MATCHING PADA AUTENTIKASI SIDIK JARI MENGGUNAKAN ALGORITMA *HOUGH TRANSFORM*

Agus Bejo[1], Timotheus Steven Gunawan[2], Risanuri Hidayat[3]

Abstract*— Nowadays, authentication using fingerprint is quite popular. One of the challenges in creating a qualified fingerprint authentication is in the matching process. There are several methods to do the matching process, and the most popular one is the hough transform technique. This technique provides good accuracy but requires a lot of computation time. Long computation time would be a problem because the authentication system will be unreliable. This research is aimed to reduce the computation time by varying step size. The step size will be perform in the rotation and translation process and aim to see the impact on accuracy and computation time. By increasing the step size, hopefully, the computation time will decrease without lowering the accuracy. The result shows that there is a decrease in the computation time but it also decreases the accuracy. To solve the decrase in accuracy due to step size variation, a modification on the hough transform technique by adding the ridge shape feature as the criteria of matching is proposed. With employing ridge shape feature, it increases the accuracy without increasing the computation time.

***Intisari*— Autentikasi menggunakan sidik jari merupakan teknik autentikasi yang sangat populer belakangan ini. Salah satu tantangan dalam mewujudkan autentikasi sidik jari yang mumpuni adalah pada proses *matching*. Terdapat banyak teknik untuk melakukan *matching*, dan yang paling populer adalah teknik *hough transform*. Teknik ini memiliki tingkat akurasi yang baik tetapi memerlukan waktu komputasi yang cukup lama. Waktu komputasi yang lama akan menjadi masalah karena sistem autentikasi akan menjadi tidak nyaman digunakan oleh *user*. Penelitian ini dilakukan untuk mengurangi waktu komputasi dari teknik *hough transform* dengan melakukan variasi *step size*. Variasi *step size* akan dilakukan pada proses translasi dan rotasi, dan akan dilihat pengaruhnya terhadap akurasi dan waktu komputasi. Harapannya, dengan dilakukan variasi *step size* maka waktu komputasi akan turun tanpa menurunkan akurasi. Hasil penelitian menunjukan adanya penurunan waktu komputasi tetapi juga menurunan akurasi. Untuk mengatasi turunya akurasi akibat variasi *step size* maka, pada penelitian ini juga akan dilakukan modifikasi algoritma *hough transform* dengan menambahakan fitur *ridge shape* sebagai kriteria kecocokan. Dengan penambahan fitur ridge shape ternyata terjadi peningkatan akurasi *matching* tanpa mempengaruhi waktu komputasi.**

***Kata Kunci*— *fingerprint matching*, *hough transform*, *step size*, fitur *ridge shape*.**

[1, 3] *Dosen, Jurusan Teknik Elektro dan Teknologi Informasi Fakultas Teknik Universitas Gadjah Mada, Jln. Grafika No. 2 55281 INDONESIA telp: 0274-552305; fax: 0274-552305; e-mail: agusbj@ugm.ac.id)*

[2]*Mahasiswa, Jurusan Teknik Elektro dan Teknologi Informasi Fakultas Teknik Universitas Gadjah Mada, Jln. Grafika No. 2 55281 INDONESIA (telp: 0274-552305; fax: 0274-552305; e-mail: timo.gunwan@gmail.com)*

I. PENDAHULUAN

Tren autentikasi yang sedang berkembang dan populer belakangan ini adalah autentikasi menggunakan karakteristik biometrik. Salah satu keunggulan dari biometrik dibanding metode lain adalalah *user* tidak perlu mengingat atau membawa apapun untuk melakukan autentikasi sehingga tidak akan ada lagi alasan lupa atau ketinggalan. Keunggulan lain adalah tingkat keamanan yang lebih baik dibanding metode-metode konvensional yang menggunakan seuatu yang user tahu dan miliki [1]. Karena informasi biometrik user tidak mudah diduplikat ataupun dicuri oleh orang lain.

Salah satu karakteristik biometrik yang sangat berkembang dan mulai dipakai secara komersial adalah sidik jari. Penggunaan sidik jari dalam proses autentikasi menjadi populer karena pengaplikasiannya tidak sulit dan dari segi harga juga terjangkau [2]. Teknologi yang ada sekarang ini juga membuat proses pengambilan informasi sidik jari menjadi mudah. Secara keseluruhan, autentikasi menggunakan sidik jari merupakan metode yang tingkat keamanannya baik, performa (akurasi, waktu eksekusi, penggunaan memori, ketahanan) yang juga baik dan dari segi harga juga terjangkau [3].

Salah satu tantangan dalam mewujudkan sistem autentikasi sidik jari yang mumpuni adalah pada proses *matching*. Proses *matching* merupakan proses yang sangat penting dari keseluruhan proses autentikasi. Jika algoritma *matching* yang digunakan memiliki performa yang buruk maka bisa terjadi kesalahan pada keputusan yang dibuat misalkan kesalahan sistem dimana sistem memutuskan bahwa seseorang adalah asli, padahal tidak. Teknik dalam melakukan *matching* ada berbagai macam. Salah satu teknik yang paling populer adalah teknik *matching* menggunakan *hough transform*. *Hough transform* merupakan teknik *matching* yang memiliki konsep mencari nilai kemiripan optimal dari semua peluang terjadinya rotasi maupun translasi pada gambar masukan sidik jari. Teknik ini memanfaatkan *minutia* data sebagai subjek *matching*. Artinya informasi yang diambil dari citra sidik jari dan informasi yang dicocokan (*matching*) adalah *minutia* data. Teknik *hough transform* merupakan teknik yang menitikberatkan pada akurasi *matching*. Untuk mendapat akurasi yang sangat baik biasanya diperlukan komputasi yang sangat banyak dan kompleks. Efeknya adalah proses *matching* akan memerlukan waktu komputasi yang panjang dan membutuhkan memori yang banyak pula.

Pada makalah ini akan dilakukan variasi *step size* sebagai sebuah cara untuk mengurangi waktu komputasi pada algoritma *matching* menggunakan teknik *hough transform*. Konsekuensi dari variasi *step size* adalah terjadinya penurunan

tingkat akurasi. Untuk memperbaiki tingkat akurasi diperkenalkan fitur informasi sidik jari baru yaitu *minutia ridge shape*. Dengan menggunakan teknik *hough transform* yang telah dimodifikasi diatas diarapkan kecepatan komputasi algoritma *matching* dapat berkurang namun tingkat akurasi dapat dipertahankan agar tidak mengalami penurunan yang signifikan.

II. *FINGERPRINT MATCHING*

A. Fitur Sidik Jari

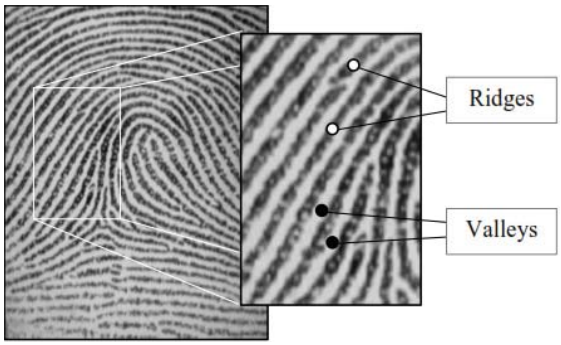
Pada sidik jari dikenal dua unsur pembentuk sidik jari yaitu r*idges* dan *valley*. *Ridges* merupakan guratan-guratan yang membentuk suatu pola tertentu pada sidik jari, sedangkan *valley* merupakan daerah diantara guratan-guratan tersebut. Pada Gbr 1 dapat dilihat *ridges* digambarkan dengan garis warna hitam dan *valley* digambarkan dengan garis putih [4].

Pola-pola khusus yang terbentuk dari *ridge* dan *valley* pada level lokal disebut dengan *minutia*. *Minutia* dapat diartikan sebagai detail-detail kecil. Dalam konteks sidik jari, *minutia* berarti berbagai macam cara *ridge* dapat berubah dan membentuk pola-pola tertentu. Sebagai contoh *ridge* dapat seketika berakhir (*ridge ending*) atau *ridge* dapat seketika terbagi menjadi dua (*birufication*). Tidak semua *minutia* dipakali dalam proses *matching*. Terdapat dua *minutia* yang sering dipakai dalam proses *matching* yaitu *ridge* ending dan *birufication*. *Ridge ending* merupakan pola dimana *ridge* berakhir. Sedangkan *birufication* merupakan pola dimana *ridge* bercabang menjadi dua [3].

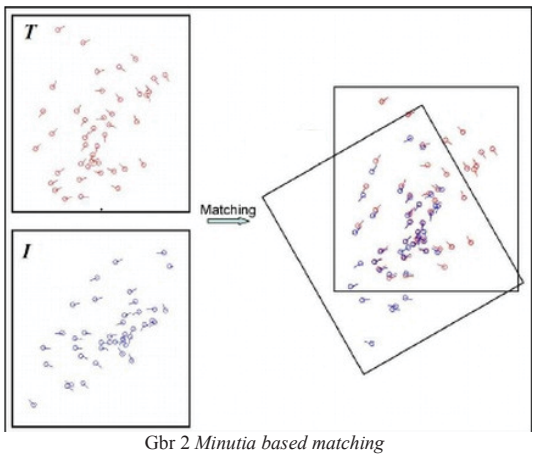
Secara umum, algoritma *fingerprint matching* dapat diklasifikasikan menjadi tiga kelas, yaitu: *Correlation-based matching, Minutia-based matching*, dan *Non-minutia feature-based matching*. Pada *Correlation-based matching*, dua buah citra sidik jari saling di tumpuk dan dihitung korelasi antar piksel yang bertumpuk untuk berbagai macam posisi. *Minutia-based matching* pada dasarnya adalah mencari alignment antara *minutia template* dan *minutia input* yang akan menghasilkan nilai maksimum pada pencocokan *minutia*. *Non-minutia feature-based matching* pada dasarnya hampir sama dengan *minutia-based matching*, bedanya adalah pembanding yang digunakan bukan *minutia* melainkan pola *ridge* (contoh: orientasi lokal dan frekuensi, bentuk *ridge*, informasi tekstur) [5].

B. Minutia-Based Matching dengan Teknik Hough Transform

Pada metode ini, pertama-tama *input* dan *template* di sesuaikan posisinya guna untuk mendapatkan jumlah terbanyak dari *minutia* yang cocok dan saling tumpang-tindih. Transformasi geometri, translasi dan rotasi, digunakan pada proses ini. Selanjutnya dihitung nilai kecocokannya. Terakhir, nilai kecocokan di bandingkan dengan nilai batas untuk menentukan apakah *input* dan *template* cocok atau tidak. Gbr 2 memperlihatkan gambaran tentang proses *matching* pada metode *minutia-based matching* [6].



Gbr 1 Citra sidik jari



Gbr 2 *Minutia based matching*

Salah satu metode pendekatan yang paling populer pada metode *minutia-based matching* adalah menggunakan metode *hough transform*. Ide dari teknik ini adalah mengubah *point pattern matching* menjadi sebuah masalah dalam menemukan puncak tertinggi di bidang *hough transform*. Pada algoritma *matching*, teknik *hough transform* digunakan untuk mencari skor maksimum yang mungkin didapat dari suatu proses *matching*.

Didefinisikan m sebagai vektor yang berisi ciri dari suatu *minutia*, yaitu posisi (koordinat x dan y), sudut orientasi (θ).

$$m = (m_x, m_y, m_\theta) \qquad (1)$$

M didefinisikan sebagai kumpulan dari semua ciri *minutia*. *Template* dilambangkan dengan T dan *input* dilambangkan dengan I. T dan I didefinisikan sebagai :

$$\boldsymbol{T} = (\boldsymbol{t_1}, \boldsymbol{t_2}, \dots, \boldsymbol{t_m} | \boldsymbol{t_i} \in M, i = 1..m) \qquad (2)$$

$$\mathbf{I} = (\boldsymbol{s_1}, \boldsymbol{s_2}, \dots, \boldsymbol{s_n} | \boldsymbol{s_j} \in M, j = 1..n) \qquad (3)$$

Didefinisikan $F_{\Delta x \Delta y \Delta \theta}$ sebagai fungsi yang memetakan vektor *minutia* $\boldsymbol{s_j}$ menjadi $\boldsymbol{s'_j}$ menggunakan transformasi geometri. Transformasi geometri yang digunakan adalah pergeseran$(\Delta x, \Delta y)$ dan rotasi berlawanan arah jarum jam dari titik origin $(\Delta\theta)$. Selanjutnya $\boldsymbol{s'_j}$ dapat di rumuskan menjadi :

$$\begin{bmatrix} s'^j_x \\ s'^j_y \\ s'^j_\theta \end{bmatrix} = \begin{bmatrix} cos\Delta\theta & -sin\Delta\theta & 0 \\ sin\Delta\theta & cos\Delta\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} s^j_x \\ s^j_y \\ s^j_\theta \end{bmatrix} + \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta\theta \end{bmatrix} \qquad (4)$$

Sepasang vektor *minutia* ($\mathbf{(t_j, s'_j)}$) dapat dikatakan cocok jika spasial distance(sD) antara T dan I lebih kecil dari toleransi r_0 dan direction difference (dD) antara T dan I lebih kecil dari toleransi θ_0. Kedua kriteria tersebut disebut dengan kriteria kecocokan, yang dirumuskan sebagai berikut.

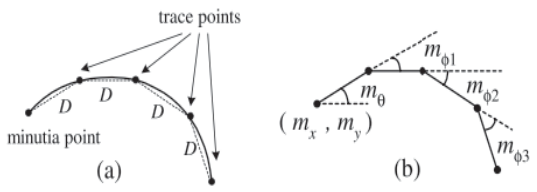
$$sD(t_i, s'_j) = \sqrt{(t_x^i - s'^j_x)^2 + (t_y^i - s'^j_y)^2} \quad (5)$$

$$dD(t_i, s'_j) = min(|t_\theta^i - s'^j_\theta|, 360^o - |t_\theta^i - s'^j_\theta|) \quad (6)$$

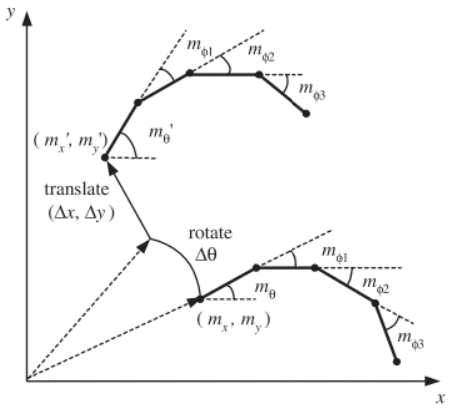
Pada teknik hough transform, dibangkitkan suatu larik yang akan digunakan sebagai tempat untuk mengumpulkan jumlah *minutia* yang *match* dari setiap transformasi dan disebut dengan larik akumulator$A[p, q, r]$. Larik akumulator ini berisi jumlah *minutia* yang *match* dari suatu transformasi $(\Delta x, \Delta y, \Delta\theta)$. Untuk setiap pasangan$(t_i, s_j)$ akan dicari semua kemungkinan transformasi yang memetakan s_j menjadi s'_j. Nilai kecocokan akan di tambahkan dengan 1 jika kriteria $sD(t_i, s'_j) \leq r_0$ dan $dD(t_i, s'_j) \leq \theta_0$ terpenuhi. Penambahan dengan 1 ini merupakan tanda bahwa *minutia input* yang sedang di lakukan proses *matching match* dengan *template*. Data transformasi $(\Delta x, \Delta y, \Delta\theta)$ akan disimpan di dalam larik akumulator ini [5]. Setelah semua pasangan I danT telah di pasangkan, data dengan jumlah *minutia* yang paling banyak *match*-nya akan merepresentasikan transformasi terbaik yang munkin terjadi pada pasangan I dan T.

C. Fitur Ridge Shape

Minutia ridge shape-based *matching* merupakan pengembangan dari metode *minutia*-based *matching*. Pada metode ini dikenalkan satu feature baru yaitu *minutia* ridge shape. *Minutia* ridge shape sendiri merupakan feature yang terdiri dari *minutia* point yang digunakan pada metode *minutia*-based *matching* dan ridge shape. Seperti yang terlihat pada Gbr 3(a), *minutia* ridge shape dapat di perkirakan dengan menarik garis linear sepanjang D. Garis linear ini menghubungkan *minutia* point dengan 4 titik telusur yang merupakan reperesentasi dari ridge shape. Koordinat lokasi dari *minutia* point dan 4 titik representasi dari ridge shape dapat merepresentasikan bentuk dari *minutia* ridge. Untuk merepresentasikan bentuk dari *minutia* ridge, didefinisikan larik $\boldsymbol{m} = (m_x, m_y, m_\theta, m_{\phi1}, m_{\phi2}, m_{\phi3})$ seperti yang ditunjukan pada Gbr 3(b). *Minutia* ridge shape direpresentasikan dengan larik *minutia* point feature (m_x, m_y, m_θ) dan perbedaan orientasi dari dua garis linear berdekatan $(m_{\phi1}, m_{\phi2}, m_{\phi3})$. Gbr 4 menunjukan bahwa parameter *ridge shape* tidak dipengaruhi oleh translasi dan rotasi. Artinya nilai $m_{\phi1}, m_{\phi2}, m_{\phi3}$ tidak akan berubah saat dilakukan transalasi dan atau rotasi [5].



Gbr 3 Fitur *minutia ridge shape*



Gbr 4 Pengaruh rotasi dan translasi pada fitur *ridge shape*

Fitur *ridge shape* akan digunakan sebagai kriteria kecocokan pada algortima *matching*. Algoritma *matching* menggunakan teknik *hough transform* akan dimodifikasi dengan menambahkan kriteria kecocokan yang berasal dari fitur *ridge shape*. ϕ_1 *input* akan dibandingkan dengan ϕ_1 *template*, ϕ_2 *input* akan dibandingkan dengan ϕ_2 *template* dan ϕ_3 *input* akan dibandingkan dengan ϕ_3 *template*. Setiap ϕ yang memiliki nilai yang sama akan membuat skor *matching* bertambah.

III. PERANCANGAN PROGRAM

Gbr 5 Blok diagram perancangan program

Perancangan program dibagi menjadi blok-blok kecil seperti yang terlihat pada Gbr 5. Pertama-tama *feature set input* dan *template* yang berisi informasi *minutia* suatu sidik

jari dibaca oleh program. Selanjutnya dilakukan transformasi geometri berupa rotasi dan translasi menggunakankan persamaan (4) kepada *minutia input*. Setelah itu dilakukan *match* untuk setiap *minutia input* dengan *minutia template.* pada proses ini akan dihitung skor *matching. Minutia* akan dianggap *match* jika memenuhi kriteria pada persamaan (5) dan (6). Selanjutnya dari semua kemungkinan rotasi dan translasi, akan dipilih skor yang paling besar. Setelah itu akan dilakukan normalisasi dan terakhir akan dilakukan pengambilan keputusan berupa *match* atau *non-match*.

IV. HASIL DAN PEMBAHASAN

Program *matching* dibuat kemudian di ujikan menggunakan database A. Pada database A terdapat terdapat 4 buah data sidik jari yang diambil pada jari berbeda dan masing-masing diambil sebanyak 6 kali. Sehingga jumlah total data sidik jari pada database A berjumlah 24. Dengan membandingkan semua kemungkinan *matching* antara *input* dan *template*, terdapat 84 pasangan *genuine* dan 216 pasangan *impostor*.

Pengujian yang dilakukan adalah pengujian pengaruh *step size* dan pengaruh penambahan fitur *ridge shape* terhadap waktu komputasi dan akurasi yang direpresentasikan lewat FRR dan FAR serta EER.

A. Pengaruh Step size terhadap Waktu Komputasi

Pengujian ini dilakukan dengan melakukan *matching* antara dua sidik jari dan mengubah-ubah nilai *step size*nya. Tabel 1 memperlihatkan hasil pengaruh pengujian *step size* terhadap waktu komputasi. Dapat dilihat bahwa terjadi penurunan waktu komputasi saat digunakan *step size* yang semakin besar. Waktu komputasi akan menjadi $\frac{t}{q^2}$, dimana t adalah waktu komputasi saat *step size* 1dan q adalah nilai *step size* yang digunakan.

TABEL 1 PENGARUH *STEP SIZE* TERHADAP WAKTU KOMPUTASI

Database A	input	template
	HCL0000	HCL0000
step size	Waktu Komputasi (detik)	
1	9,735326	
2	2,356162	
4	0,60177	
8	0,14766	
16	0,037519	

B. Pengaruh Step size terhadap FRR dan FAR

Dapat dilihat pada Tabel 2, semakin besar *step size* maka semakin besar pula nilai FRRnya. Hal ini disebabkan karena penurunan skor hasil *matching*. FRR merupakan perbandingan jumlah *genuine match* yang *"falsely rejected"* dengan jumlah total *genuine match*. Karena terjadi penurunan skor, maka jumlah *genuine match* yang nilainya dibawah nilai *threshold* (*falsely rejected*) akan menjadi lebih banyak saat digunakan *step size* yang lebih besar. Pengaruhnya adalah nilai FRR pada *threshold* akan menjadi lebih besar saat digunakan *step size* yang lebih besar. Nilai FRR yang besar menunjukan bahwa banyak *matching* yang seharusnya dianggap *match* (*accepted*) tetapi dianggap tidak *match* (*rejected*). Oleh karena itu nilai FRR yang besar menunjukan bahwa akurasi dari algoritma matching rendah.

Pada Tabel 2 juga terlihat terlihat bahwa semakin besar *step size* yang digunakan maka FARnya semakin kecil. FAR merupakan perbandingan jumlah *impostor match* yang *"falsely accepted"* dengan jumlah total *impostor match*. Karena terjadi penurunan skor akibat *step size*, maka jumlah *impostor match* yang nilainya diatas *threshold* (*falsely accepted*) akan berkurang. Pengaruhnya adalah nilai FAR pada *threshold* akan mengecil saat digunakan *step size* yang semakin besar. Nilai FAR yang kecil menunjukan bahwa hanya ada sedikit *matching* yang seharusnya dianggap tidak *match* (*rejected*) tetapi dianggap *match* (*accepted*). Oleh karena itu nilai FAR yang kecil menunjukan bahwa akurasi dari algoritma matching tinggi.

TABEL 2 PENGARUH *STEP SIZE* TERHADAP FRR DAN FAR

Database A		Threshold= 10
step size	FRR	FAR
1	1,190476	0,462963
2	1,190476	0,462963
4	2,380952	0,462963
8	8,333333	0
16	73,809524	0

C. Pengaruh Step size terhadap EER

EER merupakan titik dimana FRR dan FAR berpotongan. Tabel 3 memperlihatkan pengaruh *step size* terhadap EER. Secara umum semakin besar *step size* yang digunakan maka EER akan semakin besar. Hal ini disebabkan karena *step size* menurunkan skor hasil *matching*. Jika skor hasil *matching* turun, maka terdapat kemungkinan *genuine matching* yang awalnya di *accept*, setelah dilakukan kuantasi maka akan di rejected (*falsely rejected*). Kenaikan EER menyebabkan akurasi *matching* memburuk

TABEL 3 PENGARUH *STEP SIZE* TERHADAP EER

Database A	
step size	EER
1	0,82672
2	0,82672
4	1,058201
8	2,612434
16	18,948413

D. Pengaruh Penambahan Fitur Ridge Shape terhadap Waktu Komputasi

Tabel 4 memperlihatkan hasil pengaruh penambahan fitur ridge shape terhadap waktu komputasi. Dapat dilihat bahwa penambahan fitur *ridge shape* hampir tidak ada efeknya terhadap waktu komputasi. Hal ini dikarenakan algoritma *ridge shape* yang ditambahkan sangat sederhana dan hanya membuthkan sangat sedikit waktu untuk mengeksekusinya.

TABEL 4 PENGARUH PENAMBAHAN FITUR *RIDGE SHAPE* TERHADAP WAKTU KOMPUTASI

Database A	input	template
	HCL0000	HCL0000
step size	Waktu Komputasi (detik)	Waktu Komputasi dengan Ridge Shape (detik)
1	9,40519	9,471846
2	2,370838	2,49391
4	0,583352	0,585811
8	0,151511	0,15445
16	0,037868	0,038675

E. Pengaruh Penambahan Fitur Ridge Shape terhadap FRR dan FAR

Penambahan fitur *ridge shape* berpengaruh kepada bertambahnya skor hasil matching baik *genuine match* maupun *impostor match*. Tabel 5 memperlihatkan pengaruh penambahan *fitur ridge shape* pada *step size* tertentu terhadap FRR dan FAR pada *threshold* 10. Dapat dilihat bahwa penambahan fitur *ridge shape* akan menurunkan nilai FRR untuk *step size* apapun. Hal ini disebabkan karena terjadi kenaikan skor hasil *matching* pada *genuine match* sehingga jumlah *genuine match* yang nilainya dibawah nilai *threshold* (*falsely rejected*) akan berkurang dan menyebabkan nilai FRR pada *threshold* tersebut mengecil. Untuk FAR, terjadi kenaikan nilai FAR ketika ditambahkan *fitur ridge shape* sebagai kriteria kecocokan. Hal ini terjadi karena jumlah *impostor match* yang nilainya lebih dari threshold (*falsely accepted*) semakin banyak dikarenakan adanya penambahan skor dari penggunaan fitur *ridge shape*

TABEL 5 PENGARUH PENAMBAHAN FITUR *RIDGE SHAPE* TERHADAP FAR DAN FRR

Database A				Threshold= 10
step size	FRR	FAR	FRR dengan Ridge shape	FAR dengan Ridge Shape
1	1,190476	0,462963	0	0,925926
2	1,190476	0,462963	0	0,925926
4	2,380952	0,462963	0	0,462963
8	8,333333	0	4,761905	0,462963
16	73,80952	0	44,047619	0,925926

F. Pengaruh Penambahan Fitur Ridge Shape terhadap EER

Tabel 6 memperlihatkan pengaruh penambahan fitur *ridge shape* terhadap EER. Penambahan fitur *ridge shape* meberikan pengaruh yang baik kepada EER. Secara umun terjadi penurunan EER akibat penambahan fitur *ridge shape*. jika diperhatikan, penurunan EER paling besar terjadi pada *step size* 16 dan tidak terjadi penurunan EER pada *matching* dengan *step size* 1. Pada *matching* dengan *step size* 1, fitur *ridge shape* tidak berperan dalam penurunan EER karena hasil *matching* dengan *step size* 1 sudah akurat dan detail. Sedangkan pada *matching* dengan *step size* 16, perhitungan skor tidak akurat dan detail sehingga skornya tidak maksimal dan penambahan skor akibat fitur *ridge shape* akan memberikan efek yang besar untuk EER.

TABEL 6 PENGARUH PENAMBAHAN FITUR *RIDGE SHAPE* TERHADAP EER

Database A		
step size	EER	ERR dengan Ridge Shape
1	0,82672	0,82672
2	0,82672	0,82672
4	1,058201	0,82672
8	2,612434	2,612434
16	18,948413	12,334656

V. KESIMPULAN

Dari hasil penelitian yang telah dilakukan, dapat disimpulkan beberapa hal yaitu:

1. Penggunaan *step size* akan menyebabkan waktu komputasi turun menjadi $\frac{t}{q^2}$, dimana t adalah waktu komputasi tanpa menggunakan *step size* dan q adalah nilai *step size*.
2. Penggunaan *step size* memperkecil skor hasil *matching* sehingga pada threshold tertentu nilai FRR akan membesar dan nilai FAR akan mengecil.
3. Penggunaan *step size* secara umum akan memperbesar nilai EER yang berarti hasil akurasi *matching* memburuk. Namun demikian pada kasus tertentu dimungkinkan nilai ERR mengecil sebagai akibat dari penurunan nilai skor.
4. Penambahan fitur ridge shape tidak berpengaruh terhadap waktu komputasi
5. Penambahan fitur ridge shape memperbesar skor hasil *matching* sehingga pada threshold tertentu nilai FRR akan mengecil dan nilai FAR akan membesar.
6. Penambahan fitur ridge shape secara umum akan memperkecil EER yang berarti akurasi *matching* semakin baik.
7. Dari hasil penelitian, penambahan fitur ridge shape akan meningkatkan akurasi yang direpresentasikan sampai 65%.

REFERENSI

[1] C. Barral, "Biometrics & Security : Combining Fingerprints , Smart Cards and Cryptography," ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE, 2010.

[2] A. M. Bazen and S. H. Gerez, "Fingerprint *matching* by thin-plate spline modelling of elastic deformations," *Pattern Recognit.*, vol. 36, no. 8, pp. 1859–1867, 2003.

[3] D. Maltoni, D. Maio, A. K. Jain, and S. Prabhakar, *Handbook of Fingerprint Recognition*, 2nd ed. Springer, 2003.

[4] A. Mohammad Abdel-Mawgoud Saleh, "Enhanced Secure Algorithm for Fingerprint Recognition," 2004.

[5] A. Surya Rikin, D. Li, T. Isshiki, and H. Kunieda, "A fingerprint *matching* using *minutia* ridge shape for low cost *Match*-on-Card systems," *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.*, vol. E88-A, no. 5, pp. 1305–1312, 2005.

[6] L. Wieclaw, "A *MINUTIA*-BASED *MATCHING* ALGORITHMS IN FINGERPRINT RECOGNITION," *J. Med. informatics Technol.*, vol. 13, no. Fig 2, 2009.