OPTIMASI PCA PADA SISTEM PENGENALAN WAJAH 3 DIMENSI MENGGUNAKAN ALGORITMA GENETIKA

Adila Alfa Krisnadhi

Fakultas Ilmu Komputer, Universitas Indonesia
Kampus UI Depok, Jawa Barat 16424, Indonesia
e-mail: adila@cs.ui.ac.id

ABSTRAK

Principal Component Analysis (PCA) merupakan sebuah metode transformasi yang sangat berguna dalam sistem pengenalan wajah tiga dimensi. PCA berperan sangat baik sebagai alat pengekstraksi ciri yang sangat dibutuhkan dalam proses klasifikasi objek tigak dimensi yang diwakili oleh sekumpulan citra wajah dua dimensi. Dalam proses ekstraksi ciri, dilakukan transformasi yang sekaligus melibatkan proses reduksi dimensi untuk mendapatkan ciri-ciri optimal sebagai basis ortogonal ruang wajah. Namun, pada setiap himpunan citra wajah yang berbeda, proses ini harus dilakukan berulang-ulang karena tingkat reduksi dimensi tersebut ditentukan oleh suatu parameter proporsi kumulatif nilai eigen yang harus ditentukan secara manual dari luar sistem. Akibatnya, proses untuk mendapatkan tingkat reduksi dimensi yang terbaik menjadi terhambat karena adanya proses *trial and error* tersebut. Di sini akan dijelaskan sebuah metode untuk mengotomatisasi dan mengoptimasi proses di atas dengan menggunakan algoritma genetika. Hasil eksperimen menunjukkan kinerja yang tidak kalah bahkan mampu memperbaiki kinerja PCA tanpa dikombinasikan dengan algoritma genetika, sehingga di sini proses otomatisasi dan optimasi yang diharapkan dapat dinyatakan berhasil.

Kata kunci: *principal component analysis*, otomatisasi, optimasi, reduksi dimensi, algoritma genetika, sistem pengenalan wajah 3 dimensi.

Makalah diterima [10 Februari 2003]. Revisi akhir [10 Juni 2003].

1. PENDAHULUAN

Sistem pengenalan wajah tiga dimensi memandang wajah manusia terdiri dari dari sekumpulan ciri-ciri baik yang penting maupun tidak penting. Berdasarkan cara pandang ini, sistem pengenalan wajah melakukan klasifikasi wajah dengan cara mencocokkan ciri-ciri suatu citra wajah yang tidak diketahui ke dalam suatu kelompok citra wajah yang ciri-cirinya sudah dikenal. Oleh sebab itu, dalam sistem pengenalan wajah biasanya diterapkan suatu teknik untuk mendapatkan ciri-ciri yang penting dari sekumpulan citra wajah dalam suatu basisdata citra yang dijadikan sebagai acuan atau *reference* [3] [4].

Teknik ekstraksi ciri tersebut diwujudkan dalam suatu transformasi terhadap data awal ke dalam suatu ruang representasi yang lain. Dalam proses transformasi ini diharapkan dihasilkannya suatu optimalisasi ekstraksi ciri baik dalam hal ukuran data yang lebih kecil maupun aproksimasi yang lebih baik terhadap karakteristik data awalnya [11]. Kemudian, proses klasifikasi yang dilakukan akan diterapkan di dalam ruang representasi yang didapatkan setelah transformasi dikerjakan, yakni dengan cara melakukan komparasi dari citra yang akan diklasifikasi terhadap *template* yang dihasilkan oleh proses ekstraksi ciri tersebut [3] [9]. Proses klasifikasi ini dilakukan dengan menggunakan suatu metrik yang sesuai, misalnya menggunakan jarak Euclidean [3].

Salah satu transformasi data yang dikenal adalah *Principal Component Analysis* (PCA) yang menggunakan transformasi Karhunen-Loeve. Transformasi ini dapat melakukan reduksi dimensi pada data citra masukan, sekaligus pula mendapatkan ciri-ciri data citra wajah [3] [12].

PCA bekerja dengan memanfaatkan karakteristik ortogonalitas dari suatu ruang vektor eigen yang dihasilkan dari transformasi terhadap data citra masukan awal. Ruang eigen ini dibentangkan oleh sekumpulan vektor eigen yang berperan sebagai vektor-vektor basis ciri yang saling ortogonal [12].

Di dalam PCA ini kemudian dikerjakan suatu proses reduksi dimensi data dalam hal ini dari dimensi data yang bersesuaian dengan dimensi citra masukan direduksi sehingga cukup disimpan ciri-ciri penting dengan dimensi yang jauh lebih kecil dari data asilnya. Dalam proses ini terdapat suatu parameter nilai proporsi kumulatif nilai eigen α yang menentukan tingkat reduksi dimensi yang dihasilkan [2] [12]. Permasalahan yang kemudian muncul adalah ternyata untuk suatu himpunan data tertentu, nilai α yang optimal – menghasilkan tingkat kebenaran klasifikasi yang terbesar – harus ditemukan melalui proses eksperimen karena tidak

adanya suatu ekspresi kuantitatif antara nilai α dengan kinerja optimal yang diharapkan. Oleh sebab itu, harus dirancang suatu cara untuk memperoleh kinerja optimal tersebut secara otomatis tanpa harus melakukan berulang kali eksperimen yang memakan waktu.

Dalam paper ini disajikan metode otomatisasi dan optimasi PCA dengan memanfaatkan algoritma genetika. Penelitian lain yang berkaitan dengan penerapan algoritma genetika pada PCA dan juga transformasi Fisher dapat dirujuk di [11]. Di sini, algoritma genetika dipilih karena karakteristiknya yang cocok untuk penyelesaian permasalahan optimasi [5] [10]. Sementara itu, pengukuran atas kinerja metode ini didasarkan pada tingkat pengenalan yang dihasilkan.

2. PRINCIPAL COMPONENT ANALYSIS

Berikut ini akan dijelaskan terlebih dahulu dasar dari Principal Component Analysis (PCA) yang dipakai dalam sistem pengenalan wajah tiga dimensi. Penjelasan-penjelasan yang berkaitan dengan prosedur berikut juga terdapat di beberapa penelitian sebelumnya [2] [3] [7] [12]

Andaikan **x** adalah sebuah vektor data masukan dari suatu citra masukan M x M. yang dinyatakan dalam bentuk kolom $D = M^2$:

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \mathrm{M} \\ x_D \end{bmatrix} \tag{1}$$

Suatu matriks kovarian **A** dibentuk dari suatu himpunan data masukan $\mathbf{x}_i$, $i = 1,2,\ldots,P$; sebagai berikut:

$$\mathbf{A} = \frac{1}{P}\sum_{i=1}^{P}(\mathbf{x}_i - \mathbf{c})(\mathbf{x}_i - \mathbf{c})^T \tag{2}$$

di mana **c** merupakan vektor rata-rata dari seluruh data:

$$\mathbf{c} = \frac{1}{P}\sum_{i=1}^{P}\mathbf{x}_i \tag{3}$$

Perhatikan bahwa matriks kovarian **A** yang dihasilkan di sini merupakan sebuah matriks yang simetrik berukuran N x N, $N = D = M^2$, sehingga **A** pasti memiliki suatu himpunan vektor-eigen yang ortonormal. [1].

Dari matriks kovarian **A**, dengan menggunakan suatu metode penyelesaian permasalahan eigen, didapatkan suatu matriks **E**:

$$\mathbf{E}_{NxN} = [\mathbf{e}_1, \mathbf{e}_2, \mathrm{K}\ , \mathbf{e}_N] \tag{4}$$

yaitu matriks yang kolom-kolomnya yakni

$$\mathbf{e}_j = \begin{bmatrix} e_{1j} \\ e_{2j} \\ \mathrm{M} \\ e_{Nj} \end{bmatrix},\ j = 1, 2, \mathrm{K}\ , N \tag{5}$$

merupakan vektor-vektor eigen ortonormal dari matriks kovarian **A**. Juga didapatkan suatu vektor λ :

$$\lambda = \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \mathrm{M} \\ \lambda_N \end{bmatrix} \tag{6}$$

sebagai vektor yang elemen-elemennya merupakan nilai-nilai eigen dari matriks kovarian **A** yang bersesuaian dengan vektor-vektor eigen dalam **E** sedemikian hingga (λ_j , $\mathbf{e}_j$) merupakan pasangan nilai-vektor eigen yang bersesuaian. Dalam hal ini, matriks **E** merupakan matriks transformasi yang akan memetakan vektor data awal $\mathbf{x}_i$ menjadi vektor data $\mathbf{y}_i$ di dalam ruang eigen. Transformasi ini dilakukan menurut persamaan [12]:

$$\mathbf{y}_i = \mathbf{E}^T(\mathbf{x}_i - \mathbf{c}) \tag{7}$$

Sebaliknya, karena matriks **E** berisi vektor-vektor eigen yang ortonormal sehingga $\mathbf{E}^{-1} = \mathbf{E}^T$, maka vektor data awal xi dapat direkonstruksi dengan persamaan [12]:

$$\mathbf{x}_i = \mathbf{E}\mathbf{y}_i + \mathbf{c} \tag{8}$$

Reduksi dimensi dilakukan dengan cara memilih komponen-komponen utama (*principal component*), yakni sejumlah K, $K < N$, vektor-eigen yang bersesuaian dengan K nilai-eigen terbesar [2] [12]. K vektor-eigen tersebut menjadi elemen matriks transformasi **W** :

$$\mathbf{W} = [\varepsilon_1, \varepsilon_2, \mathrm{K}\ , \varepsilon_K] = \begin{bmatrix} w_{11} & w_{12} & \mathrm{K} & w_{1K} \\ w_{21} & w_{22} & \mathrm{K} & w_{2K} \\ \mathrm{M} & \mathrm{M} & \mathrm{O} & \mathrm{M} \\ w_{N1} & w_{N2} & \mathrm{K} & w_{NK} \end{bmatrix} \tag{9}$$

Dengan demikian kita dapat melakukan transformasi atas vektor data awal $\mathbf{x}_i$ dengan menggunakan persamaan:

$$\mathbf{y}_i' = \mathbf{W}^T(\mathbf{x}_i - \mathbf{c}) \tag{10}$$

yang sekaligus merupakan reduksi dimensi data dari dimensi N ke dimensi K, karena $\mathbf{W}^T$ berdimensi K x N dan $(\mathbf{x}_i - \mathbf{c})$ berdimensi N x 1, sehingga membuat hasil transformasi $\mathbf{y}_i'$ berdimensi K x 1. Tingkat kesalahan

kuadrat ratâ-rata (*mean square error*) pada reduksi dimensi ini adalah [12]:

$$\mathbf{err}_{mse} = \sum_{i=1}^{N-1}\lambda_i - \sum_{j=1}^{K}\lambda_j = \sum_{j=K+1}^{N-1}\lambda_j \qquad (11)$$

Pada reduksi dimensi tersebut, pemilihan K vektor-eigen yang terbesar memanfaatkan sebuah parameter proporsi kumulatif α^K [8] [12]:

$$\alpha^K = \frac{\sum_{i=1}^{K}\lambda_i}{\sum_{i=1}^{N-1}\lambda_i} \qquad (12)$$

Proporsi kumulatif menentukan berapa besar harga K yang dipakai dalam pemilihan vektor-eigen. Dalam hal ini proporsi kumulatif biasanya merupakan sebuah nilai yang sudah ditentukan terlebih dahulu. Nilai proporsi kumulatif terbaik umumnya didapat melalui eksperimen [8] [9] [12].

3. ALGORITMA GENETIKA UNTUK OTOMATISASI DAN OPTIMASI REDUKSI DIMENSI

Untuk menghindari berbagai eksperimen berulang-ulang demi memperoleh suatu nilai proporsi kumulatif α yang optimal, diterapkan algoritma genetika. Penerapan algoritma genetika disandarkan pada ekspektasi kita pada keadaan optimal dari α yakni tercapainya tingkat pengenalan yang tinggi dan ukuran hasil reduksi dimensi yang kecil. Penerapan algoritma genetika (*Genetic Algorithm = GA*) mengikuti langkah-langkah: pengkodean, penghitungan nilai *fitness*, reproduksi, *crossover* dan mutasi [5] [10].

3.1. Pengkodean

Reduksi dimensi dilakukan dengan memilih dari N vektor eigen, sejumlah k vektor eigen dengan nilai eigen terbesar. Oleh sebab itu yang menjadi domain dari kandidat solusi adalah himpunan $\mathbf{S} = \{1 \le k \le N, k \in \mathrm{Z}\}$. Masing-masing nilai k inilah yang dikodekan sebagai kandidat solusi dengan menggunakan suatu string biner dengan panjang berhingga.

Andaikan untuk suatu sesi pelatihan terdapat N vektor eigen, maka panjang string biner untuk pengkodean nilai k sebagai kandidat solusi adalah $\lceil \log_2(N-1) \rceil$, di mana $\lceil x \rceil$ menyatakan bilangan bulat terkecil yang lebih besar atau sama dengan x.

Disini terdapat kemungkinan string yang berjumlah $2^{\lceil \log_2(N-1) \rceil}$ yang memiliki interpretasi nilai k dalam domainnya yaitu pada interval $[1, N-1]$. Oleh sebab itu, misalkan suatu string biner sepanjang $n = \lceil \log_2(N-1) \rceil$ yakni: $b_1 b_2 \mathrm{K}\ b_n$, $b_i \in \{0,1\}$, akan memiliki interpretasi nilai fenotip k:

$$k = \left\lfloor 1 + \frac{t}{2^n}(N-1) \right\rfloor \qquad (13)$$

di mana

$$t = 2^{n-1}b_1 + 2^{n-2}b_2 + \mathrm{K} + 2b_{n-1} + b_n = \sum_{i=1}^{n} 2^{n-i} b_i \qquad (14)$$

Contohnya string biner 00001111111 dengan panjang 11 bit akan diinterpretasikan sebagai nilai k = 100.

3.2. Penghitungan Nilai Fitness

Setiap individu (string) mewakili suatu nilai k tertentu yang berkaitan dengan suatu matriks transformasi reduksi dengan vektor eigen yang dipilih sejumlah k. Reduksi yang optimal adalah apabila k dipilih menghasilkan tingkat pengenalan yang tinggi. Permasalahannya, tidak ada hubungan kuantitatif antara k yang dipilih dengan tingkat pengenalan yang akan diperoleh. Oleh sebab itu dari seluruh himpunan citra masukan, dipisahkan sekelompok kecil citra acuan GA untuk penghitungan nilai *fitness*.

Misalkan himpunan citra acuan GA tersebut adalah **Q***, maka fungsi *fitness* kita yang dipakai menjadi:

$$\begin{aligned} \mathit{Fitness} &= f(\mathbf{v}, \mathbf{E}, \mathbf{Q}^*) \\ &= g(k, \mathbf{E}, \mathbf{Q}^*) \end{aligned} \qquad (15)$$

Fungsi ini dihitung dengan cara menghitung tingkat pengenalan pada himpunan **Q***. Untuk suatu individu **v** yang merepresentasikan nilai k jumlah vektor eigen utama, dibuat suatu matriks **E*** yang berisi k vektor eigen utama dari ruang eigen **E**. Kemudian menggunakan matriks **E***, citra-citra acuan pelatihan pada proses transformasi Karhunen-Loeve, ditransformasikan untuk menjadi basis ciri. Selanjutnya pada citra-citra acuan GA dalam himpunan **Q***, kita hitung jarak Euclidean terdekat dengan basis ciri sebagai nilai *fitness*.

Perhatikan bahwa nilai *fitness* terbaik berhubungan dengan jarak yang terpendek (terkecil), sehingga mendapatkan nilai *fitness* sebagai fungsi yang monotonik naik untuk memudahkan konstruksi roulette wheel nantinya, maka dilakukan pemetaan sebagai berikut [6]:

$$\rho_i = \rho_{\max} - \frac{\rho_{\max} - \rho_{\min}}{\delta_{\max} - \delta_{\min}}(\delta_i - \delta_{\min}) \qquad (16)$$

di mana :

ρ_i = nilai *fitness* untuk individu ke-*i*

δ_i = jarak rata-rata individu ke-i

δ_{min} = jarak rata-rata minimum populasi

δ_{max} = jarak rata-rata maksimum populasi

ρ_{min} = nilai *fitness* minimum

ρ_{max} = nilai *fitness* maksimum

dengan jarak rata-rata suatu individu didefinisikan sebagai rata-rata jarak seluruh citra acuan GA dalam himpunan Q^* ke ruang ciri yang dibentangkan oleh matriks **E***.

3.3. Reproduksi

Proses reproduksi dalam GA di sini dilakukan dengan memakai skema *biased roulette wheel* [10]. Untuk itu dihitung berturut-turut :

a. nilai *fitness* $f(\mathbf{v}_i)$ untuk setiap kromosom $\mathbf{v}_i$, $i = 1, \ldots,$ *pop_size*.

b. total *fitness* populasi: $F = \sum_{i=1}^{pop\_size} f(\mathbf{v}_i)$.

c. peluang pemilihan p_i untuk setiap kromosom $\mathbf{v}_i$, $i = 1, \ldots,$ *pop_size* di dalam populasi: $p_i = \frac{f(\mathbf{v}_i)}{F}$.

d. nilai kumulatif peluang q_i untuk setiap kromosom $\mathbf{v}_i$, $i = 1, \ldots,$ *pop_size* di dalam populasi: $q_i = \sum_{j=1}^{i} p_j$.

Di sini setiap kromosom/string diasosiasikan dengan slot yang berkaitan dengan interval $[q_{i-1}, q_i]$, dengan $q_0 = 0$. Lalu, dibangkitkan suatu bilangan riil acak antara 0 dan 1 sebanyak *pop_size* kali. Untuk setiap bilangan riil r_j, apabila $q_{i-1} < r_j \leq q_i$, maka string ke-*i* terpilih sebagai kandidat solusi untuk generasi berikutnya. (Kecuali apabila $r_j = 0$, maka tetap string pertama yang terpilih). Dengan demikian, kita sudah mendapatkan string baru sebanyak *pop_size* untuk generasi berikutnya. Kumpulan string baru ini disebut *mating-pool*.

3.4. Crossover

Di sini, dua string *parents* dipilih dari *mating-pool* dua demi dua berturut-turut dari string yang pertama untuk dipasang-pasangkan. Selanjutnya untuk setiap dua kandidat *parents*, dibangkitkan sebuah bilangan riil acak antara 0 dan 1. Apabila didapatkan bilangan riil antara 0 dan probabilitas *crossover* p_c, maka kedua kandidat *parents* akan melakukan *crossover*, sedangkan jika didapatkan bilangan riil yang lebih besar dari p_c, maka kedua kandidat *parents* tidak akan melakukan *crossover* dan langsung dikopikan sebagai dua string kandidat dalam generasi berikutnya.

3.5. Mutasi

Operasi ini diterapkan dengan mengubah bit pada suatu posisi di dalam suatu kromosom dengan probabilitas mutasi p_m. Untuk setiap kromosom $\mathbf{v}_i$, $i = 1, \ldots,$ *pop_size*, yang telah melalui operasi *crossover*, dilakukan iterasi pada setiap bitnya. Untuk setiap bit di dalam kromosom $\mathbf{v}_i$, dibangkitkan suatu bilangan riil random antara 0 dan 1. Apabila bilangan riil tersebut lebih kecil dari p_m, maka bit tersebut mengalami mutasi, yakni berubah nilainya dari 1 ke 0 atau sebaliknya berubah dari 0 ke 1. Sedangkan bila bilangan riil acak yang didapat, lebih besar dari p_m, mutasi tidak terjadi pada bit tersebut.

3.6.Algoritma Genetika Secara Keseluruhan

Secara keseluruhan prosedur GA dapat dijelaskan dalam langkah-langkah berikut:

1. Inisialisasi populasi awal **P** dengan ukuran ***pop_size***. Set nomor generasi **i =1.**
2. Hitung nilai *fitness* semua kromosom dalam **P.** Konstruksikan *roulette wheel* dengan untuk populasi tersebut tersebut.
3. Lakukan langkah **4-7** sampai terdapat kromosom baru sejumlah ukuran populasi ***pop_size.***
4. Pilih dua individu **v** dan **w** dari populasi **P** dengan menggunakan *roulette wheel.*
5. Putuskan untuk dua kromosom **v** dan **w** yang terpilih, apakah akan terjadi *crossover* atau tidak. Jika ya, lakukan *crossover* pada dua kromosom tersebut, lalu ambil dua *offspring* baru sebagai keluaran prosedur *crossover*. Jika tidak, ambil kedua kromosom tersebut langsung sebagai keluaran prosedur *crossover*. Misalkan keluaran *crossover* ini berturut-turut dinamakan **v'** dan **w'**
6. Untuk dua kromosom **v'** dan **w'** yang dihasilkan oleh prosedur *crossover*, lakukan mutasi bit demi bit. Namakan hasilnya adalah **v''** dan **w''**
7. Masukkan kromosom **v''** dan **w''** sebagai anggota dari populasi baru.
8. Pilih kromosom-kromosom sejumlah ***pop_size*** dengan nilai *fitness* terbaik dari populasi **P** yang lama dan **P** yang baru. Kemudian jadikan kromosom-kromosom tersebut membentuk populasi **P.** Set jumlah generasi **i = i +1.**
9. Jika jumlah generasi sekarang sudah mencapai ***maxgen***, hentikan proses GA. Algoritma genetika selesai. Kromosom terbaik yang sudah disimpan, dikembalikan sebagai solusi. Jika tidak, kembali ke langkah (2).

4. EKSPERIMEN DAN ANALISA

Eksperimen dilakukan dengan mengukur tingkat pengenalan pada subset data citra yang dimiliki oleh Laboratorium Riset Kecerdasan Komputasional, Fakultas Ilmu Komputer, Universitas Indonesia. Basis data citra yang dipergunakan adalah basisdata citra 3 dimensi wajah 5 orang mahasiswa yang masing-masing diambil dari sudut pandang yang berlainan. Jumlah sudut pandang yang dipergunakan terdiri dari 13 posisi horisontal tanpa ada variasi pada posisi vertikal (sudut elevasi 0^0) di mana masing-masing sudut pandang terdiri dari 4 variasi penampakan. Sudut-sudut horisontalnya adalah sebagai berikut: adalah -90^0, -75^0, -60^0, -45^0, -30^0, -15^0, 0^0, $+15^0$, $+30^0$, $+45^0$, $+60^0$, $+75^0$, $+90^0$. Ukuran dimensi spasial yang dipakai adalah 4096 piksel. Sedangkan total ukuran basisdata citra yang dipakai adalah 260 citra.

Perbandingan dilakukan antara tiga perlakuan. Pertama, metode PCA tanpa menggunakan GA. Kedua, metode PCA dengan menggunakan GA di mana citra acuan GA adalah satu set citra frontal (sudut horisontal 0^0) yang diambil dari basis data citra yang dipakai. Ketiga, metode PCA dengan menggunakan GA di mana citra acuan GA menggunakan set citra yang sama dengan citra pelatihan yang dipakai. Percobaan dijalankan pada tiga konfigurasi dataset seperti tertera pada tabel berikut.

Tabel 1. Konfigurasi dataset untuk eksperimen

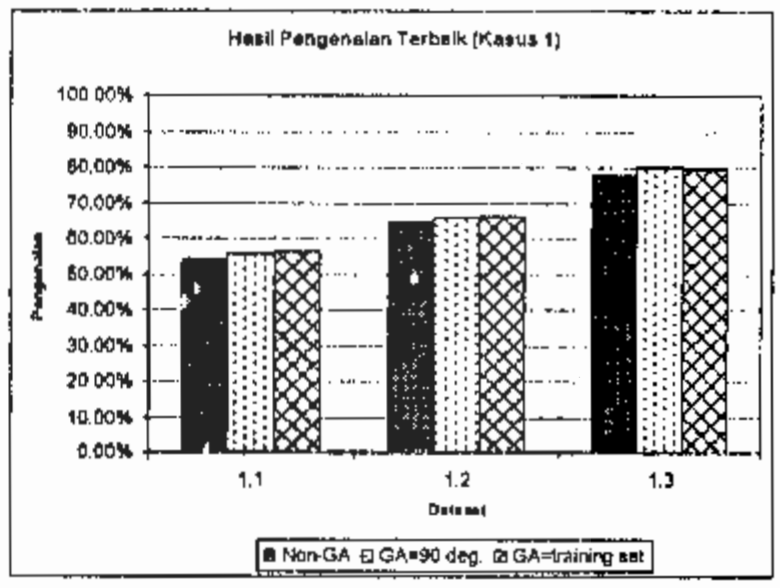
Dataset	Sudut acuan pelatihan	Jumlah citra pelatihan	Sudut acuan pengujian	Jumlah citra pengujian
1	-90, -30, +30, +90	80 (30.8%)	-75, -60, -45, -15, 0, +15, +45, +60, +75	180 (69.2%)
2	-90, -45, 0, +45, +90	100 (38.5%)	-75, -60, -30, -15, +15, +30, +60, +75	160 (61.5%)
3	-90, -60, -30, 0, +30, +60, +90	140 (53.8%)	-75, -45, -15, +15, +45, +75	120 (46.2%)

Tabel 2. Hasil eksperimen untuk ketiga dataset pada kasus pertama

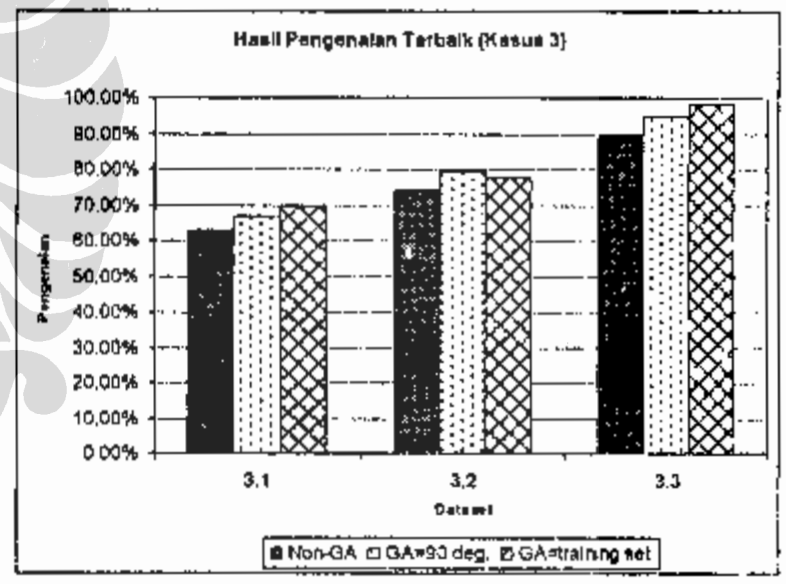
	Tingkat Pengenalan		
	Non-GA	GA-Set I	GA-Set II
Dataset 1	53.89%	55.56%	56.11%
Dataset 2	64.38%	65.63%	66.25%
Dataset 3	77.50%	80.00%	79.17%

Tabel 3 Hasil eksperimen untuk ketiga dataset pada kasus kedua

	Tingkat Pengenalan		
	Non-GA	GA-Set I	GA-Set II
Dataset 1	62.78%	66.67%	69.44%
Dataset 2	74.38%	79.38%	77.50%
Dataset 3	89.17%	95.00%	98.33%



Gambar 1. Grafik tingkat pengenalan untuk hasi. eksperimen sesuai dengan tabel 2.



Gambar 2. Grafik tingkat pengenalan untuk hasil eksperimen sesuai dengan tabel 3.

Percobaan dilakukan pada dua kasus. Pada kasus pertama, data pengujian merupakan himpunan yang sama sekali terpisah dari data pelatihan. Sementara pada kasus kedua, data pelatihan diikutkan sebagai bagian dari data pengujian. Hasil percobaan dapat dilihat pada tabel 2 dan tabel 3 serta gambar 1 dan gambar 2.

Hasil percobaan menunjukkan bahwa pemanfaatan algoritma genetika pada PCA memberikan performa pengenalan yang tidak kalah daripada PCA tanpa algoritma genetika. Hal ini menunjukkan bahwa penggunaan algoritma genetika mampu menggantikan proses *trial and error* yang memakan waktu. Sebagai

perbandingan, untuk perlakuan eksperimen tanpa algoritma genetika, tingkat pengenalan yang tercantum di atas merupakan tingkat pengenalan terbaik dari beberapa kemungkinan nilai parameter proporsi kumulatif. Dalam eksperimen ini dicobakan nilai-nilai proporsi kumulatif dari 50% sampai dengan 99% dengan interval 5%. Tingkat pengenalan terbaik untuk perlakuan eksperimen non-GA, kebanyakan didapatkan pada kisaran 80% sampai dengan 90%. Sementara pada percobaan menggunakan algoritma genetika, *trial* dengan berbagai kemungkinan nilai proporsi kumulatif tidak perlu dilakukan karena sudah secara otomatis dicari sendiri oleh algoritma genetika yang diimplementasikan.

Sedangkan pada tingkat pengenalannya itu sendiri, optimasi dapat dicapai dengan naiknya tingkat pengenalan setelah PCA dikombinasikan dengan algoritma genetika. Meskipun kenaikan performa pengenalan belum dapat dikatakan sangat signifikan, namun demikian hasil percobaan ini sudah cukup untuk menunjukkan bahwa optimasi reduksi dimensi pada PCA dapat dilakukan dengan menggunakan algoritma genetika. Di samping itu, penggunaan citra acuan GA yang berbeda tidak terlalu berpengaruh pada tingkat pengenalan. Oleh sebab itu, tidak menjadi masalah apabila citra acuan GA cukup dipilih dalam ukuran set yang kecil saja namun telah mewakili semua objek yang dilatihkan secara merata. Dari peningkatan tingkat pengenalan tersebut dapat dilihat pula bahwa pada dasarnya algoritma genetika pada reduksi dimensi dalam PCA pada dasarnya melakukan *fine tuning* atas pemilihan parameter proporsi kumulatif eigen yang menjadi dasar reduksi dimensi tersebut.

5. KESIMPULAN

Di dalam makalah ini telah ditunjukkan sebuah metode otomatisasi dan optimasi reduksi dimensi pada transformasi PCA yang dipergunakan dalam sistem pengenalan wajah 3 dimensi. Hasil percobaan menunjukkan bahwa kinerja tingkat pengenalan yang didapatkan dengan memanfaatkan algoritma genetika sebagai alat untuk otomatisasi dan optimasi minimal sama atau lebih baik daripada PCA tanpa menggunakan algoritma genetika. Secara umum peningkatan tingkat pengenalan memang belum terlalu signifikan, karena di sini skema algoritma genetika yang dipakai masih tergolong standar saja dan belum betul-betul dimodifikasi sehingga lebih cocok lagi dengan karakteristik permasalahan sistem pengenalan wajah 3 dimensi yang dicobakan. Namun demikian, hasil ini sangat penting sebagai pijakan awal untuk pengembangan dan studi-studi selanjutnya.

REFERENSI

[1] H.Anton, *Elementary Linear Algebra, 8th Edition*, John Wiley & Sons, Inc., 1999.

[2] P.N. Belhumeur, J.P. Hespanha, and D.J. Kriegman, Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7), 1997, 711 – 720.

[3] R. Brunelli and T. Poggio, Face Recognition: Features versus Templates, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(10), 1993, 1042 – 1052.

[4] I. Craw, N. Costen, T. Kato and S. Akamatsu, How Should We Represent Face for Automatic Recognition?, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(8), 1999, 725 – 736.

[5] D.E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley Publishing Company, Inc., 1989.

[6] B. Kusumoputro dan Herry, Kinerja PNN-Teroptimasi Berbasis Algoritma Genetika Dalam Pengenalan Aroma 2 Campuran, *Prosiding Ilmu Komputer dan Teknologi Informasi III*, 3(1), 2002, 251 – 256.

[7] B. Kusumoputro dan Y. Satria, Konstruksi dan Rekonstruksi Citra Wajah 3D Dalam Ruang Eigen Gabungan Berdimensi Rendah, *Jurnal Ilmu Komputer dan Teknologi Informasi*, 2(2), 2002, 49 – 54.

[8] B. Kusumoputro dan R. Sripomo, Pengembangan Sistem Penentu Sudut Pandang Wajah 3-D Dengan Menggunakan Perhitungan Jarak Terpendek Pada Garis Ciri Dalam Ruang Eigen, *Makara Seri Sains*, 6(2), 2002, 83 – 89.

[9] S.Z. Li and J. Lu, Face Recognition Using the Nearest Feature Line Method, *IEEE Transactions on Neural Networks*, 10(2), 1999, 439 – 443.

[10] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*, Berlin-Heidelberg: Springer-Verlag, 1996.

[11] R. Soelaiman dan B. Kusumoputro, Sistem Pengenalan Wajah Dengan Penerapan Algoritma Genetika Pada Optimasi Basis Proyeksi Metoda Eigenface dan Fisherface, *Prosiding Ilmu Komputer dan Teknologi Informasi III*, 3(1), 2002, 266 – 270.

[12] M. Uenohara and T. Kanade, Use of Fourier and Karhunen-Loeve Decomposition for Fast Pattern Matching With a Large Set of Templates, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(8), 1997, 891 – 897.