PENGENALAN WAJAH PELANGGAN TOKO

Semuil Tjiharjadi
Jurusan Sistem Komputer, Universitas Kristen Maranatha
Jl. Suria Sumantri 65, Bandung 40164
E-mail: semuiltj@gmail.com[1]

ABSTRAK

Pada era persaingan yang saat ini sangat ketat, mengenali dan menyapa konsumen merupakan suatu cara untuk mendekatkan diri dengan konsumen. Namun tidak semua pegawai mampu menghapal nama dari pelanggan yang datang. Untuk itu dikembangkan suatu aplikasi pemograman yang mampu mengenali pelanggan sehingga pegawai toko dapat menyapa konsumen dengan nama pelanggan tersebut. Tentunya ini membantu menciptakan retention dan meningkatkan kemungkinan pembelian kembali oleh pelanggan yang merasa diperlakukan dengan terhormat dan penuh kekeluargaan. Aplikasi komputer yang dikembangkan adalah aplikasi komputer yang memiliki kecerdasan yaitu dalam hal ini adalah bagaimana memprogram komputer agar dapat mengenali wajah seseorang hanya dengan menggunakan webcam. Sistem pengenalan wajah ini menggunakan algoritma Eigenface. Dengan menggunakan citra yang dihasilkan melalui webcam dengan menggunakan informasi mentah dari pixel citra yang kemudian direpresentasikan dalam metoda Principal Component Analysis (PCA). Adapun cara kerja algoritma eigenface adalah dengan menghitung rata-rata pixel dari gambar-gambar yang sudah tersimpan dalam suatu database, dari rata-rata pixel tersebut akan didapat nilai eigenface masing-masing gambar dan kemudian akan dicari nilai eigenface terdekat dari gambar citra wajah yang ingin dikenali. Semakin banyak gambar citra wajah pada database maka semakin tinggi tingkat keberhasilan untuk mengenali citra wajah tersebut.

Kata Kunci: pengenalan wajah, eigen face, pengolahan citra

1. PENDAHULUAN

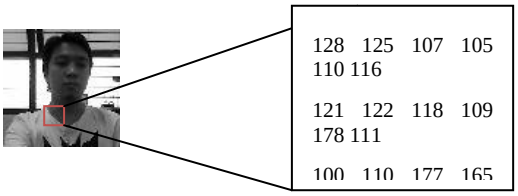
Sejak dahulu kala, proses pengolahan data telah dilakukan oleh manusia. Manusia juga menemukan alat-alat mekanik dan elektronik untuk membantu manusia dalam penghitungan dan pengolahan data supaya mendapatkan hasil yang lebih cepat. Komputer yang ditemui saat ini adalah suatu evolusi panjang dari penemuan-penemuan manusia sejak dahulu kala berupa alat mekanik maupun elektronik.

Aplikasi komputer telah bergeser dari komputasi biasa ke aplikasi komputer yang memiliki kecerdasan. Salah satu konsep kecerdasan adalah bagaimana memprogram komputer agar dapat mengenali wajah seseorang dengan menggunakan *webcam*. Pengenalan wajah ini diharapkan dapat meningkatkan aplikasi komputer khususnya dalam bidang pemasaran produk di toko dengan meningkatkan rasa kekeluargaan dengan pelanggan karena para pegawai dapat memanggil pelanggan dengan namanya.

2. PERANCANGAN

2.1 Citra Digital

Citra digital adalah citra yang dapat diolah oleh komputer. Contoh, pada gambar 2.2 sebuah citra *grayscale* ukuran 80x80 *pixel* diambil sebagian (kotak kecil) berukuran 6x6 *pixel*. Maka, monitor akan menampilkan sebuah kotak kecil. Namun, yang disimpan dalam memori komputer hanyalah angka-angka yang menunjukkan besar intensitas pada masing-masing *pixel* tersebut.



Gambar 1. Citra grayscale ukuran 80x80 pixel
(Sumber : Teori Pengolahan Citra Digital)

2.2 Pengolahan Citra

Pengolahan Citra digital merupakan bentuk dari pemrosesan informasi dengan sebuah gambar sebagai inputnya, seperti gambar sebuah foto atau *frame* dari video. Outputnya berupa gambar yang dapat digunakan sebagai fungsi dari gambar itu sendiri.

Sebuah gambar digital A(m,n) didekripsikan dalam sebuah bidang dua dimensi analog yang diperoleh dari sebuah gambar analog A(x,y) pada sebuah bidang dua dimensi kontinyu dari proses pencuplikan setiap periode yang telah didigitalisasi. Gambar kontinyu dua dimensi A(x,y) dibagi menjadi N baris dan M kolom, titik potong keduanya disebut sebagai *pixel*.

Gambar digital merupakan data numeris yang dapat diolah komputer untuk mendapatkan informasi yang ada padanya, karena gambar digital direpresentasikan dalam sebuah matriks.

2.3 Gambar digital

Sebuah gambar berwarna pada sistem digital merupakan perpaduan dari tiga macam warna RGB

 ISSN: 1907-5022

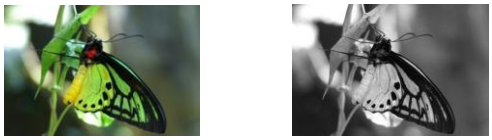
(*Red*, *Green*, *Blue*) untuk setiap titiknya, setiap komponen warna diwakili dengan satu *byte*. Jadi untuk masing-masing komponen R, G, dan B mempunyai variasi dari 0 sampai 255. Total variasi yang dihasilkan untuk sistem warna digital ini adalah 256 x 256 x 256 atau 16,777,216 jenis warna. Karena setiap komponen diwakili dengan satu *byte* atau delapan bit, maka total bit yang digunakan untuk merepresentasikan warna adalah 8 + 8 + 8 atau 24 bit.

Proses mengubah gambar warna menjadi gambar *grayscale* digunakan dalam *image processing* untuk menyederhanakan model gambar. *Grayscale* adalah warna-warna *pixel* sebuah gambar yang dikonversi menjadi gambar abu-abu. Sistem *grayscale* hanya memerlukan satu *byte* atau delapan bit untuk menyimpan data, sehingga hanya mempunyai variasi dari 0 (hitam) sampai 255 (putih).

Ada beberapa macam cara untuk mengkonversi sistem warna RGB menjadi *grayscale* yaitu :

- Dengan rata-rata setiap komponen warna RGB.
  Grayscale = (R+G+B)/3
- Menggunakan nilai maksimal komponen RGB.
  Grayscale = *Max* {R, G, B}
- Menggunakan sistem YUV (sistem warna pada NTSC), dengan cara mengambil komponen Y (iluminasi). Komponen Y sendiri diperoleh dari sitem warna RGB dengan konversi :

Grayscale = Y = 0.299 x R + 0.587 x G + 0.114 x B



Gambar 2. Perbedaan gambar berwarna dan grayscale

2.4 Pengenalan Wajah

Secara umum sistem pengenalan citra wajah dibagi menjadi 2 jenis, yaitu sistem *feature based* dan sistem *image-based*. Pada sistem pertama digunakan fitur yang diekstraksi dari komponen citra wajah (mata, hidung, mulut, dll.) yang kemudian hubungan antara fitur-fitur tersebut dimodelkan secara geometris. Sedangkan sistem kedua menggunakan informasi mentah dari *pixel* citra yang kemudian direpresentasikan dalam metode tertentu, misalnya *Principal Component Analysis* (PCA) yang kemudian digunakan untuk klasifikasi identitas citra.

2.5 *Principal Component Analysis* (PCA)

Prosedur PCA pada dasarnya adalah bertujuan untuk menyederhanakan variabel yang diamati dengan cara menyusutkan (mereduksi) dimensinya. Hal ini dilakukan dengan cara menghilangkan korelasi di antara variabel bebas melalui transformasi variabel bebas asal ke variabel baru yang tidak berkolerasi sama sekali atau yang biasa disebut dengan *principal component*.

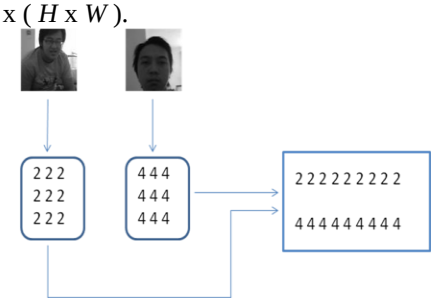
2.6 Algoritma *Eigenface*

Eigenface adalah salah satu algoritma pengenalan wajah yang didasarkan pada *Principal Component Analysis* (PCA). Untuk menghasilkan *eigenface*, sekumpulan besar citra digital dari wajah manusia diambil pada kondisi pencahayaan yang sama dan kemudian dinormalisasikan dan kemudian diolah pada resolusi yang sama (misalnya m x n), dan kemudian diperlakukan sebagai *vector* dimensi *mn* yang komponennya diambil dari nilai *pixel*-nya.

Langkah-langkah pengenalan wajah dengan algoritma *eigenface* adalah sebagai berikut:

1. Penyusunan *FlatVector*

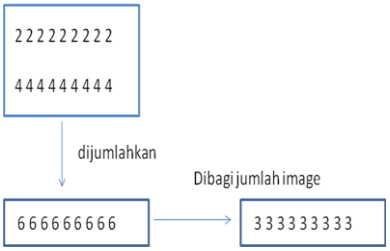
   Langkah pertama adalah menyusun sebuah *training image* menjadi 1 matriks tunggal. Misalnya image yang disimpan berukuran *H* x *W* *pixel* dan jumlahnya *N* buah, maka akan memiliki *flatvector* dengan dimensi *N* x (*H* x *W*).



Gambar 3. Contoh Penyusunan FlatVector
(Sumber : Rekayasa Sistem Pengenalan Wajah)

2. Perhitungan Rataan *FlatVector*

   Dari *flatvector* yang diperoleh, jumlahkan seluruh barisnya sehingga diperoleh matriks tunggal berukuran 1 x (*H* x *W*). Setelah itu bagi matriks tadi dengan jumlah *image* *N* untuk mendapatkan Rataan *flatVector*.

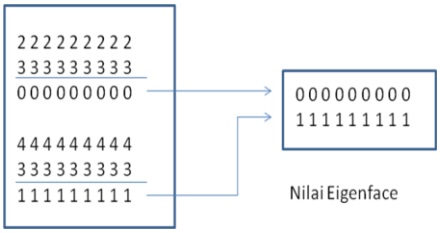


Gambar 4. Contoh Penghitungan Rataan FlatVector
(Sumber : Rekayasa Sistem Pengenalan Wajah)

3. Tentukan nilai *eigenface*

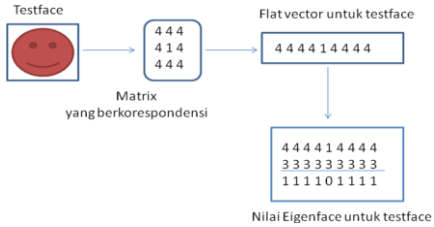
   Dengan menggunakan rataan *flatvector* akan dihitung *eigenface* untuk matriks *flatvector* yang telah disusun. Caranya dengan mengurangi baris-baris pada matriks *flatvector* dengan rataan *flatvector*. Jika didapatkan nilai di bawah nol, ganti nilainya dengan nol.

 ISSN: 1907-5022



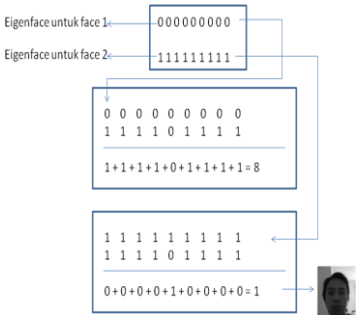
Gambar 2.6 Contoh Menentukan Nilai Eigenface
(Sumber : Rekayasa Sistem Pengenalan Wajah)

4. Proses Identifikasi
   Jika diberikan citra yang akan diidentifikasikan (*testface*), maka langkah identifikasinya adalah sebagai berikut:
   Kalkulasi nilai *eigenface* untuk matriks *testface*, dengan cara yang sama dengan penentuan *eigenface* untuk *flatvector*.



Gambar 5. Perhitungan Eigenface untuk Testface

Setelah nilai *eigenface* untuk *testface* diperoleh maka dapat dilakukan identifikasi dengan menentukan jarak (*distance*) terpendek dengan *eigenface* dari *eigenvector training image*. Caranya, tentukan nilai absolute dari penggunaan baris *i* pada matriks *eigenface training image* dengan *eigenface* dari *testface*, kemudian jumlahkan elemen-elemen penyusun *vector* yang dihasilkan dari pengurangan tadi dan ditemukan jarak *d* indeks *i*. Lakukan untuk semua baris dan cari nilai *d* yang paling kecil.

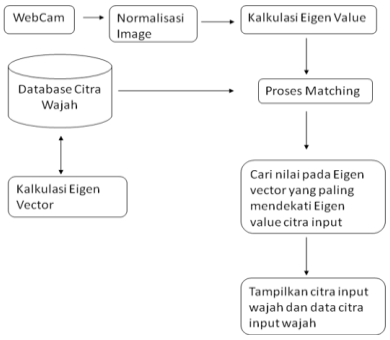


Gambar 6. Contoh Proses Identifikasi

2.7 Diagran Blok Sistem

Keterangan diagram blok :

- Citra wajah di-*capture* menggunakan *webcam*. Hasil dari *capturing* ini adalah file gambar yang bertipe *.bmp*.
- Citra wajah ini kemudian dinormalisasi dengan beberapa tahapan. Pertama, citra diturunkan kualitas warnanya menjadi *grayscale*. Ukuran dari citra wajah juga diseragamkan, menjadi berukuran 80 x 80 *pixel*.
- Setelah didapatkan citra wajah yang ternomalisasi, hitung nilai *eigen* dari citra wajah tersebut, misalnya diperoleh nilai *x*.
- Pada *database* citra wajah terdapat koleksi citra wajah. Dari koleksi ini masing-masing citra dikalkulasi nilai *eigen*-nya dan dikumpulkan dalam vektor yang kita namakan *eigenvector*. Misalkan didapat nilai (*x*1, *x*2, *x*3,...*x*n).
- Proses *matching* dilakukan dengan mencocokan nilai *x* dengan nilai-nilai pada *eigenvector* dan mencari nilai yang paling mendekati.
- Jika nilai yang paling mendekati sudah ditemukan, cari data wajah yang berkorespondensi dengan nilai tadi dan tampilkan citra wajah dan data citra wajah.



Gambar 7. Diagram Blok

2.8 Perancangan *Database*

Database yang digunakan adalah Microsoft Access 2007. *Database* terdiri dari tabel *id* untuk menyimpan data dari citra wajah dan tabel *setting* untuk menyimpan nilai persentase terdekat ketika mengenali wajah.

Pada tabel *id*, yang menjadi *primary key* adalah *id* dengan *data type AutoNumber* sebagai tempat untuk menyimpan index yang digunakan untuk index penyimpanan citra wajah. Nrp dan Nama dengan *data type text* digunakan untuk menyimpan data dari citra wajah.

Pada tabel *setting*, tidak ada *primary key* karena hanya akan berisi satu data saja. Pada nama akan diisi data *MinEigen* dan pada nilai akan berisi nilai yang menjadi persentase dalam mengenali wajah, nilai ini dapat diisi melalui program.

Tabel 1. Database Id

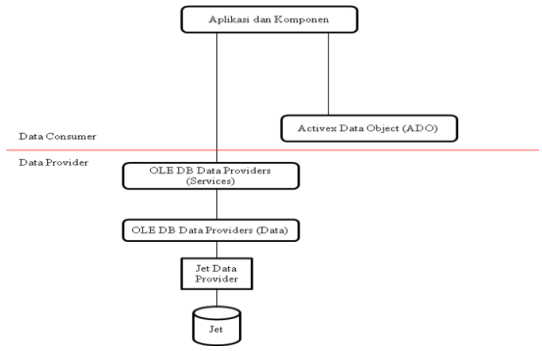
Field Name	Data Type	Field Size
Id*	AutoNumber	-
Nrp	Text	10
Nama	Text	50

Tabel 2. Database Setting

Field Name	Data Type	Field Size
Nama	Text	50
Nilai	Number	LongInteger

 ISSN: 1907-5022

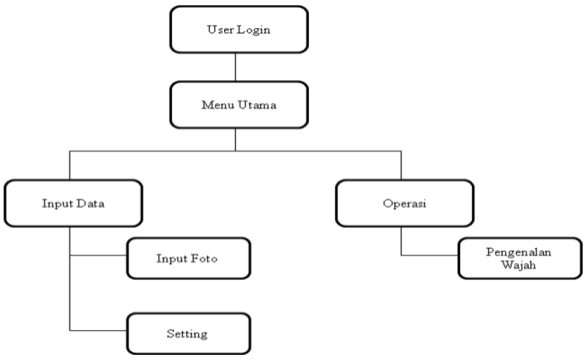
Sebagai *data access* antara *Microsoft Acces* dan Visual Basic 6, menggunakan *Activex Data Object* (ADO) sebagai *data consumers* yang menggunakan data dan OLE DB sebagai *data providers* yang menampung dan membongkar data. Hubungan antara aplikasi dan komponen dengan *database* dapat dilihat pada gambar 7 .



Gambar 8. Data Consumer dan Data Provider

2.9 Struktur Program

Struktur menu yang digunkan pada program pengenalan wajah ini dapat dilihat pada bagan berikut :



Gambar 9. Struktur Program

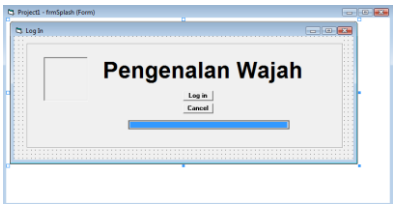
Keterangan :
User Login : tampilan awal program pengenalan wajah.
Menu Utama : tampilan menu yang terdiri dari data dan operasi.
Input Data : Digunakan untuk memasukkan data. Ada 2 pilihan submenu, yaitu **Input Foto**, digunakan untuk memanggil form pengambilan gambar citra wajah dan pengisian data dari citra wajah, dan **Setting**, digunakan untuk mengubah persentase kemiripan pengenalan wajah.
Operasi→Pengenalan Wajah : Digunakan untuk memanggil form pengenalan wajah.

2.10 Desain *Input*

Desain input dirancang sebagai tampilan antarmuka di mana pengguna dapat memasukkan *input* baik berupa data melalui *keyboard* dan melalui *webcam*. Adapun desain *input* untuk program pengenalan wajah ini terdiri dari bagian-bagian :

1. Desain *Form Login User*

Pada form ini user cukup hanya menekan *button* Login untuk dapat mengakses menu program pengenalan wajah.



Gambar 10. Desain Form Login

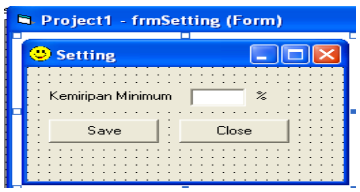
2. Desain *Form Input* Foto

Pada form ini user dapat mengambil gambar citra wajah dan mengisi data-data citra wajah tersebut. Komponen yang digunakan untuk *capture* gambar adalah *ezVidcap* dan untuk menampilkan *database* adalah *dataGrid*.

Gambar 11. Desain Form Input Foto

1. Desain *Form Setting*

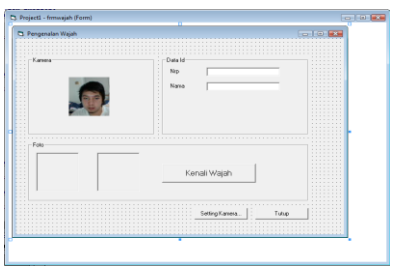
Pada form ini dapat dilakukan perubahan presentase tingkat kemiripan yang diinginkan untuk pengenalan wajah.



Gambar 12. Desain Form Setting

2. Desain *Form* Pengenalan Wajah

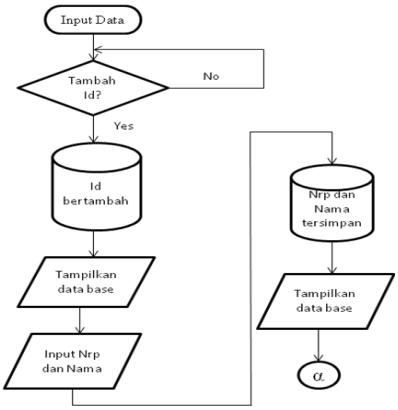
Pada form ini citra wajah di-*capture* dari webcam untuk kemudian secara terkomputerisasi dicocokkan dengan citra wajah pada *database* dan datanya. Proses pengenalan wajah dilakukan pada form ini.



Gambar 13. Desain Form Pengenalan Wajah

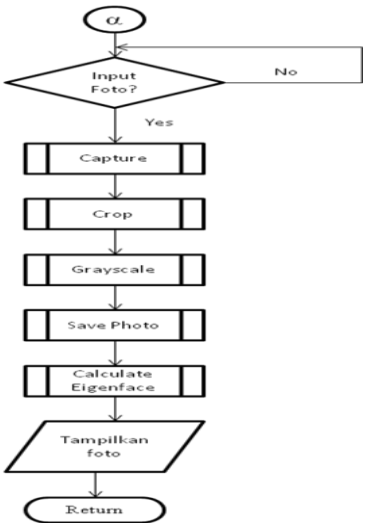
Adapun data *flowchart* dari program pengenalan wajah dibagi menjadi 3 bagian, yaitu :

1. Input Data



Gambar 11. Flowchart Input Data

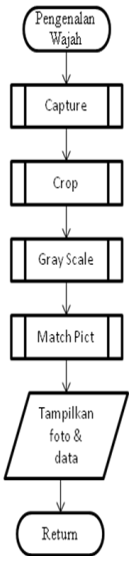
Pada proses *input data*, pertama-tama harus menambahkam *id* terlebih dahulu. Hal ini dilakukan agar *id* dari *database* dengan *type AutoNumber* bertambah. Setelah ditampilkan pada *database* dan *user* dapat mengisi *input* Nrp dan Nama yang akan disimpan dalam *database* kemudian data-data tersebut akan ditampilkan pada *form* antarmuka.



Gambar 12. Flowchart Input Foto

Pada proses *input* foto, *user* akan diminta untuk meng-*capture* citra wajah *user*. Setelah di-*capture*, citra wajah yang didapat akan diproses ukurannya menjadi 80 x 80 *pixel*, diturunkan kualitas citranya menjadi *grayscale*, disimpan dalam *database*, dan akan dihitung nilai *eigenface*-nya. Setelah itu foto akan ditampilkan pada *form* antarmuka.

Pada proses pengenalan wajah, wajah yang akan dikenali akan di-*capture* kemudian citra yang didapat akan diproses ukurannya menjadi 80 x 80 *pixel*, diturunkan kualitas citranya menjadi *grayscale*, dan kemudian pada pencocokkan gambar akan dihitung nilai *eigenface*-nya dan akan dicari nilai *eigenface* yang paling mendekati dengan citra wajah yang sudah tersimpan dalam *database*.



Gambar 13. Flowchart Pengenalan Wajah

3. PENGUJIAN

Data pengamatan pertama, didapat berdasarkan hasil percobaan dengan *background* yang sama tetapi ada benda lain pada *background* tersebut, pencahayaan yang terang, wajah yang di-*capture* tanpa ekspresi atau formal. Percobaan dilakukan pada 5 orang, setiap orang hanya memiliki 1 *database* wajah dan dilakukan percobaan sebanyak 5 kali.

Keterangan :
√ = wajah terdidentifikasi
X = wajah yang teridentifikasi tidak sesuai

Tabel 3 Data Pengamatan I

Percobaan ke-	I	II	III	IV	V
Willy	√	√	√	√	√
Oscar	√	√	X	X	X
Rina	√	X	√	X	X
Martha	√	X	√	X	√
Jimmy	√	√	√	√	√

Dari percobaan pertama, ditemukan kegagalan sebanyak 8 kali. Wajah yang teridentifikasi tidak sesuai dengan wajah yang ingin dikenali. Keberhasilan pengenalan wajah didapat hanya 68%.

Data pengamatan ke-2 didapat berdasarkan hasil percobaan dengan *background* yang sama tetapi ada benda lain pada *background* tersebut, pencahayaan yang terang, wajah yang di-*capture* tanpa ekspresi atau formal. Percobaan dilakukan pada 5 orang, setiap orang hanya memiliki 5 *database* wajah dan dilakukan percobaan sebanyak 10 kali.

Tabel 3. Data Pengamatan II

Percobaan ke-	I	II	III	IV	V	VI	VII	VIII	IX	X
Martha	√	√	√	√	√	√	√	√	√	√
Rina	√	√	√	X	√	√	√	√	√	√
Oscar	√	√	√	√	X	√	X	√	√	√
Insan	√	√	√	√	√	√	√	√	√	√
Willy	√	√	√	√	√	√	√	√	√	√

Dari percobaan ke-2, ditemukan kegagalan sebanyak 3 kali. Wajah yang teridentifikasi tidak sesuai dengan wajah yang ingin dikenali. Keberhasilan pengenalan wajah didapat 94%.

Data pengamatan ke-3 didapat berdasarkan hasil percobaan dengan *background* yang sama yaitu berwarna biru, pencahayaan yang terang, dan wajah yang di-*capture* tanpa ekspresi atau formal. Percobaan dilakukan pada 5 orang, setiap orang memiliki *database* wajah sebanyak 1 wajah dan dilakukan percobaan sebanyak 5 kali setiap orang. Hasilnya adalah sebagai berikut :

Tabel 4. Data Pengamatan III

Percobaan ke-	I	II	III	IV	V
Willy	√	√	√	√	√
Oscar	√	X	√	X	√
Rina	√	√	X	X	√
Martha	√	√	√	X	√
Insan	√	√	X	√	√

Dari percobaan ke-3, ditemukan kegagalan sebanyak 6 kali. Wajah yang teridentifikasi tidak sesuai dengan wajah yang ingin dikenali. Keberhasilan pengenalan wajah didapat 76%.

Data pengamatan ke-4 didapat berdasarkan hasil percobaan dengan *background* yang sama yaitu berwarna biru, pencahayaan yang terang, dan wajah yang di-*capture* tanpa ekspresi atau formal. Percobaan dilakukan pada 5 orang, setiap orang memiliki *database* wajah sebanyak 5 wajah dan dilakukan percobaan sebanyak 10 kali setiap orang. Hasilnya adalah sebagai berikut :

Tabel 5. Data Pengamatan IV

Perc ke	1	2	3	4	5	6	7	8	9	10
Martha	√	√	√	√	√	√	√	√	√	√
Rina	√	√	√	√	√	√	√	√	√	√
Oscar	√	√	√	√	X	√	X	√	√	√
Insan	√	√	√	√	√	√	√	√	√	√
Willy	√	√	√	√	√	√	√	√	√	√

Dari percobaan ke-2, ditemukan kegagalan sebanyak 2 kali. Wajah yang teridentifikasi tidak sesuai dengan wajah yang ingin dikenali. Keakuratan dalam pengenalan wajah menggunakan algoritma *eigenface* pada percobaan ke-2 didapat sampai 96%.

4. KESIMPULAN

Dari hasil percobaan yang sudah dilakukan, dapat ditarik kesimpulan antara lain:

- Aplikasi pengenalan wajah dapat dibuat dengan menggunakan algoritma *eigenface* untuk pengenalan wajah, *software* Visual Basic, dan Microsoft Access sebagai *database* dari gambar-gambar citra.
- Algoritma *eigenface* bekerja dengan menghitung rata-rata *pixel* dari gambar-gambar yang sudah tersimpan dalam suatu *database*, dari rata-rata *pixel* tersebut akan didapat nilai *eigenface* masing-masing gambar dan kemudian akan dicari nilai *eigenface* terdekat dari gambar citra wajah yang ingin dikenali.
- Wajah seseorang yang dapat dikenali adalah yang tersimpan dalam *database* gambar wajah yang sudah diproses dengan algoritma *eigenface*, selain itu wajah tidak akan dikenali atau wajah yang dikenali tidak sesuai.
- Keberhasilan pengenalan wajah menggunakan algoritma *eigenface* semakin tinggi bila *database* citra wajah yang tersimpan semakin banyak. Kesimpulan ini didapat dari hasil perbandingan antara data pengamatan pertama yang dicoba pada 5 orang dengan *backgrournd* yang memiliki benda lain tetapi setiap orang hanya terdapat satu *database* citra wajah, dan data pengamatan ke-2 yang dicoba pada 5 orang dengan *background* yang memiliki benda lain tetapi setiap orang terdapat 5 *database* citra wajah. Data pengamatan ke-2 dengan *database* citra wajah yang lebih banyak memiliki tingkat keberhasilan yang lebih tinggi dibandingkan dengan data pengamatan pertama yang hanya terdapat 1 *database* saja, demikian pula perbandingan antara data pengamatan ke-3 yang dicoba pada 5 orang dengan *backgrournd* tidak terdapat benda lain tetapi setiap orang hanya terdapat satu *database* citra wajah, dan data pengamatan ke-4 yang dicoba pada 5 orang dengan *background* tidak terdapat benda lain tetapi setiap orang terdapat 5 *database* citra wajah. Data pengamatan ke-4 dengan *database* citra wajah yang lebih banyak memiliki tingkat keberhasilan yang lebih tinggi dibandingkan dengan data pengamatan ke-3 yang hanya terdapat 1 *database* saja.

PUSTAKA

Fatta,Hanif al. (2009), *Rekayasa Sistem Pengenalan Wajah*, C.V ANDI OFFSET, Yogyakarta, .

MADCOMS. (2002) *Seri Panduan Pemrograman Database Visual Basic 6.0 dengan Crystal Report*, C.V ANDI OFFSET , Yogyakarta.

Mesran (2009), *Visual Basic*, Mitra Wacana Media , Jakarta.

Sutoyo, T., Edy Mulyanto. (2009). *Teori Pengolahan Citra Digital*, C.V ANDI OFFSET, Yogyakarta.

http://docs.google.com/viewer?a=v&q=cache:UbtpeK_yZKsJ:resources.unpad.ac.id/unpad-content/uploads/publikasi_dosen/PCA. (22 Juli 2010)