IMPLEMENTASI ALGORITMA FRACTAL NEIGHBOUR DISTANCE UNTUK FACE RECOGNITION

Garibaldy W Mukti

13506004
Teknik Informatika ITB
alamat : Srigading 29, Bandung 40132
email: subghost1802000@yahoo.com

ABSTRAK

Teknologi *Face Recognition* (FR) merupakan salah satu teknologi mutakhir yang berfungsi untuk mengenali wajah seseorang. Teknologi ini merupakan salah satu bagian dari teknologi *Biometric Recognition* (pengenal bagian tubuh), diantaranya *Eye Recognition, Retinal Recognition, Voice Recognition,* dll. Teknologi FR ini sering diterapkan untuk sistem keamanan dalam suatu instansi tertentu ataupun proses pendataan penduduk.

Banyak algoritma yang diimplementasi untuk mengembangkan teknologi FR ini, diantaranya *EigenFace*, *FisherFace* (pengembangan dari EigenFace), *Hidden Markov Model* (HMM), *Euclidean*, dan *Fractal Neighbour Distance* (FND). Di antara algoritma-algoritma tersebut, algoritma FND merupakan algoritma yang mirip dengan algoritma *Breadth First Search* (BFS), yaitu mencari solusi dengan memberikan nilai pada tiap simpul calon solusinya.

Penggunaan algoritma FND merupakan cara yang paling efisien dibandingkan dengan algoritma lain dalam proses FR ini karena memberikan toleransi yang tinggi terhadap cahaya dan memberikan tingkat akurasi yang tinggi. Dalam makalah ini akan dibahas mengenai skema algoritma FND, implementasinya, dan perbandingannya dengan algoritma lainnya. Selain itu dicantumkan pula ulasan tentang contoh eksperimen secara langsung proses FR ini.

Kata kunci*:* Fractal Neighbour Distance, Face Recognition, Biometric Recognition, Breadth First Search Algorithm.

1. PENDAHULUAN

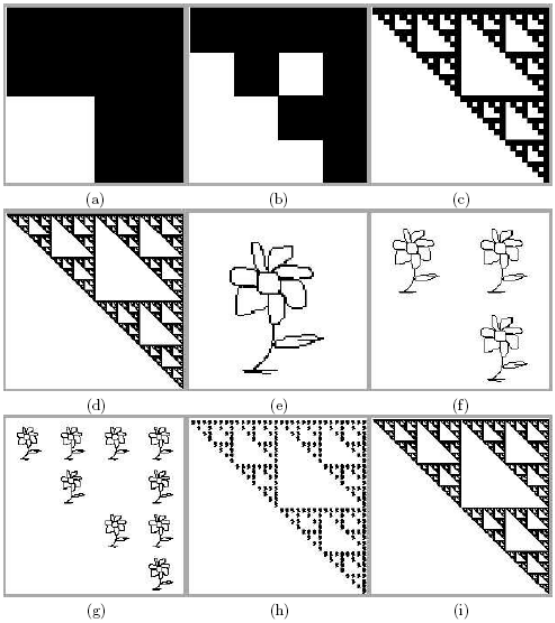
Manusia dapat mengenali suatu objek dengan tingkat akurasi dan kecepatan yang relatif cepat Dilain pihak membuat suatu sistem yang dapat melakukan hal yang sama dengan tingkat akurasi dan kecepatan yang sama dengan manusia merupakan suatu hal yang cukup sulit. Proses yang sebenarnya terjadi dalam otak manusia untuk mengenali suatu objek secara visual hingga saat masih merupakan sebuah misteri dikarenakan kompleksitas otak manusia. Oleh karena itu pebuatan sebuah sistem kecerdasan buatan untuk mengenali suatu objek masih terbatas pada meniru sebagian kecil dari proses kerja otak manusia.

Salah satu algoritma yang telah umum dipakai untuk proses pengenalan wajah manusia adalah *EigenFace*. Algoritma ini dibuat berdasarkan *Principal Components Analysis* (PCA). Salah satu keunggulan dari algoritma ini adalah sistem pengurangan dimensi yang menyebabkan proses pengenalan dapat berjalan dengan sangat cepat. Namun di lain pihak, algoritma ini memiliki kekurangan yaitu ketergantungannya terhadap pergeseran dan intensitas cahaya, yang merupakan sebuah aspek yang dominan dalam tiga komponen dasar PCA. Bila ketiga komponen utama tersebut diabaikan, kecepatan proses akan jadi lebih cepat namun komponen tersebut mungkin memiliki suatu informasi penting dalam proses pengenalan wajah itu sendiri, yang mengakibatkan penurunan tingkat akurasinya. Adapula penggunaan algoritma yang mirip dengan *EigenFace*, yaitu algoritma *Linear Discriminant Analysis* (LDA). Algoritma ini sama-sama menggunakan vektor eigen, namun pada metrik yang tersebar.

2. FRACTAL

Fractal merupakan sebuah himpunan matematis yang menunjukkan kesamaan diri sendiri terhadap semua skala pembesaran. Bagian-bagian dari fractal merupakan suatu generalisasi dari suatu objek. Gambar 2.1 merupakan contoh konstruksi sebuah objek fractal, yang disebut juga Segitiga Sierpinski. Kode fractal ini direpresentasikan dengan tiga proses transformasi gambar yang tergeneralisasi dengan dirinya sendiri. Setiap transformasi memproduksi setengah duplikat dirinya sendiri terhadap tiga lokasi yang berbeda. Hasil dari kode fractal ini adalah sebuah input gambar berwarna hitam (Gambar 2.1 (a)). Pada kelanjutan transformasi didapat sebuah pendekatan gambar yang lebih jelas (Gambar 2.1 (d)), yang akan memiliki hasil yang relatif sama pada iterasi yang lebih lanjut. Sebuah objek suatu kode fractal berbeda-beda tergantung pada kelanjutan kodenya. Sebuah objek yang asli hanya dapat dicapai pada iterasi yang tak hingga, namun pada umunya hanya dalam

beberapa proses iterasi bentuk objek sudah dapat terlihat. Objek itu sendiri sebenarnya tidak bergantung pada gambar awal proses iterasi. Pada gambar 2.1 (e - i) membuktikan bahwa walaupun gambar awalnya berbeda, hasil iterasi hingga tahap ke 4 menghasilkan gambar yang relatif sama.



Gambar 2.1. Iterasi segitiga sierpinski

Dalam proses iterasi tersebut, transformasi harus menggunakan sebuah *attractor* (generalisasi calon objek) sebagai basis perubahannya. Kondisi ini dapat terpenuhi bila transformasi tersebut, yang direpresentasikan dengan *f*, membentuk suatu aturan dalam satuan (***X***,*d*) memenuhi :

$$d(f(\mathbf{x}), f(\mathbf{y})) \leq sd(\mathbf{x}, \mathbf{y})$$

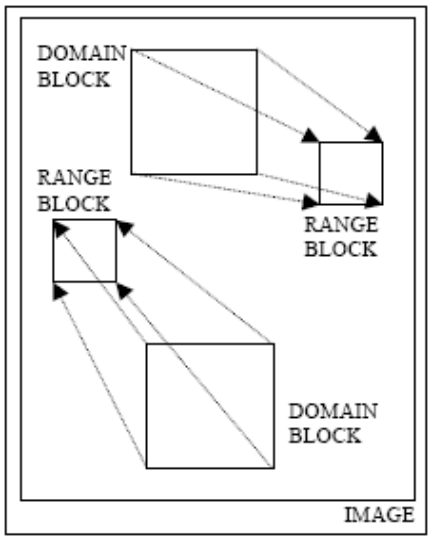
dimana $s < 1$, dan $\boldsymbol{x},\boldsymbol{y} \in \boldsymbol{X}$. ***X*** merupakan sebuah himpunan gambar yang unik, *s* merupakan konstanta faktor *f*, dan *d* merupakan jarak yang dihitung. Pengulangan *f* terhadap tiap elemen ***x*** menghasilkan sebuah *attractor* $A = f(A)$ yang unik.

2. FRACTAL IMAGE CODING

2.1 Konsep Dasar

Pada umumnya untuk menemukan kode fractal yang optimal untuk sembarang gambar merupakan hal yang sangat sulit, kecuali untuk himpunan gambar yang merupakan objek fractal asli dalam ilmu matematika. Namun, ada sebuah teknik untuk mendapatkan kode fractal untuk sembarang gambar, yaitu *Fractal Image Coding* (FIC). FIC yang berbasiskan sistem fungsi iterasi ini ditemukan oleh Barnsley dan Jaquin. Skema kode ini telah sukses dan sering dipakai untuk mengkompresi gambar.

Dalam kode fractal gambar, sebuah gambar dimasukkan kedalam sebuah himpunan persamaan. Persamaan-persamaan ini merupakan persamaan transformasi yang mengubah sebuah sub-gambar, yang disebut sebagai blok domain, menjadi sub-gambar lain, yang disebut sebagai blok range, dan pencarian terhadap pasangan gamabr yang cocok pada blok domain dilakukan terhadap seburuh blok range. Blok domain biasanya lebih besar dibandingkan dengan blok range dan mirip antar satu dengan yang lainnya setelah ditransformasi.



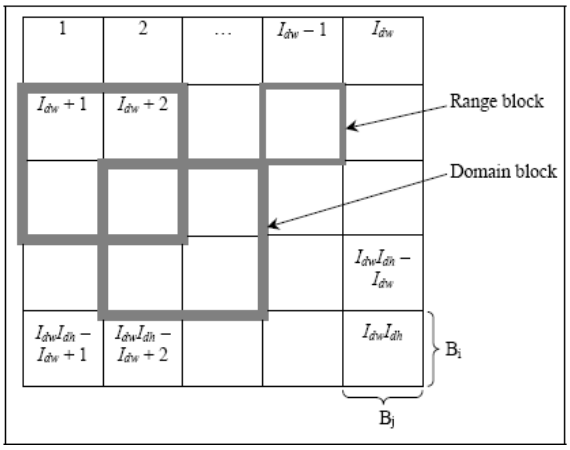
Gambar 2.2. Contoh proses matching antara blok domain dan blok range

Proses pemetaan dari blok domain ke blok range merupakan proses penyamaan blok domain dengan blok range, yaitu mengalikan tiap elemen matriks dalam blok domain dengan suatu konstanta skala α, dan menjumlahkan dengan konstanta iluminasi γ. Lokasi blok domain dan nilai α dan γ dibentuk sedemikian rupa sehingga hasil transformasinya mendekati blok range dengan batasan tertentu, misalnya waktu maksimum pencocokan gambar. Nilai didapat dari penyamaan dynamic range dan nilai rata-rata antara blok domain dan blok range. Dengan menggunakan proses ini secara berulang-ulang dan terus-menerus akan menghasilkan gambar yang mendekati gambar asli.

2.2 Notasi

Misalkan $\tau = \bigcup_{n=1}^{N} \tau_n$ dimana *N* adalah jumlah transformasi bagian dari τ_n yang memetakan blok domain ke blok range. Iterasi τ keseluruhan terhadap transformasi menghasilkan gambar yang mirip dengan gambar aslinya.Rumus transformasi itu sendiri adalah :

$$\tau_n(\mathbf{p}_i) = \alpha_n (\mathbf{D}_{n,r}\mathbf{p}_i^{(\mathbf{nD})}) + \gamma_n$$



Gambar 2.3. Pembagian Gambar Seragam. Besar blok domain 2 kali lebih besar dari blok range

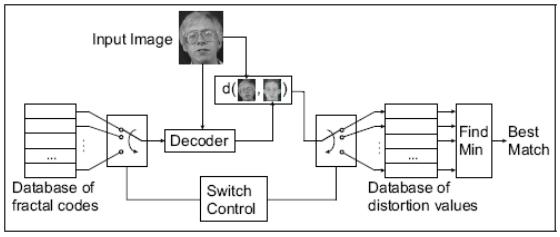
Pada transformasi ke-n, τ_n bekerja pada blok domain dengan input $\mathbf{p}_i$. Blok domain yang digunakan pada transformasi ke-n tersebut dilambangkan dengan $\mathbf{p}_i^{(\mathbf{nD})}$. Sebuah hasil desimal berdasarkan faktor saring r dilambangkan dengan $\mathbf{D}_{n,r}$, dan diikuti dengan faktor skala α_n, serta faktor penukaran iluminasi γ_n. Dalam contoh diatas, blok domain dan blok range merupakan bagian gambar yang berbentuk persegi. Sebagai tambahan, blok range tidak akan keluar dari batas sehingga $\tau_n(\mathbf{p}_i) \cap \tau_m(\mathbf{p}_i) = \varnothing$, untuk $m \neq n$. Gambar 2.3 diatas mengilustrasikan pembagian gambar menjadi beberapa blok range dengan beberapa kemungkinan blok lain.Ilustrasi diatas juga menunjukkan bahwa blok domain mengandung sub-blok yang besarnya sama dengan blok range.

3. FACE RECOGNITION

3.1 Fractal Neighbour Distance

Proses FR sebenarnya adalah menyamakan sebuah gambar yang belum teridentifikasi dengan salah satu gambar yang ada pada database. Dalam proses FR ini akan didapat sebuah tingkat kesalahan yang dilambangkan dengan F_{ERR}/F_N, dimana F_{ERR} adalah jumlah ketidaksamaan dan F_N adalah jumlah input gamabr yang dipakai. Kemudian input gambar tersebut diterima atau ditolak berdasarkan algoritma yang dibentuk dari identitas yang ada pada database.

FND memberikan perhitungan kuantitatif terhadap karakteristik input-output kode fractal suatu gambar. Jadi tiap hasil kode fractalnya dibandingkan secara langsung dengan database tanpa kecuali.



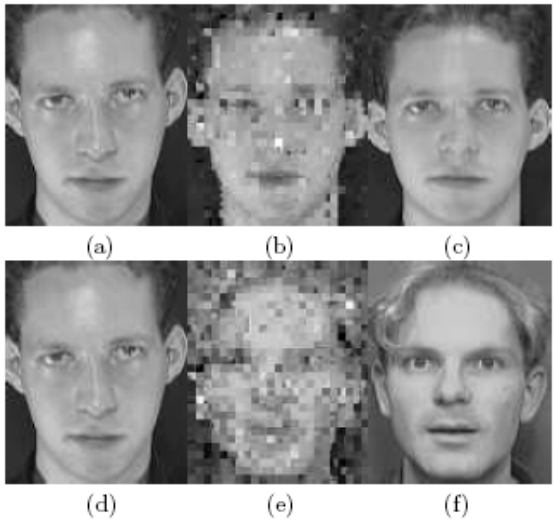
Gambar 3.1. Diagram Blok Sistem FR Berdasarkan FND

Dari gambar diatas kita bisa mengetahui proses FR dengan menggunakan FND.

3.2 Algoritma

Secara umum, algoritma FND adalah sebagai berikut :

- Bentuk suatu database yang berisi kode fractal yang merepresentasikan himpunan contoh wajah
- Tentukan suatu input wajah yang belum dikenal $\mathbf{p}_i$, dimana $1 \leq i \leq V$, dimana V adalah jumlah input wajah
- Untuk setiap kode fractal f_j dari database dimana $1 \leq j \leq W$, dimana W adalah jumlah contoh wajah, tentukan $d(f_j(\mathbf{p}_i), \mathbf{p}_i)$
- Hasil wajah yang paling cocok dari database adalah yang menghasilkan $d(f_j(\mathbf{p}_i), \mathbf{p}_i)$ paling minimum



Gambar 3.2. Hasil pengenalan wajah dengan menggunakan 2 kode fractal yang berbeda

Salah satu dari penggunaan kode fractal pada gambar diatas merupakan kode fractal pada kelas yang sama dengan input gambar (Gambar 3.2 (a) dan Gambar 3.2 (d)). Hal ini menunjukkan bahwa interpretasi kode dengan menggunakan kelas yang sama akan menghasilkan gambar yang lebih mirip. Singkatnya, gamabr diatas adalah hasil interpretasi input gambar dengan menggunakan kode fractal dari 2 gambar yang berbeda. Bila kodenya terdapat pada kelas yang

berbeda, maka nilai $d(f_j(\mathbf{p}_i), \mathbf{p}_i)$ akan besar, sehingga gambar tersebut dinyatakan berbeda.

Adapun rincian proses algoritmanya adalah seperti algoritma BFS yang mencari solusi elemen matriks yang samaantara input gambar dengan gambar pada database. Setiap elemen matriks yang merupakan informasi byte pixel pada gambar dibandingkan dengan menggunakan konstanta tertentu yang didapat dari tiap contoh gambar pada database.

Algoritma sederhana pada BFS berupa :

$$c = f + g$$

dimana c merupakan nilai cost simpul, f merupakan langkah yang diambil, dan g merupakan jarak asli tanpa ada batasan antara simpul awal dan simpul akhir.

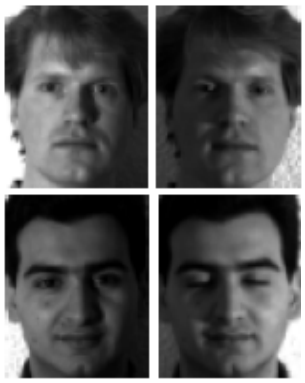
Seperti yang telah dijelaskan dalam Gambar 2.3, setiap elemen sub-gambar (blok domain) dibandingkan dengan blok range, dengan memberikan suatu cost tertentu yaitu perbedaan iluminasi dan skala. Bila menggunakan dasar dari BFS, akan terbentuk rumus :

$$c = fB + g + h$$

dimana c merupakan nilai cost simpul, f merupakan konstanta skala, g adalah konstanta iluminasi, dan B adalah nilai byte pixel. Setiap nilai tersebut akan disimpan pada saat ia menemukan posisi pixel yang seharusnya (berdasar pada database), kemudian dibandingkan dengan seluruh database yang kemudian bisa diketahui gambar mana saja yang cocok / mirip dengan input gambar.

3.3 Shifting

Proses shifting (pertukaran posisi) pixel juga diperhitungkan dalam proses FR ini, karena setipa input gambar belum tentu posisinya sesuai dengan gambar pada database. Bila hal ini tidak diperhitungkan, maka tingkat kesalahan pengenalan akan lebih besar dan data yang dihasilkan tidak akan akurat.



Gambar 3.2. Contoh perbedaan cahaya dan ekspresi

Sebagai contoh, posisi pengambilan gambar wajah seseorang pada saat proses FR lebih sering berbeda, walaupun hanya 1 pixel. Hal ini dapat terjadi saat proses pemindaian gambar ataupun ekspresi saat pengambilan gambar tersebut berlangsung. Perubahan sebesar 1 pixel tersebut dapat berdampak besar pada saat proses FR itu sendiri, karena setiap pixel dalam gambar akan diperhitungkan. Selain itu juga terdapat iluminasi cahaya yang berbeda. Shifting itu sendiri memiliki rumus :

$$\tau_n(\mathbf{p}_i) = \min((\alpha_n(\mathbf{D}_{n,r}\mathbf{p}_i^{(\mathrm{nD})}) + \gamma_n)^{right}, (\alpha_n(\mathbf{D}_{n,r}\mathbf{p}_i^{(\mathrm{nD})}) + \gamma_n)^{top}, (\alpha_n(\mathbf{D}_{n,r}\mathbf{p}_i^{(\mathrm{nD})}) + \gamma_n)^{down}, (\alpha_n(\mathbf{D}_{n,r}\mathbf{p}_i^{(\mathrm{nD})}) + \gamma_n)^{left})$$

Rumus diatas memberikan nilai minimum dari 4 percabangan arah yang ada, yaitu atas, bawah, kiri, dan kanan. Rumus ini bisa menghasilkan nilai toleransi pada saat proses FR berlangsung, sedangkan proses tiap arahnya sama seperti yang dijelaskan sebelumnya pada Bab 4.

Sebenarnya pergeseran pixel tersebut tidak terjadi hanya dalam 4 arah (atas, bawah, kiri, dan kanan), namun bisa pula terjadi seperti pada arah angin (barat laut, timur laut, tenggara, dan barat daya), namun hal tersebut tidak diperhitungkan karena dapat dicari dengan mengubah konstanta γ_n ataupun dengan membuat blok domain yang beririsan satu sama lain tetapi tidak merupakan himpunan bagian.

4. PENUTUP

4.1 Kesimpulan

Proses Face Recognition dengan menggunakan FND merupakan salah satu cara untuk mengidentifikasi suatu input gambar secara sembarang dengan data wajah yang ada dalam database. FND memberikan hasil yang memiliki tingkat akurasi yang tinggi dalam proses identifikasinya karena ia membandingkan informasi byte pixel secara fractal, yaitu sebagian-sebagian, sehingga bisa didapat tingkat akurasi yang tinggi.

Penggunaan FND ini sendiri merupakan pengembangan dari BFS yang mencari lokasi pixel yang sama dengan suatu cost tertentu. Tingkat kesamaan yang diperoleh merupakan hasil pencocokan informasi byte pixel yang didapat dari hasil pemindaian dengan informasi byte pixel gambar yang telah ada dari database.

Dalam proses pemindaiannya itu sendiri dapat terjadi beberapa kesalahan, yaitu posisi pengambilan gambar, ekspresi wajah, dan intensitas cahaya saat pengambilan gambar. Oleh karena itu pada algoritmanya perlu ditambahkan sebuah toleransi yaitu

toleransi posisi dan toleransi cahaya. Kedua hal tersebut diperhitungkan sedemikian rupa agar hasil proses FR memiliki tingkat akurasi yang tinggi.

4.2 Saran

Penerapan FND ini masih belum terlalu dipakai oleh berbagai aplikasi karena pada umumnya dengan menggunakan algoritma EigenFace proses FR sudah bisa dilakukan. Hal ini ditunjang oleh alat pindai gambar yang baik yang memberikan intensitas cahaya yang konstan dan gambar yang relatif tidak memiliki pergeseran pixel. Namun dengan menggunakan FND, tidak diperlukan alat pindai yang tingkat presisinya sangat tinggi sehingga harganya bisa lebih murah.

Jadi sebaiknya sibuat sebuah software yang dapat mengimplementasi algoritma FND dalam proses FR agar penggunaannya bisa lebih luas.

REFERENSI

[1] S. A. Rizvi, P. J. Phillips, and H. Moon. A verification protocol and statistical performance analysis for face recognition algorithms. *IEEE Conf. Computer Vision and Pattern Recognition*, pages 833–838, June 1998.

[2] B. Mandelbrot. *The fractal geometry of nature*. Freeman, San Francisco, CA, 1982.

[3] Tan, Taewoon. *Human Face Recognition Based on Fractal Image Coding.* Freeman, San Francisco, CA, 1982.