PERANCANGAN DAN PEMBUATAN PERANGKAT LUNAK PENGENALAN WAJAH DENGAN METODE MIXTURE DISTANCE

TUGAS AKHIR

Oleh :

DENI HIDAYAT
NRP. 2694.100.076

JURUSAN TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA
2000

PERANCANGAN DAN PEMBUATAN PERANGKAT LUNAK PENGENALAN WAJAH DENGAN METODE MIXTURE DISTANCE

TUGAS AKHIR

Diajukan Guna Memenuhi Sebagian Persyaratan
Untuk Memperoleh Gelar Sarjana Komputer
Pada
Jurusan Teknik Informatika
Fakultas Teknologi Industri
Institut Teknologi Sepuluh Nopember
Surabaya

Mengetahui / Menyetujui

Dosen Pembimbing I	Dosen Pembimbing II
Dr. Ir. Supeno Djanali	**Rully Soelaiman, S.Kom.**
NIP. 130 368 610	**NIP. 132 085 802**

SURABAYA
Februari, 2000

ABSTRAKSI

ABSTRAKSI

Banyak metode yang dapat digunakan untuk merancang suatu sistem pengenalan wajah, pada tugas akhir ini dirancang sebuah perangkat lunak pengenalan wajah dengan metode *mixture-distance*. Metode *mixture-distance* ini menggunakan pendekatan statistik dalam algoritma pengenalannya, disini sebuah wajah direpresentasikan kedalam sebuah vektor 30 dimensi yang didapat dari 35 titik pengukuran. Kumpulan dari vektor wajah tersebut akan membentuk suatu data training yang dimodelkan sebagai *mixture of normal densities*. Wajah yang akan dikenali dianggap sebagai suatu vektor yang akan diidentifikasi dengan menghitung probabilitas dan distance dari vektor tersebut terhadap setiap elemen data training yang ada.

KATA PENGANTAR

KATA PENGANTAR

Alhamdulillah, segala puji dan syukur kehadirat Allah SWT karena atas kehendak dan restu-Nya penulis dapat menyelesaikan Tugas Akhir ini yang berjudul :

Perancangan dan Pembuatan Perangkat Lunak

Pengenalan Wajah Dengan Metode Mixture Distance

Tugas Akhir ini merupakan syarat untuk menyelesaikan studi program sarjana pada jurusan Teknik Informatika, Fakultas Teknologi Industri, Institut Teknologi Sepuluh Nopember, Surabaya.

Penulis menyadari terselesaikannya Tugas Akhir ini tidak lepas dari bantuan banyak pihak, oleh karena itu penulis ingin menyampaikan terima kasih yang sebesar-besarnya pada semua pihak yang telah membantu pembuatan Tugas akhir ini khususnya kepada :

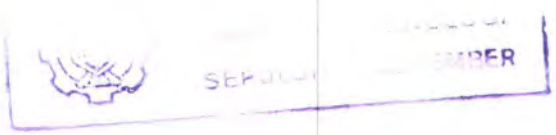
1. Kedua orang tua dan kakak-kakakku atas segala dukungan baik moril maupun materil.
2. Bapak Dr. Ir. Supeno Djanali, sebagai dosen pembimbing yang telah banyak membantu penyelesaian Tugas Akhir ini.
3. Bapak Rully Soelaiman, S.Kom. yang telah banyak meluangkan waktunya untuk membimbing dan memotivasi penulis untuk menyelesaikan Tugas Akhir ini.
4. Bapak Dr. Ir. Arif Djunaedi, M.Sc. selaku ketua jurusan Teknik Informatika ITS.

5. Seluruh dosen teknik informatika ITS atas ilmu yang telah diberikan selama perkuliahan.
6. RSK. Damayanti, Rama dan Ibu yang telah banyak memberikan bantuan dan dorongan selama penulis menempuh kuliah di ITS ini.
7. Bapak Husen dan Ibu Dewi yang telah meluangkan waktunya untuk memberikan masukan dan bimbingan yang sangat penulis butuhkan.
8. Waskitho, temanku disegala hal yang telah banyak membantu selama perkuliahan sampai akhir ujian TA, memberi dorongan semangat, nasehat dan menemani penulis bermain game, thanks berat atas semua bantuannya. Trim's juga buat Titanya yang telah minjami printer. Viva MU !, Hidup FIFA 99 !!.
9. Anton, Agus Subhan, Hamsy dan Myrna 'thanks bukunya' yang walaupun telah lulus tetap memberikan bantuan dan masukan kepada penulis.
10. Teman-teman seperjuangan Firman, Firdaus, Waskitho, Yanuar Bagus, Hamsy, Yoyok, Rizal, Anton, Adi dan Eko thanks berat atas bantuannya selama ini.
11. Rekan-rekan C0A, Anjeb, Riza, Aflah, Luthfi, Arif, Heri, Salman, Dina, Intan, Yetty, Ika, Santhy, Gusmang, Yudi, Dudi 'thanks atas kesediaannya sebagai moderator' dan seluruh rekan C0A baik yang masih ada atau yang udah cabut.
12. Tom, PW dan seluruh rekan-rekan penghuni lab AJK yang tetap setia menjaga lab kita yang tercinta. VIVAT AJK !!.
13. Seluruh staf Tata Usaha teknik informatika ITS.
14. Semua pihak yang tidak sempat penulis sebutkan disini.

Penulis menyadari bahwa dalam pembuatan Tugas Akhir ini masih banyak kekurangan dan jauh dari sempurna. Oleh karena itu penulis akan menerima adanya kritik dan saran yang membangun demi kesempurnaan Tugas Akhir ini. Akhir kata semoga Tugas Akhir ini dapat memberikan manfaat bagi semua pihak, Amin.

Surabaya, Februari 2000

Penulis



DAFTAR ISI

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR TABEL

BAB I

PENDAHULUAN

BAB I

PENDAHULUAN

1.1 Latar Belakang

Perkembangan teknologi komputer dan informasi membawa dampak positif kesegala bidang kehidupan, antara lain bidang keamanan. Dengan memanfaatkan teknologi komputer dapat dibuat suatu alat pengamanan yang dapat melakukan identifikasi terhadap seseorang secara otomatis dan akurat sehinngga mempunyai tingkat keamanan yang tinggi, salah satu aplikasinya adalah sistem pengenalan wajah (*face recognition*).

Sistem pengenalan wajah merupakan bagian dari metode pengenalan pola (*pattern recognition*) yang telah banyak dibuat, dalam pengenalan pola selain wajah, sidik jari dan suara juga banyak digunakan untuk proses identifikasi. Metode pengenalan pola ini selain dapat digunakan sebagai alat pengamanan yang berfungsi sebagai kunci akses terhadap sesuatu, juga dapat digunakan untuk membantu kepolisian misalnya untuk mengenali foto atu sidik jari seseorang yang diduga sebagai pelaku tindak kejahatan.

Setiap individu mempunyai ciri-ciri atau karakteristik yang unik dan berbeda pada setiap orang, ciri-ciri itulah yang biasanya sering digunakan sebagai *feature* dalam sistem pengenalan pola untuk keperluan identifikasi suatu obyek yang diamati. Ciri-ciri fisik yang unik dan dapat digunakan untuk identifikasi antara lain sidik jari, wajah, telapak tangan dan suara. Dari ciri-ciri tersebut, wajah merupakan ciri yang agak kompleks karena banyaknya informasi yang terdapat

didalamnya, antara lain bentuk wajah, jarak antar mata (*inter iris*), panjang dan lebar alis, rahang, warna kulit dan informasi yang lainnya [5,8]. Informasi-informasi tersebut dapat digunakan sebagai variabel-variabel yang dipakai dalam pengenalan wajah.

Dengan memanfaatkan ciri-ciri fisik yang unik dari setiap wajah seseorang maka dapat dibuat suatu sistem untuk pengenalan wajah (*face recognition*), sistem pengenalan wajah ini telah banyak dibuat dengan berbagai macam pendekatan, namun secara garis besar pendekatan yang biasa digunakan dalam algoritma pengenalan wajah (*face recognition*) antara lain pendekatan secara statistik dan jaringan syaraf tiruan (*neural network).* Dari beberapa pendekatan yang telah dipakai tidak ada satupun yang dapat menghasilkan sistem pengenalan wajah yang sempurna.

Sistem pengenalan wajah harus dapat mengatasi perubahan-perubahan atau variasi yang terjadi pada gambar suatu objek, variasi-variasi tersebut biasanya terjadi akibat distorsi pada saat pengambilan gambar, dimana pada saat pengambilan gambar ekspresi wajah seseorang dapat berubah-ubah seperti sedih, tertawa dan marah, selain itu posisi wajahpun dapat berubah miring, menoleh kekiri-kanan, menengadah atau menunduk, selain ekspresi dan posisi, jarak objek dari kamera dan pencahayaan (*illumination*) saat pengambilan gambar juga cukup berpengaruh. Idealnya sistem pengenalan wajah harus dapat mengatasi semua noise dan distorsi tersebut, namun kenyataannya tidak satupun pendekatan yang dapat mengatasinya dengan sempurna, setiap pendekatan mempunyai kelebihan dan kelemahan masing-masing yang satu sama lain dapat saling melengkapi,

pendekatan secara statistik dapat digunakan karena pada prinsipnya informasi-informasi yang didapat dari suatu wajah dapat dianggap sebagai suatu kumpulan dari data-data yang mempunyai pola tertentu. Dengan begitu pengenalan pola secara statistik (*statistical pattern recognition)* dapat digunakan untuk mengenali pola data tertentu yang akan diidentifikasi.

Berdasarkan cara ekstraksi featurenya, secara garis besar pendekatan pengenalan citra wajah dapat dibagi dalam dua pendekatan yaitu pengenalan berdasarkan ciri-ciri (*feature based recognition*) yang mengambil suatu himpunan feature (*feature set*) yang lebih kecil dibanding dengan jumlah pixel dari citra wajah, dan pendekatan kedua yaitu direct image method yang tidak melibatkan tahap ekstraksi feature [5]. Sebenarnya direct image method juga melakukan ekstraksi feature bedanya feature-featurenya terpengaruh secara signifikan oleh variasi pencahayaan (*illumination*) dan umumnya feature pada direct image method berukuran besar, misalnya berdasarkan pixel-pixel pada citra wajah tersebut, sedangkan pendekatan dengan berdasarkan ciri-ciri (*feature based*) secara umum tidak terpengaruh oleh kondisi pencahayaan dan ukuran featurenya jauh lebih kecil.

Beberapa contoh aplikasi yang menggunakan direct image method antara lain *template matching* dan *eigenfaces* oleh Turk dan Pentland. Template matching hanya efektif jika query dan image model mempunyai skala, orientasi, dan pencahayaan (*illumination*) yang sama. Sistem pengenalan wajah dengan menggunakan metode *mixture distance* ini termasuk kepada pendekatan pertama yaitu pengenalan wajah berdasarkan ciri-ciri (*feature based face recognition*).

Dalam tugas akhir ini dibuat suatu perangkat lunak pengenalan wajah dengan menggunakan metode *Mixture-Distance*, metode *mixture-distance* ini merupakan suatu algoritma pengenalan wajah dengan berdasarkan pada ciri-ciri fisik dari wajah seseorang sebagai informasi untuk membentuk pola data tertentu, yang kemudian diolah dengan menggunakan pendekatan secara statistik. Perangkat lunak pengenalan wajah ini dapat digunakan untuk mengidentifikasi wajah seseorang yang telah disimpan datanya dalam suatu database, dimana setiap wajah direpresentasikan dalam suatu vektor feature 30 dimensi yang didapat dari 35 titik pengukuran. Vektor itulah yang disimpan dalam database dan dimodelkan secara statistik, kemudian akan digunakan dalam proses pengenalan jika ada vektor baru yang akan diidentifikasi.

1.2 Tujuan

Tujuan dari pembuatan Tugas Akhir ini adalah untuk membuat suatu perangkat lunak pengenalan wajah dengan berdasarkan ciri-ciri fisik dari wajah seseorang dengan metode *mixture-distance*, yang pada implementasinya menggunakan pendekatan statistik. Perangkat lunak ini harus dapat mengatasi perubahan mimik ataupun perbedaan posisi wajah pada suatu objek yang sama dengan mengenalinya sebagai individu yang sama.

1.3 Permasalahan

Beberapa permasalahan yang akan dibahas dalam Tugas Akhir ini antara lain :

- Mendapatkan vektor feature yang merupakan representasi dari suatu citra wajah.
- Mampu memodelkan data training dalam hal ini kumpulan vektor feature dalam database kedalam sebuah model yang cukup representatif.
- Dapat mengatasi variasi-variasi dan distorsi yang terjadi pada citra wajah, sehingga dapat melakukan pengenalan dengan optimal.

1.4 Batasan Masalah

Batasan – batasan yang dipakai dalam pembuatan Tugas Akhir ini antara lain :

- Input berupa data citra dengan format bitmap (bmp).
- Input citra berukuran minimal 92 x 112.
- Input citra berupa gambar wajah penuh.
- Semua titik pengukuran harus dapat terlihat.
- Ekstraksi dilakukan secara manual.

1.5 Metodologi Penelitian

Metodologi yang dipakai dalam menyelesaikan Tugas Akhir ini adalah :

- Studi Literatur

  Mempelajari semua literatur-literatur yang berhubungan dengan masalah yang dihadapi, memahami konsep atau teori yang akan digunakan dalam pembuatan Tugas Akhir ini.

- Perancangan Sistem

  Merancang sistem yang akan dibuat meliputi perancangan database, struktur data, pemodelan data dan alur proses.

- Pembuatan Perangkat Lunak

  Pembuatan perangkat lunak merupakan implementasi dari sistem yang telah kita rancang, dan algoritma yang kita gunakan serta perancangan input dan output program.

- Evaluasi Perangkat Lunak

  Setelah perangkat lunak selesai dibuat langkah selanjutnya adalah melakukan evaluasi atau pengujian, hal ini dimaksudkan untuk mengetahui sampai sejauh mana perangkat lunak ini bekerja, apakah sesuai dengan yang kita harapkan atau masih ada kesalahan.

- Perbaikan Perangkat Lunak

  Setelah evaluasi dapat diketahui apakah perlu dilakukan perbaikan, jika masih ditemukan kekurangan atau kesalahan maka perbaikan harus dilakukan.

- Pembuatan Laporan

  Pembuatan laporan meliputi keseluruhan Tugas Akhir yang telah dibuat, termasuk penjelasan teori-teori yang digunakan, perancangan sistem dan evaluasi dan pengujian sistem.

1.6 Sistematika Pembahasan

Dalam penyusunan laporan tugas akhir ini, sistematika pembahasan yang dipakai adalah sebagai berikut :

BAB I, **Pendahuluan,** menjelaskan latar belakang, permasalahan, batasan masalah, tujuan, metodologi penelitian, dan sistematika pembahasan dari pembuatan Tugas Akhir ini.

BAB II, **Teori Penunjang,** membahas mengenai teori-teori dan konsep yang akan diterapkan dalam pembuatan perangkat lunak pengenalan wajah dan berhubungan erat dengan pembahasan pada bab-bab selanjutnya.

BAB III, **Mixture – Distance,** membahas dengan lebih jelas metode mixture-distance, pembahasan meliputi teknik pemodelan data yang digunakan, mixture density dan probabilitas.

BAB IV, **Pengenalan Wajah dengan Metode Mixture-Distance,** membahas secara jelas setiap tahapan yang ada didalam sistem pengenalan wajah dengan metode mixture-distace.

BAB V, **Perancangan dan Pembuatan Perangkat Lunak,** menguraikan perancangan sistem pengenalan wajah meliputi aliran data, aliran proses, struktur data dan modul-modul penting yang digunakan.

BAB VI, **Ujicoba dan Evaluasi Sistem,** berisi hasil uji coba yang dilakukan terhadap perangkat lunak yang dibuat dengan data-data yang telah tersedia.

BAB VII, Penutup, merupakan kesimpulan atas keseluruhan tugas akhir yang telah dibuat disertai saran-saran untuk kemungkinan pengembangan selanjutnya.

BAB II

TEORI PENUNJANG

BAB II

TEORI PENUNJANG

Pada bab ini akan dibahas beberapa konsep dan teori dasar yang akan digunakan dalam implementasi tugas akhir ini, teori-teori penunjang ini berkaitan erat dengan pembahasan dibab-bab selanjutnya, bab ini juga membahas secara singkat beberapa pendekatan yang telah banyak dilakukan berkaitan dengan sistem pengenalan pola.

2.1 Vektor dan Matriks

Beberapa hal yang dipahami mengenai vektor dan matriks dalam kaitannya dengan pembuatan tugas akhir karena vektor dan matriks ini akan banyak digunakan dalam pembahasan pada bab-bab selanjutnya.

2.1.1 Ruang Vektor

Ruang vektor dapat didefinisikan sebagai kumpulan vektor (V) dimana elemen-elemennya memenuhi sifat-sifat sebagai berikut :

- Untuk setiap $a,b \in R$ dan $u,v \in V$ berlaku $au+bv \in V$.
- Terdapat elemem $0 \in V$ sedemikian hingga :
  a. Untuk setiap $u \in V$; $0u = 0$ dan
  b. Untuk setiap $u \in V$; $0 + u = u$.

Setiap elemen dalam ruang vektor disebut vektor dan 0 disebut dengan vektor nol. Pada pembahasan ini vektor dipandang sebagai sebuah fungsi dan beberapa ruang vektor yang digunakan adalah :

- $L^2(R)$ melambangkan ruang vektor dari fungsi f(x) yang terukur dan terintegral secara kuadrat. Untuk $f(x),g(x) \in L^2(R)$, maka inner product dari $f(x)$ dengan $g(x)$ ditulis sebagai :

$$\langle f(x),g(x)\rangle = \int_{-\infty}^{+\infty} f(x)\,\overline{g(x)}\,dx\,, \qquad (2.1)$$

dengan $\overline{g(x)}$ merupakan kompleks konjugate dari $g(x)$.

- $L^2(R)^2$ adalah ruang vektor dari fungsi f(x,y) yang terukur dan terintegral secara kuadrat. Untuk $f(x,y),g(x,y) \in L^2(R^2)$,*inner product* $f(x,y)$ dan $g(x,y)$ ditulis sebagai :

$$\langle f(x,y),g(x,y)\rangle = \int_{-\infty}^{\infty} f(x,y)\overline{g(x,y)}dxdy\,, \qquad (2.2)$$

2.1.2 Ruang –n Euclidis

Jika n adalah sebuah bilangan bulat positif, maka sebuah tupel-n-terorde (ordered-n-tuple) adalah sebuah urutan dari n bilangan riel $(a_1,a_2,\ldots,a_n)$. Himpunan dari semua tupel-n-terorde dinamakan ruang-n dan dinyatakan dengan R^n. Bila n=2 atau 3, maka biasa digunakan istilah pasangan terorde dan tripel terorde, bila n=1, setiap tupel-n terorde terdiri dari satu bilangan riel, sehingga dapat dipandang sebagi himpunan bilangan riel, biasa ditulis R bukan R^1.

Berdasarkan analogi rumus-rumus yang sudah dikenal dalam R^2 dan R^3, maka dapat didefinisikan sebuah *norma Euclidis* atau panjang Euclidis dari sebuah vektor $u = (u_1, u_2, \ldots, u_n)$ didalam R^n adalah :

$$\| u \| = (u \cdot u)^{\frac{1}{2}} = \sqrt{u_1^2 + u_2^2 + \cdots + u_n^2} \qquad (2.3)$$

demikian juga, *jarak Euclidis* diantara titik $u = (u_1, u_2, \ldots, u_n)$ dan titik $v = (v_1, v_2, \ldots, v_n)$ didalam R^n didefinisikan oleh

$$d(u,v) = \| u - v \| = \sqrt{(u_1 - v_1)^2 + (u_2 - v_2)^2 + \cdots + (u_n - v_n)^2} \qquad (2.4)$$

2.1.3 Norm dan Normalisasi

Norm merupakan sebuah fungsi yang digunakan untuk mengukur panjang dari sebuah vektor. Norm dari $f(x) \in L^2(R)$ didefinisikan sebagai :

$$\| f \|^2 = \int_{-\infty}^{\infty} | f(x) |^2 \, dx \qquad (2.5)$$

Sebuah vektor u dikatakan ternormalisasi jika norm dari vektor u sama dengan satu, atau :

$$\|u\| = 1$$

Sebuah basis orthogonal yang ternormalisasi dalam norm $L^2(R)$ disebut dengan basis orthonormal, atau dengan kata lain basis $u_1, u_2, \ldots$ orthonormal jika :

$$\langle u_i, u_j \rangle = \partial_{ij} \text{ ; dengan } \partial_{ij} = \begin{cases} 1; untuk..i = j \\ 0; utk..lainnya \end{cases}$$

2.1.4 Matriks

Definisi sederhana sebuah matriks adalah sebuah susunan segi empat siku-siku dari bilangan-bilangan. Bilangan-bilangan di dalam susunan tersebut dinamakan entri dari matriks. Ukuran didalam matriks dijelaskan dengan menyatakan banyaknya baris (garis horisontal) dan banyaknya kolom (garis vertikal) yang terdapat dalam matriks tersebut. Jika A adalah sebuah matriks, maka a_{ij} menyatakan entri yang terdapat didalam baris ke i dan kolom ke j. Sebagai contoh matriks dengan ukuran 2 x 3 dapat ditulis sebagai berikut :

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \end{bmatrix}$$

secara umum sebuah matriks B_{mxn} dapat ditulis :

$$B = \begin{bmatrix} b_{11} & b_{12} & \cdots & b_{1n} \\ b_{21} & b_{22} & \cdots & b_{2n} \\ \vdots & \vdots & & \vdots \\ b_{m1} & b_{m2} & \cdots & b_{mn} \end{bmatrix}$$

Pada dasarnya sebuah vektor merupakan sebuah matrik dengan ukuran 1xn, karena setiap baris atau kolom dari sebuah matrik dapat dianggap sebuah vektor, jadi matriks dapat dibentuk dari sekumpulan vektor.

contoh matriks :

$$\begin{bmatrix} 2 & 1 & 3 \\ 3 & 3 & 4 \\ 1 & 0 & 7 \end{bmatrix} \quad \begin{bmatrix} 3 & 2 & 0 \\ 4 & 5 & 1 \end{bmatrix} \quad \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

(a) (b) (c) (d)

matrik (a) merupakan matriks dengan ukuran 3x3, matriks (b) berukuran 2x3 sedangkan matriks (c) dengan bilangan 1 berada pada diagonal utamanya disebut dengan matriks satuan atau matriks identitas (*identity matrix*) yaitu suatu matriks kuadrat yang mempunyai nilai entri 1 pada diagonal utamanya dan 0 pada entri lainnya dan dinyatakan dengan 1, jika ukurannya penting, ditulis dengan I_n untuk matriks ukuran nxn. Matriks (d) yang semua entrinya bernilai 0 disebut matriks nol (*zero matrix*), dinyatakan dengan 0 atau jika ukurannya penting ditulis 0_{mxn}. Matriks yang mempunyai banyaknya baris sama dengan banyaknya kolom disebut dengan *matriks kuadrat berorde n* (*square matrix of orde n*) atau matriks bujur sangkar seperti matriks (a), (c), (d), dan entri-entri a_{11}, a_{22}, ..., a_{nn} dinyatakan berada dalam *diagonal utama* dari matriks tersebut.

Dua buah matriks dapat melakukan operasi perkalian jika kolom matriks pertama sama dengan baris matriks kedua, dan hasilnya berupa matriks dengan jumlah baris sesuai matriks yang pertama dan kolom sesuai matriks yang kedua, atau lebih jelasnya $A_{mxn} \times B_{nxk} = C_{mxk}$. Berikut ini beberapa kaidah penting yang berlaku dalam ilmu hitung matriks, diasumsikan ukuran-ukuran matriks sedemikian sehinggga operasi-operasi ini dapat dilakukan :

(a) $A+B=B+A$ (hukum komutatif penambahan)

(b) $A+(B+C)=(A+B)+C$ (hukum asosiatif penambahan)

(c) $A(BC)=(AB)C$ (hukum asosiatif perkalian)

(d) $A(B+C)=AB+AC$ (hukum distributif)

Dari kaidah-kaidah yang berlaku dalam ilmu hitung matriks banyak kaidah-kaidah ilmu hitung bilangan riel juga berlaku, namun ada beberapa perkecualian. Salah

satu dari kekecualian yang terpenting terjadi dalam perkalian matriks, dimana dalam bilangan riel a dan b berlaku hukum komutatif ab=ba, akan tetapi kaidah ini tidak berlaku untuk ilmu hitung matriks. Untuk matriks-matriks A dan B, AB dan BA tidak perlu sama walaupun AB dan BA mempunyai ukkuran yang sama.

Jika A sebuah matriks kuadrat dan I adalah matriks satuan, dan jika AxB=I maka A dapat dibalik (*invertible*) dan B merupakan invers matriks A. Sebuah matriks yang dapat dibalik mempunyai tepat satu invers, dinyatakan dengan persamaan $A A^{-1} = I$ dan $A^{-1} A = I$. Invers dari suatu matriks memainkan peranan dalam ilmu hitung matriks yang sangat menyerupai peranan yang dimainkan oleh kebalikan a^{-1} didalam hubungan hubungan numerik $aa^{-1}=1$ dan $a^{-1}a=1$.

2.2 Pengenalan Pola

Pengenalan pola dapat dilakukan dengan berbagai macam pendekatan tergantung dari informasi yang tersedia dan tujuan yang hendak dicapai, pada intinya seluruh pendekatan dalam pengenalan pola tertuju pada pengklasifikasian suatu objek tertentu. Setiap objek yang akan dikenali mempunyai informasi-informasi yang membentuk suatu pola tertentu, sedangkan informasi itu sendiri bisa berupa suara, sinyal, data medis ataupun citra wajah. Informasi-informasi itu juga yang secara alami digunakan manusia untuk mengenali alam sekitarnya, baik manusia atau hewan secara alamiah dapat mengklasifikasikan suatu objek tertentu berdasarkan informasi yang terdapat didalamnya.

Aplikasi pengenalan pola telah banyak dimanfaatkan dalam berbagai bidang kehidupan antara lain keamanan, misalnya untuk verifikasi tanda tangan,

sidik jari atau pengenalan wajah, selain bidang keamanan juga dapat digunakan untuk menganalisa data medis untuk mendiagnosa suatu penyakit tertentu, analisa getaran gempa dan pengenalan suara juga merupakan bagian dari aplikasi pengenalan pola yang telah dikembangkan selama ini. Dalam perkembangannya beberapa macam pendekatan telah dilakukan dalam aplikasi pengenalan pola antara lain dengan pendekatan secara statistik (*statistical pattern recognition*) dan pendekatan dengan menggunakan jaringan syaraf tiruan (*neural network*).

2.2.1 Pengenalan Pola Secara Statistik (*Statistical Pattern Recognition*)

Dalam sudut pandang pengenalan pola statistik, biasanya diasumsikan pola dibentuk dari beberapa proses perubahan yang berlaku pada satu atau lebih titik awal. Biasanya titik awal (*starting point*) ini mewakili nilai rata-rata vektor pada class objek, dan proses perubahan diasumsikan dimodelkan dengan normal densities nilai rata-rata ini. Bila tersedia banyak member pada setiap class untuk training, maka dapat diramalkan sebuah vektor rata-rata dan matrix covariance untuk tiap class. Digabungkan dengan beberapa distribusi yang sebelumnya dari class-class tersebut, maka dapat dihitung posteriori probability dari sebuah vektor tak dikenal yang diberikan oleh model.

Pengenalan pola dengan pendekatan statistik berpijak pada teori-teori statistik dan probabilitas. Sekumpulan data yang didapat dari sebuah objek akan digunakan sebagai bahan pengenalan, data-data tersebut didapat dari feature-feature tertentu dari objek tersebut yang kemudian diekstrak menghasilkan data-data yang membentuk suatu vektor feature. Setiap vektor feature dari objek yang

berbeda akan menghasilkan pola yang berbeda pula, sehingga setiap objek mempenyai vektor feature yang membentuk pola tertentu. Sekumpulan vektor feature yang mempunyai pola hampir sama kemudian digolongkan kedalam sebuah class yang sama.

Pendekatan statistik biasanya digunakan untuk pola-pola yang mempunyai informasi yang terukur. Dengan memanfaatkan teori probabilitas dan konsep klasifikasi Bayesian maka pengenalan pola dapat dilakukan. Pendekatan statistik ini mempunyai peranan yang sangat besar dalam aplikasi pengenalan pola, karena teori-teori dan konsep statistik merupakan teknik analitis yang sangat penting dan dapat dimanfaatkan secara luas pada berbagai macam bidang kehidupan manusia.

2.2.2 Pengenalan Pola Menggunakan Jaringan Syaraf Tiruan (*Neural Network*)

Pendekatan dengan jaringan syaraf tiruan diilhami oleh sistem syaraf bilogis pada makhluk hidup yang mempunyai kemampuan pengenalan yang tingggi. Pengembangan metode jaringan syaraf tiruan ni melibatkan berbagai disiplin ilmu antara lain psikologi, ilmu syaraf, ilmu kedokteran dan disiplin ilmu lain yang terkait didalamnya.

Selain mempunyai kesamaan dalam srukturnya, jaringan syaraf tiruan juga memerlukan proses pembelajaran seperti halnya sistem syaraf yang sesungguhnya, namun arsitektur dan ukurannya jauh lebih sederhana. Jaringan syaraf tiruan terdiri dari berbagai macam arsitektur yang mempunyai spesifikasi sendiri. Pemanfaatan jaringan syaraf tiruan meliputi pengenalan pola dengan

kemampuan melakukan generalisasi data, klasifikasi pola ataupun optimasi. Karakteristik jaringan syaraf tiruan ditentukan oleh jumlah simpul, susunan simpul, koneksi serta proses belajarnya.

Jaringan syaraf tiruan merepresentasikan neuron, unit, sel atau node dengan elemen pemroses (*processing element* atau PE) yang saling terhubung dalam bentuk *directed graph* melalui jalur sinyal. Biasanya sejumlah elemen pemroses membentuk kumpulan tertentu yang disebut sebagai lapisan (*layer*). Metode pembelajaran pada dasarnya adlah proses pengubahan (update) nilai bobot pada koneksi antar elemen pemroses, hal ini akan menentukan karakteristik dan kemampuan jaringan syaraf tiruan.

Jaringan syaraf tiruan minimal tersusun atas lapisan input dan lapisan output. Dalam beberapa tipe jaringan diantara lapisan input dan lapisan output terdapat lapisan tersembunyi (*hidden layer*), setiap unit pada lapisan input akan terhubung kesemua unit dalam lapisan tersembunyi. Selanjutnya setiap unit pada lapisan tersembunyi nantinya akan dihubungkan kesemua unit pada lapisan output.

Pada saat ini pendekatan dengan menggunakan jaringan syaraf telah banyak dikembangkan dan sudah diterapkan dalam berbagai bidang seperti pengenalan tulisan tangan/huruf, pengenalan suara, segmentasi citra, diagnosa penyakit, pengenalan wajah dan sebagainya. Pendekatan jaringan syaraf tiruan diterapkan untuk memberikan solusi untuk masalah yang tidak dapat dirumuskan karena kompleksitasnya dan untuk pola-pola yang mempunyai informasi terukur dan saling terkait.

2.2.3 K-Means Clustering

Dalam pengenalan pola dengan menggunakan pendekatan statistik (*statistical pattern recognition*), pengelompokan data merupakan suatu tahapan proses yang sangat penting dalam kaitannya dengan klasifikasi pola. Dari kumpulan data yang tersedia dapat dikelompokan dalam beberapa klas berdasarkan pola tertentu. *Clustering* merupakan proses untuk mengelompokan data berdasarkan kemiripannya dan mengasosiasikannya kedalam suatu *cluster*. Prinsip dari *clustering* adalah memaksimalkan perbedaan antar *cluster* dan meminimkan perbedaan antar data dalam satu *cluste*r.

Algoritma k-means menginisialisasi dan memproses pusat *cluster* berdasarkan jarak pusat *cluster* dengan pola-pola sekitarnya. Pola-pola yang berdekatan jaraknya akan menjadi satu *cluste*r dengan pusat *cluster* berupa nilai rata-rata dari pola-pola tersebut. Berikut detail algoritma k-mean dengan jumlah cluster N, pada data set X yang berjumlah M :

1. Inisialisasi cluster $c_0 \ldots c_{N-1}$ dengan data vektor yang bisa dilakukan dengan random atau diurut dengan vektor pertama menjadi vektor *cluster* pertama. $C_i = X_i$ untuk $i = 0..N-1$.
2. Untuk semua vektor x, kelompokan vektor X(i) ke *cluster* j jika $\|x(i)-c_j\|^2 < \|x(i)-c_k\|^2$ untuk $k <> j$
3. Ubah vektor mean *cluster* ke i dengan persamaan sebagai berikut : $c_i = \frac{1}{n}\sum x(t)$ untuk $i = 0 \ldots M-1$, dimana n merupakan jumlah vektor x yang masuk sebagai anggota *cluster* ke-i.
4. Ulangi langkah 2 dan 3 sampai diperoleh hasil yang konvergen.

Keuntungan dari algoritma *k-means clutering* adalah hasil pengelompokan merupakan kumpulan data berdasarkan kemiripannya.

2.2.4 Algoritma Expectation Maximization

Algoritma EM biasa digunakan untuk problem *incomplete data* bila *direct maximization* pada *incomplete data likelihood* tidak dapat dilakukan. Misalkan X dan Y adalah dua ruang sampel, dan H adalah transformasi *many to one* dari X ke Y, kita asumsikan terdapat variabel random yang diamati y dalam Y yang terhubung dengan sebuah variabel random yang tidak diamati x oleh $y = H(x)$. Disini terdapat data "*copmlete*" x yang hanya diamati sebagian dalam bentuk "*incomplete data*" y.

Kita anggap $p(x \mid \theta)$ adalah distribusi parametrik dari x, dimana θ adalah vektor parameter-parameter yang nilainya diambil dari 9. Distribusi y dinotasikan dengan $q(y \mid \theta)$, juga diparameterkan oleh θ dimana

$$q(y \mid \theta) = \int_{H(x)=y} p(x \mid \theta) dx \qquad (2.6)$$

estimasi θ dari y adalah problem *incomplete data*.

Ide utama algoritma EM adalah bahwa dalam beberapa problem, estimasi θ akan mudah bila *complete* data x tersedia, sementara akan sulit bila hanya berdsarkan pada *incomplete* data y. Karena pada prakteknya hanya tersedia incomplete data y, maka tidak mungkin langsung dilakukan optimisasi *complete data likelihood* $\ln p(x \mid \theta)$, sebaliknya secara intuitif dapat dilakukan estimasi

$\log p(x \mid \theta)$ dari y dan menggunakan fungsi "*estimated*" likelihood ini untuk mendapatkan maximizer $\hat{\theta}$. Karena untuk estimasi *complete data likelihood* $\ln p(x \mid \theta)$ dibutuhkan θ, perlu digunakan pendekatan iteratif, pertama *estimate complete data likelihood* $\ln p(x \mid \theta)$ dengan nilai θ yang diketahui, kemudian *maximize* fungsi likelihood terhadap θ.

Menurut argumen heuristik ini langkah E dan M pada algoritma EM iteratif (dikenal juga dengan Generalized EM atau GEM) dapat dinyatakan secara formal sebagai :

Langkah E : menghitung

$$Q(\theta, \theta^{(p)}) = E[\log P(x \mid \theta) \mid y, \theta^{(p)}] \tag{2.7}$$

Langkah M : memilih

$$\theta^{(p+1)} \in \arg\max_{\theta \in \vartheta} Q(\theta, \theta^{(p)}) \tag{2.8}$$

dimana $\theta^{(p)}$ melambangkan nilai parameter yang diperoleh pada iterasi ke-p.

BAB III

MIXTURE DISTANCE

BAB III

MIXTURE-DISTANCE

3.1 Pendahuluan

Jika terdapat sebuah database dari *facial feature vector* $Y = \{y_i\}$, dimana setiap vektor berhubungan dengan orang yang berbeda, dan *query* q yang berisi sebuah *facial feature vector* dari orang tak dikenal yang diasumsikan berada dalam database Y, maka langkah yang akan dilakukan untuk mengenali *query* q dalam database Y adalah mencari y_i yang berhubungan dengan q.

Dengan mengabaikan *error*, dan mengasumsikan tidak ada dua orang yang benar-benar mirip, yang perlu dilakukan hanya mencari dalam database Y vektor yang tepat sama dengan q. Tetapi dalam prakteknya q tidak akan tepat sama dengan apapun di database Y secara sempurna, karena banyaknya *error*. *Error* ini dipengaruhi oleh proses ekstraksi feature yang berkaitan dengan *human operator* atau algoritma yang membangun feature vector dari sebuah photo, variasi dalam pose, karakteristik kamera dan variasi fisik dari subjek itu sendiri (ekspresi, tersenyum, sedih ,sakit dan lain-lain). Jadi proses error yang alamiah dan wajar tersebut dapat mempengaruhi cara kita dalam membandingkan *query* dengan setiap elemen database.

3.2 Analisa Data

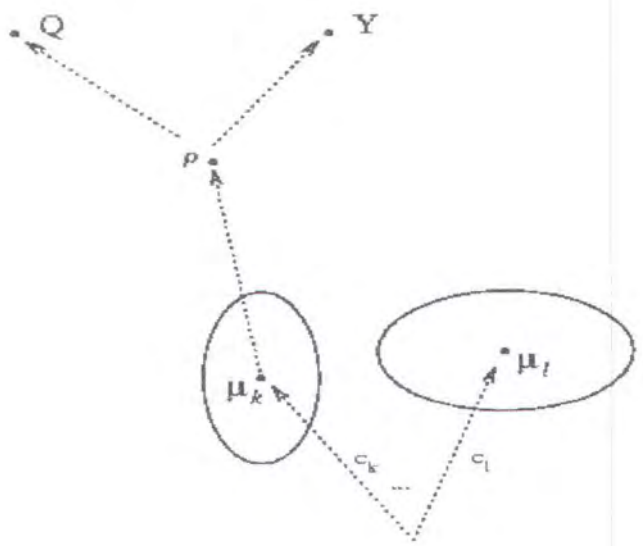
Dalam pengamatan *feature vector*, baik elemen database maupun *query* dapat dibayangkan sebagai hasil dari dua tahapan proses generatif. Tahap pertama

P menghasilkan vektor platonik p yang dapat diasumsikan sebagai representasi ideal dari setiap klas pola, dalam hal ini adalah feature wajah dari orang yang berbeda. Tahap kedua adalah proses pengamatan (*observation process*) yang menghasilkan vektor yang diamati secara terbatas. Tahap pertama berkaitan dengan variasi antar klas (*inter-class variation*), misalnya antar orang, sedangkan tahap kedua menangkap variasi dalam klas (*intra-class variation*). Secara alami proses kedua tergantung pada p dan dinotasikan dengan O_p. Lebih jauh setiap O_p diasumsikan sebagai sebuah *zero mean process*, yang secara konseptual menambahkan noise pengamatan pada vektor platonik sebagai pusatnya.

Probabilitas $\Pr(q \mid p)$ dimana sebuah *query* q telah digenerite oleh platonik p kemudian dihitung menggunakan bentuk vektor difference (q - p) dan mengevaluasi O_p, dinotasikan dengan $\Pr(q \mid p) \triangleq O_p(q-p)$. Dengan cara yang sama probabilitas $\Pr(y_i \mid p)$ dimana elemen database y_i digenerate oleh p adalah $O_p(y_i - p)$. Terakhir probabilitas $\Pr(p)$ dari p sendiri adalah $P(p)$. Untuk memutuskan semirip apakah q dan y_i pendekatan diambil dengan memfokuskan pada probabilitas dari *3-way joint event* [6] yang berisi pembentukan p, diikuti pengamatannya sebagai q, diikuti pengamatan independen kedua sebagai y_i. Integrasi atas p memberikan probabilitas yang menyatakan q dan y_i adalah pengamatan yang independen terhadap satu bentuk platonik.

Penyederhanaan yang dapat dilakukan adalah dengan mempertimbangkan y_i sebagai platonik. Ini merupakan asumsi yang aktual dan banyak dipakai dalam pengenalan pola dengan metode *nearest neighbor* [5]. *Query* dibayangkan sebagai

sebuah pengamatan dari elemen database – bukan dari elemen platonik ketiga. Diharapkan y_i tidak terlalu jauh dari p nya, dan distribusi pengamatan pada y_i mendekati distribusi pada p. Probabilitas $\Pr(q \mid y_i)$ dihitung untuk setiap y_i. Query diklasifikan berdasarkan probabilitas terbesar, ini sama dengan memaksimalkan $\Pr(y_i \mid q)$ dengan flat prior di Y.



Gambar 3.1 *3-way joint event*

3.2.1 Mixture Density

Sebuah distribusi normal dari sebuah vektor x di ruang R^n dapat dinyatakan dengan $N(\mu, \Sigma)$, dimana probabilitas *normal density*nya :

$$p(x) = (2\pi)^{-\frac{n}{2}} (\det(\Sigma))^{-\frac{1}{2}} \exp\left[-\frac{1}{2} . {}^{t}(x-\mu)\Sigma^{-1}(x-\mu) \right] \qquad (3.1)$$

Ket : Σ : Kovarian Matrik

μ : *Mean*

Pengamatan diasumsikan sebagai elemen dari R^n. Tanpa mempersempit lingkupnya, notasi akan ditetapkan pada *normal density multi-dimensional zero mean* yang didefinisikan sebagai :

$$N_\Sigma(x) = \frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}} \cdot e^{\frac{1}{2}x^t\Sigma^{-1}x} = \eta \cdot e^{\frac{1}{2}x^t\Sigma^{-1}x} \qquad (3.2)$$

dimana untuk memudahkan, konstanta di depan ditulis sebagai sebagai η.

Normal mixture kemudian didefinisikan sebagai $M = \{(c_k, \mu_k, N_{\Sigma k})\}$ sedemikian hingga $c_k \geq 0$, $\sum_{k=1}^{n} c_k = 1$, $\mu_k \in \Re^d$, dan matrix $\Sigma_k \geq 0$, misalnya sebuah operator semi-definite positif. M_k mengacu pada $(N_{\Sigma k}, \mu_k)$, dinyatakan dengan :

$$\Pr(x \mid M_k) = N_{\Sigma_k (x-\mu_k)} \qquad (3.2)$$

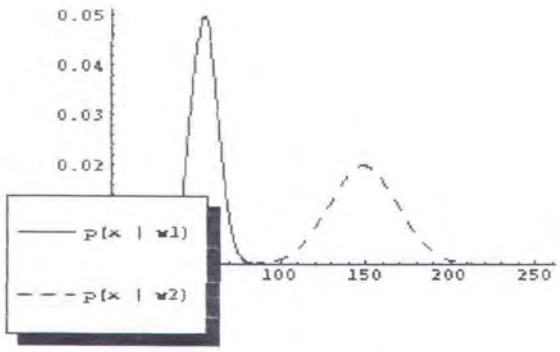
dimana :

$$\Pr(x|M_k) = \sum_k c_k . \Pr(x \mid M_k) = \sum_k c_k . N_{\Sigma_k}(x - \mu_k) \qquad (3.3)$$

Jika x sebuah feature vektor dan ω adalah sebuah klas pola, maka *mixture of normal density* atau *Gaussian Mixture* disini merupakan pdf (*probability density function*) dari vektor x dan dinyatakan dengan :

$$p(x) = \sum_{i=1}^{n} p(x \mid \omega_i) P(\omega_i) \qquad (3.5)$$

jelasnya $p(x|\omega_i)$ adalah distribusi feature x terhadap klas pola ω_i.



Gambar 3.2 Mixture Density

3.2.2 Bayes Rule

Aturan Bayes mengkonversikan sebuah *prior probabilities* $P(\omega_j)$ menjadi sebuah *posteriori probabilities* $P(\omega_j \mid x)$ dikondisikan saat pengamatan x. Dengan persamaan :

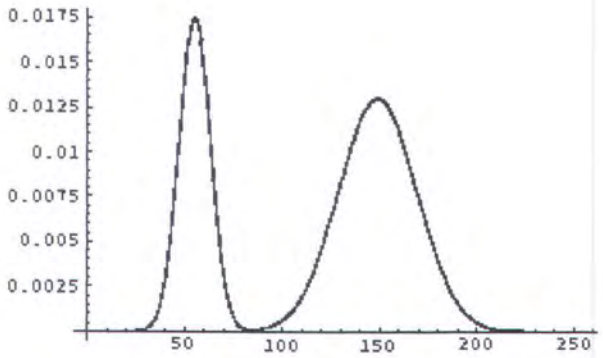
$$P(\omega_j \mid x) = \frac{p(x \mid \omega_j)P(\omega_j)}{p(x)} \tag{3.6}$$

dimana

$$p(x) = \sum_{i=0}^{n} p(x \mid \omega_i)P(\omega_i) \tag{3.7}$$

Tinjau sebuah *mixture density* dengan persamaan sebagai berikut : $p(x) = 0.35\,p(x \mid \omega_0) + 0.65\,p(x \mid \omega_1)$, dimana $p(x \mid \omega_0) = N(55,8)$ dan $p(x \mid \omega_0) = N(149,20)$, digambarkan dalam gambar 3.2. Dengan menggunakan aturan Bayes dapat dibuat suatu *decision rule* untuk menentukan keanggotaan sebuah vektor p_i dalam *mixture* sebagai berikut :

- tempatkan p_i dalam klas ω_1 jika $P(\omega_1 \mid x) > P(\omega_2 \mid x)$

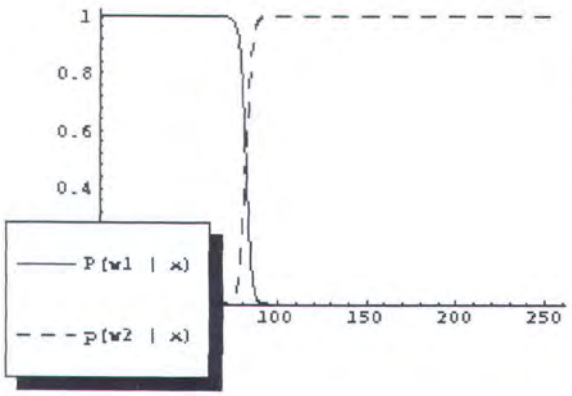


Gambar 3.3 Mixture Density dari 2 buah normal density

- tempatkan p_i pada klas ω_2 jika $P(\omega_2 \mid x) > P(\omega_1 \mid x)$

Decision rule tersebut dapat menyatakan *probability of error* mengikuti ketentuan berikut ini :

$$P(error \mid x) = \begin{cases} P(\omega_1 \mid x) & jika \quad memutuskan \quad \omega_2 \\ P(\omega_2 \mid x) & jika \quad memutuskan \quad \omega_1 \end{cases}$$



Gambar 3.4 Posteriori Probability

3.3 Mixture – Distance Function

Dalam *mixture-distance* fungsi jarak (*distance function*) dibentuk dengan memodelkan training data sebagai *mixture of normal densities*. Pemodelan dengan *mixture-distance* memunculkan dua alternatif pemilihan model yaitu *first order* (*variance*) dan *second order* (*covariance*).

Mixture-distance menggunakan sebuah mixture model M, yang merupakan kumpulan dari model-model probabilitas, M_1, M_2, ..., M_n dan parameter non negative c_1, c_2, ..., c_n yang dijumlahkan membentuk :

$$M(x) = \sum_{k=1}^{n} c_k M_k(x) \qquad (3.8)$$

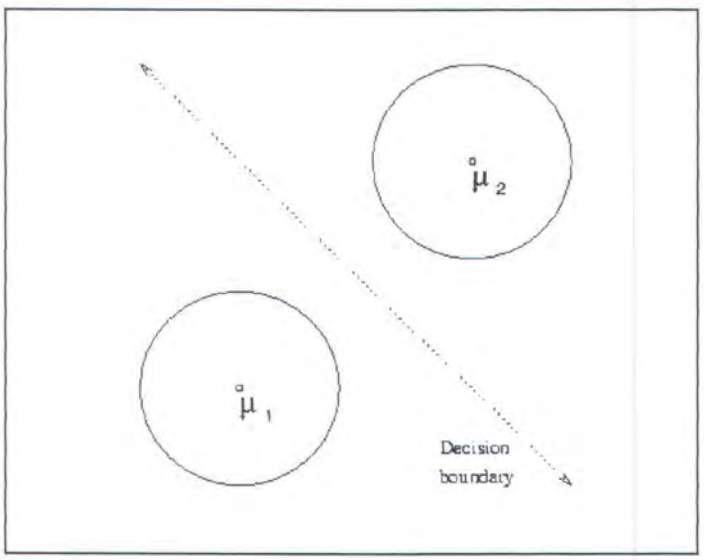
setiap element M_i dari M adalah *Gaussian* jadi M adalah *Gaussian* atau *Normal mixture*. *Normal mixture* kemudian didefinisikan sebagai $M = \{(c_k, \mu_k, N_{\Sigma k})\}$ sedemikian hingga $c_k \geq 0$, $\sum_k c_k = 1$, $\mu_k \in \Re^d$, dan matrix $\Sigma_k \geq 0$, misalnya sebuah operator semi-definite positif. $N_{\Sigma,\mu}(x)$ menyatakan *multi-variate normal density* (*Gaussian*) dengan kovarian Σ dan ekspektasi μ.

Jika diberikan sebuah himpunan vektor $x_1, \ldots, x_m$, untuk estimasi parameter normal mixture model yang menjelaskan data dengan baik, dapat dilakukan dengan menggunakan *k-means clustering* sebagai *starting point* dalam proses pembentukan klas sebagai komponen dari mixture.

Sekarang diasumsikan telah dibuat n-elemen normal mixture model M untuk memodelkan database elemen $\{y_i\}$, setiap elemen M_k adalah *normal density* N_{Σ_k,μ_k}, jadi $\Pr(x|M_k) = \Pr(x-\mu_k|\overline{M}_k)$. *Query* atau vektor baru yang akan

dikenali pada saat klasifikasi dinotasikan dengan q, dengan menggunakan probabilitas campuran $\Pr(M_k)$ kemudian dapat dihitung *posteriori probabilitas component* $\Pr(M_k|x)$. Ini dapat dibayangkan sebagai indikasi stokastik dari keanggotaan x dalam setiap komponen mixture.

Setiap komponen mixture dibentuk oleh sebuah klas pola dengan beberapa parameter yang membedakannya dengan yang lain, sehingga dapat dibayangkan bahwa klas-klas tersebut dipisahkan oleh sebuah garis batas (*decision boundary*) yang menentukan keanggotaan suatu vektor dalam suatu klas pola atau komponen mixture. Dalam kasus dimana komponen mixture hanya satu (n=1), *mixture-distance* tereduksi menjadi *Mahalanobis Distance*.



Gambar 3.5 Decision Boundary

3.3.1 Seleksi antara Fisrt Order dan Second Order Statisitik Model

Penggunaan metode mixture-distance menghadirkan masalah pemilihan model antara *first order* dan *second order statistik model* untuk setiap komponen

Gaussian (*Mixture*) [5]. *First order Gaussian model* M_1 mempunyai diagonal kovarian matriks yang berisi estimasi varian dari individu feature dan sebuah vektor mean berisi estimasi ekspektasi dari distribusi, *first order statistik model* ini biasa disebut dengan *variance model*. Sedangkan *second order mode*l M_2 tepat sama dengan *first order* hanya saja entri dari *off-diagonal* kovarian matriksnya tidak bernilai nol dan merepresentasikan estimasi feature second order statistik (*covariance*), second order statistik model biasa disebut dengan *covariance model*.

Sebuah mixture dapat dinyatakan dengan $M=(1-f)\cdot M_1 + f\cdot M_2$, dimana $f \in [0,1]$. Mixture ini secara generatif dibentuk dengan memilih M_1 dengan probabilitas 1-f dan M_2 dengan probabilitas f, dan menggambarkan sebuah pemilihan model. Kovarian matrik hasil dari data vektor dilihat sebagai $\Sigma_1 + f\cdot(\Sigma_1 - \Sigma_2)$, dapat diartikan sebagai Σ_2 dengan elemen *off-diagonal*nya dikalikan dengan f, sesuai dengan teori *heuristik* yang menyatakan "*elemen-elemen off-diagonal dari sample kovarian matrik dikalikan dengan f untuk membentuk sebuah single Gaussian model yang pendekatannya dipilih antara first order dan second order statistik*" [5].

Dalam implementasinya f adalah *scalling faktor* atau *off-diagonal weighting* yaitu sebuah faktor pengali yang merepresentasikan besarnya pengaruh dari kovarian dalam pembentukan kovarian matrik. Faktor pengali tersebut menggambarkan pilihan model yang ada yaitu $f = 0, f = \frac{1}{2}$, dan $f \approx 1$. *First order statistik model* didapat dengan menggunakan faktor pengali yang pertama $f = 0$, dimana faktor pengali ini mengalikan nilai elemen *off-diagonal* pada

kovarian matrik dengan 0 sehingga hanya diagonal matrik saja yang berpengaruh atau dengan kata lain kovarian matrik ini hanya berisi estimasi varian dari feature individu dan mengabaikan nilai kovarian. *Second order statistik model* didapat dengan menggunakan $f \approx 1$ yang membentuk *full covariance matrices*. Untuk perbandingan dapat dipilih sebuah faktor pengali $f = \frac{1}{2}$ untuk memutuskan antara first order (*diagonal variance*) dan second order model (*full covariance matrices*).

Faktor pengali diatas sangat mempengaruhi performance pengenalan wajah, secara teori *scalling faktor* diatas nilainya dapat dirubah-rubah antara 0 dan 1 untuk memperoleh performance terbaik. Setiap nilai dari faktor pengali menghasilkan performance pengenalan yang berbeda, untuk penyederhanaan dalam sistem pengenalan wajah ini dipakai tiga *scalling* faktor yaitu $f = 0, f = 0.5$ dan $f \approx 1$.

3.3.2 Fungsi Probabilitas

Dalam menentukan kedekatan suatu vektor query dengan klas yang ada digunakan sebuah fungsi probabilitas (*probabilitas function*). Dengan menggunakan algoritma *expectation maximization*, estimasi nilai probabilitas dari vektor query terhadap suatu klas dapat dihitung dan kemudian mencari nilai fungsi *likelihood* yang maksimal.

Tahap pertama adalah melakukan estimasi terhadap nilai fungsi probabilitas dari suatu vektor query terhadap klas tertentu yang dinyatakan dengan persamaan sebagai berikut :

$$p(x(t) \mid \omega_i) = \frac{1}{(2\pi)^{D/2} \Pi_j^D \sigma_{j\omega_i}} \exp\left(-\frac{1}{2} \sum_{j=0}^{D-1} \frac{(x_j(t) - m_{j\omega_i})^2}{\sigma_{j\omega_i}^2} \right) \quad (3.9)$$

Tahap kedua adalah mendapatkan nilai *likelihood* dari setiap klas data dengan vektor *query*. Dari penurunan persamaan (3.9) didapat fungsi *likelihood* dari klas data ke i terhadap vektor *query*, dijelaskan dengan penurunan rumus berikut ini :

$$\phi(x(t) \mid \omega_i) = \ln(\, p(x(t) \mid \omega_i)) \quad (3.10)$$

$$= \ln\left(\frac{1}{(2\pi)^{D/2} \prod_j^D \sigma_{j\omega_i}} \exp\left(-\frac{1}{2} \sum_{j=0}^{D-1} \frac{(x_j(t) - m_{j\omega_i})^2}{\sigma_{j\omega_i}^2} \right) \right)$$

$$= \ln\left((2\pi)^{-D/2} \prod_j^D \sigma_{j\omega_i}{}^{-1} \exp\left(-\frac{1}{2} \sum_{j=0}^{D-1} \frac{(x_j(t) - m_{j\omega_i})^2}{\sigma_{j\omega_i}^2} \right) \right)$$

$$= \ln(2\pi)^{-D/2} + \ln \prod_j^D \sigma_{j\omega_i}{}^{-1} + \ln \exp\left(-\frac{1}{2} \sum_{j=0}^{D-1} \frac{(x_j(t) - m_{j\omega_i})^2}{\sigma_{j\omega_i}^2} \right)$$

$$= -\frac{D}{2} \ln 2\pi - \prod_{j=0}^{D-1} \ln \sigma_{j\omega_i} - \frac{1}{2} \sum_{j=0}^{D-1} \frac{(x_j - m_{j\omega_i})^2}{\sigma_{j\omega_i}^2} \cdot \ln e$$

$$= -\frac{D}{2} \ln 2\pi - \prod_{j=0}^{D-1} \ln \sigma_{j\omega_i} - \frac{1}{2} \sum_{j=0}^{D-1} \frac{(x_j - m_{j\omega_i})^2}{\sigma_{j\omega_i}^2} \quad (3.11)$$

dengan

x : vektor *query*

ω_i : klas data ke i

$m_{j\omega_i}$: *mean* dari parameter ke j pada klas data ke i

$\sigma_{j\omega_i}^2$: *variance* dari parameter ke j pada data klas ke i

D : dimensi dari vektor *query*

Dari nilai *likelihood* yang didapat dari setiap klas pola, dapat menunjukan kedekatan suatu vektor dengan klas pola tersebut. Nilai *likelihood* yang terbesar menunjukan klas yang paling dekat dengan vektor tersebut.

3.3.3 Vector Quantization

Untuk melakukan klasifikasi terhadap suatu vektor yang akan dikenali dapat digunakan dua metode yaitu metode *hard vector quantization* dan *soft vector quantization* [5,7].

3.3.3.1 Hard Vector Quantization

Dalam *hard vector quantization* (*hard VQ*), setiap elemen data training menunjuk pada satu Gaussian. Sebuah elemen data training Y_i akan menunjuk pada sebuah Gaussian G_l dimana probabiltas Y_i pada G_l maksimal, dinyatakan dengan $l = \arg\max_j P(G_j \mid Y_i)$.

Kemudian untuk menentukan jarak antara vektor *query* Q dengan setiap elemen data training dihitung dengan menggunakan *Mahalanobis distance* sebagai berikut :

$$distance\ (Q, Y_i) = (Q - Y_i)^T \Sigma_l^{-1} (Q - Y_i) \tag{3.12}$$

dimana Σ_l adalah kovarian matrik dari Gaussian yang ditunjuk Y_i. Vektor Q akan dikenali sebagai titik Y_k, dimana $k = \text{argmin}_i \text{distance}(Q, Y_i)$.

3.3.3.2 Soft Vector Quantization

Soft vector quantization (*soft VQ*), mengganti penunjukan setiap elemen data training terhadap satu Gaussian dengan mempertimbangkan bahwa setiap titik dapat dimiliki oleh beberapa Gaussian secara bersamaan dan dapat dijelaskan oleh properti Gaussian tersebut.

Sebuah *query* Q diklasifikasikan dengan titik Y_l dimana $l = \arg\max_i P(Y_i \mid Q)$, dimana :

$$P(Y_i \mid Q) = \frac{P(Q \mid Y_i)P(Y_i)}{P(Q)}$$

$$= \frac{P(Q \mid Y_i)P(Y_i)}{\sum_{k=1}^{N} P(Q \mid Y_k)P(Y_k)} \qquad (3.13)$$

dimana

$$P(Q \mid Y_i) = \sum_{j=1}^{M} P(Q \mid G_j, Y_i)P(G_j \mid Y_i) \qquad (3.14)$$

$P(G_j \mid Y_i)$ adalah posterior probability dan $P(Q \mid G_j, Y_i)$ dihitung dengan mengikuti persamaan :

$$P(Q \mid G_j, Y_i) = \frac{1}{(2\pi)^{\frac{d}{2}} \mid \Sigma_j \mid^{\frac{1}{2}}} \exp -\frac{1}{2}(Q - Y_i)^T \Sigma_j^{-1}(Q - Y_i) \qquad (3.15)$$

dimana,

G_j = Gaussian ke j

Y_i = elemen database ke i

Σ_j = kovarian matrik ke j

d = dimensi vektor *query*

BAB IV

PENGENALAN WAJAH DENGAN MIXTURE DISTANCE

BAB IV

PENGENALAN WAJAH

DENGAN METODE MIXTURE DISTANCE

4.1 Pendahuluan

Pada bab ini akan dibahas secara lengkap mengenai tahap-tahapan yang akan dilakukan dalam sistem pengenalan wajah dengan metode mixture-distance. Dalam sistem ini terdapat tiga proses utama yaitu proses ekstraksi, proses pelatihan atau training dan terakhir adalah proses pengenalan.

Seacara umum struktur sistem pengenalan wajah sangat luas tetapi ada tiga komponen utama yang harus diperhatikan dalam membangun suatu sistem pengenalan wajah, yaitu mendefinisikan himpunan feature (*feature set*), ekstraksi feature dari sebuah citra dan algoritma pengenalan wajah. Mengacu kepada struktur diatas, secara garis besar tahapan proses yang dilakukan dalam pembuatan sistem pengenalan wajah ini meliputi ekstraksi feature, pemodelan data training dan proses pengenalan (*recognition*).

Ekstraksi feature meliputi penentuan informasi-informasi dari suatu wajah yang akan dijadikan feature, kemudian mengekstraknya mejadi suatu vektor feature yang dapat mewakili setiap wajah, dilanjutkan dengan memodelkan data training dan terakhir melakukan pengenalan terhadap suatu pola yang akan dikenali.

Pengenalan wajah seseorang pada dasarnya adalah mengenali informasi-informasi yang ada pada wajah itu sendiri. Wajah merupakan bentuk informasi

yang cukup komplek karena banyaknya informasi yang terdapat didalamnya, informasi tersebut meliputi bentuk wajah, panjang wajah,warna kulit wajah dan berbagai informasi lainnya. Untuk itu perlu ada metode untuk mendapatkan informasi-informasi yang cukup signifikan dari sejumlah informasi yang ada sehingga informasi yang didapat cukup mewakili dan bersifat unik untuk setiap orang.

Selain kekomplekan informasi yang terdapat pada suatu wajah, perubahan-perubahan yang terjadi pada wajah juga harus diperhatikan karena dapat menghasilkan suatu distorsi pada pola wajah tersebut. Seringkali klasifikasi, pengenalan, deskripsi pola mengalami perubahan atau deviasi dari pola ideal sehingga menghasilkan suatu variasi bagi objek tertentu, umumnya variasi yang terjadi pada citra diakibatkan distorsi pada saat pengambilan gambar objek tersebut. Saat pengambilan gambar ekspresi seseorang dapat berubah-ubah seperti tertawa, sedih, marah ataupun posisi wajah yang berubah seperti menunduk, menengadah, menoleh ataupun miring kekiri atau kekanan.

Selain perubahan ekspresi wajah, beberapa faktor lain juga dapat mempengaruhi gambar suatu objek antara lain tansformasi gambar dari bentuk wajah tiga dimensi ke dua dimensi yang dapat menyebabkan hilangnya sejumlah informasi, perbedaan pencahayaan (*illumination*) saat pengambilan gambar, jarak pengambilan gambar suatu objek akan menghasilkan suatu variasi yang berbeda pada objek tersebut.

Sejak dimulainya riset pengenalan wajah sekitar tahun 1960, secara garis besar pendekatan pengenalan wajah dapat dibagi dalam dua pendekatan yaitu

pendekatan berdasarkan ciri-ciri/feature (*feature based recognition*) yang mengambil suatu himpunan feature (*feature set*) yang lebih kecil dibanding dengan jumlah pixel dari citra wajah, dan pendekatan kedua yaitu *direct image method* yang tidak melibatkan tahap ekstraksi feature. Sebenarnya direct image method juga melakukan ekstraksi feature bedanya feature-featurenya terpengaruh secara signifikan oleh variasi pencahayaan (*illumination*) sedangkan pendekatan dengan berdasarkan ciri-ciri (*feature based*) secara umum tidak terpengaruh oleh kondisi pencahayaan.

Template matching dan penelitian yang dilakukan oleh Turk dan Pentland dalam "*eigenfaces*" termasuk yang menerapkan metode direct image. Template matching hanya efektif jika query dan image model mempunyai skala, orientasi,dan illuminasi (pencahayaan) yang sama. Sistem pengenalan wajah dengan menggunakan metode *mixture distance* ini termasuk kepada pendekatan pertama yaitu pengenalan wajah berdasarkan ciri-ciri (*feature based face recognition*).

Pada dasarnya semua sistem pengenalan wajah harus dapat mengatasi semua faktor yang menyebabkan noise ataupun distorsi terjadi, namun tidak ada satupun pendekatan yang dapat mengatasi semua faktor tersebut secara sempurna, yang dapat dilakukan adalah meminimalisasi pengaruh dari faktor-faktor tersebut, satu pendekatan mungkin dapat menangani beberapa faktor tetapi tidak untuk faktor lainnya, untuk itu banyak pendekatan dilakukan untuk mendapatkan hasil yang optimal, termasuk pendekatan yang akan digunakan dalam tugas akhir ini yang menggunakan pendekatan statistik.

4.2 Ekstraksi Feature

Proses ekstraksi pada pengenalan pola prinsipnya merupakan proses penentuan dan pengambilan informasi yang terdapat pada objek tertentu yang dianggap cukup unik dan dapat merepresentasikan objek tersebut serta membedakannya dengan objek yang lain. Hasil dari ekstraksi ini dapat direpresentasikan dalam berbagai bentuk, misalnya vektor, matriks atau bentuk lainnya. Informasi yang telah diekstrak dan direpresentasikan dalam bentuk tertentu itulah yang akan digunakan dalam proses pengenalan selanjutnya.

4.2.1 Pengertian Feature

Feature adalah pengukuran/informasi yang dapat diekstraks dari suatu obyek. Feature dapat berupa simbolik, misalnya warna atau numerik seperti ukuran tinggi, lebar dari suatu obyek, feature juga dapat berupa kombinasi keduanya simbolik dan numerik. Feature dihasilkan dari pengolahan data input oleh algoritma atau operator ekstraksi feature (*feature ekstraktor*). Feature dapat berupa data kontinyu, diskrit, ataupun diskrit biner. Ekstraksi dan pemilihan feature sangat bergantung pada masalah yang dihadapi dan pendekatan yang dipakai.

4.2.2 Vektor Feature Wajah (*facial feature vector*)

Dalam sistem pengenalan wajah ini, feature yang dimaksud adalah ciri-ciri fisik pada wajah yang unik dan dapat digunakan untuk proses identifikasi, dalam hal ini antara lain lebar mata, panjang dan tebal alis, jarak antar mata (*inter iris*)

dan ciri-ciri fisik lainnya yang cukup signifikan. Penentuan feature harus cermat dan teliti sehingga feature yang dipilih benar-benar dapat merepresentasikan wajah individu tertentu.

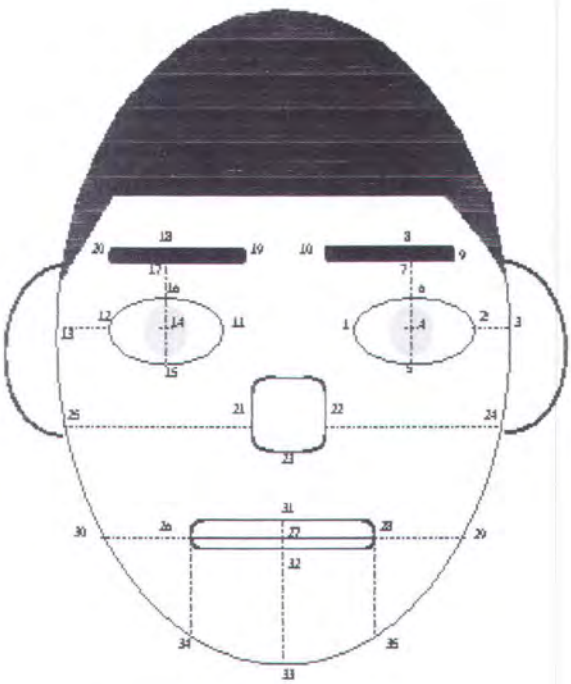
Feature dapat tersusun dalam bentuk sebuah vektor n-dimensi dinotasikan dengan x, dalam sistem pengenalan wajah ini, setiap wajah direpresentasikan kedalam suatu *vektor feature* 30 dimensi yang didapat dari 35 titik pengukuran (*points measurement*), gambar 4.1 memperlihatkan secara detail 35 titik penngukuran yang dimaksud. Penentuan titik-titik pengukuran tersebut berpedoman pada penelitian yang dilakukan oleh S.Sakamoto dan J.Tojima dalam *face feature analysis for human identification* [8], yang menghasilkan sebuah kesimpulan mengenai feature-feature yang cukup unik dan secara umum tidak banyak berubah dari suatu citra wajah dalam berbagai ekspresi.

Dalam sistem pengenalan wajah ini proses ekstraksi feature dilakukan secara manual, dengan menentukan 35 titik pengukuran menggunakan pointer mouse seperti dalam gambar 4.2. Setelah titik-titik tersebut ditentukan kemudian dilakukan komputasi untuk mendapatkan distancenya, sehinga menghasilkan 30 distance yang akan disimpan dalam sebuah vektor feature 30-dimensi seperti dalam tabel 4.1.

Untuk menghilangkan pengaruh ukuran citra maka semua distance tersebut dinormalisasi dengan jarak antar mata (*inter iris distance*) untuk mendapatkan kesamaan invariance (*similarity invariance*), maksud dari normaliasai disini adalah membagi semua distance dengan jarak antar mata sehingga didapat suatu nilai yang merepresentasikan perbandingan antara setiap

distance terhadap jarak antar mata, nilai inilah yang disimpan dalam vektor feature. Vektor yang telah ternormalisasi inilah yang merupakan hasil dari proses ekstraksi secara manual. Proses pengenalan (*recognition*) maupun proses pelatihan (*training*) harus melewati proses ekstraksi ini untuk mendapatkan vektor featurenya.

Secara lengkap titik-titik pengukuran yang digunakan dalam proses ekstraksi ini dijelaskan dalam gambar dibawah ini :



Gambar 4.1 Titik-titik pengukuran

Dari gambar diatas didapat 35 titik pengukuran yang kemudian dihitung distance antar titik tertentu dengan mengikuti aturan dalam *points measurement system*, maka akan didapatkan 30 distance yang akan disimpan dalam sebuah vektor 30 dimensi, secara lengkap penghitungan distance dari titik-titik pengukuran dapat dijelaskan oleh tabel 4.1 berikut ini :

Feature	Distance	Feature	Distance
1	0.5 * ((1,2) + (11,12))	16	0.5 * ((25,30) + (24,29))
2	0.5 * ((5,6) + (15,16))	17	0.5 * ((30,34) + (29,35))
3	(3,13)	18	0.5 * ((1,22) + (11,21))
4	(24,25)	19	(10,19)
5	(29,30)	20	0.5 * ((2,9) + (12,20))
6	(34,35)	21	0.5 * ((9,10) + (19,20))
7	(26,34)	22	0.5 * ((11,19) + (1,10))
8	(28,35)	23	0.5 * ((6,7) + (16,17))
9	(26,28)	24	0.5 * ((7,8) + (17,18))
10	(27,31)	25	0.5 * ((18,19) + (8,10))
11	(27,32)	26	0.5 * ((18,20) + (8,9))
12	(32,33)	27	(11,23)
13	(23,31)	28	(1,23)
14	(21,22)	29	0.5 * ((1,28) + (11,26))
15	0.5 * ((13,25) + (3,24))	30	0.5 * ((12,13) + (2,3))

Tabel 4.1 vektor feature 30 dimensi

4.3 Tahap Pelatihan

Tahap selanjutnya adalah proses pelatihan untuk membentuk sebuah database feature vektor dan memodelkan data training. Dalam proses training esktraksi feature dilakukan berkali-kali tergantung data training dari setiap orang yang tersedia untuk mendapatkan vektor mean yang akan disimpan dalam sebuah database *facial feature vektor*.

4.3.1 Pemodelan Data Training

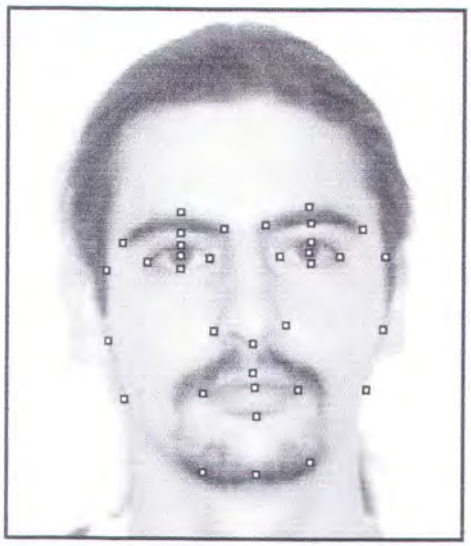
Dalam database feature vektor setiap elemen database berhubungan dengan satu orang tertentu, jadi setiap orang mempunyai satu wakil didalam database. Proses utama dalam tahap pelatihan ini adalah memodelkan data

training kedalam sebuah bentuk model statistik, data training disini adalah kumpulan vektor-vektor feature didalam database. Dari data training yang ada dalam database dikelompokan kedalam klas-klas pola menurut kedekatannya, *k-means clustering* dapat digunakan dalam proses pengelompokan ini. Setiap *cluster* dari hasil pengelompokan itu mempunyai parameter-parameter *cluster* yang meliputi *mean* sebagai pusat *cluster*, *variance*, dan *covariance matrix*, inisialisasi dilakukan untuk mendapatkan nilai parameter-parameter tersebut, sehingga data training terbentuk menjadi beberapa klas pola dengan nilai parameter masing-masing.

Data training dimodelkan sebagai *mixture of normal density* atau biasa disebut dengan *mixture density*, yang secara umum dapat dinyatakan dengan persamaan (3.8). *Mixture density* dibentuk dari klas-klas pola yang ada dimana data training dari setiap klas dimodelkan sebagai sebuah normal density $N_{(\Sigma,\mu)}$, dengan kovarian matrik Σ dan ekspektasi μ.

4.4 Tahap Pengenalan

Proses pengenalan wajah dilakukan terhadap suatu vektor tak dikenal yang akan diidentifikasi. Vektor atau pola tak dikenal tersebut dinamakan dengan *query q*, *query* ini merupakan vektor feature 30 dimensi yang didapat dari hasil ekstraksi terhadap suatu input citra wajah yang akan dikenali, seperti dalam gambar 4.2.



Gambar 4.2 35 titik pengukuran pada sebuah citra wajah

Setiap vektor fetaure yang akan dikenali melewati dua tahapan proses, proses pertama adalah melakukan klasifikasi terhadap vektor atau pola tersebut dengan setiap klas pola yang ada untuk mendapatkan probabilitas terbesar atau dengan kata lain untuk mendapatkan $\Pr(q \mid \omega_i)$ yang maksimal. Klas dengan probabilitas maksimal dapat dianggap sebagai klas pola yang paling dekat dengan vektor query tersebut. Dengan menggunakan pendekatan hard VQ [5], dimana $\sum_{k=1}^{n} \Pr(M_k) = 1$ dan memberi nilai 1 untuk klas dimana fungsi probabilitas maksimal dan yang lain 0, maka klas dengan dengan nilai 0 dapat diabaikan sehingga kita hanya bekerja dengan klas dimana fungsi probabilitas maksimal.

Tahap kedua merupakan proses penghitungan fungsi distance dari *query* terhadap setiap elemen klas. Setiap elemen klas merupakan suatu vektor feature dari orang yang berbeda, *distance* didapat dengan menggunakan persamaan (3.12). Vektor yang dikenali didapat dari nilai distance (Q, Y_i) yang paling kecil.

Sehingga secara umum dapat dikatakan bahwa hasil dari proses pengenalan adalah sebuah vektor yang mempunyai nilai probabilitas maksimum dan nilai distance minimum.

BAB V

PERANCANGAN DAN PEMBUATAN PERANGKAT LUNAK

BAB V

PERANCANGAN DAN PEMBUATAN PERANGKAT LUNAK

Bab ini akan menguraikan mengenai proses perancangan dan pembuatan perangkat lunak yang telah dibuat dalam tugas akhir ini. Perangkat lunak yang dibuat dalam tugas akhir ini merupakan implementasi dari metode pengenalan wajah dengan menggunakan mixture-distance yang telah diuraikan pada bab-bab sebelumnya.

Pembahasan mengenai perancangan perangkat lunak ini meliputi tujuan dan sasaran sistem, deskripsi dari sistem, rancangan form dan data, algoritma yang digunakan, modul-modul penting yang dipakai dan diagram alir dari sistem yang dibuat.

Perangkat lunak ini dibuat dengan menggunakan bahasa pemograman Borland Delphi yang berbasis pada sistem operasi Windows 9X. Penggunaan bahasa pemrograman tersebut dikarenakan Delphi merupakan bahasa pemrograman visual yang didukung oleh berbagai macam fasilitas yang cukup lengkap, sehingga dapat mempercepat pembuatan perangkat lunak ini, Delphi juga merupakan bahasa pemrograman yang mendukung perograman berorientasi object (OOP). Digunakannya sistem operasi windows dengan pertimbangan fasilitas dan kemudahannya dalam akses data berupa citra dan grafik, kemampuannya dalam melakukan *multitasking*, juga dikarenakan sistem operasi window merupakan sistem operasi yang umum digunakan pada komputer.

5.1 Tujuan dan Sasaran Sistem

Pembuatan sistem pengenalan wajah dengan metode mixture-distance ini adalah untuk mengimplentasikan metode mixture-distance dalam algoritma pengenalan wajah. Sasaran yang hendak dicapai adalah untuk menguji dan mengetahui sejauh mana performance metode mixture-distance jika diterapkan dalam perangkat lunak pengenalan wajah.

5.2 Deskripsi Sistem

Sistem pengenalan wajah ini terbagi dalam dua proses yang utama yaitu proses pengenalan wajah dan proses pelatihan (*training*) yang harus dilakukan dahulu sebelum proses pengenalan wajah dilakukan. Kedua proses tersebut dipisahkan dalam dua sub sistem yang berbeda, hal itu berdasarkan pada kondisi dimana proses training harus dilakukan sebelum proses pengenalan dilakukan. Proses training tidak selalu dilakukan seperti halnya proses pengenalan wajah, kedua proses ini tidak bisa berjalan pada saat yang bersamaan walaupun input yang diberikan sama.

Proses pengenalan wajah digunakan untuk mengenali citra wajah yang dimasukan ke sistem ini, sedangkan proses training dilakukan untuk membentuk database yang menyimpan informasi-informasi dari citra wajah yang dimasukan dan melakukan pemodelan pada data training yang akan digunakan selanjutnya dalam proses pengenalan wajah sesuai dengan algoritma yang dipakai yaitu *mixture-distance*.

Dalam perancangan sistem ini akan digunakan dua buah tabel, yaitu tabel untuk menyimpan citra wajah asli dan tabel untuk menyimpan data-data yang merupakan representasi dari citra wajah dalam bentuk vektor feature 30 dimensi hasil dari proses ekstraksi.

Secara global tahapan proses sistem pengenalan wajah ini diawali dengan input berupa citra wajah yang kemudian diekstraks melalui proses ekstraksi manual oleh *human operator,* hasil ekstraksi berupa sebuah vektor feature 30 dimensi yang kemudian dapat dilanjutkan untuk proses training atau recognition, untuk proses training, ekstraksi dilakukan berkali-kali sesuai dengan data training yang tersedia, kemudian dilakukan komputasi untuk menghitung *vektor mean* yang akan mewakili data training tersebut, untuk kemudian disimpan dalam database. Sedangkan untuk recognition, data hasil ekstraksi dianggap sebagi pola tak dikenal yang akan diidentifikasi, maka akan dilakukan proses komputasi untuk menghitung nilai kedekatan pola tersebut dengan setiap data training yang ada di database, output dipilih dari hasil jarak yang minimal dan nilai kemiripan (*similarity*) yang terbesar.

5.3 Perancangan Perangkat Lunak

Sub Bab ini akan membahas mengenai perancangan lunak sebelum diimplemtasikan kedalam sebuah program. Perancangan perangkat lunak ini meliputi perancangan data, perancangan proses, perancangan database dan diagram aliran data.

5.3.1 Perancangan Data

Dalam perancangan dan pembuatan setiap program selalu melibatkan dua hal yang utama yaitu proses dan data. Setiap proses selalu diawali dengan suatu data masukan (*input*) yang akan diproses kemudian akan dihasilkan suatu data keluaran (*output*). Secara garis besar perancangan data dalam tugas akhir ini dibagi menjadi tiga, yaitu data masukan, data proses, dan data keluaran.

5.3.1.1 Data Masukan

Data masukan atau input yang digunakan dalam perangkat lunak ini berupa file citra dengan format bitmap (bmp), input citra disini harus berupa citra wajah, bisa berwarna ataupun *graylevel*.

5.3.1.2 Data Proses

Data proses adalah data-data yang digunakan selama proses berlangsung, pada saat perangkat lunak dijalankan data-data yang digunakan dan dihasilkan adalah sebagai berikut :

- APoint adalah data didapat pada saat proses ekstraksi, data ini direpresentasikan dalam sebuah array satu dimensi dengan elemen TPoint yang menyimpan koordinat titik-titik pengukuran (*Points Measurement*).
- VektorFeature adalah data yang didapat dari hasil ekstraksi pada sebuah citra wajah, data ini disimpan dalam sebuah array satu dimensi dengan elemen real yang menyimpan distance dari titik-titik pengukuran.

- NormalVector adalah data yang didapat dari hasil training yang direpresentasikan dalam array satu dimensi dengan elemen real, data ini menyimpan nilai rata-rata dari vektor feature yang telah dinormalisasi dari setiap data training.
- Cluster menyimpan data-data dari suatu kelompok data tertentu. Data ini merupakan record yang menyimpan informasi-informasi mengenai suatu kelompok data (*class*).
- ArrCluster adalah data yang didapat pada saat proses pengelompokan data (*clustering*), data ini merupakan array satu dimensi dengan elemen array berupa record cluster menampung seluruh kelompok data/klas yang ada.
- VarianceVector merupakan data yang menyimpan informasi nilai varian dari suatu kelompok data, data ini berupa array satu dimensi dengan elemen real.
- MeanVector merupakan data yang menyimpan nilai rata-rata dari suatu kelompok data, data ini berupa array satu dimensi dengan elemen real.
- KovarianceMatrix merupakan data kovarian matriks dari setiap kelompok data tertentu. Data ini berbentuk matriks yang direpresentasikan kedalam array dua dimensi dengan elemen array bertype real.

5.3.1.3 Data Keluaran

Perangkat lunak yang dibuat dalam tugas akhir ini bertujuan untuk melakukan pengenalan (*recognition*) terhadap suatu citra wajah berdasarkan database citra yang telah ada, jadi output yang dihasilkan berupa citra wajah, nilai

kemiripan (*likelihood value*) dan nilai distancenya, output citra diambil dari database yang mempunyai nilai distance terkecil dan nilai kemiripan terbesar.

5.3.2 Perancangan Proses

Dalam perancangan perangkat lunak ini perancangan proses dibagi kedalam tiga tahap yang utama. Tahapan-tahapan proses tersebut yaitu tahapan awal berupa proses ekstraksi, dimana pada proses ekstraksi pertama kali akan menentukan titik-titik pengukuran, kemudian menghitung distance dari titik pengukuran tersebut dan terakhir dilakukan normalisasi vektor feature. Tahapan berikutnya adalah proses training, pertamakali akan dihitung vektor mean yang akan mewakili data training tersebut kemudian dilakukan penyimpanan data dan kemuadian data tersebut akan dikelompokan menjadi beberapa kelompko/class berdasarkan kemiripannya. Tahap terakhir adalah proses pengenalan, yang dilakukan disini adalah melakukan penghitungan distance antara query dengan setiap elemen database.

5.3.2.1 Proses Ekstraksi

Proses ekstraksi merupakan proses pertama yang akan dilakukan dalam perangkat lunak ini. Tujuan dari proses ekstraksi ini adalah untuk mendapatkan data-data yang penting yang akan digunakan untuk proses selanjutnya, dalam proses ekstraksi disini hasilnya berupa vektor feature dengan 30 dimensi.

Untuk mendapatkan vektor feature tersebut pertama kali kita harus menentukan titik-titk pengukuran (*points measurement*) dengan secara manual

menggunakan suatu pointer dalam hal ini mouse, dalam pembuatan perangkat lunak ini penentuan titik-titik pengukuran berpedoman pada penelitian yang dilakukan S.Sakamoto dan J.Tojima, yang melibatkan 35 titik pengukuran. Proses selanjutnya adalah melakukan komputasi untuk menghitung semua distance seperti pada tabel 4.1, sehingga didapat 30 buah distance yang akan disimpan dalam sebuah vektor feature. Langkah terakhir yang harus dilakukan adalah normalisasi, yaitu semua distance-distance yang telah didapat harus dinormalisasi tehadap jarak antar titik tengah kedua mata (*inter iris*), maksud normalisasi disini semua nilai distance dibagi dengan jarak antar titik tengah kedua mata, sehingga hasilnya merupakan perbandingan setiap distance terhadap jarak inter iris tersebut, hal ini dilakukan agar input citra secara teori tidak dibatasi ukurannya karena yang akan disimpan dalam database adalah skalanya bukan nilai aslinya.

Setelah melakukan proses normalisai nilai-nilai distance tersebut dimasukan kedalam sebuah array satu dimensi yaitu vektor fetaure, vektor feature inilah hasil dari proses ekstraksi yang kemudian akan digunakan dalam proses-proses selanjutnya.

5.3.2.2 Proses Training

Proses training ini dilakukan untuk mendapatkan data-data dari setiap orang yang akan disimpan dalam sebuah database. Pada tahap training ada dua sub proses yang akan dilakukan yaitu pembentukan data training dalam database dan proses pengelompokan data (*clustering*). Hasil dari proses training ini adalah suatu pemodelan data dari database facial feature vektor.

Pembentukan database diawali dengan mencari normal vektor dari data training yang tersedia. Pada tahap ini proses ekstraksi akan dilakukan berulang-ulang tergantung data pelatihan yang tersedia, setiap hasil ekstraksi ditampung dulu untuk kemudian dihitung vektor meannya yang akan disimpan dalam database.

Proses pengelompokan data dilakukan setelah database terbentuk, vektor-vektor dalam database dikelompokan berdasarkan kedekatannya dengan menggunakan *K-Means Clustering*, kemudian akan membentuk beberapa klas pola, data training dalam setiap klas dimodelkan sebagai *normal density* $N_{(\Sigma,\mu)}$, dengan covariance matriks Σ dan ekspectasi μ.

Setelah melakukan proses training maka database feature vektor akan membentuk beberapa kelompok yang telah dimodelkan sebagai normal density. Jadi hasil dari proses training adalah pembentukan database facial feature vektor dan pemodelan dari seluruh data training yang ada dalam database.

5.3.2.3 Proses Recognition

Proses terakhir dalam perangkat lunak ini adalah melakukan pengenalan jika ada suatu pola tak dikenal yang akan diidentifikasi. Pola tak dikenal dinotasikan dengan *Query* q, untuk mengenali pola ini maka dilakukan dua tahapan proses.

Tahapan proses yang pertama dilakukan untuk mengklasifikasikan pola q pada klas pola yang ada sesuai kedekatannya, dengan mencari nilai *likelihood* yang terbesar menggunakan persamaan (3.11).

kemudian tahapan proses yang kedua adalah mencari distance yang paling kecil antara *query q* dengan setiap elemen dalam klas pola tersebut dengan menggunakan persamaan (3.12). Jadi hasil dari proses *recognition* berupa vektor yang mempunyai probabilitas terbesar dan *distance* terkecil.

5.3.3 Perancangan Database

Dalam sistem pengenalan wajah ini database diperlukan untuk menyimpan data-data hasil proses trainning yang akan digunakan dalam proses pengenalan, data-data itu meliputi citra wajah asli dan informasi-informasi yang didapat dari citra wajah setelah dilakukan ekstraksi dalam hal ini adalah vektor feature.

Sistem pengenalan wajah disini melibatkan dua buah tabel, tabel pertama digunakan untuk menyimpan citra wajah, sedangkan tabel kedua untuk menyimpan feature vektor 30 dimensi. Kedua buah table ini direlasikan dengan sebuah field kunci, yaitu field kode. Database dibuat dengan menggunakan database dekstop dan bertype Paradox 7.

Berikut struktur dari tabel yang dibuat :

Kode	Gambar
101	
102	

Tabel 5.1 Tabel Citra Wajah

Keterangan :

Kode = Field kunci sebagai identitas record

Gambar = Field yang menampung citra wajah

Feature = Field yang menampung nilai setiap feature wajah

Kode	Feature1	Feature2	.	.	.	.	.	Feature30
101								
102								

Tabel 5.2 Tabel Facial feature vector

Type data dari tabel yang dibuat :

Nama Field	Type
Kode	Short Integer
Gambar	Graphics

Tabel 5.3 Tabel type data field dari tabel citra wajah

Nama Field	Type
Kode	Short Integer
Feature1	Number
Feature2	Number
Feature .. .	Number
Feature29	Number
Feature30	Number

Tabel 5.4 Tabel type data dari tabel facial feature vector

Kedua tabel itu dihubungkan oleh field kunci yaitu field kode, setiap record pada tabel citra mempunyai sebuah record pada tabel face feature, jadi pada saat proses penyimpanan, satu citra wajah melakukan penyimpanan pada dua tabel tersebut, citra wajah pada tabel pertama sedang vektor feature pada tabel kedua.

5.3.4 Diagram Aliran Data

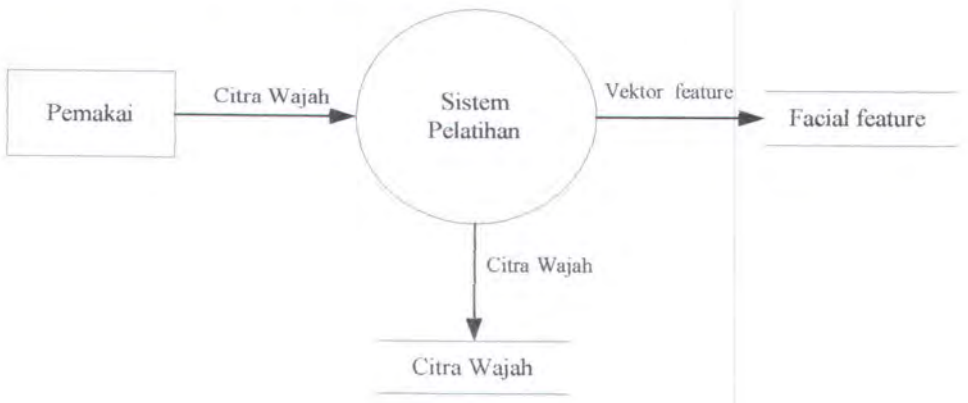
Bagian ini akan membahas perancangan sistem pengenalan wajah yang akan direpresentasi dalam gambar, yaitu menggunakan Diagram Aliran Data

(DAD) atau *Data Flow Diagram (DFD)*. Dengan menggunakan diagram aliran data, dapat memperjelas semua proses dan aliran data yang terjadi dalam sistem pengenalan wajah ini, dari diagram aliran data juga dapat diketahui data yang diolah dan dihasilkan oleh suatu proses, dengan demikian aliran data mulai awal proses sampai akhir proses dapat diketahui dengan jelas.

Seperti yang telah dijelaskan pada bagian deskripsi sistem, perangkat lunak pengenalan wajah dengan *mixture distance* ini terbagi menjadi dua sub sistem yaitu subsistem pengenalan wajah dan sub sistem pelatihan, oleh karena itu penjelasan dengan data flow daigram ini akan dibagi menjadi dua bagian pula yaitu dfd untuk subsistem pelatihan dan susbsistem pengenalan wajah.

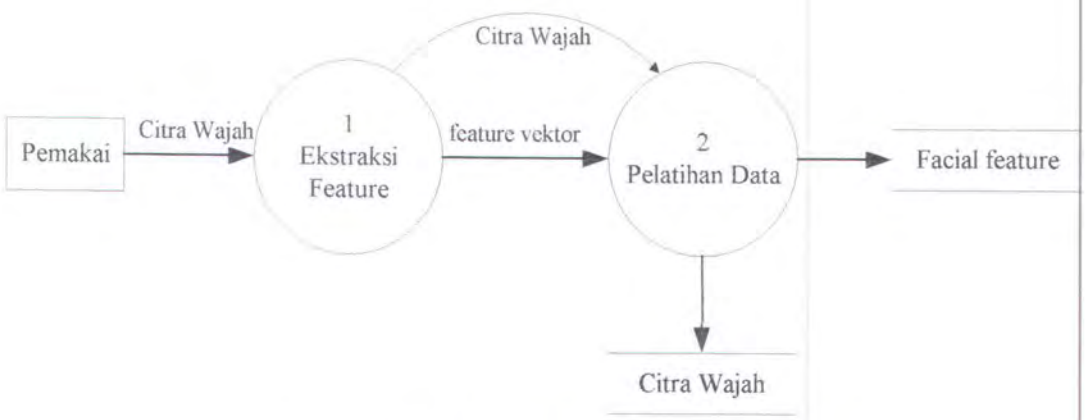
5.3.4.1 Subsistem Pelatihan

Subsistem pelatihan merupakan bagian dari sistem pengenalan wajah dengan menggunakan mixture distance, subsistem pelatihan harus dilakukan dahulu sebelum subsistem pengenalan wajah dilakukan. Secara keseluruhan subsistem pelatihan dapat dilihat pada gambar 5.1, yang menggambarkan kontek diagram atau DFD level 0 dari subsistem pelatihan.



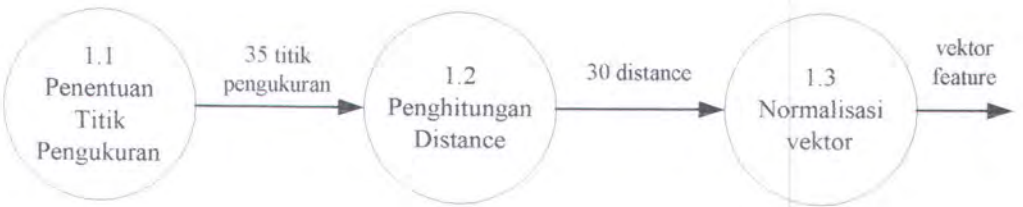
Gambar 5.1 DFD level 0 SubSistem Pelatihan

Data flow diagram diatas menggambarkan sistem pelatihan yang menerima sebuah input citra kedalam proses pelatihan dan menghasilkan penyimpanan data (*data storage*) pada sebuah basis data. Selanjutnya proses pelatihan ini dipecah-pecah menjadi beberapa subproses seperti pada gambar 5.2 yang menggambarkan DFD level 1 dari susbsistem pelatihan.



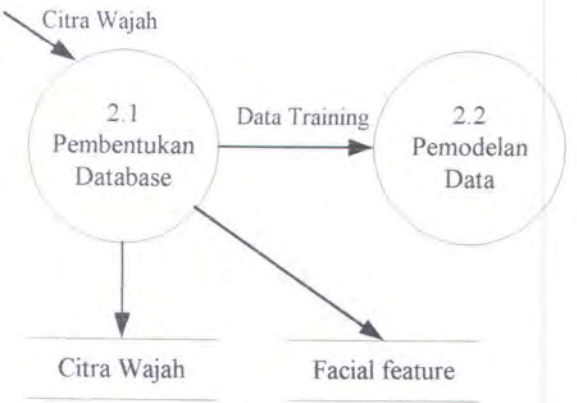
Gambar 5.2 DFD Level 1 Proses Pelatihan

Pada DFD level 1 subsistem pelatihan proses dibagi menjadi dua subproses, proses ekstraksi dan proses pelatihan. Proses ekstraksi merupakan proses yang pertama kali dilakukan untuk mendapatkan vektor feature 30 dimensi dan citra wajah yang akan disimpan.



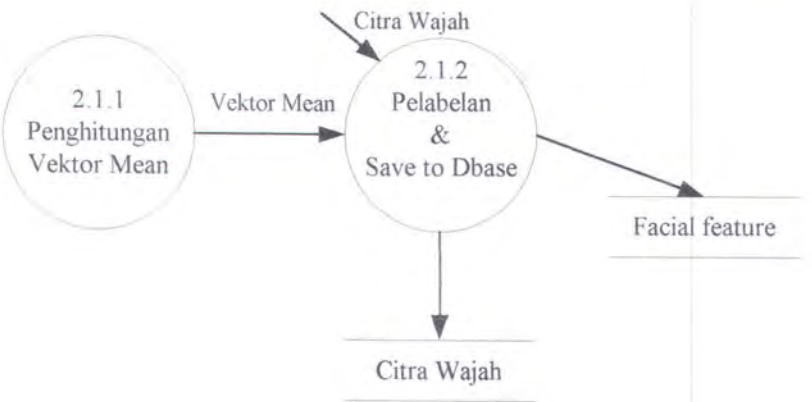
Gambar 5.3 DFD Level 2 Proses Ekstraksi Feature

Proses ekstraksi feature dilakukan secara manual dan terbagi dalam tiga proses, proses pertama yaitu melakukan penentuan 35 titik pengukuran dilanjutkan proses penghitungan distance dari titik-titik yang telah ditentukan dan terakhir proses normalisasi sehingga membentuk suatu vektor feature 30 dimensi dan sebuah citra wajah.



Gambar 5.4 DFD Level 2 Proses Pelatihan Data

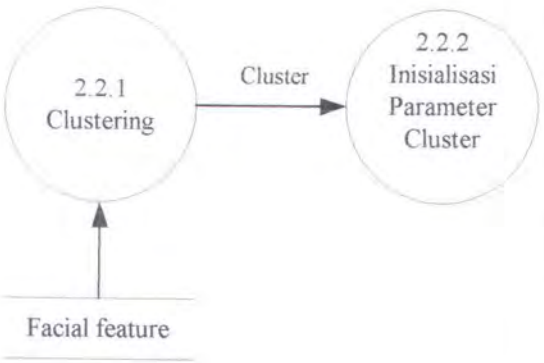
Proses pelatihan data dibagi menjadi dua yaitu proses pembentukan database dan proses pemodelan data. Pada proses pembentukan database, data yang disimpan adalah vektor mean dari vektor feature hasil ekstraksi yang telah dilakukan. Sebelum dilakukan penyimpanan dalam database, proses yang pertama



Gambar 5.5 DFD Level 3 Proses Pembentukan Database

kali dilakukan adalah penghitungan vektor mean yang kemudian dilakukan proses pelabelan untuk memberikan kode setiap feature vektor dan citra wajah.

Proses selanjutnya adalah pemodelan data, data yang dimodelkan adalah kumpulan feature vektor yang tersimpan dalam database.. Proses pemodelan terbagi dalam dua sub proses yaitu clustering dan proses untuk menginisialisasi parameter cluster.



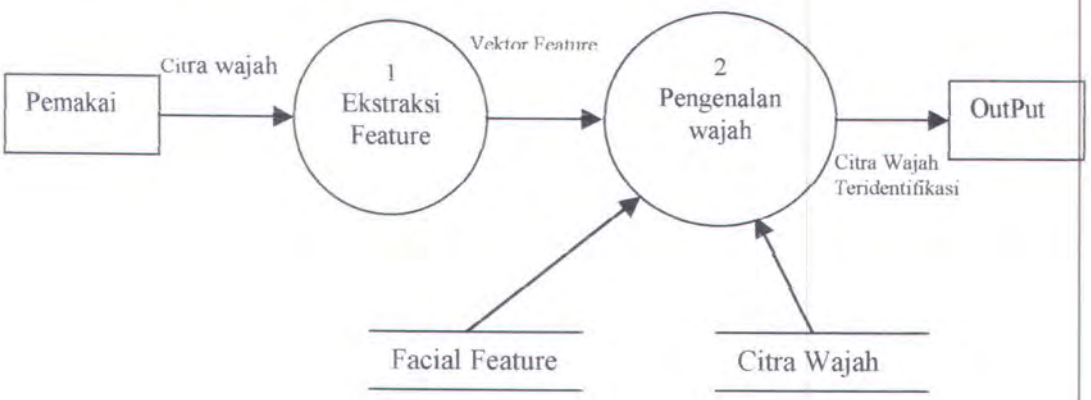
Gambar 5.6 DFD Level 3 Proses Pemodelan Data

5.3.4.2 Subsistem Pengenalan Wajah

Gambaran dari subsistem pengenalan wajah secara keseluruhan dapat digambarkan dengan Digram Aliran Data level 0 (*Data Flow Diagram*). Dari gambar DFD level 0 dapat dilihat cara kerja sistem pengenalan wajah yang secara umum menerima suatu input citra yang akan dikenali, dan kemudian menghasilkan output suatu citra yang teridentifikasi.

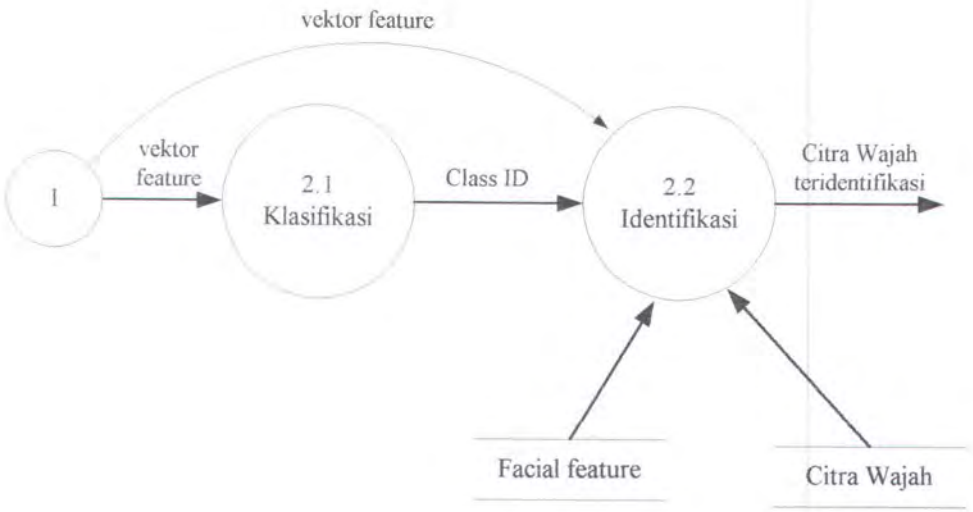
Gambar 5.7 DFD Level 0 SubSistem Pengenalan Wajah

Seperti halnya pada subsistem pelatihan, pada proses pengenalan wajah inipun melewati suatu proses ekstraksi wajah yang telah dijelaskan lebih dahulu, perbedaannya adalah hasil ekstraksi disini tidak disimpan dalam suatu database tetapi dianggap sebagi suatu pola yang akan dikenali (*query*).



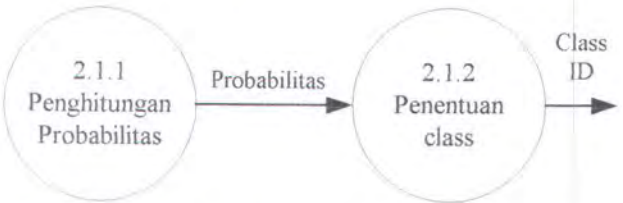
Gambar 5.8 DFD Level 1 Pengenalan Wajah

Proses pengenalan wajah terbagi dalam dua sub proses yaitu klasifikasi dan identifikasi seperti dalam gambar 5.9. Proses klasifikasi meliputi penghitungan proabilitas setiap klas pola dan penentuan klas dengan probabilitas maksimal yang menghasilkan identitas klas tersebut.



Gambar 5.9 DFD Level 2 Proses Pengenalan wajah

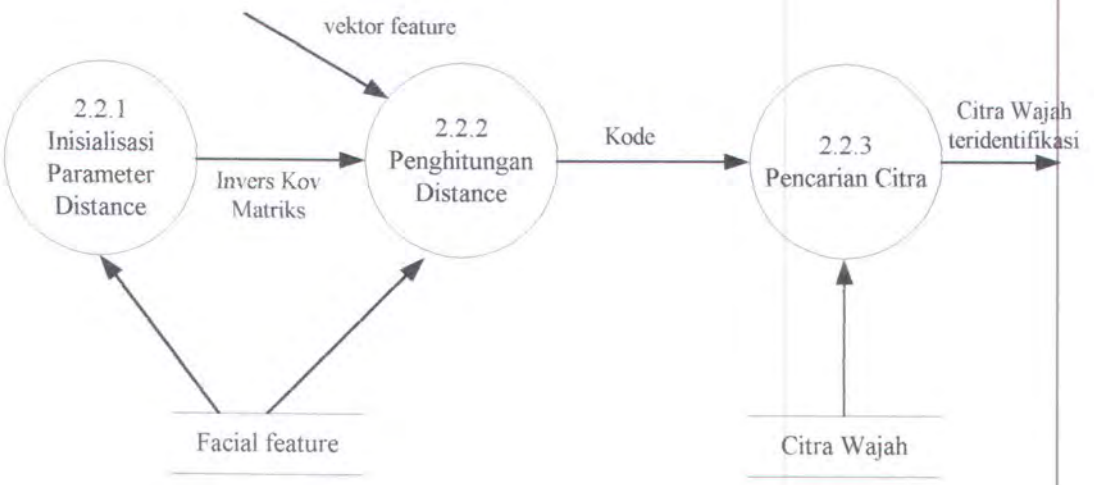
Probabilitas disini adalah probabilitas dari query terhadapa setiap klas pola yang ada untuk mendapatkan klas yang paling sesuai dengan query tersebut, dalam hal ini yaitu klas pola yang menghasilkan probabilitas maksimum. Identitas dari klas tersebut akan digunakan untuk proses selanjutnya.



Gamabar 5.10 DFD Level 3 Proses Klasifikasi

Dari hasil proses klasifikasi kemudian dilanjutkan proses identifikasi yang terbagi dalam tiga proses yaitu inisialisasi parameter distance, proses ini melakukan penghitungan parameter-parameter yang dibutuhkan, hasil dari proses ini adalah invers dari kovarian matrik klas tertentu, proses kedua melakukan penghitungan distance antara query dengan setiap vektor feature yang ada dalam klas yang telah ditentukan. Proses terakhir adalah mencari citra wajah dalam

database yang bersesuaian dengan vektor feature yang teridentifikasi. Gambar 5.11 menggambarkan secara jelas aliran data dan proses dari proses identifikasi.



Gambar 5.11 DFD Level 3 Proses Identifikasi

5.4 Pembuatan Perangkat Lunak

Sub Bab ini akan membahas tentang pembuatan perangkat lunak, dalam hal ini adalah implementasi dari perancangan yang telah dibuat kedalam sebuah progam. Pembuatan perangkat lunak ini meliputi implementasi data dan implementasi proses.

5.4.1 Implementasi Data

Bagian ini akan menjelaskan mengenai data-data yang dipakai dalam semua proses yang dilakukan sistem pengenalan wajah dan implementasinya kedalam sebuah program. Dalam program data-data itu dijabarkan dalam sebuah struktur data.

Vektor feature merupakan salah satu dari data yang penting dalam sistem pengenalan wajah, vektor ini didapat dari proses ekstraksi yang menghasilkan 30 distance sehingga ditampung dalam sebuah array satu dimensi dengan elemen array bertype real.

Pemodelan data pada sistem pengenalan wajah membentuk suatu model data yang terdiri dari beberapa cluster yang dimodelkan sebagai *normal density*, cluster-cluster tersebut mempunyai parameter-parameter yang menyimpan informasi yang ada pada cluster tersebut. Informasi-informasi itu meliputi identitas *cluster*, jumlah anggota, identitas anggota, data anggota, *mean* dan *variance*.

Informasi-informasi dalam setiap cluster itu dihasilkan oleh proses training dan akan digunakan dalam proses pengenalan. Data dalam cluster akan digunakan untuk menghitung fungsi probabilitas setiap cluster terhadap *query* yang akan diklasifikasi, untuk itu diperlukan suatu struktur data yang tepat untuk merepresentasikan cluster-cluster tersebut.

Untuk merepresentasikan cluster tersebut dibuat suatu struktur data berupa record yang diimplementasikan sebagai berikut :

```
Cluster = record
            ID     : byte;//identitas klas
            Num    : byte; //jumlah anggota klas pola
            Key    : array[1..50]of integer;
            Data   : array[1..50]of ADistance;
            Mean   : ADistance;
            Varian : ADistance
          end;
```

5.4.2 Implementasi Proses

Pada bagian ini akan dijelaskan mengenai hirarki proses dari sistem pengenalan wajah dan implementasi dari proses-proses yang telah dirancang kedalam sebuah program.

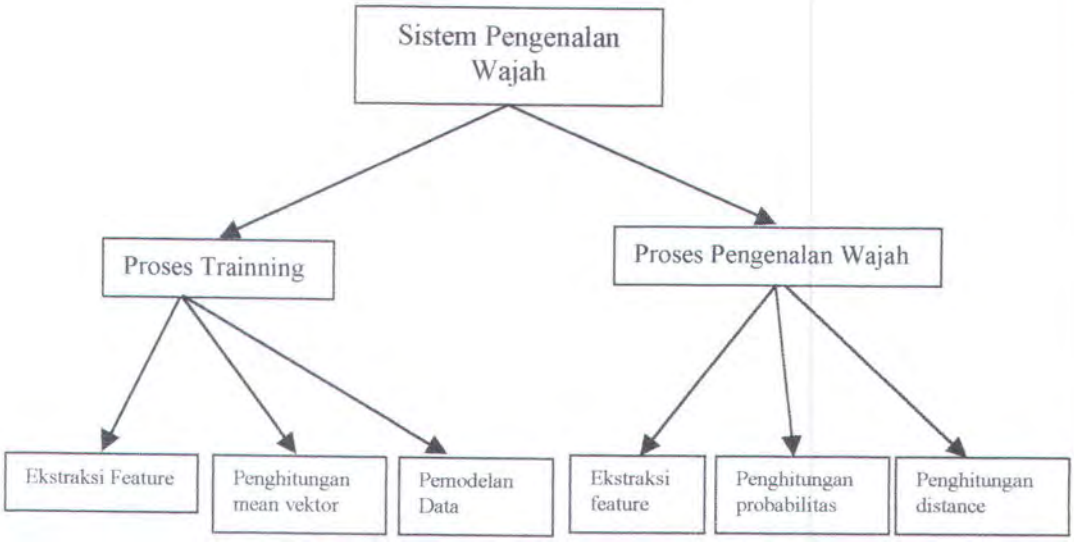
5.4.2.1 Hirarki Proses

Subbagian ini menjelaskan tentang hirarki proses dalam sistem pengenalan wajah dengan menggunakan metode mixture-distance. Seperti telah dijelaskan sebelumnya sistem pengenalan wajah dibagi menjadi dua proses yang utama yaitu proses trainning dan proses pengenalan wajah.

Proses training terdiri dari beberapa proses yaitu proses ekstraksi yang merupakan proses awal untuk mendapatkan informasi-informasi yang dibutuhkan dalam proses selanjutnya, dalam hal ini hasil dari proses ekstraksi adalah vektor feature. Proses selanjutnya adalah penghitungan mean vektor yang akan disimpan dalam database, dan proses terakhir adalah pemodelan data trainning yang ada dalam database menjadi suatu mixture density, dimana setiap klas pola dimodelkan sebagi normal density. Pemodelan inilah yang merupakan hasil dari proses training.

Proses pengenalan wajah juga diawali dengan proses ekstraksi seperti halnya proses training, hasil proses ekstraksi yang berupa vektor feature 30 dimensi ini dianggap sebagai vektor baru yang akan dikenali (*query*). Vektor ini kemudian akan digunakan untuk mencari nilai probabilitas dari masing-masing klas pola, dan untuk menghitung distance dengan vektor-vektor dari anggota klas

tertentu. Gambar berikut menjelaskan secara detail hirarki proses pada sistem pengenalan wajah.



Gambar 5.12 Hirarki Proses

5.4.2.2 Implementasi Program

Bagian ini menjelaskan implementasi proses-proses yang telah dirancang dalam sistem pengenalan wajah kedalam sebuah program, dalam program proses-proses itu diimplementasikan menjadi modul atau prosedur.

Beberapa modul dan prosedur yang penting akan dibahas disini, modul-modul tersebut antara lain :

- Proses ekstraksi, proses ini melibatkan tiga subproses yaitu penentuan titik-titk pengukuran secara manual, penghitungan distance dari titik pengukuran dan normalisasi vektor feature, semua proses tersebut diimplementasikan dalam modul/prosedur berikut :

Proses Ekstraksi

```
procedure TForm1.Image1MouseDown(Sender: TObject; Button:
TMouseButton;  Shift: TShiftState; X, Y: Integer);
var
  C,X1,Y1 : string;
begin
   if (Count<35) then //penentuan titik pengukuran
   begin
    inc(Count);
    Image1.Canvas.Rectangle(x-2,y-2,x+2,y+2);
    Points[Count].X:=X;
    Points[Count].Y:=Y;
   end else showmessage('counter = 35');
   if count=35 then
   begin
      SpeedButton1.Enabled:=true;
      SpeedButton4.Enabled:=true;
   end;
```

prosedur untuk menghitung distance antar titik-titik pengukuran

```
function TForm1.Distance(i,j : Byte): Real;
var
  PX,PY : Integer;
begin
   PX:=sqr((Points[i].X)-(Points[j].X));
   PY:=sqr((Points[i].Y)-(Points[j].Y));
   Distance:=sqrt(PX+PY) ; //hitung distance
end;
```

proses untuk normalisasi vektor feature :

```
for i:=1 to 30 do
      normal:=vec_feature[i]/distance(4,14);
```

- *Clustering*, proses ini mengelompokan data dalam database menjadi beberapa klas pola menurut kedekatannya. Dalam program ini algoritma yang digunakan adalah *K-means clustering*.

```
procedure K_Means;
begin
   table2.First;
   Num_Data:=0;

{ambil data dari tabel tampung dalam sebuah array}
   while not table2.eof do l
   begin
      inc(Num_Data);
      ArTabel[Num_Data].No:=table2.fieldvalues['Kode'];
      for i:=1 to 30 do
         ArTabel[Num_Data].Data[i]:=table2.Fields[i].value;
      table2.next;
   end;

{inisialisasi mean cluster}
   for i:=1 to Max_Cluster do
   begin
      ACluster[i].ID:=i;
      ACluster[i].Mean:=ArTabel[i].Data;
      ACluster[i].Num:=0;
   end;

{mulai pengelompokan data}
   n:=1;
   while n<99 do
   begin
      for i:=1 to Max_Cluster do
```

```
    begin
        MCount[i]:=0;
        for c:=1 to 30 do begin
          Means[i][c]:=0;
      end;

{hitung jarak dengan pusat cluster}
        for j:=1 to Num_Data do
          delta[i][j]:=dist2point(ACluster[i].mean,ArTabel[j].Data);
      end;
      for i:=1 to Num_Data do
      begin

{cari titik dengan jarak minimum dari pusat cluster kemudian isikan
 nilai parameter cluster}
        for j:=1 to Max_Cluster do
          Min[j]:=Delta[j][i];
        MinValue:=Minimal(Min,Max_Cluster);
        inc(MCount[MinValue]);
        AChuster[Minvalue].Num:=MCount[MinValue];
        x:=ACluster[MinValue].Num;
        ACluster[minValue].key[x]:=ArTabel[i].No;
        ACluster[minValue].data[x]:=ArTabel[i].data;
        for j:=1 to 30 do
        begin
          Means[MinValue][j]:=Means[MinValue][j]+ArTabel[i].Data[j];
          ACluster[MinValue].Data[x][j]:=ArTabel[i].Data[j];
        end;
      end;
```

```
{perbaharui nilai mean cluster dengan nilai yang baru}
    for i:=1 to Max_Cluster do
    begin
      if MCount[i]<>0 then
      begin
        for j:=1 to 30 do
          Means[i][j]:=Means[i][j]/Mcount[i];
          ACluster[i].Mean:=Means[i];
      end;
    end;
    inc(n);
  end;
end;
```

- Proses inisialisasi parameter *cluster*, proses ini melakukan penghitungan untuk menginisialisasi nilai parameter dari klas-klas pola yang telah terbentuk. Proses ini diimplementasikan dalam program sebagai berikut :

```
{ hitung nilai mean dari semua feature yang ada}
procedure Means(Member : Byte);
begin
  for i:=1 to 30 do
    begin
      tmp:=0;
      for j:=1 to member do
        tmp:=tmp+Classarr[j,i];
      meansarr[i]:=tmp/Member;
    end;
end;
```

```
{hitung variance dari semua feature }
procedure variance(Member : Byte);
begin
   for i:=1 to 30 do
   begin
     tmp:=0;
     for j:=1 to member do
         tmp:=tmp+sqr(Classarr[j,i]-meansarr[i]);
     varianarr[i]:=tmp/(member-1);
  end
end
```

- Kovariance Matriks merupakan salah satu parameter dari klas pola yang penting, penghitungan kovariance matriks diimplentasikan dalam program sebagai berikut :

```
function CovarianceMatrix(Mtr:Matrix;Member:byte) : Matrix;
var
   M : Matrix;
Begin
{ambil means dari matriks dan tampung data matriks}
   means1:=Kovariansi(Mtr,member);
   for i:=1 to 30 do
    for j:=1 to member do
      artmp[i,j]:=Mtr[j,i];

{mulai hitung nilai untuk setiap entri dari matriks}
   for i:=1 to 30 do
   begin
     for k:=1 to 30 do
      begin
```

```
{perhitungan nilai untuk diagonal entri dari kovarian matrik}
        if i=k then
        begin
          tmp:=0;
          for j:=1 to member do
            tmp:=tmp+sqr(artmp[i,j]-means1[i]);
        end else

{perhitungan nilai untuk  off-diagonal entri dari kovarian matrik}
        begin
          tmp:=0;
          for j:=1 to Member do
            tmp:=tmp+(artmp[i,j]-means1[i])*(artmp[k,j]-means1[k])*skala;
        end;

{isi nilai entri dari kovarian matrik}
        M[i,k]:=tmp/member;
    end;
   CovarianceMatrix:=M;
```

;

- Nilai likelihood didapat dengan menggunakan persamaan (3.14) yang mencerminkan kemiripann (*similarity*) antara *query* dengan klas pola tertentu, diimplementasikan dalam sebuah prosedur berikut ini :

```
function Likelihood(x,v,m:Adistance) : real;
begin
    Kons:=-(30/2)*ln(2*3.14);  { konstanta
    sum1:=0;                        dan inisialisasi variabel }
    sum2:=0;
    tmp:=0;
    for i:=1 to 30 do
```

```
{mulai menghitung nilai likelihood}
  if v[i]<>0 then
    begin
      sum1:=sum1+ln(sqrt(v[i]));
      sum2:=sum2+sqr(x[i]-m[i])/v[i];
    end;
    tmp:=Kons-sum1-sum2;
    Likelihood:=tmp;
 end;
```

- Mahalanobis distance digunakan untuk mencari distance antara *query* dengan setiap elemen klas pola tertentu. Dengan menggunakan persamaan (3.15) penghitungan distance diimplentasikan sebagai berikut :

```
function Mahalanobis(Q,Y:ADistance;CVM:Matrix) : Real;
 begin
 { tampung selisih vektor query dan elemen klas pola}
   for i:=1 to 30 do
     tmp[1,i]:=ABS(Q[i]-Y[i]);

 {Transpose matriks}
   for j:=1 to 30 do
     tmp2[j,1]:=tmp[1,j];

 {Hitung mahalanobis distance}
   mul1:=Multiplication(tmp,CVM,1,30,30);
   mul2:=Multiplication(Mul1,tmp2,1,30,1);
   Mahalanobis:=Mul2[1,1];
end;
```

- Tahap terakhir merupakan modul utama dalam program yang dibuat, dimana didalamnya dipanggil prosedur-prosedur yang telah dijelaskan diatas, meliputi proses klasifikasi, proses identifikasi dan ditambah dengan proses pencarian citra yang bersesuaian dengan vektor feature yang teridentifikasi. Implementasinya dalam program sebagai berikut :

```
procedure Tform1.Findnow;
var
  Query,AMean,
  AVarian     : ADistance; { parameter pencarian}
begin
{inisialisasi awal variabel}
   Likevalue:=-1000;
   Tmp_like:=0;
   Vector_feature;

{tampung vektor query dalam sebuah array}
   for i:=1 to 30 do
     Query[i]:=vec_feature[i]/distance(4,14);

{mulai proses klasifikasi untuk mendapatkan klas pola yang terdekat
 dengan menggunakan fungsi likelihood}
   for i:=1 to Max_Cluster do
   begin
      tmp_like:=Likelihood(Query,Acluster[i].Varian,ACluster[i].Mean);
      if Likevalue<tmp_like then
       begin
          Likevalue:=tmp_like;
          No:=Acluster[i].ID;
      end;
   end;
```

```
{ambil data dari klas pola yang terpilih}
  for i:=1 to ACluster[no].num do
    for j:=1 to 30 do
      clas[i,j]:=Acluster[no].Data[i,j];
    member:=ACluster[no].Num;

 { hitung kovarian matrik dan invers matrik dari klas pola}
  Kov:=CovarianceMatrix(Clas,member);
  invers:=Kov;
  MatrixInversion(Kov,30);

{pencarian distance minimum dengan mahalanobis distance}
 Likevalue1:=1000;
 for i:=1 to ACluster[no].Num do
 begin
    tmp_like1:=mahalanobis(Query,Acluster[No].Data[i],Kov);
    if likevalue1>tmp_like1 then
    begin
      likevalue1:=tmp_like1;
      nok:=ACluster[No].key[i];
    end;
 end;

{pencarian citra wajah dalam database}
 table1.setkey;
 if table1.findkey([Nok]) then {proses jika kode ditemukan}
 begin
     dbimage1.visible:=true;
     EDIT2.TEXT:=floattostr(LikeValue1);
     S1:='Identitas No : '+inttostr(table1.FieldValues['Kode']);
     messagedlg(S1,mtinformation,[mbok],0);
 end
```

```
 else
{ jika citra wajah tidak ditemukan }

  begin
     messagedlg('Image Not Found',mtinformation,[mbok],0);
    edit.text:=floattostr(tmp_like1);
 end;
 edit3.text:=floattostr(likevalue);
end;
```

BAB VI

UJI COBA DAN EVALUASI SISTEM

BAB VI

UJICOBA DAN EVALUASI SISTEM

Pada bab ini akan membahas hasil dari ujicoba yang telah dilakukan sistem dengan sampel wajah yang telah tersedia, dan evaluasi terhadap performance sistem yang telah dibuat mulai dari input citra sampai output (hasil pemrosesan) yang dihasilkan sistem. Evaluasi dilakukan untuk mengetahui kelebihan dan kekurangan program sehingga dapat dilakukan perbaikan-perbaikan yang diperlukan, evaluasi juga digunakan untuk keperluan pengembangan selanjutnya untuk mendapatkan suatu sistem yanng lebih baik dari yang ada sekarang.

Data input berupa sebuah citra wajah dengan ukuran tertentu yang akan dikenali dan output berupa citra wajah yang dikenali dari database wajah yang telah disediakan. Tolok ukur dari keberhasilan sistem pengenalan wajah ini adalah kemampuan sistem untuk mengenali input citra wajah yang diberikan sehingga mendapatkan output berupa citra wajah dari individu yang sama.

6.1 Data Ujicoba

Data input berupa citra wajah didapat dari beberapa situs web yang menyediakan database wajah untuk umum antara lain dari MIT Media Lab, Olivetti Laboratory dan Yale University, selain dari sumber diatas beberapa citra wajah yang dipakai dalam ujicoba ini didapat dengan menggunakan fasilitas capture dari kamera tele conference.

Ujicoba ini melibatkan 300 citra wajah dari 30 individu, dimana setiap individu mempunyai 10 variasi wajah, sebagian dari citra wajah itu digunakan sebagai data pelatihan (*training set*) dan sisanya untuk data ujicoba sistem (*test set*). Citra wajah dapat berupa *grey level* ataupun berwarna (*full color*), sedangkan ukuran citra secara umum tidak dibatasi tetapi untuk kemudahan ukuran citra disarankan tidak lebih kecil dari 92x112 pixel.

Gambar 6.1 Sampel Citra Wajah

6.2 Ujicoba Perangkat Lunak

Ujicoba dilakukan terhadap 30 individu dimana setiap individu mempunyai 10 macam variasi gambar. 5 gambar dipakai untuk data training dan sisanya untuk melakukan ujicoba, sehingga citra wajah yang digunakan untuk data training dan data ujicoba masing-masing sebanyak 150 buah. Seperti telah dibahas pada bab sebelumnya dalam mixture-distance ini terdapat dua macam pemodelan yaitu *first order statistik* dan *second order statistik* yang pada prakteknya dilakukan dengan memilih nilai *scalling factor* (*off diagonal weighting*) antara 0

(*first order*) dan 1 (*second order*). Dalam penggunaan sistem pengenalan wajah ini *scalling factor* dapat diberi nilai antara 0 dan 1 ($0 \leq f \leq 1$), tetapi dalam ujicoba ini hanya akan memakai tiga nilai *scalling factor* yaitu $f = 0$, $f = \frac{1}{2}$, dan $f \approx 1$.

6.2.1 Ujicoba dengan First Order Statistik Model

Fisrt order statistik model ($f = 0$) atau disebut juga dengan *variance model* mengalikan entri dari *off diagonal* kovarian matrik dengan 0, sehingga hanya entri pada diagonal matrik saja yang berpengaruh dalam penghitungan distance. Setiap individu menggunakan 5 data citra wajah dalam proses pelatihan yang diambil secara acak dari 10 variasi wajah, hasil dari ujicoba dapat dilihat pada tabel 6.1 berikut :

Pola	Berhasil	Gagal	Persentase Keberhasilan
1	5	0	100%
2	5	0	100%
3	5	0	100%
4	5	0	100%
5	4	1	80%
6	4	1	80%
7	3	2	60%
8	4	1	80%
9	4	1	80%
10	4	1	80%
11	5	0	100%
12	5	0	100%
13	5	0	100%
14	4	1	80%
15	3	2	60%
16	4	1	80%

17	5	0	100%
18	5	0	100%
19	5	0	100%
20	5	0	100%
21	5	0	100%
22	5	0	100%
23	4	1	80%
24	5	0	100%
25	5	0	100%
26	5	0	100%
27	4	1	80%
28	5	0	100%
29	4	1	80%
30	5	0	100%
Total	136	14	91%

Tabel 6.1 Hasil Ujicoba dengan Fisrt Order Statistik Model

6.2.2 Ujicoba dengan Second Order Statistik Model

Second order statistik model ($f \approx 1$) atau disebut juga *covariance model* memberikan suatu bentuk *full covariance matrice* yang didapat dari suatu klas data tertentu. *Scalling factor* disini diambil dari nilai yang mendekati 1, dalam hal ini $f = 0.99$ untuk menghilangkan pengaruh dari bentuk kovarian matrik yang tidak ideal (*ill conditional*) untuk mendapatkan invers dari kovarian matrik tersebut. Setiap individu menggunakan 5 citra wajah dalam proses training yang diambil secara acak dari 10 variasi wajah yang ada, hasil ujicoba dapat dilihat pada tabel 6.2 berikut :

Tabel 6.2 Hasil Ujicoba dengan Second Order Statistik Model

Pola	Berhasil	Gagal	Persentase Keberhasilan
1	4	1	80%
2	4	1	80%

3	4	1	80%
4	5	0	100%
5	3	2	60%
6	4	1	80%
7	3	2	60%
8	4	1	80%
9	5	0	100%
10	5	0	100%
11	2	3	40%
12	3	2	60%
13	4	1	80%
14	4	1	80%
15	5	0	100%
16	4	1	80%
17	3	2	60%
18	2	3	40%
19	3	2	60%
20	3	2	60%
21	3	2	60%
22	2	3	40%
23	4	1	80%
24	4	1	80%
25	5	0	100%
26	5	0	100%
27	4	1	80%
28	5	0	100%
29	3	2	60%
30	5	0	100%
Total	114	36	76%

6.2.3 Ujicoba dengan Scaling Factor Tertentu

Scaling factor digunakan untuk memilih pemodelan antara *first order statistik* dan *second order statistik*, nilai yang dapat diberikan antara 0 dan 1 ($0 \leq f \leq 1$). Pada dasarnya nilai *scaling factor* dapat diubah-ubah sekehendak kita untuk mendapatkan performance pengenalan yang terbaik, dalam ujicoba ini

nilai yang akan dipakai adalah $f = \frac{1}{2}$. Setiap individu menggunakan 5 citra wajah dalam proses training yang diambil secara acak dari 10 variasi wajah yang ada, hasil ujicoba dapat dilihat pada tabel 6.3 berikut :

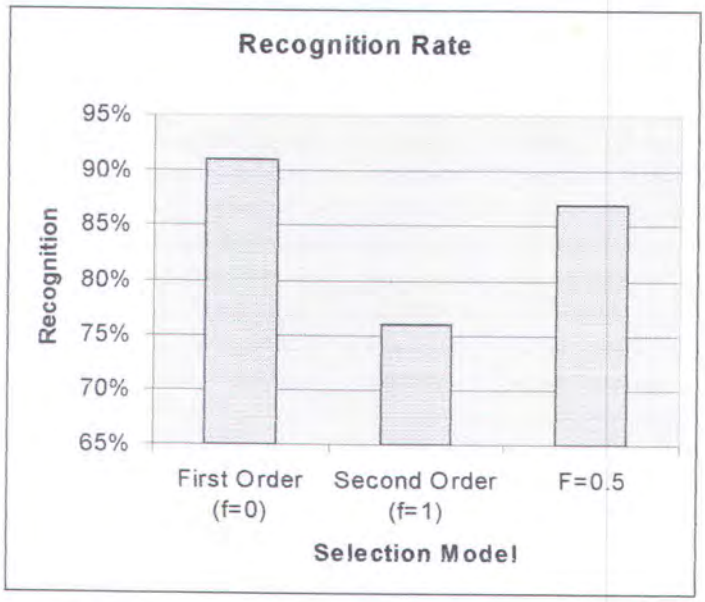
Tabel 6.3 Hasil Ujicoba dengan Scaling Factor = 0.5

Pola	Berhasil	Gagal	Persentasi Keberhasilan
1	5	0	100%
2	5	0	100%
3	4	1	80%
4	5	0	100%
5	4	1	80%
6	4	1	80%
7	3	2	60%
8	4	1	80%
9	5	0	100%
10	4	1	80%
11	5	0	100%
12	5	0	100%
13	5	0	100%
14	4	1	80%
15	5	0	100%
16	4	1	80%
17	5	0	100%
18	2	3	40%
19	3	2	60%
20	5	0	100%
21	4	1	80%
22	4	1	80%
23	4	1	80%
24	5	0	100%
25	5	0	100%
26	5	0	100%
27	5	0	100%
28	5	0	100%
29	3	2	60%
30	5	0	100%
Total	131	19	87%

6.3 Evaluasi Sistem

Pada sub bab ini akan dibahas mengenai hasil yang didapat dari ujicoba, mengamati tingkat keakuratan sistem pengenalan wajah, dan faktor-fakor yang mempengaruhinya kemampuan sistem dalam pengenalan wajah. Dari hasil ujicoba yang telah dilakukan diperoleh performance yang berbeda untuk setiap model yang ada, secara umum first order statistik mempunyai performace pengenalan yang tertinggi dengan tingkat pengenalan 91% kemudian pengenalan dengan scalling factor $f=0.5$ dengan teingkat pengenalan 87% dan terakhir second order statistik dengan 76%, lebih jelasnya dapat dilihat pada gambar 6.2.

Faktor yang mempengaruhi proses pengenalan dapat terjadi pada saat proses ekstraksi maupun proses pengenalan. Saat proses ekstraksi wajah setiap citra wajah akan menghasilkan vektor feature yang berbeda walaupun dari satu citra wajah yang sama, hal ini disebabkan proses ekstraksi dilakukan secara manual sehingga error dalam melakukan ekstraksi selalu ada. Untuk satu citra



Gambar 6.2 Tingkat Pengenalan dari setiap Model

wajah yang sama jika diekstraks dua kali akan menghasilkan dua vektor feature yang berbeda karena terjadinya kesalahan pada saat penentuan titik pengukuran. Sedangkan pada saat proses pengenalan, tingkat pengenalan terhadap suatu input citra dipengaruhi oleh nilai *scalling factor* yang diberikan, setiap nilai dapat menghasilkan *recognition rate* yang berbeda.

BAB VII

PENUTUP

BAB VII

PENUTUP

Pada bab ini akan dibahas mengenai beberapa kesimpulan yang dapat diambil dari keseluruhan tugas akhir yang telah dibuat. Kesimpulan diambil berdasarkan hasil ujicoba yang telah dilakukan terhadap sistem pengenalan wajah dengan data yang tersedia.

Bab ini juga memberikan saran-saran untuk perbaikan dan pengembangan selanjutnya untuk mendapatkan sistem pengenalan wajah yang lebih baik. Perbaikan dapat dilakukan pada praproses ataupun pada proses training dan pengenalannya, sehingga dapat menghasilkan peningkatan kemampuan sistem.

7.1 Kesimpulan

Dari ujicoba yang telah dilakukan sistem pengenalan wajah ini dapat diambil beberapa kesimpulan :

1. Perangkat lunak yang diimplementasikan dengan metode mixture distance dapat digunakan untuk sistem pengenalan wajah dengan input yang berupa citra dua dimensi.
2. Metode mixture distance merupakan suatu metode yang sangat baik untuk diterapkan dalam sistem pengenalan wajah dengan tingkat pengenalan tertinggi mencapai 91%.
3. Kemampuan pengenalan yang dimiliki *metode mixture distance* dipengaruhi oleh beberapa faktor yaitu proses ekstraksi, data training saat proses pelatihan,

pemilihan model antara *fisrt order* dan *second order* dan pemberian nilai *scalling factor* (*off diagonal weighting*). Untuk *first order* diperoleh tingkat keberhasilan 76%, *second order* mencapai tingkat keberhasilan 91% dan *scalling factor* dengan $f=0.5$ mempunyai tingkat keberhasilan 87%.

7.2 Saran

Dari pembuatan tugas akhir ini, beberapa saran berikut ini dapat dipertimbangkan untuk pengembangan selanjutnya untuk mendapatkan suatu sistem yang lebih baik :

1. Dalam tugas akhir ini penentuan titik pengukuran dilakukan secara manual, untuk otomatisasi maka perlu dibuat suatu ekstraksi otomatis yang dapat melakukan penentuan titik pengukuran secara otomatis.
2. Perlu dilakukan ujicoba terhadap database citra yang lebih besar untuk mengetahui kemampuan sistem pengenalan wajah dengan metode *mixture distance* dalam menangani database yang cukup besar.

DAFTAR PUSTAKA

DAFTAR PUSTAKA

1. Agus Subhan Akbar, *Perancangan dan pembuatan perangkat lunak pengenalan wajah dengan jaringan syaraf berdasarkan keputusan probabilistik*. Teknik Informatika Institut Teknologi Sepuluh Nopember Surabaya, 1999.
2. Al Fathoni Aminudin, *Perancangan dan pembuatan perangkat lunak pengenalan wajah dengan jaringan syaraf konvolusional*. Teknik Informatika Institut Teknologi Sepuluh Nopember Surabaya, 1998.
3. Christophe Couvreur. *The EM algorithm : A guide tour*. Polytechnique de Mons.
4. Howard Anton, *Elementary Linear Algebra*. Drexel University, 1991.
5. Ingemar J.Cox, Joumana Ghosn, and Peter N.Yianilos. *Feature-based recognition using mixture-distance*. In International Conference on Computer Vision and Pattern, IEEE Press, 1996.
6. Peter Yianilos. *Metric learning via normal mixtures*. Technical report, NEC Research Institute, 1995.
7. Steve Lawrence, Peter Yianilos, and Ingemar Cox. *Face recognition using mixture-distance and raw image*. NEC Research Institute, 1996.
8. S.Sakamoto and J.Tojima. *Face feature analysis for human identification*. Technical Report, NEC Central Research Laboratory, 1994.

LAMPIRAN

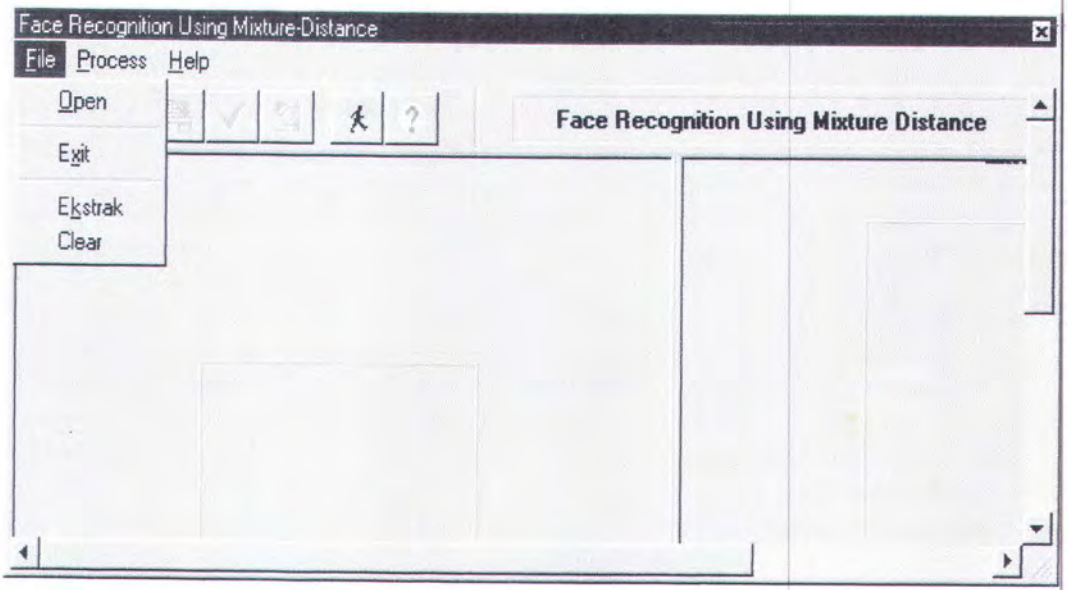
USER MANUAL

Berikut ini petunjuk penggunaan perangkat lunak pengenalan wajah, beberapa fasilitas menu yang tersedia dalam perangkat lunak ini antara lain :

Menu File

Dalam menu file ini terdapat beberapa sub-menu yaitu :

- Sub-menu Open, berfungsi untuk membuka file gambar yang akan diproses.
- Sub-menu Exit, berfungsi untuk keluar dari program.
- Sub-menu Ekstrak, berfungsi untuk melakukan ekstraksi pada input citra.
- Sub-menu Clear, berfungsi untuk membersihkan gambar dari titik pengukuran.

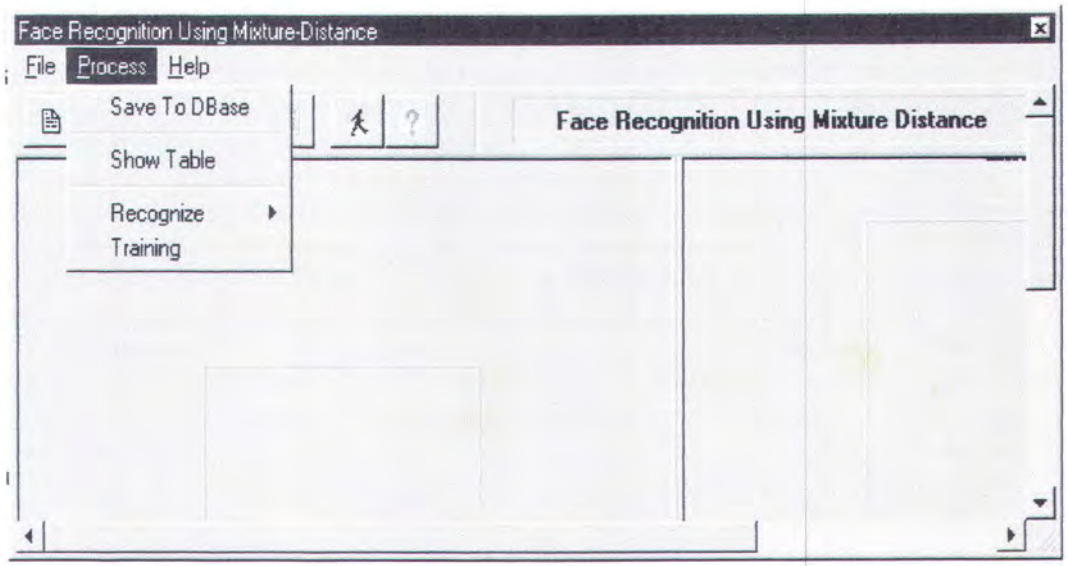


Tampilan dari Menu File

Menu Process

Fasilitas-fasilitas yang termasuk dalam menu process yaitu :

- Save to DBase, berfungsi untuk memasukan data hasil ekstraksi kedalam database.
- Show Table, berfungsi untuk menampilkan tabel yang ada dalam database.
- Recognize, berfungsi untuk melakukan proses pengenalan, terdiri dari tiga jenis pilihan model pengenalan :
  - Fisrt Order Statistic Model, melakukan proses pengenalan berdasarkan pendekatan fisrt order.
  - Second Order Statistic Model, melakukan proses pengenalan berdasarkan pendekatan second order.
  - Scaling Factor (0<f<1), melakukan proses pengenalan dengan scaling factor tertentu antara 0 dan 1.
- Training, berfungsi untuk melakukan proses pemodelan data training jika telah memasukan data baru dalam database.

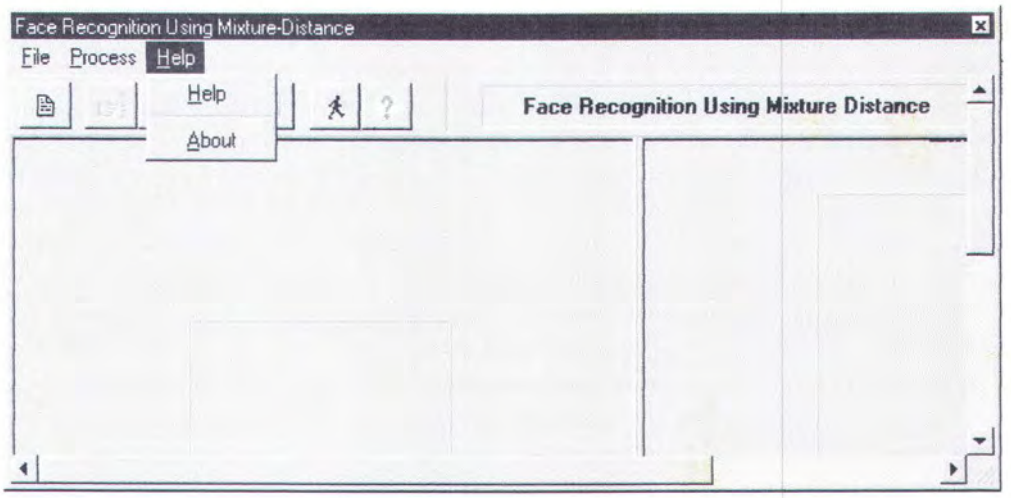


Tampilan Menu Process

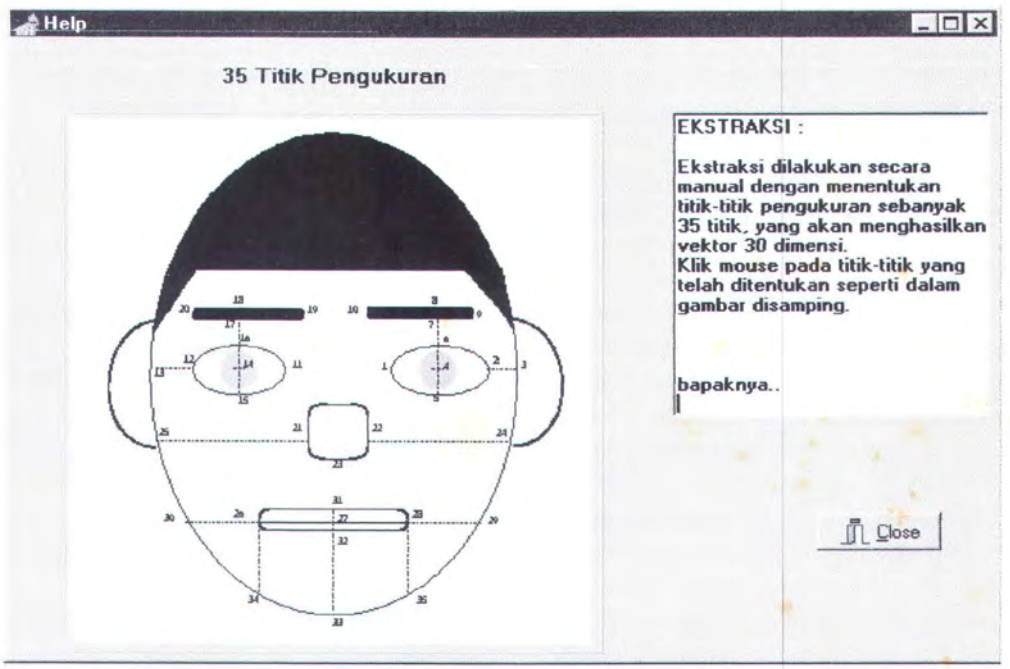
Menu Help

Menu ini berisi sub-menu help dan about.

- Sub-menu help, berisi petunjuk untuk melakukan proses penentuan titik pengukuran.
- Sub-menu about, berisi informasi pembuat program.



Tampilan Menu Help



Tampilan Sub-Menu Help