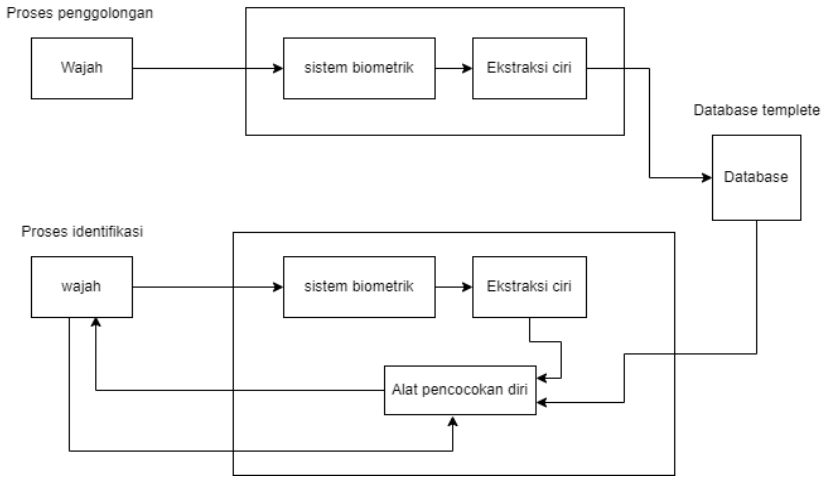
BAB II
TINJAUAN PUSTAKA

2.1 Sistem Biometrika

Sistem biometrik merupakan teknologi pengenalan diri yang menggunakan bagian tubuh manusia. Sistem pengenalan diri adalah sistem yang menggunakan teknologi komputer untuk mengidentifikasi seseorang secara otomatis. Sistem akan mencari dan mencocokan identitas seseorang dengan suatu data acuan yang telah disiapkan sebelumnya. Mekanis sistem biometrik dapat dapat digambarkan dalam beberapa fase, seperti pada gambar 2.1.



Gambar 2.1 Sistem Biometrik

Pada Gambar 2.1, dapat dilihat fase pertama adalah fase penggolongan (*enrolment).* Pada fase penggolongan masukan wajah akan dipindai (*scan)* oleh sensor biometrik, yang merupakan representasi karakteristik digital. Selanjutnya fase pencocokan, dalam fase ini masukan basis data akan dicocokkan dengan identifikasi data. Dapat dimungkinkan adanya reduksi, sehingga dihasilkan representasi digital. Hasil ini akan diproses dengan ekstraktor ciri untuk menghasilkan suatu representasi yang ekspresif dalam bentuk template. Sedangkan pada fase pengenalan, karakteristik individu dibaca oleh pembaca. Selanjutnya dikonversi dengan format digital, untuk diproses sebagai ekstraktor ciri. Hasil ciri ini selanjutnya dicocokan dengan identifikasi individu.

2.2 Pengolahan Citra

Pengolahan citra adalah salah satu bentuk pemrosesan informasi dengan inputan berupa citra dan juga output yang berupa citra atau bagian yang termasuk citra tersebut. Salah satu tujuan pemrosesan ini untuk memperbaiki kualitas citra agar lebih mudah diinterpretasi oleh manusia dan juga pemrosesan ini juga bisa digunakan sebagai perbandingan antara sebuah citra dengan citra yang lain [6]. Operasi pada pengolahan citra secara umum dapat diklasifikasikan sebagai berikut:

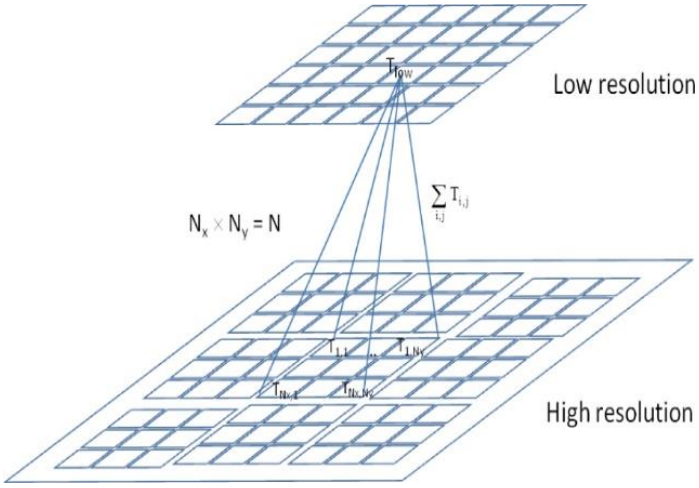
1. Perbaikan kualitas citra *(image enhancement),* contohnya perbaikan pencahayaan antara gelap/terangnya citra.
2. Restorasi citra (*image restoration*) contohnya penghilau derau dan kesamaran (*deblurring*).
3. Pemanfaatan citra (*image compression*).
4. Segmentasi citra (*image segmentation*).
5. Pengorakan citra (*image analysis*).
6. Rekonstruksi citra (*image reconstruction*).

2.3 Citra *Grayscale*

Proses yang banyak dilakukan dalam *image processing* adalah mengubah citra berwarna menjadi citra *grayscale*. Hal ini dilakukan dengan menyederhanakan model citra berwarna menjadi citra (RGB) yang terdiri dari tiga layer matrik yaitu R-*layer,* G-*layer* dan B-*layer* [7]. Citra *grayscale* mempunyai kemungkinan warna hitam untuk nilai minimal dan warna putih untuk nilai maksimal. Banyaknya warna tergantung pada jumlah bit yang disediakan di memori untuk menampung kebutuhan warna tersebut. Semakin besar jumlah bit warna yang disediakan di memori, maka semakin halus gradasi warna yang terbentuk [5].

2.4 *Downscaling*

Teknik *downscaling* adalah suatu proses transformasi pada suatu citra dari suatu grid atau suatu piksel dari skala yang besar menjadi skala yang lebih kecil [8]. Pada gambar 2.2 adalah ilustrasi dari proses *downscaling.*



Gambar 2.2 *Downscaling*

2.5 Pengenalan Wajah

Pengenalan wajah adalah suatu metode pengenalan yang berpusat pada wajah. Pengenalan ini dapat dibagi menjadi dua bagian yaitu dikenali dan tidak dikenali pada sebuah citra yang terdapat didalam database [3]. Secara umum sistem pengenalan wajah dibagi menjadi dua jenis, yaitu sistem *feature based* dan sistem *image based*. Sistem yang diekstraksi dari komponen wajah terdiri dari (mata, hidung, mulut, dan lain-lain) yang kemudian hubungan antara fitur tersebut dimodelkan secara geometris. Sedangkan sistem kedua yaitu menggunakan informasi mentah dari piksel citra yang kemudian direpresentasikan dalam metode tertentu [6].

2.6 *Principal Component Analysis*

Menggunakan wajah sebagai pengenalan memiliki banyak keuntungan, termasuk kepraktisannya dalam mengidentifikasi sebuah citra. Masalah utamanya adalah sebuah citra yang mewakili gambar yang terdiri dari vektor yang berukuran relatif besar. Ada banyak teknik untuk mereduksi dimensi dari citra yang akan diproses salah satunya algoritma *Eigenface*. *Eigenface* merupakan algoritma yang didasarkan pada PCA **[**5].

Metode *Eigenface* adalah hasil transformasi citra-citra wajah menjadi sebuah set fitur karakteristik wajah dalam bentuk *eigenvector* dari matriks kovarian citra-citra wajah tersebut [9]. Untuk menghasilkan *Eigenface,* sekumpulan citra digital

dari wajah manusia diambil pada kondisi pencahayaan yang sama kemudian dinormalisasi dan diproses pada resolusi yang sama misalnya citra berukuran N = $w \times h$, kemudian citra tersebut diperlakukan sebagai vektor dimensi $w \times h$ dimana komponennya diambil dari nilai piksel citra [3].

Berikut representasi citra menggunakan PCA untuk memperoleh bobot dari citra.

1. Membuat citra dengan dimensi $w \times h$ pada citra wajah I

   Keterangan:

   w = *Lebar citra*

   h = *Tinggi citra*

   I = *Citra pada suatu kelas tertentu*

2. Membaca nilai dari citra wajah $I = (I_1, I_2, I_3, \ldots, I_M)$

3. Mengubah citra wajah menjadi vektor berukuran $N \times 1$, contoh:

   $I_1 = \begin{bmatrix} \boldsymbol{a1} \\ \boldsymbol{a2} \\ \vdots \\ \boldsymbol{aN} \end{bmatrix}$, $I_2 = \begin{bmatrix} \boldsymbol{b1} \\ \boldsymbol{b2} \\ \vdots \\ \boldsymbol{bN} \end{bmatrix}$ dan seterusnya sampai I_M.

   $\boldsymbol{\Gamma}_i = I_1 + I_2 + I_3 + \ldots + I_M$

   Keterangan:

   M = *banyaknya citra pada dataset*

   N = *banyaknya piksel pada sebuah citra*

4. Menghitung rata rata matriks

$$\boldsymbol{\Psi} = \frac{\mathbf{1}}{\boldsymbol{M}} \sum_{\boldsymbol{i}=\mathbf{1}}^{\boldsymbol{M}} \boldsymbol{\Gamma}_{\mathbf{1}} \tag{2.1}$$

5. Setelah mendapatkan nilai rata rata, maka matriks citra wajah dikurangi matriks rata-rata wajah.

$$\phi_i = \Gamma_i - \boldsymbol{\Psi} \tag{2.2}$$

6. Menghitung *matrix covarian*

$$\boldsymbol{C} = \boldsymbol{\phi}\ \boldsymbol{\phi}^{\mathbf{T}} \tag{2.3}$$

dimana $\phi = \begin{bmatrix} \phi\mathbf{1} \\ \phi\mathbf{2} \\ \vdots \\ \phi\boldsymbol{N} \end{bmatrix}$

7. Menghitung nilai eigen (λ) dan vektor eigen (V)

$$\begin{gathered} \boldsymbol{C}\ \boldsymbol{V} = \lambda\ \boldsymbol{V} \\ \mathbf{(C - \lambda I)}\ \boldsymbol{V} = \mathbf{0} \\ \mathbf{(\Phi\ \Phi^{T} - \lambda I)}\ \boldsymbol{V} = \mathbf{0} \end{gathered} \tag{2.4}$$

8. Mencari nilai *Eigenface* (μ)

$$\mu = \sum_{\boldsymbol{i=1}}^{\boldsymbol{M}} \boldsymbol{v}\phi_i \tag{2.5}$$

9. Citra wajah diklasifikasikan dengan memproyeksikan wajah ke ruang wajah dengan persamaan

$$\omega_{\mathbf{k}} = \mu_{\mathbf{i}}^{\mathbf{T}}(\Gamma_{\boldsymbol{i}} - \boldsymbol{\Psi}) \tag{2.6}$$

Bobot membentuk $\Omega_{\boldsymbol{i}}^{\boldsymbol{T}} = [\omega_{\mathbf{1}},\ \omega_{\mathbf{1}}, \ldots . \omega_{\boldsymbol{n}},]$, yang berisi proyeksi ke setiap vektor eigen. Klasifikasi dilakukan dengan menghitung jarak $\Omega_{\boldsymbol{i}}$ dari Ω, di mana Ω merupakan vektor bobot yang mendefinisikan beberapa kelas.

2.7 K - *Nearest Neighbor*

Pada metode klasifikasi KNN, citra masukan akan diuji berdasarkan jarak fiturnya dengan fitur citra lain didalam basis data untuk memperoleh satu nilai citra dengan jarak fitur paling minimum atau bisa disebut dengan ketetanggan terdekat.

Pada metode klasifikasi KNN, didapatkan satu citra yang minimum dengan citra uji. Sedangkan pada metode klasifikasi dihasilkan citra sampel sebanyak *k* yang memiliki jarak minimum dari citra uji. Citra uji akan diklasifikasikan ke dalam suatu kelas dengan jumlah citra sample sejumlah *k*. Nilai *k* pada KNN ini mempengaruhi performansi dari metode KNN tersebut.

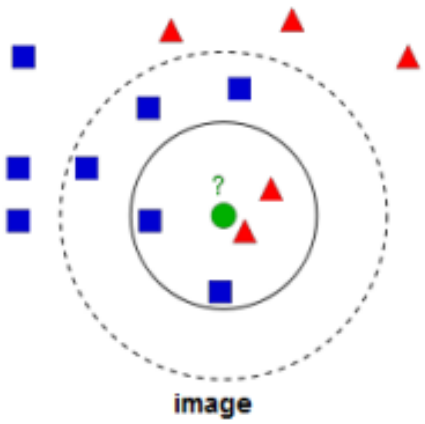
Metode KNN melakukan pencocokan atau pengenalan berdasarkan jumlah tetangga terdekat untuk menentukan kelasnya. Untuk mencari jarak kelas menggunakan perhitungan jarak *Euclidean Distance*. Tahapan dalam metode klasifikasi KNN yaitu [3]:

a. Menentukan nilai *k*

b. Menghitung jarak antara citra uji dengan seluruh citra pada basis data menggunakan persamaan *euclidean distance*, dan menentukan citra terdekat dengan citra uji berdasarkan nilai *k*.

$$\boldsymbol{d_i} = \sqrt{\sum_{\boldsymbol{i=1}}^{\boldsymbol{p}} (\boldsymbol{x_{2i}} - \boldsymbol{x_{1i}})^2} \tag{2.7}$$

c. Menentukan hasil klasifikasi berdasarkan kelas yang memiliki anggota terbanyak.



Gambar 2.3 Klasifikasi KNN

Gambar 2.2 merupakan ilustrasi sistem proses KNN, contoh seperti pada gambar data diklasifikasikan dalam dua klasifikasi biru dan merah. Data baru berwarna hijau dicek klasifikasinya berdasarkan jarak terdekat dan nilai *k* yang digunakan. Jika *k* = 3 maka terlihat data baru yang paling dekat dan paling banyak adalah kelas warna merah [10].

2.8 Python

Python di kembangkan oleh Guildo Van Rossum, programmer asal Belanda, pada tahun 1991 di Amsterdam, sebagai kelanjutan dari bahasa pemrograman ABC. Python adalah bahasa pemrograman interpretatif yang dianggap mudah dipelajari serta berfokus pada keterbacaan kode. Python secara umum berbentuk pemrograman berorientasi objek, pemrograman imperatif, dan pemrograman fungsional [7]. Pustaka yang digunakan dalam perancangan program sistem pengenalan wajah diantaranya adalah OpenCV, Matplotlib, Numpy, Pandas, Random, Time dan Math.

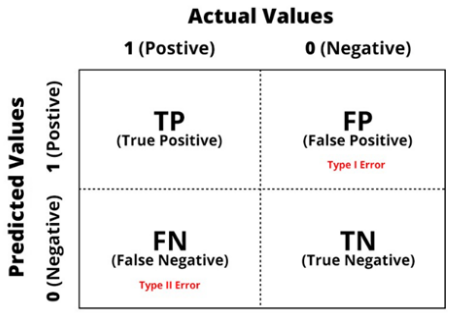
2.8.1 OpenCV

OpenCV adalah sebuah pustaka yang bersifat *open source* yang dikembangkan oleh Intel yang fokus untuk menyederhanakan pemrograman terkait dengan pengolahan citra digital. OpenCV memiliki banyak fitur terkait visi komputer *(computer vision)* antara lain: pengenalan wajah, deteksi wajah, Kalman filtering, dan berbagai jenis metode *Artificial Intelligence* (AI). OpenCV dapat berjalan di berbagai bahasa pemrograman, seperti C, C++, Java, Python, dan juga mendukung berbagai platform sistem operasi seperti Windows, Linux, Mac OS, iOS dan Android [11].

2.9 *Confusion Matrix*

Pada dasarnya *confusion matrix* memberikan informasi tentang perbandingan hasil klasifikasi yang dilakukan oleh sistem (model) dengan hasil klasifikasi yang sebenarnya. *confusion matrix* dalam bentuk tabel menggambarkan kinerja model klasifikasi pada serangkaian data uji dengan nilai nyata yang diketahui. Pada

gambar 2.4 merupakan gambar *confusion matrix* dengan 4 kombinasi nilai prediksi dan nilai aktual yang berbeda.



Gambar 2.4 *Confusion Matrix*

Terdapat 4 istilah sebagai representasi hasil proses klasifikasi pada *confusion matrix*, yaitu sebagai berikut:

1. *True Positive* (TP) adalah kondisi dimana data positif yang telah diprediksi dengan hasil benar dikenali.
2. *False Positive* (FP) adalah kondisi dimana data citra tidak didalam dataset namun bisa dikenali.
3. *True Negative* (TN) adalah kondisi dimana data citra benar ada, namun salah dikenali
4. *False Negative* (FN) adalah Kondisi dimana citra tidak ada didalam data set dan tidak berhasil dikenali

Berikut adalah rumus *confusion matrix*. untuk mencari nilai dari akurasi, presisi dan *recall* [12].

$$\boldsymbol{akurasi} = \frac{\boldsymbol{TP} + \boldsymbol{TN}}{\boldsymbol{Total}} \tag{2.8}$$

$$\boldsymbol{presisi} = \frac{\boldsymbol{TP}}{\boldsymbol{TP} + \boldsymbol{FP}} \tag{2.9}$$

$$\boldsymbol{recall} = \frac{\boldsymbol{TP}}{\boldsymbol{TP} + \boldsymbol{FN}} \tag{2.10}$$

2.10 Waktu Komputasi

Waktu komputasi adalah sebuah kondisi lamanya pengoprasian yang dibutuhkan pada suatu sistem. Untuk mendapatkan waktu komputasi dibutuhkan perhitungan untuk mendapatkan nilai waktu yang dibutuhkan. Maka dari itu berikut adalah parameter persamaan untuk mencari waktu komputasi.

$$\boldsymbol{W_k = W_s - W_m} \tag{2.11}$$

Keterangan:

W_k = waktu komputasi (detik)

W_s = waktu selesai (detik)

W_m = waktu mulai (detik)