Um Novo Algoritmo Paralelo de Afinamento

FRANCESCO ARTUR PERROTTI
ROBERTO DE ALENCAR LOTUFO

Grupo de Computação de Imagens
Departamento de Engenharia de Computação e Automação Industrial
Faculdade de Engenharia Elétrica
C.P. 6101, Unicamp
13081 - Campinas - SP
lotufo@dca.fee.unicamp.br

Abstract. This paper presents a new parallel thinning algorithm. It is a one step fully parallel algoritm processing a window of 3x3 in the neigboring map of the image. It is simple, fast, isotropic and has built-in noise sensibility parameters.

1 Introdução

Existem muitas aplicações onde se faz necessário uma descrição estrutural da forma dos objetos de uma imagem para seu reconhecimento ou codificação. Exemplos típicos são o reconhecimento de caracteres, ideogramas, ícones, símbolos, mapas e texturas. Uma boa maneira de representar a forma de um objeto é através de seu esqueleto.

Considerável esforço tem sido empregado na busca de algoritmos de afinamento que sejam rápidos e eficientes [1-13]. Em particular os algoritmos paralelos estão merecendo a atenção de um número cada vez maior de autores [1-5] graças ao crescente uso de sistemas de processamento paralelo aplicado à computação de imagens. Infelizmente não é possível obter algoritmos completamente paralelos utilizando operadores com suporte 3x3 [4] e vários trabalhos procuram contornar este problema utilizando suportes maiores [1],passos (sub-ciclos) [2-4], ou partição da imagem [4].

O algoritmo apresentado aqui é:

- Totalmente paralelo com suporte 3x3 no processamento do mapa de vizinhança.
- Utiliza operadores isotrópicos.
- Resulta esqueleto de boa qualidade próximo do eixo mediano.
- Possui parâmetros para controlar a sensibilidade a ruído na imagem.

O algoritmo não garante que as linhas do esqueleto tenham sempre um pixel de espessura. Como discutido na seção 2, tal garantia implica na perda de isotropia do algoritmo.

Na próxima seção o algoritmo proposto é situado em relação aos outros existentes e seus conceitos básicos são apresentados. Na seção 3 é apresentada a notação utilizada. O algoritmo é apresentado na seção 4 e a seção 5 mostra experimentos realizados com este e outros algoritmos. Finalmente a seção 6 tece comentários finais e apresenta as conclusões.

2 Afinamento e Esqueletos

2.1 Conceitos Básicos

- Dado o pixel p, um pixel é vizinho-8 de p se tem uma aresta ou um vértice em comum com p (ver fig. 3).
- Um pixel é vizinho-4 de p se tem uma aresta em comum com p.
- A imagem é binária, com pixels tendo valores 0 ou 1.
- Um pixel é considerado ativo ou parte do objeto se o seu valor for 1 e desativado ou pertencente ao fundo quando seu valor for zero.
- Considera-se imagem com conexão 8-4, ou seja, os pixels dos objetos são conectados por vizinhança-8 e os pixels do fundo por vizinhança-4.
- Um algoritmo de afinamento é iso trópico se o espelhamento da imagem (vertical, horizontal ou ambos) e rotacões de múltiplos de 90 graus não alteram o esqueleto gerado.

Sinha [11] define o esqueleto de um caracter como sendo o que resta deste após a remocão da informação sobre sua espessura. Podemos definir o afinamento de imagens como sendo o processo de redução

da espessura dos objetos de uma imagem. No afinamento, a idéia é erodir as bordas dos objetos preservando suas propriedades topológicas de conectividade até que sobrem apenas linhas nas medianas.

Segundo Zuo e Hall [4], um algoritmo de afinamento paralelo deve tentar atingir os seguintes objetivos:

- A conectividade dos objetos e do fundo da imagem é preservada no esqueleto.
- As curvas e uniões das curvas do objeto devem estar representadas no esqueleto.
- As curvas do esqueleto devem ser tão finas quanto possível e próximas da mediana das regiões dos objetos.
- O afinamento deve ser realizado no menor número de iterações possível.

O'Gorman [12] acrescenta que a posição aproximada dos pontos terminais das linhas deve ser mantida.

Vale notar que estes objetivos não são suficientes para definir um único esqueleto para cada objeto possível [9]. O fato é que não existe uma definição formal de esqueleto, o que permite variações no esqueleto gerado. Diferentes algoritmos de afinamento aplicados a um mesmo objeto podem produzir esqueletos diferentes e ainda atingir os objetivos acima.

2.2 Algoritmos de Afinamento

Tem havido uma grande proliferação de algoritmos de afinamento utilizando uma variedade de métodos. De maneira geral, para chegar ao esqueleto, os algoritmos podem corroer a imagem de maneira iterativa [1-8], ou aplicar em sequência uma série de procedimentos que resultem no esqueleto [5,9,10].

Os algoritmos de afinamento podem ser classificados em sequenciais e paralelos. Nos algoritmos sequenciais, os pixels são processados sequencialmente um de cada vez e a avaliação de cada pixel depende da avaliação do pixel anterior. O volume de processamento necessário para obter o esqueleto é muito menor do que nos algoritmos paralelos por que estes tem que avaliar todos os pixels da imagem a cada iteração. Portando em máquinas sequenciais os algoritmos sequenciais são mais rápidos que os paralelos.

2.3 Algoritmos Paralelos

Um algoritmo é paralelo quando o processo de desativação de um pixel não interfere no processo de outros pixels da imagem, podendo um pixel ser avaliado ao mesmo tempo que seus vizinhos. A grande vantagem dos algoritmos paralelos é que cada região da imagem pode ser tratada de forma independente, o que é apropriado a sistemas de processamento paralelo.

O problema de conseguir algoritmos paralelos reside no fato de que para apagar um pixel e ao mesmo tempo manter a conectividade do esqueleto, é preciso conhecimento da vizinhança imediata do pixel.

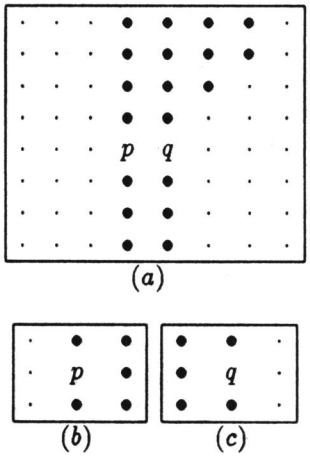


Figura 1: (a) imagem, (b) e (c) região de 3x3 em torno de p e q respectivamente.

No exemplo da figura 1, para manter a conectividade do esqueleto, q deve ser mantido se p for desativado e vice-versa. No caso de algoritmos paralelos, a avaliação de p não depende da avaliação de q e janelas de vizinhança 3x3 não fornecem informação suficiente para esta decisão.

Para resolver esta situação, 3 soluções são encontradas na literatura:

1. Uso de janelas maiores. Alguns algoritmos usam vizinhanças de 3x4, 4x4 ou 5x5 pixels para tomar a decisão. Uma interessante solução proposta por Chin e Wan [1], usa um conjunto de 8 máscaras 3x3 e mais duas máscaras 4x1.
2. Divisão de cada iteração em passos (em geral 2 ou 4), onde cada passo considera um conjunto diferente de configurações de vizinhança. Um exemplo deste tipo de solução é o popular algoritmo de Zhang/Suen [2]. Configurações simétricas como as de p e q, são tratadas em passos diferentes e a conectividade do esqueleto é garantida usando apenas suporte 3x3. Esta solução tem a desvantagem de aumentar o volume de processamento, porque cada passo (ou sub-iteração) de cada iteração exige o processa-

mento de todos os pixels da imagem. Em geral, o tempo de processamento deste tipo de algoritmo aumenta com o aumento do número de passos.

3. Divisão da imagem em dois campos [4] de forma que um pixel que pertenca ao campo 1, só tenha entre seus vizinhos-4, pixels do campo 2 e viceversa. As itereções processam alternadamente os pixels de apenas um dos campos. Hall [4], prova que este tipo de algoritmo é sempre mais rápido do que os algoritmos que usam passos.

Uma quarta solução usando o mapa de vizinhança da imagem binária é proposta neste trabalho e descrita a seguir.

2.4 O algoritmo proposto

A idéia básica é efetuar o afinamento pelo processamento do mapa de vizinhança da imagem. Cada ponto (x,y) no mapa, representa o número de vizinhos da região 3x3 em torno do pixel (x,y) da imagem. Portanto, uma janela de 3x3 no mapa de vizinhança armazena informação sobre uma área de 5x5 pixels. O acréscimo de informação dado a cada ponto no mapa de vizinhança, permite que o algoritmo resolva situações que seriam impossíveis de resolver com uma janela 3x3 em uma imagem binária. Um exemplo é apresentado na figura 2.

Considerando apenas a posição e número de vizinhos na vizinhança 3x3, p e q têm exatamente a mesma configuração e é impossível que uma regra baseada apenas nessas informações estabeleça qualquer diferença entre eles. No entanto, se for considerado o valor dos vizinhos de p na vizinhança 3x3 no mapa de vizinhança, a diferença torna-se visível. No exemplo da figura, existe um ponto com valor 8 que é vizinho de p garantindo que p esteja em uma região com no mínimo 3 pixels de espessura. Já a partir dos valores dos vizinhos de q, é possível deduzir (embora não seja necessário) que a região em torno de q tem 2 pixels de largura.

Um esqueleto ideal deveria ter apenas linhas com um pixel de espessura, mas como as imagens são discretas, para conseguir isto em algum momento o algoritmo precisa tomar decisões arbitrárias. Um exemplo típico é mostrado na figura 1. O pixel p ou o pixel q deve ser desativado e nenhum deles é o ponto médio da região. Qualquer critério usado para escolher um dos dois será arbitrário e nesse momento o esqueleto deixa de ser simétrico e isotrópico. Em geral, nos algoritmos com dois ou quatro passos esta decisão é implementada quando é decidido qual será o passo inicial. Embora o algoritmo possa definir

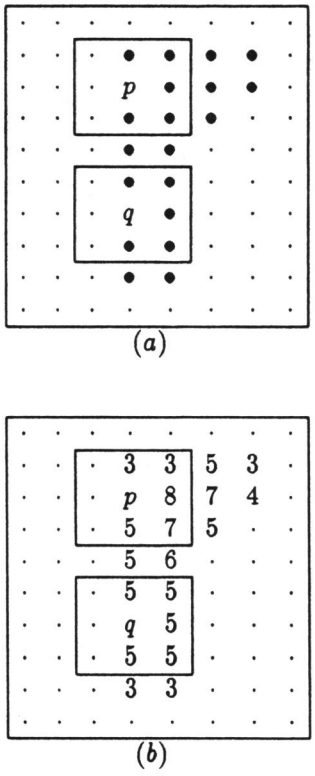


Figura 2: (a) imagem original, (b) mapa de vizinhança da imagem.

uma ordem para os passos, esta será sempre circular e escolher o passo inicial é uma decisão arbitrária.

É importante notar que como o algoritmo proposto utiliza apenas operadores isotrópicos, não garante que o esqueleto tenha apenas linhas com um pixel de largura. Se isto for desejável, outro algoritmo de afinamento pode ser usado para retirar os pixels excedentes. Caso este segundo algoritmo seja iterativo, apenas uma iteração será suficiente.

2.5 Sensibilidade a Ruídos no Contorno

A sensibilidade a ruídos está relacionada com o conjunto de configurações de vizinhança que classificam os pixels como pontos terminais. Quanto maior o número de configurações neste conjunto, maior será o número de pontos terminais no esqueleto. Quanto mais precisamente os pontos terminais representarem as protuberâncias e pontas do objeto, mais o esqueleto estará sujeito a perturbações provocadas por ruído na imagem.

Em uma imagem de baixa resolução, cada pixel é percentualmente muito mais significativo que em uma imagem de alta resolução. Da mesma maneira, cada região do objeto tem importância para o esque-

leto proporcional à relação entre a área do objeto e a área da região. Assim, pequenas protuberâncias no contorno do objeto podem ter importância maior ou menor conforme o caso. Por este e outros motivos, é desejável que exista uma maneira de ajustar a sensibilidade do algoritmo a pequenas modificações no contorno, de modo a permitir uma filtragem de ruídos na imagem.

3 Definições e Notação

O conjunto V_p contém todos os vizinhos-8 do pixel p.

p_8	p_1	p_2
p_7	p	p_3
p_6	p_5	p_4

$$V_p = [p_1,\ p_2,\ p_3,\ p_4,\ p_5,\ p_6,\ p_7,\ p_8]$$

Figura 3: Conjunto de vizinhos de um pixel

O mapa de vizinhança é uma matriz de inteiros com as mesmas dimensões que a imagem. Cada ponto no mapa armazena o número de vizinhos do pixel que representa se este for ativo. Para pixels desativados o valor no mapa é zero (ver figura 2). O mapa de vizinhança é dado por N(p):

$$N(p) = \begin{cases} \sum_{i=1}^{8} p_i & \text{se } p \neq 0, p_i \in V_p \\ 0 & \text{se } p = 0 \end{cases} \qquad (1)$$

p_{max} é o valor máximo encontrado nos vizinhos de p no mapa de vizinhança.

$T(p)$ retorna o número de objetos 4-conectados em V_p. Também pode ser interpretado como sendo o número de transições de fundo para objeto que ocorrem na vizinhança-8 de p. $T(p)$ é dado por:

$$T(p) = \sum_{i=1}^{8} \begin{cases} 1 & \text{se } p_i = 0 \text{ e } p_{i+1} = 1 \\ 0 & \text{em outro caso} \end{cases} \qquad (2)$$

onde $p_9 = p_1$

Quanto $T(p) = 1$, então p não é o ponto de junção entre duas regiões distintas do objeto e a desativação de p não desconecta o esqueleto. Se $T(p) > 1$, é necessário processamento adicional para que a conectividade do esqueleto seja garantida. No caso da Fig. 4 onde $T(p) = 2$, existem duas regiões distintas do objeto conectadas a p. Se p for desativado não se garante que o esqueleto se mantenha conectado.

O conjunto ordenado H_p é definido para $N(p) = 7$. Ele contém todos os pontos na vizinhança-8 de p

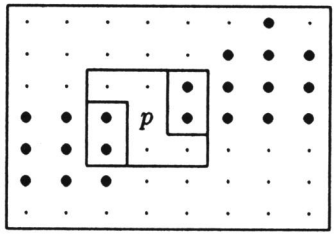
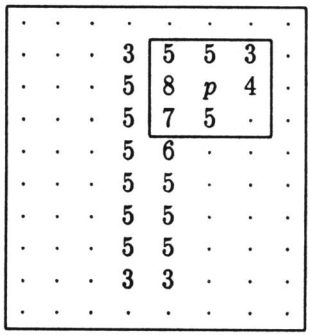


Figura 4: Exemplo de configuração com $T(p) = 2$.

no mapa de vizinhança. Em H_p, os pontos estão ordenados de maneira que o primeiro elemento do conjunto seja o vizinho desativado de p. Os elementos seguintes são encontrados percorrendo a vizinhança de p em sentido horário. O iésimo elemento de H_p é chamado de hi.



$$H_p = [0,\ 5,\ 7,\ 8,\ 5,\ 5,\ 3,\ 4]$$

Figura 5: Exemplo de configuração do conjunto H_p.

4 O Algoritmo

Cada iteração testa cada pixel ativo com as condições abaixo. O pixel p é apagado se (I) ou (II) forem satisfeitas:

$$(I) \quad Min \leq N(p) < 7 \ \ e \qquad (3)$$

$$T(p) = 1 \ \ e \qquad (4)$$

$$p_{max} > Max(N(p) + 1,\ R) \qquad (5)$$

onde: $1 \leq Min \leq 4$

$2 \leq R \leq 6$

$R > Min$

$$
\begin{array}{lll}
(II) & N(p) & = 7 \ \ e \qquad (6) \\
 & h_4 & = h_5 = h_6 = 8 \ \ e \qquad (7) \\
 & (h_2, h_3) & \in [(6,8),(5,7),(4,6),(4,5)] \ \ e \ (8) \\
 & (h_7, h_8) & \in [(8,6),(7,5),(6,4),(5,4)] \qquad (9) \\
 & \text{onde} & (h_i, h_j) \text{ é um par ordenado.}
\end{array}
$$

Quando $N(p) < 7$, a condição (3) garante que o pixel testado não é interno ao objeto e não é ponto terminal do esqueleto. A condição (4) garante que p não é o ponto de junção entre duas ou mais regiões do objeto. Finalmente a condição (5) testa a espessura da região em torno de p.

É válido supor que um pixel p que está na borda de um objeto, tenha vizinhos mais internos ao objeto do que ele próprio. Significa que deve existir um vizinho v onde $N(v) > N(p)$, ou seja, $p_{max} > N(p)$. Experimentalmente foi escolhida a condição de $p_{max} > N(p) + 1$ para preservar a conectividade do esqueleto.

Os parâmetros Min e R, permitem controlar a sensibilidade à ruído nas bordas aumentando ou diminuindo o número de configurações de vizinhança que permitem a desativação do pixel.

O parâmetro R é o fator limitante que impede que a erosão seja excessiva. Está relacionado com a espessura da região em torno do pixel em avaliação porque estabelece um valor mínimo para p_{max}.

O valor de Min especifica o número mínimo de vizinhos que um pixel deve ter para que possa ser desativado. Se $N(p) < Min$, p é considerado ponto terminal do esqueleto e deve ser mantido. Quanto maior o valor de Min, mais pontos terminais surgirão no esqueleto.

Aumentando o valor de Min, a sensibilidade do algoritmo aumenta, já que um número menor de configurações será testado para desativação do pixel. A probabilidade de um pixel ser apagado diminui e portanto o número de iterações pode mudar. Para $Min = 4$, o número de pontos terminais no esqueleto é maximo e para $Min = 1$ é mínimo. Valores típicos para Min são 3 e 4.

A condição (I) é suficiente para se chegar ao esqueleto, porém a inclusão de apagamento do pixel na condição (II) diminui o número de iterações do algoritmo. Na condição (II), quando $N(p) = 7$, a condição (7) seleciona apenas as configurações de vizinhança onde existem 3 pontos com valor 8 nas posições 4, 5 e 6 de H_p. Se a configuração satisfaz (6) e (7), então ela tem uma rotação ou inversão que combina com uma das duas máscaras apresentadas na figura 6.

As condições (8) e (9) são simétricas, uma é a inversão da outra. Existem 11 configurações possíveis de vizinhança (desconsiderando as rotações) que satisfazem a condição (8). A figura 7 mostra todas as configurações que satisfazem as condições (6), (7) e (8).

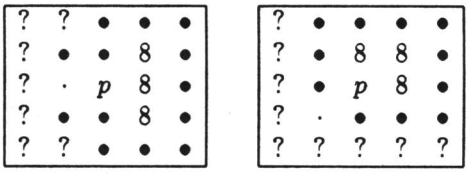


Figura 6: Configurações que satisfazem as condições (6) e (7)

A condição de término do algoritmo é que ocorra uma iteração onde nenhum pixel é desativado. No caso particular em que $R = 6$, o número de pixels com mais de 6 vizinhos pode ser controlado em cada iteração, de forma a evitar a última iteração. O esqueleto não deve conter pontos com 7 ou 8 vizinhos, caso em que pelo menos mais uma iteração é necessária. Por outro lado, para que um pixel seja desativado é necessário que tenha 7 ou 8 vizinhos. O mesmo raciocínio não é válido quando $R < 6$, porque podem existir pixels onde $p_{max} > R$ que devem ser mantidos.

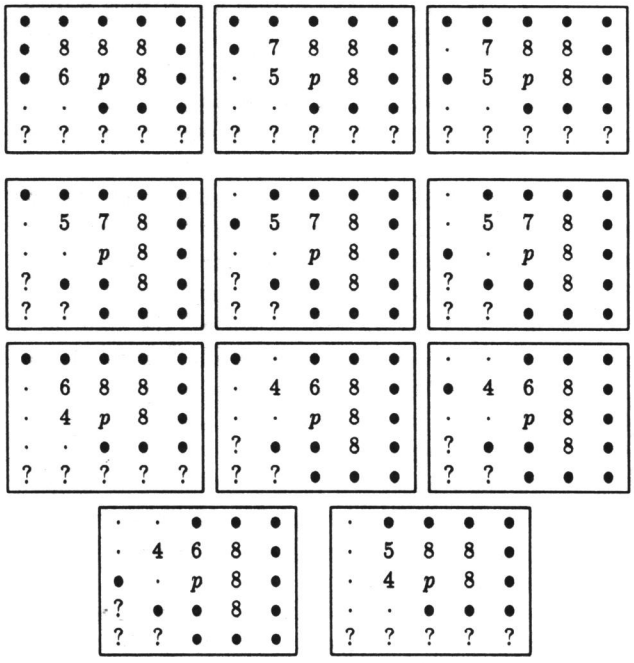


Figura 7: Configurações de vizinhança que satisfazem (6), (7) e (8).

5 Implementação e Experimentos

Algumas implementações de algoritmos de afinamento utilizam o conceito de look-up table como forma de reduzir o tempo de processamento da imagem. A look-up table é uma tabela onde estão armazenados dados relativos a cada configuração de vizinhança possível. Em uma janela $3x3$, existem 8 vizinhos e portanto 256 configurações diferentes de vizinhança. A posição de cada configuração na tabela é encontrada associando um bit a cada vizinho do pixel considerado e formando um número com os bits. Dados típicos que podem estar armazenados na look-up table são $N(p)$ e $T(p)$, mas conforme a implementação, pode ser viável armazenar outras informações relativas à vizinhança do pixel.

O uso da look-up table diminui o volume de processamento durante o afinamento, mas obriga que todos os vizinhos de cada pixel sejam percorridos para encontrar a posição na tabela. Por este motivo o método é raramente usado. No caso deste algoritmo, seu uso é justificável. Se cada ponto do mapa de vizinhança armazenar em vez de $N(p)$, a posição na look-table, sempre que um pixel p for apagado, são percorridos apenas os vizinhos ativos do pixel. Em cada visita, o bit do vizinho que representa o pixel é zerado. Desta maneira a posição na tabela está sempre atualizada e as funções $N(p)$ e $T(p)$ ficam disponíveis sem processamento adicional.

Para comparação com outros algoritmos, a look-up table não foi implementada. Na verdade o mapa de vizinhança já é uma tabela que armazena $N(p)$ de todos os pixels, e sem a necessidade de visitar todos os vizinhos para encontrar a posição que deve ser consultada. Os vizinhos ativos do pixel p devem ser visitados apenas quando p é desativado afim de manter o mapa de vizinhança atualizado. Para isso, todos vizinhos ativos de p devem ter seu valor decrementado, já que todos perderam um vizinho.

A função $T(p)$ foi implementada de maneira a retornar um valor booleano. Retorna TRUE quando $T(p) = 1$ e FALSE em outro caso. Desta maneira não é preciso visitar todos os vizinhos de p todas as vezes que a função é chamada.

A tabela 1 mostra o tempo medido e o número de iterações necessárias para afinar 5 imagens de dimensões diferentes com o algoritmo proposto e o algoritmo de Zhang/Suen.

A coluna do pré-processamento informa o tempo usado para criar o mapa de vizinhança inicial. O mapa é atualizado durante o processo. A coluna do afinamento mostra o tempo gasto para executar todas as iterações necessárias. A coluna Total, é a soma do tempo de pré-processamento e do tempo de afinamento. Para o acabamento foram usadas 2 sub-iterações com o algoritmo de Zhang/Suen sobre a imagem afinada. A coluna que mostra as iterações não inclui o acabamento. Todos os programas foram codificados em C, usando implementação sequencial e executados em um AT-386. Os tempos em segundos são a média de várias execuções sobre as imagens.

R/Min	1	2	3	4
2	1344	-	-	-
3	1341	1318	-	-
4	1318	1318	1304	-
5	1318	1318	1304	1237
6	1296	1296	1292	1237

Tabela 2: Número de pixels desativados na imagem 1 da tabela 1, usando todas as combinações válidas dos parâmetros Min e R.

A figura 8 mostra os resultados intermediários após cada iteração do algoritmo sobre a imagem 5 da tabela 1. Na figura 9, é apresentado o esqueleto final, com acabamento do algoritmo de Zhan/Suen, de todas as combinações dos parâmetros Min e R, bem como o esqueleto gerado por outros algoritmos de afinamento. O número de pixels apagados em cada combinação de Min e R, é mostrado na tabela 2. A imagem utilizada para o exemplo da figura 9 é a imagem 1 da tabela 1.

6 Conclusões

Este trabalho apresenta um novo algoritmo de afinamento que introduz o conceito de processamento de suporte 3x3 no mapa de vizinhança. Trabalha em um passo porque cada ponto no mapa fornece uma quantidade de informação maior do que um pixel em uma imagem binária poderia fornecer. Processar o mapa de vizinhança em uma janela 3x3 é equivalente a dispor de um suporte 5x5 na imagem binária. Isto permite que o algoritmo chegue a mais conclusões a respeito da região considerada tornando o processamento mais inteligente e eficiente.

O algoritmo proposto tem a vantagem de ser simples em conceito, embora sua implementação em sistemas paralelos seja mais complexa que os outros algoritmos porque são necessários mais canais de comunicação entre os processadores. É isotrópico, já que nenhuma relação leva em consideração a orientação das configurações de vizinhança. Possui parâmetros que permitem ajustar a sensibilidade do esqueleto à ruidos no contorno. Finalmente é rápido e produz esqueletos de boa qualidade.

	Algoritmo proposto					Alg. Zhang/Suen	
	Tempo (s)						
Imagem	Pre-Proc	Afinam	Total	Acabam	Iter	Tempo	Sub-Iter
1	0.22	1.04	1.06	0.17	9	3.73	17
2	0.33	1.98	2.31	0.22	14	7.32	26
3	0.44	2.41	2.85	0.66	9	4.12	11
4	0.99	6.48	7.47	1.82	9	12.72	11
5	0.06	0.22	0.28	0.06	6	0.17	10
Total	2.04	14.17	16.21	2.93	47	28.06	75

Tabela 1: Tempo de processamento e número de iterações com várias imagens.

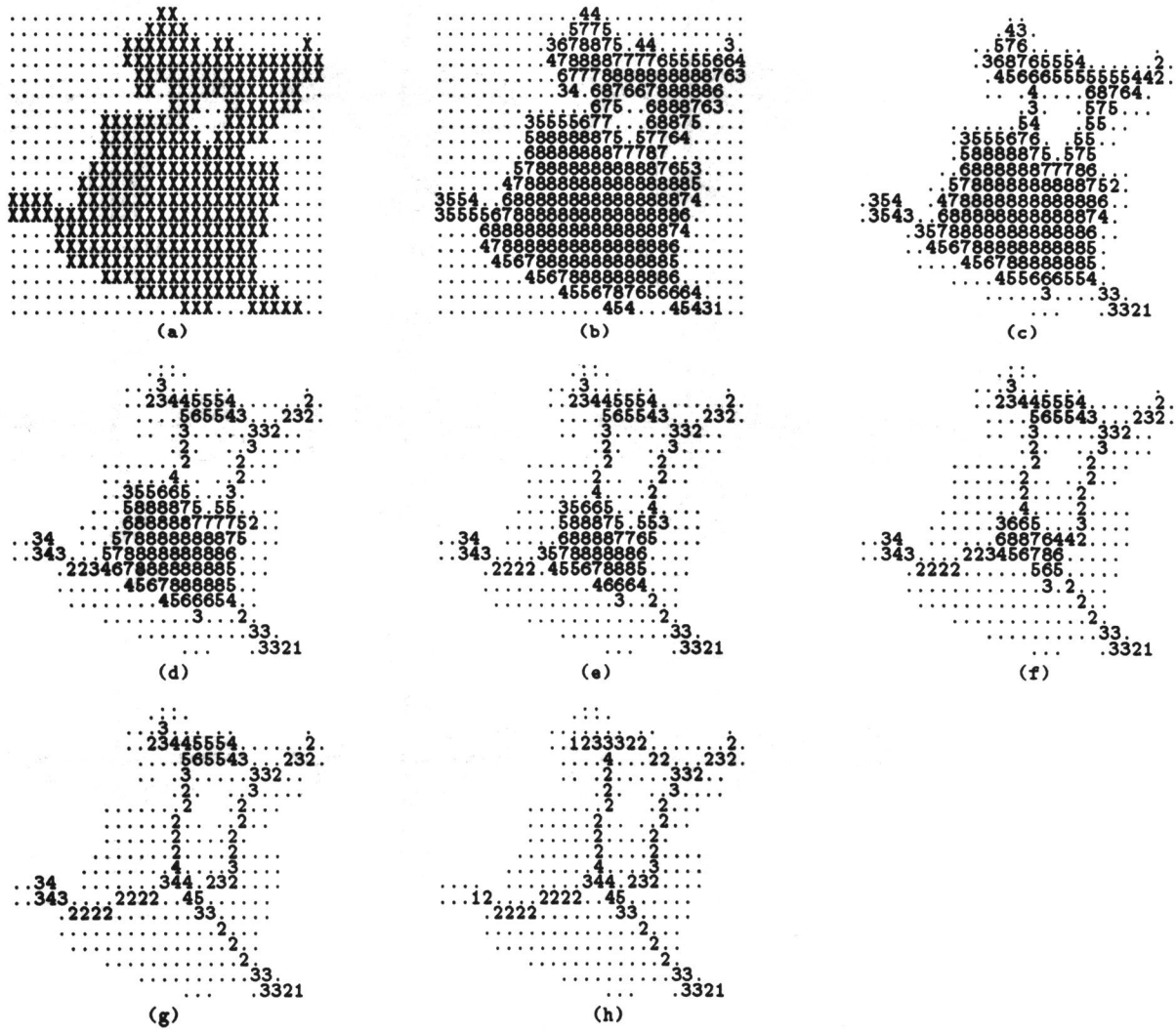


Figura 8: (a) imagem original, (b) mapa de vizinhança, (c) a (g) mapa depois de 1,2,3,4 e 5 iterações respectivamente usando $Min = 2$ e $R = 4$. (h) após o acabamento com duas sub-iterações do algoritmo de Zhang/Suen.

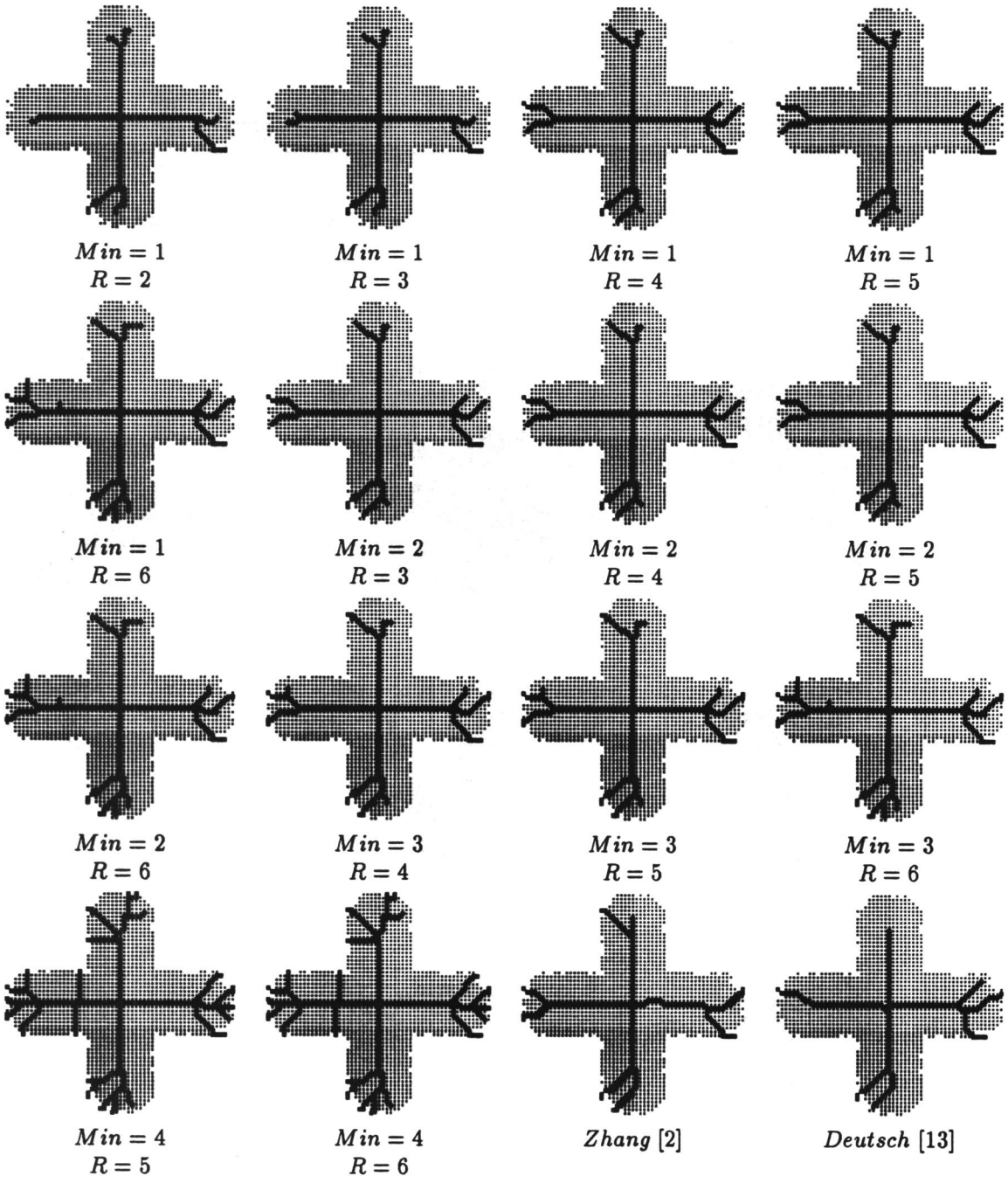


Figura 9: Exemplo de afinamento da imagem 1 da tabela 1, com todas as combinações válidas dos parâmetros *Min* e *R*, e o resultado obtido com o algoritmo de Zhang/Suen [2] e o algoritmo de Deutsch [13].

7 Referências

1. R.T. Chin e H.K. Wan, "A One-Pass Thinning Algorithm and Its Parallel Implementation", Comput. Vision Graphics Image Process., vol 40, 1987, pp 30-40.

2. T.Y. Zhang e C.Y. Suen, "A Fast Paralel Algorithm for Thinning Digital Patterns.", Comm. ACM, vol 27, No. 3, 1984, pp 236-239.

3. S. Suzuki e K. Abe, "Binary Picture Thinning by an Iterative Parallel Two-Subcicle Operation", Pattern Recog. Vol 20. No 3, 1987.

4. Z. Guo e R. W. Hall, "Parallel Thinning with Two-Subiteration Algorithms", Image Proces. and Comp. Vision, Vol 32, No 3, 1989.

5. E.R. Davies e P.N. Plummer, "Thinning Algorithms: A Critique And A New Methodology", Pattern Recog. vol 14, 1981, pp 53-63.

6. R. W. Hall, "Fast Parallel Thinning Algorithms: Parallel Speed and Connectivity Preservation", Comm. ACM, Vol 32, No. 1, 1989, p 124.

7. T. Pavlidis, "A Thinning Algoritm for Discrete Images", Comput. Graph. and Image Proces. Vol 13, pp 142-157

8. P.C.K. Kwok, "A Thinning Algorithm by Contour Generation", Comm. ACM, vol 31, No. 11, 1988, pp 1314-1324.

9. N.J. Naccache e R. Shinghal, "SPTA: A Proposed Algoritm for Thinning Binary Patterns", IEEE Trans. Vol SMC-14, No. 3, pp 409-418.

10. C. Arcelli e G. Sanniti di Baja, "A Width-Independent Fast Tinning Algorithm", IEEE Trans. on Patter Analysis and Mach. Intell., PAMI-7, No.4, 1985, pp 463-474.

11. R.M.K. Sinha, "A Width-Independent Algoritm for Caracter Skeleton Estimation.", Comput. Vision Graphics Image Process., vol 40, 1987, pp 388-397.

12. L. O'Gorman, "k x k Thinning", Comput. Vision Graph. Image Proc., vol 51, 1990, pp 195-215.

13. E. S. Deutsch, "Thinning algorithms on rectangular, hexagonal, and triangular arrays", Comm. Ass. Comput. Mach., vol 15, 1972, pp 827-837.