임의 형상의 윤곽선 시퀀스 정보로부터 형상 특징의 효율적인 연산 방법

김성옥[†] · 김동규[††] · 김민환[†††]

요 약

윤곽선 시퀀스는 임의 형상을 간단하면서도 정확하게 표현할 수 있는 좋은 표현법이 될 수 있다. 그러나, 형상을 구성하는 화소로부터 쉽게 구할 수 있는 면적, 무게중심, 오리엔테이션 방향, 투영 히스토그램 등과 같은 형상 특징들을 윤곽선 시퀀스로부터 직접 구하기는 어렵기 때문에, 윤곽선 시퀀스를 임의 형상에 대한 표현법으로 잘 사용하지 못하였다. 본 논문에서는 형상 내부의 연속된 화소들로 구성된 수직(또는 수평)의 라인 세그먼트를 의미하는 크로스 섹션 개념을 이용하여, 윤곽선 시퀀스로부터 형상 특징들을 쉽게 구할 수 있음을 보인다. 윤곽선 시퀀스를 한번 순차적으로 탐색함으로써 크로스 섹션을 효율적으로 구할 수 있는 방법을 제안한다. 또한, 이진 영상으로부터 여러 형상의 윤곽선 시퀀스를 자동으로 추출할 수 있는 효율적인 방법도 함께 제안한다. 제안된 방법들은 형상 내부에 홀(hole)이 있는 경우에도 적용할 수 있다. 결과적으로, 윤곽선 시퀀스가 임의 형상 영역에 대한 매우 효과적인 표현이 될 수 있음을 밝힌다.

An Efficient Shape-Feature Computing Method from Boundary Sequences of Arbitrary Shapes

Seongok Kim[†], Dongkyu Kim[††] and Minhwan Kim[†††]

ABSTRACT

A boundary sequence can be a good representation of arbitrary shapes, because it can represent them simply and precisely. However, boundary sequences have not been used as a representation of arbitrary shapes, because the pixel-based shape-features such as area, centroid, orientation, projection and so forth, could not be computed directly from them. In this paper, we show that the shape-features can be easily computed from the boundary sequences by introducing the cross-sections that are defined as vertical (or horizontal) line segments in a shape. A cross-section generation method is proposed, which generates cross-sections of the shape efficiently by tracing the boundary sequence of the shape once. Furthermore, a boundary sequence extraction method is also proposed, which generates a boundary sequence for each shape in a binary image automatically. The proposed methods work well even if a shape has holes. Eventually, we show that a boundary sequence can be used effectively for representing arbitrary shapes.

Key words: Shape feature, boundary following, connected component labeling

1. 서 론

물체 형상에 대한 표현 및 분석은 영상처리 및 패턴인식 분야에서 매우 중요한 연구주제로서, 좋은 형상 표현법은 형상 데이터에 대한 효율적인 압축 및 전송을 가능하게 하며, 형상 비교 및 인식에 효과적이다. 이에, 형상에 대한 표현법으로서 형상 영역을 나타내기 위한 멤버쉽 배열(membership array), 피라미드(pyramids), 사분트리(quadtree) 등이 있으며, 윤곽선(boundary)을 표현하기 위한 체인 코드(chain code), Ψ-s 플롯(Ψ-s plot), 경사밀도(slope density)

[†] 종신회원, 한국신발피혁연구소 자동화연구부 연구부장
[††] 부산대학교 정보컴퓨터공학부 조교수
[†††] 종신회원, 부산대학교 컴퓨터공학과 교수

함수, 각종 근사화 (approximation) 방법 등이 개발되었다. 그러나, 형상에 대한 좋은 표현법이 되기 위해서는 다음과 같은 조건을 잘 만족하도록 설계되어야 한다[1].

(1) 간단하면서도 압축된 표현이 가능해야 함.

(2) 영역을 정확하게 표현할 수 있어야 함.

(3) 차후 응용 연산에 효과적이어야 함.

먼저 위에서의 영역 표현법은 압축된 표현을 제대로 제공하지 못할 뿐만 아니라, 차후의 응용 연산에서도 효과적이지 못하다. 한편, 윤곽선 표현 기법은 일반적으로 영역 표현법에 비해 단순하면서도 압축된 표현이 가능하나, 차후 응용 연산에 효과적으로 사용하기가 어렵다. 예를 들어, 윤곽선 정보로부터 주어진 영역에 대한 면적, 무게중심, 오리엔테이션 (orientation) 방향, 투영 히스토그램 등을 직접 구하기가 어려운 것으로 알려져 있다.

근래에는 모폴로지(morphology) 연산을 통해 형상의 영역을 간단한 구조적 구성요소의 조합으로 구조적인 표현을 할 수 있는 기법들이 개발되어 발표되고 있다[2-4]. 여기에는, 형상을 구성하는 화소(pixel)를 표현하는데 있어, 구조적 구성요소에 의한 중복 표현이 허용되는가 여부에 따라 모폴로지 골격 변환 (Morphological Skeleton Transform, MST) 방법과 모폴로지 형상 분할 (Morphological Shape Decomposition, MSD) 방법으로 구분된다[3]. 이 방법들은 일종의 영역 표현법으로서, 수학적인 근거에 기반을 두고 있으면서도 매우 효율적인 압축 표현을 제공한다. 또한, 형상의 개략화(abstraction)된 표현도 허용한다. 그러나, 형상 분석을 위해 필요한 특징(feature)들, 예를 들어 무게중심, 오리엔테이션 방향, 투영 히스토그램, 둘레 길이, 오일러 넘버 등과 같은 것을 주어진 표현으로부터 바로 구할 수가 없다. 이것은 구조적 구성요소에 의한 화소의 중첩 표현과 구조적 구성요소 간의 경계 표현의 부적절성으로부터 기인한다. 또한, 변환 시간에 대한 부담이 많아, 최근에는 신속한 변환 기법에 대한 연구결과도 제안되고 있다[4].

본 논문에서는, 형상을 나타내는데 있어 인간에게 보다 자연스럽게 받아들여지는 윤곽선 정보를 표현하되, 단점으로 지적되는 형상 특징 연산의 비효율성을 개선시킬 수 있는 방법을 제안한다. 윤곽선 정보는 윤곽선 구성 화소들을 순서대로 나열해 표현한 윤곽선 시퀀스(boundary sequence) 표현법을 사용한다. 이것은 형상에 대해 8-방향 윤곽선 추적 방법[1]을 통해 쉽게 구해지는 것으로서, 체인 코드의 보다 직접적인 표현 방법이라고 볼 수 있다. 따라서, 체인 코드 표현법과 마찬가지로 형상의 이동이나 회전에 무관하게 동일한 표현법을 제공할 수 있다. 본 연구에서는, 이러한 윤곽선 시퀀스로부터 화소 위치 정보를 순차적으로 한번 탐색하면서 크로스 섹션 (cross-section)을 생성하여 형상 특징을 직접 효율적으로 구할 수 있는 방법을 제안한다. 크로스 섹션은 주어진 영역 내의 연속된 화소들로 이루어진 하나의 라인 세그먼트(수직 또는 수평)를 나타낸다. 이에 따라, 형상에 대한 윤곽선 표현법이 실제적으로 매우 유용한 것임을 보이고자 한다.

또한, 하나의 이진 영상에서의 각 형상에 대한 윤곽선 시퀀스를 자동적으로 생성해 내는 효율적인 방법도 제안한다. 이 방법에서는 주어진 이진 영상을 래스터 스캐닝 방법으로 한번 살펴 보면서, 윤곽선 추적 방법[1]을 활용하여 모든 윤곽선 시퀀스를 자동으로 생성해낼 수 있다. 이 방법은 일종의 연결요소 레이블링(connected component labelling) 방법[1, 5,6]으로 볼 수 있으며, 잡영 가지(parasitic branch)가 있거나 기형적으로 생긴 복잡한 형상에 대해서도 적용할 수 있다. 또한, 영역 내부에 홀(hole)이 있을 경우에는, 영역의 외부 윤곽선 시퀀스 뿐만 아니라 홀의 윤곽선에 해당하는 내부 윤곽선 시퀀스도 동시에 생성해낼 수 있다.

본 논문의 2장에서는 크로스 섹션을 정의하고, 3장에서는 이러한 크로스 섹션들로부터 형상 특징들을 효율적으로 계산하는 방법을 보인다. 4장에서는 윤곽선 시퀀스로부터 크로스 섹션을 효율적으로 생성하는 알고리즘을 제시하고, 5장에서는 이진 영상으로부터 윤곽선 시퀀스를 자동으로 생성하는 방법을 소개한다. 마지막으로, 6장에서는 결론을 맺고 향후과제를 제시한다.

2. 크로스 섹션의 정의

만약 영상에서 하나의 영역이 두개의 함수 즉, 그림 1(a)에서와 같이 연속하는 상위 함수 $f(x,y)$와 하위 함수 $g(x,y)$로 구성되어 있다면 영역에 대한 면적 A는 식 (1)과 같이 쉽게 계산됨을 알 수 있다.

$$A = \int_{x_{\min}}^{x_{\max}} (f(x) - g(x))\,dx = \int_{x_{\min}}^{x_{\max}} v(x)\,dx \qquad (1)$$

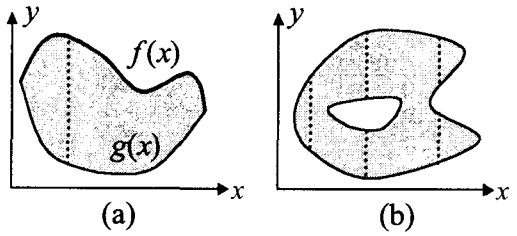

그림 1. 영상에서의 크로스 섹션 형태

그림 1(a)의 점선을 하나의 크로스 섹션이라고 정의하면, 영역의 면적은 영역 내의 각 열에서의 크로스 섹션들의 길이를 적분하여 나타낼 수 있음을 알 수 있다. 그림 1(b)에서와 같이 내부에 홀을 가지고 있거나 영역의 형상이 매우 복잡한 경우에도 크로스 섹션의 길이를 모두 합함으로써 면적을 계산해 낼 수 있다.

디지털 영상에서의 크로스 섹션을 정의하기 위해, 먼저 영역 R과 이것의 윤곽선 시퀀스를 식 (2)와 같이 정의한다.

$$R = \{(x,y) | (x,y) \in region\}$$
$$BS(R) = \{(x(u), y(u)), u = 0,1,\ldots,n-1\} \quad (2)$$

여기에서, $|x(k)-x(k-1)| \le 1$, $|y(k)-y(k-1)| \le 1$ 이다.

윤곽선 시퀀스로부터 크로스 섹션 $C(x, y_l : y_u)$은 식 (3)과 같이 정의한다.

$$C(x, y_l : y_u) = \left\{ (x,y) \middle| \begin{array}{l} y_l \le y \le y_u, (x,y) \in R, \\ (x, y_l) \ and \ (x, y_u) \in BS(R) \end{array} \right\} \quad (3)$$

여기에서, (x,y_l)과 (x,y_u)는 각각 크로스 섹션의 하단점(lower end-point)과 상단점(upper end-point)을 나타낸다.

따라서, 영역 R에 대한 크로스 섹션들의 집합을 Ω라고 하면, 그 영역은 식 (4)에서와 같이 크로스 섹션의 합집합으로 표현할 수 있다.

$$R = \bigcup_{C(x,y_l:y_u) \in \Omega} C(x, y_l : y_u) \quad (4)$$

3. 크로스 섹션을 이용한 영역의 형상특징 계산

다음과 같은 영역의 형상 특징들은 크로스 섹션을 사용하지 않고도 쉽게 계산할 수 있다.

• 바운딩 박스(bounding box)와 가로/세로비(aspect ratio)[7]

바운딩 박스는 윤곽선 시퀀스를 구성하는 화소의 좌표값으로부터 쉽게 결정할 수 있는 x_{min}, x_{max}, y_{min}, y_{max}로 표현할 수 있으며, 가로/세로비는 바운딩 박스의 폭(width=$x_{max}-x_{min}+1$)에 대한 높이(height=$y_{max}-y_{min}+1$)의 비로 나타낼 수 있다.

• 둘레 길이 (perimeter)

둘레 길이 P는 여러 가지 방법으로 구할 수 있지만[1], 일반적인 표현으로 윤곽선 시퀀스에서의 화소 개수로 표현할 수 있다.

• 오일러 넘버 (Euler number)

오일러 넘버는 일반적으로 연결요소의 개수에서 홀의 개수를 뺀 것으로 정의한다. 그런데, 어떤 한 영역 R에 대한 연결요소는 하나이므로, 홀의 개수를 $N(H)$라 하면 오일러 넘버는 $E = 1 - N(H)$ 로 나타낼 수 있다.

영역을 구성하는 전체 화소로부터 결정되는 다음과 같은 형상 특징들은 크로스 섹션을 이용하여 간단하게 계산할 수 있다. 크로스 섹션 $C(x,y_l:y_u)$의 길이 $l(x,y_l:y_u)$는 식 (5)와 같다.

$$l(x, y_l : y_u) = y_u - y_l + 1 \quad (5)$$

• 면적

$$A = \sum_{C(x,y_l:y_u) \in \Omega} l(x, y_l : y_u)$$

• 간결도 (compactness[1] or thinness ratio[4])

$$T = P^2 / A$$

• 무게중심 (centroid)

$$\bar{x} = \frac{1}{A} \sum_{C(x,y_l:y_u) \in \Omega} x\, l(x, y_l : y_u)$$
$$\bar{y} = \frac{1}{2A} \sum_{C(x,y_l:y_u) \in \Omega} (y_l + y_u) l(x, y_l : y_u)$$

여기서 $\bar{x}$는 간단히 구해지고 $\bar{y}$는 아래 식 (6)에서 $(y_u - y_l)$대신에 크로스 섹션의 길이인 $l(x, y_l : y_u)$을 대입함으로써 얻어진 것이다.

$$\bar{y} = \frac{1}{A} \sum_{C(x,y_l:y_u) \in \Omega} \{y_l + (y_l + 1) + \ldots + y_u\}$$
$$= \frac{1}{A} \sum_{C(x,y_l:y_u) \in \Omega} \frac{(y_l + y_u)(y_u - y_l)}{2} \quad (6)$$

• 수직 투영 (vertical projection)

$$V[i] = \sum_{C(x,y_l:y_u)\in\Omega \& x=i} l(x, y_l : y_u)$$

• 수평/대각선 투영 (horizontal and diagonal projections)

수평 크로스 섹션 $C_h(y,x_l:x_r)$을 2장에서의 크로스 섹션과 유사하게 식 (7)과 같이 정의하면, 수평 투영은 식 (8)과 같이 기술할 수 있다.

$$C_h(y, x_l : x_r) = \left\{ (x,y) \middle| \begin{array}{l} x_l \le x \le x_r, (x,y) \in R, \\ (x_l, y) \; and \; (x_r, y) \in BS(R) \end{array} \right\} \quad (7)$$

$$H[i] = \sum_{C_h(y,x_l:x_r)\in\Omega_h \,\&\, y=i} l_h(y, x_l : x_r) \quad (8)$$

대각선 투영도 유사하게 정의되는 대각선 크로스 섹션으로부터 구할 수 있다.

• 런 길이 코드 (run length code)

영역의 한 행에 대한 하나의 런은 수평 크로스 섹션을 이용하여 (x_l, $x_y - x_l + 1$)로 나타낼 수 있고 한 행에 여러 개의 런이 존재할 경우에도 반복적 표현으로 가능하며, 전체적인 런의 개수는 수평 크로스 섹션의 개수와 같게 된다.

• 오리엔테이션(orientation) 방향

어떤 긴(elongated) 형태의 형상에 대해서 최소관성을 갖는 축을 그 형상의 오리엔테이션을 나타내는 것으로 정의하는데(그림 2), 그 축의 방향 θ는 다음의 a, b, c 세 요소에 의해 결정되며[1,7], 크로스 섹션을 이용하여 계산이 가능하다.

$$\tan 2\theta = b/(a-c)$$

$$a = \sum_{(x,y)\in R} (x-\bar{x})^2 = \sum_{C(x,y_l:y_u)\in\Omega} (x-\bar{x})^2 l(x, y_l : y_u)$$

$$b = 2\sum_{(x,y)\in R} (x-\bar{x})(y-\bar{y})$$

$$= \sum_{C(x,y_l:y_u)\in\Omega} (x-\bar{x})[|n_u|(|n_u|+1) - |n_l|(|n_l|+1)]$$

$$c = \sum_{(x,y)\in R} (y-\bar{y})^2 = \frac{1}{6} \sum_{C(x,y_l:y_u)\in\Omega} \begin{bmatrix} n_u(|n_u|+1)(2|n_u|+1) \\ -n_l(|n_l|+1)(2|n_l|+1) \end{bmatrix}$$

$$n_u = y_u - \bar{y}, \quad n_l = y_l - \bar{y}$$

4. 크로스 섹션 생성

4.1 크로스 섹션 생성의 기본 개념

어떤 영역에 대한 모든 크로스 섹션은 이 영역에 대한 윤곽선 시퀀스를 순차적으로 한번 살펴보면서 계산할 수 있다. 윤곽선 시퀀스는 영역의 윤곽선을 시계방향으로 추적하면서 추출한 것으로 가정한다. 예를 들어, 그림 3에서와 같이 홀을 내포하는 영역은 외부 윤곽선 시퀀스와 내부 윤곽선 시퀀스로 구성되며, 각각 시계방향 순서로 좌표값들이 나열된다. 여기에서, x값이 a, b, c일 때의 크로스 섹션은 2장에서의 정의에 따라 그림 3에서와 같이 만들어진다.

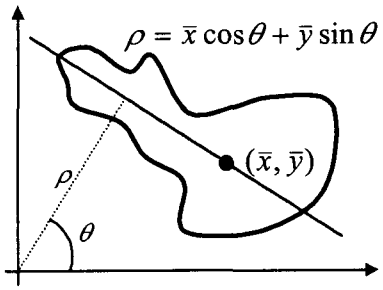

그림 2. 형상의 최소 관성 축 방향

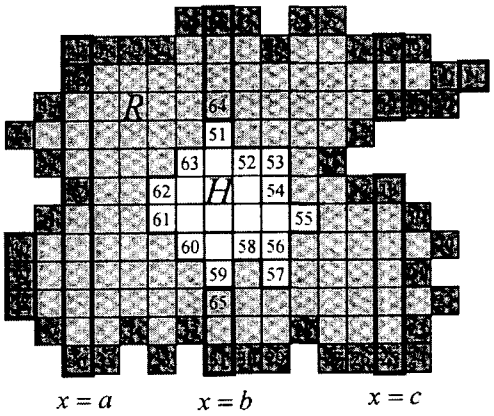

그림 3. 여러 가지 형태의 크로스 섹션들

윤곽선 시퀀스의 각 좌표값을 순차적으로 읽어 들여 크로스 섹션의 상단점과 하단점을 결정하기 위해서는 각 점들이 이동하는 진행방향에 대한 정보가 매우 유용하게 사용된다. 어떤 한 점의 진행방향은 이전 점과의 관계로부터 좌우상하 네 방향으로 결정할 수 있다. 먼저, 좌우 진행방향의 변화를 분석하여 크로스 섹션의 상단점 및 하단점을 결정할 수 있다. 그림 4는 진행방향의 변화를 나타내는 대표적인 6가지의 예를 보이고 있다.

그림 4에서 T_1과 T_2는 진행방향이 바뀌지 않고 이

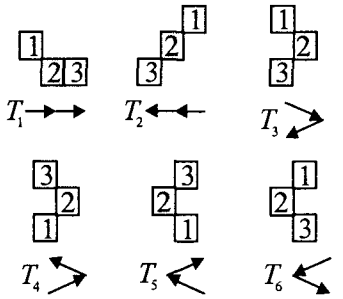

그림 4. 진행방향 변화의 대표적인 예

동되는 것을 보여준다. 이때의 가운데 점 p_2는 T_1에서는 크로스 섹션의 상단점(예, 그림 3의 8번 점)으로 결정하고, T_2에서는 하단점(예, 그림 3의 14번 점)으로 결정한다. 그림 4의 T_3, T_4, T_5, T_6 은 진행방향이 바뀌는 대표적인 경우로서, T_3 의 p_2는 우 극단점, T_5의 p_2 는 좌 극단점으로서 그 자체가 크로스 섹션이 된다. 그림 3에서의 20번 점과 44번 점이 각각 이 경우에 해당한다. 그러나, T_4와 T_6에서 p_2는 각각 그림 3에서의 42번 점과 16번 점에 해당하는 것으로서, 크로스 섹션의 상단점 또는 하단점이 될 수 없다. 여기에서, T_3과 T_5의 경우는 점의 이동방향이 지역적으로(locally) 시계 방향이며, T_4와 T_6의 경우에는 반시계 방향임을 알 수 있다. 시계/반시계 이동방향의 결정은 극단점의 이전 점과 다음 점의 y값 차이로부터 쉽게 결정할 수 있다. 진행방향이 좌에서 우로 변하면서 시계 방향의 이동 경우(T_3)에는 y값 차이가 양수이고, 진행방향이 우에서 좌로 변하면서 시계 방향의 이동 경우(T_5)에는 y값 차이가 음수이다.

진행방향이 상하 방향인 경우에는, 수직 이동이 시작되기 바로 직전에서의 진행방향에 따라 상단점과 하단점을 결정하는 방법을 달리 해야 한다. 직전 진행방향이 오른쪽인 경우에는 수직 이동이 종료된 후의 제일 상위점(그림 3의 47번 점)이 상단점이 되며, 왼쪽인 경우에는 제일 하위점(그림 3의 24번 점)이 하단점이 된다.

한편, 그림 4의 T_3 또는 T_5와 같은 경우이면서 수직 이동이 있을 경우(그림 3의 38-40번 점)에는, 제일 상위점과 하위점이 해당 크로스 섹션의 상단점과 하단점이 된다.

4.2 크로스 섹션 생성 알고리즘

위에서 설명한 크로스 섹션 생성에 대한 알고리즘을 정리하면 다음과 같다.

// input : a boundary sequence for a region, R
$BS(R)=\{(x_0,y_0),(x_1,y_1),..,(x_{n-1},y_{n-1})\}$ //

1. Trace $BS(R)$ until a point (x_i, y_i) whose x-difference Δx_i $(= x_i - x_{i-1}) \neq 0$ is found. If there is not such a point, Insert_CS-list(x_0, min$\{y_i \mid (x_i, y_i) \in BS(R)\}$) and Insert_CS -list($x_0$, max$\{y_i \mid (x_i, y_i) \in BS(R)\}$). Then, exit.
2. Set the direction flag $F_d = \Delta x_i$ and the starting point $s = (x_i, y_i)$. Set $Y_{pre} = y_i$ and $Y_{current} = y_i$.
3. Set $j = (i+1 \bmod n)$ and move to next point (x_j, y_j). Compute Δx_j $(= x_j - x_i)$.
   3.1 If $\Delta x_j = F_d$, Insert_CS-list $(x_i, Y_{current})$. Then set $Y_{pre} = Y_{current}$ and $Y_{current} = y_j$.
   3.2 If $\Delta x_j = -F_d$, then
      3.2.1 If $F_d \cdot (Y_{pre} - y_j) \geq 0$, Insert_CS-list $(x_i, Y_{current})$ and Insert_CS-list(x_i, y_i). Then, set $Y_{pre} = y_i$, $Y_{current} = y_j$, and $F_d = \Delta x_j$.
      3.2.2 Else, set $Y_{current} = y_j$ and $F_d = \Delta x_j$.
   3.3 If $\Delta x_j = 0$ and $F_d = \Delta y_j$, then $Y_{current} = y_j$.
4. Repeat step 3 until $(x_j, y_j) = s$.

Step 1은 시작점을 찾기 위한 것으로서, 시작점의 직전 점은 시작점의 오른쪽 또는 왼쪽에 위치해야 한다. 만약 영역의 모든 화소가 한 화소 폭으로 된 수직 연결로만 되어 있다면, 크로스 섹션 생성은 그 상위점과 하위점을 각각 크로스 섹션의 상단점과 하단점으로 결정하고 종료한다.

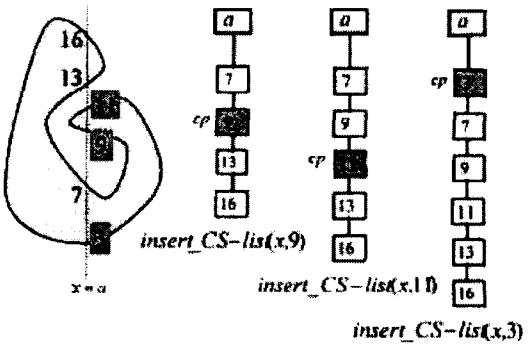

그림 5. 크로스 섹션 리스트에서의 상단점 및 하단점 추가 및 정렬

크로스 섹션 리스트(CS_list)는 $(x_{max}-x_{min}+1)$개 버켓(bucket)으로 구성된다. 버켓[x]는 크로스 섹션 $C(x,y_l:y_u)$들의 상단점과 하단점에 대한 y값들이 오름차순으로 정렬되게 유지한다. 이 과정은 그림 5에서 보듯이 별도로 기술한 프로시져 Insert _CS-list(x,y)가 현재 처리 중인 y값을 이미 처리된 버켓[x]의 y값들 사이의 해당 위치에 찾아서 삽입함으로써 처리한다. 이러한 과정을 거쳐 버켓[x]로부터 y값들을 순서적으로 쌍을 이루어 나가면 크로스 섹션이 만들어지게 된다. 그림 5에서 3개의 크로스 섹션$C(a,3:7)$, $C(a,9:11)$과 $C(a,13:16)$이 만들어지는 과정을 볼 수 있다.

영역 R내에 홀 H가 있는 경우에는 R에 대한 크로스 섹션 리스트 R_list와 H에 대한 크로스 섹션 리스트 H_list를 같은 버켓[x]끼리 합병정렬(merge sorting)하여 크로스 섹션을 결정한다. 이때, 그림 3의 점 51번 및 점 59번과 같은 H의 크로스 섹션의 상단점과 하단점은 최종 크로스 섹션에 포함되지 않아야 한다. 따라서, 크로스 섹션 리스트 $H\text{-}list$를 생성할 때, 상단점 y_u는 본래의 y값에 1이 더해진 값을 취하고 하단점 y_l은 1을 뺀 값을 취한다. 다음은 그림 3에서 $x=b$일 때의 크로스 섹션을 합병정렬을 통해 구하는 과정이다.

R_list 버켓[b]에서의 y값 : $y(31)$, $y(2)$
H_list 버켓[b]에서의 y값 : $y(59)$, $y(51)$
H_list 버켓[b]에서의 수정된 y값 : $y(64)$, $y(65)$
합병정렬 결과 : $y(31)$, $y(65)$, $y(64)$, $y(2)$
CS_list : $C(b,\ y(31):y(65))$, $C(b,\ y(64):\ y(2))$

5. 윤곽선 시퀀스의 효율적 추출 방법

앞에서는 하나의 영역이 윤곽선 시퀀스에 의해 효과적으로 표현될 수 있음을 알아보았다. 여기에서는 그림 6에서와 같이 이진 영상으로부터 각 영역에 대한 윤곽선 시퀀스를 구하는 방법을 알아본다.

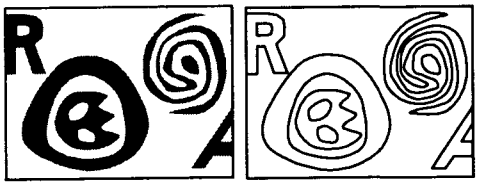

그림 6. 이진 영상에서의 윤곽선 시퀀스 추출결과

만약 하나의 영역 R이 여러 개의 홀 $H_1,\cdots,H_m$을 포함하고 있다면, 이 영역에 대한 윤곽선 시퀀스 $BS(R)$는 외부(exterior) 및 내부(interior) 윤곽선 시퀀스들로 다음과 같이 표현한다.

$BS(R)$: $EBS(R) \rightarrow IBS(H_1) \rightarrow \ldots \rightarrow IBS(H_m)$

이 방법에서는 주어진 영상을 래스터-스캔(raster-scan) 방식으로 스캐닝하면서, 새로운 물체 영역을 만날 때마다 윤곽선 추적 방법[1]으로 윤곽선 시퀀스를 생성한다. 생성된 윤곽선 시퀀스에 속하는 화소에는 고유의 레이블(label)을 붙임으로써, 차후의 스캐닝 과정에서 중복된 윤곽선 시퀀스가 생성되지 않도록 한다.

한 스캔 라인에서 스캐닝 화소 p는 다음과 같은 세 가지 상태에 놓일 수 있으며, 다른 상태로의 천이(transition)되는 과정은 그림 7과 같다.

상태 S_0 : $p \in$ 배경영역
상태 S_1 : $p \in$ 물체영역
상태 S_2 : $p \in$ 홀 영역

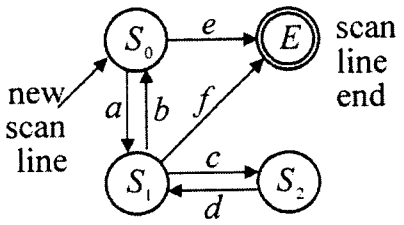

그림 7. 스캐닝 상태의 천이도

각 스캐닝 라인별로 새로운 상태 천이 과정을 거치는데, 외부 및 내부 윤곽선 시퀀스를 추출하기 위한 상태 천이 조건은 다음과 같이 정리할 수 있다.

a, d : 물체 영역 또는 레이블된 화소를 만났을 때
b : 배경 화소를 만났으면서 내부에 반복내포 홀(nested-hole) 카운터가 0일 때.
c : 배경 화소를 만났으면서 내부에 반복내포 홀 카운터가 0이 아닐 때.
e : 어떠한 물체 영역 또는 레이블된 화소도 만나지 못했을 때
f : 더 이상의 배경 화소를 만나지 못했을 때

상태 천이 조건에서 반복내포 홀 카운터는 홀 안에 다른 물체영역이 존재하는 경우가 반복되어 나타나는 것을 처리하기 위한 것이다. 이 카운터는 상태

S_1에서 새로운 홀을 생성하거나 이미 생성된 홀에 대한 레이블 화소를 만났을 때 1씩 증가되고, 상태 S_2에서 홀 레이블 화소를 만났을 때 1씩 감소한다. 한편, 각 상태에 진입한 경우에는 각각 다음과 같은 작업을 실행한다.

S_0 : 레이블되지 않은 물체 영역 화소를 만나게 되면, 새 물체 영역에 대해 윤곽선 추적을 통해 새로운 외부 윤곽선 시퀀스를 생성한다.

S_1 : 배경 화소를 만났으면서 반복내포 홀 카운터가 0이면, 새로운 홀에 대한 내부 윤곽선 시퀀스를 윤곽선 추적을 통해 생성한다. 이때, 내부 윤곽선 시퀀스는 해당 외부 윤곽선 시퀀스에 자료구조적으로 연결하여 표현한다.

S_2 : 레이블되지 않은 물체 영역 화소를 만나면, 홀 내에 존재하는 새로운 물체 영역에 대한 외부 윤곽선 시퀀스를 생성한다.

E: 현재 스캐닝 라인에 대한 처리를 끝낸다.

그림 8은 하나의 영상에 대한 윤곽선 시퀀스들을 추출하는 과정의 예를 보여주고 있다. 하나의 스캔 라인에 있는 모든 화소들이 배경 화소이면 상태 천이 조건은 e가 되어 해당 스캐닝 라인에 대한 처리를 종료한다(그림 8(a)). 만약 S_0에서 레이블되지 않은 물체 영역 화소를 만나게 되면, 그림 8(a)에서 보듯이 물체 영역(R_1)에 해당하는 윤곽선을 추적하여 새로운 외부 윤곽선 시퀀스를 생성한다. 상태 S_1에서 배경 화소를 만났을 때, 반복내포 홀 카운터가 0이면(그림 8(b)) 천이 조건이 b가 되는 반면에, 반복내포 홀 카운터가 0이 아닌 경우에는(그림 8(d)) 천이 조건은 c가 된다. 그림 8(c)에서는 상태 S_1에서 배경 화소를 만났으면서 반복내포 홀 카운터가 0이 되어 새로운 홀(H_1)을 생성하는 경우와 이미 추출된 홀을 만나게 되어 상태 S_2로 바로 천이하는 경우를 보이고 있다. 또한, 상태 S_2에서 레이블되지 않은 물체 영역을 만나게 되어 새로운 영역(R_3)에 대한 외부 윤곽선 시퀀스를 생성하는 예도 보이고 있다. 그림 8에서 추출된 윤곽선 시퀀스들은 다음과 같다.

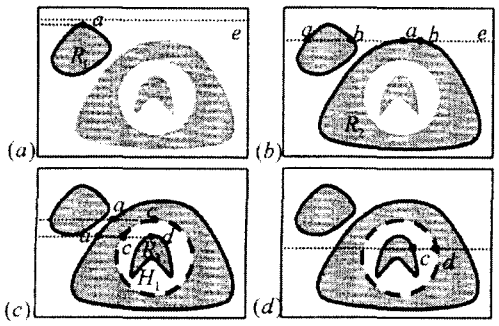

그림 8. 윤곽선 시퀀스 추출 과정 예

$BS(R_1)$: $EBS(R_1)$
$BS(R_2)$: $EBS(R_2) \rightarrow IBS(H_1)$
$BS(R_3)$: $EBS(R_3)$

6. 결 론

본 논문에서는 형상의 윤곽선 시퀀스로부터 생성된 크로스 섹션을 이용하여 물체의 형상 특징들을 효율적으로 계산할 수 있는 방법을 제안하였다. 이때, 크로스 섹션의 생성 알고리즘은 윤곽선 시퀀스를 한번 순차적으로 탐색하여 모든 크로스 섹션들을 결정할 수 있으며, 또한 형상 특징도 크로스 섹션 정보로부터 바로 계산할 수 있다. 따라서, 형상 특징들을 계산하는 시간은 영상에서의 전체 화소 수가 아니라 윤곽선 화소의 수에 따라 결정된다. 또한, 이진 영상을 한번 스캐닝하면서 모든 형상에 대한 윤곽선 시퀀스를 생성할 수 있는 방법도 제안하였다. 이 방법은 원-패스(one-pass) 연결성분 레이블링 알고리즘으로도 활용이 가능하다.

참 고 문 헌

[1] R. Jain, R. Kasturi, and B. G. Schunck, *MACHINE VISION*, McGraw-Hill, Singapore, 1995.

[2] I. Pitas and A. N. Venetsanopoulos, Morphological Shape Decomposition, IEEE trans. on PAMI, Vol. 12, No. 1, pp. 38-45, 1990.

[3] J. M. Reinhardt and W. E. Higgins, Comparison Between the Morphological Skeleton and Morphological Shape Decomposition, IEEE trans. on PAMI, Vol. 18, No. 9, 1996.

[4] J. Xu, Efficient Morphological Shape Representation without Searching, in Proc. 1998 IEEE Int. Conf. Image Processing, Chicago,

IL, Oct. 1998.
[5] A. Rosenfeld and A. C. Kak, *DIGITAL PICTURE PROCESSING*, Academic Press, New York, 1982.
[6] A. Rosenfeld and J.L. Pfalts, Sequential operations in digital picture processing, Journal of ACM, Vol. 13, No 4, pp.471-494, Oct. 1966.
[7] S. E. Umbaugh, *COMPUTER VISION AND IMAGE PROCESSING*, Prentice-Hall, London, 1998.

김 성 옥

1981년 부산대학교 수학과 학사
1998년 부산대학교 대학원 컴퓨터공학과 석사
1999년~현재 부산대학교 대학원 컴퓨터공학과 박사과정
1982년~1989년 한국기계연구원 연구원
1989년~현재 한국신발피혁연구소 자동화연구부 연구부장
관심분야 : 화상처리, 컴퓨터비전, 패턴인식

김 동 규

1992년 서울대학교 컴퓨터공학과 학사
1994년 서울대학교 대학원 컴퓨터공학과 석사
1999년 서울대학교 대학원 컴퓨터공학과 박사
1999년 9월 부산대학교 컴퓨터정보통신연구소/BK 사업단 기금교수
2001년~현재 부산대학교 정보컴퓨터공학부 조교수
관심분야 : 컴퓨터 보안 및 암호학, Bioinformatics

김 민 환

1980년 서울대학교 전기공학과 학사
1983년 서울대학교 대학원 컴퓨터공학과 석사
1988년 서울대학교 대학원 컴퓨터공학과 박사
1991년~1992년 Univ. of Washington 객원연구원
1986년~현재 부산대학교 컴퓨터공학과 교수
관심분야 : 화상처리 및 이해, 칼라공학