PENGGUNAAN METODE THINNING DALAM PENGOLAHAN CITRA

DINAMIKA
TEKNIK
Vol. II, No. 1
Januari 2008
24 - 33

Ari Endang Jayati
Dosen Universitas Semarang

Abstract

In image processing, to get the desired image, there are many ways to manipulate it. Thinning (which is called skeletonizing) is a method to represent the image transformation to a graph shape by reducing the particular information in image. Thinning method can be implemented in OCR (Optical Character Recognition) , industry and forestry. There are many algorithm to apply this method, in the meanwhile Thinning Binary Regions Algorithm.

Keywords : Thinning, skeletonizing, thinning binary region

A. PENDAHULUAN

Pada proses pengolahan citra, agar didapatkan citra yang baik sesuai dengan yang diinginkan, dapat dilakukan beberapa manipulasi terhadap citra tersebut. Beberapa contoh pemrosesan citra yang dapat dilakukan adalah proses transformasi, dekomposisi, rekonstruksi, *blurring*, *edge detection*, *thinning* dan lain sebagainya.

Dari sekian banyak pemrosesan citra yang dapat dilakukan, ada suatu pemrosesan yang dapat menghilangkan informasi dari citra yang tidak diinginkan, misalnya saja dalam suatu peta daerah geografis, jika hanya ingin melihat daerah yang terletak pada dataran tinggi, maka dapat melakukan proses *thinning* pada citra tersebut untuk mendapatkan informasi yang diinginkan. Selain menghilangkan informasi-informasi yang tidak diinginkan, proses *thinning* ini juga dapat mengkompresi citra tersebut, sehingga ukuran *file* citra yang dihasilkan tidak terlalu besar (Ananta, 2003)

Thinning merupakan suatu langkah *preprocessing* yang penting dalam bermacam operasi penganalisaan suatu citra antara lain pengenalan karakter yang bersifat optik, pengenalan sidik jari dan pemrosesan dokumen. *Thinning* melibatkan penghapusan titik (*point*) atau lapisan (*layer*) dari *outline* suatu gambar sampai semua

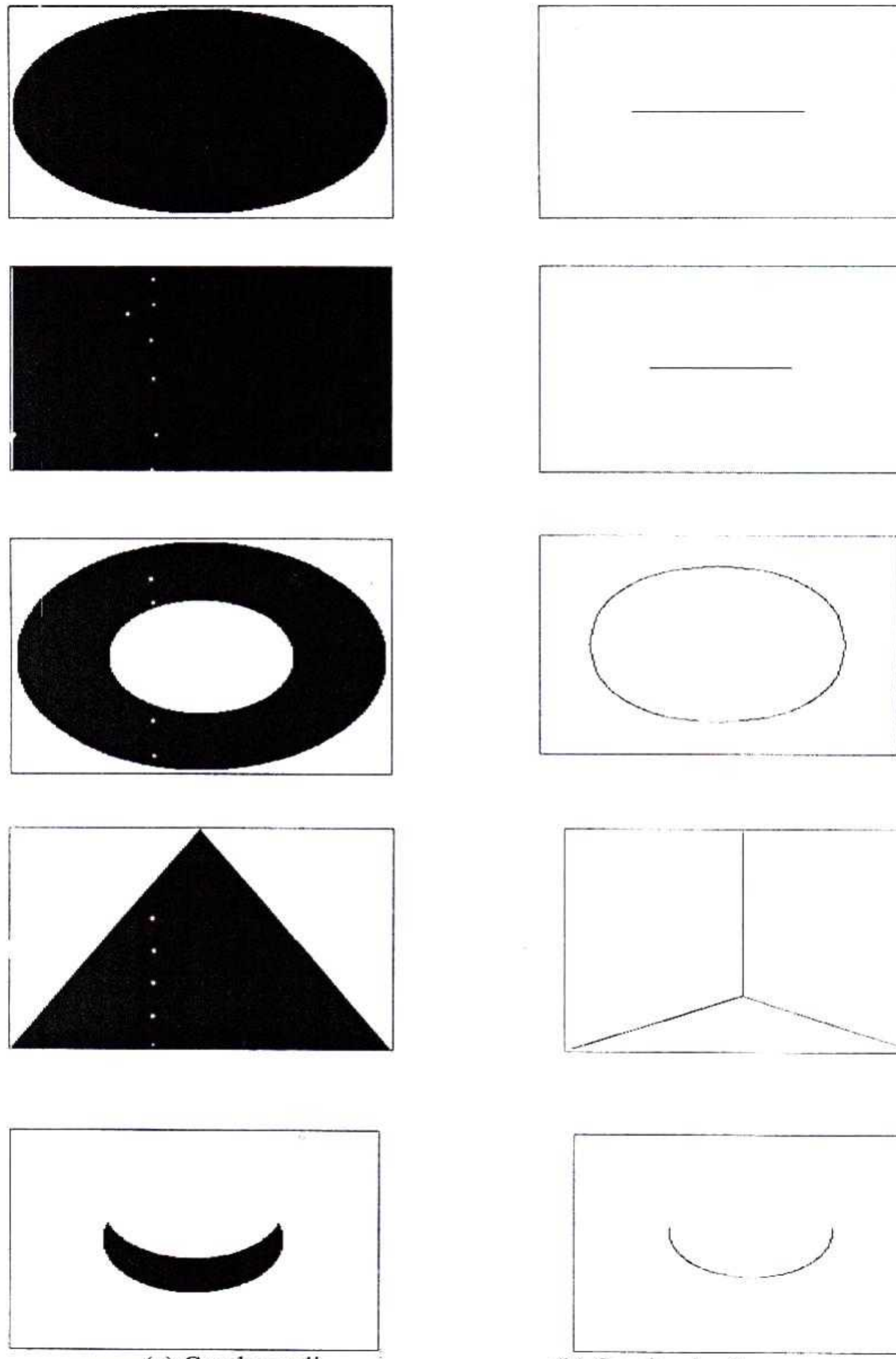
garis atau kurva mempunyai ukuran sebesar satu piksel. Garis atau kurva yang dihasilkan tersebut bisa disebut kerangka obyek. Tidak ada definisi yang baku mengenai kerangka secara matematik; menerapkan metode *thinning* yang berbeda pada suatu gambar akan menghasilkan hasil yang berbeda juga. Pendekatan umum dalam mendapatkan kerangka suatu obyek terdiri dari penghapusan (pada setiap iterasi) seluruh piksel sisi kecuali piksel yang mengacu kerangka tersebut. Piksel sisi adalah piksel yang menunjukkan batas (juga *hole*) yang terdapat pada suatu gambar (Dewa, 2003).

Thinning dapat diterapkan dalam beberapa hal, terutama untuk *skeletonization*. Dalam hal ini, *thinning* dipakai untuk merapikan hasil dari deteksi sisi dengan menipiskan semua garis sampai lebarnya hanya satu piksel. Seperti halnya operasi morfologis lainnya maka *thinning* biasanya diterapkan pada citra biner, dan menghasilkan citra biner yang lain sebagai hasil.

Thinning dapat diaplikasikan untuk objek-objek yang terdiri dari garis-garis (lurus atau melengkung/kurva). Metode ini tidak berlaku untuk objek yang memiliki bentuk yang menutupi daerah luas. *Thinning* lebih banyak merupakan suatu proses penengah, untuk mempersiapkan objek agar bisa dianalisa. Proses-proses berikutnya menentukan sifat-sifat dari kerangka tersebut.

B. DEFINISI THINNING

Thinning (biasa disebut juga *skeletonizing*) adalah suatu metode untuk merepresentasikan transformasi suatu bentuk citra ke bentuk *graph* dengan mereduksi informasi tertentu dalam citra tersebut (Xiwen, 1999). *Thinning* ini biasa digunakan mencari bentuk dasar/rangka/skeleton dari suatu citra. Contohnya pada PCB (*printed circuit boards*) untuk mengetahui aliran/arus data pada PCB tersebut. Selain untuk kompresi suatu citra, kegunaan lain dari *thinning* adalah untuk mencari informasi tertentu dari suatu citra dengan menghilangkan informasi yang tidak diperlukan. Misalnya saja untuk mencari dataran tinggi dalam peta geografis. Gambar berikut memperlihatkan gambar orisinal dan gambar hasil thinning.



(a) Gambar asli (b) Gambar hasil

Gambar 1. Contoh Gambar Hasil *Thinning* (Dewa, 2003)

C. ALGORITMA THINNING

Sebagian besar algoritma *thinning* merupakan algoritma yang bersifat iteratif. Dalam sebuah iterasi piksel sisi diperiksa berdasarkan beberapa kriteria untuk menentukan apakah suatu piksel sisi dihapus atau tidak. Ada beberapa jenis algoritma *thinning* yaitu *sequential* dan *parallel*. Jenis *sequential* menggunakan hasil dari iterasi sebelumnya dan hasil yang didapatkan sejauh ini dalam iterasi yang sekarang untuk memroses piksel yang sekarang. Jadi pada setiap ujung iterasi sejumlah piksel telah diproses terlebih dahulu. Hasil ini dapat digunakan secepatnya untuk memroses piksel selanjutnya.

Sedangkan jenis *parallel*, hanya hasil dari iterasi sebelumnya yang mempengaruhi keputusan untuk menghapus suatu titik pada iterasi yang sekarang. Contoh dari algoritma *parallel* adalah *Zhang-Suen Thinning Algorithm*. Sebagian besar aplikasi menggunakan salah satu dari dua strategi ini untuk menipiskan (*to thin*) bermacam bentuk. Algoritma yang satu bisa menghasilkan kerangka yang benar untuk beberapa bentuk tetapi dapat menghasilkan kerangka yang salah (buruk) dengan menggunakan algoritma yang lain. Sangat sulit untuk mengembangkan sebuah algoritma *thinning* yang umum dan yang dapat menghasilkan hasil yang memuaskan untuk seluruh macam bentuk gambar (Dewa, 2003).

Algoritma paralel melakukan pendekatan dengan cara mengunjungi semua *pixel* yang ada pada bitmap, cara ini digunakan untuk menemukan *dark point*. Dark point kemudian diklasifikasikan menjadi *edge dark point dan non-edge dark point*, tetapi hanya *edge point* yang perlu dipertimbangkan. Test akan diberlakukan pada tiap-tiap *edge point* pada delapan ketetanggaan. *Non-safe point* pada akhirnya akan dibuang, sedangkan untuk *break* dan *end* point, karena keduanya ini merupakan *safe point* maka kedua point ini tidak dibuang. (Dewa, 2003).

Berdasarkan kompleksitas waktu, algoritma paralel terdiri dari tiga komponen, yaitu :

1. Pada *sub-iteration* tiap-tiap *pixel* pada bitmap harus diperhitungkan untuk menentukan *dark pixel*. Jumlah operasi sangat tergantung dari besarnya area dari bitmap yang diolah.

2. Setiap dark *pixel* harus dihitung untuk *edge point*. Banyaknya jumlah operasi tergantung dari besarnya area dari obyek pada setiap operasi.
3. Jumlah operasi tergantung dari ketebalan dari obyek yang di proses.

Meskipun waktu yang dibutuhkan algoritma paralel lebih cepat dibanding dengan algoritma *sequential*, namun ternyata ada beberapa masalah dasar yang timbul. Pada beberapa algoritma paralel sebuah garis panjang dengan ketebalan 2 *pixel* akan dihilangkan, karena pada saat awal proses berjalan, *point-point* di kedua sisi dari garis tidak merusak hubungan dari pola jika dihitung secara terpisah. Jika kedua sisi dihitung dengan menggunakan paralel menggunakan hasil dari proses sebelumnya, maka akan langsung dihilangkan, karena hasil dari menghilangkan satu sisi tidak akan diketahui oleh sisi lain pada saat terjadi proses yang sama. Hal ini adalah masalah pengertian penggunaan *source* yang terjadi ketika beberapa proses paralel menggunakan memori yang sama.

Jelas terlihat bahwa keperluan dari multiple *sub-iteration* pada algoritma paralel dan kemungkinan dua *pixel* dengan kerangka pada pencarian *contour* mungkin menjadi bagian dari masalah *mutual exclusion*. Pada algoritma paralel sebuah *pixel* diproses berdasarkan pada status sebelumnya, sehingga pada saat *pixel* dianggap sudah ada pada paralel, maka semua *pixel* akan dihilangkan.

Ketika masalah dilihat dari sudut yang berbeda, terlihat bahwa pada kedua pencarian *contour*, *pixel* akan dihapus dari *contour* tanpa tahu apa yang terjadi pada obyek. Hasilnya adalah salah satu dari semua *pixel-pixel* yang ada akan dihilangkan, untuk menghindari terjadinya hal ini , maka sebuah kurva yang tipis akan dihasilkan pada akhir iterasi.

Penyelesaian untuk masalah ini adalah dengan menganggap hasil yang diperoleh dari mengolah *pixel-pixel* tertentu. Jika sebuah *pixel* dihilangkan, *contour* yang baru yang akan di buka. Jadi ketika *contour* itu di lewati, satu bagian dari *contour* yang baru dihasilkan untuk setiap *pixel* pada *contour* yang dikunjungi itu. Bagian ini akan diperiksa untuk *break point*, dan informasi ini tersedia ketika sub-rangkaian *pixel-pixel* pada rangkaian *pixel* atau pada rangkaian *pixel* yang akan

dikunjungi berikutnya. Pada akhir iterasi, sebuah *contour* akan ada untuk iterasi berikutnya tanpa harus menghilangkan *contour* yang lama. Suatu saat algoritma akan mempunyai pengetahuan yang lengkap tentang apa dari hasil obyek ketika *contour* yang sekarang dihilangkan.(Dewa, 2003).

Algoritma *thinning binary regions* merupakan contoh algoritma thinning *sequential* yang memberikan aspek sebagai berikut : (1) *thinning* tidak menghapus *point* terakhir, (2) *thinning* tidak merusak konektivitas, dan (3) *thinning* tidak menyebabkan pengikisan berlebihan dari *region*. Diasumsikan *region points* memiliki nilai 1 dan *background points* memiliki nilai 0. Metode ini terdiri dari 2 langkah dasar yang dikenakan terhadap *contour points* dari suatu *region*, dimana *contour points* adalah sembarang piksel dengan nilai 1 dan memiliki paling sedikit satu dari 8-tetangga bernilai 0. Algoritma ini menggunakan tanda untuk memilih piksel mana yang akan dihapus. Aturannya, 8-tetangga terdekat dari setiap piksel p_1 dinomori p_2 (untuk piksel di atas p_1) sampai dengan p_9 sesuai dengan arah jarum jam (Ananta, 2003).

p_9	p_2	p_3
p_8	p_1	p_4
p_7	p_6	p_5

Gambar 2. Piksel Citra

Langkah 1 Memberi tanda sebuah *contour point* p_1 untuk dihapus apabila memenuhi semua kondisi-kondisi berikut ini :

(a) $2 \leq N(p_1) \leq 6$;

(b) $S(p_1) = 1$;

(c) $p_2 \cdot p_4 \cdot p_6 = 0$;

(d) $p_4 \cdot p_6 \cdot p_8 = 0$;

dikunjungi berikutnya. Pada akhir iterasi, sebuah *contour* akan ada untuk iterasi berikutnya tanpa harus menghilangkan *contour* yang lama. Suatu saat algoritma akan mempunyai pengetahuan yang lengkap tentang apa dari hasil obyek ketika *contour* yang sekarang dihilangkan.(Dewa, 2003).

Algoritma *thinning binary regions* merupakan contoh algoritma thinning *sequential* yang memberikan aspek sebagai berikut : (1) *thinning* tidak menghapus *point* terakhir, (2) *thinning* tidak merusak konektivitas, dan (3) *thinning* tidak menyebabkan pengikisan berlebihan dari *region*. Diasumsikan *region points* memiliki nilai 1 dan *background points* memiliki nilai 0. Metode ini terdiri dari 2 langkah dasar yang dikenakan terhadap *contour points* dari suatu *region*, dimana *contour points* adalah sembarang piksel dengan nilai 1 dan memiliki paling sedikit satu dari 8-tetangga bernilai 0. Algoritma ini menggunakan tanda untuk memilih piksel mana yang akan dihapus. Aturannya, 8-tetangga terdekat dari setiap piksel p_1 dinomori p_2 (untuk piksel di atas p_1) sampai dengan p_9 sesuai dengan arah jarum jam (Ananta, 2003).

p_9	p_2	p_3
p_8	p_1	p_4
p_7	p_6	p_5

Gambar 2. Piksel Citra

Langkah 1 Memberi tanda sebuah *contour point* p_1 untuk dihapus apabila memenuhi semua kondisi-kondisi berikut ini :

(a) $2 \leq N(p_1) \leq 6$;

(b) $S(p_1) = 1$;

(c) $p_2 \cdot p_4 \cdot p_6 = 0$;

(d) $p_4 \cdot p_6 \cdot p_8 = 0$;

di mana $N(p_1)$ adalah jumlah dari tetangga-tetangga dari p_1 yang bukan nol; sehingga,

$N(p_1) = p_2 + p_3 + p_4 + p_5 + p_6 + p_7 + p_8 + p_9$

dan $S(p_1)$ adalah jumlah dari transisi 0 ke 1 dalam urutan p_2, p_3, p_4, p_5, p_6, p_7, p_8, p_9.

Langkah 2 Hampir sama dengan langkah 1, hanya saja pada langkah 2 ini bagian (c) dan (d) berubah menjadi sebagai berikut :

(c)' $p_2 \cdot p_4 \cdot p_8 = 0$;

(d)' $p_2 \cdot p_6 \cdot p_8 = 0$;

Langkah 1 diterapkan untuk *border pixel* pada *binary region*. Jika satu atau lebih dari kondisi (a) sampai dengan (d) tidak dipenuhi, maka nilai dari *point* yang diperiksa tidak berubah (tidak perlu diberi tanda). *Points* yang diperiksa tidak akan dihapus sampai semua *border points* selesai diproses. Hal ini dilakukan untuk mencegah agar tidak terjadi perubahan pada struktur data saat pengeksekusian algoritma. Setelah langkah 1 telah selesai dilakukan terhadap semua *border points* , semua yang telah diberi tanda dihapus (diubah ke 0). Lalu, langkah 2 baru dijalankan terhadap hasil data persis sama seperti pada langkah 1.

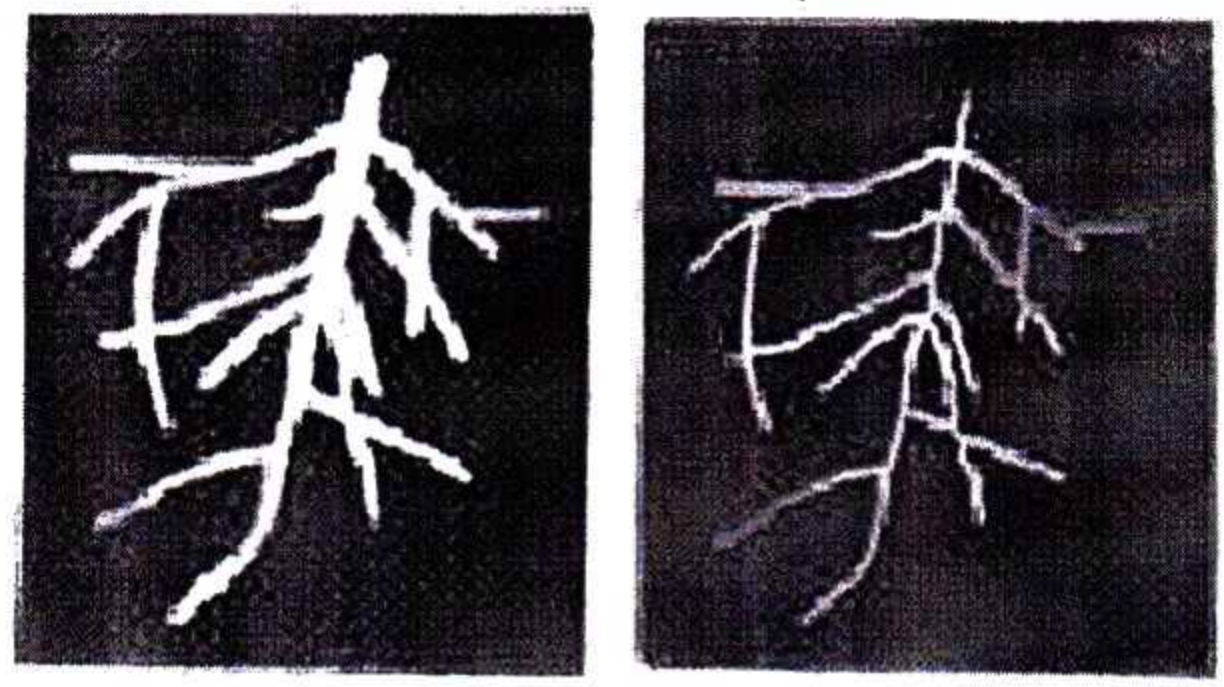
Jadi satu iterasi dalam algoritma *thinning* ini terdiri dari :

1. Pengerjaan langkah 1 untuk memberi tanda pada *border points* untuk dihapus.
2. Penghapusan *points* yang telah diberi tanda.
3. Pengerjaan langkah 2 untuk memberi tanda *border points* yang tersisa untuk dihapus.
4. Penghapusan *points* yang telah diberi tanda.

Prosedur dasar ini akan beriterasi hingga tidak ada *points* yang dapat dihapus lagi sehingga hasil yang didapat adalah skeleton (kerangka) dari *region*.

Proses *thinning* ini, menghilangkan informasi-informasi tertentu dalam citra, dengan tetap mempertahankan informasi yang paling utama atau kerangka utama citra tersebut. Jadi misalnya terdapat suatu citra yang cukup tebal, jika dilakukan

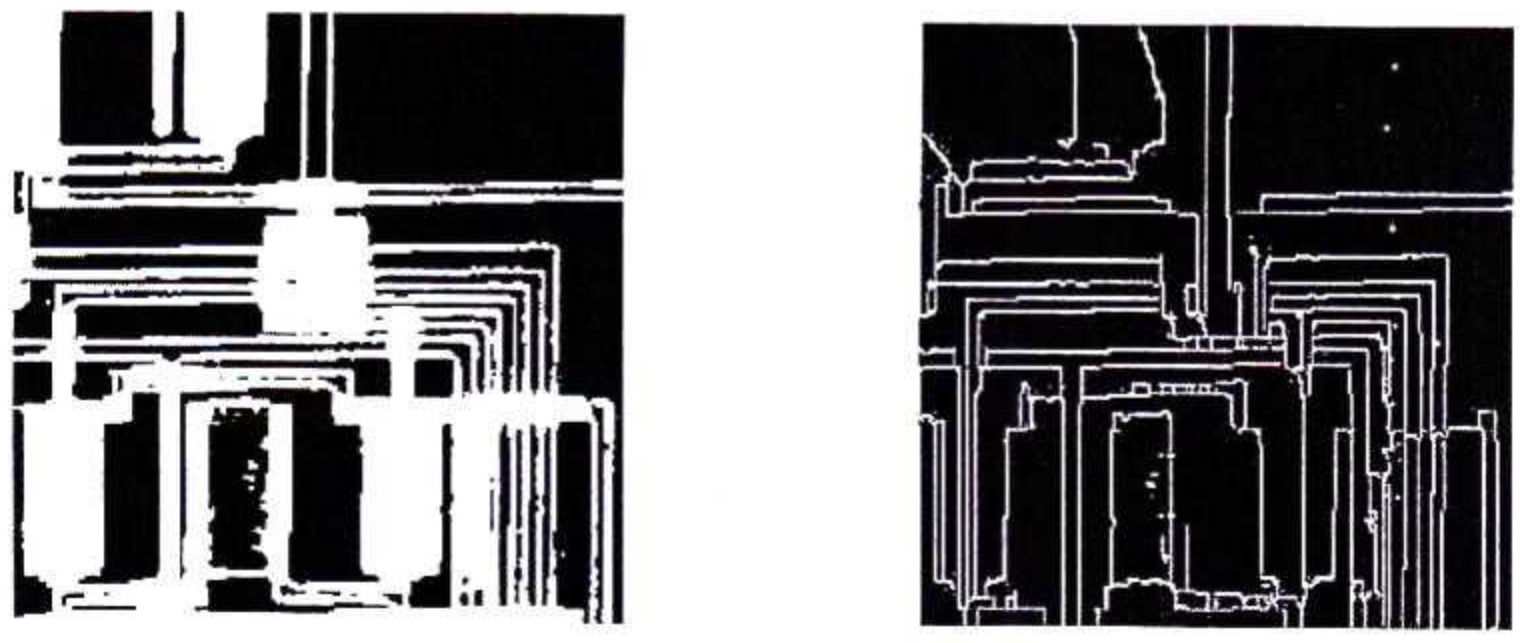
proses *thinning* pada citra tersebut, maka citra yang tersisa hanyalah kerangka utama dari citra tersebut. Contohnya dapat dilihat dalam gambar akar berikut, (a) gambar akar yang terlihat cukup tebal, sedangkan dalam (b) setelah dilakukan proses *thinning*, maka gambar akar yang terlihat hanya tinggal pola atau kerangka utamanya saja.



(a) Gambar asli (b) Gambar hasil

Gambar 3. Contoh Proses Thinning (Xiwen 1999)

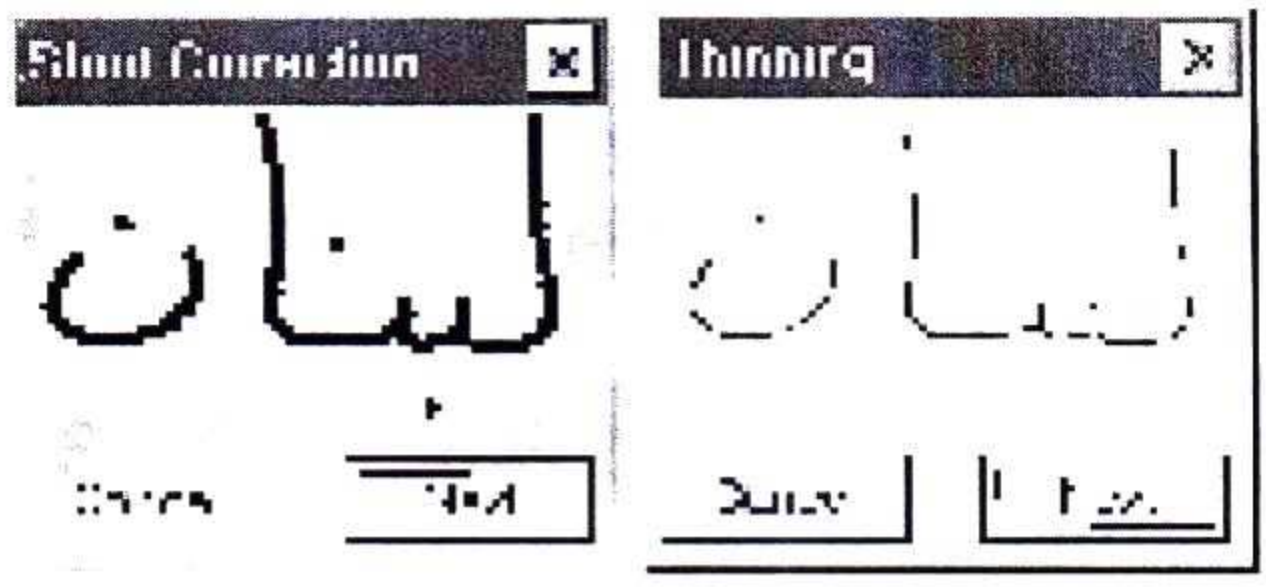
Pada bidang industri sering digunakan proses *thinning* untuk mencari kerangka utama dari sirkuit seperti gambar di bawah ini :



(a) Gambar sebuah sirkuit (b) Gambar hasil *thinning*

Gambar 4. Penerapan Thinning dalam Industri Elektronik (Xiwen, 1999)

Thinning juga sering digunakan dalam implementasi OCR (*Optical Character Recognition*).



(a) Gambar Karakter asli (b) Gambar hasil Thinning

Gambar 5. Penerapan Thinning dalam OCR (Fahmy, 2000)

D. KESIMPULAN

- *Thinning* merupakan metode untuk merepresentasikan transformasi citra ke bentuk *graph* dengan mereduksi informasi tertentu dalam citra tersebut.
- Metode *thinning* dapat diimplementasikan dalam OCR (*Optical Character Recognition*), bidang industri dan kehutanan.
- Algoritma yang digunakan untuk menerapkan metode ini bermacam-macam, salah satunya adalah algoritma *thinning binary regions* yang tidak menyebabkan pengikisan berlebihan dari *region*.

E. DAFTAR PUSTAKA

- Ananta, Adito, dkk., (2003), *Penggunaan Metode Thinning Pada Pemrosesan Gambar*, Universitas Indonesia, Jakarta.
- Dewa, Arya, dkk. (2003), *Thinning*, Universitas Indonesia, Jakarta.
- Fahmy, Maged Mohamed Mahmoud dan Somaya Al Ali, (2000), *Automatic Recognition Of Handwritten Arabic Characters Using Their Geometrical Features*, diakses dari http://www.ici.ro/ici/revista/sic2001_2/art1.htm, Internet; diakses pada tanggal 15 November 2003.
- Xiwen, Luo dan Wu Changgao. (1999), The Application of Digital *Image Processing to the Analysis of Root Pattern and Architecture*, diakses dari http://www.lib.ksu.edu/depts/issa/china/icae/part5/ic167.pdf; Internet, diakses pada tanggal 15 November 2003.