APLIKASI SISTEM PENGENALAN WAJAH MENGGUNAKAN METODE EIGENFACE BERBASIS J2SE

SKRIPSI

Disusun oleh :

NATALINO HERCIO S DC AMARAL
NIM. 07.12.535

JURUSAN TEKNIK ELEKTRO S-1
KONSENTRASI TEKNIK KOMPUTER & INFORMATIKA
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI NASIONAL MALANG
2012

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

LEMBAR PERSETUJUAN

APLIKASI SISTEM PENGENALAN WAJAH MENGGUNAKAN METODE EIGENFACE BERBASIS J2SE

SKRIPSI

Disusun dan Diajukan Sebagai Salah Satu Syarat Untuk Memperoleh Gelar Sarjana Teknik

Disusun Oleh :

NATALINO HERCIO SILVANIA DC AMARAL

NIM : 07.12.535

Mengetahui

Ketua Jurusan Teknik Elektro S-1

Ir. Yusuf Ismail Nakhoda, MT

NIP.Y.101880089

Diperiksa dan Disetujui

Dosen Pembimbing I	Dosen Pembimbing II
(Irmalia Suryani Faradisa, ST, MT)	**(Sotyohadi, ST)**
NIP.P. 1030000365	**NIP.Y.1039700309**

JURUSAN TEKNIK ELEKTRO S-1
KONSENTRASI TEKNIK KOMPUTER DAN INFORMATIKA
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI NASIONAL MALANG
2011

SURAT PERNYATAAN ORISINALITAS

Yang bertanda tangan dibawah ini :

Nama : Natalino Hercio S DC Amaral

Nim : 07.12.535

Program Studi : Teknik Elektro

Konsentrasi : Teknik Komputer dan Informatika

Dengan menyatakan bahwa skripsi yang saya buat adalah hasil karya sendiri, tidak merupakan plagiasi Dari karya orang lain.dalam skripsi ini tidak memuat karya orang lain, kecuali dicantumkan sumbernya sesuai dengan ketentuan yang berlaku.Demikian surat ini saya buat, dan apa bila dikemudian hari ada pelangaran atas surat pernyataan ini, saya bersedia menerima sangsinya.

Malang, 25 Februari 2013

Yang membuat pernyataan

Natalino Hercio Dc Amaral

Nim : **0712535**

ABSTRAKSI

APLIKASI SISTEM PENGENALAN WAJAH MENGGUNAKAN METODE EIGENFACE BERBASIS J2SE

Natalino Hercio S Dc Amaral

07.12.535

Email : nhercio@yahoo.com

Jurusan Teknik Elektro

Kosentrasi Teknik Komputer dan Informatika

Fakultas Teknik Industri

Institut Teknologi Nasional Malang

Dosen Pembimbing I : Irmalia Suryani Faradisa, ST,MT
Dosen Pembimbing II : SotyoHadi, ST

Face recognition atau pengenalan wajah adalah salah satu teknologi biometrik yang telah banyak diaplikasikan dalam sistem keamanan selain pengenalan retina mata,
pengenalan sidik jari dan iris mata.

Dalam implementasinya, pengenalan wajah dengan menggunakan sebuah webcam untuk menangkap wajah seseorang
kemudian dibandingkan dengan wajah sebelumnya yang telah disimpan di dalam database tertentu.

Tugas akhir ini membahas tentang algoritma yang diperlukan agar komputer dapat mengenali dan mengidentifikasi suatu wajah testing yang diinputkan berdasarkan wajah training yang telah diinput sebelumnya.

Kata Kunci : Metode Eigenface, Java , MySQL.

KATA PENGANTAR

Puji syukur kami panjatkan kehadirat Tuhan Yang Maha Esa atas anugerah dan karunianaya, sehingga penulis dapat menyelesaikan skripsi dengan judul "**APLIKASI SISTEM PENGENALAN WAJAH MENGGUNAKAN METODE EIGENFACE BERBASIS J2SE**" dengan lancar. Skripsi merupakan persyaratan kelulusan Studi di Jurusan Teknik Elektro S-1 Konsentrasi Teknik Komputer dan Informatika ITN Malang dan untuk mencapai gelar Sarjana Teknik.

Keberhasilan penyelesaian laporan skripsi ini tidak lepas dari dukungan dan bantuan berbagai pihak. Untuk itu penyusun menyampaikan terima kasih kepada :

1. Bapak Ir. Soeparno Djiwo, MT selaku Rektor Institut Teknologi Nasional Malang.
2. Bapak Ir. Sidik Noetjahjono, MT selaku Dekan Fakultas Teknologi Industri Institut Teknologi Nasional Malang
3. Bapak Ir. Yusuf Ismail Nakhoda, MT selaku Ketua Jurusan Teknik Elektro S-1,
4. Ibu Irmalia Suryani F aradisa, ST, MT selaku Dosen pembimbing I.
5. Bapak Sotyohadi, ST, selaku Dosen Pembimbing II.
6. Ayah dan Ibu serta saudara-saudara yang selalu memberikan do'a, motivasi dan semangat.
7. Teman - teman Muabere di Karanglo yang selalu memberikan motivasi dan semangat.

Penulis telah berusaha semaksimal mungkin dan menyadari sepenuhnya akan keterbatasan pengetahuan dalam menyelesaikan laporan ini. Untuk itu penyusun mengharapkan saran dan kritik yang membangun dari pembaca demi kesempurnaan laporan ini.

Harapan penulis semoga laporan skripsi ini memberikan manfaat bagi perkembangan ilmu pengetahuan dan pembaca.

Malang, Agustus 2012

Penulis

DAFTAR ISI

DAFTAR GAMBAR

BAB II LANDASAN TEORI

BAB III PERANCANGAN DAN DESAIN SISTEM

BAB IV IMPLEMENTASI DAN PENGUJIAN

DAFTAR TABEL


INSTITUT TEKNOLOGI NASIONAL
ITN
MALANG

BAB I
PENDAHULUAN

1.1. Latar Belakang

Dewasa ini perkembangan ilmu dan teknologi yang berkaitan dengan pengukuran dan analisis statistik data biologis (biometrik) berkembang dengan demikian pesatnya ini dikarenakan sebuah ciri biologi dapat memberikan informasi yang unik berkaitan dengan identifikasi masing – masing individu. Dalam teknologi informasi, biometrik biasanya merujuk kepada teknologi untuk mengukur dan menganalisis karakteristik tubuh manusia seperti sidik jari, retina, mata, pola suara, dan pola wajah yang terutama sekali digunakan untuk proses otentifikasi.

Pengenalan wajah manusia adalah salah satu dari pengenalan pola yang penting. Karena inilah maka peneliti mengambil judul Tugas Akhir "APLIKASI SISTEM PENGENALAN WAJAH MENGGUNAKAN METODE *EIGENFACE*". Untuk menggunakannya sistem mengambil foto citra wajah dan mencocokan dengan citra wajah pada database yang disimpan.

Pencocokan kemiripan wajah dilakukan dengan algoritma pengenalan wajah *Eigenface*. *Eigenface* adalah kumpulan dari eigenvector yang digunakan untuk masalah computer vision pada pengenalan wajah manusia. Menurut Lyman (2007), *eigenface* adalah sekumpulan standardized face ingredient yang diambil dari analisis statistik dari banyak gambar wajah. Satu wajah manusia dapat dipandang sebagai kombinasi dari wajah-wajah standar ini. sehingga jika ingin merekam wajah seseorang untuk pengenalan wajah maka bisa digunakan jauh lebih sedikit fitur dari pada yang ditangkap oleh foto digital.

Untuk menghasilkan *eigenface*, sekumpulan besar citra digital dari wajah mausia diambil pada kondisi pencahayaan yang sama dan kemudian dinormalisasi selanjutnya diolah pada resolusi yang sama. Lalu citra diambil dari nilai pixel-nya. Untuk menentukan *eigenface* dari sekumpulan citra wajah, digunakan algoritma *eigenface* berdasarkan Principle Component Analysis (PCA).

1.2. Rumusan Masalah

Permasalahan yang dibahas adalah bagaimana menggunakan citra wajah hasil bidikan capture webcam sebagai pembanding citra wajah untuk mengetahui hasil nilai kemiripan wajah.

1.3. Tujuan

Tujuan penulis skripsi ini adalah :

1. Mengimplementasikan ilmu yang sudah didapat selama menempuh kuliah di Institut Teknologi Nasional Malang.
2. Membuat program pengenalan wajah dengan memakai Metode *Eigenface*
3. Untuk mengetahui hasil perbandingan nilai perbandingan citra wajah latih dengan citra wajah baru.

1.4. Batasan Masalah

Batasan masalah dalam penelitian ini antara lain:

1. Aplikasi yang dibuat dalam penilitian ini hanya untuk sebuah pengenalan wajah.
2. Pengambilan data trining dan data tes menggunakan *webcame*
3. Metode yang dipakai adalah metode *Eigenface*
4. Tidak sampai membuat aplikasi untuk absensi atau pengenalan wajah pelangan toko dan lain-lain. Tetapi hanya pengenalan secara umum.
5. Citra wajah yang diggunakan hanya dengan format .jpg
6. Hasil pengenalan yang dihasilkan sistem pada aplikasi yang telah diterapkan berupa citra wajah.

1.5. Metode Penilitian

Adapun metode penelitian yang digunakan adalah sebagai berikut:

1. Studi literatur

   Pengumpulan data yang dilakukan dengan mencari bahan-bahan kepustakaan dan referensi dari berbagai sumber sebagai landasan teori yang ada hubunganya dengan permasalahan yang dijadikan objek penelitian.

2. Analisa Kebutuhan Sistem

   Data dan informasi yang telah diperoleh akan dianalisa agar didapatkan kerangka global yang bertujuan untuk mendefinisikan kebutuhan sistem di mana nantinya akan digunakan sebagai acuan perancangan sistem.

3. Perancangan dan Implementasi

   Berdasarkan data dan informasi yang telah diperoleh serta analisa kebutuhan untuk membangun sistem ini, akan dibuat rancangan kerangka global yang menggambarkan mekanisme dari sistem yang akan dibuat dan diimplementasikan kedalam system.

4. Eksperimen dan Evaluasi

   Pada tahap ini, sistem yang telah selesai dibuat akan diuji coba, yaitu pengujian berdasarkan fungsionalitas program, dan akan dilakukan koreksi dan penyempurnaan program jika diperlukan.

1.6. Sistematika Penulisan

Untuk mempermudah dan memahami pembahasan penulisan skripsi ini, maka sistematika penulisan disusun sebagai berikut :

Bab I : PENDAHULUAN

Berisi Latar Belakang, Rumusan Masalah, Tujuan Penelitian, Pembatasan Permasalahan, Metode Penelitian dan Sistematika Penulisan.

Bab II : TINJAUAN PUSTAKA

Berisi tentang landasan teori mengenai permasalahan yang berhubungan dengan penelitian yang dilakukan.

Bab III : PERANCANGAN DAN ANALISA SISTEM

Dalam bab ini berisi mengenai analisa kebutuhan sistem baik software maupun *hardware* yang diperlukan untuk membuat kerangka global yang menggambarkan mekanisme dari sistem yang akan dibuat.

Bab IV : PEMBUATAN DAN PENGUJIAN SISTEM

Berisi tentang implementasi dari perancangan sistem yang telah dibuat serta pengujian terhadap sistem tersebut.

Bab V : PENUTUP

Merupakan bab terakhir yang memuat intisari dari hasil pembahasan yang berisikan kesimpulan dan saran yang dapat digunakan sebagai pertimbangan untuk pengembangan penulisan selanjutnya.


INSTITUT TEKNOLOGI NASIONAL
ITN
MALANG

BAB II
LANDASAN TEORI

2.1 Pengertian Citra

Citra merupakan hasil keseluruhan suatu sistem perekaman data. Secara teoristis citra dapat dikelompokkan menjadi 2 (dua) macam, yaitu citra kontinu dan citra diskrit (citra digital). Citra kontinu dihasilkan dari sistem optik yang menerima sinyal analog contoh : mata manusia, kamera analog. Sedangkan citra digital dihasilkan melalui proses digitalisasi terhadap citra kontinu. Contoh kamera digital, scanner. (Rinaldi Munir : 2004).

2.1.1. Citra Optik

Citra bersifat optik biasanya disebut citra fotografik yang berbentuk foto. Citra bersifat optik ini secara teoristis merupakan citra kontinu (merekam data secara langsung dalam suatu bidang). Kontinu dalam pengertian nilai keabuan dinyatakan dengan posisi angka tak terhingga.

2.1.2. Citra Analog

Analog berhubungan dengan hal yang kontinu dalam satu dimensi, contohnya adalah bunyi diwakili dalam bentuk analog, yaitu suatu getaran gelombang udara yang kontinu di mana kekuatannya diwakili sebagai jarak gelombang. Hampir semua kejadian alam boleh diwakili sebagai perwakilan analog seperti bunyi, cahaya, air, elektrik, angin dan sebagainya. Jadi citra analog adalah citra yang terdiri dari sinyal-sinyal frekuensi elektromagnetis yang belum dibedakan sehingga pada umunya tidak dapat ditentukan ukuranya.

2.1.3. Citra Digital

Citra digital adalah citra yang dinyatakan secara diskrit (tidak kontinu), baik untuk posisi koordinatnya maupun warnanya. Dengan demikian citra digital dapat digambarkan sebagai suatu matriks, di mana indeks baris dan indeks kolom dari matriks menyatakan posisi suatu titik di dalam citra digital dan harga dari elemen matriks menyatakan warna citra pada titik tersebut. Dalam citra digital yang dinyatakan sebagai susunan matriks seperti ini, elemen-elemenya matriks tadi disebut juga dengan istilah piksel yang berawal dan kata picture element.

Citra digital tersusun atas titik-titik yang biasanya berbentuk persegi panjang yang secara beraturan membentuk baris-baris dan kolom-kolom. Setiap titik memiliki koordinat dan biasanya dinyatakan dalam bilangan bulat positif, yaitu 0 atau 1 bergantung pada sistem yang digunakan. Format nilai pixel sama dengan format citra keseluruhan. Pada kebanyakan sistem pencitraan, nilai ini biasanya berupa bilangan bulat positif juga. Format citra digital yang banyak digunakan, yaitu:

1. Citra Biner (*Monokrom*)

   Citra monokrom atau citra hitam-putih merupakan citra satu kanal dimana citra f(x,y) merupakan fungsi tingkat ke abuan dari hitam ke putih.

2. Citra Skala Keabuan (*Gray Scale*)

   Dikatakan format citra skala keabuan karena pada umunya warna yang dipakai adalah warna hitam sebagai warna minimum dan warna putih sebagai warna maksimalnya, sehingga warna antara ke dua warna tersebut adalah abu-abu.

3. Citra Berwarna

   Citra warna terdiri atas 3 layer matriks, yaitu R-layer, G-layer, B-layer, sistem warna RGB (*Red Green Blue*) menggunakan sistem tampilan grafik kualitas tinggi (*High Quality Raster Graphic*) yaitu mode 24 bit, setiap komponen warna merah, hijau, biru masing-masing mendapatkan alokasi 8 bit untuk menampilkan warna. Pada sistem warna RGB, tiap pixel akan dinyatakan dalam 3 parameter dan bukan nomor warna, setiap warna mempunyai range nilai 00 (angka desimalnya adalah 0) dan f(angka desimalnya 255) atau mempunyai nilai derajat keabuan 256 = 2^8. Dengan demikian, range warna yang digunakan adalah (2^8) (2^8) (2^8) = (2^{24}) (atau dikenal dengan istilah True Color pada Windows). Nilai warna yang digunakan merupakan gabungan warna cahaya merah, hijau dan biru.

2.2 Metode *Eigenface*

Konsep *Eigenface* pertama kali dicetuskan oleh Turk dan Pentland pada tahun 1991. Sebelum memasuki penjelasan lebih rinci mengenai konsep ini, ada baiknya kita menkaji prinsip dasar dan gambaran umum metode *Eigenface*.

Metode *Eigenface* memperlakukan permasalahan pengenalan wajah sebagai pengenalan berdimensi dua, dengan alasan bahwa wajah pada umumnya tampil pada posisi vertical dan oleh karenanya dapat dideskripsikan oleh sekumpulan tampilan karakterisrik dua dimensi. Citra-citra diproyeksikan ke dalam suatu "ruang wajah" yang paling baik mengkodekan variasi diantara citra-citra tersebut. Ruang wajah inilah yang oleh Turk dan Pentland (1991) dijuluki sebagai "*eigenface*", yaitu *eigenvectors* dari kumpulan wajah terkait.

Prinsip dasar dari metode *eigenface* adalah bagaimana caranya untuk mengekstrak informasi yang relevan dari sebuah citra wajah, mengubahnya ke dalam set kode yang paling efisien, dan membandingkan kode wajah ini dengan basis data berisi beragam wajah yang telah dikodekan secara serupa.

Turk dan Pentland menyebutkan bahwa secara matmatis konsep *eigenface* bertujuan untuk mencari principal components dari suatu distribusi wajah-wajah, yaitu *Eigenface* dari matriks kovarian (*covariance matrix*) dari sekumpulan citra wajah. Dalam pembahasan ini principal component dari sebuah matriks adalah vektor-vektor Eigen terbesar.

Lebih lanjut, Eigenvector ini dapat dilihat sebagai sebuah kumpulan fitur yang secara bersama-sama akan mempresentasikan karakter dari variasi diantara citra-citra wajah. Dari gambaran konsep ini dapat dilihat bahwa yang perlu diuraikan dalam pembahasan berikutnya adalah konsep Eigenvector dan matriks kovarain.

2.2.1. Implementasi *Eigenface*

Langkah-langkah yang digunakan dalam metode *Eigenface* adalah sebagai berikut:

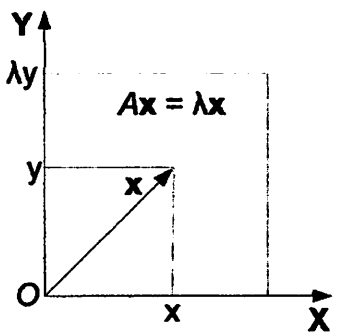
1. Persiapan kumpulan citra latih (*training set*). Setiap citra diubah ke dalam satu vektor, yaitu dengan cara menggabungakn beris-baris pixel dari citra aslinya.citra *grayscale* dengan r buah baris dan c buah kolom misalnya, akan direpresentasikan sebagai vector dengan r x c buah elemen. Seluruh citra di kumpulan ini kemudian disimpan dalam satu matriks T, dimana setiap elemen barisnya mewakili sebuah citra.
2. Perhitungan dan pengurangan citra tengah. Setiap citra asli dalam T harus Dikurangi dengan citra tenggah a.

3. Penghitungan vector Eigen dan nilai Eigen dari matriks kovarian S. Setiap vektor Eigen memilikin dimensi yang sama dengan citra aslinya, sehingga ia dapat dianggap sebagai sebuah citra. Vektor-vektor Eigen dari matriks kovarian inilah yang disebut sebagai *Eigenface*.
4. Pemilihan principal components. Matriks kovarian berukuran D x D akan menghasilkan vector eigen sebanyak D buah. Yang harus kita lakukan adalah menyimpan vector-vektor Eigen dengan nilai Eigen bersesuaian yang terbesar.

2.2.2. *Eigenface* dan Eigenvector

Vector Eigen (Eigenvector) dari sebuah transformasi linear didefinisikan sebagai suatu vector non-nol dimana, ketika transformasi tersebut diaplikasikan kepadanya, vector ini kemungkinan dapat berubah panjangnya, tetapi tidak memungkinkan berubah arahnya.

Untuk lebih jelasnya, diberikan gambar di bawah ini:



Gambar 0.1. Vektor Eigen

Pada gambar di atas, A merupakan sebuah transformasi linear. Vektor non-nol x didefinisikan sebagai Eigenvector dari transformasi A karena ia memenuhi persamaan $Ax = \lambda x$ untuk suatu nilai scalar λ. Pada kasus ini, scalar λ merupakan nilai Eigen (Eigenvalue) dari A yang bersesuaian dengan Eigenvector x.

2.2.3. Principal Component Analysis (*PCA*)

Metode Principal Component Analysis (PCA) dibuat pertama kali oleh para ahli statistik dan ditemukan oleh Karl Pearson pada tahun 1901 yang memakainya ada bidang biologi. Kemudian tidak ada perkembangan baru pada teknik ini, dan. perkembangannya baru mulai pesat pada akhir tahun 1930 dan awal 1940. Setelah itu perkembangannya berkurang sebentar sampai komputer telah berhasil didesain

sehingga dapat mengaplikasikan teknik ini pada masalah-masalah yang masuk akal. Pada tahun 1947 teori ini muncul lagi dan cukup independen sebagai teori probabilitas yang ditemukan oleh Karhunen, dan kemudian dikembangkan oleh Loeve pada tahun 1963, sehingga teori ini juga dinamakan Karhunen-Loeve transform pada bidang ilmu telekomunikasi. PCA adalah sebuah transformasi linier yang biasa digunakan pada kompresi data. PCA adalah sebuah teknik statistika yang berguna pada bidang pengenalan, klasifikasi dan kompresi data citra. PCA juga merupakan teknik yang umum digunakan untuk menarik fitur-fitur dari data pada sebuah skala berdimensi tinggi. Dengan cara mentransformasikan citra ke dalam *eigenfaces* secara linier, proyeksikan citra ke dalam bentuk skala berdimensi n, yang menampakkan properti dari sampel yang paling jelas sepanjang koordinat.

Fitur yang paling signifikan yang ada pada citra akan menjadi principal component yang akan digunakan untuk pengolahan selanjutnya. Dalam prosesnya principal component analysis menggunakan vektor-vektor yang disebut dengan eigenvector dan nilai-nilai yang disebut dengan eigenvalue untuk mendapatkan fitur yang paling signifikan pada dataset. Principal component dicari dengan hubungan:

$$AC = \lambda C$$

di mana A adalah matriks yang akan dicari principal componentnya, C adalah principal component atau disebut dengan eigenvector dan λ adalah eigenvalue. Andaikan A adalah sebuah matriks berdimensi n x n, eigenvalue dari matriks A diperoleh dengan hubungan:

$$\det(A-\lambda I) = 0$$

di mana I adalah matriks identitas dari A dan λ adalah eigenvalue dari matriks A. Mencari nilai *Eigenvector* dapat dicari dengan memecahkan $(A- \lambda I)\ v = 0$, dalam beberapa kasus dapat dijumpai suatu matriks tanpa *eigenvalues*, misalnya:

$$A = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$$

di mana karakteristik polinomial-nya adalah λ2 + 1 sehingga *eigenvalue*-nya adalah bilangan bilangan kompleks *i, -i*. Eigenvalue juga tidak riil. Sebagai contoh diberi matrik citra $A = \begin{bmatrix} 2 & 1 \\ 0 & 3 \end{bmatrix}$, polinomial karakteristiknya dapat dicari sebagai berikut:

$$det\begin{bmatrix} 2-\lambda & 1 \\ 0 & (3-\lambda) \end{bmatrix} = \lambda^2 - 5\lambda + 6 = 0$$

ini adalah persamaan kuadrat dengan akarnya λ1 = 2 dan λ2 = 3. Substitusikan λ2 = 3 ke dalam persamaan. Misalnya Y0 adalah Eigenvector yang berasosiasi dengan Eigenvalue λ2 = 3. Set Y0 dengan nilai:

$$Yo = \begin{bmatrix} X_o \\ Y_o \end{bmatrix}$$

Substitusikan Y0 dengan v pada persamaan:

$$(A - \lambda I)\, v = 0, diperoleh \begin{bmatrix} (2-3)X_o + Y_o = 0 \\ 0 + (3-3)Y_o = 0 \end{bmatrix}$$

dapat disederhanakan lagi menjadi Y0 = -X0

Sehingga *Eigenvector* untuk *Eigenvalue* = 3 adalah

$$Yo = \begin{bmatrix} X_o \\ Y_o \end{bmatrix} = \begin{bmatrix} X_o \\ X_o \end{bmatrix}$$

$$Yo = X_o \begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

2.3 Matriks

Secara informal suatu matriks adalah kumpulan bilangan-bilangan yang berbentuk persegi panjang. Kumpulan bilangan yang disusun berdasarkan baris dan kolom yang didefinisikan dengan jelas. Secara formal suatu matriks A adalah kumpulan suatu objek yang disusun berdasarkan baris dan kolom yang didefinisikan dengan jelas, yakni dapat dituliskan sebagai berikut:

$$A = \begin{bmatrix} a11 & a12 & a13 & \dots & a1n \\ a21 & a22 & a23 & \dots & a2n \\ \dots & \dots & \dots & \ddots & \vdots \\ am1 & am2 & am3 & \dots & amn \end{bmatrix}$$

Bilangan-bilangan $a_{11}, a_{12}, \dots a_{mn}$ yang menyusun rangkaian itu disebut *Elemen* atau *Unsur* dari matriks itu. Indeks pertama menunjukkan *baris* dan indeks ke dua menunjukkan *kolom* di mana elemen itu berada. Ordo sebuah matriks ditentukan oleh banyaknya baris dan kolomnya, matriks *A* di atas mempunyai ordo *m dan n*, ditulis *m* x *n*.

Matriks bujur sangkar adalah matriks yang jumlah baris dan kolomnya sama atau *m* = *n* dan dikatakan berordo *n*. Elemen-elemen dari matriks bujursangkar mulai dari ujung kiri atas sampai ujung kanan bawah secara diagonal yaitu elemen-elemen a_{11}, a_{12}, ..., amn disebut *diagonal utama matriks*. Elemen-elemen dari kiri bawah sampai kanan atas an1, ..., a1n dinamakan diagonal ke dua.

Perkalian Matriks

Ada banyak cara untuk mengalikan dua buah matriks antara lain misalnya hanya mengalikan elemen-elemen yang bersesuaian. Untuk menunjukkan hubungan antara perkalian matriks dengan penggunaan tersebut, perkalian dua buah persamaan berikut:

$$a_{11}x_1 + a_{12}x_2 = d_1$$

$$a_{21}x_1 + a_{22}x_2 = d_2$$

di mana $x1$ dan $x2$ adalah harga-harga yang belum diketahui sedang $a11$, $a12$, $a21$, $a22$, $d1$ dan $d2$ adalah konstanta-konstanta. Persamaan tersebut dapat dituliskan dalam perkalian matriks sebagai berikut:

$$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} = \begin{bmatrix} d_1 \\ d_2 \end{bmatrix}$$

Contoh perkalian matriks, Hasil kali AB untuk matriks A dan B yang diberikan berikut ini:

$$A = \begin{bmatrix} 4 & -1 & 0 \\ 2 & 5 & -3 \end{bmatrix} \quad dan \quad B = \begin{bmatrix} -3 \\ 3 \\ 1 \end{bmatrix}$$

Perkalian A dan B menghasilkan:

$$AB = \begin{bmatrix} 4 \times (-3) - 1 \times 3 + 0 \times 1 \\ 2 \times (-3) - 5 \times 3 - 3 \times 1 \end{bmatrix} = \begin{bmatrix} -15 \\ 6 \end{bmatrix}$$

Pada contoh ini Pada contoh perkalian BA tidak dapat dilakukan karena banyak kolom B tidak sama dengan banyak baris A.

2.4 Sejarah JAVA

Pada 1991, sekelompok insinyu Sun dipimpin oleh Patrick Naughton dan James Gosling ingin merancang bahasa komputer untuk perangkat konsumer seperti *cable TV Box*. Karena perangkat tersebut tidak memiliki banyak memori, bahasa harusberukuran kecil dan mengandung kode yang liat. Juga karena manufaktur – manufaktur berbeda memilih *processor* yang berbeda pula, maka bahasa harus bebas dari manufaktur manapun. Proyek ini diberi nama kode "Green".

Kebutuhan untuk fleksibilitas, kecil, liat dan kode yang netral terhadap *platform* mengantar tim mempelajari implementasi Pascal yang pernah dicoba. Niklaus Wirth, pencipta bahasa Pascal telah merancang bahasa portabel yang menghasilkan *intermediate code* untuk mesin hipotesis. Mesin ini sering disebut

dengan mesin maya *(virtual machine)*. Kode ini kemudian dapat digunakan di sembarang mesin yang memiliki *interpreter*. Proyek Green menggunakan mesin maya untuk mengatasi isu utama tentang netral terhadap arsitektur mesin.(Dasar Pemrograman Berorientasi Objek dengan JAVA 2, 2008)

Karena orang – orang di proyek Green berbasis C++ dan bukan Pascal maka kebanyakan sintaks diambil dari C++, serta mengadopsi orientasi objek dan bukan prosedural. Mulanya bahasa yang diciptakan diberi nama "Oak" oleh James Gosling yang mendapat inspirasi dari sebuah pohon yang berada pada seberang kantornya, namun dikarenakan nama Oak sendiri merupakan nama bahasa pemrograman yang telah ada sebelumnya, kemudian SUN menggantinya dengan JAVA. Nama JAVA sendiri terinspirasi pada saat mereka sedang menikmati secangkir kopi di sebuah kedai kopi yang kemudian dengan tidak sengaja salah satu dari mereka menyebutkan kata JAVA yang mengandung arti asal bijih kopi. Akhirnya mereka sepakat untuk memberikan nama bahasa pemrograman tersebut dengan nama Java.

Bahasa/Alat pengembangan	Arsitektur Program			
	Modul Web Server	Scripting Web Server	Modul Web Browser	Scripting Web Browser
Java	servlet	JSP	Applet	Javascript
C++	CGI exe		ActiveX*	
Perl	CGI script			
Phyton	CGI script			
PHP		PHP script		
Visual Basic		ASP*	ActiveX*	VB Script*

*) Hanya di landasan Windows, tidak bisa di Linux.

Produk pertama proyek Green adalah Star 7 (*7), sebuah kendali jarak jauh yang sangat cerdas. Dikarenakan pasar masih belum tertarik dengan produk konsumer cerdas maka proyek Green harus menemukan pasar lain dari teknologi yang diciptakan. Pada saat yang sama, implementasi WWW dan Internet sedang mengalami perkembangan pesat. Di lain pihak, anggota dari proyek Green juga menyadari bahwa Java dapat digunakan pada pemrograman internet, sehingga penerapan selanjutnya mengarah menjadi teknologi yang berperan di web.

Sebagai sebuah bahasa pemrograman, Java dapat membuat seluruh bentuk aplikasi, desktop, web dan lainnya, sebagaimana dibuat dengan menggunakan bahasa pemrograman konvensional yang lain.

Java adalah bahasa pemrograman yang berorientasi objek (OOP) dan dapat dijalankan pada berbagai platform sistem operasi. Perkembangan Java tidak hanya

terfokus oada satu sistem operasi, tetapi dikembangkan untuk berbagai sistem operasi dan bersifat *open source.*

2.4.1. Pengertian JAVA

Berdasarkan *white paper* resmi dari SUN, Java memiliki karakteristik berikut :

1. Sederhana (*Simple*)

   Bahasa pemrograman Java menggunakan Sintaks mirip dengan C++ namun sintaks pada Java telah banyak diperbaiki terutama menghilangkan penggunaan pointer yang rumit dan *multiple inheritance*. Java juga menggunakan *automatic memory allocation* dan *memory garbage collection.*

2. Berorientasi objek (*Object Oriented*)

   Java mengunakan pemrograman berorientasi objek yang membuat program dapat dibuat secara modular dan dapat dipergunakan kembali. Pemrograman berorientasi objek memodelkan dunia nyata kedalam objek dan melakukan interaksi antar objek-objek tersebut.

3. Terdistribusi (*Distributed*)

   Java dibuat untuk membuat aplikasi terdistribusi secara mudah dengan adanya *libraries* networking yang terintegrasi pada Java.

4. Interpreted

   Program Java dijalankan menggunakan interpreter yaitu *Java Virtual Machine* (JVM). Hal ini menyebabkan *source code* Java yang telah dikompilasi menjadi Java *bytecodes* dapat dijalankan pada platform yang berbeda-beda.

5. Robust

   Java mempuyai reliabilitas yang tinggi. Compiler pada Java mempunyai kemampuan mendeteksi error secara lebih teliti dibandingkan bahasa pemrograman lain. Java mempunyai *runtime-Exception handling* untuk membantu mengatasi error pada pemrograman.

6. Secure

   Sebagai bahasa pemrograman untuk aplikasi internet dan terdistribusi, Java memiliki beberapa mekanisme keamanan untuk menjaga aplikasi tidak digunakan untuk merusak sistem komputer yang menjalankan aplikasi tersebut.

7. Architecture Neutral

   Program Java merupakan *platform independent*. Program cukup mempunyai satu buah versi yang dapat dijalankan pada platform berbeda dengan *Java Virtual Machine*.

8. Portable

   Source code maupun program Java dapat dengan mudah dibawa ke platform yang berbeda-beda tanpa harus dikompilasi ulang.

9. Performance

   Performance pada Java sering dikatakan kurang tinggi. Namun performance Java dapat ditingkatkan menggunakan kompilasi Java lain seperti buatan Inprise, Microsoft ataupun Symantec yang menggunakan *Just In Time Compilers* (JIT).

10. Multithreaded

    Java mempunyai kemampuan untuk membuat suatu program yang dapat melakukan beberapa pekerjaan secara sekaligus dan simultan.

11. Dynamic

    Java didesain untuk dapat dijalankan pada lingkungan yang dinamis. Perubahan pada suatu *class* dengan menambahkan properties ataupun method dapat dilakukan tanpa menggangu program yang menggunakan *class* tersebut.

2.4.2. Sebuah *Deployment Environment*

Terdapat dua komponen utama dari *Deployment Environment*. Yang pertama adalah JRE, yang terdapat pada paket J2SDK, mengandung kelas – kelas untuk semua paket teknologi Java yang meliputi kelas dasar dari Java, komponen GUI dan sebagainya. Komponen yang lain terdapat pada Web Browser. Hampir seluruh Web Browser komersial menyediakan *interpreter* dan *runtime environment* dari teknologi Java.

2.4.3. Sebuah *Development Environment*

Sebagai sebuah peralatan pembangun, teknologi Java menyediakan banyak *tools* : *compiler, interpreter,* penyusun dokumentasi, paket kelas dan sebagainya.

2.4.4. Fitur dari JAVA

Adapun fitur – fitur dari JAVA antara lain adalah:

2.4.5. Java Virtual Machine (JVM)

JVM adalah sebuah mesin imajiner (maya) yang bekerja dengan menyerupai aplikasi pada sebuah mesin nyata. JVM menyediakan spesifikasi hardware dan platform dimana kompilasi kode Java terjadi. Spesifikasi inilah yang membuat aplikasi berbasis Java menjadi bebas dari *platform* manapun karena proses kompilasi diselesaikan oleh JVM.

Aplikasi program Java diciptakan dengan *file* teks berinteraksi. *java.* Program ini dikompilasi menghasilkan satu berkas *bytecode* berektensi. *.class* atau lebih. *Bytecode* adalah serangkaian instruksi serupa intruksi kode mesin. Perbedaannya adalah kode mesin harus dijalankan pada sistem komputer dimana kompilasi ditujukan, sementara *bytecode* berjalan pada *interpreter* yang tersedia di semua *platform* sistem komputer dan sistem operasi.

2.4.6. *Garbage Collection*

Program Java melakukan *garbage collection* yang berarti program tidak perlu menghapus sendiri objek – objek yang tidak digunakan lagi. Fasilitas ini mengurangi beban pengelolaan memori oleh pemrogram dan mengurangi atau mengeliminasi sumber kesalahan terbesar yang terdapat pada bahasa yang memungkinkan alokasi dinamis.

2.4.7. *Code Security*

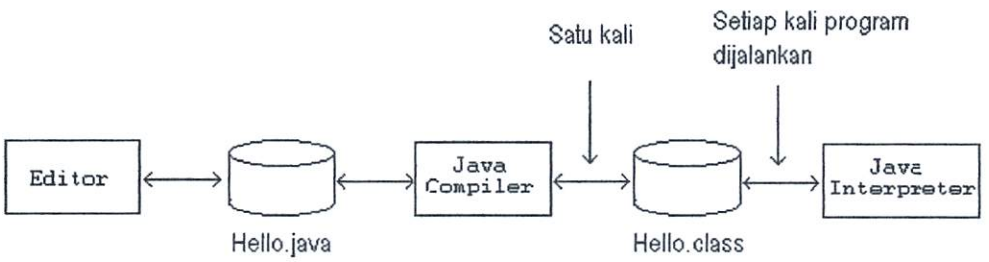
Code Security terimplementasi pada Java melalui penggunaan Java Runtime Environment (JRE). Java menggunakan model pengamanan 3 lapis untuk melindungi sistem dari untrusted Java Code.

1. Pertama, *class-loader* menangani pemuatan kelas Java ke *runtime interpreter.* Proses ini menyediakan pengamanan dengan memisah kelas – kelas yang berasal dari *local disk* dengan kelas – kelas yang diambil dari jaringan.
2. Bytecode verifier membaca bytecode sebelum dijalankan dan menjamin bytecode memenuhi aturan – aturan dasar Java.
3. Manajemen keamanan menangani keamanan tingkat aplikasi dengan mengendalikan apakah program berhak mengakses sumber daya seperti sistem file, port jaringan, proses external dan sistem *windowing.*

Setelah seluruh proses tersebut selesai dijalankan, barulah kode program di eksekusi.

2.4.8. *Fase – fase Pemrograman JAVA*

Gambar dibawah ini menjelaskan aliran proses kompilasi dan eksekusi sebuah program Java :



Gambar : Aliran proses dan eksekusi program JAVA

Langkah pertama dalam pembuatan sebuah program berbasis Java adalah menuliskan kode program pada *text editor*. Contoh *text editor* yang dapat digunakan antara lain : notepad, vi, emacs dan lain sebagainya. Kode program yang dibuat kemudian tersimpan dalam sebuah berkas berekstensi *.java.*

Setelah membuat dan menyimpan kode program, kompilasi file yang berisi kode program tersebut dengan menggunakan Java Compiler. Hasil dari adalah berupa berkas bytecode dengan ekstensi .class.

Berkas yang mengandung bytecode tersebut kemudian akan dikonversikan oleh Java Interpreter menjadi bahasa mesin sesuai dengan jenis dan platform yang digunakan.

Proses	Tool	Hasil
Menulis kode program	*Text editor*	Berkas berekstensi *.java*
Kompilasi program	Java Compiler	Berkas berekstensi *.class* (Java Bytecodes)
Menjalankan program	Java Interpreter	Program Output

2.4.9. NetBeans IDE

Pembuatan Program Java dapa dulakukan dengan menggunakan notepand, akan tetapi untuk meningkatkan produktivitas dan efektifitas dalam proses pengembangan makan IDE (Integrated Devlopment Environment) atau Text Editoryang tepat akan membantu. IDE yang ada bersifat komersial/propiertari, ada juga yang open source. Open source Text Editor Java yang paling populer saat ini adalah Jedit (http://www.jedit.org) sedangkan IDE java yang murni dibuat dengan libary java/ JSDK dari sun adalah NetBeans yang pada penulisan ini mencapai versi 6.8.

Kemampuan NetBeans bisa ditingkatkan dan ditambahkan melalui modul-modul plugin seperti modul untuk game aplication atau beberapa pulgin yang mendukung web service dan masih banyak yang lainnya yang tidak bisa disebutkan dan bahkan kita juga mendapatkan UML Designer untuk keperluan mendesain system yang akan kita buat.

2.5 MySQL (Database)

MySQL ada sebuah perangkat lunak sistem manajemen basis data SQL (*database* management *system*) atau DBMS yang multithread, multi-user, dengan sekitar 6 juta instalasi di seluruh dunia. MySQL AB membuat MySQL tersedia di berbagai perangkat lunak gratis di bawah lisensi GNU *General Public Licence* (GPL), tetapi mereka juga menjual dibawah lisensi komersial untuk kasus – kasus dimana penggunanya tidak cocok dengan penggunaan GPL.

MySQL adalah sebuah implementasi dari sistem manajemen basisdata relasional (**RDBMS**) yang didistribusikan secara gratis dibawah **lisensi GPL** (General Public License). Setiap pengguna dapat secara bebas menggunakan MySQL, namun dengan batasan perangkat lunak tersebut tidak boleh dijadikan produk turunan yang bersifat komersial. MySQL sebenarnya merupakan turunan salah satu konsep utama dalam basisdata yang telah ada sebelumnya; **SQL** (Structured Query Language). SQL adalah sebuah konsep pengoperasian basisdata, terutama untuk pemilihan atau seleksi dan pemasukan data, yang memungkinkan pengoperasian data dikerjakan dengan mudah secara otomatis.

Kehandalan suatu sistem basisdata (**DBMS**) dapat diketahui dari cara kerja pengoptimasi-nya dalam melakukan proses perintah-perintah SQL yang dibuat oleh pengguna maupun program-program aplikasi yang memanfaatkannya. Sebagai peladen basis data, MySQL mendukung operasi basisdata transaksional maupun operasi basisdata non-transaksional. Pada modus operasi non-transaksional, MySQL dapat dikatakan unggul dalam hal unjuk kerja dibandingkan perangkat lunak peladen basisdata kompetitor lainnya. Namun demikian pada modus non-transaksional tidak ada jaminan atas reliabilitas terhadap data yang tersimpan, karenanya modus non-transaksional hanya cocok untuk jenis aplikasi yang tidak membutuhkan reliabilitas data seperti aplikasi blogging berbasis web (wordpress), CMS, dan sejenisnya. Untuk kebutuhan sistem yang ditujukan untuk bisnis sangat disarankan untuk menggunakan modus basisdata transaksional, hanya saja sebagai konsekuensinya unjuk kerja MySQL pada modus transaksional tidak secepat unjuk kerja pada modus non-transaksional.


INSTITUT TEKNOLOGI NASIONAL
ITN
MALANG

BAB III
ANALISA PERANCANGAN SISTEM

3.1. Analisis Sistem

Analisis sistem ini merupakan tahap yang bertujuan untuk memahami system. Yang meliputi tentang desain, implementasi desain dalam mendeteksi wajah menggunakan metode *eigenface*. Pada sub bab ini akan menjelaskan bahwa mengenai aplikasi system pengenalan wajah. Desain aplikasi ini meliputi desain data desain data, algoritma yang di gunakan dalam system yang digambarkan dengan flowchart dan desain proses. Desain data berdasarkan berisikan penjelasan data yang diperlukan untuk dapat menerapkan proses pendeteksi wajah menggunakan metode *eigenface*.

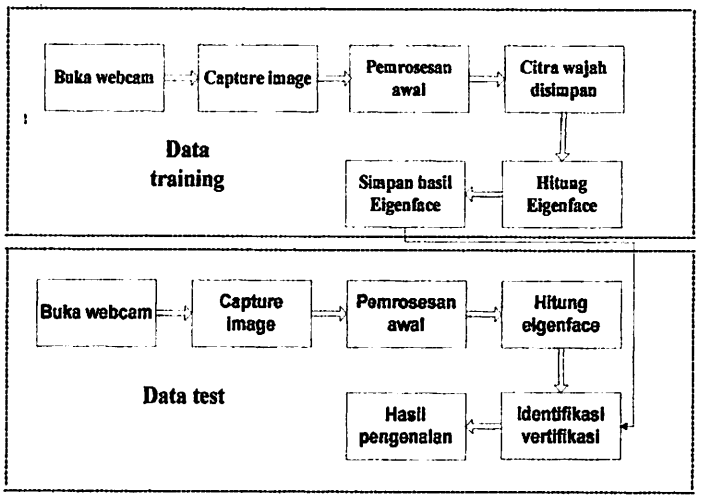
3.2. Analisa Kebutuhan Sistem

Sistem pengenalan wajah yang akan di implementasikan secara keselurahan memiliki kebutahan *hardware* dan *software* sebagai berikut:

1. *Hardware* yang digunakan dalam desain sistem pengenalan wajah ini adalah:
   a. Personal Computer (PC)
   b. *webcam*
2. *Software* yang digunakan dalam desainsisem pengenalan wajah adalah :
   a. Sistem operasi windows XP
   b. Bahasa pemrograman yang digunakan untuk menu tampilan desain adalah JAVA.
   c. Database yang digunakan adalah MySQL.

3.3. Perancangan Sistem

Berikut ini adalah tentang perancangan system pengenalan wajah. Pengenalan wajah secara umum memiliki beberapa tahap yaitu tahap pengambilan citra, lalu di lanjutkan dengan tahap perhitungan *eigenface*, setelah kedua tahap ini selesai maka di lanjutkan tahap ketiga yaitu tahap pengenalan wajah.



Gambar 3.1 blok diagram system pengenalan wajah

Dari gambar blok diatas merupakan blok diagram *software* untuk pengenalan dari system yang digunakan, dimana jalanya system dibagi menjadi 2, yaitu blok pengabilan data dan blok pengenalan wajah. Jalanya system blok pengambilan data adalah sebagai berikut :

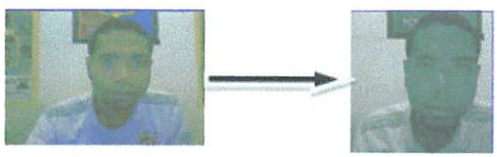
1. Aktifkan *webcam* untuk menampilkan gambar yang ditangkap *webcam* kedalam aplikasi
2. Penangkapan citra wajah (*image campturing*) dapat dilakukan secara langsung (*real time*) mengunakan *webcam*, setelah terdeteksi adanya gambar wajah pada tampilan windows dari *webcam*.
3. Kemudian dilakukan proses pemrosesan awal yang meliputi, normalisasi ukuran citra, RGB ke *grayscale*.
4. Simpan data wajah yang di ambil.
5. Kemudian dilakukan proses PCA untuk mengutip bagian terpenting dengan metode *eigenface* sehinga di dapatkan *eigenvector* dan *eigenvalue* dari gambar tersebut.
6. Data yang telah disimpan nantinya digunakan sebagai nilai pembanding pada proses perhitungan jarak pada pengenalan wajah.

Sedangkan untuk proses pengenalan wajah adalah sebagai berikut :

1. Aktifkan *webcam* untuk menampilkan gambar yang ditangkap *webcam* kedalam aplikasi.
2. Penangkapan citra wajah (*image campturing*) dapat dilakukan secara langsung (real *time*) mengunakan *webcam*, setelah terdeteksi adanya gambar wajah pada tampilan windows dari *webcam*.
3. Kemudian dilakukan proses pemrosesan awal yang meliputi, normalisasi ukuran citra, RGB ke *grayscale*, historgram equalization untuk memperbaiki kualitas citra input agar memudahkan proses pengenalan tampa menghilangkan informasi utamanya, resize untuk membuang bagian daerah selain wajah sehinga hanya bagian wajah saja yang diproses dan normalisasi pencahayaan ketika mengambil citra input.
4. Kemudian dilakukan proses PCA untuk mengutip bagian terpenting dengan metode *eigenface* sehinga di dapatkan *eigenvector* dan eigenvalue dari gambar tersebut.
5. Proses pengenalan wajah dengan menghitung jarak antara fitur wajah yang ada dalam data dengan fitur wajah baru. Jarak yang dapat dicari yang terkecil untuk identifikasi.

3.3.1. Pemrosesan awal

Konversi *image* ke *grayscale*



Sebelum citra diproses dengan masing-masing algoritma,citra wajah diproses dengan proses trianing. Proses training dilakukan dengan mengubah *image* RGB menjadi *image grayscale* untuk mendapatkan suatu vector cirri dan mendapatkan jarak distance yang nantinya akan digunakan pada proses pengenalan dengan masing-masing algoritma.

Tahap *grayscale* citra adalah kegiatan untuk menyederhanakan model citra . tiga layer pada citra warna. R-layer, G-layer dan B-layer diubah menjadi satu layer *grayscale*. Untuk mengubah citra berwarna yang memiliki nilai matriks masing-

1. Abfolge der Bilder mit Nummern versehen
2. Bildbeschreibung mit Angabe der Nummer
3. Kennzeichnung der Bilder, die nicht zur Bildgeschichte gehören
4. Erzählung schriftlich in eigenen Worten wiedergeben
5. Bilder beschneiden, sodass die Geschichte klar erkennbar ist

masing R, G, dan B menjadi citra *grayscale* dengan membagi jumlah ketiga layer. Dengan persamaan dibawah ini.

$$f_0(x,y) = \frac{f_i^R(x,y) + f_i^G(x,y) - f_i^B(x,y)}{3}$$

Gambaran proses konversi RGB ke dalam *grayscale* melalui perhitungan manual. Misalkan kita memiliki citra berukuran 3x4 piksel dengan nilai RGBnya sebagai berikut :

(x,y)	0	1	2	3
0	2 3 2	4 3 3	7 2 3	6 7 3
1	3 1 2	3 2 5	4 5 3	5 4 6
2	3 2 1	2 4 4	3 2 6	3 7 5

Tabel 1 konversi manual RGB ke *Grayscale*

$$fo(0,0) = \frac{2+3+2}{3} = 2{,}3$$

$$fo(0,1) = \frac{4+4+3}{3} = 3{,}3$$

$$\ldots\ldots\ldots\ldots\ldots\ldots$$

$$fo(2,3) = \frac{3+7+5}{3} = 5$$

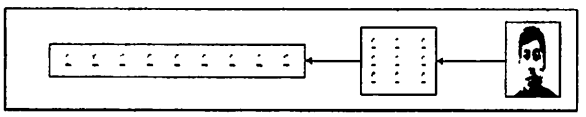
Perhitungan tersebut dilakukan untuk setiap kordinat, sehingga matriks citra hasil konversi sebagai berikut :

x,y	1	2	3	4
0	2	3	4	5
1	2	3	4	5
2	2	3	4	5

Setelah didapatkan citra yang sudah ter*grayscale*, selanjutnya citra diproses dengan mengubahnya menjadi vector matriks untuk mendapatkan nilai citra vector citra tersebut.

3.3.2. Perhitungan Proses Pengenalan Wajah

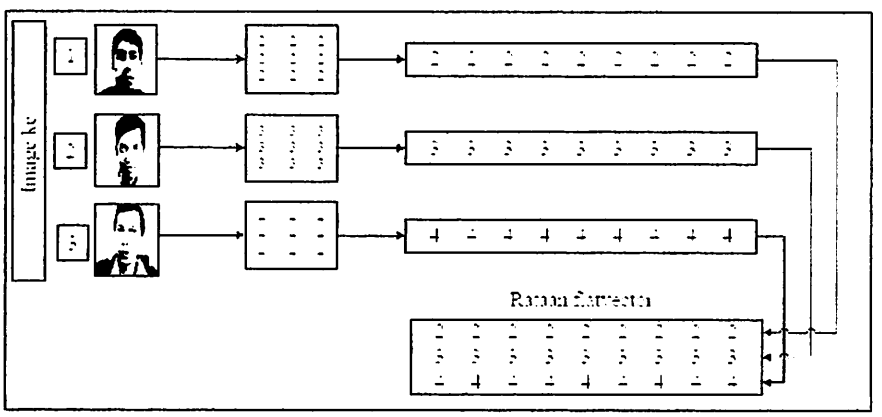
Setelah didapatkan *image* ,selajutnya ubah menjadi vector satu dimensi untuk mempermudah dalam perhitungan distance dari setiap *image* yang nantinya akan digunakan dalam proses pengenalan. Misalkan dalam traning *image*, terdapat *image* dengan ukuan 3x4 piksel maka kita akan mempunyai *flatvector* dengan ukuran 1x9.



Gambar 3.3 proses reshape 2D ke 1D vector *image*

1. Penyusunan *Flatvector*

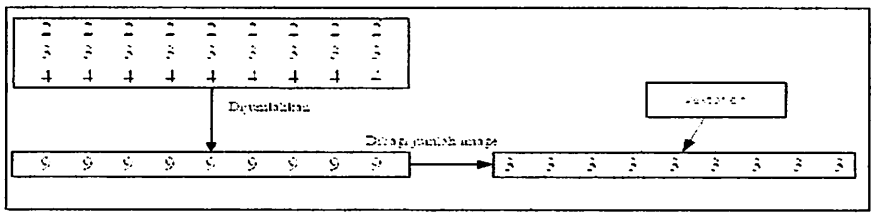
Selanjutnya adalah menyusun seluruh traning *image* menjadi satu matriks tunggal. Missal *image* yang disimpan berukuran H x W dan jumlah N buah, sehinga memiliki vector cirri dimensi N x (W x H). Terdapat 3 *image* dengan ukuran 3 x 4 piksel, maka akan mempunyai *eigenvector* ukuran 3 x 9 .



Gambar 3.4 Penyusunan Flatvektor

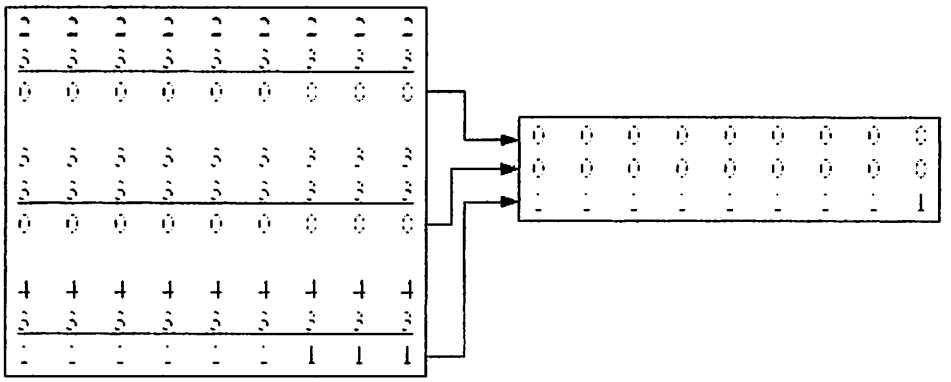
2. Perhitungan Rataan Flatvektor

Dari vector cirri yangtelah diperoleh jumlah seluruh barisnya sehinga di peroleh matriks berukuran 1 x (WxH). Setelah itu bagi matriks tersebut dengan jumlah *image*, untuk mendapatkan rata-rata vector cirri



Gambar 3.5 nilai rata-rata flatvektor

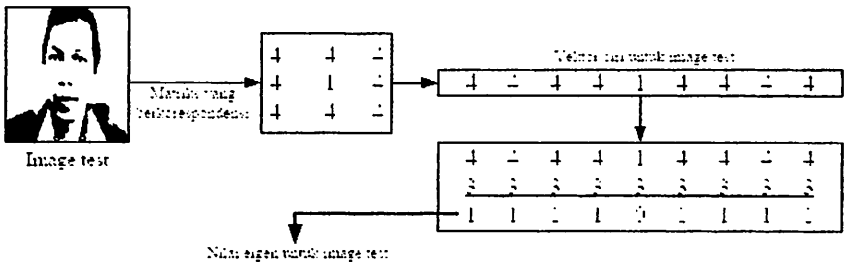
3. Proyeksi *image* ke facespace, proyeksi vector wajah akan dibandingkan dengan vector sesuai. Dngan mengunakan rata-rata vector cirri, akan dihitung *eigenface* untuk matriks vector cirri yang telah disusun, caranya dengan mengurangi baris-baris pada matriks vector ciri jika didapatkan nilai dibawah nol akan diganti dengan nol.



Gambar 3.6 Perhitungan *Eigenface*

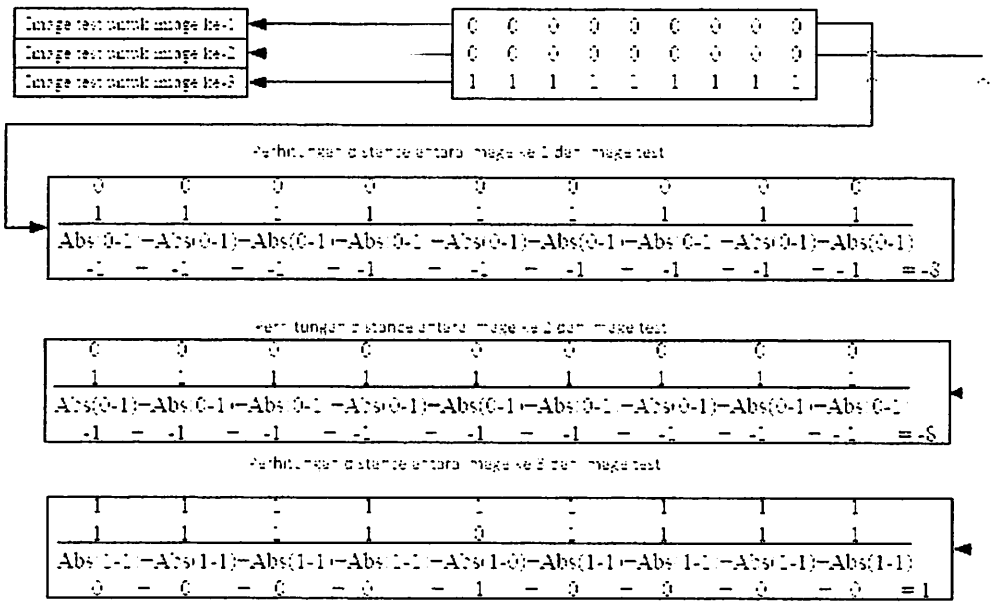
4. Ektraksi PCA *image* test

Hasil proyeksi tersebut diektraksi dengan perhitungan PCA untuk mendapatkan feture dari *image*, feature adalah komponen-komponen penting *image-image* traning yang didapat dari proses traning, feature-feature inilah yang nantinya akan digunakan untuk menidentifikasi *image* yang akan dikenali. Kalkulasi nilai *eigenface* untuk matriks *eigenface*, dengan cara yang sama dengan penentuan *eigenface* untuk vector ciri.



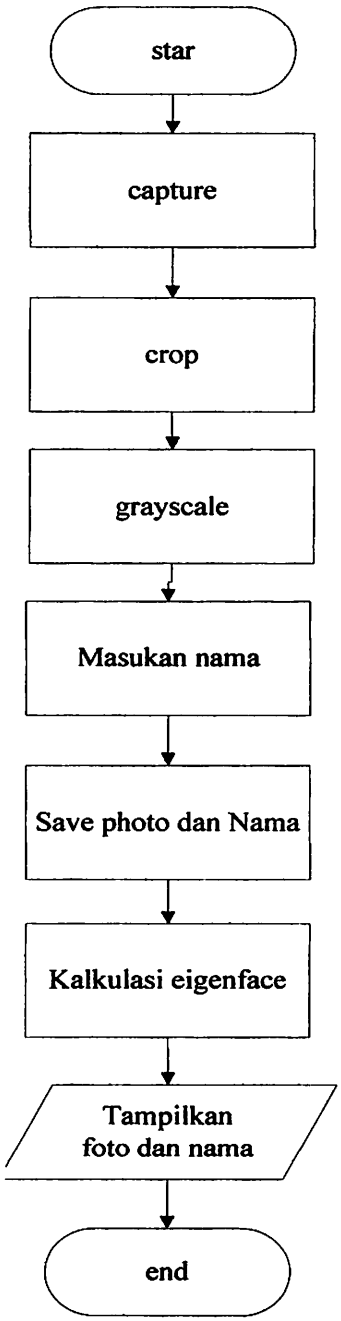
Gambar 3.7 Perhitungan Untuk *Eigenface* Testface

5. Cari nilai distance minimum antara *image* test dan *image* traning, bandingkan nilai distance minimum dengan *image* yang di capture dengan *image* yang sudah ada di database, setelah nilai *eigenface* untuk *image* test diperoleh maka kita bias dilakukan identifikasi dengan menentukan jarak dengan *eigenface* dari *eigenvector* traning *image*. Caranya dengan menentukan nilai absolute pada pengurangan *i* pada matriks *eigenface* traning *image* dengan ignface test *image*. Kemudian jumlahkan elemen-elemen penyusun vector yang dihasilkan dari pengurangan tadi dan temukan jarak d indeks *i*. lakukan untuk semua baris. Cari nilai *d* yang paling kecil



Gambar 3.8 Proses Identifikasi input *image* test

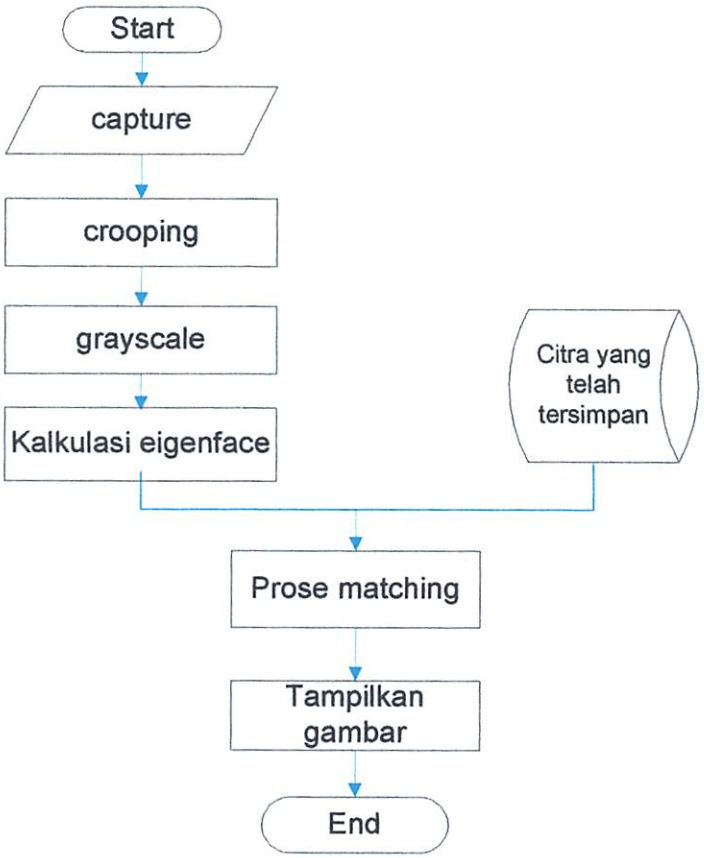
3.4. Proses untuk memasukan data traning berupa Foto dan Nama



Gambar 3.4 Flowchard data traning

3.5. Diagram Flowchard Pengenalan wajah

Proses pengenalan wajah ini di lakukan pada saat sudah ada data kumpulan data citra training.



Gambar 3.5 Flowchard Proses Pengenalan wajah

3.5.1. Perangsangan DataBase

Database yang diggunakan adalah MySQL. Database terdiri dari table id untuk penyimpanan data dari citra wajah dan table setting ubuk menyimpan nilai persentase terdekat ketika mengenali wajah. Pada table id, yang menjadi primary key adalah id dengan data type auto number sebagai tempat untuk menyimpan indeksnya yang digunakan untuk indeks penyimpanan citra wajah. Nrp dan nama sebagai data type text digunakan untuk menyimpan data dari data wajah. Pada table setting tidak ada primary key karena hanya akan berisi satu data saja pada nama.

Table 1 Database id

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

Table 2 database setting

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


INSTITUT TEKNOLOGI NASIONAL
ITN
MALANG

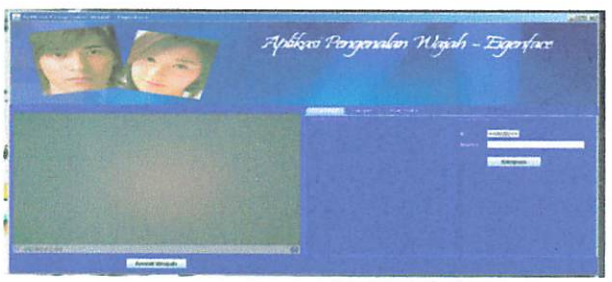
BAB IV

IMPLEMENTASI PENGUJIAN SISTEM

4.1. Form Menu Utama

4.1.1. Form Menu Utama

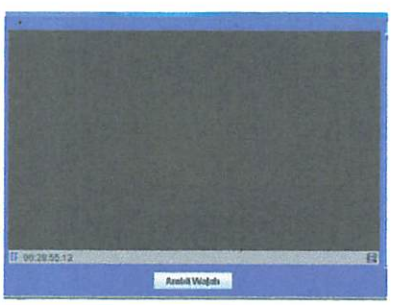
Tampilam awal dari Aplikasi Sistem Biometrika Identifikasi Wajah dengan Algoritma *Eigenface* yaitu tampilan Form Main Menu Terdiri dari tombol Ambil wajah untuk menuju ke form input, tombol Training untuk menuju menampilkan gambar yang telah diinput, tombol Target untuk menampilkan gambar target sebagai data test, Tombol Lihat Data untuk melihat data yang telah disimpan dalam database, Tombol Simpan untuk menyipan data.



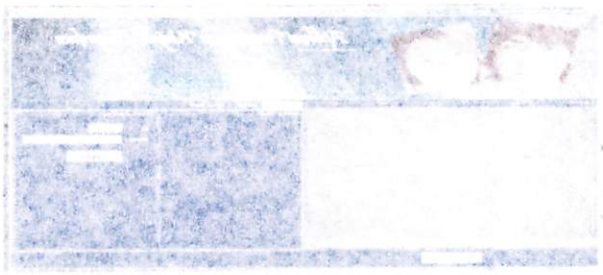
Gambar 4.1 Tampilan Main Menu

4.1.2. Form Ambil Wajah

From ini berfungsi untuk menampilkan gambar yang akan dicapture

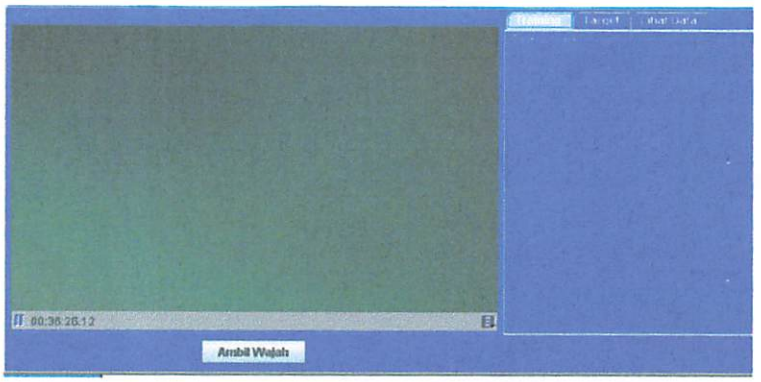


Gambar 4.2 Tampilan Form Input citra wajah



4.1.3. Form Taraget

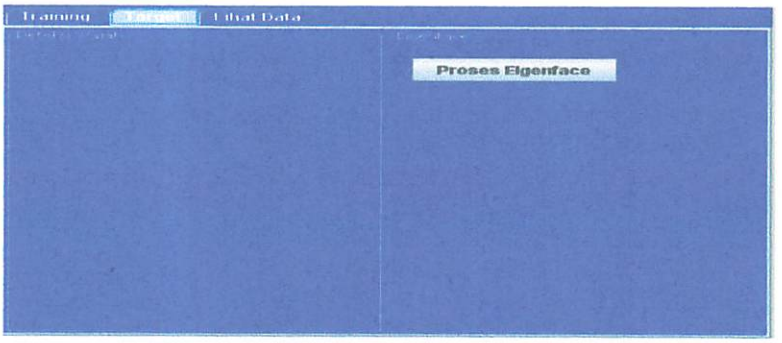
Proses data target menggunakan webcam untuk melakukan capture deteksi gambar wajah seperti pada saat pengisian foto untuk data mahasiswa. Hasil capture kemudian dicocokkan dengan foto taining yang ada pada tabel citra latih dan dilakukan proses identifikasi dengan mencocokan data yang ada.



Gambar 4.3 Tampilan Form Target

4.1.4. Form Proses *Eigenface*

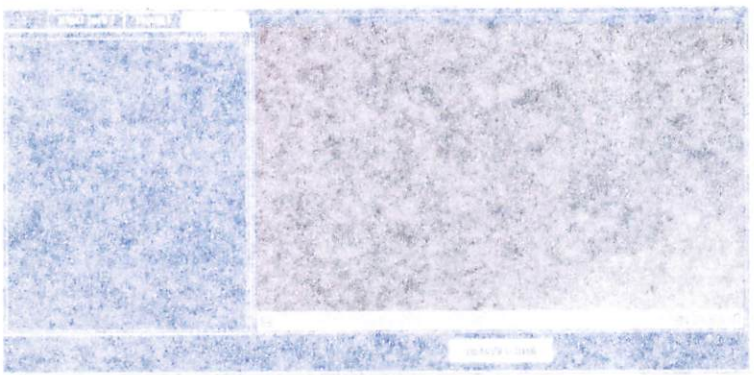
Pada Form Proses *Eigenface* digunakan untuk proses matching wajah yaitu mencari citra wajah yang mirip, yang telah di simpan.

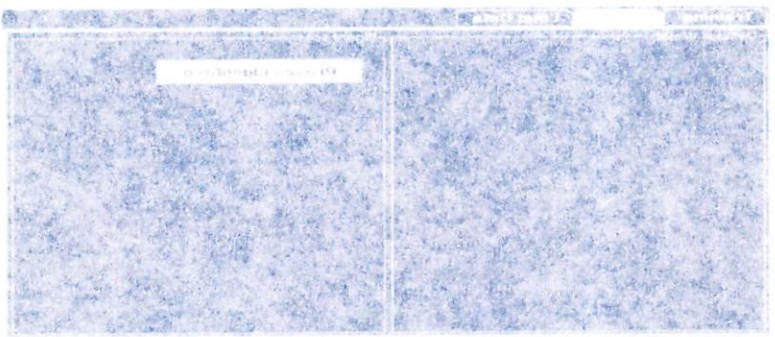


Gambar 4.4 proses *s*

4.2. Implementasi dan Pengujian Aplikasi

Tahap implementasi merupakan lanjutan dari tahap analisa sistem dan perancangan sistem yang sudah dibahas di bab sebelumnya, dalam aplikasi pengenalan wajah menggunakan metode *Eigenface*.





4.2.1. Pengujian untuk capture wajah

Posisi wajah di depan camera selanjutnya untuk mencapture Wajah dengan menekan tombol Ambil Wajah.

Gambar 4.5 Tampilan hasil capture

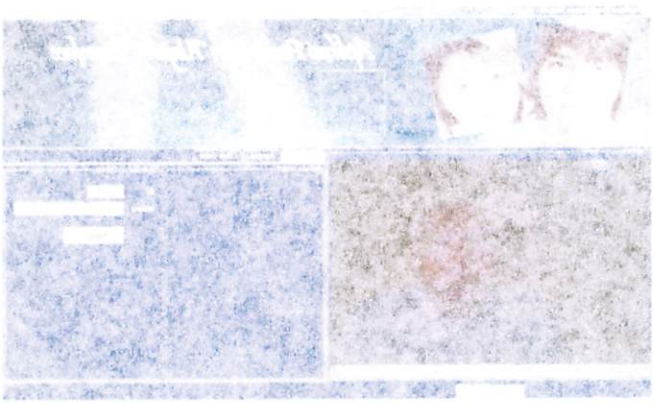
4.2.2. Hasil Grayscale

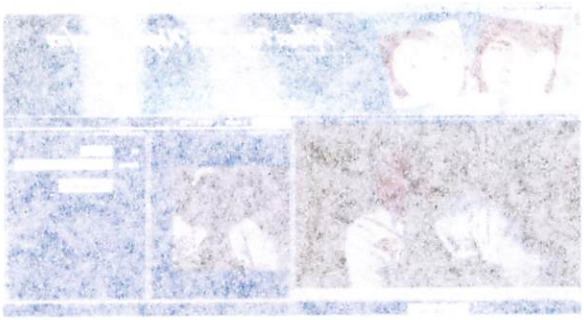
Hasil image RGB yang telah di telah dirubah ke grayscale

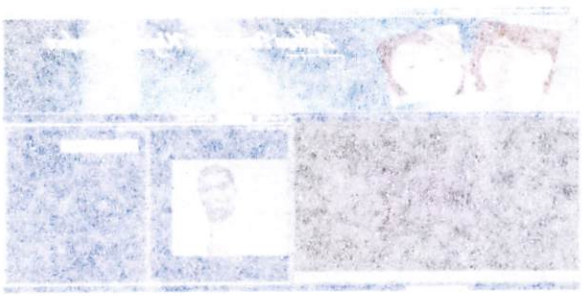
Gambar 4.6 Tampilan Proses grayscale

4.2.3. Proses *Eigenface*

Gambar 4.7 Tampilan proses *Eigenface*

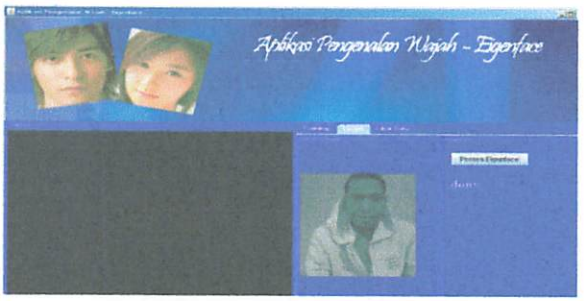






4.2.4. Hasil proses *Eigenface*

Hasil proses *Eigenface* ini yang akan menampilkan image dan nama yang cocok dengan data traning yang tersimpan dalam datbase.



Gambar 4.8 hasil image yang cocok

4.3. Pengujian

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

Keterangan :

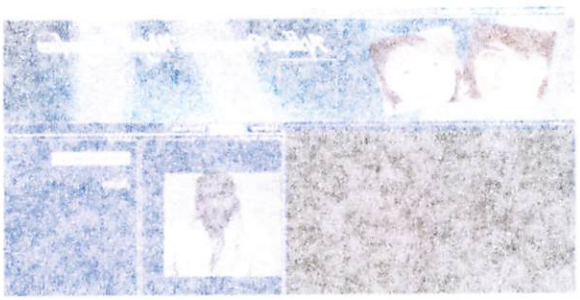
√ = wajah teridentifikasi

X = wajah yang teridentifikasi tidak sesuai

Tabel 3 Data Pengamatan I

Percobaan Ke-	I	II	III	IV	V
Alcino	√	√	√	√	√
Hercio	√	√	X	X	X
Natalino	√	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 sebenyak 10 kali.

Tabel 4 Data Pengamatan II

Percobaan Ke-	I	II	III	IV	V	VI	VII	VIII	IX	X
Alcino	√	√	√	√	√	√	√	√	√	√
Hercio	√	√	√	X	√	√	√	√	√	√
Natalino	√	√	√	√	X	√	X	√	√	√
Martha	√	√	√	√	√	√	√	√	√	√
Jimmy	√	√	√	√	√	√	√	√	√	√

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

Data pengamatan ke-3 didapat berdasarkan hasil percobaan dengan *background* yang sama yaitu berwarna biru, pencahayaan yang terang, 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 5 Data Pengamatan III

Percobaan Ke-	I	II	III	IV	V
Alcino	√	√	√	√	√
Hercio	√	√	X	X	√
Natalino	√	X	√	X	√
Martha	√	√	√	X	√
Jimmy	√	√	X	√	√

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

Percobaan Ke-	**I**	**II**	**III**	**IV**	**V**	**VI**	**VII**	**VIII**	**IX**	**X**
Alcino	√	√	√	√	√	√	√	√	√	√
Hercio	√	√	√	√	√	√	√	√	√	√
Natalino	√	√	√	√	X	√	X	√	√	√
Martha	√	√	√	√	√	√	√	√	√	√
Jimmy	√	√	√	√	√	√	√	√	√	√

Dari percobaan ke-4, 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-4 didapdat sampai 96%.


INSTITUT TEKNOLOGI NASIONAL
ITN
MALANG

BAB V
PENUTUP

5.1. Kesimpulan

Dari hasil penelitian, perancangan dan implementasi yang telah dilakukan maka dapat disimpulan bahwa :

1. Proses Aplikasi Sistem Biometrika Identifikasi Wajah dengan Algoritma *Eigenface*, dilakukan dengan melakukan input citra wajah dan input data pada data training. Kemudian saat data test dilakukan deteksi wajah, jika hasil cocok dengan data citra wajah pada database. Maka proses pengenalan wajah berhasil dan dinyatakan cocok.
2. Jarak antara *webcam* berpengaruh dalam proses pendeteksian wajah, jarak antara camera dengan wajah sekitar 10 cm. Jarak minimal 7 cm dan Jarak maximun 9,07 cm. Dibawah atau lebih dari jarak tersebut maka hasilnya akan kabur.
3. Proses pengenalan akan berjalan dengan baik bila capture wajah hasil deteksi jelas dan tidak kabur.

5.2. Saran

1. Agar pengembangan sistem dilakukan dengan cara membadingkan dengan metode pengenalan wajah lainnya.misalnya metode Quickprop pada jaringan syarat tiruan.
2. Penggunaan perancangan *software* dengan java mengalami kesulitan, karena tidak adanya operasi-operasi interval dalam perhitungan matriks. Kesulitan lainnya adalah belum adanya komponen untuk mengendalikan *webcam*, misalnya bagaimana agar webcam dapat melakukan capture gambar. Sebaiknya menggunakan *software* dengan komponen yang lebih lengkap.


INSTITUT TEKNOLOGI NASIONAL
ITN
MALANG

DAFTAR PUSTAKA

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

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:resource.unpad.ac.id/unpad-content/uploads/publikasi_dosen_PCA. (22 juli 2010)

Romdhani s., face recognition using Principal Component Analysis, http://eeapp.elec.gla.ac.uk/~romdhni/pca_doc/pca_doc_toc.htm.1997 **Diakses**

Wibowo, B,B.,pngenalan wajah menggunakan *(component analysis principal),* Tugas Akhir Mahasiswa S-1 Teknik Elektro Univrsitas Diponegoro, Smarang, 2005.

Prasetyo, Edi., " *Desain variasi wajah dengan variasi ekspresi dan posisi menggunakan Metode Eigenface*" Makalah Skripsi Universitas Gunadrama, Jakarta,2005. http://prasedi.blogspot.com/2005/05/yz-tanma-2.html **Diakses**

Mukit, Garailbaldi W, "*Implementasi Algoritma Metode Eigenface untuk face Recognition",* Makalah Skripsi Institut Teknologi Nasional Bandung, Bandung 2006.


INSTITUT TEKNOLOGI NASIONAL
ITN
MALANG

LAMPIRAN

PERKUMPULAN PENGELOLA PENDIDIKAN UMUM DAN TEKNOLOGI NASIONAL MALANG
INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSTRI
FAKULTAS TEKNIK SIPIL DAN PERENCANAAN
PROGRAM PASCASARJANA MAGISTER TEKNIK
PT. BNI (PERSERO) MALANG
BANK NIAGA MALANG
Kampus I : Jl. Bendungan Sigura-gura No. 2 Telp. (0341) 551431 (Hunting), Fax. (0341) 553015 Malang 65145
Kampus II : Jl. Raya Karanglo, Km 2 Telp. (0341) 417636 Fax. (0341) 417634 Malang

BERITA ACARA UJIAN SKRIPSI
FAKULTAS TEKNOLOGI INDUSTRI

Nama	:	**Natalino Hercio S Dc Amaral**
Nim	:	**07.12.535**
Jurusan	:	**Teknik Elektro**
Konsentrasi	:	**Teknik Informatika & Komputer S-1**
Masa Bimbingan	:	**Semester Genap 2012**
Judul	:	**Aplikasi Sistem Pengenalan Wajah Menggunakan Metode Eigenface Berbasis J2SE**

Dipertahankan dihadapan Tim Penguji Skripsi Jenjang Program Strata Satu (S-1)

Pada Hari : Rabu
Tanggal : 08 Agustus 2012
Dengan Nilai : 71,3 (B+)

PANITIA UJIAN SKRIPSI

Ketua Majelis Penguji

Ir. Yusuf Ismail Nakhoda, MT
NIP.Y.1018800189

Sekretaris Majelis Penguji

Dr. Eng. Aryuanto Soetedjo, ST. MT
NIP.Y.1030800417

ANGGOTA PENGUJI

Dosen Penguji I

Yuli Wahyuni, ST. MT
NIP.Y.1031200456

Dosen Penguji II

Ir.Eko Nurchayo, MT
NIP.Y.1039700309

BENUTZUNGSVEREINBARUNG FÜR EIGENE FAHRZEUGE

Name	Nachname Vorname 5 Buchstaben
Tag	07.12.2023
Nummer	Laufende Nummer
Kennzeichen	Fahrzeug Kennzeichen & Kennzeichen 5-7
Name Mitarbeiter	Mitarbeiter Kürzel 2012
Modell	Modell Name Fahrzeug Typ Hersteller
	Hersteller Fahrzeug Typ Modell

Unterschreiben Sie Ihre Fragen Sie Ihre Fragen Sie Ihre Fragen

Ihre Frage	Ihre Frage
Ihre Frage	Ihre Frage
Ihre Frage	Ihre Frage

Ihre Antwort

Ihre Antwort

Ihre Antwort

Ihre Antwort

Ihre Antwort

Ihre Antwort

Ihre Antwort

Ihre Antwort

Ihre Antwort

Ihre Antwort

PERKUMPULAN PENGELOLA PENDIDIKAN UMUM DAN TEKNOLOGI NASIONAL MALANG
INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSTRI
FAKULTAS TEKNIK SIPIL DAN PERENCANAAN
PROGRAM PASCASARJANA MAGISTER TEKNIK
PT. BNI (PERSERO) MALANG
BANK NIAGA MALANG
Kampus I : Jl. Bendungan Sigura-gura No. 2 Telp. (0341) 551431 (Hunting), Fax. (0341) 553015 Malang 65145
Kampus II : Jl. Raya Karanglo, Km 2 Telp. (0341) 417636 Fax. (0341) 417634 Malang

FORMULIR PERBAIKAN SKRIPSI

Dalam pelaksanaan Ujian Skripsi Jenjang Strata 1 Jurusan Teknik Elektro Konsentrasi Teknik Informatika & Komputer, maka perlu adanya perbaikan skripsi untuk mahasiswa :

Nama	: **Natalino Hercio S Da Costa Amaral**
Nim	: **07.12.535**
Jurusan	: **Teknik Elektro S-1**
Konsentrasi	: **Teknik Informatika dan Komputer**
Masa Bimbingan	: **Semester Genap 2011-2012**
Judul Skripsi	: **APLIKASI SISTEM PENGENALAN WAJAH MENGGUNAKAN METODE EIGENFACE BERBASIS J2SE**

No	Penguji	Tanggal	Uraian	Paraf
1.	Penguji I	02 Agustus 2012	Perbaikan Laporan Bab I: • Tujuan pada laporan singkronkan dengan rumusan masalah • Batasan maslah diperbaiki	

Disetujui:

Penguji I

Yuli Wahyuni,ST, MT
NIP.P.1031200456

Penguji II

Ir. Eko Nurcahyo, MT
NIP. Y. 1028700172

Mengetahui:

Dosen Pembimbing I

Irmalia Suryani Faradisa,ST, MT
NIP.P. 1030000365

Dosen Pembimbing II

Sotyohadi, ST
NIP.Y. 10397003091

PENGATURAN PERMULAAN SKRIPSI

Dalam pelaksanaan Ujian Skripsi Jurusan Sistem Informasi Fakultas Teknik Universitas Kristen Satya Wacana maka perlu diatur permulaan skripsi untuk mahasiswa :

Nama : Nanang Haryo Bimo
Nim : 672012035
Jurusan : Sistem Informasi
Konsentrasi : Sistem Informasi dan Komputer
Tahun Pelajaran : 2011-2012
Judul Skripsi : APLIKASI SISTEM PENGENALAN WAJAH MENGGUNAKAN METODE EUCLIDEAN DISTANCE

No	Progres	Tanggal	Hasil	Paraf
1.	Progres I	02 Agustus 2012	Perbaikan Laporan Bab I • Lapiran pada lapiran ditambahkan dengan metode/ rancangan • Biaya dan bab lainnya dipertebal	

Mengetahui,

Pembimbing I

Ir. Ida Nurcahyo, MT
NIP. K. 1028700172

Pembimbing II

Ir. Hendra Supriyanto, MT
NIP. K. 1028700172

Menyetujui,

Dosen Pembimbing I

Ir. Ida Nurcahyo, MT
NIP. K. 1028700172

Dosen Pembimbing II

Ir. Hendra Supriyanto, MT
NIP. K. 1028700172

I ERKUMPULAN PENGELOLA PENDIDIKAN UMUM DAN TEKNOLOGI NASIONAL MALANG
INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSTRI
FAKULTAS TEKNIK SIPIL DAN PERENCANAAN
PROGRAM PASCASARJANA MAGISTER TEKNIK
NI (PERSERO) MALANG
ANK NIAGA MALANG
Kampus I : Jl. Bendungan Sigura-gura No. 2 Telp. (0341) 551431 (Hunting), Fax. (0341) 553015 Malang 65145
Kampus II : Jl. Raya Karanglo, Km 2 Telp. (0341) 417636 Fax. (0341) 417634 Malang

Nomor Surat : ITN-205/EL-FTI/2012
Lampiran : -
Perihal : BIMBINGAN SKRIPSI

Kepada : Yth. Bapak/Ibu **Irmalia Suryani Faradisa, ST, MT**
Dosen Teknik Elektro S-1
ITN MALANG

Dengan Hormat

Sesuai dengan permohonan dan persetujuan dalam Proposal Skripsi untuk mahasiswa :

Nama	: **NATALINO HERCIO SILVANIA DA COSTA AMARAL**
Nim	: **0712535**
Fakultas	: **Teknologi Industri**
Program Studi	: **Teknik Elektro S-1**
Konsentrasi	: **Teknik Komputer & Informatika**

Maka dengan ini pembimbingan tersebut kami serahkan sepenuhnya kepada Saudara/i selama masa waktu :

" Semester Genap Tahun Akademik 2011-2012 "

Demikian agar maklum dan atas perhatian serta bantuannya kami sampaikan terima kasih.

Mengetahui
Ketua Program Studi Teknik Elektro S-1

Ir. Yusuf Ismail Nakhoda, MT
NIP.Y. 1018800189

PROGRAM STUDI TEKNIK ELEKTRO S-1
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI NASIONAL MALANG
Kampus II : Jl. Raya Karanglo Km. 2 Telp. (0341) 417636 Malang

Lampiran : 1 (satu) berkas
Pembimbing Skripsi

Kepada : Yth. Bapak/Ibu **Irmalia Suryani Faradisa, ST, MT**
Dosen Teknik Elektro S-1
ITN Malang

Yang bertanda tangan dibawah

Nama	: **NATALINO HERCIO SILVANIA DA COSTA AMARAL**
Nim	: **0712535**
Jurusan	: **Teknik Elektro S-1**
Konsentrasi	: **Teknik Komputer & Informatika**

Dengan ini mengajukan permohonan, kiranya Bapak/Ibu bersedia menjadi Dosen Pembimbing untuk penyusunan Skripsi dengan judul :

"APLIKASI SISTEM PENGENALAN WAJAH MENGGUNAKAN METODE EIGEN FACE BERBASIS J2SE "

Demikian permohonan kami buat dan atas kesediaan Bapak kami ucapkan terima kasih.

Mengetahui
Ketua Program Studi Teknik Elektro S-1

Ir. Yusuf Ismail Nakhoda, MT
NIP.Y. 1018800189

Hormat Kami

NATALINO HERCIO SILVANIA DA COSTA AMARAL
NIM. 0712535

PROGRAM STUDI TEKNIK ELEKTRO S-1
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI NASIONAL MALANG
Kampus II : Jl. Raya Karanglo Km. 2 Telp. (0341) 417636 Malang

Lampiran : 1 (satu) berkas
Pembimbing Skripsi

Kepada : Yth. Bapak/Ibu **Sotyohadi, ST**
Dosen Teknik Elektro S-1
ITN Malang

Yang bertanda tangan dibawah

Nama	: **NATALINO HERCIO SILVANIA DA COSTA AMARAL**
Nim	: **0712535**
Jurusan	: **Teknik Elektro S-1**
Konsentrasi	: **Teknik Komputer & Informatika**

Dengan ini mengajukan permohonan, kiranya Bapak/Ibu bersedia menjadi Dosen Pembimbing untuk penyusunan Skripsi dengan judul :

"APLIKASI SISTEM PENGENALAN WAJAH MENGGUNAKAN METODE EIGEN FACE BERBASIS J2SE "

Demikian permohonan kami buat dan atas kesediaan Bapak kami ucapkan terima kasih.

Mengetahui
Ketua Program Studi Teknik Elektro S-1

Ir. Yusuf Ismail Nakhoda, MT
NIP.Y. 1018800189

Hormat Kami

NATALINO HERCIO SILVANIA DA COSTA AMARAL
NIM. 0712535

PROGRAM STUDI TEKNIK ELEKTRO S-1
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI NASIONAL MALANG
Kampus II : Jl. Raya Karanglo Km. 2 Telp. (0341) 417636 Malang

PERNYATAAN KESEDIAAN DALAM PEMBIMBINGAN SKRIPSI

Sesuai permohonan dari mahasiswa/i :

Nama	: **NATALINO HERCIO SILVANIA D \ COSTA AMARAL**
Nim	: **0712535**
Semester	: **V (Lima)**
Jurusan	: **Teknik Elektro S-1**
Konsentrasi	: **Teknik Komputer & Informatika**

Dengan ini menyatakan bersedia/tidak bersedia*) Membimbing skripsi dari mahasiswa tersebut, dengan judul :

" APLIKASI SISTEM PENGENALAN WAJAH MENGGUNAKAN METODE EIGEN FACE BERBASIS J2SE "

Demikian surat pernyataan ini kami buat agar dapat dipergunakan seperlunya.

Hormat Kami

Sotyobadi, ST
NIP.Y. 1039700309

Catatan :

Setelah disetujui agar formulir ini Diserahkan mahasiswa/I yang bersangkutan kepada jurusan untuk diproses lebih lanjut

*) Coret yang tidak perlu

INSTITUT TEKNOLOGI NASIONAL
Jl. Raya Karanglo, KM 2
MALANG

FORMULIR BIMBINGAN SKRIPSI

Nama : Natalino Hercio S dc Amaral
Nim : 07.12.535
Masa Bimbingan : 21 mei 2012 s/d 20 july 2012
Judul Skripsi : Aplikasi Sistem Pengenalan Wajah Menggunakan Metode Eigenface Berbasis J2SE

No	Tanggal	Uraian	Paraf Pembimbing
1	14-07-12	Bab 3, data tulis eigen face proses	
2	Tgl 18/07/12		
3	25-07-12	Revisi BAB III	
4	Tgl 25/07/12	BAB IV	
5	28/07/12	Revisi BAB V	
6	29/07/12	Acc Simpan hasil	
7		Acc BAB V	
8		Acc ujian Skripsi	
9			
10			

Malang,
Dosen Pembimbing I

Irmalia S. Faradisa, ST, MT

AMBIL GAMBAR

```
private void bdeteksiwajahActionPerformed(java.awt.event.ActionEvent evt) {
      // TODO add your handling code here:try{
      try{
            //  Thread.sleep(1000);//wait 10 seconds before capturing photo

        FrameGrabbingControl fgc = (FrameGrabbingControl)
player.getControl("javax.media.control.FrameGrabbingControl");

        Buffer buf = fgc.grabFrame();//grab the current frame on video screen

        BufferToImage btoi = new BufferToImage((VideoFormat) buf.getFormat());

       img = btoi.createImage(buf);
     saveImagetoFile(img, "src/jjilexample/MyPhoto.jpg");
     }
    catch (Exception e) {
     }
   // Main mnc=new Main();
    // mnc.detekwajah();


    BufferedImage test=null;
  try{
     test = ImageIO.read(new File("src/jjilexample/result.jpg"));
   // xx(getGrayScale(test)); setKoordinatCropFace(test);


  }catch (IOException r){

  }


    ImageCropper ic=new ImageCropper();
    try{
       ic.ck(60, 260, 20, 220);
       showwajah();
    }catch (Exception e){

    }

  }
```

CROP

```
private void setKoordinatCropFace(BufferedImage img){
```

```
    //BufferedImage                                                    imgHasil=new
BufferedImage(img.getWidth(),img.getHeight(),BufferedImage.TYPE_INT_RGB);
    BufferedImage imgHasil=img;

    int getG[][]=new int[imgHasil.getHeight()][imgHasil.getWidth()];
    int xx=0;
    int xx1;
    int yy=0;
    int yy1;

    int gray=0;
    for(int y=0;y<imgHasil.getHeight();y++){
      for(int x=0;x<imgHasil.getWidth();x++){
        //imgHasil.setRGB(x, y, 0);
         gray=convertToGrayscale(img.getRGB(x, y));
        if(gray==255){
          gray=255;
        }else{
          gray=0;
        }
        getG[y][x]=gray;
      }
    }


          int a=10000000;
         int b=100000000;
         int c;
         int d;

  for(int y=0;y<imgHasil.getHeight();y++){
     for(int x=0;x<imgHasil.getWidth();x++){
       if(getG[y][x]==255){
         a=x;
         b=y;
         break;
       }
  }
     if(y==b){
       break;
     }
  }
  X1=a;
  Y1=b;
        System.out.println("x,y = "+a+","+b);
```

```
    for(int y=0;y<imgHasil.getHeight();y++){
       for(int x=0;x<imgHasil.getWidth();x++){
         if(getG[y][x]==255){


            xx=x;
            yy=y;

         }
    }

    }
    System.out.println("x2,y2 = "+xx+","+yy);
X2=xx;
Y2=yy;

  }



   BufferedImage buffer;
  int rowCount=0;
  Object[][] data1= null;
  private Object[][] getData_Training() throws IOException{

    try {


      Statement st = koneksi.getConnection().createStatement();
       ResultSet rs = st.executeQuery(
            "Select * from training");

       rs.last();
        rowCount= rs.getRow();
       rs.beforeFirst();

       data1= new Object[rowCount][3];
       int no=-1;
       while (rs.next()) {
          no=no+1;
          data1[no][0]=rs.getString("ID");
          data1[no][1]=rs.getString("nama");
          data1[no][2]=rs.getBlob("face1");
         // Blob blob = (Blob)rs.getBlob("face1");
```

```
            // ImageIcon icon = new ImageIcon(blob.getBytes(1,gbr));
        }

        st.close();


    }



    catch (SQLException e){
        System.out.println("koneksi gagal " + e.toString());

    }

    return data1;
  }


  private int convertToGrayscale(int rgb){
    int gray;
    gray=(int)((((rgb>>16) & 0xFF)+((rgb>>8) & 0xFF)+((rgb>>0) & 0xFF))/3);
    return gray;
  }
```

SIMPAN

```
private void bSimpanActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:

    try{
      if(txtNama.getText().equals("")){
        JOptionPane.showMessageDialog(this,
            "Masukkan Nama Anda dong ?!", "Informasi",
            JOptionPane.INFORMATION_MESSAGE);

      }else{
        save();
        LabelFace.setIcon(null);
        tampilTabel();
      }

    }catch (FileNotFoundException r){

    }
  }
```

HAPUS

```
private void bHapus1ActionPerformed(java.awt.event.ActionEvent evt) {
// TODO add your handling code here:
   hapusAll=true;
   int confirm;
      confirm=JOptionPane.showConfirmDialog(this,
                  "Apakah Anda Yakin Ingin Menghapus Semua Record?", "Konfirmasi",
                  JOptionPane.YES_NO_OPTION);
     if(confirm==0){
      hapus();
     tampilTabel();}
}
```

DATA TRANING

```
private String getData_TrainingHasil(){

   String hasil="";
      try {
        Statement st = koneksi.getConnection().createStatement();
         ResultSet rs = st.executeQuery(
               "Select * from training order by jml desc limit 0,1");
         rs.last();
         rs.beforeFirst();

         while (rs.next()) {
            hasil=rs.getString("nama");
        }
        st.close();
     }
     catch (SQLException e){
        System.out.println("koneksi gagal " + e.toString());
     }
      return hasil;
   }


   private Object[][] getDat() {
   Object[][] data= null;


      try {
```

```
        Statement st = koneksi.getConnection().createStatement();
         ResultSet rs = st.executeQuery(
              "Select * from training");

         rs.last();
         int row= rs.getRow();
         rs.beforeFirst();

         data= new Object[row][2];
         int no=-1;
         while (rs.next()) {
            no=no+1;
            data[no][0]=rs.getString("ID");
            data[no][1]=rs.getString("nama");
           //  data1[no][2]=rs.getBlob("face1");
            // Blob blob = (Blob)rs.getBlob("face1");


            // ImageIcon icon = new ImageIcon(blob.getBytes(1,gbr));
        }

        st.close();


      }



      catch (SQLException e){
          System.out.println("koneksi gagal " + e.toString());

      }

       return data;
    }
    JTable table1;


private void tampilTabel() {

        String[] columnNames =
          {"kode", "Nama"};

     table1 = new JTable(
         getDat(), columnNames);
```

```
      jScrollPane1.setViewportView(table1);
      table1.setBackground(Color.BLUE);
      table1.setForeground(Color.WHITE);
    /*
        table.getSelectionModel().addListSelectionListener(new ListSelectionListener() {

   public void valueChanged(ListSelectionEvent e) {
      int selectedRow = table.getSelectedRow();
     System.out.println(String.valueOf(selectedRow));
   }
});*/

   }
boolean hapusAll=true;
void hapus(){

String sql="";
       try {



        PreparedStatement pStatement = null;
        if(hapusAll==true){
           sql ="delete from training ";
            pStatement = koneksi.getConnection().prepareStatement(sql);
        }else{
         sql ="delete from training " +
              " where id=? ";
          pStatement = koneksi.getConnection().prepareStatement(sql);

        pStatement.setString(1, usrId);
        }


        int intTambah= pStatement.executeUpdate();
        if (intTambah>0)
           JOptionPane.showMessageDialog(this,
                 "Hapus data sukses", "Informasi",
                 JOptionPane.INFORMATION_MESSAGE);
        else
           JOptionPane.showMessageDialog(this,
                 "Hapus data gagal", "Informasi",
                 JOptionPane.INFORMATION_MESSAGE);

        pStatement.close();
```

```
        }

        catch (SQLException e){
           System.out.println("koneksi gagal " + e.toString());
        }
}
}
```

DATA TRAINING HASIL

```
private String getData_TrainingHasil(){

   String hasil="";
      try {
        Statement st = koneksi.getConnection().createStatement();
         ResultSet rs = st.executeQuery(
              "Select * from training order by jml desc limit 0,1");
         rs.last();
         rs.beforeFirst();

         while (rs.next()) {
            hasil=rs.getString("nama");
        }
       st.close();
     }
     catch (SQLException e){
        System.out.println("koneksi gagal " + e.toString());
     }
      return hasil;
   }
```

TAMPILAN TABEL

```
private void tampilTabel() {

       String[] columnNames =
         {"kode", "Nama"};

    table1 = new JTable(
        getDat(), columnNames);

      jScrollPane1.setViewportView(table1);
      table1.setBackground(Color.BLUE);
      table1.setForeground(Color.WHITE);
```

```
    /*
        table.getSelectionModel().addListSelectionListener(new ListSelectionListener() {

   public void valueChanged(ListSelectionEvent e) {
      int selectedRow = table.getSelectedRow();
     System.out.println(String.valueOf(selectedRow));
   }
});*/

   }
boolean hapusAll=true;
void hapus(){

String sql="";
       try {



        PreparedStatement pStatement = null;
        if(hapusAll==true){
           sql ="delete from training ";
            pStatement = koneksi.getConnection().prepareStatement(sql);
        }else{
         sql ="delete from training " +
              " where id=? ";
          pStatement = koneksi.getConnection().prepareStatement(sql);

        pStatement.setString(1, usrId);
        }


        int intTambah= pStatement.executeUpdate();
        if (intTambah>0)
           JOptionPane.showMessageDialog(this,
                "Hapus data sukses", "Informasi",
                JOptionPane.INFORMATION_MESSAGE);
        else
           JOptionPane.showMessageDialog(this,
                "Hapus data gagal", "Informasi",
                JOptionPane.INFORMATION_MESSAGE);

        pStatement.close();


    }
```

```
        catch (SQLException e){
            System.out.println("koneksi gagal " + e.toString());
        }
}
}
```

MENU NILAI EIGEN

```
private void Menentukan_nilai_eigen(){

    for(int y=0;y<160;y++){
        for(int x=0;x<160;x++){
            if(Gambar1[y][x]==Gambar2[y][x]){
                jml=jml+1;
            }
        }
    }
}
```

KONVERT TO GRAYSCALE

```
private int convertToGrayscale(int rgb){
        int gray;
        gray=(int)((((rgb>>16) & 0xFF)+((rgb>>8) & 0xFF)+((rgb>>0) & 0xFF))/3);
        return gray;
    }
```

NILAI GRAYSCALE

```
private BufferedImage getGrayScale(BufferedImage img){
        //BufferedImage                                                    imgHasil=new
BufferedImage(img.getWidth(),img.getHeight(),BufferedImage.TYPE_INT_RGB);
        BufferedImage imgHasil=img;
        System.out.println("y= "+imgHasil.getHeight());
        System.out.println("x= "+imgHasil.getWidth());
        for(int y=0;y<imgHasil.getHeight();y++){
            for(int x=0;x<imgHasil.getWidth();x++){
                //imgHasil.setRGB(x, y, 0);
                int gray=convertToGrayscale(img.getRGB(x, y));
                Gambar1[y][x]=gray;

                Color color=new Color(gray,gray,gray);
                imgHasil.setRGB(x, y, color.getRGB());
                //System.out.println(((color.getRGB()>>16)&0xFF)+"
"+((color.getRGB()>>8)&0xFF)+" "+((color.getRGB()>>0)& 0xFF));
```

```
        }
      }
      return imgHasil;
   }
private BufferedImage getGrayScale1(BufferedImage img){
      //BufferedImage                                                    imgHasil=new
BufferedImage(img.getWidth(),img.getHeight(),BufferedImage.TYPE_INT_RGB);
      BufferedImage imgHasil=img;
      System.out.println("y= "+imgHasil.getHeight());
      System.out.println("x= "+imgHasil.getWidth());
      for(int y=0;y<imgHasil.getHeight();y++){
         for(int x=0;x<imgHasil.getWidth();x++){
            //imgHasil.setRGB(x, y, 0);
            int gray=convertToGrayscale(img.getRGB(x, y));
            Gambar2[y][x]=gray;

            Color color=new Color(gray,gray,gray);
            imgHasil.setRGB(x, y, color.getRGB());
            //System.out.println(((color.getRGB()>>16)&0xFF)+"
"+((color.getRGB()>>8)&0xFF)+" "+((color.getRGB()>>0)& 0xFF));

         }
      }
      return imgHasil;
   }
```

KONEKSI WEBCAM

```
public class Formcam extends javax.swing.JFrame {
int X1=0;
int X2=0;
int Y1=0;
int Y2=0;
Koneksi koneksi=new Koneksi();
   CaptureDeviceInfo device;
   MediaLocator ml;
   Player player;
   Component videoScreen;

Image img;
private void center(){
      Dimension dimension = Toolkit.getDefaultToolkit().getScreenSize();
      Dimension d = this.getSize();
      if(d.height>dimension.height){
         d.height=dimension.height;
```

```
        }
        if(d.width>dimension.width){
           d.width=dimension.width;
        }
        this.setLocation((dimension.width-d.width)/2,(dimension.height-d.height)/2);
       // this.setBounds((dimension.width-d.width)/2,(dimension.height-d.height)/2, 600, 300);

    }
    /** Creates new form Formcam */
    public Formcam() {

        initComponents();
       ShowWebCam();
      koneksi.konn();
        center();
       tampilTabel();
    }
 private void ShowWebCam() {
        try {
//gets a list of devices how support the given videoformat
            Vector deviceList = CaptureDeviceManager.getDeviceList(new RGBFormat());
            System.out.println(deviceList.toString());

//gets the first device in deviceList
            device = (CaptureDeviceInfo) deviceList.firstElement();

            ml = device.getLocator();

            player = Manager.createRealizedPlayer(ml);

            player.start();

            videoScreen = player.getVisualComponent();
          //  Frame frm = new Frame();
           // frm.setBounds(10, 10, 900, 700);//sets the size of the screen

// setting close operation to the frame
          //  frm.addWindowListener(new WindowAdapter() {

           //     public void windowClosing(WindowEvent we) {
           //        System.exit(0);
           //     }
          //  });

//place player and video screen on the frame
            panelCam.add(videoScreen, BorderLayout.CENTER);
```

```
        panelCam.add(player.getControlPanelComponent(), BorderLayout.SOUTH);
        panelCam.setVisible(true);

//capture image


      //  saveImagetoFile(img, "MyPhoto.jpg");//save the captured image as MyPhoto.jpg

      } catch (Exception e) {
         System.out.println(e);
      }
   }
```

SAVE IMAGE TO FILE

```
private void saveImagetoFile(Image img, String string) {
      try {
         int w = img.getWidth(null);
         int h = img.getHeight(null);
         BufferedImage bi = new BufferedImage(w, h, BufferedImage.TYPE_INT_RGB);
         Graphics2D g2 = bi.createGraphics();

         g2.drawImage(img, 0, 0, null);

         g2.dispose();

         String fileType = string.substring(string.indexOf('.') + 1);

         ImageIO.write(bi, fileType, new File(string));

      } catch (Exception e) {
      }
   }
```

DETEKSI WAJAH ACTION PERFOMED

```
   private void bdeteksiwajahActionPerformed(java.awt.event.ActionEvent evt) {
      // TODO add your handling code here:try{
      try{
            //  Thread.sleep(1000);//wait 10 seconds before capturing photo

         FrameGrabbingControl fgc = (FrameGrabbingControl)
player.getControl("javax.media.control.FrameGrabbingControl");

         Buffer buf = fgc.grabFrame();//grab the current frame on video screen
```

```
        BufferToImage btoi = new BufferToImage((VideoFormat) buf.getFormat());

        img = btoi.createImage(buf);
      saveImagetoFile(img, "src/jjilexample/MyPhoto.jpg");
      }
     catch (Exception e) {
      }
   //  Main mnc=new Main();
    // mnc.detekwajah();


     BufferedImage test=null;
    try{
      test = ImageIO.read(new File("src/jjilexample/result.jpg"));
    //  xx(getGrayScale(test));  setKoordinatCropFace(test);


    }catch (IOException r){

    }


     ImageCropper ic=new ImageCropper();
     try{
        ic.ck(60, 260, 20, 220);
        showwajah();
     }catch (Exception e){

     }

    }
```

HAPUS ACTION PERFORMED

```
private void bHapus1ActionPerformed(java.awt.event.ActionEvent evt) {
// TODO add your handling code here:
   hapusAll=true;
   int confirm;
      confirm=JOptionPane.showConfirmDialog(this,
                "Apakah Anda Yakin Ingin Menghapus Semua Record?", "Konfirmasi",
                JOptionPane.YES_NO_OPTION);
     if(confirm==0){
      hapus();
     tampilTabel();}
}

   /**
```

```
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        java.awt.EventQueue.invokeLater(new Runnable() {

            public void run() {
                new Formcam().setVisible(true);
            }
        });
    }

    // Variables declaration - do not modify
    private javax.swing.JLabel LabelFace;
    private javax.swing.JLabel LabelFace1;
    private javax.swing.JButton bCek;
    private javax.swing.JButton bHapus;
    private javax.swing.JButton bHapus1;
    private javax.swing.JButton bSimpan;
    private javax.swing.JButton bdeteksiwajah;
    private javax.swing.JPanel hhh;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JPanel jPanel10;
    private javax.swing.JPanel jPanel11;
    private javax.swing.JPanel jPanel12;
    private javax.swing.JPanel jPanel13;
    private javax.swing.JPanel jPanel2;
    private javax.swing.JPanel jPanel3;
    private javax.swing.JPanel jPanel4;
    private javax.swing.JPanel jPanel5;
    private javax.swing.JPanel jPanel6;
    private javax.swing.JPanel jPanel7;
    private eigenface.jPanel jPanel8;
    private javax.swing.JPanel jPanel9;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JTabbedPane jTabbedPane1;
    private javax.swing.JTabbedPane jTabbedPane2;
    private javax.swing.JLabel lHasil;
    private javax.swing.JPanel panelCam;
    private javax.swing.JTextField txtID;
    private javax.swing.JTextField txtNama;
    // End of variables declaration

    private BufferedImage icon(BufferedImage img){
```

PROSES EIGENFACE

```
/*
 * Copyright (c) 2011. Philipp Wagner <bytefish[at]gmx[dot]de>.
 * Released to public domain under terms of the BSD Simplified license.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions are met:
 *   * Redistributions of source code must retain the above copyright
 *     notice, this list of conditions and the following disclaimer.
 *   * Redistributions in binary form must reproduce the above copyright
 *     notice, this list of conditions and the following disclaimer in the
 *     documentation and/or other materials provided with the distribution.
 *   * Neither the name of the organization nor the names of its contributors
 *     may be used to endorse or promote products derived from this software
 *     without specific prior written permission.
 *
 *   See <http://www.opensource.org/licenses/bsd-license>
 */

#include "capture/core/core.hpp"
#include "capture/contrib/contrib.hpp"
#include "capture/highgui/highgui.hpp"

#include <iostream>
#include <fstream>
#include <sstream>

using namespace cv;
using namespace std;

static Mat norm_0_255(InputArray _src) {
    Mat src = _src.getMat();
    // Create and return normalized image:
    Mat dst;
    switch(src.channels()) {
    case 1:
        cv::normalize(_src, dst, 0, 255, NORM_MINMAX, CV_8UC1);
        break;
    case 3:
        cv::normalize(_src, dst, 0, 255, NORM_MINMAX, CV_8UC3);
        break;
    default:
        src.copyTo(dst);
        break;
    }
    return dst;
}

static void read_csv(const string& filename, vector<Mat>& images, vector<int>&
labels, char separator = ';') {
    std::ifstream file(filename.c_str(), ifstream::in);
    if (!file) {
        string error_message = "No valid input file was given, please check
the given filename.";
```

```
        CV_Error(CV_StsBadArg, error_message);
    }
    string line, path, classlabel;
    while (getline(file, line)) {
        stringstream liness(line);
        getline(liness, path, separator);
        getline(liness, classlabel);
        if(!path.empty() && !classlabel.empty()) {
            images.push_back(imread(path, 0));
            labels.push_back(atoi(classlabel.c_str()));
        }
    }
}

int main(int argc, const char *argv[]) {
    // Check for valid command line arguments, print usage
    // if no arguments were given.
    if (argc < 2) {
        cout << "usage: " << argv[0] << " <csv.ext> <output_folder> " << endl;
        exit(1);
    }
    string output_folder;
    if (argc == 3) {
        output_folder = string(argv[2]);
    }
    // Get the path to your CSV.
    string fn_csv = string(argv[1]);
    // These vectors hold the images and corresponding labels.
    vector<Mat> images;
    vector<int> labels;
    // Read in the data. This can fail if no valid
    // input filename is given.
    try {
        read_csv(fn_csv, images, labels);
    } catch (cv::Exception& e) {
        cerr << "Error opening file \"" << fn_csv << "\". Reason: " << e.msg
<< endl;
        // nothing more we can do
        exit(1);
    }
    // Quit if there are not enough images for this demo.
    if(images.size() <= 1) {
        string error_message = "This demo needs at least 2 images to work.
Please add more images to your data set!";
        CV_Error(CV_StsError, error_message);
    }
    // Get the height from the first image. We'll need this
    // later in code to reshape the images to their original
    // size:
    int height = images[0].rows;
    // The following lines simply get the last images from
    // your dataset and remove it from the vector. This is
    // done, so that the training data (which we learn the
    // cv::FaceRecognizer on) and the test data we test
    // the model with, do not overlap.
    Mat testSample = images[images.size() - 1];
    int testLabel = labels[labels.size() - 1];
    images.pop_back();
    labels.pop_back();
```

```
    // The following lines create an Eigenfaces model for
    // face recognition and train it with the images and
    // labels read from the given CSV file.
    // This here is a full PCA, if you just want to keep
    // 10 principal components (read Eigenfaces), then call
    // the factory method like this:
    //
    //      cv::createEigenFaceRecognizer(10);
    //
    // If you want to create a FaceRecognizer with a
    // confidence threshold (e.g. 123.0), call it with:
    //
    //      cv::createEigenFaceRecognizer(10, 123.0);
    //
    // If you want to use _all_ Eigenfaces and have a threshold,
    // then call the method like this:
    //
    //      cv::createEigenFaceRecognizer(0, 123.0);
    //
    Ptr<FaceRecognizer> model = createEigenFaceRecognizer();
    model->train(images, labels);
    // The following line predicts the label of a given
    // test image:
    int predictedLabel = model->predict(testSample);
    //
    // To get the confidence of a prediction call the model with:
    //
    //      int predictedLabel = -1;
    //      double confidence = 0.0;
    //      model->predict(testSample, predictedLabel, confidence);
    //
    string result_message = format("Predicted class = %d / Actual class =
%d.", predictedLabel, testLabel);
    cout << result_message << endl;
    // Here is how to get the eigenvalues of this Eigenfaces model:
    Mat eigenvalues = model->getMat("eigenvalues");
    // And we can do the same to display the Eigenvectors (read Eigenfaces):
    Mat W = model->getMat("eigenvectors");
    // Get the sample mean from the training data
    Mat mean = model->getMat("mean");
    // Display or save:
    if(argc == 2) {
        imshow("mean", norm_0_255(mean.reshape(1, images[0].rows)));
    } else {
        imwrite(format("%s/mean.png", output_folder.c_str()),
norm_0_255(mean.reshape(1, images[0].rows)));
    }
    // Display or save the Eigenfaces:
    for (int i = 0; i < min(10, W.cols); i++) {
        string msg = format("Eigenvalue #%d = %.5f", i,
eigenvalues.at<double>(i));
        cout << msg << endl;
        // get eigenvector #i
        Mat ev = W.col(i).clone();
        // Reshape to original size & normalize to [0...255] for imshow.
        Mat grayscale = norm_0_255(ev.reshape(1, height));
        // Show the image & apply a Jet colormap for better sensing.
        Mat cgrayscale;
        applyColorMap(grayscale, cgrayscale, COLORMAP_JET);
```

```
        // Display or save:
        if(argc == 2) {
            imshow(format("eigenface_%d", i), cgrayscale);
        } else {
            imwrite(format("%s/eigenface_%d.png", output_folder.c_str(), i),
norm_0_255(cgrayscale));
        }
    }

    // Display or save the image reconstruction at some predefined steps:
    for(int num_components = min(W.cols, 10); num_components < min(W.cols,
300); num_components+=15) {
        // slice the eigenvectors from the model
        Mat evs = Mat(W, Range::all(), Range(0, num_components));
        Mat projection = subspaceProject(evs, mean, images[0].reshape(1,1));
        Mat reconstruction = subspaceReconstruct(evs, mean, projection);
        // Normalize the result:
        reconstruction = norm_0_255(reconstruction.reshape(1,
images[0].rows));
        // Display or save:
        if(argc == 2) {
            imshow(format("eigenface_reconstruction_%d", num_components),
reconstruction);
        } else {
            imwrite(format("%s/eigenface_reconstruction_%d.png",
output_folder.c_str(), num_components), reconstruction);
        }
    }
    // Display if we are not writing to an output folder:
    if(argc == 2) {
        waitKey(0);
    }
    return 0;
}
```

```
/*
Copyright (c) 2011. Philipp Wagner <bytefish[at]gmx[dot]de>.
 * Released to public domain under terms of the BSD Simplified license.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions are met:
 *   * Redistributions of source code must retain the above copyright
 *     notice, this list of conditions and the following disclaimer.
 *   * Redistributions in binary form must reproduce the above copyright
 *     notice, this list of conditions and the following disclaimer in the
 *     documentation and/or other materials provided with the distribution.
 *   * Neither the name of the organization nor the names of its contributors
 *     may be used to endorse or promote products derived from this software
 *     without specific prior written permission.
 *
 *   See <http://www.opensource.org/licenses/bsd-license>
 */

#include "capture/core/core.hpp"
#include "capture/contrib/contrib.hpp"
#include "capture/highgui/highgui.hpp"

#include <iostream>
#include <fstream>
#include <sstream>

using namespace cv;
using namespace std;

static Mat norm_0_255(InputArray _src) {
    Mat src = _src.getMat();
    // Create and return normalized image:
    Mat dst;
    switch(src.channels()) {
    case 1:
        cv::normalize(_src, dst, 0, 255, NORM_MINMAX, CV_8UC1);
        break;
    case 3:
        cv::normalize(_src, dst, 0, 255, NORM_MINMAX, CV_8UC3);
        break;
    default:
        src.copyTo(dst);
        break;
    }
    return dst;
}

static void read_csv(const string& filename, vector<Mat>& images, vector<int>&
labels, char separator = ';') {
    std::ifstream file(filename.c_str(), ifstream::in);
    if (!file) {
        string error_message = "No valid input file was given, please check
the given filename.";
        CV_Error(CV_StsBadArg, error_message);
    }
    string line, path, classlabel;
    while (getline(file, line)) {
        stringstream liness(line);
```

```
        getline(liness, path, separator);
        getline(liness, classlabel);
        if(!path.empty() && !classlabel.empty()) {
            images.push_back(imread(path, 0));
            labels.push_back(atoi(classlabel.c_str()));
        }
    }
}

int main(int argc, const char *argv[]) {
    // Check for valid command line arguments, print usage
    // if no arguments were given.
    if (argc < 2) {
        cout << "usage: " << argv[0] << " <csv.ext> <output_folder> " << endl;
        exit(1);
    }
    string output_folder;
    if (argc == 3) {
        output_folder = string(argv[2]);
    }
    // Get the path to your CSV.
    string fn_csv = string(argv[1]);
    // These vectors hold the images and corresponding labels.
    vector<Mat> images;
    vector<int> labels;
    // Read in the data. This can fail if no valid
    // input filename is given.
    try {
        read_csv(fn_csv, images, labels);
    } catch (cv::Exception& e) {
        cerr << "Error opening file \"" << fn_csv << "\". Reason: " << e.msg
<< endl;
        // nothing more we can do
        exit(1);
    }
    // Quit if there are not enough images for this demo.
    if(images.size() <= 1) {
        string error_message = "This demo needs at least 2 images to work.
Please add more images to your data set!";
        CV_Error(CV_StsError, error_message);
    }
    // Get the height from the first image. We'll need this
    // later in code to reshape the images to their original
    // size:
    int height = images[0].rows;
    // The following lines simply get the last images from
    // your dataset and remove it from the vector. This is
    // done, so that the training data (which we learn the
    // cv::FaceRecognizer on) and the test data we test
    // the model with, do not overlap.
    Mat testSample = images[images.size() - 1];
    int testLabel = labels[labels.size() - 1];
    images.pop_back();
    labels.pop_back();
    // The following lines create an Fisherfaces model for
    // face recognition and train it with the images and
    // labels read from the given CSV file.
    // If you just want to keep 10 Fisherfaces, then call
    // the factory method like this:
```

```
    //
    //      cv::createFisherFaceRecognizer(10);
    //
    // However it is not useful to discard Fisherfaces! Please
    // always try to use _all_ available Fisherfaces for
    // classification.
    //
    // If you want to create a FaceRecognizer with a
    // confidence threshold (e.g. 123.0) and use _all_
    // Fisherfaces, then call it with:
    //
    //      cv::createFisherFaceRecognizer(0, 123.0);
    //
    Ptr<FaceRecognizer> model = createFisherFaceRecognizer();
    model->train(images, labels);
    // The following line predicts the label of a given
    // test image:
    int predictedLabel = model->predict(testSample);
    //
    // To get the confidence of a prediction call the model with:
    //
    //      int predictedLabel = -1;
    //      double confidence = 0.0;
    //      model->predict(testSample, predictedLabel, confidence);
    //
    string result_message = format("Predicted class = %d / Actual class =
%d.", predictedLabel, testLabel);
    cout << result_message << endl;
    // Here is how to get the eigenvalues of this Eigenfaces model:
    Mat eigenvalues = model->getMat("eigenvalues");
    // And we can do the same to display the Eigenvectors (read Eigenfaces):
    Mat W = model->getMat("eigenvectors");
    // Get the sample mean from the training data
    Mat mean = model->getMat("mean");
    // Display or save:
    if(argc == 2) {
        imshow("mean", norm_0_255(mean.reshape(1, images[0].rows)));
    } else {
        imwrite(format("%s/mean.png", output_folder.c_str()),
norm_0_255(mean.reshape(1, images[0].rows)));
    }
    // Display or save the first, at most 16 Fisherfaces:
    for (int i = 0; i < min(16, W.cols); i++) {
        string msg = format("Eigenvalue #%d = %.5f", i,
eigenvalues.at<double>(i));
        cout << msg << endl;
        // get eigenvector #i
        Mat ev = W.col(i).clone();
        // Reshape to original size & normalize to [0...255] for imshow.
        Mat grayscale = norm_0_255(ev.reshape(1, height));
        // Show the image & apply a Bone colormap for better sensing.
        Mat cgrayscale;
        applyColorMap(grayscale, cgrayscale, COLORMAP_BONE);
        // Display or save:
        if(argc == 2) {
            imshow(format("fisherface_%d", i), cgrayscale);
        } else {
            imwrite(format("%s/fisherface_%d.png", output_folder.c_str(), i),
norm_0_255(cgrayscale));
```

```
        }
    }
    // Display or save the image reconstruction at some predefined steps:
    for(int num_component = 0; num_component < min(16, W.cols);
num_component++) {
        // Slice the Fisherface from the model:
        Mat ev = W.col(num_component);
        Mat projection = subspaceProject(ev, mean, images[0].reshape(1,1));
        Mat reconstruction = subspaceReconstruct(ev, mean, projection);
        // Normalize the result:
        reconstruction = norm_0_255(reconstruction.reshape(1,
images[0].rows));
        // Display or save:
        if(argc == 2) {
            imshow(format("fisherface_reconstruction_%d", num_component),
reconstruction);
        } else {
            imwrite(format("%s/fisherface_reconstruction_%d.png",
output_folder.c_str(), num_component), reconstruction);
        }
    }
    // Display if we are not writing to an output folder:
    if(argc == 2) {
        waitKey(0);
    }
    return 0;
}
```

```
Display or save the image reconstruction at some predefined steps:
for(int num_component = 0; num_component < min(16, W.cols); num_component++) {
    // Slice the Fisherface from the model:
    Mat ev = W.col(num_component);
    Mat projection = subspaceProject(ev, mean, images[0].reshape(1,1));
    Mat reconstruction = subspaceReconstruct(ev, mean, projection);
    // Normalize the result:
    reconstruction = norm_0_255(reconstruction.reshape(1, images[0].rows));
    // Display or save:
    if(argc == 2) {
        imshow(format("fisherface_reconstruction_%d", num_component),
reconstruction);
    } else {
        imwrite(format("%s/fisherface_reconstruction_%d.png",
output_folder.c_str(), num_component), reconstruction);
    }
}
```

```
/*
 * Copyright (c) 2011. Philipp Wagner <bytefish[at]gmx[dot]de>.
 * Released to public domain under terms of the BSD Simplified license.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions are met:
 *   * Redistributions of source code must retain the above copyright
 *     notice, this list of conditions and the following disclaimer.
 *   * Redistributions in binary form must reproduce the above copyright
 *     notice, this list of conditions and the following disclaimer in the
 *     documentation and/or other materials provided with the distribution.
 *   * Neither the name of the organization nor the names of its contributors
 *     may be used to endorse or promote products derived from this software
```

```
 *     without specific prior written permission.
 *
 *   See <http://www.opensource.org/licenses/bsd-license>
 */

#include "opencv2/core/core.hpp"
#include "opencv2/contrib/contrib.hpp"
#include "opencv2/highgui/highgui.hpp"

#include <iostream>
#include <fstream>
#include <sstream>

using namespace cv;
using namespace std;

static void read_csv(const string& filename, vector<Mat>& images, vector<int>&
labels, char separator = ';') {
    std::ifstream file(filename.c_str(), ifstream::in);
    if (!file) {
        string error_message = "No valid input file was given, please check
the given filename.";
        CV_Error(CV_StsBadArg, error_message);
    }
    string line, path, classlabel;
    while (getline(file, line)) {
        stringstream liness(line);
        getline(liness, path, separator);
        getline(liness, classlabel);
        if(!path.empty() && !classlabel.empty()) {
            images.push_back(imread(path, 0));
            labels.push_back(atoi(classlabel.c_str()));
        }
    }
}

int main(int argc, const char *argv[]) {
    // Check for valid command line arguments, print usage
    // if no arguments were given.
    if (argc != 2) {
        cout << "usage: " << argv[0] << " <csv.ext>" << endl;
        exit(1);
    }
    // Get the path to your CSV.
    string fn_csv = string(argv[1]);
    // These vectors hold the images and corresponding labels.
    vector<Mat> images;
    vector<int> labels;
    // Read in the data. This can fail if no valid
    // input filename is given.
    try {
        read_csv(fn_csv, images, labels);
    } catch (cv::Exception& e) {
        cerr << "Error opening file \"" << fn_csv << "\". Reason: " << e.msg
<< endl;
        // nothing more we can do
        exit(1);
    }
    // Quit if there are not enough images for this demo.
```

```
    if(images.size() <= 1) {
        string error_message = "This demo needs at least 2 images to work.
Please add more images to your data set!";
        CV_Error(CV_StsError, error_message);
    }
    // Get the height from the first image. We'll need this
    // later in code to reshape the images to their original
    // size:
    int height = images[0].rows;
    // The following lines simply get the last images from
    // your dataset and remove it from the vector. This is
    // done, so that the training data (which we learn the
    // cv::FaceRecognizer on) and the test data we test
    // the model with, do not overlap.
    Mat testSample = images[images.size() - 1];
    int testLabel = labels[labels.size() - 1];
    images.pop_back();
    labels.pop_back();
    // The following lines create an LBPH model for
    // face recognition and train it with the images and
    // labels read from the given CSV file.
    //
    // The LBPHFaceRecognizer uses Extended Local Binary Patterns
    // (it's probably configurable with other operators at a later
    // point), and has the following default values
    //
    //      radius = 1
    //      neighbors = 8
    //      grid_x = 8
    //      grid_y = 8
    //
    // So if you want a LBPH FaceRecognizer using a radius of
    // 2 and 16 neighbors, call the factory method with:
    //
    //      cv::createLBPHFaceRecognizer(2, 16);
    //
    // And if you want a threshold (e.g. 123.0) call it with its default
values:
    //
    //      cv::createLBPHFaceRecognizer(1,8,8,8,123.0)
    //
    Ptr<FaceRecognizer> model = createLBPHFaceRecognizer();
    model->train(images, labels);
    // The following line predicts the label of a given
    // test image:
    int predictedLabel = model->predict(testSample);
    //
    // To get the confidence of a prediction call the model with:
    //
    //      int predictedLabel = -1;
    //      double confidence = 0.0;
    //      model->predict(testSample, predictedLabel, confidence);
    //
    string result_message = format("Predicted class = %d / Actual class =
%d.", predictedLabel, testLabel);
    cout << result_message << endl;
    // Sometimes you'll need to get/set internal model data,
    // which isn't exposed by the public cv::FaceRecognizer.
    // Since each cv::FaceRecognizer is derived from a
```

```
    // cv::Algorithm, you can query the data.
    //
    // First we'll use it to set the threshold of the FaceRecognizer
    // to 0.0 without retraining the model. This can be useful if
    // you are evaluating the model:
    //
    model->set("threshold", 0.0);
    // Now the threshold of this model is set to 0.0. A prediction
    // now returns -1, as it's impossible to have a distance below
    // it
    predictedLabel = model->predict(testSample);
    cout << "Predicted class = " << predictedLabel << endl;
    // Show some informations about the model, as there's no cool
    // Model data to display as in Eigenfaces/Fisherfaces.
    // Due to efficiency reasons the LBP images are not stored
    // within the model:
    cout << "Model Information:" << endl;
    string model_info = format("\tLBPH(radius=%i, neighbors=%i, grid_x=%i,
grid_y=%i, threshold=%.2f)",
            model->getInt("radius"),
            model->getInt("neighbors"),
            model->getInt("grid_x"),
            model->getInt("grid_y"),
            model->getDouble("threshold"));
    cout << model_info << endl;
    // We could get the histograms for example:
    vector<Mat> histograms = model->getMatVector("histograms");
    // But should I really visualize it? Probably the length is interesting:
    cout << "Size of the histograms: " << histograms[0].total() << endl;
    return 0;
}
```

PROSES EIGENFACE

```
/*
 * Copyright (c) 2011. Philipp Wagner <bytefish[at]gmx[dot]de>.
 * Released to public domain under terms of the BSD Simplified license.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions are met:
 *   * Redistributions of source code must retain the above copyright
 *     notice, this list of conditions and the following disclaimer.
 *   * Redistributions in binary form must reproduce the above copyright
 *     notice, this list of conditions and the following disclaimer in the
 *     documentation and/or other materials provided with the distribution.
 *   * Neither the name of the organization nor the names of its contributors
 *     may be used to endorse or promote products derived from this software
 *     without specific prior written permission.
 *
 *   See <http://www.opensource.org/licenses/bsd-license>
 */

#include "capture/core/core.hpp"
#include "capture/contrib/contrib.hpp"
#include "capture/highgui/highgui.hpp"

#include <iostream>
#include <fstream>
#include <sstream>

using namespace cv;
using namespace std;

static Mat norm_0_255(InputArray _src) {
    Mat src = _src.getMat();
    // Create and return normalized image:
    Mat dst;
    switch(src.channels()) {
    case 1:
        cv::normalize(_src, dst, 0, 255, NORM_MINMAX, CV_8UC1);
        break;
    case 3:
        cv::normalize(_src, dst, 0, 255, NORM_MINMAX, CV_8UC3);
        break;
    default:
        src.copyTo(dst);
        break;
    }
    return dst;
}

static void read_csv(const string& filename, vector<Mat>& images, vector<int>&
labels, char separator = ';') {
    std::ifstream file(filename.c_str(), ifstream::in);
    if (!file) {
        string error_message = "No valid input file was given, please check
the given filename.";
```

```
        CV_Error(CV_StsBadArg, error_message);
    }
    string line, path, classlabel;
    while (getline(file, line)) {
        stringstream liness(line);
        getline(liness, path, separator);
        getline(liness, classlabel);
        if(!path.empty() && !classlabel.empty()) {
            images.push_back(imread(path, 0));
            labels.push_back(atoi(classlabel.c_str()));
        }
    }
}

int main(int argc, const char *argv[]) {
    // Check for valid command line arguments, print usage
    // if no arguments were given.
    if (argc < 2) {
        cout << "usage: " << argv[0] << " <csv.ext> <output_folder> " << endl;
        exit(1);
    }
    string output_folder;
    if (argc == 3) {
        output_folder = string(argv[2]);
    }
    // Get the path to your CSV.
    string fn_csv = string(argv[1]);
    // These vectors hold the images and corresponding labels.
    vector<Mat> images;
    vector<int> labels;
    // Read in the data. This can fail if no valid
    // input filename is given.
    try {
        read_csv(fn_csv, images, labels);
    } catch (cv::Exception& e) {
        cerr << "Error opening file \"" << fn_csv << "\". Reason: " << e.msg
<< endl;
        // nothing more we can do
        exit(1);
    }
    // Quit if there are not enough images for this demo.
    if(images.size() <= 1) {
        string error_message = "This demo needs at least 2 images to work.
Please add more images to your data set!";
        CV_Error(CV_StsError, error_message);
    }
    // Get the height from the first image. We'll need this
    // later in code to reshape the images to their original
    // size:
    int height = images[0].rows;
    // The following lines simply get the last images from
    // your dataset and remove it from the vector. This is
    // done, so that the training data (which we learn the
    // cv::FaceRecognizer on) and the test data we test
    // the model with, do not overlap.
    Mat testSample = images[images.size() - 1];
    int testLabel = labels[labels.size() - 1];
    images.pop_back();
    labels.pop_back();
```

```
    // The following lines create an Eigenfaces model for
    // face recognition and train it with the images and
    // labels read from the given CSV file.
    // This here is a full PCA, if you just want to keep
    // 10 principal components (read Eigenfaces), then call
    // the factory method like this:
    //
    //      cv::createEigenFaceRecognizer(10);
    //
    // If you want to create a FaceRecognizer with a
    // confidence threshold (e.g. 123.0), call it with:
    //
    //      cv::createEigenFaceRecognizer(10, 123.0);
    //
    // If you want to use _all_ Eigenfaces and have a threshold,
    // then call the method like this:
    //
    //      cv::createEigenFaceRecognizer(0, 123.0);
    //
    Ptr<FaceRecognizer> model = createEigenFaceRecognizer();
    model->train(images, labels);
    // The following line predicts the label of a given
    // test image:
    int predictedLabel = model->predict(testSample);
    //
    // To get the confidence of a prediction call the model with:
    //
    //      int predictedLabel = -1;
    //      double confidence = 0.0;
    //      model->predict(testSample, predictedLabel, confidence);
    //
    string result_message = format("Predicted class = %d / Actual class =
%d.", predictedLabel, testLabel);
    cout << result_message << endl;
    // Here is how to get the eigenvalues of this Eigenfaces model:
    Mat eigenvalues = model->getMat("eigenvalues");
    // And we can do the same to display the Eigenvectors (read Eigenfaces):
    Mat W = model->getMat("eigenvectors");
    // Get the sample mean from the training data
    Mat mean = model->getMat("mean");
    // Display or save:
    if(argc == 2) {
        imshow("mean", norm_0_255(mean.reshape(1, images[0].rows)));
    } else {
        imwrite(format("%s/mean.png", output_folder.c_str()),
norm_0_255(mean.reshape(1, images[0].rows)));
    }
    // Display or save the Eigenfaces:
    for (int i = 0; i < min(10, W.cols); i++) {
        string msg = format("Eigenvalue #%d = %.5f", i,
eigenvalues.at<double>(i));
        cout << msg << endl;
        // get eigenvector #i
        Mat ev = W.col(i).clone();
        // Reshape to original size & normalize to [0...255] for imshow.
        Mat grayscale = norm_0_255(ev.reshape(1, height));
        // Show the image & apply a Jet colormap for better sensing.
        Mat cgrayscale;
        applyColorMap(grayscale, cgrayscale, COLORMAP_JET);
```

```
        // Display or save:
        if(argc == 2) {
            imshow(format("eigenface_%d", i), cgrayscale);
        } else {
            imwrite(format("%s/eigenface_%d.png", output_folder.c_str(), i),
norm_0_255(cgrayscale));
        }
    }

    // Display or save the image reconstruction at some predefined steps:
    for(int num_components = min(W.cols, 10); num_components < min(W.cols,
300); num_components+=15) {
        // slice the eigenvectors from the model
        Mat evs = Mat(W, Range::all(), Range(0, num_components));
        Mat projection = subspaceProject(evs, mean, images[0].reshape(1,1));
        Mat reconstruction = subspaceReconstruct(evs, mean, projection);
        // Normalize the result:
        reconstruction = norm_0_255(reconstruction.reshape(1,
images[0].rows));
        // Display or save:
        if(argc == 2) {
            imshow(format("eigenface_reconstruction_%d", num_components),
reconstruction);
        } else {
            imwrite(format("%s/eigenface_reconstruction_%d.png",
output_folder.c_str(), num_components), reconstruction);
        }
    }
    // Display if we are not writing to an output folder:
    if(argc == 2) {
        waitKey(0);
    }
    return 0;
}
```

```
/*
 * Copyright (c) 2011. Philipp Wagner <bytefish[at]gmx[dot]de>.
 * Released to public domain under terms of the BSD Simplified license.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions are met:
 *   * Redistributions of source code must retain the above copyright
 *     notice, this list of conditions and the following disclaimer.
 *   * Redistributions in binary form must reproduce the above copyright
 *     notice, this list of conditions and the following disclaimer in the
 *     documentation and/or other materials provided with the distribution.
 *   * Neither the name of the organization nor the names of its contributors
 *     may be used to endorse or promote products derived from this software
 *     without specific prior written permission.
 *
 *   See <http://www.opensource.org/licenses/bsd-license>
 */

#include "capture/core/core.hpp"
#include "capture/contrib/contrib.hpp"
#include "capture/highgui/highgui.hpp"

#include <iostream>
#include <fstream>
#include <sstream>

using namespace cv;
using namespace std;

static Mat norm_0_255(InputArray _src) {
    Mat src = _src.getMat();
    // Create and return normalized image:
    Mat dst;
    switch(src.channels()) {
    case 1:
        cv::normalize(_src, dst, 0, 255, NORM_MINMAX, CV_8UC1);
        break;
    case 3:
        cv::normalize(_src, dst, 0, 255, NORM_MINMAX, CV_8UC3);
        break;
    default:
        src.copyTo(dst);
        break;
    }
    return dst;
}

static void read_csv(const string& filename, vector<Mat>& images, vector<int>&
labels, char separator = ';') {
    std::ifstream file(filename.c_str(), ifstream::in);
    if (!file) {
        string error_message = "No valid input file was given, please check
the given filename.";
        CV_Error(CV_StsBadArg, error_message);
    }
    string line, path, classlabel;
    while (getline(file, line)) {
        stringstream liness(line);
```

```
        getline(liness, path, separator);
        getline(liness, classlabel);
        if(!path.empty() && !classlabel.empty()) {
            images.push_back(imread(path, 0));
            labels.push_back(atoi(classlabel.c_str()));
        }
    }
}

int main(int argc, const char *argv[]) {
    // Check for valid command line arguments, print usage
    // if no arguments were given.
    if (argc < 2) {
        cout << "usage: " << argv[0] << " <csv.ext> <output_folder> " << endl;
        exit(1);
    }
    string output_folder;
    if (argc == 3) {
        output_folder = string(argv[2]);
    }
    // Get the path to your CSV.
    string fn_csv = string(argv[1]);
    // These vectors hold the images and corresponding labels.
    vector<Mat> images;
    vector<int> labels;
    // Read in the data. This can fail if no valid
    // input filename is given.
    try {
        read_csv(fn_csv, images, labels);
    } catch (cv::Exception& e) {
        cerr << "Error opening file \"" << fn_csv << "\". Reason: " << e.msg
<< endl;
        // nothing more we can do
        exit(1);
    }
    // Quit if there are not enough images for this demo.
    if(images.size() <= 1) {
        string error_message = "This demo needs at least 2 images to work.
Please add more images to your data set!";
        CV_Error(CV_StsError, error_message);
    }
    // Get the height from the first image. We'll need this
    // later in code to reshape the images to their original
    // size:
    int height = images[0].rows;
    // The following lines simply get the last images from
    // your dataset and remove it from the vector. This is
    // done, so that the training data (which we learn the
    // cv::FaceRecognizer on) and the test data we test
    // the model with, do not overlap.
    Mat testSample = images[images.size() - 1];
    int testLabel = labels[labels.size() - 1];
    images.pop_back();
    labels.pop_back();
    // The following lines create an Fisherfaces model for
    // face recognition and train it with the images and
    // labels read from the given CSV file.
    // If you just want to keep 10 Fisherfaces, then call
    // the factory method like this:
```

```
    //
    //      cv::createFisherFaceRecognizer(10);
    //
    // However it is not useful to discard Fisherfaces! Please
    // always try to use _all_ available Fisherfaces for
    // classification.
    //
    // If you want to create a FaceRecognizer with a
    // confidence threshold (e.g. 123.0) and use _all_
    // Fisherfaces, then call it with:
    //
    //      cv::createFisherFaceRecognizer(0, 123.0);
    //
    Ptr<FaceRecognizer> model = createFisherFaceRecognizer();
    model->train(images, labels);
    // The following line predicts the label of a given
    // test image:
    int predictedLabel = model->predict(testSample);
    //
    // To get the confidence of a prediction call the model with:
    //
    //      int predictedLabel = -1;
    //      double confidence = 0.0;
    //      model->predict(testSample, predictedLabel, confidence);
    //
    string result_message = format("Predicted class = %d / Actual class =
%d.", predictedLabel, testLabel);
    cout << result_message << endl;
    // Here is how to get the eigenvalues of this Eigenfaces model:
    Mat eigenvalues = model->getMat("eigenvalues");
    // And we can do the same to display the Eigenvectors (read Eigenfaces):
    Mat W = model->getMat("eigenvectors");
    // Get the sample mean from the training data
    Mat mean = model->getMat("mean");
    // Display or save:
    if(argc == 2) {
        imshow("mean", norm_0_255(mean.reshape(1, images[0].rows)));
    } else {
        imwrite(format("%s/mean.png", output_folder.c_str()),
norm_0_255(mean.reshape(1, images[0].rows)));
    }
    // Display or save the first, at most 16 Fisherfaces:
    for (int i = 0; i < min(16, W.cols); i++) {
        string msg = format("Eigenvalue #%d = %.5f", i,
eigenvalues.at<double>(i));
        cout << msg << endl;
        // get eigenvector #i
        Mat ev = W.col(i).clone();
        // Reshape to original size & normalize to [0...255] for imshow.
        Mat grayscale = norm_0_255(ev.reshape(1, height));
        // Show the image & apply a Bone colormap for better sensing.
        Mat cgrayscale;
        applyColorMap(grayscale, cgrayscale, COLORMAP_BONE);
        // Display or save:
        if(argc == 2) {
            imshow(format("fisherface_%d", i), cgrayscale);
        } else {
            imwrite(format("%s/fisherface_%d.png", output_folder.c_str(), i),
norm_0_255(cgrayscale));
```

```
        }
    }
    // Display or save the image reconstruction at some predefined steps:
    for(int num_component = 0; num_component < min(16, W.cols);
num_component++) {
        // Slice the Fisherface from the model:
        Mat ev = W.col(num_component);
        Mat projection = subspaceProject(ev, mean, images[0].reshape(1,1));
        Mat reconstruction = subspaceReconstruct(ev, mean, projection);
        // Normalize the result:
        reconstruction = norm_0_255(reconstruction.reshape(1,
images[0].rows));
        // Display or save:
        if(argc == 2) {
            imshow(format("fisherface_reconstruction_%d", num_component),
reconstruction);
        } else {
            imwrite(format("%s/fisherface_reconstruction_%d.png",
output_folder.c_str(), num_component), reconstruction);
        }
    }
    // Display if we are not writing to an output folder:
    if(argc == 2) {
        waitKey(0);
    }
    return 0;
}

Display or save the image reconstruction at some predefined steps:
for(int num_component = 0; num_component < min(16, W.cols); num_component++) {
    // Slice the Fisherface from the model:
    Mat ev = W.col(num_component);
    Mat projection = subspaceProject(ev, mean, images[0].reshape(1,1));
    Mat reconstruction = subspaceReconstruct(ev, mean, projection);
    // Normalize the result:
    reconstruction = norm_0_255(reconstruction.reshape(1, images[0].rows));
    // Display or save:
    if(argc == 2) {
        imshow(format("fisherface_reconstruction_%d", num_component),
reconstruction);
    } else {
        imwrite(format("%s/fisherface_reconstruction_%d.png",
output_folder.c_str(), num_component), reconstruction);
    }
}

/*
 * Copyright (c) 2011. Philipp Wagner <bytefish[at]gmx[dot]de>.
 * Released to public domain under terms of the BSD Simplified license.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions are met:
 *   * Redistributions of source code must retain the above copyright
 *     notice, this list of conditions and the following disclaimer.
 *   * Redistributions in binary form must reproduce the above copyright
 *     notice, this list of conditions and the following disclaimer in the
 *     documentation and/or other materials provided with the distribution.
 *   * Neither the name of the organization nor the names of its contributors
 *     may be used to endorse or promote products derived from this software
```

```
 *     without specific prior written permission.
 *
 *   See <http://www.opensource.org/licenses/bsd-license>
 */

#include "opencv2/core/core.hpp"
#include "opencv2/contrib/contrib.hpp"
#include "opencv2/highgui/highgui.hpp"

#include <iostream>
#include <fstream>
#include <sstream>

using namespace cv;
using namespace std;

static void read_csv(const string& filename, vector<Mat>& images, vector<int>&
labels, char separator = ';') {
    std::ifstream file(filename.c_str(), ifstream::in);
    if (!file) {
        string error_message = "No valid input file was given, please check
the given filename.";
        CV_Error(CV_StsBadArg, error_message);
    }
    string line, path, classlabel;
    while (getline(file, line)) {
        stringstream liness(line);
        getline(liness, path, separator);
        getline(liness, classlabel);
        if(!path.empty() && !classlabel.empty()) {
            images.push_back(imread(path, 0));
            labels.push_back(atoi(classlabel.c_str()));
        }
    }
}

int main(int argc, const char *argv[]) {
    // Check for valid command line arguments, print usage
    // if no arguments were given.
    if (argc != 2) {
        cout << "usage: " << argv[0] << " <csv.ext>" << endl;
        exit(1);
    }
    // Get the path to your CSV.
    string fn_csv = string(argv[1]);
    // These vectors hold the images and corresponding labels.
    vector<Mat> images;
    vector<int> labels;
    // Read in the data. This can fail if no valid
    // input filename is given.
    try {
        read_csv(fn_csv, images, labels);
    } catch (cv::Exception& e) {
        cerr << "Error opening file \"" << fn_csv << "\". Reason: " << e.msg
<< endl;
        // nothing more we can do
        exit(1);
    }
    // Quit if there are not enough images for this demo.
```

```
    if(images.size() <= 1) {
        string error_message = "This demo needs at least 2 images to work.
Please add more images to your data set!";
        CV_Error(CV_StsError, error_message);
    }
    // Get the height from the first image. We'll need this
    // later in code to reshape the images to their original
    // size:
    int height = images[0].rows;
    // The following lines simply get the last images from
    // your dataset and remove it from the vector. This is
    // done, so that the training data (which we learn the
    // cv::FaceRecognizer on) and the test data we test
    // the model with, do not overlap.
    Mat testSample = images[images.size() - 1];
    int testLabel = labels[labels.size() - 1];
    images.pop_back();
    labels.pop_back();
    // The following lines create an LBPH model for
    // face recognition and train it with the images and
    // labels read from the given CSV file.
    //
    // The LBPHFaceRecognizer uses Extended Local Binary Patterns
    // (it's probably configurable with other operators at a later
    // point), and has the following default values
    //
    //      radius = 1
    //      neighbors = 8
    //      grid_x = 8
    //      grid_y = 8
    //
    // So if you want a LBPH FaceRecognizer using a radius of
    // 2 and 16 neighbors, call the factory method with:
    //
    //      cv::createLBPHFaceRecognizer(2, 16);
    //
    // And if you want a threshold (e.g. 123.0) call it with its default
values:
    //
    //      cv::createLBPHFaceRecognizer(1,8,8,8,123.0)
    //
    Ptr<FaceRecognizer> model = createLBPHFaceRecognizer();
    model->train(images, labels);
    // The following line predicts the label of a given
    // test image:
    int predictedLabel = model->predict(testSample);
    //
    // To get the confidence of a prediction call the model with:
    //
    //      int predictedLabel = -1;
    //      double confidence = 0.0;
    //      model->predict(testSample, predictedLabel, confidence);
    //
    string result_message = format("Predicted class = %d / Actual class =
%d.", predictedLabel, testLabel);
    cout << result_message << endl;
    // Sometimes you'll need to get/set internal model data,
    // which isn't exposed by the public cv::FaceRecognizer.
    // Since each cv::FaceRecognizer is derived from a
```

```
    // cv::Algorithm, you can query the data.
    //
    // First we'll use it to set the threshold of the FaceRecognizer
    // to 0.0 without retraining the model. This can be useful if
    // you are evaluating the model:
    //
    model->set("threshold", 0.0);
    // Now the threshold of this model is set to 0.0. A prediction
    // now returns -1, as it's impossible to have a distance below
    // it
    predictedLabel = model->predict(testSample);
    cout << "Predicted class = " << predictedLabel << endl;
    // Show some informations about the model, as there's no cool
    // Model data to display as in Eigenfaces/Fisherfaces.
    // Due to efficiency reasons the LBP images are not stored
    // within the model:
    cout << "Model Information:" << endl;
    string model_info = format("\tLBPH(radius=%i, neighbors=%i, grid_x=%i,
grid_y=%i, threshold=%.2f)",
            model->getInt("radius"),
            model->getInt("neighbors"),
            model->getInt("grid_x"),
            model->getInt("grid_y"),
            model->getDouble("threshold"));
    cout << model_info << endl;
    // We could get the histograms for example:
    vector<Mat> histograms = model->getMatVector("histograms");
    // But should I really visualize it? Probably the length is interesting:
    cout << "Size of the histograms: " << histograms[0].total() << endl;
    return 0;
}
```