Pengenalan Pola Wajah Menggunakan Metode *Block-Eigenface* pada Raspberry Pi

(Face Recognition Using Block-Eigenface on Raspberry Pi)

Arbhi Anggara*, I Wayan Agus Arimbawa
Dept Informatics Engineering, Mataram University
Jl. Majapahit 62, Mataram, Lombok NTB, INDONESIA
Email: arbhianggara24@gmail.com, arimbawa@unram.ac.id

Penulis korespondensi

Abstract **Rapid technological development has created a human face recognition system. The use of this facial recognition system spread to various fields. One of them is the security field that uses this system for authentication which is usually applied to Internet of Things based systems. The Raspberry Pi microprocessor is used as a place to implement a face recognition system to fulfil the IoT requirements. This facial recognition system measures qualitatively the characteristics of the human face, such as the shape of the face, eyes, nose, and mouth. There are many methods for measuring facial characteristics. One of them is the Block-Eigenface method which is a derivative of the Eigenface method. This method is a development method that applies *blocking* to the original method. Two datasets were used in this study, the first was a dataset from the results of data collection with 100 images of 10 classes (dataset Informatics), and the second dataset was commonly used was the ORL dataset with 400 images of 40 classes. The blocking done has an effect on the accuracy and computational time of the original method. In the Informatics dataset the Eigenface method got 84% results with a classification time of 0.003 seconds, the Block-Eigenface method got an accuracy of 98% with a classification time of 0.094 seconds. The ORL dataset provided an average accuracy of 55.42% with an average classification time of 0.014 seconds on the Eigenface method and 82.22% with an average classification time of 0.019 seconds on Block-Eigenface.**

***Key words*: Face Recognition, Raspberry Pi, Eigenface, Block-Eigenface, Internet of Things.**

I. PENDAHULUAN

Secara visual wajah merupakan bagian dari tubuh manusia yang paling mudah untuk dikenali oleh orang lain. Hal ini karena wajah manusia memiliki ciri tersendiri seperti mata, hidung, mulut, bentuk wajah dan sebagainya. Dampak perkembangan teknologi menyebabkan bukan hanya manusia yang dapat mengenali wajah, namun kini komputer dengan program pengenalan wajah juga dapat melakukannya. Menggunakan *Biometric-based techniques* yakni sebuah teknik yang menggunakan bentuk fisik manusia (karakteristik) untuk diukur dan dihitung hingga mendapatkan informasi yang dibutuhkan [1].

Untuk dapat mengenali wajah seseorang, ada dua proses utama yang dilakukan, yaitu, ekstraksi fitur dan klasifikasi. Sistem pengenalan pola mengambil karakteristik wajah, kemudian diukur secara kuantitatif dengan menggunakan pengolahan citra dan penggunaan metode atau algoritma tertentu untuk mendapatkan ekstraksi ciri dari wajah tersebut [2].

Terdapat banyak metode untuk ekstraksi fitur yang diterapkan pada pengenalan pola wajah, salah satunya adalah metode yang diusulkan dalam penilitian ini yaitu metode *Block-Eigenface*. Metode ini didasari oleh algoritma *Eigenface* yang efektif digunakan untuk pengenalan wajah. Teknik *blocking* bukan hal baru pada dunia pengenalan pola. Teknik ini adalah sebuah proses di mana citra dibagi menjadi beberapa bagian untuk mendapatkan detail fitur dari citra tersebut. Penerapannya telah dilakukan pada banyak metode, namun tidak dengan Pada penelitian ini, akan ditambahkan proses *blocking* pada algoritma *Eigenface* sehingga namanya menjadi *Block-Eigenface*.

Metode *Eigenface* memproses suatu citra wajah secara utuh (melakukan pendekatan holistik), namun pada penelitian ini akan dilakukan pembagian atau *blocking* pada setiap citra wajah yang akan diproses. *Blocking* yang dilakukan bertujuan untuk melokalisasi fitur-fitur wajah atau menemukan *local descriptor* dari blok yang ditinjau.

Penelitian metode *Block-Eigenface* ini dilakukan untuk mengetahui tingkat optimal untuk jumlah *blocking* yang diterapkan pada program, waktu komputasi, dan akurasi yang dibandingkan dengan metode aslinya yakni *Eigenface*. Pengimplementasian metode ini akan dilakukan pada Raspberry Pi. Penggunaan Raspberry Pi pada penelitian ini untuk memberikan gambaran, bagaimana performa metode ini jika diterapkan pada sistem berbasis IoT yang kini banyak dikembangkan.

II. TINJAUAN PUSTAKA DAN TEORI PENDUKUNG

A. Tinjauan Pustaka

Teknik biometrik (*Biometric-based techniques*) dibagi menjadi dua kategori yaitu berdasarkan perilaku dan

berdasarkan fisiologis. Diantara tipe fisiologis, pengenalan wajah hingga saat ini masih menjadi bahan penelitian sejak tahun 1960 dengan cakupan yang luas untuk terus dilakukan penyempumaan [3].

Pengenalan wajah yang memiliki sejarah panjang, tentu telah banyak melahirkan metode-metode pada tiap penelitian. Diantaranya terdapat metode *Convolutional Neural Network* (CNN), metode LDA, dan metode PCA. Metode yang dikembangkan pada tiap penelitian diharapkan dapat bekerja lebih optimal dari sebelumnya.

Sebuah penelitian pengenalan wajah dengan metode CNN menggunakan 4 *dataset*. Jumlah total citra yang digunakan yaitu 126 citra dari semua *dataset*. Variasi intensitas pencahayaan dilakukan pada setiap *dataset*. Hasil akurasi rata-rata dari setiap *dataset* yang diuji yaitu 89% [3].

Sebuah penelitian dengan metode LDA dan KNN dilakukan dengan menggunakan 66 citra wajah dari 22 subjek. Nilai akurasi rata-rata yang didapat pada penelitian ini adalah 98,33% [4]. Penelitian ini telah menghasilkan nilai akurasi yang cukup tinggi. Namun, jika ingin melakukan penelitian dengan metode serupa, yaitu LDA, maka disarankan untuk menggunakan metode klasifikasi yang berbeda sebagai pembandingnya.

Penelitian pengenalan wajah dengan metode *Eigenface* diimplementasikan ke dalam Raspberry Pi mendapatkan EER sebesar 13,33% [6]. Digunakan 9 citra untuk masing-masing subjek yang berjumlah 5 orang. Penelitian ini dengan metode yang digunakan masih mendapatkan akurasi di bawah 90% yang masih dapat ditingkatkan.

Sebuah sistem presensi berbasis metode *Eigenface* memanfaatkan prinsip-prinsip PCA [7]. Penilitian dengan menguji ekspresi, aksesoris, dan jarak subjek mendapatkan akurasi rata-rata sebesar 69,33%. Akurasi yang didapat cukup rendah dengan parameter pengujian yang bervariasi.

Penelitian pengenalan wajah dengan metode *Eigenface* untuk pengenalan wajah [8]. Hasil pengujian memberikan angka akurasi 88% untuk database berisi 10 data wajah dan akurasi 52% untuk *database* dengan jumlah 20 data wajah.

Dari hasil penelitian ekstraksi ciri wajah manusia dengan menggunakan metode PCA berhasil didapatkan nilai akurasi mencapai 98% [2]. Metode PCA merupakan metode yang mendasari munculnya metode *Eigenface*, sehingga dapat digunakan sebagai sebuah referensi dalam penelitian ini. Terdapat penelitian yang menggunakan metode yang sama yaitu PCA yang disandingkan dengan KNN. Perbandingan nilai terkecil dicari menggunakan *Nearest Neighbor*. Pengujian menghasilkan nilai keberhasilan 82,81% terhadap wajah dengan ekspresi senyum dan tanpa ekspresi pada 8 orang dan 16 wajah [9].

Dari beberapa penelitian-peneliatian sebelumnya, metode *Eigenface* atau PCA sudah banyak digunakan dan beberapa telah mencapai angka akurasi yang sangat tinggi. Namun penerapan metode ini masih sekedar menggunakan algoritma aslinya saja. *Blocking* sebenarnya tidak asing pada penelitian tentang pengenalan pola. Penerapannya sering kali dilakukan pada metode baru yang bertujuan menambah detail fitur wajah yang dihitung. Pada metode *Eigenface* hal tersebut ingin dilakukan untuk mengetahui apakah pengaruhnya terhadap akurasi yang akan diperoleh dan waktu komputasi yang digunakan.

III. METODE PENILITIAN

A. Alat dan Bahan

Pada penelitian pengenalan wajah menggunakan metode *Block-Eigenface* ini disiapkan alat dan bahan yang dibutuhkan dalam membangun sistemnya, yaitu :

1. Komputer/Laptop
   Pada penelitian ini hampir seluruh prosesnya dilakukan pada komputer, baik itu tahapan merancang maupun membangun sistem. Selain itu komputer juga digunakan saat melakukan pengujian pada sistem.
2. Raspberry Pi
   Komponen yang satu ini akan menjadi tempat pengujian terhadap metode yang dikembangkan. Tipe yang digunakan yaitu Raspberry Pi 3 B dengan *storage* 16 GB dan RAM 1 GB. Kamera webcam juga akan terpasang langsung pada mikroprosesor ini untuk melakukan pengambilan gambar.
3. *Webcam*
   Penggunaan *webcam* pada penelitian ini sebagai alat untuk pengambilan gambar wajah. Kamera yang digunakan merupakan kamera webcam dengan resolusi 1,3 megapiksel yang dipasang pada Raspberry dan komputer.
4. Data Wajah
   Dataset wajah dari 10 subjek dengan 15 foto untuk setiap subjeknya dikumpulkan untuk penelitian ini untuk *dataset* informatika dan total 400 citra untuk *dataset* lainnya yaitu ORL *dataset*. Kedua *dataset* tersebut digunakan untuk pelatihan dan pengujian.

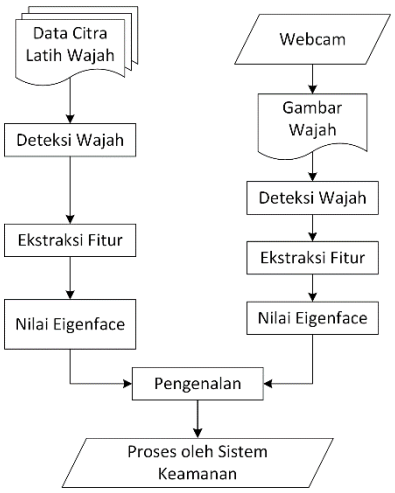
B. Prosedur Proses Penelitian

Penelitian ini dimulai dengan melakukan perencanaan terkait bagaimana sistem pengenalan wajah akan dibuat dan apa saja yang akan dibutuhkan untuk pengembangannya. Selanjutnya, mempersiapkan alat dan bahan dibutuhkan dalam sistem yang akan dibangun. Kemudian, dilakukan studi literatur untuk mempelajari metode yang akan digunakan sebagai dasar penelitian ini yaitu metode *Eigenface* yang akan dikembangkan menjadi sebuah metode baru yaitu, metode *Block-Eigenface*. Pembangunan sistem dilakukan sesuai dengan metode *Block-Eigenface* yang kemudian akan diuji dan dilakukan perbaikan untuk mendapatkan kinerja sistem yang optimal.

C. Perancangan Sistem

Penelitian ini menggunakan foto wajah sebagai objek dengan beberapa subjek berbeda. Sistem yang dibuat melakukan proses pelatihan pada komputer/laptop, sedangkan untuk proses pengujian dilakukan pada Raspberry Pi. Hal ini dilakukan karena untuk melakukan pelatihan (meliputi ekstraksi fitur untuk data latih yang cukup banyak) dibutuhkan sumberdaya yang besar.

Raspberry Pi jika dibandingkan dengan komputer tentu terpaut jauh kapabilitasnya dalam hal sumberdaya. Perbandingan tersebut menjadi dasar pembagian yang dilakukan pada penelitian ini. Oleh karena itu, hasil ekstraksi fitur yang didapat pada pelatihan di komputer kemudian di-*import* ke Raspberry Pi. Secara umum alur kinerja sistem yang dirancang pada tugas akhir ini ditunjukkan pada Gambar 1.



Gambar 1. Diagram kinerja sistem kunci otomatis

Pada Gambar 1 menunjukkan proses pelatihan, pengujian, sampai didapatkan suatu citra wajah sebagai dasar pembaharuan status. Prosesnya dimulai dari pelatihan pada bagian kiri diagram yang melakukan ekstraksi fitur terhadap kumpulan *database* citra latih wajah untuk mendapatkan nilai *Eigenface* dari citra latih wajah sebagai dasar klasifikasi. Selanjutnya, proses pengujian pada bagian kanan diagram dilakukan pengambilan citra wajah yang diujikan melalui webcam. Gambar diproses untuk mendeteksi wajah yang kemudian diesktraksi fiturnya untuk mendapatkan nilai *Eigenface*. Proses klasifikasi melakukan pencocokkan terhadap nilai *Eigenface* dari hasil ekstraksi pada pelatihan dan pengenalan yang kemudian diproses oleh Raspberry Pi untuk memberikan nilai *output*.

C.1. Pendektesian Wajah dengan Viola Jones

Citra wajah yang telah ditangkap kamera kemudian diproses dengan pengkonversian dari RGB ke *Grayscale*. Setelah mode warna citra telah berubah maka selanjutnya dilakukan pendeteksian wajah menggunakan metode Viola-Jones. Pada tahapan ini wajah yang terdeteksi pada citra kemudian di-*crop* untuk dipisahkan dari gambar aslinya yang masih memiliki banyak derau (*noise*). Citra wajah ini kemudian di-*resize* dari ukuran *original* ke ukuran 100 x100 piksel.

C.2. Preprocessing

Dataset citra yang sebelumnya dibagi menjadi testing data dan *training* data pada tahap ini hanya digunakan bagian *training* data saja yang berjumlah 100 citra. Terdapat 10 kelas subjek dengan masing-masing 10 citra tiap kelasnya. Setiap citra yang menjadi sampel berformat JPG dalam ruang warna RGB.

Pertama-tama seluruh citra dikonversi ke dalam *grayscale*. Ruang warna ini akan lebih memudahkan dalam proses selanjutnya karena hanya menyisakan satu bentuk matriks untuk diolah. Proses (Lihat Gambar 2) selanjutnya adalah pendeteksian dan pemotongan bagian wajah dengan *library* OpenCV, yaitu metode Viola Jones (Haar Cascade). Hal ini dilakukan agar citra yang di dalamnya mengandung unsur wajah dapat dipotong untuk keefektifan dalam *training* data yang akan dibuat. Ketika wajah telah dideteksi, citra akan dipotong dengan ukuran 100x100 piksel tepat pada bagian wajah saja. Kemudian diekualisasi histogram untuk mendapatkan kualitas citra yang lebih baik.

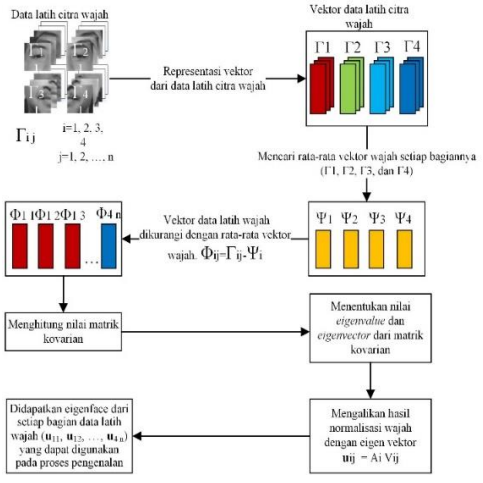
Gambar 2. *Preprocessing* citra

C.3. Ektraksi Fitur

Citra yang tadinya masih mengandung unsur selain wajah (*background*), kini telah dipotong dan berfokus dibagian wajah saja. Langkah selanjutnya adalah ekstraksi fitur dengan metode yang dikembangkan, yaitu *Block-Eigenface*. Citra wajah yang telah didapatkan sebelumnya kini dipotong lagi (*blocking*), namun secara merata sesuai jumlah pikselnya. *Blocking* yang diterapkan pada citra berjumlah 4 *block* dan 16 *block* (2^n di mana n=0, 1, dan 2). Berikut ilustrasi *blocking* pada Gambar 3.

Gambar 4. *Blocking* citra

Di bawah ini merupakan diagram alur proses pelatihan menggunakan metode *Block-Eigenface*. Gambar 4 secara garis besar menjelaskan bagaimana peran metode *Block-Eigenface* dalam ekstraksi fitur yang memanfaatkan metode PCA.



Gambar 4. Alur proses pelatihan dengan *Block-Eigenface*

Pada Gambar 4 dijelaskan bagaimana alur proses pelatihan dengan metode *Block-Eigenface* yang di dalamnya terdapat ekstraksi fitur. Untuk lebih detailnya berikut ini merupakan algoritma dari metode *Block-Eigenface* :

1. Menyiapkan M citra wajah yang akan dibagi menjadi 1 *block* citra (2^j *block* di mana j=0, 1, dan 2). Citra latih wajah pada diagram menggunakan 4 *block*.
2. Mengkonversi seluruh *block* citra data latih ke dalam bentuk vector (Γ_{ij}). (i=banyak *block* dan j=banyak citra)
3. Menghitung nilai rata-rata vektor *block* citra (Ψ_i). Kemudian mengurangi nilai vektor *block* citra dengan nilai rata-rata untuk mendapatkan nilai selisih (Φ_{ij}).

$$\Psi_i = (\Gamma_{ij}+\Gamma_{ij})/j \quad (1)$$

$$\Phi_{ij} = \Gamma_{ij} - \Psi_i \quad (2)$$

4. Menghitung nilai matrik kovarian dengan

$$C=A.A^T \quad (3)$$

5. Menghitung nilai *eigenvector* dan *eigenvalue* yang bisa dilakukan setelah mendapatkan matrik kovarian.
6. Menghitung nilai *eigenface* dengan mengalikan hasil normalisasi setiap *block* citra dengan nilai *eigenvector* yang dipilih berdasarkan *eigenvalue*.

$$u_i = A_i^T \text{ x } v_i \quad (4)$$

Metode *Eigenface* ini diterapkan pada setiap *block* untuk mendapatkan nilai fiturnya. Total ekstraksi yang dilakukan sebanyak tiga kali, yaitu untuk 4 *block*, 16 *block*, dan tanpa *block*. Ekstraksi fitur tanpa *block* merupakan metode *Eigenface* yang datanya akan digunakan sebagai tolak ukur keefektifan *Block-Eigenface*. Semua hasil ekstrasi fitur untuk setiap *blocking* kini siap untuk diekspor ke Raspberry Pi dalam format berkas Ms. Excel (*.*xls*). berkas hasil ekstraksi berisi fitur-fitur dari setiap *blocking* dimana masing-masing *blocking* menghasilkan 3 berkas fitur. Sehingga total berkas yang dibuat adalah 9 berkas dari metode *Eigenface* dan 2 jenis dari *Block-Eigenface*.

C.4. Proses Klasifikasi

Pada proses klasifikasi, ada beberapa langkah yang dilakukan, yaitu :

1. Citra wajah uji yang ingin dikenali dirubah ke dalam bentuk vektor ($\Gamma_{i\ new}$).
2. Kurangi vektor wajah baru dengan *mean* dari *dataset*, seperti :

$$\Phi_{i\ new} = \Gamma_{i\ new} - \Psi_i \quad (5)$$

3. Proyeksi citra wajah uji ke dalam *facespace*, seperti berikut :

$$\mu_i = v_i{}^T \text{ x } \Phi_{i\ new} \quad (6)$$

4. Merepresentasikan *w* ke dalam vector Ω, seperti berikut :

$$\Omega_i = [w_1, w_2, w_3, \ldots w_j] \quad (7)$$

Untuk membandingkan jarak minimum antara *eigenface* citra uji dan *dataset* digunakan metode *Euclidian Distance*. Metode ini akan mengurangi nilai dari matrik u_i dengan nilai μ_i seperti persamaan berikut :

$$d_i = \sqrt{\sum_{i=1}^{n}(u_i - \mu_i)^2} \quad (8)$$

Maka setelah mendapatkan nilai *d* untuk tiap blok kemudian dicari nilai selisih paling minimum.

D. Teknik Pengujian

Untuk menguji metode yang dikembangkan, maka dilakukan pengujian terhadap metode tersebut. Pengujian dilakukan dengan membandingkan beberapa variabel, diantaranya *blocking*, akurasi, waktu komputasi dan titik optimal antara ketiganya. Titik optimal tersebut menjadi tolak ukur sampai sejauh mana *blocking* dapat menghasilkan akurasi yang tinggi dan waktu komputasi minimum.

D.1. Waktu Komputasi pada Blocking

Pada penelitian ini, waktu komputasi menjadi variabel yang penting untuk dipertimbangkan. Karena sistem yang dibuat diimplentasikan pada sebuah mikroprosesor dengan sumberdaya yang bisa dikatakan terbatas. Perlu diketahui relasi antara waktu komputasi, *blocking*, dan akurasi. Untuk menjadikan sistem ini optimal, maka waktu yang singkat dan akurasi tinggi menjadi target yang ingin dicapai. Dengan demikian, waktu klasifikasi setiap citra wajah pada setiap kelas yang di *blocking* perlu dicatat untuk dibandingkan.

D.2. Akurasi Antara Eigenface dan Block-Eigenface

Setelah medapatkan nilai dari kombinasi antara 3 parameter (*blocking*, waktu, dan akurasi), kemudian dilakukan perbandingan terhadap akurasi dari metode *Eigenface* (*non-block*) dan *Block-Eigenface* (*blocking*). Seluruh citra *input* yang dikenali, dibagi dengan jumlah citra pada *dataset*, kemudian dikalikan seratus persen. Untuk menghitung akurasi dari kedua metode tersebut digunakan persamaan berikut ini.

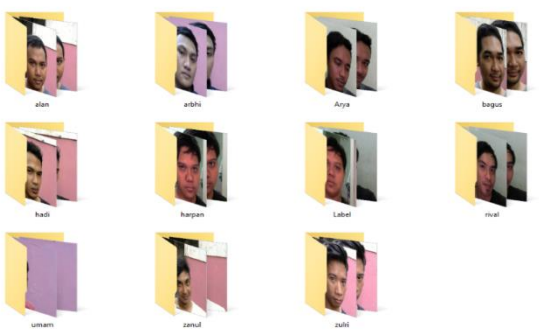
$$\text{Akurasi} = \frac{M}{N} \times 100\ \% \quad (9)$$

IV. Hasil Dan Pembahasan

A. Pengumpulan Data

Pengembangan metode *Block-Eigenface* dilakukan dengan membangun sebuah sistem pengenalan wajah yang menggunakan data wajah dari 10 subjek dengan masing-masing 15 citra untuk dijadikan *dataset*. Pengumpulan data ini mengambil citra dari subjek dengan pose-pose yang telah ditentukan. Setiap citra yang diambil akan dikelompokkan sesuai dengan kelas subjeknya, dan dimasukkan ke dalam *folder* kelas subjek seperti pada Gambar 5.

E-ISSN:2541-0806
P-ISSN:2540-8895



Gambar 5. *Folder* kelas *dataset*

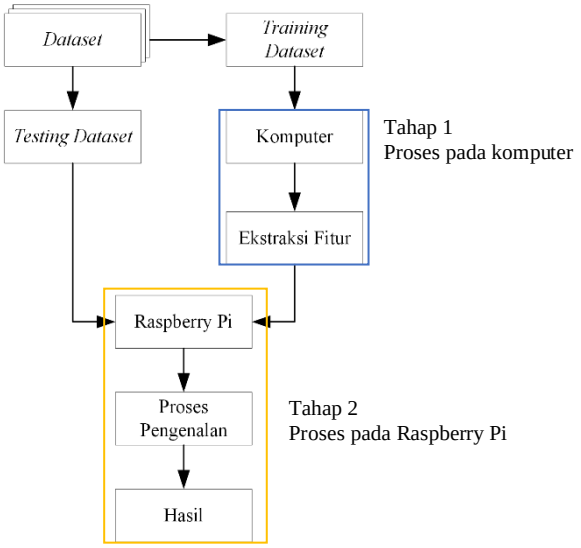
Dataset yang telah dikumpulkan dibagi menjadi 2 bagian, yaitu bagian *training* yang berjumlah 100 citra dan bagian *testing* dengan 50 citra. Pemberian label dilakukan pada 100 citra dari *training data* untuk ke-10 kelas subjek seperti yang ditunjukan pada Gambar 6.

Gambar 6. *Dataset* dengan label kelas

Pada penelitian ini digunakan 2 *dataset*. Salah satunya *dataset* yang telah dijelaskan sebelumnya yang dinamakan *dataset* informatika. *Dataset* kedua yang digunakan yaitu, *dataset* ORL. Penggunaan *dataset* ORL bertujuan untuk dijadikan perbandingan terhadap *dataset* Informatika, sebab telah banyak digunakan pada penelitian sejenis. *Dataset* ORL ini nantinya hanya akan diproses pada komputer tidak dilanjutkan ke mikroprosesor.

B. Proses Kerja Sistem

Pengembangan sistem pengenalan wajah menggunakan metode *Block*-Eigeface ini dilakukan dalam dua tahap utama. Tahap pertama merupakan pengekstraksian fitur dari *training* data pada komputer, sedangkan tahap kedua klasifikasi pada microprocessor Raspberry Pi. Hal ini dilakukan agar RasPi tidak terbebani tugas yang terlalu berat untuk melakukan *training* data mengingat kapabilitasnya yang kecil karena sumberdaya yang terbatas. Gambar 7 merupakan diagram blok alur kedua tahapan tersebut.



Gambar 7. Proses pengembangan sistem

B.1. Proses pada komputer

Keterbatasan sumberdaya menjadi alasan utama dalam membagi keseluruhan kerja sistem dalam dua tahap. Tahap pertama yaitu proses kerja pada komputer, dilakukan persiapan sebelum melakukan ekstraksi fitur. *Preprocessing* menjadi tahapan awal untuk membentuk *training dataset*. *Preprocessing* bertujuan untuk memudahkan proses selanjutnya yakni ekstraksi fitur. Berikut penjelasan untuk *preprocessing* dan ekstraksi fitur.

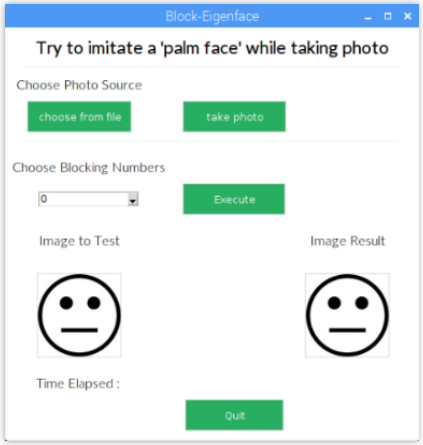
B.2. Proses pada Raspberry Pi

Pada RasPi proses yang dilakukan adalah klasifikasi. Dalam memudahkan interaksi, dilakukan perancangan GUI sederhana untuk mengeksekusi program. Berikut tampilan sederhana yang dibuat.

B.2.1. Graphical User Interface (GUI)

Berikut ini merupakan GUI (Gambar 8-11) yang telah dibuat untuk memudahkan dalam eksekusi program. GUI yang dibuat dikembangkan pada Raspberry Pi menggunakan *module* Tkinter yang merupakan salah satu modul milik Python.

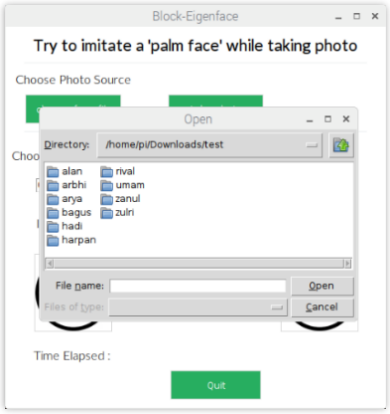
1. Tampilan utama



Gambar 8. Tampilan utama

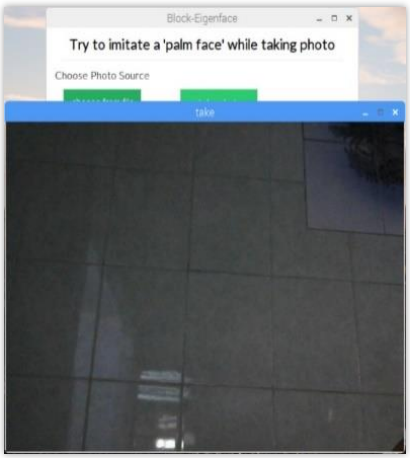
2. Memilih citra dari direktori

E-ISSN:2541-0806
P-ISSN:2540-8895



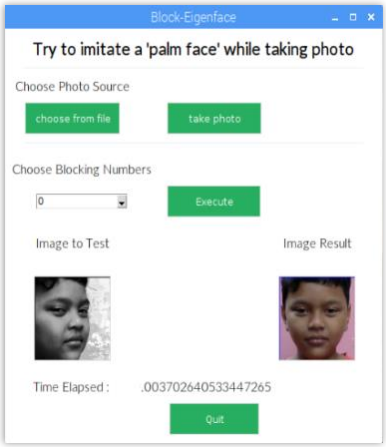
Gambar 9. Memilih citra dari direktori

3. Jendela pengambilan gambar



Gambar 10. Memilih citra dari direktori

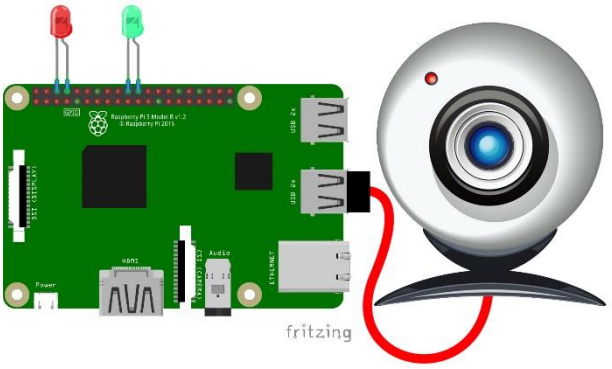
4. Hasil proses pengenalan



Gambar 11. Hasil proses pengenalan

B.2.2. Pengimplementasian metode pada sistem keamanan rumah berbasis IoT

Pada penelitian ini, implementasi metode yang dikembangkan dilakukan pada Raspberry Pi. Tujuan utama penggunaan mikroprosesor ini ialah untuk pengembangan lebih lanjut di bidang IoT. Sistem keamanan rumah merupakan salah satu produk IoT yang dapat menggunakan pengenalan wajah pada otentikasi. Berikut merupakan diagram secara umum posisi sistem pengenalan wajah pada sistem keamanan rumah berbasis IoT.



Gambar 12. Rangkaian alat

Pada Gambar 12 dapat dilihat bentuk rangkaian untuk simulasi pengenalan wajah. Rangkaian ini menjadi salah satu bagian kecil untuk otentikasi pada sistem keamanan rumah. LED yang terpasang pada Raspberry akan menjadi indikator yang mengindikasikan hasil pengenalan. Dapat dikatakan bahwa Raspberry Pi pada sistem ini menjadi salah satu sensor yang keluarannya akan diproses lebih lanjut oleh sistem keamanan rumah. Proses pengenalan yang dilakukan oleh RasPi menghasilkan nilai 1 (LED hijau menyala) yang berarti mendeteksi wajah yang telah terdaftar pada *database*-nya. Apabila tidak terdeteksi wajah yang dikenali, maka RasPi akan memberikan nilai keluaran 0 (LED merah menyala).

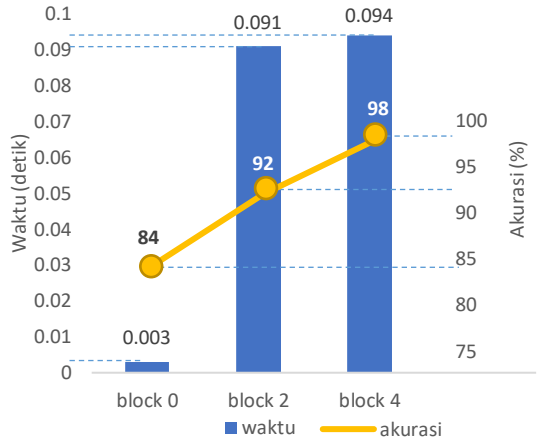
C. Pengujian

Pengujian pada penelitian ini dilakukan pada *Raspberry* Pi dan komputer. Pencatatan waktu dan penghitungan akurasi pada setiap jumlah *blocking* yang diterapkan menjadi variabel pada pengujian ini. Data untuk pengujian merupakan *testing data* yang diambil sekitar 30% dari masing-masing *dataset* yang ada ada dan sisanya menjadi data *training*. Terdapat dua *dataset* yang digunakan yaitu *dataset* Informatika dan *dataset* ORL. Berikut proses pengujian yang dilakukan pada masing-masing *dataset*.

C.1. Dataset Informatika.

Dataset dibagi menjadi 2/3 data *training* dan 1/3 data yang didapat diekspor ke Raspberry Pi. Klasifikasi yang dilakukan pada Raspberry Pi mencatat akurasi dan waktu komputasi.

Dataset ini merupakan data hasil pengumpulan citra wajah mahasiswa informatika. Terdapat 10 subjek dengan masing-masing 15 citra setiap subjek. Seluruh data dibagi seperti pembagian yang dijelaskan sebelumnya, yaitu 100 *training* data dan 50 *testing* data.



Gambar 13. Grafik waktu komputasi dan akurasi tiap *blocking*.

C.1.1. Akurasi *dataset* Informatika

Pada Gambar 13 disajikan data grafik tingkat akurasi (garis) dan waktu (batang) untuk setiap *blocking*. Tingkat akurasi terbawah ditempati oleh metode 0 *block*, yaitu *Eigenface* dengan akurasi 89%. Kemudian disusul oleh 2 *block* dan 4 *block* dengan nilai masing-masing 92% dan 98%. Dari peningkatan akurasi tiap bertambahnya *block* yang diterapkan dapat diketahui bahwa *blocking* berpengaruh pada nilai akurasi. Hal ini dikarenakan bertambahnya konsentrasi komputasi untuk setiap citra baik pada *training data* maupun saat klasifikasi pada *testing data*. Citra yang berukuran 100x100 Piksel dibagi menjadi 4 bagian pada 2 *block* dan 16 bagian pada 4 *block*. Setiap bagian citra ini akan memberikan nilai fiturnya masing-masing yang kemudian digabungkan kembali untuk didapatkan nilai modusnya. Nilai modus pada gabungan *block* citra inilah yang menjadi patokan klasifikasi.

Dapat dikatakan bahwa semakin besaran *block* mendekati ukuran piksel dari citra yang diproses, maka akan semakin tinggi akurasi yang didapatkan. Namun, hal ini tidak menjamin keefektifan program yang dibuat karena waktu dan sumberdaya yang dibutuhkan akan semakin besar.

C.1.2. Waktu komputasi *dataset* Informatika

Dari grafik pada Gambar 16 dapat diamati tingkat keefektifan program yang dibuat dengan menamPilkan perbedaan waktu untuk tiap *block* yang diterapkan (diagram batang). Terdapat perbedaan yang cukup signifikan pada saat *blocking* mulai diterapkan. Selisih sekitar 0,087 detik diantara metode *Eigenface* dengan *Block-Eigenface* 2 dan 4 *block* . selisih waktu yang didapat ini tentu saja memiliki alasan yang sama seperti pada analisa akurasi. Semakin besar nilai *blocking* yang diterapakan, semakin banyak waktu yang dibutuhkan.

Blocking yang diterapkan pada metode ini sangat berpengaruh pada kedua variabel ini. Ketika metode dikomputasi tanpa *blocking*, waktu yang diperlukan berada pada posisi paling cepat sekitar 0.003693 detik, namun akurasinya berada pada posisi paling bawah yaitu 84%. *Blocking* dengan 2 bagian menghasilkan nilai ditengah-tengah untuk catatan waktu dan akurasinya, yaitu 0.09177 detik dan 92%. Sedangkan untuk 4 *block* membutuhkan waktu paling lama, walaupun memiliki selisih sekitar 0.00279 detik terhadap 2 *block* yaitu 0.094 detik dan 98%.

Dengan demikian dapat disimpulkan semakin banyak jumlah *blocking* maka akan semakin bertambah waktu komputasi yang dibutuhkan, namun akan menghasilkan akurasi yang lebih tinggi. Mengacu pada selisih akurasi yang cukup jauh antara metode *Eigenface* dan *Block-Eigenface*, maka *blocking* sangat baik untuk diterapkan demi meningkatkan performa metode *Eigenface*. Melihat selisih waktu komputasi yang masih relatif kecil (di bawah satu detik) antara *block* dan *non-block* tentu saja akurasi masih bisa dijadikan prioritas yang diambil dalam penelitian ini. Sehingga *blocking* dengan nilai terbesar yaitu 4 *block* menjadi Pilihan terbaik berdasarkan akurasi dan waktu komputasinya.

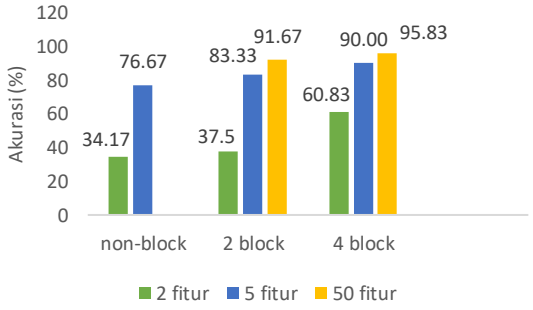
C.2. Dataset ORL

Dataset dibagi menjadi 30% data *training* dan 70% *test*. Semua proses meliputi preprosesing sampai klasifikasi dilakukan pada komputer.

Dataset ini digunakan sebagai salah satu variabel dalam pengujian metode yang dikembangkan. Pengujian untuk *dataset* ini dilakukan pada komputer. *Dataset* ini terdiri dari 40 subjek dengan masing-masing 10 citra wajah tiap subjeknya. Total jumlah data yang digunakan adalah 400 citra yang terdiri dari 280 citra sebagai *data traning* dan 120 citra sebagai *data testing*. 120 citra *data testing* dikumpulkan dengan mengambil 3 citra dari setiap subjek.

Pada *dataset* ini dilakukan pengujian terhadap variasi jumlah fitur pada proyeksi *Eigen*. Tujuannya untuk mengetahui pada jumlah fitur berapa metode ini akan memberikan akurasi dan waktu terbaiknya. Jumlah fitur ini. Berikut nilai akurasi da n waktu komputasi yang diperoleh dari pengujian yang telah dilakukan, masing-masing dapat dilihat pada Gambar 14 dan Gambar 15.

C.2.1. Akurasi *dataset* ORL

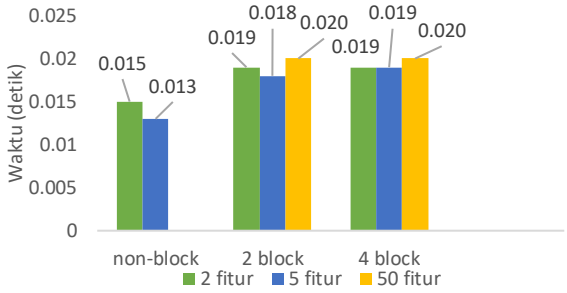


Gambar 14. Grafik akurasi tiap *blocking dataset* ORL

Pada Gambar 14 dapat diketahui bahwa peningkatan akurasi pada *dataset* ORL dengan *dataset* Infomatika memiliki kesamaan. Di mana pertambahan nilai *blocking*

berbanding lurus dengan peningkatan akurasi. Jika diamati, semakin banyak jumlah fitur yang digunakan semakin tinggi akurasi yang didapat pada setiap *blocking*. Namun, nilai maksimal yang bisa diperoleh *non-blocking* hanya 76.67% pada jumlah fitur 5 buah. Tidak diterapkannya 50 fitur pada *non-blocking* dikarenakan tidak memungkinkan untuk melakukan komputasinya. Hal ini terjadi akibat besarnya nilai matriks yang harus dikomputasi sehingga memunculkan error "*Out of memory*". Sedangkan pada 2 *block* dan 4 *block*, penggunaan 50 fitur dapat diterapkan karena matriks yang dikomputasi telah terbagi menjadi beberapa bagian yang lebih kecil. Dilihat dari grafik, akurasi tertinggi didapatkan pada 4 *block* dengan nilai fitur sebanyak 50.

C.2.2. Waktu komputasi *dataset* ORL



Gambar 15. Grafik waktu komputasi tiap *blocking dataset* ORL

Pada Gambar 15 disajikan data waktu komputasi *non-block* dan *blocking*. Waktu yang tercatat paling cepat yakni waktu pada metode *non-block* dengan 5 fitur. Jika pada akurasi jumlah fitur yang digunakan berbanding lurus dengan tingkat akurasi, maka pada waktu komputasi hal tersebut tidak berlaku. Pada data dalam grafik terlihat naiknya nilai fitur yang digunakan tidak selalu meningkatkan waktu komputasi, bahkan ada yang mengalami penurunan. Hal ini disebabkan oleh waktu komputasi persatuan data *test* tidak selalu sama, namun tidak terlampau jauh. Selain itu jumlah data dikenali pada setiap pertambahan fitur semakin meningkat, sehingga pada perhitungan rata-rata waktu nilai pembaginya menjadi lebih besar.

V. KESIMPULAN DAN SARAN

A. Kesimpulan

Berdasarkan penelitian yang dilakukan, maka didapatkan kesimpulan sebagai berikut:

1. Mengimplementasikan metode *Block-Eigenface* dapat dilakukan dengan menjalankan proses *training* pada komputer dan proses klasifikasi pada Raspberry Pi. Hasil ekstraksi fitur pada proses *training* diekspor ke Raspberry Pi untuk nanntinya digunakan pada proses klasifikasi.
2. Proses *blocking* yang diterapkan meningkatkan akurasi metode *Eigenface* dengan nilai optimal yang didapat sebesar 98% pada *dataset* Informatika dan 95.83% pada *dataset* ORL.
3. Waktu komputasi rata-rata dari akurasi optimal masing-masing 0.094 detik pada *dataset* Informatika dan 0.02 detik pada *dataset* ORL.
4. Peningkatan waktu komputasi yang masih berada di bawah 0.1 detik dapat ditoleransi apabila dibandingkan dengan peningkatan akurasinya yang cukup signifikan.

B. Saran

Jika dilakukan penelitian lebih lanjut pada kasus ini dapat mempertimbangkan saran-saran sebagai berikut:

1. Normalisasi diperlukan setelah *cropping* bagian wajah dan *resizing* dilakukan.
2. Perlunya penerapan reduksi dimensi pada perhitungan matriks kovarian yang akan memudahkan komputasi untuk proses selanjutnya.
3. Terkait dengan poin ke-2, maka fitur yang digunakan pada saat proyeksi *Eigen* dapat ditambah tidak terbatas pada 50 fitur.
4. Menambah jumlah *training* data untuk menambah variasi pada *dataset.*
5. Pengujian dengan metode K-fold dapat meningkatkan reliabilitas metode *Block Eigenface.*

Daftar Pustaka

[1] D. N. Parmar and B. B. Metha, "Face Recognition Methods and Applications," *IJCTA*, vol. 4, no. 1, pp. 84-86, 2013.

[2] M. Suriakin, B. Kanata and I. S. Wijaya, "Ekstraksi Ciri Wajah Manusia Menggunakan Algoritma Principal Component Analysis (PCA) untuk Sistem Pengenalan Wajah" *Dielektrika*, vol. 1, no. 1, pp. 16-23, 2014.

[3] M. Zufar and B. Setiyono, "Convolutional Neural Networks untuk Pengenalan Wajah Secara Real-Time," *Jurnal Sains dan Seni ITS*, vol. 5, no. 2, pp. 72-77, 2016.

[4] Fandiansyah, J. Yusmah and I. P. Ningrum, "Pengenalan Wajah Menggunakan Metode Linier Discriminant Analysist dan k Nearest Neighbor," *ULTIMATICS*, vol. 9, no. 1, pp. 1-9, 2017.

[5] W. M. Saputra, S. M. Helmie Arif Wibawa and S. M. Nurdin Bahtiar, "Pengenalan wajah menggunakan Algoritma Eigenface dan euclidian distance," *JOINT*, vol. 2, no. 1, pp. 102-110, 2013.

[6] I. Budiawan and Andriana, "Pengujian Pengenalan Wajah Menggunakan Raspberry Pi," *J.Oto.Ktrl.Inst (J.Auto.Ctrl.Inst)*, vol. 6, no. 2, pp. 135-144, 2014.

[7] S. Wardoyo, R. Wiryadinata and R. Sagita, "Sistem Presensi Berbasis Agoritma Eigenface," *SETRUM*, vol. 3, no. 1, pp. 61-68, 2014.

[8] M. R. Muliawan, B. Irawan and Y. Brianorman, "Implementasi Pengenalan Wajah Dengan Metode Eigenface Pada Sistem Absensi" *Jurnal Coding, Sistem Komputer Untan*, vol. 3, no. 1, pp. 41-50, 2015.

[9] D. E. Pratiwi and A. Harjoko, " Implementasi Pengenalan Wajah Menggunakan PCA," *IJEIS*, vol. 3, no. 2, pp. 175-184, 2013.

[10] Y. Lukito and A. R. Chrismanto, "Perbandingan Metode-Metode Klasifikasi untuk Indoor Positioning System," *Jurnal Teknik Informatika dan Sistem Informasi*, vol. 1, no. 2, pp. 123-131, 2015.

[11] R. Wiryadinata, R. Sagita, S. Wardoyo and Priswanto, "Pengenalan Wajah Pada Sistem Presensi Menggunakan Metode *Dynamic Times Wrapping*, *Principal Component Analysis* dan Gabor Wavelet," *Dinamika Rekayasa*, vol. 12, no. 1, pp. 1-8, 2016.

[12] F. Syuhada, I. G. P. S. Wijaya and F. Bimantoro, "Pengenalan Wajah Untuk Sistem Kehadiran Menggunakan Metode Eigenface dan Euclidian Distance," *J-COSINE*, vol. 2, no. 1, pp. 18-23, 2018.

[13] H. M. Shadiq, Sidjadi and Darjat, "Perancangan Kamera Pemantau Nirkabel Menggunakan Rasberry Pi Mode lB," *Transient*, vol. 3, no. 4, pp. -, 2014.