4x4 세선화 알고리즘에 관한 연구

A Study on The Improved 4x4 Thinning Algorithm

김훈기(컴퓨터소프트웨어과)
Hoon-ki Kim(Dept. of Computer Software)

keyword : thinning(세선화), core-line detection(중심선 검출) , data reduction

ABSTRACT : Image thinning is a common binary image processing operation and is usually applied to the gray scale images that have been thresholded to a binary representation. The purpose of thinning is to reduce the image components to their essential information so that further analysis are facilitated. This thinning algorithms have been used extensively for processing thresholded images, data reduction, pattern recognition, counting and labeling of connected regions, and curve fitting in computer animation.

This paper suggest “Improved 4x4 Thinning Method(I4TM)”. The I4TM uses the same criteria which are used k x k thinning, but this algorithm is faster than k x k thinning when 4 x 4 window is used for thinning. Criteria are given by which 4 x 4 method thins to minimally 8-connected lines while retaining connectivity and endpoint.

1. 서론

세선화(thinning)는 이진값을 갖는 이미지의 그림 부분을 중심선이나 뼈대그림으로 근사화하여 선으로 줄이는 이미지 프로세싱 과정이다. 이러한 세선화를 뼈대화(skeletonizing) 또는 중심선 검출(core-line detection)이라고 하기도 한다.

그 목적은 이미지를 주요한 정보만으로 축소시켜 이후의 이미지 처리를 용이하게 하기 위함이다. 이러한 세선화 알고리즘은 데이터 줄이기, 패턴 인식, 영역 계산 등의 이미지 처리에 광범위하게 사용된다. 또한, 컴퓨터 에니메이션에서 커브 피팅의 전단계로 사용된다.

일반적인 세선화 방법은 3x3 의 윈도우를 전체 이미지로 이동시키며 세선화 기준을 만나면 그 중앙 픽셀을 제거하는 방식으로 진행된다. Lawrence O'Gorman 은 이러한 3x3 방식을 kxk 로 일반화하였다. 이는 kxk 의 윈도우를 이용하는 방식으로 일정한 기준에 합치되면 중앙의 (k-2) x (k-2) 픽셀을 제거하는 방식이다. 이 방법의 장점은 굵은 선들에 대하여 적은 반복계산이 필요로 하다는 것이다. 그러나 k 값을 크게 할 경우 그 결과가 거칠어지고, 중앙선과 거리가 멀어지며, 끝선이 제거되어 길이가 짧아진다는 문제점이 있다.

세선화의 요구조건은 다음과 같다.

a. 연결되어 있는 이미지 영역은 연결된 선 구조로 세선화된다.
b. 세선화된 결과는 최소한 8 방향연결이다.
c. 끝선의 위치는 유지되도록 근사화한다.
d. 세선화된 결과는 이미지의 중앙선으로 근사화하여야 한다.
e. 세선화에 의하여 발생하는 예외적인 돌출을 최소화한다.

세선화의 결과가 요구조건 a.와 같이 연결을 유지하는 것이 중요하다. 요구조건 b.에 의하여

세선화된 선은 8 방향연결을 유지하며 최소한의 픽셀로 이루어져야 한다. 3x3 영역의 8 개의 가장자리에 있는 픽셀들은 중앙 픽셀과 연결되어야 한다.

본 연구에서는 향상된 4x4 세선화 방법(I4TM)를 제안하였다. 이 방법은 kxk 와 같은 세선화 기준을 사용한다. 그러나 이 알고리즘은 같은 4x4 를 사용할 경우 kxk 세선화 알고리즘보다 빠르다.

2 장에서는 코아 픽셀의 제거하기 위한 기준, 즉 4x4 윈도우를 기준으로 세선화하기 위하여 픽셀을 제거하는 필요조건에 대하여 논하였다. 또한 세선화를 빠르게 진행하기 위하여 이전의 주변 픽셀 맵과 바로 위 픽셀의 주변 픽셀 맵을 이용하여 빠르게 주변 픽셀 맵을 구성하는 방법에 대하여 3 장에서 논하고 있다. 4 장에서는 본 논문에서 제안한 I4TM 알고리즘을 설명하고 있으며, 5 장에서 기존의 알고리즘과 제안한 알고리즘을 적용한 결과를 보여주고 있다.

2. 세선화 기준

여기서는 픽셀 제거를 위한 기준에 대하여 서술한다. Fig.1 는 윈도우를 나타내고 있으며 이에 따르는 용어에 대하여 정의하면 다음과 같다.

0	1	2
7	X	3
6	5	4

0	1	2	3
11	X	X	4
10	X	X	5
9	8	7	6

(a) 3 x 3 window (b) 4 x 4 window

Fig.1 3x3 and 4 x 4 window

- 윈도우

픽셀 (x,y) 에서의 윈도우 $W(x,y,k),\ k=3,4$ 는 $k\times k$ 영역으로 이루어진다. 여기서 x 와 y 의 범위는 $\{((x,y)+r):-1\le r\le 2\}$ 이다.

- 코아

$k\times k$ 윈도우에서 코아는 중심의 2 x 2 픽셀 영역을 말한다. 코아는 $R(x,y,k),k=3,4$ 로 나타낸다. 여기서 x 와 y 의 범위는 $\{((x,y)+r):0\le r\le 1\}$ 이다. Fig.1 에 X 로 표시되어 있다.

- 근접픽셀

$k\times k$ 윈도우에서 코아의 $4(k-1)$ 주변 픽셀을 근접픽셀이라 한다. 이를 $\eta(i)$ 로 나타낸다. 여기서, $i=0,...,4(k-1)-1$ 로 위의 왼쪽 픽셀로부터 시계 방향으로 순서가 메겨진다. Fig.1 (b)에서 0 에서 11 번까지 표시되어 있다.

- 코너

$k\times k$ 윈도우에서 코너는 모서리의 4 개 픽셀을 가리킨다. Fig.1 (b)의 0, 3, 6, 9 번을 나타낸다.

- 측면

측면 픽셀은 코너가 아닌 근접 픽셀을 가리킨다.

- ON 과 OFF

이진 픽셀 값을 가리킬 때 사용된다. ON 값은 전경(foreground) 값을 가리키며, 굵은 선이나 X 픽셀로 표시한다. OFF 값은 배경(background)값을 나타내며 여기서는 빈 영역으로 표시한다. 코아는 코아의 모든 픽셀이 ON 이면 ON 이며, 모든 픽셀이 OFF 이면 OFF 이다. 마찬가지로 측면은 측면의 모든 픽셀이 ON 이면 ON 이며, 모든 픽셀이 OFF 이면 OFF 이다.

- 지움

세선화 과정에서 어떤 픽셀이 지움으로 설정되면 지워지며, 이후 픽셀은 OFF 가 된다.

세선화의 기준을 서술하기 전에 몇몇 변수에 대하여 정의한다.

$\phi_0(\eta)$: 주변픽셀에서 OFF 픽셀로 구성되는 픽셀 체인의 최대 길이.

$\phi_1(\eta)$: 주변픽셀에서 ON 픽셀로 구성되는 픽셀 체인의 최대 길이.

$\chi(\eta)$: 주변픽셀에서 ON 픽셀로 구성되는 픽셀체인의 개수로 이를 코아의 연결성(connectivity)라 한다. ON 픽셀값을 1 로, OFF 픽셀값을 0 으로 할 때 주변픽셀$\eta(i)$에 대한 연결성은 다음 식(1)로 구할 수 있다.

$$\chi(\eta)=\frac{1}{2}\sum_{\substack{i=0\\ \{i,i-1\}\neq n*3}}^{4(k-1)-1}|\eta(i)-\eta(i-1)|+\frac{1}{2}\sum_{m=n*3}|\eta(m+1)-\eta(m-1)|$$

$$+\sum_{m=n*3}\eta(m)\,|\eta(m)-\eta(m-1)\|\eta(m)-\eta(m+1)| \qquad (1)$$

여기서, $n=0,1,2,3$ 이고 $\eta(-1)=\eta(4k-5)$ 이다.

식(1)의 첫번째 항은 코너를 제외한 주변픽셀에서의 ON/OFF 전이를 계산한 것이다. 둘째 항은 코너에서의 전이(ON 에서 OFF 또는 OFF 에서 ON)를 계산한 것이며, 셋째 항은 코너가 ON 일 경우 이중 전이(OFF, ON, OFF 순서)를 계산한 것이다.

세선화 기준은 다음과 같이 나타낼 수 있다.

4 x 4 윈도우에서 코아가 ON 인 경우 다음과 같은 경우에 제거할 수 있다.

$\chi(\eta)=1$: 연결성을 유지하기 위한 조건

$\phi_0(\eta)>2$: 선 끝을 유지하기 위한 조건

$\phi_1(\eta)>2$: OFF 영역의 ON 영역으로의 침투를 막기 위한 조건

3. 주변픽셀 맵

세선화 과정은 주변픽셀의 비트맵을 유지하면 더욱 효과적으로 실행될 수 있다. 이미지를 위에서 아래로, 좌에서 우로 주사(scan)한다고 가정하면, 이 접근 방법은 주변픽셀의 오른쪽 측면을 유지하기 위하여 이미지에 대한 두 가지의 접근이 필요하다. 하나의 접근은 이미지의 아래 오른쪽 픽셀에 대한 것이고, 다른 하나는 이전의 주사선(scanline)에 대한 주변픽셀 맵을 포함하는 주사선 버퍼(scanline buffer)에 대한 것이다. Fig.2 과 같은 주변 픽셀에 대하여 주어진 픽셀(Q)의 픽셀 맵은 식(2)와 같이 계산 과정을 통하여 얻을 수 있게 된다.

w0	w1	w2	w3
x0	x1	x2	x3
y0	y1	y2	y3
z0	z1	z2	z3

4x4 윈도우의 픽셀 주소

= w0w1w2w3 x0x1x2x3 y0y1y2y3 z0z1z2z3(hex)

3x3 윈도우의 픽셀 주소

= w0w1w2 x0x1x2 y0y1y2(hex)

Fig.2 Bit Assignments for neighborhood map encoding

Q = [lshift(p,l)] | [lshift(q,4)] | i(x+2,y+2) (2)

여기서,

Q : 주어진 픽셀의 주변 픽셀 맵

p : 이전 픽셀의 주변 픽셀 맵

q : 바로 위 픽셀의 주변픽셀 맵

i : 아래 오른쪽 픽셀 값

이 연산 과정을 그림으로 표현하면 다음 Fig.3 와 같이 표현된다.

1 0 0 0 인코딩값 : 1000 0111 0111 0111 → 0x8777
0 1 1 1 의미 : 끝점이 아니다.
0 1 1 1 연결성에 영향을 미친다.
0 1 1 1

0 0 0 0 인코딩값 : 0000 0111 0111 0001 → 0x0771
0 1 1 1 의미 : 끝점이 아니다.
0 1 1 1 연결성에 영향을 미치지 않는다.
0 0 0 1

0 1 1 0 인코딩값 : 0110 0111 0111 0001 → 0x6771
0 1 1 1 의미 : 끝점이 아니다.
0 1 1 1 연결성에 영향을 미치지 않는다.
0 0 0 1

Fig.3 Neighborhood map encoding

$$q \quad \begin{matrix} q_0 & q_1 & q_2 & q_3 \\ q_4 & q_5 & q_6 & q_7 \\ q_8 & q_9 & q_{10} & q_{11} \\ q_{12} & q_{13} & q_{14} & q_{15} \end{matrix}$$

$$\begin{matrix} p_0 & p_1 & p_2 & p_3 \\ p_4 & p_5 & p_6 & p_7 \\ p_8 & p_9 & p_{10} & p_{11} \\ p_{12} & p_{13} & p_{14} & p_{15} \end{matrix} \qquad \begin{matrix} p_2 & p_3 & q_2 & q_3 \\ p_6 & p_7 & q_6 & q_7 \\ p_{10} & p_{11} & i_{00} & i_{01} \\ p_{14} & p_{15} & i_{10} & i_{11} \end{matrix}$$

p Q

Fig. 4 Neighborhood map computation

일단 주변 픽셀에 대한 코드가 계산되면, 경계의 방향은 적절한 방향 마스크(3x3 윈도우의 경우, 북측=200, 남측 = 002, 동측=010, 서측 040)와 주변 픽셀맵을 AND 연산하여 쉽게 알 수 있다. 결과가 1 인 경우 그 방향의 픽셀이 1 이며, 따라서 그 방향의 경계점이 아닌 것을 알 수 있다.

Table 1. Direction mask

	4 x 4 윈도우	3 x 3 윈도우
북측	0x6000	0x200
남측	0x0006	0x002
동측	0x0110	0x010
서측	0x0880	0x040

4. I4TM 알고리즘

I4TM 알고리즘은 병렬, 이진값 알고리즘이다. 모든 픽셀들은 연속적으로 계산된다. 하나의 패스에서의 세선화 결과는 이 패스의 세선화 동작에 영향을 미치지 않는다. 이를 위해서는 각각 4 개의 부 사이클로 분리된 반복계산을 하며, 세선화는 4 개의 부 사이클에서 북측, 남측, 동측, 북측 경계인 윈도우에 대하여 적용된다.

윈도우의 둘레 방향을 지정하는데 사용되는 법칙은 다음과 같다.

북측 경계 : OFF 값 만을 가진 북쪽 측면

남측 경계 : OFF 값 만을 가진 남쪽 측면, 윈도우는 북측 경계 윈도우가 아니다.

동측 경계 : OFF 값 만을 가진 동쪽 측면, 윈도우는 북측경계나 남측경계 윈도우가 아니다.

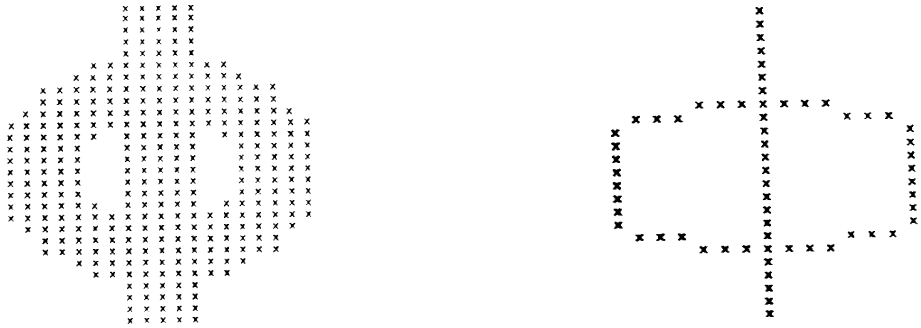
서측 경계 : OFF 값 만을 가진 서쪽 면, 윈도우는 북측경계, 남측경계, 동측경계 윈도우가 아니다.

```
count = 1
while(count) {
    count = 0;
    for(N,S,E,W){
        make Neighborhood address map;
        count += examine the 4x4 window and delete the core;
        count += examine the 3x3 window and delete the pixel;
    }
```

}

Fig.5 I4TM algorithm

이 알고리즘을 이용한 원 이미지에 대한 세선화 결과를 그림으로 표현하면 Fig.6 과 같다.



(a) Orginal image (b) thinned image

Fig.6 Thinning process using I4TM algorithm

```
count = 1
while(count) {
    conut = 0;
    for(N,S,E,W) {
        set the flag at the pixel to be thinned using 4x4 window (scan the whole image);
        set the flag at the pixel to be thinned using 3x3 window(scan the whole image);
        count = delete the pixel flaged(scan the whole image);
    }
}
```

Fig.7 O'Gorman algorithm

5. 결과분석 및 결론

O'Gorman 알고리즘은 모든 방향에 대하여 두 번의 테스트와 한 번의 표시한 픽셀 지우기로 최소한 이미지에 대한 12 번의 참조가 필요하다. 그러나 이 알고리즘은 이미지에 단지 4 번의 참조만이 필요하다. 주변픽셀주소 맵을 만드는 동안 지울 픽셀을 테스트할 수 있기 때문이다. 지울 픽셀을 결정하는 것은 주변픽셀 맵의 정수 값에 의하여 미리 계산된 테이블을 사용한다.

이전 주사선의 주변 픽셀 값을 독립된 버퍼에 저장하고 있고 이 값들은 계속 갱신되기 때문에 코아 값 제거 과정이 즉각적으로 이루어 질 수 있다.

이 알고리즘은 많은 실행시간을 줄일 수 있으나 대신 많은 메모리 공간을 필요로 한다.

Fig.8 은 테스트 이미지와 계산된 결과를 나타낸다. 결과의 성능을 비교하기 위하여 또 다른 두 알고리즘을 구현하였다. Table 2.는 성능 테스트 결과를 나타낸다.

Table 2. Performance comparison

이미지	I4TM	O'Gorman 방법
Fig.8	0.33 초	0.85 초

(a)Original image

(b) Thinned image

Fig.8 Test Image and Thinned Image

참고문헌

(1) Lawrence O'Gorman "k x k Thinning" Computer Vision, Graphics, and Image Processing 51, 1990 pp.195-215

(2) Joseph M. Cychosz "Efficient Binary Image Thinning Using Neighborhood Maps" Academic Press Inc. 1994

(3) Gerhard X.Ritter "Handbook of Computer Vision Algorithm in Image algebra", pp.151-156, 1994

(4) Rafael C. Gonszlez " Digital Image Processing", pp.491-494, 1993

(5) Lawrence O'Gorman "Document Image Analysis", IEEE Computer Society Press 1995, pp.7-11

(6) A. Rosenfeld "A Characterization of parallel Thinning algorithms", Information Control, 29: 286-291,

1975

(7) J.Davidson, “Thinning and skeletonizing: A tutorial and overview”, in Digital Image Processing: Fundamentals and applications, New York: Marcel Dekker, Inc., 1991

(8) B. Jang and R. Chin, “Analysis of thinning algorithms using mathematical morphology”, IEEE Transactions Pattern Analysis and Machine Intelligence, vol.12, pp.541-551, June 1990

(9) L. Lam, S. Lee, and C. Suen, “Thinning methodologies – A Comprehensive survey”, IEEE Transactions Pattern Analysis and Machine Intelligence, vol.14, pp.868-885, June 1992