksi embedding. 6. Perhitungan Euclidean Distance. 7. Keputusan accept/reject. Perbaikan kualitas citra memastikan embedding yang stabil sehingga perhitungan Euclidean Distance lebih akurat. Langkah ini meningkatkan akurasi, menurunkan FAR dan FRR, serta memperkuat keamanan sistem face recognition.

2.7.3 Proses Representasi Citra

Proses representasi citra suatu tahapan yang mengubah citra wajah dari bentuk visual menjadi bentuk numerik berupa vektor fitur. Vektor fitur ini berisi nilai-nilai yang mewakili karakteristik unik wajah seperti pola pencahayaan, struktur bentuk, hingga intensitas pixel. Dalam sistem face recognition berbasis Euclidean Distance, representasi citra memegang peran penting karena sistem hanya dapat membandingkan wajah melalui bentuk angka, bukan gambar.

1. Tujuan Representasi Citra

Tahap ini bertujuan untuk:

- Mengubah citra wajah menjadi data numerik yang dapat dihitung.
- Menyederhanakan citra agar lebih mudah dan cepat diproses.
- Mengambil fitur unik wajah yang membedakan satu wajah dengan yang lain.
- Menghasilkan vektor fitur yang stabil dan konsisten untuk perhitungan jarak Euclidean Distance.

2. Tahapan Proses Representasi Citra

a. Normalisasi Citra

Normalisasi dilakukan untuk memastikan seluruh citra memiliki kondisi yang sama.

Langkah umum normalisasi:

Menyamakan ukuran citra (misal 100×100 pixel). Menstabilkan pencahayaan (brightness dan contrast). Memposisikan wajah tegak dan berada di tengah citra. Tujuan: menghindari perubahan nilai fitur karena beda pencahayaan atau posisi.

b. Ekstraksi Area Wajah (Face Region Extraction)

Pada tahap ini, sistem mengambil area penting dari wajah dan menghilangkan bagian yang tidak relevan.

Area yang dipertahankan:

- Mata
- Hidung
- Mulut
- telinga

Latar belakang (background) dibuang karena tidak memiliki kontribusi pada identifikasi wajah.

c. Transformasi Citra ke Data Numerik

Citra kemudian diubah menjadi bentuk angka:

- Citra diubah menjadi grayscale 0–255.
- Setiap pixel menjadi nilai intensitas.
- Semua nilai pixel disusun menjadi vektor fitur.

Contoh:

Citra 100×100 pixel → menghasilkan 10.000 angka fitur.

Contoh vektor: [12, 45, 88, 120, 210, 255,]

d. Reduksi Dimensi (Opsional)

Jumlah fitur yang terlalu besar dapat memperlambat komputasi. Oleh karena itu digunakan:

- PCA (Principal Component Analysis)
- LDA (Linear Discriminant Analysis)

Manfaat reduksi dimensi:

- Menghapus fitur yang tidak penting. -Mempercepat proses *matching*.
- Mengurangi noise pada data fitur.

3. Output dari Representasi Citra

Hasil akhir dari proses ini adalah vektor fitur wajah yang berisi nilai-nilai numerik. Vektor ini berfungsi sebagai "sidik wajah digital" yang digunakan untuk perhitungan Euclidean Distance.

2.8 Pengenalan Wajah

Secara umum, sistem pengenalan wajah dapat diklasifikasikan menjadi dua kategori utama, yaitu sistem berbasis fitur (feature-based) dan sistem berbasis citra (image-based). Pada sistem berbasis fitur 0, proses identifikasi dilakukan dengan mengekstraksi ciri-ciri khusus dari bagian-bagian wajah seperti mata, hidung, dan mulut. Ciri-ciri tersebut kemudian dimodelkan secara geometris untuk membentuk representasi yang dapat digunakan dalam proses pengenalan.

Sebaliknya, sistem berbasis citra menggunakan informasi mentah langsung dari piksel wajah tanpa mengekstraksi bagian-bagian tertentu. Data piksel tersebut kemudian diolah menggunakan metode representasi tertentu, misalnya Principal Component Analysis (PCA) atau transformasi wavelet, untuk menghasilkan model yang dapat digunakan dalam proses klasifikasi identitas wajah (Muwardi & Adisaputro, 2021).

2.8.1 Python

Python sebuah bahasa pemrograman tingkat tinggi (high level language) yang dikembangkan oleh Guido van Rossum pada tahun 1989 dan diperkenalkan untuk pertama kalinya pada tahun 1991.Dalam bahasa python sendiri terdapat bahasa tingkat rendah (low level language) yang berhubungan dengan bahasa mesin atau assembly.

Python memiliki beberapa kelebihan dibandingkan dengan bahasa pemrograman lainnya. Pemrograman tingkat tinggi (High Level Language). mudah dipelajari, mudah digunakan, Mudah dalam pengembangan, manajemen memori dinamis. pemrograman berorientasi objek (Object Oriented Programming), platform independent, bersifat open source dan gratis

2.8.2 Euclidean Distance

Jarak digunakan untuk mengukur tingkat kemiripan (similarity degree) atau perbedaan (dissimilarity degree) antara dua vektor fitur hasil ekstraksi citra. Penilaian dilakukan berdasarkan nilai skor jarak yang menggambarkan tingkat

kedekatan antar vektor.

Semakin kecil nilai jarak, semakin besar tingkat kemiripan dua citra wajah, sedangkan nilai jarak yang besar menunjukkan perbedaan karakteristik yang signifikan. Metode yang banyak diterapkan dalam sistem pengenalan wajah menggunakan Euclidean Distance. Metode ini menghitung jarak antara dua titik dalam ruang berdimensi n dengan menggunakan akar kuadrat dari jumlah selisih kuadrat tiap komponen fitur.

Rumus matematis:

$D(x, y) = \sqrt{\Sigma(xi - yi)^2}$

Keterangan:

D (x, y) → jarak Euclidean antara dua vektor fitur x dan y.

xi, yi → nilai komponen ke-i dari masing-masing vektor n → jumlah dimensi atau fitur yang dibandingkan

Perhitungan:

Misalkan dua vektor fitur wajah diperoleh dari proses ekstraksi citra:

x = [2, 4, 6]

y = [3, 5, 9]

Maka perhitungannya sebagai berikut:

$D(x, y) = \sqrt{((2 - 3)^2 + (4 - 5)^2 + (6 - 9)^2)}$

$D(x, y) = \sqrt{(1 + 1 + 9)}$

$D(x, y) = \sqrt{11}$

D (x, y) = 3.3166

Hasil perhitungan menunjukkan nilai jarak sebesar 3.3166.

Nilai ini dapat digunakan untuk menentukan apakah dua citra memiliki kemiripan yang tinggi atau rendah.

Jika dibandingkan dengan nilai ambang batas tertentu (*threshold*), sistem dapat memutuskan apakah wajah yang diuji sesuai dengan data wajah yang tersimpan di basis data.

Rumus dari *Euclidean Distance:*

$dij = \quad d_{ij} = \sqrt{\sum_{i=1}^{n}(x_i - x_j)^2}$

Contoh :

Terdapat 2 vektor ciri berikut:

A = [0,3,4,5]

B = [7,6,3,-1]

Euclidean Distance dari vektor A dan B adalah :33

$$d_{AB} \equiv \sqrt{(0-7)^2 + (3-6)^2 + (4-3)^2 + (5-(-1))^2}$$

$$= \sqrt{49 + 9 + 1 + 36 = 9{,}747}$$

2.9 **Webcam**

Webcam (singkatan dari *web camera*) termasuk kamera digital berukuran kecil yang dapat terhubung ke komputer melalui port USB atau COM. Perangkat ini banyak diproduksi dengan berbagai merek, seperti LogiTech, Itech, dan SunFlower. Resolusi gambar yang dihasilkan umumnya berkisar antara 352 × 288 piksel hingga 100 × 100 piksel, namun beberapa tipe mampu mencapai kualitas 2 megapiksel. Perkembangan teknologi saat ini memungkinkan kamera digital dan telepon pintar berfungsi sebagai webcam.

Secara umum, istilah *webcam* digunakan untuk menyebut teknologi pemantauan berbasis kamera. Dalam praktiknya, kata *web* sering diganti sesuai dengan fungsi atau objek yang dipantau, misalnya *StreetCam* untuk menampilkan kondisi jalan, *MetroCam* untuk panorama kota, *TraffiCam* untuk memantau lalu lintas, *WeatherCam* untuk cuaca, hingga *VolcanoCam* untuk mengamati aktivitas gunung berapi.

Webcam biasanya dilengkapi kabel penghubung fleksibel yang tersambung ke papan sirkuit (*Printed Circuit Board/PCB*). Ujung lainnya terpasang konektor yang dapat diatur sesuai kebutuhan sudut pandang, ketinggian, dan arah kamera. Setiap webcam juga memiliki perangkat lunak bawaan (*software*) yang berfungsi menangkap gambar secara berkelanjutan atau dalam interval waktu tertentu, kemudian menyiarkannya melalui koneksi internet.

Proses penyiaran dilakukan dengan beberapa metode, salah satunya melalui konversi gambar menjadi format JPEG yang diunggah ke *web server* menggunakan protokol FTP (*File Transfer Protocol*). Kecepatan *frame rate* menunjukkan seberapa banyak gambar dapat ditangkap dan dikirim setiap detik. Untuk

menghasilkan video *streaming* yang halus dibutuhkan kecepatan minimal 10 *frame per second* (fps), sedangkan kualitas ideal berada pada 20 fps.

Kinerja *frame rate* yang tinggi memerlukan dukungan koneksi internet berkecepatan besar. Beberapa webcam modern tidak lagi bergantung pada komputer karena telah memiliki perangkat lunak dan *web server* internal. Kamera jenis ini dikenal dengan sebutan *network camera*, cukup disambungkan ke jaringan internet agar dapat beroperasi secara mandiri.

Fungsi webcam kini berkembang luas, mencakup kegiatan *video conferencing*, *video messaging*, *internet dating*, *home monitoring*, *image sharing*, *video interviews*, hingga *video phone calls*. Pada penggunaan konferensi video, kamera berukuran kecil sering dipilih karena mudah diintegrasikan dengan komputer. Dalam beberapa kasus, kamera analog juga masih digunakan dengan bantuan *video capture card* agar dapat diakses secara daring.

Kemajuan teknologi terkini menghadirkan webcam dengan fitur tambahan seperti mikrofon internal dan sistem *noise cancellation* untuk mengurangi gangguan suara dari sekitar. Dengan begitu, komunikasi saat konferensi menjadi lebih jelas dan fokus pada pembicara yang berada di depan kamera.

Secara umum, fitur-fitur yang tersedia pada webcam mencakup:

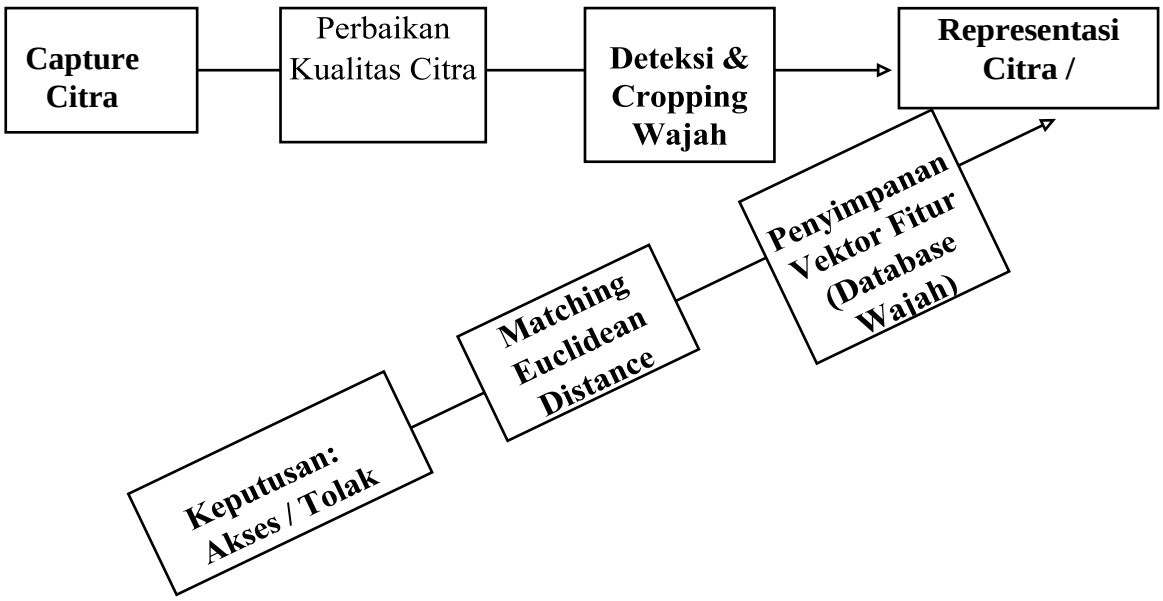
1. **Motion sensing** : kamera menangkap gambar ketika mendeteksi adanya pergerakan di area pantauan.
2. **Image archiving** : pengguna dapat menyimpan hasil tangkapan gambar secara berkala atau hanya pada waktu tertentu sesuai pengaturan interval.
3. **Video messaging** : beberapa aplikasi pesan instan mendukung pengiriman video langsung dari webcam.
4. **Advanced connections** : perangkat dapat dihubungkan ke sistem *home theater* melalui kabel maupun koneksi nirkabel.
5. **Automotion** : kamera robotik yang mampu melakukan pergerakan *pan* dan *tilt* dengan pengaturan posisi otomatis.
6. **Streaming media** : sistem profesional menggunakan kompresi MPEG4 untuk menghasilkan siaran audio dan video secara langsung (*real-time*).

7. **Custom coding** : pengguna dapat menambahkan kode pemrograman agar kamera menjalankan fungsi tertentu, misalnya memperbarui tampilan secara otomatis.
8. **AutoCam** : fitur yang memberi kemudahan bagi pengguna untuk membuat halaman web pribadi berisi siaran dari webcam melalui server milik produsen perangkat.

BAB III
PERANCANGAN SISTEM

3.1 Perancangan Sistem

Simulator matriks berperan sebagai komponen utama yang akan digunakan dalam implementasi sistem yang dirancang dalam penelitian ini. Untuk menggambarkan keseluruhan arsitektur sistem, diperlukan diagram blok yang mampu memberikan visualisasi menyeluruh mengenai alur dan fungsi dari masing-masing bagian sistem. Setiap blok pada diagram tersebut merepresentasikan proses spesifik yang memiliki fungsi tersendiri dalam menunjang kinerja sistem secara keseluruhan. Dalam penelitian ini, pendekatan yang digunakan melibatkan penerapan metode *Euclidean Distance* untuk ekstraksi ciri tekstur, sebagai algoritma klasifikasi guna mendukung proses pengambilan keputusan dalam sistem yang dikembangkan, diagram blok memungkinkan batasan yang lebih terarah untuk perancangan sistem yang mendeteksi wajah . Berikut diagram blok rancangan sistem.



Gambar 3. 1 Diagram Blok Sistem

3.1.1 Capture Citra (Pengambilan Gambar)

Tahap pertama berupa pengambilan gambar wajah menggunakan kamera. Proses ini meliputi: Kamera menangkap wajah secara *real-time*. Mengambil beberapa frame untuk memastikan kualitas.

Menyimpan citra ke buffer untuk diproses lebih lanjut. Tujuan: mendapatkan gambar awal sebagai input sistem.

3.1.2 Perbaikan Kualitas Citra (Image Enhancement)

Setelah gambar didapat, dilakukan peningkatan kualitas agar lebih mudah dianalisis. Perbaikan mencakup: Normalisasi pencahayaan – Penyesuaian kontras dan brightness - Noise reduction - Filter smoothing atau sharpening Tujuan: menghasilkan citra yang lebih bersih dan stabil.

3.1.3 Deteksi Wajah & Cropping (Face Detection)

Pada tahap ini, sistem mencari lokasi wajah dalam gambar.
Biasanya menggunakan: YOLO Face Setelah wajah ditemukan: Sistem melakukan cropping pada bagian wajah saja. Bagian selain wajah dibuang. Tujuan: fokus pada area yang relevan agar akurasi meningkat.

3.1.4 Representasi Citra / Ekstraksi Fitur

Proses meliputi: Mengubah wajah menjadi grayscale. Meratakan ukuran (misal 100×100 pixel). Mengambil nilai pixel sebagai vektor fitur. Bisa dilakukan reduksi dimensi (misal PCA) untuk: mengurangi noise mempercepat proses perhitungan Hasil akhir: Wajah diwakili dalam bentuk angka seperti: [12, 46, 80, 120, 200, 255,]

3.1.5 Penyimpanan Vektor Fitur (Database Wajah)

fitur yang sudah dihasilkan disimpan ke dalam database. Struktur database biasanya berisi: ID pengguna, Nama, Waktu, Status Tujuan: menyediakan dataset untuk proses perbandingan nanti.

3.1.6 Matching Menggunakan Metode Euclidean Distance

Tahap ini membandingkan vektor wajah yang baru dengan vektor yang ada di database Rumus Euclidean Distance **:**

$$\left(\boldsymbol{d = sqrt(x1 - y1)^2}\ )\ (\boldsymbol{x2\ - y)^2})(+ \cdots +)\ (\boldsymbol{xn - \ yn)^2})\right)$$

Interpretasi: d kecil → wajah mirip/sama d besar → wajah berbeda sistem biasanya menetapkan nilai threshold untuk menentukan batas maksimal jarak yang masih dianggap cocok.

3.1.7 Keputusan (Akses Diterima / Ditolak)

Setelah nilai jarak dibandingkan: Jika jarak ≤ threshold → akses diberikan Jika jarak > threshold → akses ditolak Sistem dapat menampilkan: Nama

pengguna persentase kecocokan log waktu dan status aksesTahap akhir ini adalah penentu apakah seseorang boleh masuk atau tidak.

3.1.8 Hasil Akuisisi Citra

Hasil akuisisi citra adalah output berupa gambar wajah yang berhasil ditangkap kamera sebagai langkah awal dalam sistem face recognition. Citra ini menjadi dasar seluruh proses selanjutnya, mulai dari deteksi wajah, peningkatan kualitas citra, ekstraksi fitur, hingga perhitungan Euclidean Distance.

Hasil akuisisi citra memiliki beberapa tujuan utama, yaitu:

1. Menyediakan citra mentah untuk diproses dalam sistem pengenalan wajah.
2. Menangkap bentuk dan struktur wajah secara jelas agar fitur mudah diekstraksi.
3. Mengurangi risiko kesalahan pada tahap pendeteksian dan pengenalan.
4. Menjamin konsistensi data agar kualitas citra tetap stabil untuk setiap proses pengenalan.

a. Kamera

- Resolusi menentukan ketajaman citra. Frame rate memengaruhi kelancaran penangkapan wajah real-time. Sensor (CMOS/IR) mempengaruhi sensitivitas cahaya.

b. Pencahayaan

- Pencahayaan cukup menghasilkan citra dengan noise rendah. Cahaya terlalu terang menyebabkan over-exposure, terlalu gelap menyebabkan blur dan noise.

c. Jarak Kamera ke Objek

- Jarak ideal 30–80 cm. Jika terlalu jauh: detail wajah tidak terbaca. Jika terlalu dekat: distorsi terjadi.

d. Sudut Wajah

- Sudut wajah frontal paling optimal. Kemiringan ekstrem mengurangi keberhasilan pendeteksian.

e. Pergerakan Wajah

- Gerakan cepat dapat menyebabkan citra kabur. Sistem biasanya mengharuskan wajah stabil selama kurang dari 1 detik.

4. Bentuk Hasil Akuisisi Citra

Citra hasil akuisisi dapat berupa:

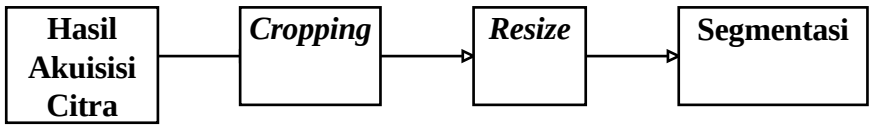
- RGB (warna): paling umum digunakan. Grayscale: lebih sederhana untuk komputasi. Infrared (IR): untuk keadaan minim cahaya.

Format citra yang umum:

- JPG - PNG

3.1.9 Pre-Processing

Pre-processing adalah proses tahap awal yang bertujuan untuk mengolah citra mentah agar siap digunakan dalam proses pemrosesan lanjutan. Proses ini dilakukan untuk memperbaiki kualitas citra masukan, baik dari segi kontras, ketajaman, maupun mengurangi gangguan (noise), sehingga fitur penting dalam citra dapat diekstraksi secara lebih akurat dan optimal pada tahap berikutnya dihasilkan.



Pada tahap *pre-processing* terdapat beberapa langkah, yaitu :

1. *Cropping*

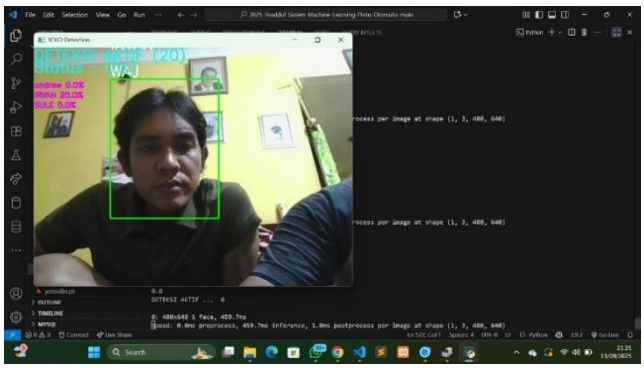
Cropping memotong bagian tertentu dari citra untuk mengambil area yang relevan dan menghilangkan bagian yang tidak diperlukan dalam sistem face recognition, cropping digunakan untuk mengambil area wajah secara presisi dari gambar atau frame kamera sehingga hanya informasi penting yang diproses. Cropping memastikan bahwa data yang masuk ke tahap ekstraksi fitur merupakan wajah murni, bukan seluruh background atau objek lain setelah cropping, citra wajah biasanya diubah ke ukuran standar, misalnya 100×100 px atau 128×128 px, agar digunakan untuk perhitungan Euclidean Distance. Dengan cropping yang tepat semua wajah mempunyai ukuran yang sama, - Perhitungan jarak Euclidean lebih konsisten - background terhapus - fitur wajah lebih jelas -noise berkurang

2. *Resize*

Seluruh citra, baik yang digunakan pada tahap pelatihan maupun pengujian, diseragamkan ukurannya ke dalam dimensi tertentu guna memastikan keseragaman struktur data yang diolah oleh sistem. Pada penelitian ini, proses penyesuaian ukuran (resizing) dilakukan dengan menetapkan resolusi citra menjadi 804 × 652 piksel, sehingga memungkinkan proses ekstraksi ciri dan klasifikasi berjalan secara konsisten tanpa terpengaruh oleh variasi dimensi awal dari masing-masing gambar.

3. Segmentasi

Proses mencuplik beberapa bagian gambar untuk mengidentifikasi sifat teksturnya. Dalam penelitian ini, penulis menemukan Ciri khas berupa lingkaran berwarna putih yang dikenal dengan istilah *Arcus Senilis* dapat dikenali melalui proses segmentasi manual yang dilakukan secara spesifik pada bagian tepi atau batas luar dari struktur face recognition. Teknik ini bertujuan untuk mengekstraksi pola visual tertentu yang muncul pada daerah perifer pada face sebagai indikasi adanya perubahan.



Gambar 3. 6 Citra Hasil *Pre-Processing*

3.2.1 Parameter Performasi Sistem

Setelah proses selesai, sistem harus diuji. Pengujian ini berguna untuk mengetahui apakah sistem pengaplikasian deteksi face recognition berhasil atau tidak melalui face recognition dengan metode euclidean. Berikut adalah beberapa parameter performansi sistem yang dibutuhkan:

1. Tingkat ketepatan kinerja sistem

Sistem merupakan indikator kuantitatif yang digunakan untuk menilai sejauh mana sistem mampu mengenali dan mencocokkan data masukan dengan informasi aktual, sehingga menghasilkan output yang benar dan sesuai harapan. Nilai akurasi menunjukkan persentase keberhasilan sistem dalam melakukan identifikasi atau klasifikasi secara benar dibandingkan dengan total jumlah data yang diuji. Nilai yang lebih tinggi dari akurasi sistem menunjukkan bahwa sistem bekerja lebih baik. Secara garis besar, dapat dituliskan sebagai berikut:

$$Akurasi = \frac{Jumlah\ data\ benar}{Jumlah\ seluruh\ data}\ X\ 100\%$$

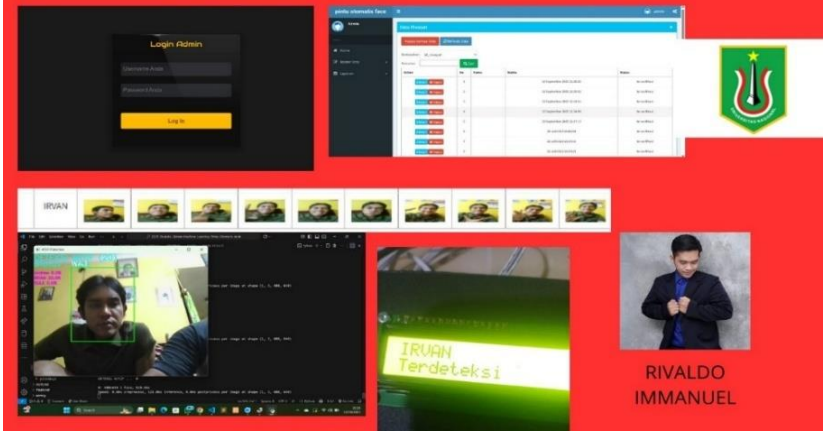
2. Durasi Pemrosesan Sistem

Durasi yang dibutuhkan oleh sistem untuk menyelesaikan seluruh tahapan pemrosesan, mulai dari awal hingga akhir, dikenal sebagai waktu pemrosesan komputasi. Pengukuran terhadap waktu ini sangat penting karena dapat digunakan untuk memprediksi performa sistem dalam menyelesaikan tugas tertentu. Semakin singkat waktu yang dibutuhkan, maka sistem dianggap memiliki efisiensi yang lebih tinggi. Nilai waktu pemrosesan ini dapat dihitung menggunakan pendekatan matematis sebagaimana dijelaskan dalam rumus berikut:

$$Waktu\ komputasi = Waktu\ selesai - Waktu\ mulai$$

3. Halaman Face Recognition

dari halaman menunjukkan halaman Face Recognition grafis sistem yang dibuat, di mana pengguna dapat mengunggah gambar face recognition yang ada di komputer mereka. Setelah itu, tekan tombol Proses untuk melihat hasil segmentasi pada face recognition dan melihat hasil klasifikasi gambar pada dibawah ini.



Gambar 3.8 Halaman Sensor Face Recognition Sesudah Running

BAB IV
PENGUJIAN SISTEM DAN ANALISIS

Dalam perancangan sistem deteksi face recognition, diperlukan beberapa sistem ini didukung oleh kombinasi komponen perangkat keras dan perangkat lunak yang saling mendukung dalam pelaksanaan proses penelitian. Adapun rincian spesifikasi dari kedua komponen tersebut dijelaskan sebagai berikut :

1. Tipe Komputer : HP.
2. *Processor* : Intel® Core™ i5-120U CPU @ 1.40GHz.
3. RAM : 16 GB.
4. *Harddisk* : 500GB.
5. Kamera ponsel dengan lensa 8 Megapiksel dengan teknik pengambilan foto yang telah ditentukan melalui HP.
6. Sistem Operasi : Windows 11
7. *Software* : Visual Studio Code

4.1 Tahapan Pengujian Sistem

Untuk mengoptimalkan pengujian sistem dan analisis, beberapa skenario parameter akan digunakan. Tahap pertama adalah menemukan parameter terbaik untuk setiap skenario, yang akan menjadi parameter tahap uii. Ciri input adalah gambar iris mata yang diambil menggunakan kamera telepon dengan resolusi 8 Megapiksel.

1. Tahapan Pertama

Sebanyak 10 gambar face recoginition digunakan untuk setiap untuk di daftarkan ke sistem skenario pertama, digunakan dua jenis data yaitu data pelatihan dan data pengujian. Untuk proses pelatihan, setiap kelas dibekali dengan sebanyak 10 gambar dengan muka dengan gaya yang berbeda setiap orangnya yang mewakili karakteristik masing-masing kategori, 10 gambar face untuk agar bisa kedetek, 10 gambar untuk setiap data latih, dan 10 gambar untuk setiap face diuji. Database akan digunakan untuk menyimpan gambar yang dilatih, termasuk face recognition. Pada tahap pre-processing, gambar yang telah diakuisisi akan diresize, disegmentasi, dan diubah menjadi euclidean distance.

2. Tahapan Kedua

Setelah proses pra-pemrosesan selesai dilakukan, tahap selanjutnya adalah ekstraksi fitur, yang difokuskan pada analisis karakteristik tekstur menggunakan metode euclidean. Teknik ini digunakan untuk memperoleh nilai rata-rata dari beberapa parameter statistik orde dua, meliputi kontras, korelasi, energi, dan homogenitas. Pada tahap ini, setiap face dianalisis untuk diidentifikasi fitur-fitur teksturnya berdasarkan prinsip yang ditetapkan oleh metode eudclidean. Hasil dari proses ekstraksi ini kemudian digunakan sebagai data representatif untuk proses pengujian pada tahap klasifikasi Selanjutnya, mengubah nilai jarak piksel (d) menjadi 1, 2 dan 3, dan level kuantisasi (n) menjadi 8, 16 dan 32.

3. Tahapan Ketiga

Setelah ciri diekstraksi, gambar diklasifikasikan atau dikelompokkan menjadi yang terdaftar atau tidak terdaftar, dan disitu sistem mendetek. Ini Proses klasifikasi dilakukan dengan menerapkan algoritma euclidean. Dalam penerapannya, digunakan berbagai jenis fungsi kernel, seperti Gaussian, Linear, Radial Basis Function (RBF), dan Polynomial, yang berfungsi untuk mentransformasikan data ke dalam ruang berdimensi lebih tinggi. Selain itu, pendekatan multikelas seperti One Against One (OAO) dan One Against All (OAA) juga digunakan sebagai strategi pengklasifikasian, dan keduanya berperan sebagai parameter utama dalam proses pengujian kinerja model klasifikasi.

4. Tahapan Keempat

Langkah akhir dalam proses penelitian ini, dilakukan dalam mendapatkan waktu komputasi dan akurasi terbaik dengan mengubah parameter metode euclidean.

4.2 Analisis Pengujian Sistem

Sebagai bagian dari proses pemrograman sistem, beberapa parameter dikumpulkan untuk digunakan untuk melakukan pengujian sistem. Masing-masing hasil dari setiap tes ditunjukkan dalam bentuk tabel dan kesimpulan dalam bentuk grafik. Dalam penelitian ini, skenario pengujian sistem dilakukan dengan menggunakan metode ekstraksi ciri euclidean distance. Dalam pengujian ini, parameter fitur orde dua, level kuantisasi dan jarak piksel, serta jenis kernel dan

multiclass yang digunakan, diamati perubahan dalam tingkat akurasi dan waktu komputasi..

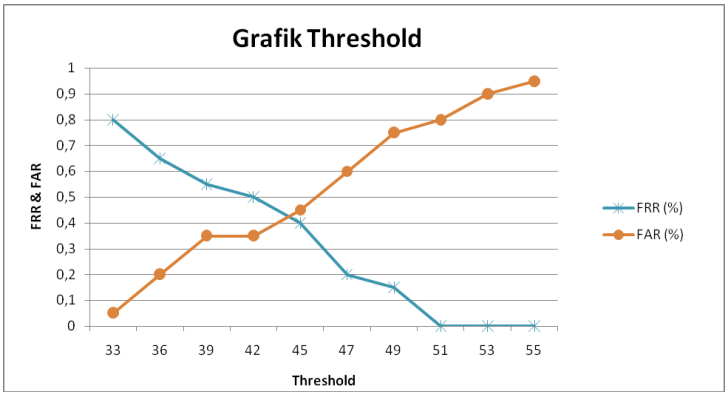
4.2.1 Evaluasi terhadap Variasi Proporsi Data Pelatihan dan Pengujian

Dalam percobaan mengambil nilai threshold terdapat 2 percobaan yaitu uji coba dengan menggunakan orang yang sebagai anggota dan yang bukan. Untuk mencari nilai threshold yang cocok menggunakan formulasi yang disebut dengan *False Acceptance Rate* (FAR) dan *False Rejection Rate* (FRR) . FAR merupakan rata-rata kesalahan ketika bukan anggota tapi dikenali dan FRR sebuah rata-rata kesalahan anggota tetapi tidak dikenali. Rentang uji coba nilai yang digunakan dalam mencari nilai threshold yaitu 33 sampai 55, hal ini berdasarkan minimum nilai threshold yang sering muncul ketika terjadi pencocokan. Berikut ini table percobaan dalam mencari nilai threshold.

Tabel 4.1 Nilai FRR dan FAR

No.	Threshold	FRR	FAR
1	33	0,8	0,05
2	36	0,65	0,2
3	39	0,55	0,35
4	42	0,5	0,35
5	45	0,4	0,45
6	47	0,2	0,6
7	49	0,15	0,75
8	51	0	0,8
9	53	0	0,9
10	55	0	0,95

Sedangkan dibawah ini grafik threshold yang menunjukkan nilai FRR dan FAR



Gambar 4.2 Grafik FAR dan FRR

Dalam grafik diatas menunjukkan nilai FAR dan FRR terjadi pertemuan pada rentang threshold pada titik 42 dan 45. Dilihat dari titik pertemuan tersebut, terjadi cenderung mendekati titik 45,dengan demikian dapat diambil kesimpulan nilai threshold yang cocok digunakan dalam keadaan itu yaitu 44.

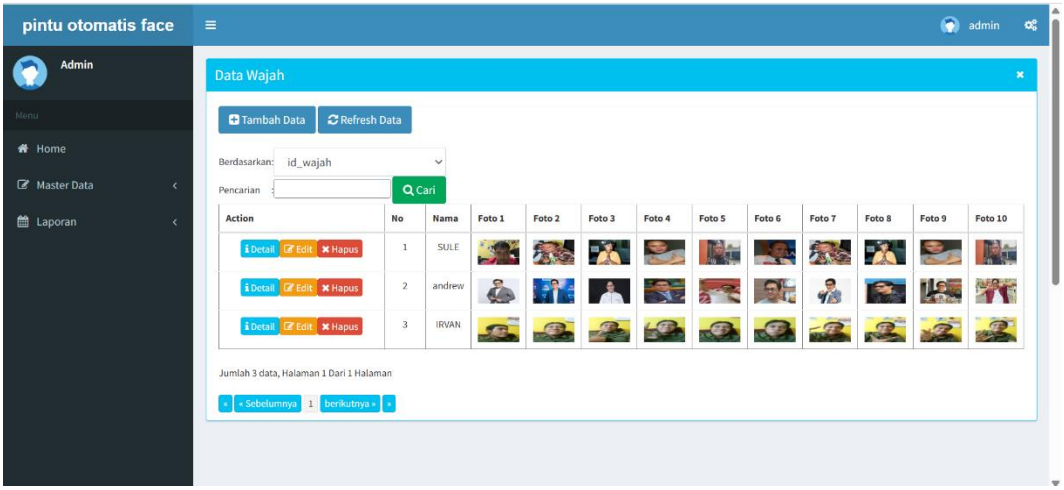
Hasil uji coba sistem monitoring ruangan dengan menggunakan 3 jarak yaitu 1m,1,5m, dan 2m dengan masing –masing jarak memiliki 10 kondisi sebagai berikut :

Tabel 4.2 Macam – macam Kondisi

Kondisi	Keterangan
1	Mengahadap depan
2	Mengahadap ke Kanan 25^0
3	Mengahadap ke xKanan 45^0
4	Menghadap ke Kiri 25^0
5	Menghadap ke Kiri 45^0
6	Menghadap ke Atas 25^0
7	Menghadap ke Atas 45^0
8	Menghadap ke Bawah 25^0
9	Menghadap ke Bawah 45^0
10	Menghadap kedepan dan memejamkan mata

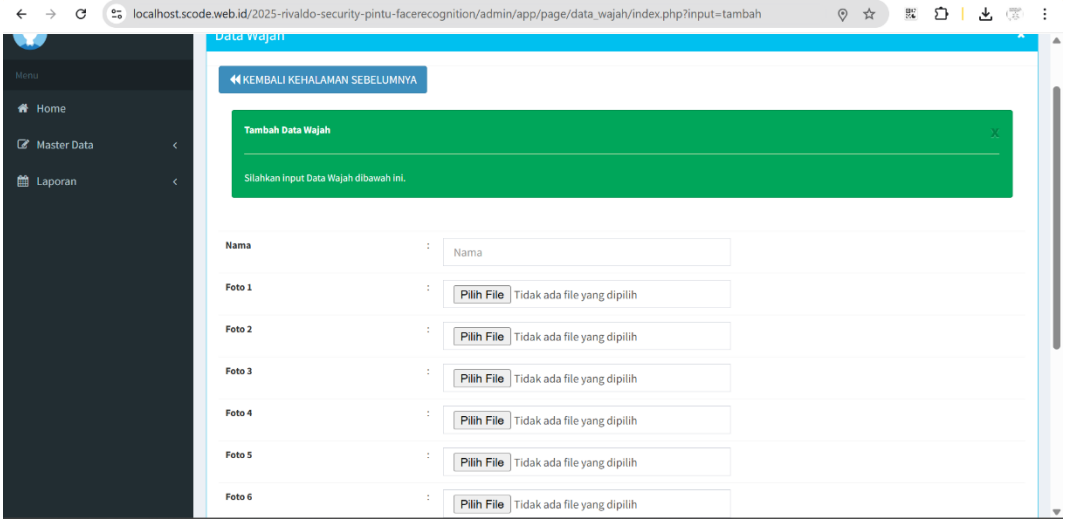
4.2.2 Implementasi Interface

Interface aplikasi sistem monitoring ruangan terdapat 4 menu, yaitu menu tambah data anggota, daftar data anggota, edit anggota , dan hapus anggota Berikut ini user interface dari web monitoring



Gambar 4.1 Tampilan user pada website

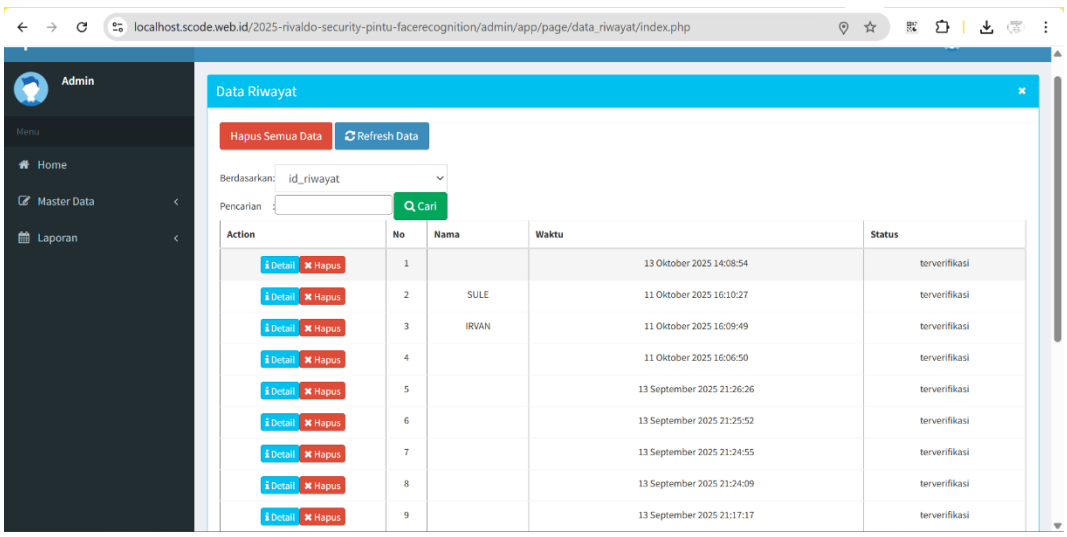
2. **Form Tambah Data Anggota**



Gambar 4.2.2 Form Tambah Wajah

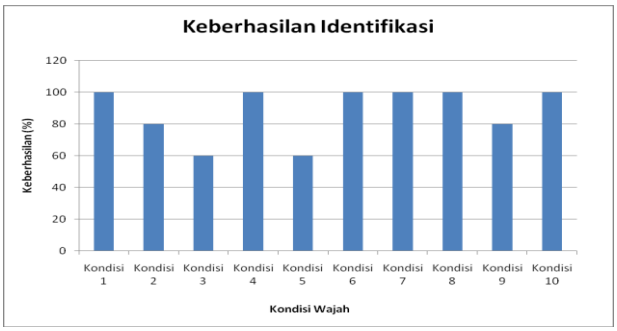
3. **Form Lihat Monitoring**

Untuk mengetahui orang yang masuk ke ruangan yang dilengkapi ada nama, waktu yang berisi tanggal dan jam sama status terverifikasi akses dibuka berada di form lihat monitoring, utnuk mengakses form ini yaitu klik master data dan klik data riwayat lihat monitoring siapa yang masuk berapa orang



Berikut ini merupakan tabel dan grafik hasil uji coba:

a. Hasil simulasi berdasarkan variasi data training jarak 1 Meter Adapun grafik percobaan jarak 1 m sebagai berikut



Gambar 4.13 Grafik keberhasilan 1 Meter

Dan tabel dibawah ini merupakan rincian dari grafik diatas:

Tabel 4.13 Keberhasilan Dari Grafik

Posisi	Jumlah Terdeteksi	Jumlah Berhasil
Kondisi 1	5	5
Kondisi 2	5	4
Kondisi 3	5	3
Kondisi 4	5	5
Kondisi 5	5	3
Kondisi 6	5	5
Kondisi 7	5	4
Kondisi 8	5	4
Kondisi 9	5	3
Kondisi 10	5	5

Dari percobaan diatas didapatkan beberapa analisa sebagai berikut :

Jarak objek dengan webcam sangat mempengaruhi dalam akurasi keberhasilan dalam mengidentifikasi wajah. Hal tersebut bisa dilihat pada grafik semakin jaraknya jauh akurasi semakin menurun

4.2.3 Source Code

```
import numpy as np
import face_recognition
import cv2
import os
import time
import requests
import shutil

# Coba impor pustaka face_recognition, instal jika belum ada
try:
    import face_recognition
    from tqdm import tqdm
    import pickle
    import cv2
    from ultralytics import YOLO
    import numpy as np
    import json
    import serial
    import serial.tools.list_ports
    from datetime import datetime
    from datetime import date
except ImportError:
    print("Library tidak ditemukan, menginstal sekarang...")
    os.system(
        "pip install numpy==1.24.3 face-recognition-models
    face_recognition==1.3.0 tqdm==4.67.0 pickle-mixin==1.0.2
    opencv-python==4.10.0.84 ultralytics==8.3.32 requests==2.32.3
    pyserial==3.5 keyboard==0.13.5")

    from tqdm import tqdm
```

```
    import pickle
    import cv2
    from ultralytics import YOLO
    import numpy as np
    import face_recognition  # Coba impor kembali setelah instalasi
    import json
    import serial
    import serial.tools.list_ports
    from datetime import datetime
    from datetime import date

# Inisialisasi timer
timer_start = None
timer1 = 0
timer2 = 0
nama = ""
id = ""
sekali_kirim = 0
elapsed_time = 0
remaining_time = 0
start_encoding = 0

# Mendapatkan folder saat ini
current_directory = os.path.dirname(os.path.abspath(__file__))

print("Tanggal :", str(date.today()))
print("Folder saat ini:", current_directory)

################################################################
# Default config
```

```
config_path = os.path.join(current_directory, 'config.json')
default_config = {
"kamera" : 0,
"timer_verifikasi" : 5,  # Timer countdown dalam detik
"folder_wajah" : "dataset_wajah",
"model_yolo" : "model_yolov8.pt",
"toleransi" : 0.5 , # semakin kecil semakin detail perbedaan wajah, tapi sensitifitas berkurang,
"timer" : 5,  # absensi dimulai setelah wajah terdeteksi selama 5 detik
"baudrate": 115200,
"encoding_path" : "encoding_wajah.bin"
}

########################################################################
# jangan diubah, jgn dihapus
url = "https://localhost.scode.web.id/2025-rivaldo-security-pintu-facerecognition/api/api.php"
url_absen = "https://localhost.scode.web.id/2025-rivaldo-security-pintu-facerecognition/api/api.php?id_wajah="

# Cek dan buat file config jika belum ada
if not os.path.exists(config_path):
    with open(config_path, 'w') as file:
        json.dump(default_config, file, indent=4)

# Baca config
with open(config_path, 'r') as file:
    config = json.load(file)

data_dataset = []
```

```
89. data_semua = {}
90. data_nama = []
91. status_nama = []
92. konfirmasi_nama = []
93. # Load YOLO model
94. yolo_model =
    YOLO(os.path.join(current_directory,config["model_yolo"]))
95. # Open the camera
96. cam = cv2.VideoCapture(config["kamera"])
97. timer_verifikasi = config["timer_verifikasi"]
98. durasi_deteksi = config["durasi_deteksi"]
99. folder_wajah = config["folder_wajah"]
100.        toleransi = config["toleransi"]
101.        timer = config["timer"]
102.        ports = serial.tools.list_ports.comports()
103.        baud_rate = config["baudrate"]
104.        encoding_path =
    os.path.join(current_directory,config["encoding_path"])
105.

106.        # Check if the camera is opened successfully
107.
108.        if not cam.isOpened():
109.            while (1):
110.                print("Error: Could not open camera.")
111.                print("Periksa nomor kamera / sambungan kamera ...
    ")
112.                time.sleep(1)
113.        else:
114.            print("Camera opened successfully.")
115.            response = requests.get(url_absen)
116.
```

```
117.        def proses_kirim_serial(pesan):
118.            pesan = pesan +"\n"
119.            global ser
120.            if pesan == "inisiasi\n":
121.                try:
122.                    print("MENCARI PORT OTOMATIS ..... ")
123.                    print(ports)
124.                    gagal = 0
125.                    for k in ports:
126.                        try:
127.                            print("Mencoba port", k)
128.                            k = str(k).split(" ")[0]
129.                            ser = serial.Serial(k, baud_rate, timeout=0.5,
    write_timeout=0.5)
130.                            print(f"Terhubung ke {k} dengan baud rate
    {baud_rate}")
131.                            time.sleep(1)
132.                            break
133.                        except:
134.                            print("Gagal Terhubung", k)
135.                            gagal += 1
136.                            time.sleep(0.1)
137.
138.                    if len(ports) == 0:
139.                        print("TIDAK ADA PORT SERIAL")
140.
141.                    if gagal == len(ports):
142.                        print("Port terdeteksi:", ports)
143.                        print("Pastikan port tidak dibuka di aplikasi lain
    !!")
144.                        port_serial = input("Masukkan nama port serial
    yang benar: ")
```

```
145.                     ser = serial.Serial(port_serial, baud_rate,
    timeout=0.5, write_timeout=0.5)
146.
147.                 time.sleep(2)  # tunggu koneksi stabil
148.             except serial.SerialException as e:
149.                 print(f"Gagal membuka port serial: {e}")
150.
151.         elif pesan == "get_sensor\n":
152.             ser.write(pesan.encode())
153.             print("Mengirim permintaan sensor:", pesan)
154.             data = ""
155.             while (1):
156.                 ser.write(pesan.encode())
157.                 time.sleep(0.1)
158.                 data = ser.readline().decode('utf-8',
    errors='ignore').strip()
159.                 if (len(data) > 0):
160.                     break
161.
162.
163.
164.             data = data.strip()
165.             print("Data Sensor Diterima:", data)
166.             return data
167.
168.         else:
169.             ser.write(pesan.encode())
170.             print("Mengirim pesan serial:", pesan.encode())
171.

172.     def web(message):
173.         global url_absen
```

```
174.            response = requests.get(url_absen+message)
175.
176.            print(response)
177.            time.sleep(2)
178.
179.        def hapus_semua_dalam_folder(folder_path):
180.            # Membaca data dataset yang ada
181.            if not os.path.exists(folder_wajah):
182.                os.mkdir(folder_wajah)
183.
184.            for item in os.listdir(folder_path):
185.                item_path = os.path.join(folder_path, item)
186.                try:
187.                    if os.path.isfile(item_path) or
    os.path.islink(item_path):
188.                        os.remove(item_path)  # Hapus file atau symlink
189.                    elif os.path.isdir(item_path):
190.                        hapus_semua_dalam_folder(item_path)  # Hapus
    isi subfolder dulu
191.                        os.rmdir(item_path)  # Hapus subfolder setelah
    kosong
192.                except Exception as e:
193.                    print(f"Gagal menghapus {item_path}: {e}")
194.
195.        def image_manager():
196.            global url, data_dataset, folder_wajah, start_encoding
197.            print("image manager")
198.
199.            folder_dummy = "dummy"
200.
201.            # Buat folder dummy jika belum ada
202.            if not os.path.exists(folder_dummy):
```

```
203.             os.mkdir(folder_dummy)
204.
205.         # Ambil data orang dari server
206.         respon = requests.get(url=url)
207.         print(respon.text)
208.
209.         # Proses setiap orang dan simpan ke folder dummy
210.         for orang in eval(respon.text):
211.             id = orang["id"]
212.             nama = orang["nama"]
213.
214.             nama_folder_dummy = os.path.join(folder_dummy, id
    + "_" + nama)
215.             os.makedirs(nama_folder_dummy, exist_ok=True)
216.
217.             for i in range(10):
218.                 url_gambar = orang["foto" + str(i + 1)]
219.                 url_gambar = url_gambar.replace("\\",
    "").replace("////", "//")
220.                 print("URL : ", url_gambar)
221.
222.                 try:
223.                     gambar_respon = requests.get(url_gambar)
224.                     if gambar_respon.status_code == 200:
225.                         path_dummy = os.path.join(
226.                             nama_folder_dummy, nama + "_" + str(i +
    1) + ".jpg")
227.                         with open(path_dummy, "wb") as f:
228.                             f.write(gambar_respon.content)
229.                     else:
230.                         print(f"Gagal mengunduh gambar dari
    {url_gambar}")
```

```
                except Exception as e:
                    print(f"Error saat mengunduh: {e}")

        # Hitung total ukuran file di folder dummy
        def total_ukuran_folder(path):
            total = 0
            for root, dirs, files in os.walk(path):
                for file in files:
                    full_path = os.path.join(root, file)
                    total += os.path.getsize(full_path)
            return total

        ukuran_dummy = total_ukuran_folder(folder_dummy)
        ukuran_wajah = total_ukuran_folder(folder_wajah) if
    os.path.exists(folder_wajah) else 0

        print(f"Ukuran dummy: {ukuran_dummy / 1024:.2f}
    KB")
        print(f"Ukuran wajah: {ukuran_wajah / 1024:.2f} KB")

        # Bandingkan, jika beda maka ganti isi folder wajah
        if ukuran_dummy != ukuran_wajah:
            print("Ukuran berbeda, mengganti isi folder wajah")
            start_encoding = 1
            # Hapus semua isi folder wajah
            if os.path.exists(folder_wajah):
                shutil.rmtree(folder_wajah)
            os.makedirs(folder_wajah, exist_ok=True)

            # Pindahkan isi folder dummy ke folder wajah
            for nama_subfolder in os.listdir(folder_dummy):
                asal = os.path.join(folder_dummy, nama_subfolder)
```

```
                tujuan = os.path.join(folder_wajah,
    nama_subfolder)
                shutil.move(asal, tujuan)

        # Hapus folder dummy
        shutil.rmtree(folder_dummy)

        # Perbarui data_dataset
        data_dataset.clear()
        for k in os.listdir(folder_wajah):
            for l in os.listdir(os.path.join(folder_wajah, k)):
                data_dataset.append(l)

        print("Selesai. Data dataset:", data_dataset)

    def encoding_wajah():
        global data_nama, data_semua, folder_wajah,
    yolo_model, encoding_path
        print("Mulai proses Encoding (perlu waktu tergantung
    jumlah foto)")

        for nama in os.listdir(folder_wajah):
            poto = []
            sub_folder = os.path.join(folder_wajah, nama)

            if os.path.isdir(sub_folder):
                for b in tqdm(os.listdir(sub_folder)):
                    path_image = os.path.join(sub_folder, b)

                    if os.path.isfile(path_image):
                        frame = cv2.imread(path_image)
                        results = yolo_model(frame)
```

```
290.
291.                     for r in results:
292.                         boxes = r.boxes.xyxy.tolist()
293.                         for box in boxes:
294.                             x1, y1, x2, y2 = map(int, box[:4])
295.
296.                             # Simpan kotak dari YOLO pada frame
    asli
297.                             cv2.rectangle(frame, (x1, y1), (x2, y2), (0,
    255, 0), 2)  # Hijau: YOLO
298.
299.                             # Deteksi wajah pada hasil crop
300.                             face_locations =
    face_recognition.face_locations(frame)
301.                             face_encodings =
    face_recognition.face_encodings(frame, face_locations)
302.
303.                             for (top, right, bottom, left) in
    face_locations:
304.                                 # Gambar kotak wajah dari
    face_recognition (biru)
305.                                 cv2.rectangle(frame, (left, top), (right,
    bottom), (255, 0, 0), 2)
306.
307.                             # Resize hasil crop untuk ditampilkan
    (tetap 480x480)
308.                             display_frame = cv2.resize(frame, (480,
    480))
309.                             cv2.imshow("Deteksi Wajah",
    display_frame)
310.                             cv2.waitKey(500)
311.
```

```
312.                             if face_encodings:
313.                                 poto.append(face_encodings[0])
314.
315.             if poto:
316.                 data_semua[nama] = poto
317.                 print(f"{nama} selesai.")
318.             else:
319.                 print("WAJAH TIDAK ADA PADA GAMBAR.")
320.                 while True:
321.                     pass
322.
323.         with open(encoding_path, "wb") as f:
324.             pickle.dump(data_semua, f)
325.         print("Data encoding disimpan dalam file
    encoding_wajah.bin")
326.         cv2.destroyAllWindows()
327.
328.     def inisiasi():
329.         global data_semua, data_nama,
    folder_wajah,start_encoding
330.
331.         print(" >>>>>  Jika ada perubahan pada folder nama /
    foto, wajib menghapus file encoding_wajah.bin <<<<<  ")
332.         time.sleep(3)
333.         image_manager()
334.
335.         try:
336.             with open(encoding_path, "rb") as f:
337.                 print("Memuat data encoding sebelumnya .. ")
338.                 data_semua = pickle.load(f)
339.                 f.close()
340.         except:
```

```
341.                print("Gagal memuat data encoding sebelumnya,
    membuat file baru ...")
342.                encoding_wajah()
343.
344.            data_nama = list(data_semua.keys())
345.            print("Data nama : ", data_nama)
346.
347.            if data_nama != os.listdir(folder_wajah) or
    start_encoding == 1 :
348.                print("perubahan data terdeteksi, membuat encoding
    baru ... ")
349.                os.remove(encoding_path)
350.                data_nama = []
351.                data_semua = {}
352.                encoding_wajah()
353.                with open(encoding_path, "rb") as f:
354.                    print("Memuat data encoding yang telah dibuat .. ")
355.                    data_semua = pickle.load(f)
356.                    f.close()
357.
358.                data_nama = list(data_semua.keys())
359.                print("Data nama : ", data_nama)
360.                start_encoding = 0
361.
362.            for l in (data_nama):
363.                status_nama.append(0)
364.                konfirmasi_nama.append(0)
365.
366.        def deteksi():
367.            global timer_start, timer_deteksi, elapsed_time,
    sekali_kirim, last_detected_name, timer_verifikasi, nama, id,
    data_nama
```

```
368.
369.         # Inisialisasi variabel
370.         timer_start = None
371.         timer_deteksi = 0
372.         elapsed_time = 0
373.         sekali_kirim = False
374.         last_detected_name = None
375.
376.         while True:
377.             # Baca frame dari kamera
378.             res, frame = cam.read()
379.             nama_terdeteksi = None
380.             data = proses_kirim_serial("get_sensor")
381.             print("data : ",data)
382.
383.             if data == "1":
384.                 timer_deteksi = time.time()
385.
386.             durasi = time.time() - timer_deteksi
387.             print( durasi)
388.             if durasi < durasi_deteksi :
389.                 cv2.putText(frame,f"DETEKSI AKTIF
    ({durasi_deteksi - int(durasi)}) ", (0, 25),
390.                             cv2.FONT_HERSHEY_SIMPLEX, 1,
    (255, 255, 0), 2)
391.                 print("DETEKSI AKTIF ... ", int(time.time() -
    timer_deteksi))
392.                 # Periksa apakah frame berhasil dibaca
393.                 if not res:
394.                     print("Error: Could not read frame.")
395.                     break
396.
```

```
397.                # Jalankan YOLO untuk deteksi
398.                results = yolo_model(frame)
399.
400.                # Loop setiap hasil deteksi
401.                if len(results) == 1:
402.                    nama_terdeteksi = "Tidak Dikenali"
403.                    nama = nama_terdeteksi
404.                    id = ""
405.                    for r in results:
406.                        boxes = r.boxes.xyxy.tolist()
407.                        for box in boxes:
408.                            # Ekstraksi koordinat bounding box
409.                            x1, y1, x2, y2 = map(int, box[:4])
410.
411.                            # Potong dan proses gambar untuk
    pengenalan wajah
412.                            if y1 >= 20:
413.                                image_rgb = frame[y1-20:y2, x1:x2]
414.                                cv2.rectangle(frame, (x1, y1-20),
415.                                        (x2, y2), (0, 255, 0), 2)
416.                            else:
417.                                image_rgb = frame[y1:y2, x1:x2]
418.                                cv2.rectangle(frame, (x1, y1),
419.                                        (x2, y2), (0, 255, 0), 2)
420.
421.                            # Penyesuaian kontras dan kecerahan
422.                            alpha = 1.5
423.                            beta = 20
424.                            image_rgb = cv2.convertScaleAbs(
425.                                image_rgb, alpha=alpha, beta=beta)
426.
```

```
                        input_face_encodings =
    face_recognition.face_encodings(
                            image_rgb)

                        koordinat = 0
                        # Cek kecocokan dengan data wajah yang
    dikenal
                        for k in data_nama:
                            konfirmasi_nama[data_nama.index(k)] =
    0
                            for l in data_semua[k]:
                                matches =
    face_recognition.compare_faces(
                                    [l], input_face_encodings[0],
    tolerance=toleransi) if input_face_encodings else []
                                if True in matches:
                                    konfirmasi_nama[data_nama.index(k
    )] += 1

                            koordinat += 10

                            probabilitas = (
                                konfirmasi_nama[data_nama.index(k)]
    / len(data_semua[k]) * 100)
                            print("Probabilitas : ", k,
                                probabilitas, "%")
                            # Tampilkan nama di frame
                            cv2.putText(frame, k.split("_")[1] + " " +
                                    str(round(probabilitas, 1)) +
                                    "%", (0, 60 + koordinat*2),
                                    cv2.FONT_HERSHEY_SIMPLE
    X, 0.5, (255, 0, 255), 2)
```

```
451.
452.                             if
    (konfirmasi_nama[np.argmax(konfirmasi_nama)] != 0):
453.                                 nama_terdeteksi =
    data_nama[np.argmax(
454.                                     konfirmasi_nama)]
455.
456.                             try:
457.                                 nama = nama_terdeteksi.split("_")[1]
458.                                 id = nama_terdeteksi.split("_")[0]
459.                             except:
460.                                 pass
461.
462.                         # Tampilkan nama di frame
463.                         cv2.putText(frame, nama, (x1, y1 - 80),
464.                                     cv2.FONT_HERSHEY_SIMPLEX,
    1, (255, 255, 255), 2)
465.                         cv2.putText(frame, id, (x1, y1 - 25),
466.                                     cv2.FONT_HERSHEY_SIMPLEX,
    1, (255, 255, 255), 2)
467.
468.                 # Logika Timer
469.                 if nama_terdeteksi != "Tidak Dikenali":
470.                     # kode untuk kontrol esp32
471.
472.
473.                     if nama_terdeteksi != last_detected_name:
474.                         # Nama baru terdeteksi, reset timer
475.                         timer_start = time.time()
476.                         sekali_kirim = False
477.                         last_detected_name = nama_terdeteksi
478.
```

```
                    elapsed_time = time.time() - timer_start
                    remaining_time = timer_verifikasi - elapsed_time
                    if remaining_time < 0:
                        remaining_time = 0

                    if remaining_time <= 0 and not sekali_kirim:
                        print("..... MENGIRIM ABSEN .....")
                        proses_kirim_serial("@"+nama)
                        sekali_kirim = True
                        cv2.putText(frame, "MENGIRIM ABSEN ...", (0, 100),
                                    cv2.FONT_HERSHEY_SIMPLEX, 1, (255, 255, 0), 2)
                        web(id)

                else:
                    # Reset jika tidak ada nama yang terdeteksi
                    timer_start = None
                    elapsed_time = 0
                    sekali_kirim = False
                    last_detected_name = None

                # Tampilkan timer di frame
                if timer_start:
                    cv2.putText(frame, f"Status : ({round(remaining_time, 1)}s)", (0, 50),
                                cv2.FONT_HERSHEY_SIMPLEX, 1, (255, 255, 0), 2)

            # Tampilkan frame
            cv2.imshow('YOLO Detection', frame)
```

```
506.
507.             # Tombol keluar
508.             if cv2.waitKey(1) & 0xFF == ord('q'):
509.                 break
510.
511.         # Bersihkan sumber daya
512.         cam.release()
513.         cv2.destroyAllWindows()
514.
515.     if __name__ == "__main__":
516.         inisiasi()
517.         proses_kirim_serial("inisiasi")
518.         deteksi()
519.
```

1.3.3 Euclidean Distance

metode euclidean distance untuk mendapatka jarak terkecil antara bobot orang masuk yang fungsinya utuk mengenali wajah orang masuk. Berikut ini merupakan sourcecode dari euclidean distance : selisih = np.asarray(w - w_in) dst = np.sqrt(np.Num(selisih**2, axis=1) kecil = min(dst) “kecil” merupakan nilai terkecil dari euclidean distance, yang nantinya akan dibandingkan dengan nilai threshold jika sistem membaca maka wajah dianggap cocok maka dianggap wajah sudah di input dan kedetek