論文94-31B-4-13

두 Strip 직선의 분리 회전을 이용한 디지탈 곡선의 직선 근사화

(A Method of the Linear Approximation of Digital Curves By the Separate Rotation of Two Strip Lines)

柳 承 必*

(Sung Pil Lyu)

要 約

디지탈 곡선의 다각형근사화 방법중, 근사화된 직선이 주어진 곡선과의 거리가 허용오차 이내에 있으면서 빠른 수행속도를 가지는 방법으로, Roberige가 제안한 strip을 이용한 근사화 방법이 있다. 그러나 이방법은 break point수가 많이 추출되는 단점이 있다. 후에, Leng 및 Yang은 이러한 단점을 보완하기 위하여 strip을 회전시키는 Dynamic Stip Algorithm을 제안하여 속도를 줄이는 대신에 break point 수를 감소시켰다.

이 논문에서는 strip의 두 직선을 분리하여 회전시킴으로 써, 후자의 방법에 비해, break point 수는 거의 일치하면서, 보다 빠른 속도를 가지는 방법을 제안한다.

Abstract

Roberige had proposed a method of linear approximation using strips. This method is known to be fast relatively, and the distance between the given curve and calculated straight line is not greater than an allowable error. But this method generates many break points. Later, Leng and Yang proposed the dynamic strip algorithm with the rotation of strips which reduced the number of break points at the sacrifice of the speed.

The method using the seperate rotation of two strip lines proposed in this paper is faster than Leng and Yang 's but generates almost the same number of break points as Leng and Yang' s.

Ⅰ. 서 론

디지탈곡선은 여러개의 연속된 점으로 표현되는 곡선으로, 연속되는 점과 점사이는 직선벡터로 표현될 수 있다. 이와같이 표현되는 곡선은 일정한 시간간격으로 표본화된(sampled) 파형(waveform)데이터나, 컴퓨터 CRT등에 나타나는 화상의 윤곽선등이 디지탈곡선에 해당한다. 만약 어떤 디지탈곡선이 어떤 형태를 나타내기 위해서 충분하고 많은 데이타로 이루어져 있고, 우리가 이를 보다 단순한 형태 또는, 적은 데이터로 표현해야 할 필요가 있을 때 디지 탈곡선의 직선근사화는 유용한 방법중의 하나이다. 디지탈곡선의 직선근사화는 주어진 어떤 곡선을 거리가 일정한 오차 이내에 있는 근사화된 직선으로 표현하는 방법으로서, 형태묘사, 데이터압축 및 잡음제거에 유리하며, 이는 패턴인식, 화상처리 및 그래픽등에

* 正會員, 世明大學校 電子計算學科
(Dept. of Computer Science, Semyung Univ.)
接受日字 : 1993年 9月 7日

효과적으로 사용될 수 있다.

이와같이 직선 근사화방법의 성능을 결정하는 중요한 기준으로는, - 주어진 디지탈 곡선과 근사화된 직선과의 거리가 허용오차 이내에 있을 것

- 적은 break point수(또는 근사화된 직선의 수)
- 빠른 근사화 속도
- 적은 기억장소
- 곡선을 구성하는 점들 간의 거리에 무관하게 수행될 것

등이 있다. 그러나 현재까지 이들 조건을 동시에 최대로 만족하는 방법은 없고, 각각의 기준은 상호배치되는 경향이 있다.

디지탈곡선의 직선근사화는 방법의 종류는, 디지탈 곡선상의 한 점에 대한 프로세스의 수행회수에 따라, 주어진 디지탈 곡선상의 어떤 한 점에 대해 한 번이상의 프로세스를 거치는 방법[1-5]과, 한 점에 대해 한 번의 프로세스를 거치는 방법(sequential[5,6] 또는 scan along[12] 방법이라고도 한다)[6-11] 이 있다. 대체로 한점에 대해 여러번 처리하는 방법들은 구해진 직선과 주어진 곡선간의 거리가 허용오차 이내에 있고, break point수가 적은 장점이 있으나, 속도면에서는 후자에 비해 뒤떨어지는 단점이 있다. sequential 방법중에서는 주어진 곡선과 근사화된 곡선간의 거리가 허용오차 이내에 있으면서 속도가 빠른 방법으로 Strip을 이용한 방법[10,11]과 CI(Cone Intersection)를 이용한 방법[7-9] 등이 있다. 이들 방법 중에 Roberge[10]가 제안한 방법이 가장 속도가 빠르나, 추출된 근사화 직선의 수가 CI를 이용한 방법보다 2-3배 많은 단점이 있다.[12] 그후에, Leng 및 Yang[11]은 이러한 결점을 보완하여 근사화된 직선의 수가 CI를 이용한 방법보다 적으면서, 비교적 빠른 방법을 제시한 바 있다. 그런데, 이 방법은 종래의 고정된 strip을 회전시킬 수 있게 하여 종래의 방법[10]에 비해 근사화된 직선의 수는 줄어들었지만 strip을 변화시키기 위해서 삼각함수를 포함하여 많은 계산이 요구된다.[11] 그래서 Leng 및 Yang은 삼각함수대신에 Lookup Table을 만들어 속도를 개선하였으나[11], Table자체에 약간의 오차를 가지고 있고, 역시 계산량이 많고 복잡하여 Roberige[10]방법에 비해 속도가 약 1/2 수준으로 떨어진다.

이 연구에서 같은 점을 중심으로 두 직선이 회전하는 Leng 및 Yang의 방법 대신에 strip을 구성하는 평행한 두 직선이 직선의 시작점을 중심으로 회전할 수 있도록 하여, 삼각함수 계산없이 벡터를 이용한 각도계산을 할 수 있도록 하였다. 벡터를 이용하여 각도계산을 하는 경우가 감승제 만으로 가능하므로[9], 처리속도는 Roberige 방법[10]에 가까우면서, 근사화된 직선의 수는 Leng 및 Yang[11]과 거의 일치하는 방법을 제시한다.

Ⅱ. 근사화 방법

디지탈 곡선을 나타내는 점들 P_0 에서 P_n (n=1,2,3,…)까지 점들이 있고, 시작점 P_0부터 거리 e 이상 떨어진 최초의 점 $P_i(0<i\leq n)$가 있을 때, 그림 1과 같이 Roberige[10]가 제시한 방법에서 EF=1일 때의 strip을 구한다. 여기서 이 strip을 기준 strip이라하고, 점 P_0를 중심으로 e(허용거리오차)만큼 떨어진 점중에서, 점 P_0와 점 P_i를 잇는 직선 s를 기준으로 90도 방향으로

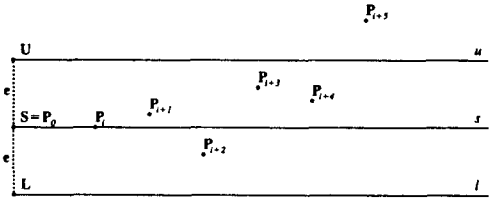


그림 1. 기준 Strip
Fig. 1. Reference Strip.

존재하는 점을 상위점(그림 1의 점 U)이라 하고, -90도 방향으로 존재하는 점을 하위점(그림 1의 점 L)이라 한다. 점 U를 포함하는 strip의 상위 경계선 u를 상한선, 점 L을 포함하는 하위경계선 l을 하한선이라 한다(그림 1참조) 이 때, 기준 strip을 대표하는 점 $U=(U_x,U_y)$ 및 $L=(L_x,L_y)$는 식 (1)-(7)로 계산된다.

$$f = e/\sqrt{DSQR} \tag{1}$$
$$e_x = D_y * f \tag{2}$$
$$e_y = D_x * f \tag{3}$$
$$u_x = S_x - ex \tag{4}$$
$$u_y = S_y + ey \tag{5}$$
$$L_x = S_x + ex \tag{6}$$
$$L_y = S_y - ey \tag{7}$$

여기서,

e : allowable distance error

(S_x,S_y) : start point S of a line segment

(D_x,D_y) : Pi- S

DSQR : square of distance between S and P_i

이상과 같이 기준 strip내에 있는 점들은 모두 하나의 근사화된 선분으로 취급되며, 이 점들은 모두 직선 s로 부터 거리 e 이내에 있게 된다. 이와 같은 방법으로 기준 strip이내에 있는 점들을 찾아 내어 근사화 하는 방법이 Roberige방법[10] 이며, 이 방법의 경우 strip이 고정되어 있으므로 그림 1에서 점 P_{i+5}과 같이 Strip으로부터 조금이라도 벗어나는 경우, 앞의 점들($P_0, P_1, \cdots, P_{i+4}$)과 같은 선분으로 분류되지 못하게되고, 이는 break point수가 증가하는 요인이 된다. 그 후에 Leng 및 Yang은 이 방법을 개선하여 그림 1에서 점 S를 기준으로 직선 u와 l을 회전시켜 가능한 많은 점들을 포함하는 strip을 찾아냄으로써 break point를 줄이는 방법을 제안하였다.[11] 그러나 이 방법은 break point수는 줄일 수 있었으나, 직선 u 및 l을 회전 시키기위한 각도 계산을 위하여 삼각함수 사용이 불가피하고 이것이 속도를 느리게하는 주요 원인이 되었다.

이 연구에서는 한점 S를 중심으로 u와 l을 회전시키는 Leng 및 Yang의 방법과는 달리 두점 U와 L을 고정하고 이를 중심으로 각각 직선 u와 l을 회전시킴으로서 각도 비교를 위한 삼각함수 계산없이 가감승제 계산만으로 처리할 수 있는 방법을 다음과 같이 제안한다.

만약, 그림 1의 기준 strip을 그림 2 (a)와 같이 점 U,L을 고정시키고, 평행한 두 직선인 상한선 및 하한선을 시계방향으로 각도 β 만큼 회전하여 P_{i+5}를 포함하는 새로운 strip을 만들면, 이 strip은 기준 strip보다 폭이 작으므로 이 strip내에 있는 모든 점도 역시 직선 s'로부터 거리 e이내에 있게된다. 즉, U 및 L을 기준으로 회전한 Strip내에 있는 모든 점은 허용오차 e 이내에 있는 같은 근사화된 직선으로 간주될 수 있다.

이 때, 회전된 새로운 strip의 폭 w는

$$w = 2 * e * \cos\beta \qquad (8)$$

where,

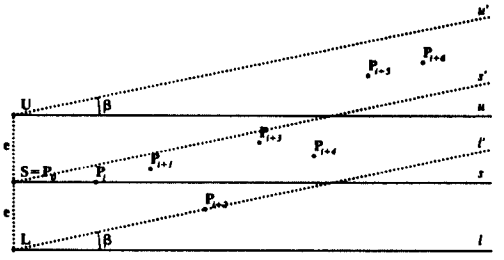
β : 기준 strip으로부터 회전각도

이며, S로 부터 거리가 e 이상 떨어진 P_i를 포함하기 위해서는 회전할 수 있는 최대각도는 45도 미만이므로 w는 식 (8)로 부터

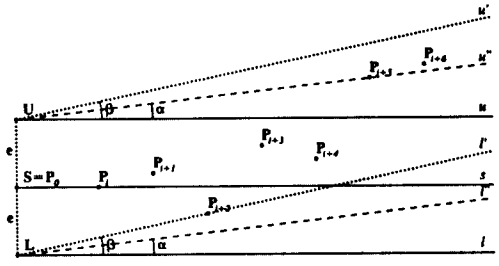
$$\sqrt{2} * e < w \leq 2 * e \qquad (9)$$

가 된다. 본 방법에서는 이상과 같이 상한선과 하한선을 움직여 만든 가변 strip내에 있는 모든 점들이 허용오차 이내에 있게 되므로 이들을 하나의 직선으로 간주하게 된다.

다음의 각절은 이상의 근사화 방법을 구현하기 위하여 strip의 회전범위, strip의 회전 각도비교방법 및 이를 이용한 근사화 알고리듬을 제시한다.



(a) Approximation by the rotation of Strip



(b) Maximum and Minimum Rotation

그림 2. Strip의 회전

Fig. 2. Rotation of Strip.

1. Strip의 회전범위

이상의 방법에서 stip을 움직일 수 있는 범위는 같은 근사화 직선으로 분류된 모든 점을 포함하는 최대 각도로 표현된다. 예를 들면, 그림 1에서의 점 P_0부터 점 P_{i+5}를 포함하도록, 그림 2(a)와 같이 최대한 회전할 수 있는 각도는 점 L에서 점 P_{i+3}를 잇는 하한선과 기준 strip의 하한선과 이루는 각도 β가 된다. 한편 점 P_{i+5} 까지 모든 점을 포함하도록 strip이 움직일수 있는 최소각도는 그림 2(b)와 같이, 점 U에서 점 P_{i+5}를 잇는 상한선과 기준 strip의 상한선과 이루는 각도 α가 된다. 따라서 P_{i+5} 다음에 처리되는 새로운 점 P_{i+6}이 strip을 회전할수 있는 범위내에 있으면, P_{i+5}와 같은 선분의 일부로 처리된다. 즉, 점 P_{i+6}이 점 P_{i+5}와 같은 선분의 일부가 되기 위해서, 그림 2(b)의 직선 u'와 l"사이에 존재하면 된다. 이와 같이, 새로운 점 P_{i+6}이 위의 범위내에 들기위한 조건은 점 U에서부터 점 P_{i+6}에 이르는 선분을 u_{i+6} 이라

하고, 이 선분과 기준 상한선 u와 이루는 각도를 α_{i+6}라 할 때, 직선 u_{i+6}는 직선 u'보다 아래에 있어야 하므로, α_{i+6}와 β 의 관계는,

$$\beta \geq \alpha_{i+6} \tag{10}$$

이고, 점 L에서부터 점 P_{i+6}에 이르는 선분을 l_{i+6} 라 하고, 이 선분과 기준 하한선 l과 이루는 각도 β_{i+6}라 할 때, 직선 l_{i+6}는 직선 l"보다 위에 있어야 하므로, β_{i+6}와 α 의 관계는

$$\alpha \leq \beta_{i+6} \tag{11}$$

가 된다. 만약 P_{i+6}이 위의 조건을 만족하지 못할 경우 P_{i+6}은 새로운 선분의 일부로 간주되며, 이때, 점 P_{i+5}는 P_0로부터 P_{i+5}에 이르는 선분의 마지막점이며, 새로운 선분의 시작점 즉, break point가 된다. 한편 점 P_{i+6}이 식 (10)과 식 (11)을 만족하면, 다음점 P_{i+7}이 존재하는 범위를 결정하기 위해서 점 P_{i+6}을 포함하여 strip의 최대 및 최소 회전각도를 갱신해야 하며, 최대 회전각도 β 가 갱신될 조건은

$$\beta > \beta_{i+6} \tag{12}$$

이며, 최소 회전각도 α 가 갱신될 조건은

$$\alpha < \alpha_{i+6} \tag{13}$$

이다. 한편, 최소 회전 각도는 최대 회전 각도보다 크지 않아야 하므로, α와의 관계는,

$$\alpha \leq \beta \tag{14}$$

와 같다.

2. 각도 비교 방법

이 방법은 위의 식 (10)-(13)과 같이 대부분 각도의 비교가 주로 요구되는데, 삼각함수를 이용하여 각도를 직접 계산하고 이를 비교하는 경우, 계산속도가 느려질 수 있다. 그러나, 단순히 두 직선이 이루는 각도의 크기를 비교할 경우, 삼각함수의 계산 없이 좌표의 곱셈에 의해 간단히 해결하는 방법을 참고문헌 [9]에 제시되어 있으므로, 여기서는 참고문헌 [9]에 제시되어 있는 벡터 외적에 의한 방법을 이용하여 각도비교를 한다.

정의 1) (x, y) 평면상에 원점으로 부터 두 점 A=(A_x, A_y) 및 B=(B_x, B_y)에 이르는 두개의 벡터 **A**, **B**가 있다고 할 때, 두 벡터의 외적(cross product),

$$\mathbf{A} \times \mathbf{B} = |\mathbf{A}| * |\mathbf{B}| * \sin\theta * \mathbf{U} = \left(A_x * B_y - A_y * B_x\right) * \mathbf{U}$$

U: **A**, **B**에 수직인 단위벡터

에서, AXB를 **A**×**B**의 scalar값으로 정의한다. 즉,

$$AXB = A_x * B_y - A_y * B_x$$

라 한다. 여기서, 문자 A와 B는 각각 벡터 **A** 및 **B**를 나타내고, 대문자 X는 cross product 연산자를 의미한다. ■

벡터의 외적의 성질상 벡터 **B**가 벡터 **A**에 대해 시계방향으로 존재하면 AXB는 0 보다 크고, 반시계방향으로 존재할 때는 0보다 작게된다.[9] 이 절에서는 이러한 성질 및 위의 정의를 이용하여 두 직선(또는 벡터)의 각도를 비교한다.

일반적으로, 디지탈 곡선을 구성하고 있는 점 P_0, $P_1, \cdots, P_n$이 주어져 있을 때, 시작점 $S=P_0$로 부터 최초로 e이상 떨어진 점 $P_i(0<i\leq n)$가 있고, S와 P_i로 부터 생성되는 기준 strip에서 상위점 U와 점 P_i, P_{i+1}, $\cdots$, $P_m(m\leq n)$들과 연결하는 직선들을 각각 u_i, u_{i+1}, $\cdots$, u_m 이라하고, 기준 strip의 하위점 L과 점 P_i, P_{i+1}, $\cdots$, P_m들과 연결하는 직선들을 각각 l_i, l_{i+1}, $\cdots$, l_m 이라 할 때, 점 U로 부터 시계방향으로 각도가 가장 작은 직선 $u_{min}(=u_j,\ i\leq j\leq m)$와 이 직선을 구성하는 곡선상의 최초의 점을 $U_{min}(=P_j,\ i\leq j\leq m)$가 있고, 점 L과 주어진 곡선상의 점과 연결하는 직선들중 시계방향으로 각도가 가장 큰 직선 $l_{max}(=l_k, i\leq k\leq m)$와 이 직선을 구성하는 곡선상의 최초의 점 $L_{max}(L_{max}=l_k,\ i\leq k\leq m)$가 있을 때 (예를 들면, 그림 2(b)에서 $U_{min}=P_{i+5}$ 이고, $L_{max}=P_{i+3}$ 다), 점 U로 부터 U_{min}에 이르는 벡터를 $\mathbf{U}_{min}$, 점 L로 부터 점 L_{max}에 이르는 벡터를 $\mathbf{L}_{max}$, 점 U 및 L로 부터 임의의 점 $P_x(i\leq x\leq m)$에 이르는 벡터를 각각 $\mathbf{U}_x$ 및 $\mathbf{L}_x$라 하면, 식 (10)-(14)의 식은 각각,

$$L_{max} X U_x \leq 0 \tag{15}$$
$$U_{min} X L_x \geq 0 \tag{16}$$
$$L_{max} X L_x \geq 0 \tag{17}$$
$$U_{min} X U_x \leq 0 \tag{18}$$
$$U_{min} X L_{max} \geq 0 \tag{19}$$

와 같이 표현된다.

3. 계산의 효율화

이상과 같이 점 하나에 대해, 식 (15) 또는 (16)에

서 break point가 발생하면 2 번의 곱셈으로 가능하나, 최악의 경우 식 (15)-(18)을 수행하면 8번의 곱셈이 필요하므로, 이의 계산을 줄이기 위해 식 (15)-(18)를 다음과 같이 변경할 수 있다.

만약, 앞절의 식 (15)-(19)에서 $U_{min}=U_i$, $L_{max}=L_j$ $(0<i,j<x<n)$라하고, 임의의 점 P_{k-1}에서 점 P_k에 이르는 벡터를 $dP_k(i,j<k<x)$이라고 정의하면,

$$\mathbf{U}_x = \mathbf{U}_i + d\mathbf{P}_{i+1} + d\mathbf{P}_{i+1} + \cdots + d\mathbf{P}_x \quad (20)$$

$$\mathbf{L}_x = \mathbf{L}_j + d\mathbf{P}_{j+1} + d\mathbf{P}_{j+1} + \cdots + d\mathbf{P}_x \quad (21)$$

가 되므로, 식 (15)-(18)은 식 (20)-(21)로 부터

$$L_{max}XU_x = L_{max}X(U_i + dP_{i+1} + dP_{i+2} + \cdots + dP_x) \le 0 \quad (22)$$

$$U_{min}XL_x = U_{min}X(L_j + dP_{j+1} + dP_{j+2} + \cdots + dP_x) \ge 0 \quad (23)$$

$$L_{max}XL_x = L_{max}X(L_j + dP_{j+1} + dP_{j+2} + \cdots + dP_x) \ge 0 \quad (24)$$

$$U_{min}XU_x = U_{min}X(U_i + dP_{i+1} + dP_{i+2} + \cdots + dP_x) \le 0 \quad (25)$$

가 된다. 다시 위의 식 (22)-(25)는 $\mathbf{U}_k$ 및 $dP_k(i,j<k<x)$에 대한 정의로 부터

$$L_{max}XU_x = L_{max}XU_{x-1} + L_{max}XdP_x \le 0 \quad (26)$$

$$U_{min}XL_x = U_{min}XL_{x-1} + U_{min}XdP_x \ge 0 \quad (27)$$

$$L_{max}XL_x = L_{max}XL_{x-1} + L_{max}XdP_x \ge 0 \quad (28)$$

$$U_{min}XU_x = U_{min}XU_{x-1} + U_{max}XdP_x \le 0 \quad (29)$$

가 된다. 즉, 식 (26)-(29)식은 $x>i$ 및 $x>j$ 에 대해서는 $U_{min}XdP_x$ 와 $L_{max}XdP_x$의 누적으로 이루어지므로 4번의 곱셈과 6 번의 가감산으로 결정된다.

그런데 식 (28) 또는 (29)에서 $U_{min}=U_x$ $(i<x<n)$ 또는 $L_{max}=L_x$ $(j<x<n)$와 같이 갱신이 일어나는 경우, 새로운 L_{max} 또는 U_{min}에 대해 $L_{max}XU_x$ 또는 $U_{min}XL_x$의 계산이 필요하고, 이에따라 2 번의 곱셈이 더 필요하게 된다.(L_{max}, U_{min}이 동시 발생하는 경우도 $L_{max}XU_x=-U_{min}XL_x$가 되므로 역시 2번의 곱셈 추가만으로 가능하다)

이상으로부터 식 (26)-(29)에서 4번 또는 6번의 곱셈이 필요한데, 만약 L_{max} 또는 U_{min} 의 갱신을 가능한 피할 수 있으면 곱셈 계산을 줄일 수 있으므로 처리 속도면에서 더욱 유리하다. 다음의 경우는 약간의 비교에 의해서 L_{max} 또는 U_{min} 의 갱신을 피할 수 있는 방법을 제시한다. 만약, 점 P_x, P_{x+1}, P_{x+2}, $\cdots$ P_{x+s} $(x+s\le n)$에 대해, 식 (26), (27) 및 (28)이 연속적으로 만족된다고 가정하고, 이를 U_{min} 의 갱신 없이 알 수 있다면 굳이 모든점에 대해 U_{min} 를 갱신할 필요는 없고, 오직 마지막 점 P_{x+s} 에 대해서만 U_{min}를 갱신하면 된다. 만약, P_x 에 대해서 식 (26), (27) 및 (28)이 성립하고,점 P_x, P_{x+1}, $\cdots$, P_{x+s} 에 대해, U_{min}의 갱신없이 k=1,...,s 인 모든 점 P_{x+k}에 대해

$$L_{max}XU_{x+k} \le 0 \quad (30)$$

$$L_{max}XdP_{x+k} \le 0 \quad (31)$$

를 만족한다고 하면, 식 (19) 및 (31)로 부터

$$U_{min}XdP_{x+k} \ge 0 \quad (32)$$

이다. 따라서, 식 (27)과 (32)로 부터,

$$U_{min}XL_x + U_{min}dP_{x+1} + U_{min}XdP_{x+2} + \cdots + U_{min}XdP_{x+k} \ge 0 \quad (33)$$

이고, 이는 다시, 정의에 의해서,

$$U_{min}X(L_x + dP_{x+1} + dP_{x+2} + \cdots + dP_{x+k}) = U_{min}XL_{x+k} \ge 0 \quad (34)$$

한편 식 (29)와 (32)로 부터 같은 방법으로,

$$U_{min}X(U_x + dP_{x+1} + dP_{x+2} + \cdots + dP_{x+k}) = U_{min}XU_{x+k} \ge 0 \quad (35)$$

가 된다. 즉, 식 (30), (34) 및 (35)는 식 (26), (27) 및 (28)과 대응되는 식으로 식 (30) 및 (31)로 부터 구해질 수 있다. 다시 말하면 식 (30) 및 (31) 식은 점 P_{x+k}가 strip내에 있는 점이며, 이점에서 U_{min}이 갱신되는 충분조건이 된다. 또한 이 조건은 U_{min}의 갱신없이 조사할 수 있으므로 앞서 언급한 바와 같이 이 경우, 점 P_x, P_{x+1}, $\cdots$, P_{x+s-1} 에 대해서 U_{min}의 갱신이 필요 없게되고, 최종적으로 $U_{min}=U_{x+s}$로 갱신하면된다. 같은 방법으로 P_x 에 대해서 식 (26), (27) 및 (29)가 성립하고, P_{x+1}, P_{x+1}, $\cdots$, P_{x+s} 에 및 k=1,2,$\cdots$,s인 모든 k에 대해서 아래 조건이 성립하면, 점 P_x, P_{x+1}, $\cdots$, P_{x+s-1} 에 대해서 L_{max} 갱신의 생략이 가능하다.

$$U_{min}XL_{x+k} \ge 0 \quad (36)$$

$$U_{min}XdP_{x+k} \ge 0 \quad (37)$$

위의 식 (30) 및 (31) 식은 break point를 검사하는 조건이므로 만약 주어진 점 P_{x+k}가 break point가 아니고, dP_{x+k}의 각도가 임의의 값을 가질 때, 식 (31) 또는 (37)의 식이 만족할 확률은 약 50%이다. 즉, break point가 발생하지 않으면, 연속적으로

L_{max} 또는 U_{min} 의 갱신조건을 만족하고 이를 생략할 수 있는 확률이 약 절반에 해당된다.

4. 알고리듬

이상으로부터 본 방법의 알고리듬은 다음과 같이 나타낼 수 있다.

(1) S로 부터 P_i까지 거리가 허용거리오차 e보다 클 때까지 i 를 증가시킨 다음, 기준 strip의 점 U, L 및 최대회전 각도를 가지는 벡터 U_{min}, L_{max}를 구한다. (초기에 i=1, S=P_0)

(2) i<n 이면 i=i+1로두고 (3)을 수행하고, 아니면, 종료한다.

(3) 점 P_i가 P_{i-1}과 같은 선분이 되기위한 허용범위내에 드는지 조사하고, 허용범위내에 들면 step(4)를 수행하고, 허용범위내에 들지 않으면, 점 P_{i-1}을 break point로 간주하고, S=P_{i-1}로 둔다음, step (1)을 수행한다.

(4) 벡터 L_{max} 및 U_{min} 갱신보류 표시가 있으면 step (5)를 수행하고, 없으면 최대 및 최소회전 각도를 나타내는 벡터 L_{max} 및 U_{min} 갱신조건을 만족하는지 조사하여, 조건을 만족하면 해당 벡터의 갱신보류표시를 한다음 step (2)를 수행한다.

(5) 벡터 L_{max} 및 U_{min} 갱신보류조건을 조사하고 만족하면 step(2)를 수행하고, 아니면 점 P_{i-1}에서 해당 벡터를 갱신한다.

여기서, S는 segment의 시작점을, P_i (i=0,1,2, …,n)는 주어진 곡선상의 점을 나타낸다. Step (1)의 기준 strip은 II장 1절에 있는 방법과 같이 구하며, 최초의 U_{min}는 2절에 설명한 바와 같이 U를 기점으로하여 점 P_i에 이르는 벡터이고, L_{max}는 L을 기점으로하여 점 P_i에 이르는 벡터가 된다. Step (3)에서 P_i가 허용범위내에 있는지 조사하기 위해 3절의 식 (26) 및 (27)을 적용하고, Step (4)의 갱신조건은 L_{max} 및 U_{min} 갱신 조건은 각각 식 (28) 및 (29)와 같다. 그리고, step (5)의 L_{max} 또는 U_{min} 갱신보류조건은 각각 식 (30) 및 (31) 또는 (36) 및 (37)을 적용한다.

III. 실험

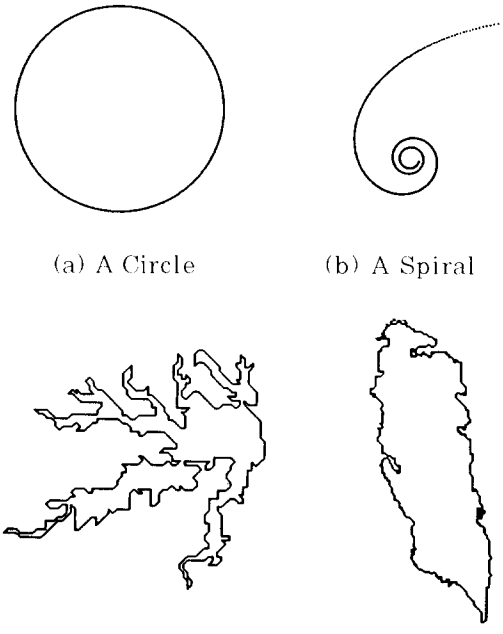
이 절에서는 본방법의 성능(속도 및 break point 수)을 다른 알고리듬과 비교하기 위해서 다른 문헌들 [10,11,12] 등에서 이미 사용한 곡선들에 대하여 실험하였다. 여기서 비교한 방법들은 그 알고리듬이 상세히 나와 있는 Roberige 방법 [10] 과, strip을 이용한 방법과의 성능비교에 자주 비교가 되는 CI를 이용한 Williams 방법 [7] 이다. 한편 Leng 및 Yang의 방법은 구체적인 알고리듬을 제시하지 않고 있으므로 문헌 [11] 에서 제시한 실험결과를 위의 방법과 상대비교하여 성능을 비교검토 하였다.

이 논문에 사용된 곡선은 Roberige [10] 와 Leng 및 Yang [11] 의 strip 알고리듬 실험에 사용된 각각 1001개의 점으로된 원 $x^2+y^2=1$ 인 곡선 (그림 3(a)) 및 나선형 $r\theta=1$ 인 곡선(그림 3(b))과, Dunham [12] 이 사용한 그림으로, 402개 및 512개의 점으로 구성되어 있는 곡선(그림 3(c),(d))이다.

실험시 허용오차에 있어서, 그림 3 (a),(b)는 점간의 거리가 1 또는 $\sqrt{2}$인 8방향 체인코드이고, 그림 3 (c),(d)는 점간의 거리가 약 0.001-0.01 사이인 곡선의 일부이다. 따라서, 허용오차는 주어진 곡선의 점간의 거리에 따라 이미 여러 논문이 실험한 바 있는 실험 허용오차에 대해 비교하였다. [10,11,12]

여기서 사용한 시스템은 IBM-PC486이고, 사용언어는 PASCAL이다.

본 방법은 표 1. 에서 보는 바와같이, 처리속도를 결정하는 중요 연산인, 제곱근의 계산회수가 Roberige [10] 방법보다 적으며, 곱셈의 수는 break point 발생



(a) A Circle (b) A Spiral

(c) Lake Lewisville (d) Lung

그림 3. 알고리듬 실험을 위한 곡선들
Fig. 3. Curves for the Experiment of Algorithms.

표 1. 그림 3에 대한 실험 결과

Table 1. Results of experiments for Fig. 3.

	허용 오차	Roberige 방법				Williams 방법				본 방법			
		time	br#	sqrt	m/d	time	br#	sqrt	m/d	time	br#	sqrt	m/d
Fig. 3(a)	0.00001	522	1001	999	7995	708	1001	1999	7000	610	1001	1000	8999
	0.0001	346	501	500	5999	676	251	1999	5003	285	201	200	5399
	0.001	215	144	143	4571	675	73	1999	4649	237	60	59	5002
	0.01	176	45	44	4087	659	23	1951	4490	219	20	19	4854
	0.1	148	14	13	3661	610	7	1787	4201	209	7	6	4665
Fig. 3(b)	0.00001	499	1001	999	7995	664	910	1999	6730	505	764	763	7814
	0.0001	319	467	465	5859	665	262	1999	5087	292	214	213	5560
	0.001	214	146	145	4569	665	76	1991	4662	236	64	63	5050
	0.01	159	44	43	3835	615	24	1803	4270	219	21	20	4749
	0.1	143	12	11	3513	390	6	1065	3233	198	6	5	4340
Fig. 3(c)	0.5	121	329	238	2874	319	245	1055	2996	109	168	82	3253
	1.0	82	137	121	2299	274	106	907	2303	104	91	70	2989
	2.0	71	81	73	2132	258	57	839	2075	88	44	35	2762
	3.0	77	51	47	2180	241	38	770	1933	83	24	23	2600
	4.0	61	36	33	1970	247	24	787	1880	83	18	17	2594
	5.0	77	22	21	2073	220	17	713	1746	71	12	8	2507
Fig. 3(d)	0.5	82	254	158	1906	247	191	803	2310	77	116	44	2443
	1.0	61	92	74	1478	209	71	687	1701	71	52	40	2195
	2.0	50	49	40	1385	219	32	699	1589	71	26	23	2214
	3.0	50	29	26	1485	198	21	645	1521	55	14	10	2013
	4.0	50	25	23	1552	209	15	677	1489	60	11	10	2005
	5.0	49	19	16	1512	204	10	665	1409	60	9	8	1905

time : 수행시간(msec)

br# : break point수

sqrt : 제곱근계산회수

m/d : 곱셈 또는 나눗셈 계산회수

후에 기준 strip 계산시 4회, 그외에는 U_{min} 및 L_{max}의 갱신이 발생하는 경우에만 2회정도 Roberige [10] 방법보다 많게된다. 그런데 만약 U_{min} 및 L_{max}의 갱신이 발생할 확률이 50%로 가정하면, 평균 1회 정도의 곱셈계산만이 Roberige방법보다 더 필요하게 된다. 한편, 본방법에 의한 strip범위는 Roberige의 strip의 범위를 포함하므로, 표 1에서 보는 바와 같이 break point 수는 항상 적거나 같게되고, 이에따라 기준 strip의 계산이 Roberige방법보다 줄게되고, 아울러 제곱근계산의 회수가 Roberige 방법보다 적어지므로, 속도 면에서, 제곱근계산이 곱셈계산과 속도가 같다고 가정해도 본방법은 Roberige 방법의 80%정도 이상의 속도를 가진다. 한편, Leng 및 Yang의 방법 [11] 은 Roberige방법의 50%정도의 속도를 가지므로 본방법은 Roberige 방법을 기준으로 Leng 및 Yang의 방법에 비해 상대적으로 약 30% 정도 빠르다고 할 수 있다. 그리고 앞서 언급한바와 같이 break point수는 Roberige 방법에 의한 것보다 50%이내로 줄어들며, Leng 및 Yang [11] 의 방법과 비교하면, 일부 허용오차에 따라 break point수에 약간의 차이가 있으나, 이는 전체 break point수의 1% 미만에 해당한다.(표 1. 및 문헌 [11] 참조)

Ⅳ. 결론

Leng 및 Yang [11] 은 Roberige [10] 이 제시한 strip 알고리듬이 break point수가 많은 단점을 개선하여, 직선의 시작점을 중심으로 strip을 회전하는 방법을 제시하여 break point수를 1/3 수준으로 줄였으나, 속도면에서 Roberige [10] 의 방법에 비해 절반 수준으로 떨어지고, strip의 회전 각도비교를 위해서 삼각함수를 lookup table을 이용해야 하므로 (Table을 이용하지 않는 경우 속도가 더욱 느려짐 [11]), 알고리듬이 복잡하여 Roberige [10] 방법보다 적용하기 어려운 문제점이 있다. 본 방법에서는 2절에서와 같이 기준 strip의 상, 하위점을 기준으로 strip을 회전하고, 또 이의 회전각도비교를 벡터를 이용함으로써, 삼각함수를 이용하지 않고 제곱근 및

곱셈(또는 나눗셈)등을 이용하여 계산으로 이루어진다. 본방법에서 계산속도를 결정하는 제곱근 및 곱셈의 계산회수는 break point발생시에 제곱근 1회, 곱셈 평균 10회이고(Roberige [10] 방법의 경우, 각각 1회, 8회), 그외의 경우에는 한 점당 평균 5회(Roberige [10] 방법의 경우 4회) 정도의 곱셈으로 이루어진다. 그러나, break point수에 있어서는 Roberige방법의 50%미만의 수준으로 Leng 및 Yang [11] 의 방법과 유사하므로, 속도 및 break point수를 종합할 때, 이 논문에서 제안한 방법은 Strip을 이용한 종래 방법들의 단점을 보완한 방법이라할 수 있다.

參 考 文 獻

[1] U. Ramer, "An iterative procedure for the polygonal approximation of plane curves," Computer Graphics and Image Processing 1, pp.244-256, 1972.

[2] T. Pavlidis and S. Horowitz, "Segmentation of plane curves," *IEEE Trans.* on Computer 23, pp.860-870, 1974.

[3] T. Pavlidis, "Polygonal approximation by newton's method," *IEEE Trans* on Computer 26, pp.800-807, 1977.

[4] U. Montanari, "A note on minimal length polygonal approximation to a digitized contour," ACM Trans. on Communication 13, pp.41-47, 1970.

[5] Y. Kurozumi and W.A. Davis, "Polyginal approximation by the minimax method," Computer Graphics and Image Processing 19, pp.248-264, 1982.

[6] K. Wall and P. E. Danielsson, "A fast method for polygonal approxi mation of digitized curves," Computer Vision, Graphics, and Image Processing 28, pp. 220-227, 1984.

[7] C. M. Williams, "An efficient algorithm for the piecewise linear approximation of planar curves," Computer Graphics and Image Processing 8, pp.280-293, 1978.

[8] J. Sklansky and V. Gonzales, "Fast polygonal approximation of digit ized curves," Pattern Recognition 12, pp. 327-331, 1980.

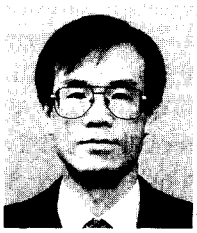
[9] 류승필, 권오석, 김태균, "벡터를 이용한 2 차 평면 곡선의 다각형 근사화," 전자공학회논문지, 제 27권 제12호, pp.43-49, 1990.

[10] J.Roberge, "A data reduction algorithm for plannar curves," Computer Vision, Graphics, and Image Processing 29, pp. 168-195, 1985.

[11] M. Leng and Y. Yang, "Dynamic strip algorithm in curve fitting," Computer Vision, Graphics, and Image Processing 51, pp146-165, 1990.

[12] J. G. Dunham, "Optimum uniform piecewise linear approximation of planar curves," *IEEE Trans.* on PAMI 8, pp.67-75, 1986.

著 者 紹 介



柳 承 必(正會員)

1955年 6月 16日生. 1979年 2月 서울대학교 전자공학과 졸업. 1980年 7月 ~ 1993年 3月 한국원자력연구소. 1987年 3月 충남대학교 대학원 전자공학석사. 1991年 8月 충남대학교 대학원 전자공학과박사. 1993年 3月 ~ 현재 세명대학교 전자계산학과 전임강사. 주관심 분야는 인공지능, 패턴인식, 컴퓨터 그래픽스 등임.