論文 90-27-3-7

디지탈 곡선의 다각형 근사화

(A Method of Polygonal Approximation of Digital Curves)

柳 承 必*, 權 五 錫**, 金 太 均**

(Sung Pil Lyu, O Sok Kwon, and Tae Kyun Kim)

要 約

곡선의 다각형 근사화 방법은 화상분석 또는 데이타 압축등에 많이 사용된다. 다각형 근사화 방법으로, 적은 break point수를 갖고, Sequential Process로 결과를 얻을 수 있는 Cone intersection 방법이 있다. 여기서는 화소간의 거리가 일정한 디지탈 곡선의 경우, 종래의 cone intersection 방법을 정수계산을 이용하여 속도를 향상시키는 방법을 제안한다.

Abstract

Polygonal approximation of digital curves is useful for the image analysis or data compression. There are methods of polygonal approximation using cone intersection which have relatively smaller number of break points and are executed in sequential process.

Here, a method of polygonal approximation is proposed, which is modified from Sklansky and Gonzales' method, and improves the speed by using integer operations.

I. 서 론

곡선의 다각형 근사화 방법은 화상분석 및 데이타 압축에 많이 사용되는 방법이다. 다각형 근사화 방법은 거리, 각도, 면적 및 구조적 특성등을 이용하여 많은 연구가 있어 왔다.[1-7] 이들의 방법으로는 한 점에 대해서 두번 이상 수행하는 방법과, 한점에 대해서 한번씩 수행하는 방법[4-7] 등이 있는데 대체로 후자가 전자에 비해 속도가 빠르다.

한편, 한점에 대해 한번씩 수행하는 방법으로는 구조적 특성, 면적 및 Cone intersection 등을 이용하는 방법이 있는데, 구조적 특성을 이용하는 방법[4]은 허용오차를 1인 경우로 제한하고 있으므로 그 활용의 폭이 좁고, 면적을 이용하는 방법[5]은 곡선의 형태에 따라 허용오차가 일정하지 않아, 비교적 많은 segment(또는 break point)가 생기게 된다. 그런데, cone intersection을 이용하는 방법은 허용오차가 거의 일정한 값을 가지게 되어, 다른 방법에 비해 적은 segment를 발생하므로, 현재까지 방법중에는 가장 효율적인 방법으로 알려져 있으나, 구조해석에 의한 방법이나, 면적을 이용하는 방법에 비해 속도가 늦은 단점이 있다.

Cone intersection 방법으로는 Williams[6]가 제안한 방법과 Sklansky[7]가 제안한 방법등이 있는데, 삼각함수 Tangent의 반각 특성을 이용하여, 삼각함수의 계산없이 segment를 추출하는 Williams의 방법이 속도면에서 우수하고, 접선의 각도를 이용하는 Sklansky의 방법은 전자에 비해 break point수가 적은 것

*正會員, 韓國原子力研究所 計測制御研究室
(Korea Atomic Energy Research Institute, IC Dept.)

**正會員, 忠南大學校 電子計算機工學科
(Dept. of Comput. Eng., Choognam Nat'l Univ.)

接受日字 : 1990年 1月 18日

으로 알려져 있다.[11]

여기서는 화소간의 거리가 균등한 디지탈 곡선(선을 구성하는 이웃하는 점들 간의 거리가 1 또는 $\sqrt{2}$ 인 디지탈 곡선)에 대해서, Sklansky의 방법을 수정하여, 각도의 비교대신에 길이의 비교로 곡선으로부터 segment들을 추출하는 방법을 제안한다.

한편, 여기에서는 길이 및 접점의 위치에 관한 변수들을 모두 정수화시키고, 이들 정수화된 변수의 가감산 및 비교만으로 segment들을 추출하여, 종래의 cone intersection 방법에 비해 속도를 개선시켰다.

Ⅱ. 다각형 근사화 방법

1. 접선

Sklansky가 제안한 cone intersection 방법은 그림 1과 같이 두 접선 사이에 다음 점이 존재하면, 그점은 같은 segment가 되고, 아니면 그 점은 다른 segment의 일부가 된다. 이들 접선은, 시작점이후 현재점(p_n, $n=1,2,\cdots$)까지 각 점을 중심으로 반지름 e (허용오차)의 원을 구성하고, segment시작점(break point)으로 부터 원에 접하는 두개의 접선을 구한 다음, 양의 x축으로 부터 각도를 계산하여, 값이 큰 접선을 TCONEi ($1 \leqq i \geqq n$), 값이 작은 것을 BCONEi라 할 때, TCONEi중에 각도가 가장 작은 것이 상위접선, BCONEi중에 가장 큰 것이 하위접선이다. 즉, 시작점과 p_{n+1}을 잇는 직선이 이루는 각도가 상하위접선 사이에 있으면, 시작점에서 p_{n+1}까지 하나의 segment가 되고, 아니면, 시작점에서 p_n 까지 하나의 segment로 되고, p_n은 새로운 시작점(break point)이 된다. 이와같이 Sklansky 방법은 매점마다, 3개의 각도를 계산해야 하고, 이 계산은 이 방법의 수행속도를 결정하게 된다.

그런데, 만약 한점에서 접선까지의 거리를 쉽게 알 수 있다면, 거리의 크기를 고려하여, 현재점과 현재점으로 부터 만들어질 접선들이 상하위접선 이내에 있는지 여부를 경우를 미리 판정하여, 접선의 계산수를 줄일 수 있다.

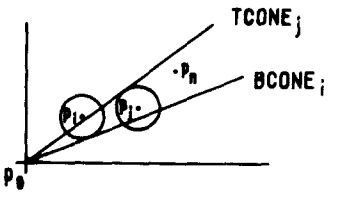


그림 1. Cone intersection
Fig. 1. Cone intersection.

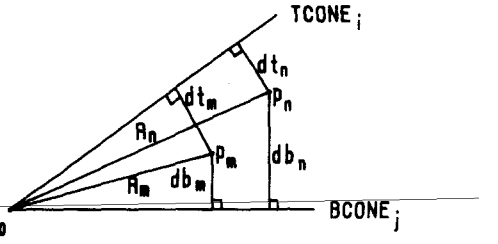


그림 2. 점과 접선과의 거리
Fig. 2. Distance between points and tangent lines.

그림 2에서,

p_0 : segment 시작점
p_m, p_n : 디지탈곡선상의 점들
R_m, R_n : 원점에서 p_m, p_n까지의 길이
dt_m, dt_n : p_m, p_n으로부터 상위 접선까지의 거리
db_m, db_n : p_m, p_n으로부터 하위 접선까지의 거리
TCONEi, BCONEj : 점 p_m까지 상위접선 및 하위접선
$i, j, m < n$ 이고, $i, j \leqq m$

라고 할 때, 만약 $R_n \geqq R_m$이고, $dt_m \geqq dt_n \geqq 0$ (0보다 작으면 cone 외부에 있는 것으로 간주)이면, $db_m < db_n$이 된다. 따라서 이 때, TCONEi < TCONEn 이므로 상위접선은 변하지 않으므로 TCONEn의 계산은 불필요하다. 한편, $db_m < db_n$에서 만약 $db_n \leqq e$ (허용오차)이면, BCONEj ≧ BCONEn이므로, 역시 BCONEn의 계산이 필요없고, $db_n > e$이면, 새로운 최대 하위접선은 BCONEn이 된다. 그런데 만약, $dt_{n+k-1} \geqq dt_{n+k}$ $(k=1,2,\cdots)$이라고 하면, 위와 같은 이유로 $db_{n+k+1} < db_{n+k}$이고, 이 경우, 하위접선은 적어도 p_{n+k}까지 계속 갱신되면서, p_{n+k}까지 같은 line segment로 된다. 따라서, 이와같이 접선으로부터 거리가 계속 짧아지는 경우, 하위접선을 계속 갱신할 필요없이, 최종적으로 p_{n+k}에서 생성되는 $BCONE_{n+k}$를 하위접선으로 하면 된다. 한편, 초기에 원점으로, 일정한 수의 점까지 R의 값은 e보다 작을 수 있고, 이 때, 거리오차는 e 미만이므로 접선을 역시 계산할 필요가 없다. 그리고, 위의 결과는 상하위접선 바꾸어도 결과가 같으므로, 이상으로 부터, 그림 3과 같이 조건에 따라 상하접선중의 한쪽 또는 양쪽의 계산 또는 한점으로 부터 두접선까지의 거리중 하나의

```
if last_R<=e then begin
    if R>e then find TCONE and BCONE
    end
else
    if R<last_R then begin
        if dt>e then find TCONE;
        if db>e then find BCONE;
        end
    else
        if dt>e then begin
           if db>e then  find TCONE and BCONE
           else if db>last_db then find TCONE
           end
        else begin
           if bt>e and dt>last_dt then find BCONE
        end
```

그림 3. 상하위접선 갱신 조건
Fig. 3. Conditions for updating cone.

계산을 피할 수 있다.

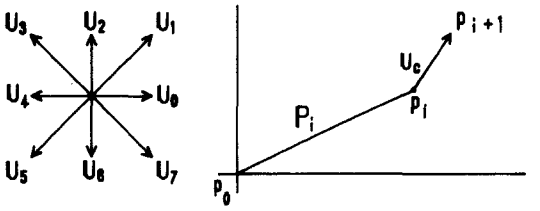
2. 변수의 정수화

1) 길이의 계산 및 비교

그림 3의 알고리듬에서와 같이 거리는 비교를 위해서만 사용되므로 반드시 정확한 실제값이 필요한 것이 아니고, 상대적인 비교가 가능하도록 하면 된다.

한편, 화소간의 거리가 1 또는 $\sqrt{2}$로 표시되는 화상의 윤곽선의 경우 길의의 비교는 정수의 가감산 또는 비교에 의해 정확하게 이루어진다.

그림 4 (a)에서 화소간 8 방향 단위벡터를 U0, U1, …, U7으로 정의하고, 그림 4 (b)에서 원점부터 어떤 점 pi까지를 벡터 P_i라 할 때, P_i를 서로 이웃하는



(a) (b)

그림 4. 8 방향 단위벡터 및 두점간의 벡터
(a) 단위벡터
(b) 두점간의 단위벡터

Fig. 4. 8 directional unit vectors and a unit vector between 2 points.
(a) unit vectors.
(b) a unit vector between 2 points.

두종류의 8 방향 단위벡터로 표현할 수 있다.[8,9]

$$P_i = m*Ua + n*Ub \quad (1)$$

(여기서 m, n은 0 이상의 정수이고, a, b는 연속된 정수로, a는 짝수, b는 홀수이다. 0은 7과 연속된 정수로 간주한다.)
가 된다. 한편, |Pi|는

$$|Pi| = \sqrt{(m+n)^2+n^2} \quad (2)$$

이다. 따라서, P_i의 길이는 m과 n이 결정되고, a가 짝수, b가 홀수이면, 단위벡터 Ua, Ub가 어떤 종류이든 관계없이 일정하다. 그리고, 만약 Pi에 단위벡터 Uc을 더하여 이를 P_{i+1}라 하고,

$$P_{i+1} = P_i + Uc = m'*Ud + n'*Ue \quad (3)$$

(c=0, 1, …, 7 이고, d, e는 연속된 정수로 d는 짝수, e는 홀수이다.)
라 할 때,

[정의] m+n과 m'+n'의 차이는 1 이하이다.
[증명]

사선방향의 벡터 Ub를, Ub=Ua+Uf (a, f는 짝수)로 나타내면, Pi와 Uc는 각각,

$$P_i = (m+n)*Ua + n*Uf \quad (4)$$
$$Uc = s*Ua + t*Uf \quad (5)$$

로 표현되고, 여기서 s, t는

$$-1 \leqq s,\ t \leqq 1 \quad (6)$$

인 정수이다. 따라서, P_{i+1}은 (3), (4), (5)로 부터,

$$P_{i+1} = (m+n+s)*Ua + (n+t)*Uf \quad (7)$$

가 된다. (7)식으로 부터,

i) m+n+s>n+t≧0 이면,

$P_{i+1} = (m+s-t)*Ua + (n+t)*Ub$

이므로 m'+n'=m+n+s 이고, |(n'+m')−(m+n)| = |s| ≦ 1 이다.

ii) n+t>m+n+s≧0 이면,

$P_{i+1} = (t-m-s)*Uf + (m+n+s)*Ub$

가 되고, 이 때, m'+n'=n+t이고, |(n'+m')−(m+n)| = |t−m| 이다. 한편, m+n+s>n+t에서 t−m>s 이므로 이를 만족하는 t−m은 0 또는 1이다.

iii) m+n+p<0인 경우,

m=n=0 이고, p=−1이 되므로,

$P_{i+1} = Uc$

이다. 이때, $m'+n'=1$ 이므로, 정리가 성립한다.

iv) $n+t<0$ 이고, $m+n+p \geqq 0$ 이면,

$n=0$, $q=-1$ 이므로, 식 (6)으로 부터,

$P_{i+1}=(m+s)*Ua+Ug$ $(Ug=-Uf,\ g<>f)$

여기서 $(m+s)=0$ 이면, $m'+n'=1$ 이므로 정리가 성립한다. 만약 여기서 $(m+s) \geqq 1$ 이면,

$P_{i+1}=(m+s+1)*Ua+Uh$ $(Uh=Ug+Ua)$

이므로, $m'+n'=m+s$ 이고, $|(n'+m')-(m+n)| = |s| \leqq 1$ 이다.

한편 P_i와 P_{i+1}의 길이의 비교는 만약

i) $m'+n'=m+n$ 이고, $n'=n$ 이면,

$|P_{i+1}| = |P_i|$

ii) $n'+m'>m+n$ 이거나, $m'+n'=m+n$ 이고, $n'>n$ 이면,

$|P_{i+1}| > |P_i|$

iii) $m'+n'<m+n$ 이거나, $m'+n'=m+n$ 이고, $n'<n$ 이면,

$|P_{i+1}| < |P_i|$

가 된다. 여기서 Ua, Ub, m, n은 x_i와 y_i의 비교 및 감산에 의해서 간단히 계산된다. 예를 들면, $x_i \geqq 0$, $y_i \geqq 0$, $y_i<x_i$ 이면, 식 (1)에서, $Ua=0$, $Ub=1$ 이고, 또, $m=x_i-y_i$, $n=y_i$가 된다.

2) 접점

원의 접점의 위치를 정수화하기 위해, 여기에서는 원을 8각형 근사화하였다. 근사화 방법은 그림 5(a)에서와 같이 두면,

$$|s*U1| = |2*h*U0| \quad (8)$$

$$e=s+h \quad (9)$$

가 되고, 이를 다시 표현하면,

$$h=(\sqrt{2}-1)*e \quad (10)$$

$$s=e-h \quad (11)$$

이 된다. 여기서 s, h 값을 정수화하기 위해 h 값의 소숫점 이하를 반올림 또는 버림등을 하여 정수화한다. 반올림하는 경우, e=10일 때, 중심으로부터 가장 먼점의 거리는 10.77이고, 가장 가까운 거리는 9.90이 된다. 이와 같이 대체로 e가 10이하일 경우 반지름의 길이는 한 화소 미만의 오차를 갖게 된다.

이상과 같이 원을 8각형 근사화하면, 임의의 점으로 부터 접점의 위치는 그림 5 (b)와 같이 8구역으로 나누어지며, 각 구역마다 접점의 위치는 일정하게 된다. 예를 들면, 점 p0로 부터 원의 상위 접선은 접점 ti를 지나고, 하위의 접선은 그림 6 (c)와 같이 점 bi를 지나게 된다.

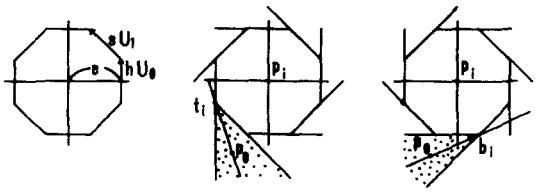


그림 5. 원의 8각형 근사화 및 접점위치
(a) 8각형 근사화된 원
(b) 상위 접점
(c) 하위 접점

Fig. 5. Octagonal approximation of circle and position of tangent points.
(a) octagonal approximation of circle,
(b) upper tagent point,
(c) lower tangent point.

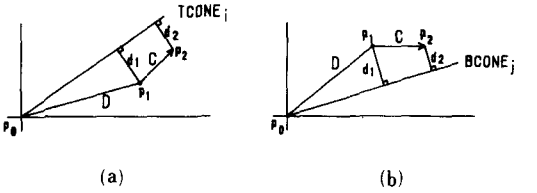


그림 6. 점과 접선 간의 거리
(a) 상위접선까지 거리
(b) 하위접선까지거리

Fig. 6. Distance from a point to tangent line.
(a) to upper tangent line,
(b) to lower tangent line.

3) 접선으로 부터 임의의 점까지 거리비교

그림 6 (a)와 같이 이웃하는 두점 p1, p2가 있고, 이 점들로부터 두 점까지의 거리를 각각 d1, d2라 하면, d1과 d2의 크기 비교는 직선의 기울기 a와 두 점을 연결하는 8방향 체인코드 C(C=0, 1, 2, …, 7)에 의해서 결정된다. 만약 직선의 기울기가 $0<a<1$ 이면, C가 5, 6, 7, 0 중에 하나면, d2<d1 이고, 아니면, d2>d1이 된다. 직선의 기울기가 $0 \leqq a<1$일 때, 2.3절의 디지탈직선 생성 알고리듬에 의하면, 이 직선은 8방향 체인코드 0 또는 1로 표현된다. 디지탈 직선의 체인코드 종류중 크기가 작은 것(0과 7인 경우 7)을 D(예를 들어, $0 \leqq a<1$이면, 체인코드 종류는 0 또는 1을 가지므로 이 때, D=0이다.)로 표현하면, 그림 6 (a)와 같이 점들이 직선아래에 있는 경우 D, C 및 d1, d2의 관계는,

```
if D has two kinds of chain code then
  if D<C and C≦D+4 or C≦D-4 then d2<d1
                            else d2>d1
else
  if C=D or C=D-4 or C=D+4 then d2=d1
  else if D<C and C<D+4 or C<D-4 then d2<d1
                                 else d2>d1
```

가 되고, 그림 6 (b)의 경우, d1, d2의 관계의 부등호가 위의 결과와 반대가 된다. 한편, 접선으로부터 한 점까지의 거리는 접선상의 주어진 기준점과 이웃 두 점을 포함하여 세점(그림 3에서 R≧last-R인 경우, 접선진행방향으로 주어진 점과 우측점 두점, R<last-R인 경우, 주어진 점과 좌측점 두점)으로 부터 가장 짧은 거리를 한 점과 접선간의 거리로 한다. 그리고 가장 짧은 거리를 나타내는 접선상의 점을 다음 기준점으로 하고, 만약 두개의 점에서 같은 짧은 거리가 계산되면, 현재의 기준점이 아닌 다른 점을 다음 기준점으로 한다. 이때 거리계산은 1)절과 같은 방법으로 하고, 접선상에 주어지는 최초의 기준점은 2)절에서와 같이 생성되는 접점으로 한다.

4) 디지탈 직선 생성 알고리듬

상위 접선의 식을

$$yt = gt/ct * xt \qquad (gt, ct는\ 정수)$$

또 하위접선의 식을,

$$yb = gb/cb * xb$$

라 하고, 두 접선의 기울기가 0과 1 사이라고 하면, 두 접선사이(cone)에 있는 임의의 점의 좌표를 (i, j) (i, j≧0인 정수)는,

$$\lceil gb/cb * i \rceil \leqq j \leqq \lfloor gt/ct * i \rfloor$$

가 된다. 이는 x, y의 대칭을 이용하여 모든 영역에 적용할 수 있다.

한편, 위의 식의 좌우측항을 만족하는 디지탈 직선의 생성은 Pham 알고리듬[10]으로부터, 쉽게 구할 수 있다(그림 7).

위의 알고리듬에서 direction은 단위벡터의 종류를 말하며, 0인 경우, 짝수 단위벡터(체인코드)를, 1인 경우 홀수 단위벡터들 의미한다. 예를 들면, 하위접선의 기울기가 1/3인 직선일 때, 위의 그림 5 (b)알고리듬을 적용하면, 초기에 eb=0, gb=1, cb=3이 되고, 이를 한번씩 수행하면, direction의 순서는 100100… 순으로 만들어진다. 이때, 1은 단위벡터 U1을, 0는 단위벡터, U0를 의미하므로, 최초의 점이 (0, 0)일 때, 다음 점들은, (1, 1), (1, 2), (1, 3), (2, 4)… 으로 된다. 한편, 3)절에서 정의한 접선의 방향 D가 짝수인 경우에, 상위접선은 그림 5의 (a) 알고리듬

```
et:=et-gt;
if et>0 then direction:=0
        else begin
             et:=et+ct; direction:=1;
             end
```

(a)

```
eb:=eb+gb;
if eb<=0 then direction:=0
         else begin
              eb:=eb-cb; direction:=1;
              end
```

(b)

그림 7. 디지탈 직선 생성 알고리듬
(a) 디지탈 직선 $j = \lfloor gt/ct * i \rfloor$ 생성 알고리듬
(b) 디지탈 직선 $j = \lceil gb/cb * i \rceil$ 생성 알고리듬

Fig. 7. Digital line generation algorithm.
(a) digital line $j = \lfloor gt/ct * i \rfloor$ generation algorithm,
(b) digital line $j = \lceil gb/cb * i \rceil$ generation algorithm.

을, 하위접선의 경우는 그림 5 (b)의 알고리듬을 적용하면 되고, D가 홀수인 경우 이와 반대의 알고리듬을 적용하면 된다. 그리고, 만약 진행방향이 기울기와 반대방향으로 가는 경우, 반대의 알고리듬을 적용하는데, 알고리즘 (a)의 경우, 수행이 끝나면, 항상 et>0이고, 알고리듬 (b)의 경우, 항상 eb≦0이 되므로, 알고리듬을 바꾸어 적용하는 경우, e의 값이 범위에 어긋나면, c를 더하거나 빼서, 범위를 맞춘다. 이와같이 하여, 단위벡터가 정해지면, 이를 현재까지 벡터에서 빼주면 된다.

위의 예에서 알고리듬 (b)를 적용하는 도중, 점(2, 4)에서 역방향으로 진행할 경우, 그림 5 (a)의 알고리듬을 적용하게 되는데, 이 때, eb=−2이고, et>0이므로 et=c−2=1이 된다. 그리고, gt=gb=1, ct=cb=3이므로, 알고리듬을 적용하면, 1001…이 되고, 이것으로 부터 벡터를 구해 하나씩 빼가면, (1, 3), (1, 2), (1, 1), … 등으로 진행된다.

Ⅲ. 알고리듬 및 실험

이 논문에서는 Sklansky 방법과 같이 접선의 각도를 구하는 것이 아니라, 한 점으로부터 접점을 구하고, 시작점과 접점을 연결하는 디지탈직선으로 부터 다음점 사이의 거리를 계산한 다음, 이들을 이용하여, line segment를 추출한다. 다음 그림 8은 본 방법의 알고리듬이다.

```
for all points do begin
    if R<=e then begin
       calculate R;
       if R>e then find TCONE and BCONE
       end
    else begin
       calculate R;
       find dt and db;
       if bt<0 or dt<0 then begin
            Mark last point (lx,ly) as break point;
            if R>=e then find TCONE and BCONE
          end
       else
          if R<last_R then begin
               if dt>e then find TCONE;
               if db>e then find BCONE;
               end
          else
               if dt>e then begin
                  if db>e then  find TCONE and BCONE
                  else if db>last_db then find TCONE
                  end
               else begin
                  if bt>e and dt>last_dt then find BCONE
                  end
    end;
    last_R:=R;
    (lx,ly):=(x,y)
end;

R : 시작점에서 현재점까지 거리
TCONE : 상위접점 위치계산
BCONE : 하위접점 위치계산
find dt and db : 접선으로부터 거리계산 및 다음 기준점 위치결정
dt>last_dt, db>last_db : 2,3절과 같이 비교
```

그림 8. 다각형 근사화 알고리듬
Fig. 8. Algorithm of polygonal approximation.

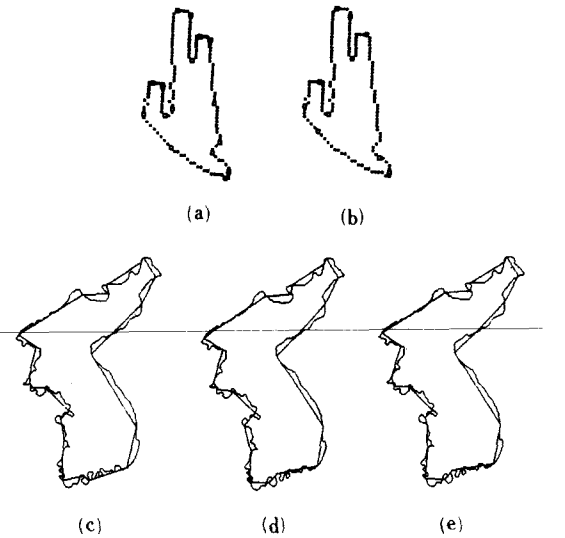


그림 9. 다각형 근사화 예
Fig. 9. Example of polygonal approximation.

그림 9의 (a), (b)는 Dunham[11]이 여러 알고리듬의 비교에 사용한 염색체 그림으로 허용오차가 각각 1, 2일 때 결과(Sklansky 방법, Williams 방법 및 본 방법의 결과가 일치)로서 화소수가 148개인 비교적 단순한 도형의 경우이고, 그림 9의 (c), (d), (e)는 곡선을 구성하는 화소의 수가 1177개인 비교적 복잡한 화상으로 허용오차가 10인 경우, 본방법(s, h는 소숫점 이하 반올림), Sklansky 방법 및 Williams 방법을 적용했을 때의 결과이다. 한편, Table 1은 위 그림에 대하여, 3가지 방법에 대해, 허용오차를 각각 달리하여 얻어지는 break point수와 수행속도의 비교표이다. 비교표에서 보는 바와 같이 본 방법은 break point 수는 Sklansky 방법 및 Williams 방법과 거의 일치하면서, 속도는 보다 빠른 것을 나타내고 있다. (IBM-PC AT, 20MHz, PASCAL 언어사용).

표 1. 그림 9의 실험결과(break point수/수행시간(초))
Table 1. Results of experiment for fig. 9. (no. of break points/execution time (sec))

입력	허용오차	본방법	Sklansky방법	Williams방법
염색체	1	16 (0.059)	16 (0.385)	16 (0.286)
	2	10 (0.052)	10 (0.384)	10 (0.280)
	3	9 (0.049)	9 (0.390)	9 (0.260)
	4	7 (0.048)	7 (0.373)	7 (0.257)
한국지도	1	181 (0.483)	179 (3.46)	180 (2.31)
	2	102 (0.395)	100 (3.46)	101 (2.25)
	5	41 (0.297)	42 (3.19)	42 (2.09)
	10	22 (0.269)	22 (2.96)	23 (1.98)
	22	10 (0.258)	12 (2.96)	12 (1.98)

Ⅳ. 결 론

곡선의 다각형 근사화 방법은 데이타 압축 및 영상 분석에 많이 이용된다. 이에 관하여 많은 알고리듬들이 제시되었으나, cone intersection[6,7]에 의한 방법이 비교적 좋은 결과를 주는 것으로 Dunham[11]은 실험으로 보여준 바 있다. 그러나 이 방법은 실수의 곱셈 및 제곱근 또는 삼각함수의 계산에 의존해야 하므로 그 속도가 늦은 것이 단점이라 할 수 있다.

한편, 화상분석 또는 데이타 압축을 위해 제공되는 입력화상이 거리가 균등한 화소로 이루어진 경우가 많이 있다. 이 논문에서는, 이들 화상내의 주어진 곡선이 이웃절들 간의 거리가 1 또는 $\sqrt{2}$인 경우에, Sklansky 방법을 수정하여, 각도 비교를 길이의 비교 전환하고, 이들 거리 및 접점위치의 변수를 정수로 근사화함으로서, 속도를 향상시켰다. 또한 정수에 의한 근사화에도 불구하고 e가 10 이하일 경우, 다른 방법과 break point 위치 및 숫자가 거의 일치하여, 다른 cone intersection 방법의 효율성을 그대로 유지한다.

參 考 文 獻

[1] U. Ramer, "An iterative procedure for the polygonal approximation of plane curves," CGIP 1, pp. 244-256, 1972

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

[3] Y. Kurozumi and W.A. Davis, "Polyginal approximation by the minimax method," CGIP 19, pp. 248-264, 1982.

[4] S. Shlien, "Segmentation of digital curves using linguistic techniques," CVGIP 22, pp. 277-286, 1983.

[5] K. Wall and P.E. Danielsson, "A fast method for polygonal approximation of digitized curves," CVGIP 28, pp. 220-227, 1984.

[6] C.M. Williams, "An efficient algorithm for the piecewise linear approximation of planar curves," CGIP 8, pp. 280-293, 1978.

[7] J. Sklansky and V. Gonzales, "Fast polygonal approximation of digitizec curtves," PR 12, pp. 327-331, 1980.

[8] H. Freeman, "On the encoding of arbitrary geometric configuration," *IRE Trans. on Electronic computers* 10, pp. 260-268, 1961.

[9] A. Rosenfeld, "Digital straight line segment," *IEEE Trans. on Comput.* C-23, pp. 1264-1269, 1974.

[10] S. Pham, "Digital straight segments," CVGIP 36, pp. 10-30, 1986.

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

著 者 紹 介

柳 承 必 (正會員) 第26卷 第12號 參照
현재 한국 원자력 연구소
선임 연구원

●

權 五 錫 (正會員) 第26卷 第12號 參照
현재 충남대학교 전자계산기
공학과 조교수

金 太 均 (正會員) 第26卷 第12號 參照
현재 충남대학교 전자계산기
공학과 부교수