論文 88-25-1-8

이차원 곡선의 고속 다각형 근사화 방법에 관한 연구

(A Study on the Fast Method for Polygonal Approximation of Chain-Coded Plane Curves)

趙 顯 喆*, 朴 來 弘**, 李 商 郁***

(Hyun Chul Cho, Rae Hong Park and Sang Uk Lee)

要 約

물체 형태묘사(shape description)를 위하여 물체의 둘레로 주어지는 이차원 곡선의 sequential한 고속 다각형 근사화 방법을 제안하였다. 제안된 방법은 한점에서부터 직선까지의 거리오차에 의해 다각형 근사화를 하게되며 직선의 기울기의 세분화를 통하여 그 성능을 향상시켰다. 또한 인간의 시각적인 특성을 고려하여 이를 구현하기 위하여 누적거리 오차와 가변 거리오차 임계값을 제안하였다.

제안된 방법의 수행속도 면에서의 효율성을 확인하기 위하여 기존의 방법들 중에서 가장 고속으로 알려진 Wall의 방법과 비교하였다. 그 결과 성능면에서는 거의 유사한 반면 수행속도는 제안된 방법이 Wall의 방법보다 2배이상 빠름을 실험을 통하여 확인하였다.

Abstract

For shape description, a fast sequential method for polygonal approximation of chain-coded plane curves which are object boundaries is proposed. The proposed method performs polygonal approximation by use of the distance error from one point to a line, and its performance is enhanced by the smoothed slopes of lines. Furthermore, accumulated distance error and variable distance error threshold are proposed in order to consider and implement the visual characteristics of the human being.

I. 서 론

컴퓨터 비젼(computer vision)의 목표라고 할 수 있는 형태 이해와 인식의 전단계로서 형태묘사(shape description)를 하게 되며 특히 물체 형태의 효율적이고 간결한 묘사는 영상의 특징추출(feature extraction), 데이타 압축(data compression), 잡음제거 등에 매우 유용하며[1] 이를 위하여 형태 묘사에 필요한 점(분할점)만을 찾아서 이 분할점들 사이를 직선으로 근사화시키는 다각형 근사화 방법이 많이 사용되고 있다.

일반적으로 다각형 근사화 방법은 local한 방법과 global한 방법으로 구분되며[2,3] local한 방법에는 곡선의 화소들을 차례로 조사하여 곡률과 같은 곡선의 묘사 요소(descriptive element)가 최대, 최소, 0, 또는 어떤 임계값(threshold) 이상 혹은 이하가 되는 점을 분할점으로 정하는 방법[4-6]과 디지탈 직선의 구조적

*準會員, 國防科學研究所
(Agency for Defence Development)
**正會員, 西江大學校 電子工學科
(Dept. of Elec. Eng., Sogang Univ.)
***正會員, 서울大學校 制御計測工學科
(Dept. of Control and Instru. Eng., Seoul Nat'l Univ.)

接受日字 : 1987年 2月 18日
(※ 본 연구는 과학기술처 특정과제 연구비 지원하에 수행되었음.)

이며 규칙적인 성질을 이용한 linguistic방법[7,8]등이 있으며 global한 방법은 전체 곡선을 미리 정한 직선이나 원과 같은 model segment로 어떤 임계값 내에서 근사화가 될 때까지 계속 나누거나 합하면서 분할점을 결정하는 것이다.[1,9] Local한 방법은 모든 화소의 정보를 동시에 필요로 하지 않고 수행속도가 빠르기 때문에 필요한 기억장치가 절약되고 실시간 처리도 가능한 장점이 있는 반면에 불필요한(suboptimal) 분할점들이 많이 생기는 단점이 있으며 global한 방법은 대부분 이와는 반대의 장단점을 갖는다.

본 논문에서는 local한 방법으로서 sequential한 형태의 고속 수행이 가능한 다각형 근사화 방법을 제안하였다. 서론에 이어 Ⅱ장에서는 제안된 다각형 근사화 방법을 설명하고 Ⅲ장에서는 제안된 방법과 기존의 고속 알고리즘인 Wall의 방법의 계산량을 비교하였으며 Ⅳ장에서는 실험을 통하여 얻은 두 방법의 다각형 근사화 결과와 그 수행 시간을 비교하였고 Ⅴ장에서 결론을 내렸다.

Ⅱ. 제안된 고속 다각형 근사화 방법

여기서는 본 논문에서 제안한 다각형 근사화 방법에 대하여 설명한다. 이 방법은 이웃한 두 점을 연결하는 직선과 그 이후의 점들간의 누적거리오차(accumulated distance error)와 가변 거리오차 임계값(variable distance error threshold)을 이용해 분할점을 구하여 다각형 근사화를 하는 방법이다. 이 방법은 곡선의 local한 위치 정보만을 사용하여 화소들의 좌표들이 입력되는 대로 분할점의 여부를 결정하는 sequential한 방법으로서 suboptimal한 결과를 낳지만 고속 수행이 가능하여 실시간 처리와 같이 고속처리가 요구되는 환경에 적합하다.

1. 방법

n개의 chain code로 이루어진 이 차원 상의 폐곡선 $P=\{P_i(X_i, Y_i), i=1, \cdots, n\}$을 고려한다. 그림 1과 같이 곡선상의 두점 P_i와 P_{i+1}을 연결하는 직선의 식을 $y=f(x)$ (혹은 $x=g(y)$), 원점으로부터 이 직선까지의 거리를 d, x축과 d가 이루는 각도를 θ, 그리고 그 이후의 또 다른 한 점 $P_k(X_k, Y_k)$ $(k>i+1)$에서 이 직선까지의 y축 방향으로의 거리(| 기울기 | ≤1) 혹은 x축 방향으로의 거리(| 기울기 | >1)를 E_k라고 하면 두점을 연결하는 직선의 식과 E_k는 식(1), (2)로 주어진다.

$$x\cos\theta + y\sin\theta = d \qquad (1)$$

$$y = \{(Y_{i+1}-Y_i)/(X_{i+1}-X_i)\} * (x-X_{i+1}) + Y_{i+1} \quad (\text{기울기} \neq \infty)$$

$$x = X_i = X_{i+1} \quad (\text{기울기} = \infty)$$

$$E_k = \begin{cases} |f(X_k)-Y_k|, & (|\text{기울기}| \leq 1) \\ |g(Y_k)-X_k|, & (|\text{기울기}| > 1) \\ |X_k-X_{i+1}|, & (\text{기울기} = \infty) \end{cases} \qquad (2)$$

이제 점 $P_i(X_i, Y_i)$와 $P_{i+1}(X_{i+1}, Y_{i+1})$을 연결하는 직선의 방정식을 식(1)에 따라 구하고 $P_k(X_k, Y_k)$ $(k>i+1)$부터 시작하여 거리오차 E_k를 구하여 E_k가 일정한 임계값 이하이면 P_k는 이 직선에 포함되는 것으로 판단하여 비분할점이 되고 E_k가 임계값 보다 큰 값이 나타날 때까지 k를 하나씩 증가시키면서 위의 과정을 반복한다. E_k가 임계값 보다 큰 점을 P_{c+1}이라고 하면 그 바로 전의 점 $P_c(X_c, Y_c)$를 분할점으로 결정하고 P_i부터 P_c까지를 하나의 직선 성분으로 근사화할 수 있게 된다. 계속해서 P_c와 P_{c+1}을 연결하는 직선의 식을 고려하여 나머지 점들에 대해서 이 전의 과정을 반복하게 된다.

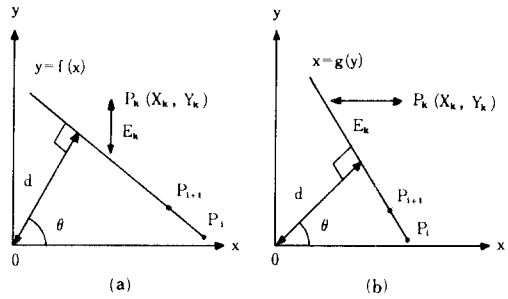

그림 1. 점 P_k에서 직선까지의 단순거리오차 E_k
(a) | 기울기 | ≤1 (b) | 기울기 | >1

Fig. 1. Distance error E_k from P_k to a line formed by P_i and P_{i+1}
(a) | slope | ≤1. (b) | slope | >1.

2. 직선의 기울기의 세분화 (Smoothed Slopes)

앞 절에서 제안한 방법의 가장 큰 문제점은 chain code로 이루어진 곡선의 경우에 그림 2와 같이 이웃한 두 점을 연결하는 직선의 기울기는 항상 45°의 배수로만 이루어지므로 직선의 기울기가 0, 1, ∞, −1 의 4경우로 제한된다는 기울기 표현상의 제약 또는 부정확성이라고 할 수 있고 이것으로 인하여 불필요한 분할점이 너무 많이 발생하는 경우가 빈번하게 된다. 따라서 이 문제점을 해결하기 위하여 시작점 P_i와 이웃점 P_{i+1}에 의한 직선 대신 P_i와 P_{i+2}에 의한 직선을 사용하여 좀 더 기울기의 표현을 세분화 할 수 있도록 하였다.

P_i와 P_{i+2}에 의한 직선을 사용하였을 때 생길 수 있는 기울기는 1, 2, ∞, 1/2, −1, −2, −1/2, 0의 8가지 경우가 있게되므로 P_i와 P_{i+1}에 의한 직선의 기울기의 경우 보다 그 표현이 세분화됨을 알 수 있다. 또한 P_{i+1}

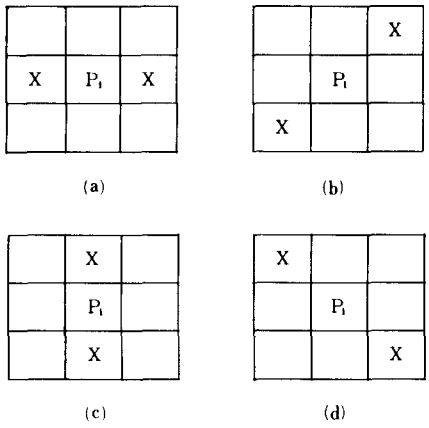

그림 2. P_i와 P_{i+1}에 의한 직선의 기울기
(X : P_{i+1}의 존재 가능위치)
(a) 기울기 = 0 (b) 기울기 = 1
(c) 기울기 = ∞ (d) 기울기 = −1

Fig. 2. The slopes of a line formed by P_i and P_{i+1}
(X : possible position of P_{i+1}).
(a) slope = 0. (b) slope = 1.
(c) slope = ∞. (d) slope = −1.

에서 이 직선까지의 거리 오차는 최대 1이 되므로 (기울기 = 0 또는 ∞인 경우) 임계값을 1 이상으로 하면 P_{i+1}은 이 직선에 포함됨을 알 수 있다. 이것을 더욱 확장하면 P_i와 P_{i+3}에 의한 직선의 식을 사용할수 있고 이 경우에는 기울기가 16개로 더욱 세분화된다. 이러한 세분된 기울기를 갖는 직선을 사용하여 얻을 수 있는 효과의 한 예를 그림 3에 나타냈다. 그림 3(a), (b)는 각각 4, 8개의 기울기를 사용한 경우인데 여기서 보듯이 8 개의 기울기를 사용한 경우가 4 개의 기울기를 사용한 경우보다 불필요한 분할점이 덜 생기는 것을 알 수 있다.

3. 누적 거리오차와 가변거리 오차 임계값

그림 4 (a), (b)에서와 같이 동일한 사이각 θ와 같은 길이의 직선성분 bc를 갖는 두 곡선의 경우에 인간의 눈에는 그림 4 (a) 보다는 (b)가 즉 ab의 길이가 길수록 직선 성분bc의 현저함이 약하게 느껴진다.

이러한 인간의 시각적인 측면을 고려하면 모든 점에서 동일한 거리오차 임계값을 적용하는 것보다는 직선의 시작점에서 멀리 떨어진 점일수록 보다 큰 거리오차를 허용하는 것이 바람직함을 알 수 있다. 따라서 여기서는 점 P_i와 P_{i+3}에 의한 직선의 경우를 가정하고 단순거리 오차 E_k와 고정거리 오차임계값 대신 점 P_k에서의 누적거리 오차와 가변거리 오차임계값을 사용한다. 점 P_k에서의 단순거리 오차, 누적거리 오차, 고정거리 오

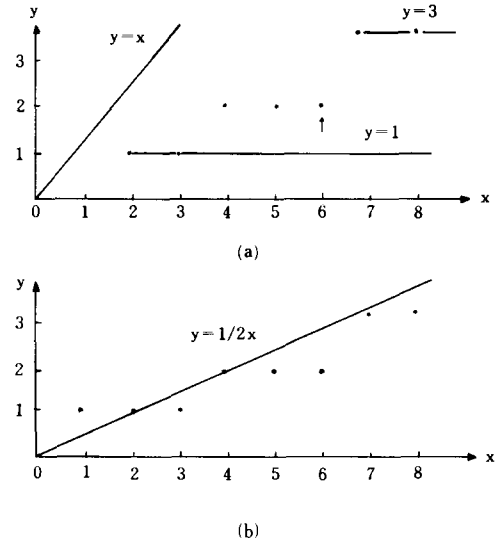

그림 3. 세분화된 기울기의 효과의 한 예
(↑ = 분할점, 거리오차 임계값 = 1)
(a) P_i와 P_{i+1}에 의한 직선들
(4 개의 기울기 : 2개의 분할점)
(b) P_i와 P_{i+3}에 의한 직선
(8 개의 기울기 : 분할점 없음)

Fig. 3. An expamle of the effect of smoothed slopes
(↑ = critical pts., distance error th. = 1).
(a) Lines formed by P_i and P_{i+1}
(4different slopes : 2 critical pts.).
(b) Line formed by P_i and P_{i+3}
(8 different slopes : 0 critical pt.).

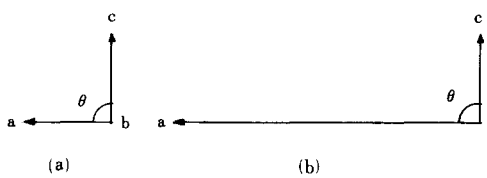

그림 4. 동일한 사이각 θ와 같은 길이의 직선 성분 bc를 갖는 두 곡선

Fig. 4. 2 curves with identical angle θ and line segment bc.

차 임계값, 가변거리 오차임계값을 각각 E_k, AE_k FT, VT_k라 하고 다음과 같이 정의한다.

$$AE_k \equiv AE_{k-1} + E_k \quad (k \geq i+3,\ AE_{i+3} \equiv 0) \qquad (3)$$

$$VT_k \equiv FT * D_k \qquad (4)$$

$$D_k \equiv D_{kx} \equiv |X_k - X_{i+3}| \quad (|\text{기울기}| \leq 1) \qquad (5)$$

$$D_{ky} \equiv |Y_k - Y_{i+3}| \quad (|\text{기울기}| > 1)$$

P_{i+2}에서 멀리 떨어진 점일수록 가변 임계값 VT_k는 더욱 커지게 되며 따라서 더 많은 거리 오차가 허용된다. 또한 동일한 가변 임계값 VT_k의 경우에 P_k이전의 점들의 직선과의 거리 오차에 따라 P_k의 거리오차의 허용 범위가 달라진다. 즉 P_k이전의 점들이 직선에 가까이 일치된 경우에는 점 P_{k-1}까지의 누적오차 AE_{k-1}이 작기 때문에 P_k의 거리오차 E_k의 허용범위가 커져서 어느 정도의 거리 오차 E_k로는 가변 임계값 보다 크게 되지 않지만 그렇지 않은 경우에는 누적오차 AE_{k-1}이 크기 때문에 E_k의 허용범위가 좁아져서 AE_k가 금방 가변 임계값을 초과하여 분할점으로 결정된다. 따라서 P_k의 분할점 결정 여부가 P_k의 단순거리 오차 E_k 뿐아니라 그 이전 점들의 거리 오차에도 영향을 받는다는 의미를 지니게 되므로 한 점의 분할점 결정에 인간의 시각 특성을 고려하여 보다 많은 정보를 이용한다는 장점이 있다.

표 1에 제안된 방법의 수행 과정을 Pascal 언어 형태로 요약하여 나타냈다.

Ⅲ. 제안된 방법과 Wall의 방법의 계산량 비교

이 장에서는 제안된 방법의 고속 수행능력을 보이기 위하여 기존의 고속 알고리즘인 Wall의 방법[6]과 그 계산량을 산출하여 비교한다.

제안된 방법의 경우 각 화소에 대한 계산은 E_k, AE_k, VT_k를 계산하고 분할점인 경우에는 새로운 직선의 기울기를 계산하는 것이 필요하며 이를 위해 필요한 계산량은 다음과 같다.

우선 단순거리 오차 E_k를 구하기 위해서는 그림 1(a), (b)의 경우에 (기울기≠∞) 덧셈 1번, 뺄셈 2번, 곱셈 (혹은 나눗셈)과 절대값 연산이 각각 1번씩 필요하며 그림 1(b)의 기울기=∞의 경우에는 뺄셈과 절대값 연산만이 각각 1번씩 필요하다. 그리고 누적거리 오차 AE_k를 구하기 위해 덧셈 1번이 필요하며 가변거리 오차 임계값 VT_k를 구하기 위해 곱셈과 절대값 연산이 각각 1번씩 (기울기=∞인 경우에는 뺄셈 1번 추가) 필요하게 된다. 또한 분할점으로 결정된 점에서는 새로운 직선의 기울기를 구하기 위하여 뺄셈 2번, 나눗셈과 절대값 연산 그리고 | 기울기 | 와 1을 비교하는 연산이 각각 1번씩 필요하게 된다.

한편 단위 길이당 누적 면적편차를 사용하는 Wall의 방법은 한 점에 대하여 덧셈, 뺄셈 각각 5번과 곱셈 3번의 계산량이 필요하며 이 방법의 simplified version인 경우에는 덧셈 2번, 뺄셈 5번, 곱셈 1번, 절대값 연산 3번, 비교 연산 1번의 계산량이 필요하다. 제안된 방법에서 계산량이 많은 경우(기울기≠∞, 0)

표 1. 제안된 방법의 수행과정
Table 1. The procedure of the proposed method.

```
Procedure Approximation;
Begin
 i : =1;
 For the entire coordinates of the boundary Do Begin
     k : =i+3;
     If (X_i − X_{i+2}) is not zero Then Begin
       SLOPE : =ABS((Y_i − Y_{i+2})/(X_i − X_{i+2}));
       If SLOPE< =1 Then Begin
         E_k: =ABS(f(X_k) − Y_k);
         AE_k : =E_k ; VT_k : =FT;
         While AE_k <= VT_k Do Begin
               k : =k+1;
               E_k: =ABS(f(X_k) − Y_k);
               AE_k : =AE_{k-1}+E_k;
               VT_k : =FT * D_kx
         End;
         save the (k−1) th point in the stack;
         (* This point is the critical point. *)
         i : =k−1
       End
       Else Begin
         E_k: =ABS(g(Y_k) − X_k);
         VT_k : =FT * D_ky;
         The same routine as above for
         obtaining the critical points is used
       End
     End
     Else Begin
       E_k: =ABS(X_k − X_{i+2});
       VT_k : =FT * D_ky;
       The same routine as above for
       obtaining the critical points is used
     End
   End
 End; (* Procedure Approximation *)
```

만을 가정하고 두 방법에 필요한 계산량을 표2, 3에 비교하였는데 표 2에서 제안된 방법의 계산량이 Wall 의 방법의 계산량 보다 더 적음을 알 수 있으며 표3에서는 제안방법이 Wall의 simplified version보다 뺄셈, 절대값 연산, 비교 연산의 횟수는 더 작고 곱셈 횟수만 한번 더 많은 것을 알 수 있다.

표 2. 제안된 방법과 Wall의 방법의 계산량 비교
(( ) 안은 분할점인 경우의 계산량)
Table 2. Comparison of computational amounts between the proposed and Wall's methods (( ) : computational amounts in case of critical points).

	덧셈	뺄셈	곱셈	나눗셈	절대값	비교
제안된 방법	2	2(2)	2	(1)	2(1)	(1)
Wall의 방법	5	5	3	−	−	−

표 3. 제안된 방법과 Wall의 simplified version과의 계산량 비교
(( )안은 분할점인 경우의 계산량)

Table 3. Comparison of computational amounts between the proposed and Wall's simplified version.
(( ) : computational amounts in case of critical points).

	덧셈	뺄셈	곱셈	나눗셈	절대값	비교
제안된 방법	2	2(2)	2	(1)	2(1)	(1)
Wall의 방법	2	5	1	-	3	1

Ⅳ. 실험 및 결과

이 장에서는 제안된 방법과 Wall의 방법을 사용하여 분할점을 찾은 실험결과에 대하여 살펴본다. 실험에 사용된 영상은 128×128 크기의 원과 한국 지도 2가지이며 경계검출, thresholding, 세선화, 추적의 전처리 과정을 거쳐 물체둘레의 좌표를 구한 후 이 좌표들을 최종 입력으로 사용하여 CDC Cyber 180-860에서 Pascal 언어로 처리하였다.

그림 5(a), (c)와 6(a), (c)는 고정거리 오차 임계값을 각각 1,2로 했을 때 제안된 방법에 의한 분할점이며 그림 5(b), (d)와 6(b), (d)는 단위길이 당 면적 편차 임계값을 각각 1,2로 했을 때 Wall의 방법에 의한 분할점이다. 제안된 방법의 경우 8개의 세분된 기울기를 갖는 직선을 사용하였으며 두 방법의 수행속도를 정량적으로 비교하기 위하여 CDC Cyber 170-860의 Pascal 언어에서 제공하는 "CLOCK"이라는 함수를 사용하여 계산시간을 측정하고 그 결과를 두 방법에 의한 분할점의 갯수와 함께 표4, 5에 나타냈다.

그림5, 6의 다각형 근사화를 위한 곡선분할 결과를 살펴보면 표4에서 보듯이 임계값이 1인 경우에는 분할점의 수와 그 위치가 거의 비슷함을 알 수 있다. 즉 두 방법은 거의 유사한 다각형 근사화 결과를 낳게 된다. 또한 곡선의 화소들이 일직선을 이루고 있는 경우(한국 지도에서 좌측 하단의 중간부근)에 누적거리 오차는 계속 0인데 반해서 가변거리오차 임계값은 일직선의 거리가 길어질수록 커지기 때문에 화소들이 직선에서 벗어나기 시작해도 어느 정도까지는 이 점들을 비분할점으로 판정하게 되는데 고정거리 오차 임계값이 2인 경우에는 그 정도가 심한 경우가 생길 수 있어서 그림6(d)의 경우에는 좌측 하단에서 상당히 벗어난 점을 분할점으로 결정하는 단점이 있다. 이런 단점은 인간의 시각적인 특성을 고려하기 위해 사용한 거리와 가변 임계값의 단순한 상관관계를 보다 적절히 규정함으로써 보완할 수 있으리라 생각되며 다음과 같은 방법을 고려할 수 있을 것이다. 즉 누적거리 오차와 가변거리 오차임계값이 어느 임계값 이상으로는 차이가 나지 않도록 하면 위에서 언급한 단점을 제거할 수 있으리라 생각되며 이 임계값은 실험을 통하여 정할 수 있을 것이다.

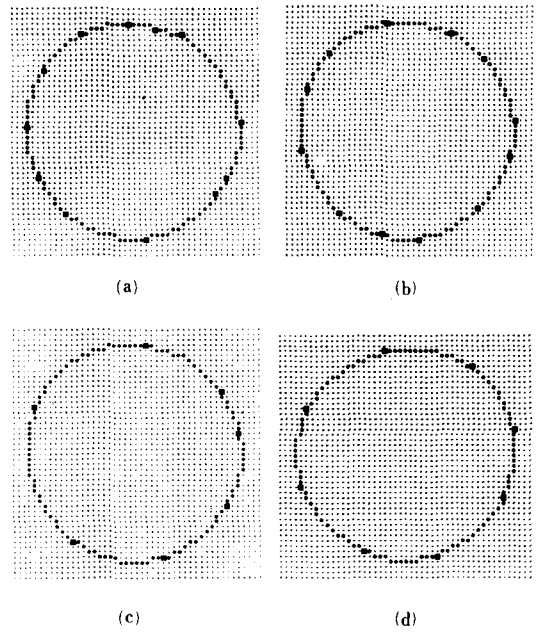

그림 5. 원에 대한 실험결과(ı : 분할점)
(a) 제안된 방법(고정 임계값=1)
(b) Wall의 방법(면적오차 임계값=1)
(c) 제안된 방법(고정 임계값=2)
(d) Wall의 방법(면적오차 임계값=2)

Fig. 5. Results for a circle (ı : critical point).
(a) Proposed method (fixed th. =1).
(b) Wall's method (area deviation th=1).
(c) Proposed method (fixed th. =2).
(d) Wall's method (area deviation th. =2).

표 4. 제안된 방법과 Wall의 방법의 수행 시간과 분할점 수의 비교
(수행 시간/분할점 수, 시간 단위 : msec)

Table 4. Comparison of the computation time and the number of critical points between the proposed and Wall's method (time/number, time unit : msec).

사용영상 / 사용방법	원(화소수:115)		지도(화소수:229)	
	임계값:1	임계값:2	임계값:1	임계값:2
제안된 방법	2 / 12	2 / 7	4 / 37	4 / 27
Wall의 방법	5 / 12	5 / 8	10/35	9 / 18

표 5. 제안된 방법과 Wall의 simplified version과의 수행시간과 분할점 수의 비교 (수행 시간/분할점 수, 시간 단위 : msec)

Table 5. Comparison of the computation time and the number of critical points between the proposed and Wall's simplified version (time/number, time unit : msec).

사용영상 / 사용방법	원(화소수:115)		지도(화소수:229)	
	임계값:1	임계값:2	임계값:1	임계값:2
제안된 방법	2 / 12	2 / 7	4 / 37	4 / 27
Wall의 방법	2 / 10	2 / 6	5 / 19	5 / 12

그러나 수행 속도에서는 제안된 방법이 Wall의 방법보다 2배이상 빠름을 알 수 있으며 Wall의 simplified version과 비교하면 화소수가 적은 원인 경우에는 제안 방법과의 수행속도의 차이를 알 수 없지만 화소 수가 많은 지도의 경우에서 제안방법의 수행속도가 약간 빠름을 알 수 있다. 그런데 표 2 의 계산량 비교에서 예측되는 것 보다 표 4 에서의 실제 계산은 더 차이가 나는 것을 알 수 있는데 이것은 제안된 방법에서는 기울기가 ∞ 혹은 0 인 경우에는 계산량이 더욱 감소하며 또한 P_k가 분할점으로 결정되면 새로운 직선을 구하기 위해 사용되는 점인 P_k, P_{k+2}와 그 사이점 P_{k+1}에 대해서는 필요한 계산량이 없기 때문이라고 생각된다.

V. 결 론

이차원 곡선을 고속으로 다각형 근사화할 수 있는 방법을 제안하였으며 실험을 통하여 기존의 고속 알고리즘인 Wall의 방법에 비해 2 배 이상 빠른 고속 수행 능력을 확인하였다.

제안된 방법에서는 한 점으로부터 직선까지의 거리 오차와 세분화된 직선의 기울기와 함께 인간의 시각 특성을 고려하여 이것을 실현시키기 위하여 누적 거리 오차와 가변거리 오차 임계값을 제안하여 사용하였는데 실험결과 임계값이 1인 경우에는 Wall의 방법과 유사한 결과와 함께 수행 속도면에서 월등히 뛰어났지만 임계값이 2인 경우에는 수행속도의 월등함에 비해 근사화 결과는 약간 떨어지는, 즉 임계값에 상당히 민감한 단점이 관찰되었으며 이런 단점은 거리와 임계값의 상관관계를 보다 정확히 정의함으로써 해결할 수 있으리라고 생각된다.

제안된 고속 다각형 근사화 방법은 일반적인 다각형 근사화 방법들이 사용되는 분야, 즉 구문론적 패턴 인식에서의 패턴 원소(primitive) 추출과 데이타 압축등에 사용될 수 있으며 특히 수화(sign language) 전송[10]과 같이 연속되는 화면을 사용하기 위하여 고속 처리가 요구되는 환경에 적합하다.

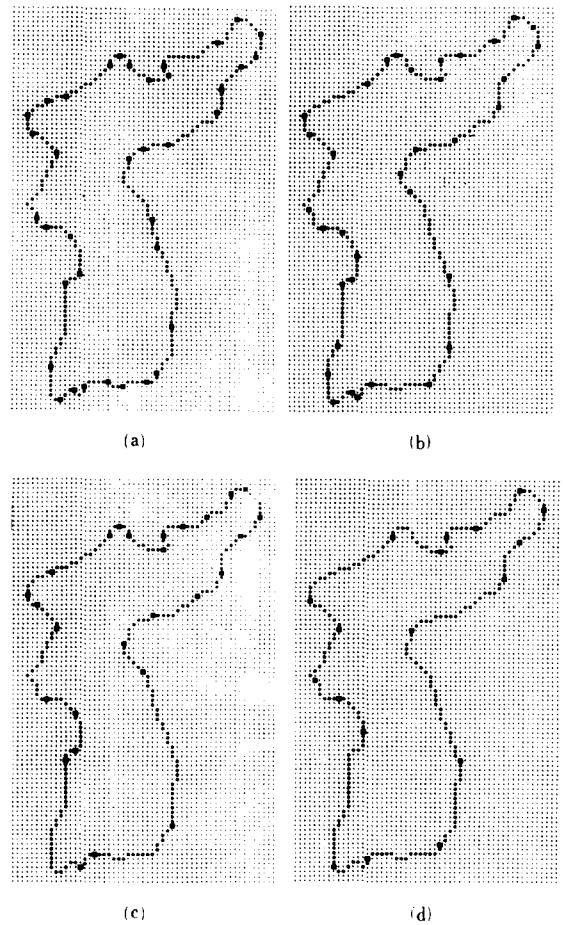

그림 6. 한국 지도에 대한 결과(ι : 분할점)
(a) 제안된 방법(고정 임계값=1)
(b) Wall의 방법(면적오차 임계값=1)
(c) 제안된 방법(고정 임계값=2)
(d) Wall의 방법(면적오차 임계값=2)

Fig. 6. Results for the Korean map (ι : critical point).
(a) Proposed method (fixed th. =1).
(b) Wall's method (area deviation th. =1).
(c) Proposed method (fixed th. =2).
(d) Wall's method (area deviation th. =2).

參 考 文 獻

[1] T. Pavlidis and S.L. Horowitz, "Segmentation of plane curves," *IEEE Trans. Comput.*, vol. C-23, no. 8, pp. 860-870, Aug. 1974.

[2] M.D. Levine, *Vision in Man and Machine*, New York: McGraw-Hill, 1985.

[3] M.A. Fischler and R.C. Bolles, "Perceptual organization and curve partitioning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-8, no. 1, pp. 100-105, Jan. 1986.

[4] A. Rosenfeld and E. Johston, "Angle detection on digital curves," *IEEE Trans. Comput.*, vol. C-22, no. 9, pp. 875-878, Sept. 1973.

[5] H. Freeman and L.S. Davis, "A corner-finding algorithm for chain coded curves," *IEEE Trans. Comput.*, vol. C-26, no. 3, pp. 297-303, Mar. 1977.

[6] K. Wall and P.E. Danielsson. "A fast sequential method for polygonal approximation," *Computer Vision, Graphics, and Image Processing*, vol. 28 pp. 220-227, 1984.

[7] A. Rosenfeld, "Digital straight line segments," *IEEE Trans. Comput.*, vol. C-23, no. 12, pp. 1264-1269, Dec. 1974.

[8] S. Shlien, "Segmentation of digital curves using linguistic techniques," *Computer Graphics and Image Processing*, vol. 22, pp . 277-286, 1983.

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

[10] M.S. Landy and Y. Cohen, "Vectorgraph Coding: Efficient coding of line drawing," *Computer Vision, Graphics, and Image Processing*, vol. 30, pp. 331-344, 1985.