윤곽선 추적 기반의 세선화 알고리듬 비교

김 기 수, 이 정 환, 심 재 창
안동대학교 컴퓨터공학과
전화 : (0571) 850-5713 / 팩스 : (0571) 850-5480

Comparison of Contour-Tracing Based Thinning Algorithms

Ki Su Kim, Jung-hwan Lee, and Jae Chang Shim
Department of Computer Engineering Andong National Univ.
E-mail : k1g2s3@andong.ac.kr, jhlee@andong.ac.kr, jcshim@andong.ac.kr

Abstract

This paper performed a comparative study on contour tracing based thinning algorithms. These algorithms are widely used for extracting skeleton due to its superiority to other techniques in processing speed to perform comparison. We selected general eight images consisted of fingerprints, characters, and figures. According to experimental results, we compared each algorithm in performance and ability made skeleton.

I. 서 론

세선화는 컴퓨터 기술의 초창기에 형태 분석을 쉽게 할 수 있고 또한 정보의 양을 최소화시켜야 하는 필요성이 제기되면서 나타났다[1]. 세선화는 지문 인식, 문자 인식, 생체 임상 의학 시스템등 많은 이미지 분석 처리에서 중요한 전처리과정으로 사용되어왔다[2].

세선화에 대한 기본적인 요건으로는

- 영상 결과의 두께는 한 화소이고
- 위치는 도형이나 선에서 항상 중심에 위치하며
- 연결성을 유지하고,
- 길이가 계속적으로 줄어들지 않으며
- 입력 영상의 잡음이나 요철이 골격선 유지에 영향을 주어서는 안 된다는 것

등으로 요약될 수 있다[2-3].

세선화의 여러 가지 방법 중에서 윤곽을 이용한 세선화 방법은 영상에서 순차적으로 영상을 검색하면서 윤곽화소를 만나면 그 윤곽을 계속 추적하면서 세선화를 수행하는 방법으로 다른 방법 보다 수행 속도가 빨라 많이 연구되고 있다[2-8].

본 논문에서는 윤곽선 추적 세선화 알고리듬들을 분석하고, 알고리듬간에 수행 속도와 골격선의 완성 정도를 비교하였다.

II. 세선화 알고리듬

세선화를 정의하면 계수화된 입력 패턴의 위상적인 (Topological)특성을 유지하는 골격선(Skeleton)의 추출로서 정의될 수 있다[4]. 주어진 이진 영상을 검사하는 방법에 따라 순차 세선화방식(Sequential thinning)과 병렬 세선화방식(Parallel thinning)으로 크게 나누어진다[1]. 순차 세선화방식은 화소들을 각 반복 단계에서 고정된 순서로 검사하면서 제거하므로, N번째 반복 단계에서 화소의 제거 여부는 현재까지 수행된 모든 연산 결과에 의존한다. 이 방식은 다시 모든 화소를 검사하는 수평 주사(Raster scan) 알고리듬과 윤곽선만을 추적하면서 검사하는 윤곽선 추적(Contour- tracing) 알고리듬으로 다시 나눌 수 있다. 병렬 세선화방식은 반복 단계에서 화소들을 독립적으로 검사하므로, N번째 반복 단계에서의 화소의 제거 여부는 (N-1)번째 반복 단계후의 결과에만 의존한다. 이 방식에는 1, 2, 4-부사이클(subcycle)등의 종류가 존재한다[1].

윤곽선 추적(Contour-tracing) 알고리름은 영상에서 순차적으로 영상을 검색하면서 윤곽화소를 만나면 그 윤곽을 계속 추적하면서 세선화를 수행하는 방법으로

모든 영상 화소를 비교하는 다른 방법보다 수행 속에서 우수하다[2-8].

Ⅲ. 윤곽선 추적 세선화 알고리듬

많은 세선화 알고리듬 연구들은 속도 향상에 초점을 맞추어, 전경 화소(Foreground pixel)와 배경 화소(Background pixel)를 구분 없이 모든 화소를 검사하는 수평 주사 방법보다는 윤곽선만을 검사하는 방법을 대부분 다루었다[2-8].

Pavlidis[3]의 다중화소(multiple)는 윤곽 화소가 순차적으로 탐색되면서 분류되어질 때, 이웃한 화소들에 의하여 쉽게 결정된다. 다음의 조건들 중 하나라도 만족하면 다중화소이다.

- 윤곽 추적 동안에 2번 이상 탐색된다.
- 내부에 이웃한 화소를 갖지 않는다.
- 윤곽선에는 속하지만 바로 앞 또는 뒤에서 탐색되지 않는 최소한 하나의 4-이웃을 갖는다.

윤곽선을 추적하면서 모든 다중화소를 찾고, 그들의 화소 값을 증가시키고, 마지막으로 화소의 값이 2인 모든 화소를 지운다. 그러나 골격선의 연결성이 끊어지는 경우가 있으므로, 이전 추적에서 골격화소로 결정된 8-이웃의 화소가 있거나 바로 앞과 뒤에서 탐색된 화소와 이루는 각이 90 °를 이룬다면 해당 화소는 골격화소로 결정한다. 각 화소에 대한 속성을 다음 집합들로 구분한다.

- S = 골격으로 결정된 화소들
- Q = 세선화가 적용될 화소들
- B = Q 중 윤곽 화소들
- L = B 중 다중 화소가 아닌 화소들
- M = B 중 다중 화소들
- K = B 중 S에 8-이웃한 화소가 있는 화소들

알고리즘 1. A thinning algorithm for discrete binary images[3].

STEP 1. S를 초기화
STEP 2. Q에 화소가 없을 때까지 STEP 1 - 7 수행
STEP 3. 윤곽선 추적 알고리즘을 이용하여 B를 생성
STEP 4. 다중화소 조건으로 B를 재조사하여 L 과 M 을 생성
STEP 5. B를 조사하여 K를 생성
STEP 6. S = S ∪ M ∪ K
STEP 7. Q = Q - B

이기중[4]의 알고리듬은 윤곽선을 찾아 추적하는 부분과 찾아진 윤곽선을 제거하는 부분으로 크게 나눌 수 있다. 윤곽선의 시작점을 찾기 위해서 한 번의 수평 주사를 행한다. 표 2는 윤곽선 추적 알고리듬이다. 윤곽선 추적을 하면서 입력 영상의 골격선을 산출하기 위하여 채택한 조건들은 아래와 같다. 윤곽선 상의 화소가 아래 조건을 모두 만족하지 않으면 해당 화소는 제거된다.

- 연결성이 존재한다. 즉, 그림 1의 창틀 중에서 하나라도 만족하는 화소는 연결성이 있다. X, Y는 적어도 하나는 양수이고, N은 '0' 아닌 값을 가져야 한다.
- 8-이웃에서 '1'화소가 하나이다.

X	X	X
0	P	0
Y	Y	Y

X	0	Y
X	P	Y
X	0	Y

0	P
N	0

P	0
0	N

0	N
P	0

N	0
0	P

그림 1. 이기중[4]의 연결성 검사용 창틀

Fig 1. Connectivity condition templates in Lee[4]

알고리즘 2. A thinning algorithm using contour-tracing for binary images[4].

STEP 1. 영상을 왼쪽에서 오른쪽으로 주사하던 중 처음 만나는 '1'화소를 시작점으로 한다. 시작점은 다음의 두 조건을 만족해야 한다.
1.1 4-이웃 중에 '0'인 화소가 있다.
1.2 전에 방문되지 않았다.
STEP 2. 시작점이 다음을 만족하면 홀이고, 그렇지 않으면 바깥 층이다.
2.1 4-이웃 중에 전에 방문된 화소가 있다.
STEP 3. 바깥 층인 경우에는 아래를 수행한다.
3.1 이전 방향은 P5이다.
3.2 이전 방향으로부터 시계 방향으로 두 번째 화소부터 검사한다.
3.3 8-이웃을 반시계 방향으로 검사한다.
3.4 '1'인 화소를 만나면 그 방향이 다음 윤곽선이다.
STEP 4. 홀인 경우에는 아래를 수행한다.
4.1 이전 방향은 P1이다.
4.2 이전 방향으로부터 시계 방향으로 두 번째 화소부터 검사한다.
4.3 8-이웃을 시계 방향으로 '1'화소를 검사한다.
4.4 '1'인 화소를 만나면 그 방향이 다음 윤곽선이다.
STEP 5. 시작점이고 8-이웃 중에 더 이상 방문할 화소가 없을 때까지 STEP 3 이하를 수행한다.
STEP 6. 모든 화소를 방문하기 위해 STEP1 이하를 수행한다.

Liu[5]의 가장 핵심 부분은 현재 가장 바깥쪽의 골격이 아닌 화소를 지운다면, 그 화소의 근처에 있는 내부 화소가 다음의 제거 단계 수행 때 가장 적당하다는 것에 의해 새로운 연속적인 윤곽선을 생성한다. 이 과정은 모든 골격 화소를 찾을 때가지 계속된다. 그림 2는 제거 조건에 쓰이는 창틀이다. A, B는 적어도 하나 이상은 '0'이 아니어야 하며, 90° 씩 변화된 형태도 고려해야 한다.

알고리즘 3. A new contour coherence based thinning algorithm[5].

STEP 1. 입력 영상을 수평 주사한다. 바깥 윤곽과 내부 윤곽을 포함하여 8방향 체인 코드를 이용하여 윤곽선을 추적한다. 동시에 STEP 2 - 4를 수행
STEP 2. 4-이웃 중에서 '0'을 가지는 가장 작은 방향(d)을 결정
STEP 3. 해당 화소(q)의 좌표를 $S_x[d/2][i]$, $S_y[d/2][i]$에 저장

STEP 4. $S_t[d/2]+1$, q = 255
STEP 5. $S_t[4] = S_t[0] + S_t[1] + S_t[2] + S_t[3]$
STEP 6. $S_t[4] < 1$ 까지 STEP 7 - 14 수행
STEP 7. d = 0, 1, 2, 3 각각 STEP 8 - 14를 수행
STEP 8. i = 0 에서 $S_t[d]$ 까지 STEP 9를 수행
STEP 9. 그림 2의 조건에 맞는다면 q = 2, 모두 맞지 않다는면 q = 3
STEP10. i = 0 에서 $S_t[d]$ 까지 STEP11-14를 수행
STEP11. 만일 q = 3 이면 STEP12-13을 수행
STEP12. q = 0
STEP13. q의 4-이웃 중에 1을 가지고 있는 화소(q')가 있다면 STEP 2 - 4 을 수행
STEP14. $S_t[4] = S_t[0] + S_t[1] + S_t[2] + S_t[3]$

A	A	A
0	P	0
B	B	B

A	A	A
A	P	0
A	0	B

그림 2. Liu[5]의 연결성 검사용 창틀

Fig 2. Connectivity condition templates in Liu[5].

신웅철[6]에서는 영상에 대해 외부 윤곽 추적, 영역 채색 및 내부윤곽 검색, 화소제거 단계를 거치면서 세선화 된 영상을 얻는다. 한 영역을 만나면 먼저 외부 윤곽을 반 시계 방향으로 추적하면서 윤곽화소의 위치를 저장한다. 내부 윤곽의 유무를 검사하기 위해 영역을 채색하며, 내부 윤곽을 만나면 시계방향으로 윤곽을 추적하면서 윤곽 좌표를 저장한다. 이때 처리는 외부 윤곽의 처리와 내부 윤곽의 처리는 방향만 반대일 뿐 나머지 처리는 동일하다. 처리 알고리듬은 한 영역에서 외부 및 내부 윤곽이 검색되면 저장된 윤곽 좌표에 차례로 세선화 조건을 적용한다.

알고리즘 4. A contour-tracing algorithm using templates[6].

STEP 1. 입력 영상을 왼쪽에서 오른쪽으로 위에서 아래로 순차 주사
STEP 2. '1' 화소를 만나면
  2.1 반시계 방향으로 8 방향 추적 법으로 추적하며 윤곽정보 부여 및 좌표값 저장
  2.1.1 체인 코드 방향이 아래로 향하면 체인 코드 도착지점의 화소 값을 'S'를 부여
  2.1.2 체인 코드 방향이 위로 향하면 체인 코드 도착지점의 화소 값을 'E'를 부여
  2.2 시작 지점으로 되돌아오면 왼쪽 아래 방향의 대각 연결 영역 확인
  2.2.1 영역이 존재하면 계속하여 STEP 1 수행
  2.2.2 영역이 없으면 STEP 2 수행
STEP 3. 영역의 가장 윗줄부터 가장 아래 줄까지 각줄 마다 'S'에서 'E'사이를 블럭 번호로 채움
  3.1 'S'에서 'E'사이에 '0'인 값을 만나면 채움일시 중단
  3.1.1 위 주사선 동일 위치부터 시계 방향으로 윤곽 추적 및 좌표 값 저장('S', 'E', 블록번호부여는 STEP 2와 동일)
  3.1.2 시작지점으로 되돌아오면 바로 이전 화소지점부터 블록번호를 1증가시키고, STEP 1 수행
STEP 4. 화소 제거는 저장된 좌표 값을 순차적으로 방문하면서 Zhang-Suen의 화소제거 조건과 Holt의 계단 모형 제거 조건을 이용하여 세선화
STEP 5. 채색된 값을 모두 영역 값으로 변환
STEP 6. 제거된 화소가 하나라도 존재하면 STEP 1-5를 반복

Shim[7]은 Vossepoel[8]이 안고 있는 시작점 처리, 중심 골격 선으로의 세선화, 그리고 4개의 90° 코너 처리 문제 등을 개선한 알고리즘이다. 그리고 Vossepoel[8]에서 제시한 화소 삭제 테이블을 수정하여, 일어나지 않는 경우를 제외하므로 수행 속도를 향상시켰다.

알고리즘 5. Contour following thinning using binning[7].

STEP 1. 모든 영상의 화소를 모두 순차 주사할 때까지 STEP 2 - 5 수행
STEP 2. '1'화소를 만나면 외곽 선을 추적한다.
STEP 3. 영역 채색 및 내부 윤곽을 검색한다.
STEP 4. 윤곽 화소를 8가지로 분류하여 각각의 버퍼에 저장한다.
STEP 5. 8-이웃 화소를 기준으로 윤곽 화소를 지운다.

Ⅳ. 실험 및 고찰

본 논문에서는 실험을 위하여 지문, 문자와 도형, 문서 영상으로 구성하였다. 구성된 모든 영상은 세선화가 쓰여졌던 분야 중에서 가장 일반적이고 보편적인 영상들을 선택했으며, 이진화가 수행된 영상들이다. 각각의 크기는 지문영상 280 x 280, 문자와 도형 256 x 256, 문서 영상 1600 x 2324를 가진다. 실험을 수행한 환경은 Intel PentiumII Celeron333A, 128M RAM, Windows98,Visual C++6.0 이다. 각 알고리듬의 수행 시간과 결과 영상에 대한 평가는 표 1-2와 같다.

Pavlidis[3]은 작은 영상에서는 처리속도가 빠르나 큰 영상에는 처리속도가 늦어지고, 홀의 윤곽선 탐색과 두께가 두 줄인 영상에는 바른 처리를 하지 못했다. 이기중[4]는 처음 찾았던 윤곽 화소의 시작점을 기억하고 있으면서 윤곽 추적을 하므로, 상당한 용량의 기억 장소를 절약했다. Liu[5]는 단 한번의 윤곽선 추적만을 수행하고, 윤곽 화소가 지워지면 4-이웃 화소를 이용하여 다시 윤곽선을 구성하므로 적은 소요시간을 나타냈다. 신웅철[6]은 수행 시간에서도 월등했으며, Liu[5]에서 나타난 두께가 두 줄에서 한 가닥씩 나누어지는 경우에 세선화에서 누락되는 문제를 Holt[9]의 전처리 알고리즘을 수행하여 해결하였다. Shim[7]의 결과는 시간적인 면에서 가장 우수한 결과를 나타냈으며, 출력된 결과에서도 다른 전처리나 후처리 알고리즘을 사용하지 않고도 한 화소 두께의 세선화 결과를 나타냈다.

Ⅴ. 결 론

본 논문에서는 윤곽선 추적 세선화 알고리듬들을 분석하고, 수행 속도와 골격선의 완성 정도를 알고리

듬간에 비교하였다. 각기 논문에 의거하여 알고리듬을 기술하였고, 다양한 입력 영상들을 처리하여 결과를 비교하였다. 수행 속도와 골격화 정도라는 양쪽을 모두 고려했을 때에 Shim[7]이 가장 우수하다는 결론을 내렸다.

표 1. 알고리듬들의 수행시간

Table 1. Execution time of algorithms.

(unit: sec)

Algorithm / Image	Pavlidis [3]	이기중 [4]	Liu[5]	신웅철 [6]	Shim[7]
General-1	0.17	0.90	0.11	0.11	0.05
General-2	0.17	0.54	0.11	0.17	0.06
Fingerprint-1	0.22	2.42	0.22	0.33	0.16
Fingerprint-2	0.17	3.24	0.17	0.39	0.17
Fingerprint-3	0.22	1.60	0.16	0.33	0.22
Fingerprint-4	0.22	1.98	0.21	0.33	0.16
Large text-1	6.48	15.0	1.48	2.36	0.88
Large text-2	6.26	16.2	1.59	2.63	0.99

표 2. 알고리듬들의 세선화 결과 평가

Table 2. Evaluation of each algorithm's thinning quality in the result images.

(O : 양호 △ : 보통 × : 실패)

Algorithm / Image	Pavlidis [3]	이기중 [4]	Liu[5]	신웅철 [6]	Shim[7]
General-1	×	O	O	O	O
General-2	×	O	O	O	O
Fingerprint-1	×	△	O	O	O
Fingerprint-2	△	△	O	O	O
Fingerprint-3	O	△	O	O	O
Fingerprint-4	O	△	O	O	O
Large text-1	△	△	△	O	O
Large text-2	△	△	△	O	O

참고문헌

[1] Louisa Lam, Seong-Whan Lee and Ching Y. Suen, "Thinning Methodologies - A Comprehensive Survey", *IEEE Trans. on PAMI*, Vol. 14, No. 9, Sep. 1992, pp. 869-885.

[2] Paul C. K. Kwok, "A Thinning Algorithm by Contour Generation", *Commun. ACM*, Vol. 31, No. 11, Nov. 1988, pp. 1314-1324.

[3] T. Pavlidis, "A Thinning Algorithm for Discrete Binary Images", *Comput. Graphics Image Proc.* 13, 1980, pp. 142-157.

[4] 李起仲, "경계선 추적을 이용한 이진 화상의 순차 세선화 알고리즘", 忠南大學校 大學院 碩士學位論文, 1989. 10.

[5] Liu Xiao-Lin and Xie Dong, "A New Contour Coherence Based Thinning Algorithm", *IEEE TENCON'93 Beijing*, pp. 631-635.

[6] 申雄澈, "창틀을 이용한 윤곽추적 세선화 방법", 安東大學校 大學院 碩士學位論文, 1998. 12.

[7] J. C. Shim, C. Dorai and J. J. Lee, "Contour following thinning using binning", IBM Technical Report, Aug. 1999.

[8] A. M. Vossepoel, J. P. Bys and G. Koelewijn, "Skeletons from chain-coded contours", IEEE, 1990, pp. 70-74.

[9] Holt, C. M., Stewart, A., Clint, M. and R. H. Perrott. "An Improved Parallel Thinning Algorithm", *Commun. ACM*, Vol. 30, No. 2, 1987, pp. 156-160.

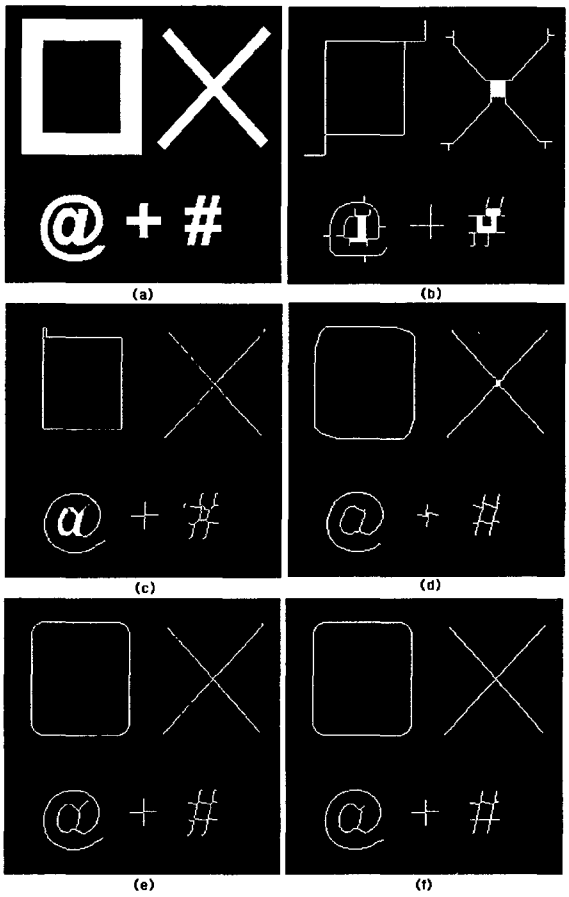

그림 3. 세선화 알고리즘들의 결과 영상

(a) 원영상 (b) Pavlidis[3]의 결과 (c) 이기중[3]의 결과 (d) Liu[5]의 결과 (e) 신웅철[7]의 결과 (f) Shim[8]의 결과

Fig 3. A result images of thinning algorithms

(a) Origin imgae (b) Pavlidis[3]'s result (c) Lee[4]'s result (d) Liu[6]'s result (e) Shin[7]'s result (f) Shim[8]'s result