허용 오차를 만족하는 호의 추출
(Detection of Circular Arcs within Tolerant Error)

류 승 필 †
(Sung-Pil Lyu)

요 약 호는 패턴인식에 있어서 매우 중요한 특징으로 간주된다. 이 논문에서는 기하학적 분석을 이용하여 2차원 곡선으로부터 호를 추출하고 그 호의 중심과 반지름을 계산하는 방법을 제시한다.

제안된 방법에 의해 추출된 호는 원래의 곡선으로부터 거리오차가 항상 허용치 이내의 값을 만족한다. 제안된 방법이 부드럽게 연결된 2차원 곡선이나 잡음이 심한 2차원 곡선 내에 있는 호의 추출에도 유용함을 실험으로 확인하였다.

키워드 : 호 추출, 호 근사화, 호의 반지름 및 중심

Abstract Arcs are usually treated as significant features in the field of pattern recognition. This paper presents a method to detect arcs from digital planar curves and estimate their arc centers and radii by using geometric analysis.

The deviation of distance between the original curve and the detected arc by the proposed method does not exceed a tolerant error. The experimental results show that the proposed method is available for the detection of arcs from not only smooth but also heavily noisy curves.

Key words : circular arc detection, circular arc approximation, radius and center of arc

1. 서 론

건설, 기계, 전기, 전자 관련 도면에 많은 심벌들이 사용되고 있으며, 이들 도형 내에 있는 심벌을 인식하여 설계자동화를 위한 노력이 계속되어 왔다. 도형내의 심벌들은 직선과 곡선으로 이루어진 경계를 가지며, 이 경계의 모양은 패턴인식의 중요한 요소로 이용되고 있다. 대부분의 심벌들은 기하학적인 모양을 하고 있으며, 주로 직선, 원 또는 타원의 일부로 이루어진다. 그래서 2차원곡선을 직선 또는 호로 근사화하거나[1-3], 호의 경우 이들의 중심이나 반지름을 찾는 연구가 많이 진행되어 왔다.

호를 추출하기 위한 기본적인 연구 분야로는 주어진 영상으로부터 호의 분리(segmentation)에 관한 연구 분야[4,5], 호의 일부로 가정되는 점들(또는 선분들)로부터 중심을 찾는(estimation) 연구 분야[6,7], 그리고 이미지로부터 호를 분리하고 그것의 중심과 반지름을 추출(detection)하는 연구 분야[8-14]들이 있다.

호의 추출(detection)에 있어서 호의 분리과정은 매우 중요하며, 호의 분리방법으로는 크게 둘로 나눌 수 있는데 하나는 원래 이미지로부터 직접 호를 찾는 방법들[4,5,8-11]이 있고, 또 하나는 경계선 또는 세선화된 골격선 등 두께가 없는 디지털 곡선에 대해서 체인코드나 다각형 근사화된 선분들로 벡터화하여 이들 벡터를 이용하여 이미지로부터 호를 분리하는 방법들이 있다[12-14]. 이중 전자의 방법은 잡음에 강하고 전처리 과정이 필요 없는 대신에 시간복잡도가 매우 높아 고수준의 화질에 적용하는 데에는 한계가 있다. 한편 본 연구의 관심분야인 후자의 방법은 비교적 빠른 처리를 할 수 있으나 경계선 추출 또는 골격선의 세선화와 같은 전처리 과정이 필수적이라 할 수 있다. 후자의 방법으로는 경계선을 다각형 근사화하여 추출된 호와 비교하여 일정한 오차 이내에 있으면 호의 일부로 간주하는 방법[12,13]과 경계선의 곡률의 크기에 따라 호를 분리하고 분리된 호에 대해서 최소자승오차[6]를 이용하여 원의 중심과 반지름을 구하는 방법[14]이 있다.

경계선을 다각형 근사화하여 호를 찾아내는 방법인 Rosin 방법[12]과 Dosch[13]의 방법은 호에서 제외되어야할 경계 일부가 추출할 호의 일부로 처리되므로 이것이 오차의 원인이 되며 이 오차로 인해 호의 분리가 잘 못되는 경우가 생긴다. 한편 Lim[14] 방법의 경우 비록

· 본 연구는 세명대학교 교내학술연구비지원으로 수행되었음
† 종신회원 : 세명대학교 컴퓨터학과 교수
lsp415@ailab.semyung.ac.kr
논문접수 : 2005년 1월 26일
심사완료 : 2005년 7월 29일

경계선이 실제 호로부터 오차 이내에 있다하더라도 곡률변화가 크면 다른 호로 분리되며, 반면에 실제 호로부터 오차가 크더라도 곡률변화가 매우 작은 경우에 분리가 제대로 되지 않는 단점이 있다.

이 논문은 체인코드로 표현되는 경계선으로부터 호를 추출하는 방법으로 기하학적인 분석을 이용하여 경계선으로부터 호를 분리하고 중심과 반지름을 찾아내는 새로운 방법을 제시한다. 제안된 방법은 2차원 곡선상의 점이 입력되는 즉시에 breakpoint(호와 호를 분리하는 점) 판정이 이루어지므로 호의 추출에 있어서 breakpoint 뒤에 연결된 점들의 영향을 받지 않으며 추출된 호로부터의 주어진 호 간의 거리가 항상 허용오차를 만족한다. 본 방법은 부드럽게 연결된 곡선이나, 곡률의 변화가 심하고 잡음이 많은 곡선 내에 있는 호들도 추출이 가능하다.

본 논문 제2장의 2.1절에서는 디지털호의 반지름과 중심후보를 찾는 기본원리를 기술하고, 2.2절은 2.1절에서 제안한 방법을 선형화하여 간단하게 중심후보를 찾는 방법을, 2.3절은 호를 추출하는 알고리즘을 제시하였다. 끝으로 제3장에서는 실험으로 경계선을 이용한 다른 방법들과 비교 분석하였고, 제4장 결론 순으로 기술하였다.

2. 호의 중심 및 반지름 추출 방법

제안하는 방법을 설명하기위해서, Z를 점 P_0, P_1 및 P_{n-1}등 n개의 점으로 이루어져 있는 디지털 곡선을 나타내는 점들의 집합이라 하고,

$$Z=\{P_0,\ P_1,\ P_2,\ \ldots,\ P_{n-1}\}$$

로 표시할 때, 다음을 정의한다.

ARC(Z,i,j) : Z내 $P_i,\ldots,P_j$까지의 점(이후 원주점이라 한다)들의 집합 $(0\leqq i\leqq j\leqq n-1)$

$m(\overline{AB})$, $m(\overrightarrow{AB})$: 점 A에서 B에 이르는 선분 $\overline{AB}$ 또는 벡터 $\overrightarrow{AB}$의 길이

$err(A,B,C) = |m(\overline{CA})-m(\overline{CB})|$

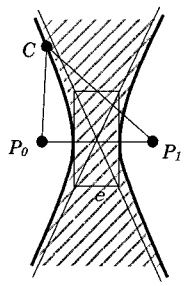
2.1 호추출 기본원리

Z내의 점 P_0와 P_1에 대해서, $err(P_0,P_1,C) \leqq e$라고 하면,

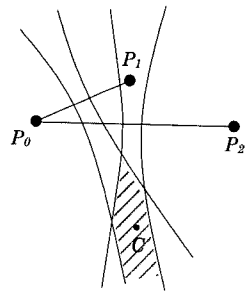
$$|\, m(\overline{CP_1})-m(\overline{CP_0})|\leq e \qquad (1)$$

로 나타낼 수 있으며, 이때 식 (1)을 만족하는 점 C는 그림 1(a)의 쌍곡선들 사이의 영역 내에 존재하는 점들 중의 하나가 된다.

만약 또 다른 점 P_2에 대해서, $err(P_0,P_2,C)\leqq e$ 이 성립된다면, 점 P_0, P_1과 P_2에 대해 위의 조건을 동시에 만족하는 C가 존재하는 영역은 두 개의 쌍곡선 내에 공통영역으로 그림 1(b)의 빗금친 부분과 같으며, 그림



(a) P_0와 P_1로부터 거리차이가 e인 점 C



(b) P_0,P_1 및 P_2로부터 거리의 차이가 e이내인 점 C의 존재영역

그림 1 원주점들로부터 거리의 차이가 허용오차(e)이내에 있는 점들의 영역

2(b)의 공통영역에 있는 임의의 점 C를 중심으로 하고, 반지름을 $\overline{CP_0}$로 하는 원을 그리는 경우 P_0, P_1과 P_2는 모두 원으로부터 e이내에 있게 된다. 그리고 ***ARC*** $(Z,0,n)$ 내의 $0\leqq i\leqq n-1$ 인 i에 대해서

$$err(P_0,P_i,C) \leqq e \qquad (2)$$

를 만족하는 영역의 내의 점들의 집합을 $\boldsymbol{H_i}$라 하고, $\boldsymbol{H_0}$부터 $\boldsymbol{H_i}$까지의 공통영역을

$$\boldsymbol{G_i} = \boldsymbol{H_0} \cap \boldsymbol{H_1} \cap \ldots \cap \boldsymbol{H_i}$$

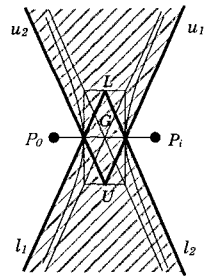
로 표현하자. 여기서 만약 $\boldsymbol{G_i}\neq\boldsymbol{\varnothing}$(공집합)이 아니면, ***ARC***$(Z,0,i)$의 원주점들은 $\boldsymbol{G_i}$ 내의 임의의 점 C를 중심으로 하고 반지름 $m(\overline{CP_0})$인 원으로부터 모두 e이내에 있게 된다. 한편, $0<i+1\leqq n-1$에 대해서 $\boldsymbol{G_i}\neq\boldsymbol{\varnothing}$이 아니고, $\boldsymbol{G_{i+1}}=\boldsymbol{\varnothing}$라고 하면, $err(P_0,P_{i+1},C)\leqq e$을 만족하는 영역 $\boldsymbol{H_{i+1}}$과 $\boldsymbol{G_i}$와의 공통영역이 없는 것을 의미하므로 $\boldsymbol{G_i}$는 영역 $\boldsymbol{H_{i+1}}$바깥에 존재한다. 따라서 $\boldsymbol{G_i}$ 내의 어떤 점을 선택하여 반지름 $m(\overline{CP_0})$인 원을 만들더라도 P_{i+1}는 그 원으로부터 e보다 더 멀리 위치하게 된다. 여기서, $\boldsymbol{G_i}\neq\boldsymbol{\varnothing}$이 아니고, $\boldsymbol{G_{i+1}}=\boldsymbol{\varnothing}$일 때, 점 P_{i+1}을 breakpoint라 하면, breakpoint P_{i+1}를 만나기 전까지 ***ARC*** $(Z,0,i)$는 $\boldsymbol{G_i}$ 내의 한 점 C를 중심으로, 반지름 $m(\overline{CP_0})$를 가지는 원으로부터 허용오차 e 이내에 있는 점들이다.

즉, G_i는 디지털 곡선 Z 내에 있는 점들 P_0, P_1, ..., P_i와 거리오차 e 이내에 있는 원의 중심 후보들의 집합을 의미하며, G_i 내에 한점 C(Gi의 무게 중심)를 정하면, 반지름 $r=m(\overline{CP_0})$ 원의 추출이 이루어진다.

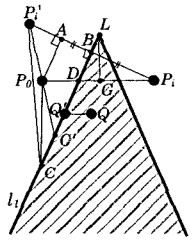
2.2 H영역 경계선의 선형화 및 최대허용오차

앞 절에서 G_i 영역은 식 (2)를 만족하는 원의 중심들이 존재하는 영역이다. 정확한 영역으로 좀 더 적절한 중심을 구할 수 있지만, 쌍곡선의 공통영역 G_i를 정확하게 구하고 쌍곡선의 일부인 영역의 경계선을 표현하는데 많은 계산과 정보가 필요하다. 그래서 식 (2)를 만족하는 공통 영역을 구하기 위한 계산량과 영역 표현을 위한 정보량을 줄이기 위하여 H_i영역의 경계선대신에 H_i영역을 포함하는 선형화된 경계선을 제안한다.

그림 2(a)의 빗금친 영역은 식 (2)를 만족하는 쌍곡선의 두 점근선을 선분 $\overline{P_0P_i}$에 평행한 방향으로 $\frac{e}{2}$만큼 좌측 및 우측으로 각각 이동하여 그 사이에 만들어지는 영역이다. 여기서 이 영역내의 점들의 집합을 X_i라 정의하면, X_i은 H_i을 포함하는 선형화된 영역을 가진다.



(a) 단순화된 원의 중심가능 영역 X_i



(b) 최대거리오차를 가지는 중심위치 C'

그림 2 X_i 영역 및 최대거리오차

정리 1) X_i 영역 내 임의의 점 C에 대해

$$err(P_0,P_i,C) < e_X \tag{3}$$

를 항상 만족한다. 여기서 $e_X = \sqrt{2}e$이다.

증명) 그림 2(a)는 좌우상하 대칭이므로 증명을 단순화하기 위해서 P_0, P_i의 중심 G을 기준으로 좌측하단 부분(3/4분면에 있는 공통영역)만을 고려하기로 한다. 그림 2(b)에서 X_i영역 내 임의의 점 Q가 있고, l_1 상에 선분 $\overline{P_0P_i'}$와 평행한 선분 $\overline{QQ'}$ 가 되도록 점 Q'를 정하면, $\overline{Q'P_0} < \overline{QP_0}$ 이고, $\overline{Q'P_i} > \overline{QP_i}$ 이므로,

$$|\overline{Q'P_i}-\overline{Q'P_0}| > |\overline{QP_i}-\overline{QP_0}|$$

이다. 즉, 영역내부에 있는 점은 P_0와 P_i로부터 거리오차가 항상 경계선 l_1 에 있는 어떤 한 점보다 오차가 작으므로 주어진 영역 내에서 오차가 가장 큰 점은 반드시 l_1상에 존재한다. 한편, 그림 2(b)에서 l_1위에 임의의 점 C를 정하고, 그림 2(b)에서 직선 l_1에 대해서 점 P_i을 대칭 이동시켜 그 점을 P_i'라 하면 $m(\overline{CP_i})=m(\overline{CP_i'})$이다. 따라서 삼각형 $\triangle CP_i'P_0$에서, $err(P_0,P_i,C)$의 최대값은 C가 $P_i'P_0$의 연장선상의 점 C'에 위치할 때 $m(\overline{P_0P_i'})$가 되므로

$$|m(\overline{CP_i'})-m(\overline{CP_0})| \leq m(\overline{P_0P_i'}) \tag{4}$$

이다. 한편, 그림 2(a)에서 쌍곡선에 접하고, 점근선을 대각선으로 하는 사각형은 가로 e이고, 대각선길이 $m(\overline{P_0P_i})$이다. 따라서 그림 2(b)에서 $\triangle AP_iP_0$는 $\triangle GLD$와 닮은꼴이므로 $m(\overline{AP_0})=e$이다. 또한, $\triangle BP_iD$도 $\triangle GLD$와 닮은꼴이므로, $t=m(\overline{AP_i})/m(\overline{P_0P_i})$라 두면,

$$m(\overline{P_iP_i'}) = 2m(\overline{BP_i}) = 2tm(\overline{DP_i}) = t(e+m(\overline{P_0P_i}))$$
$$= te+m(\overline{AP_i})$$

이므로

$$m(\overline{AP_i'}) = m(\overline{P_0P_i'})-m(\overline{AP_i}) = te \tag{5}$$

이다. 따라서 식 (4)에서

$$err(P_0,P_i,C) = |m(\overline{CP_i'})-m(\overline{CP_0})| \leq m(\overline{P_0P_i'})$$
$$= \sqrt{m(\overline{AP_0})^2+m(\overline{AP_i'})^2}$$

이고, $m(\overline{AP_0})=e$, $0\leq t<1$이므로, 식 (5)를 이 식에 적용하면

$$err(P_0,P_i,C) \leq e\sqrt{1+t^2} < \sqrt{2}e$$

이다. □

따라서, X 영역들의 공통집합을

$$A_i = X_0 \cap X_1 \cap \ldots \cap X_i \tag{6}$$

라 두면, A_i는 디지털곡선 Z내에 있는 점들 P_0, P_1, ..., P_i와 거리오차 $\sqrt{2}e$이내에 있는 원의 중심 후보들의 집합을 의미한다.

2.3 호 추출 알고리즘

호의 추출은 앞서 언급한 중심후보 영역 A를 구하면 이 영역으로부터 허용오차(정리 1참조)이내에 있는 호의 중심(A영역 경계선 꼭지점들의 위치 평균점)과 반지름

을 구할 수 있다. 이 절은 $\boldsymbol{A}$영역을 구하기 위한 방법과 조건을 기술하고 이를 이용한 호 추출 알고리즘을 설명한다.

2.3.1 A영역의 분할과 중심후보영역의 선택

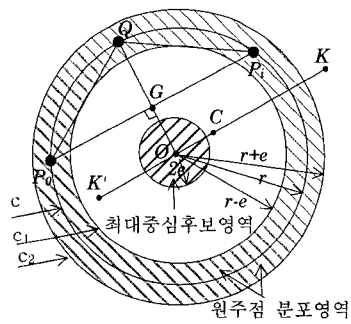
식 (6)에서 $\boldsymbol{A}$영역은 X영역들의 공통영역이며, $\boldsymbol{X}$영역은 그림 2(a)에서처럼 두 개의 cone이 합쳐진 형태인데 이 논문에서는 두개의 cone 중에 하나를 중심후보영역에서 제외시킴으로써 약간의 중심후보를 희생하는 대신 계산 속도를 향상시키는 방법을 제안한다.

그림 2(a)에서 점 U를 정점으로 하고 u_1과 u_2를 경계선으로 하는 영역을 $\boldsymbol{U}_i$로, 점 L을 정점으로 l_1과 l_2를 경계선으로 하는 영역 $\boldsymbol{L}_i$ 로 나눈다면

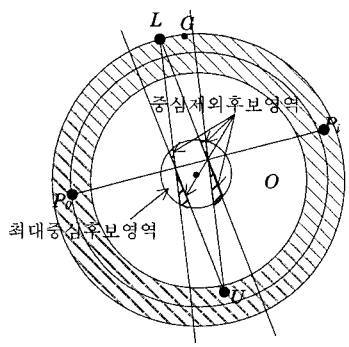
$$\boldsymbol{X}_i = \boldsymbol{U}_i \cup \boldsymbol{L}_i$$

로 둘 수 있다.

한편, 그림 3에서 점 O를 구하고자하는 원 c의 실제 중심이라 하고 디지털 호 $\boldsymbol{ARC}(Z,0,i)$가 원 c로부터 오차 e이내에 존재한다고 가정하면, 호가 분포하는 범위는 그림 3의 원 c_1과 c_2사이(바깥의 빗금친 부분)에 반드시 존재하며, 그림 3의 최대 중심후보 영역을 벗어나는 어떤 점도 식 (2)를 만족하는 중심후보가 될 수 없다. 예를 들면, 그림 3의 점 C가 최대 중심후보 영역 외부에 있는 점으로 중심의 후보라 가정하면, 반지름 $m(\overline{CK})$= $m(\overline{CK'})$인 원을 그릴 때, K가 원 c_2로부터 e이내에 있기 위해서 $m(\overline{OK}) < r+2e$라야 하며, 이때 $m(\overline{CK}) < m(\overline{OK})-2e < r$이다. 그리고 $m(\overline{OK'}) < m(\overline{CK'})-2e < r-2e$가 되며, 이 때 K'는 원 c_1으로부터 e이상 떨어지게 되므로 C는 허용오차 e를 만족하는 원의 중심후보가 될 수 없다. 따라서 현 $\overline{P_0P_i}$의 중심위치가 O에서 $2e$보다 멀리 있는 경우, $\boldsymbol{L}_i$와 $\boldsymbol{U}_i$의 공통영역을 제외한 $\boldsymbol{L}_i$나 $\boldsymbol{U}_i$ 중 하나는 중심후보영역을 벗어나므로(그림 2참조) 후보영역에서 제외해도 무방하다.



(a) 최대 중심후보 영역



(b) 중심제외후보 영역

그림 3 최대 중심후보 영역과 중심제외후보 영역

한편, $\overline{P_0P_i}$가 최대 중심후보 영역을 통과하는 경우 $\boldsymbol{L}_i$나 $\boldsymbol{U}_i$ 중 하나를 택하면 나머지 영역의 일부가 중심후보에서 제외될 수 있는데 최대 중심후보 영역 내에서 제외되는 영역($\boldsymbol{L}_i$와 $\boldsymbol{U}_i$의 공통영역을 제외한 $\boldsymbol{X}_i$영역으로 그림 3(b)의 중심제외후보 영역)은 높이가 $4e$(최대중심후보 영역 지름), 폭이 $4e^2/h$ (여기서 $h=\sqrt{m(\overline{P_0P_i})^2/4-e^2}$) 인 삼각형보다 작은 영역이면서, $\boldsymbol{X}$영역 경계선근처 영역이고 $\boldsymbol{X}$영역 경계선은 최대허용오차를 가지는 영역이다 (그림 3(b) 중심제외후보 영역). 그리고 $e \ll r$이라고 가정하면 $h \cong r$이므로 중심제외후보 영역은 최대 중심후보영역 폭 $4e$보다 훨씬 작고, 중심후보들이 한계(허용)오차 근처의 값을 가지므로 본 방법에서는 이를 무시한다.

따라서 그림 3(a)와 같이 중심이 $\overline{P_0P_i}$보다 아래($\overline{P_0P_i}$ 방향을 x축에 일치시킬 때 x축 기준)에 있으면 식 (6)에서 $\boldsymbol{X}_i$대신 $\boldsymbol{L}_i$를 선택하고, 반대로 위에 있으면 $\boldsymbol{X}_i$대신 $\boldsymbol{U}_i$를 선택하여 $\boldsymbol{A}_i$를 갱신하는데 이때 어느 쪽을 선택하느냐에 따라서 중심제외후보 영역 중의 일부가 중심영역에서 제외된다. 예를 들면 그림 3(b)에서 $\boldsymbol{L}_i$가 선택되면 중심제외후보 영역 중의 $\overline{P_0P_i}$보다 윗부분에 있는 영역은 중심에서 제외된다.

2.3.2 중심의 상대적 위치

$\boldsymbol{X}_i$ 대신 $\boldsymbol{L}_i$ 또는 $\boldsymbol{U}_i$를 선택하기 위해서 $\overline{P_0P_i}$에 대해 O의 상대적인 위치가 필요하다. 오차가 없는 디지털 호 $\boldsymbol{ARC}(Z,P0,i)$내의 임의의 점 Q에 대해서 만약 $\overline{P_0P_i}$가 중심 O를 지난다면, 즉 호의 각도가 π이면 벡터의 내적

$$\overrightarrow{P_0Q} \cdot \overrightarrow{QP_i} = 0 \qquad (7)$$

이다.

그러나, 실제 $\boldsymbol{ARC}(Z,P0,i)$ 내의 점들은 대략 허용오차 e이내의 오차를 가지므로 식 (7)을 만족하더라도 $\overline{P_0P_i}$가 O로부터 거리오차를 가질 수 있다. 만약 Q가 P_0와 P_i로부터 같은 거리에 있고, $e \ll r$이라고 가정하면, P_0나 P_i가 원 c_1위에 있고 Q가 c_2에 있거나, P_0나 P_i가 원 c_2위에 있고 Q가 c_1에 있을 때 Q가 O로부터

최대 약 $2e$정도 떨어지므로 Q는 최대중심후보영역이내 또는 매우 가까운 거리에 있다고 볼 수 있다. 이 경우 앞서 언급한 바와 같이 U_i 나 L_i중 어떤 영역을 중심후보 영역으로 선택해도 무방하다.

따라서, 공통영역을 구하기 위해 식 (6)대신에 다음의 식들을 적용한다. 만약 i=1,2,..인 모든 i에 대해서 $\overrightarrow{P_0Q} \cdot \overrightarrow{QP_i} \geq 0$ 이고, $i+1$에 대해서 $\overrightarrow{P_0Q} \cdot \overrightarrow{QP_{i+1}} < 0$ 를 만족하면

$$A_{Lk} = L_0 \cap L_1 \cap \ldots \cap L_i \cap U_{i+1} \cap \ldots \cap U_k \quad (8)$$

이고(여기서 P_k는 breakpoint), 그리고 .만약 i=1,2,..인 모든 i에 대해서 $\overrightarrow{P_0Q} \cdot \overrightarrow{QP_i} < 0$이고, $i+1$에 대해서 $\overrightarrow{P_0Q} \cdot \overrightarrow{QP_{i+1}} \geq 0$를 만족하면

$$A_{Uk} = U_0 \cap U_1 \cap \ldots \cap U_i \cap L_{i+1} \cap \ldots \cap L_k \quad (9)$$

를 적용하여 중심후보영역을 구한다. 즉, 호보다 중심이 아래에 있으면 식 (8)을 적용하고, 위에 있으면 식 (9)를 적용한다.

한편, 매우 작은 호에 대해서도 현 $\overline{P_0P_i}$와 점 Q의 최대 가능거리가 $2e$이다. 예를 들면, 그림 3(a)에서 P_0와 P_i가 원 c_1위에 있고 Q가 c_2에 있는 경우 현 $\overline{P_0P_i}$와 점 Q의 거리가 최소 $2e$이므로 지름보다 훨씬 짧은 현 $\overline{P_0P_i}$에 대해 식 (7)을 적용해서는 안 된다. 따라서

$$m(\overline{P_0P_i}) > 4e \quad (10)$$

를 만족할 때 식 (7)을 적용한다.

식 (10)을 만족하지 못하는 현의 길이가 매우 짧은 경우 호 중심의 상대적 위치를 알 수 없으므로 이때는 L_i와 U_i를 A_i 대신에 각각 2개의 중심후보영역 A_{Li} 또는 A_{Ui}와 연산하여 저장한다.

그리고, 식 (7)을 만족하는 $m(\overline{P_0P_i})$가 $2(r-e)$ 미만이 될 수 있으며 식 (7)의 전제조건인 식 (10)을 만족하기 위해서는

$$r > 3e \quad (11)$$

를 만족해야하며 이 식은 반지름 r에 대한 허용오차 e의 한계를 결정하는데 사용된다.

한편 식 (7)의 Q는 P_0나 P_i로부터 가까울수록 오차에 민감해지므로 가능한 두 점의 가운데 있는 점을 선택하는 것이 유리하다. 그런데 2차원 평면상의 디지털 곡선 경우에 원주점들이 화소간격으로 연속적으로 주어지고, $\overrightarrow{P_0Q} \cdot \overrightarrow{QP_i}$=0을 만족하는 $ARC(Z,0,i)$는 반원 크기에 해당하는 디지털호로서 수직 및 수평 방향과 사선방향의 단위 선분이 골고루 분포한다고 보아도 무방하므로 $ARC(Z,0,i)$의 점들 중 가운데 점, 즉 i가 짝수인 경우 $i/2$번째 점, 홀수인 경우 $(i-1)/2$번째 점(이후 $P_{i/2}$로 표시)을 Q 대신 사용한다.

2.3.3 호추출 알고리즘

다음은 식 (7)~(10) 등을 이용한 2차원 평면상에 있는 디지털 곡선으로부터 하나의 호를 추출하는 알고리즘이다.

알고리즘1 int An_Arc_Detection(int k)

```
{
//Step 1 : 초기화
(1-1)   A_L0=A_U0={실수평면 전 영역};
(1-2)   breakpoint=false;
(1-3)   P0 = Pk
(1-4)   i=k+1;

//Step 2 : 중심후보영역 구하기
        while(!breakpoint && (i<=n)) ; //n은 경계화소 총수
        {
(2-1)     Q=P_{i/2} ; // P0와 Pi의 중간점 구하기
(2-2)     L_{i+1} 및 U_{i+1}을 구한다;
(2-3)     if (A_Li ≠∅) { //호보다 중심이 아래에 있는 경우
            if((m(P0P_{i+1})>4e) && (P0Q · QP_{i+1} ≥0)
                || (m(P0P_{i+1})≤4e)) A_{Li+1}=A_Li ∩ L_{i+1} ;
            else A_{Li+1} = A_Li ∩ U_{i+1} ;
          }
(2-4)     if (A_Ui ≠∅) { //호보다 중심이 위에 있는 경우
            if((m(P0P_{i+1})>4e) && (P0Q · QPi<0)
                || (m(P0P_{i+1})≤4e)) A_{Ui+1}=A_Ui ∩ U_{i+1};
            else A_{Ui+1} = A_Ui ∩ L_{i+1} ;
          }
(2-5)     if((A_{Li+1} ==∅) && (A_{Ui+1} == ∅))
            breakpoint=true;
(2-6)     if(!breakpoint) i++;
        } // end of while

 //Step 3 : 중심 및 반지름 구하기
(3-1)   if(A_Li ≠∅) { C=Center of A_Li;
        r=m(CP0); }
(3-2)   else if (A_Ui ≠∅) { C=Center of A_Ui;
        r=m(CP0); }
(3-3)   return(i);
    } //end of An_Arc_Detection()
```

호와 중심의 관계는 반드시 호를 중심으로 위 알고리즘의 A_{Li+1}와 A_{Ui+1} 중 하나는 먼저 중심에서 벗어나는 (Ø) 영역이 된다. 만약 두 영역 모두 Ø가 되면, P_{i+1}은 식 (3)의 허용오차 e_X를 벗어나는 breakpoint가 되며, 이때 A_{Li} 또는 A_{Ui} 중 Ø가 아닌 영역에서 하나의 점을 선택하여 중심을 구하게 된다(2.4절 참조). 그러나 경우에 따라 A_{Li} 와 A_{Ui} 영역 모두 Ø가 아닐 수 있으며, 이때 호 $ARC(Z,0,i)$은 직선의 일부이거나 아니면 현의 길이가 매우 짧은($4e$ 이하인) 호에 해당한다.

그리고 Step (3-1) 및 (3-2)의 Center는 A_{Li} (또는 A_{Ui})영역 꼭지점의 위치중심을 의미한다.

아래 그림 4는 위 알고리즘을 적용하여 $A_i(A_{Li+1})$영역을 찾고 그 중심을 추출한 예를 보여준다.

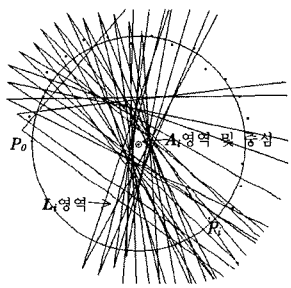


그림 4 A 영역의 예

한편 폐곡선 경계선내에 여러 개의 호가 존재할 경우 호를 추출하는 알고리즘은 다음과 같다.

알고리즘2 Arcs_Detection()

```
{
   int i=0, n;
   bool first=true;
   경계선 Z 읽어오기; //Z(2장 참조)는 디지털 곡선
   while (i<n){
      i=An_Arc_detection(i);
      if (first && (i<n)) {
          P0부터 Pi를 Pn뒤에 연결;
         n=n+i+1;
         first=false;
      }
      else 중심과 반지름 출력
   }
} //end of Arcs_Detection
```

폐곡선의 경우 시작점과 끝점이 연결되어 있으므로 시작점의 위치에 따라 같은 호가 그 점을 전후로 2개의 호로 분리될 수 있다. 본 방법에서는 이를 방지하기위해 첫 번째로 추출된 호는 무시하고 두 번째부터 호로 간주한다. 그리고 앞에서 무시된 호는 맨 마지막 점 P_n에 연결하여 다시 처리된다. 그러나 첫 번째 추출된 호의 끝점이 Z의 끝점과 일치($i=n$)하는 경우는 전체가 하나의 호이므로 더 이상 처리할 필요가 없다.

3. 구현 및 실험

3.1 측정값에 대한 고찰

위의 주요 측정값들을 얻기 위해서 반지름 크기 100이고 중심의 위치를 (0,0)로 하는 목표 원 c에 대해, 잡음(random noise)의 최대 변화를 ±N/2으로 하여 n개의 원주점이 일정한 각도로 분포되어 있는 100개의 디지털 곡선 Z에 대해서 실험하였다.

실험에 측정된 주요 변수는

$|\Delta C|_{max}$: 원c의 중심과 추출된 원들의 중심간 거리 최대오차

$|\Delta C|_{avg}$: 원c의 중심과 추출된 원들의 중심간 거리 오차평균

r_{max} : 추출된 원들 중 최대 반지름

r_{min} : 추출된 원들 중 최소 반지름

$|\Delta r|_{avg}$: 원c의 반지름에 대한 추출된 원들의 반지름 오차평균

d_{max} : 추출된 원과 Z내의 원주점들 간의 최대거리

이며, 여기서 e는 식 (2)을 만족하는 허용오차이며, 실제 최대허용오차는 식 (3)을 만족하는 c허용오차 e_X로서 e의 $\sqrt{2}$배가 된다(그림 5참조).

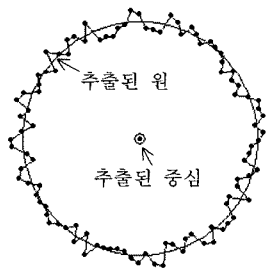


그림 5 잡음이 있는 곡선에 대해 본 방법을 적용한 예 (n=100, N=10, e=20)

다음의 표 1은 n과 N을 변화시키면서 각각 100개의 디지털 원 Z에 대해서 실험한 결과이다. 여기서 허용오차 e는 최소한 잡음의 크기 N보다는 커야 하며, 표 1에서는 N의 2배를 e로 실험하였다. 표 1에서 d_{max}는 항상 허용오차 $e_X(=\sqrt{2}e)$이내에 있고, 중심은 최대중심후보 영역(그림 3 참조)내에 있는 것을 확인할 수 있다.

표 2는 디지털원에 대해서 잡음의 크기를 변화시키지 않고 허용오차 e를 증가시킨 후 주요 변수의 변화를 측

표 1 원주점의 수와 잡음 크기에 따른 주요변수 변화

n	N	e	$\|\Delta C\|_{max}$	$\|\Delta C\|_{avg}$	r_{max}	r_{min}	$\|\Delta r\|_{avg}$	d_{max}
3	1	2	0.59	0.30	100.43	99.62	0.14	0.02
3	5	10	2.82	1.49	101.94	97.86	0.72	0.47
3	10	20	5.83	3.12	103.33	95.07	1.66	1.96
3	20	30	12.13	6.44	106.56	89.75	3.44	4.70
10	1	2	1.01	0.39	100.41	99.57	0.15	1.24
10	5	10	4.42	1.74	101.80	97.87	0.81	6.03
10	10	20	9.66	3.49	103.37	95.01	1.74	11.70
10	20	30	16.48	6.26	106.76	90.95	3.20	21.77
100	1	2	1.13	0.41	100.27	99.67	0.11	1.49
100	5	10	3.91	1.42	101.79	97.82	0.70	6.62
100	10	20	7.49	2.95	103.47	95.01	1.43	12.23
100	20	30	12.94	4.83	105.98	91.19	2.82	23.89

표 2 허용오차에 크기에 따른 주요변수 변화

n	N	e	$\|\Delta C\|_{max}$	$\|\Delta C\|_{avg}$	R_{max}	R_{min}	$\|\Delta R\|_{avg}$	d_{max}
100	2	2	1.00	0.41	100.75	99.41	0.24	2.14
100	2	5	2.16	0.75	100.78	99.28	0.26	3.37
100	2	10	3.45	1.25	100.86	98.29	0.40	4.69
100	2	20	6.81	2.57	102.29	98.48	0.79	8.26
100	2	30	9.59	3.81	105.08	98.22	2.10	12.38

표 3 호의 각도 크기에 따른 주요 변수의 변화

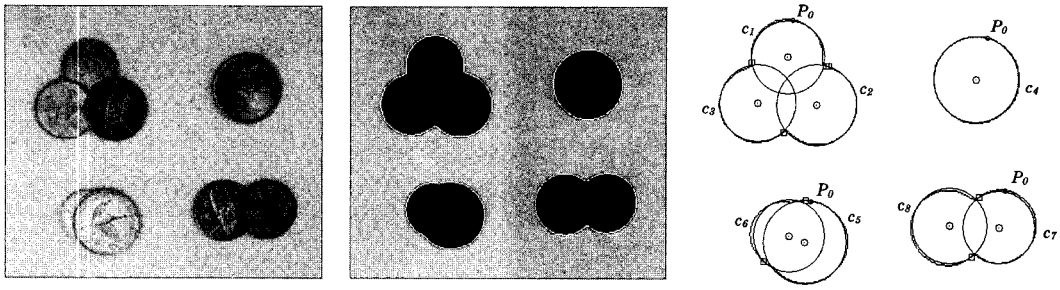
n	θ	$N(e)$	$\|\Delta C\|_{max}$	$\|\Delta C\|_{avg}$	R_{max}	R_{min}	$\|\Delta R\|_{avg}$	d_{max}
17	$\pi/3$	2(4)	35.38	11.19	133.57	92.05	10.27	2.88
25	$\pi/2$	2(4)	13.53	2.84	111.34	95.30	2.38	3.02
33	$2\pi/3$	2(4)	4.11	1.50	103.41	97.46	0.98	2.62
50	π	2(4)	2.68	0.74	101.04	99.10	0.35	2.78
75	$3\pi/2$	2(4)	1.46	0.60	100.68	99.15	0.28	2.50
100	2π	2(4)	1.88	0.65	100.73	99.15	0.27	2.67

정한 결과로, e의 증가에 따라 $\boldsymbol{X}$영역이 가지는 자체 오차가 커짐에 따라 오차가 증가하는 것을 보여준다. 보다 정확한 변수 값을 구하기 위해서 허용오차의 크기를 가능한 작게 하는 것이 필요하지만, 중심 위치 평균오차($|\Delta C|_{avg}$)와 반지름오차(R_{max}, R_{min} 및 $|\Delta R|_{avg}$)는 중심위치 최대오차($|\Delta C|_{max}$) 및 d_{max}와 달리 e의 크기에 비례하지 않으며, 한계 오차에 가까운 e=30(식 (11) 참조)일 때를 제외하고 N내외에 안정된 값을 보인다.

한편, 호의 각도는 주요 변수들의 오차와 많은 관계가 있다. 추출해야할 호의 각도가 작을수록 잡음이 곡률에 미치는 영향이 커진다. 표 3은 카메라영상이나, 스캐너로 이미지를 입력할 때 보통 1~2 화소 크기의 양자화 오차가 발생하는데(그림 6참조) 표 3은 2화소 이내의 잡음을 가지는 호에 대해서 허용오차(e=4)를 적용하여 각도와 중심 구한 결과이다. 표 3에서 본 방법에 의한 호추출은 $\theta \geq \pi/2$이면 원주점과의 추출된 호와의 거리오차와 반지름의 평균오차($|\Delta R|_{avg}$) 및 중심의 위치 평균오차($|\Delta C|_{avg}$) 모두 $e_X(4\sqrt{2})$이내를 만족하는 것을 보여준다.

그림 6은 디지털 카메라로 4종류의 동전을 서로 겹치거나, 떨어지게 이미지를 생성하고(그림 6(a), 해상도 180dpi, 이미지 크기 336×276), 이 이미지를 바탕화면을 기준으로 단순하게 임계값을 설정한 다음 동전의 경계선을 추출한 그림(그림 6(b))이다.

그리고 경계선의 폐곡선 상에 최상위에 해당하는 점들을 시작점(그림 6(c)의 점 P_0들)으로 하여 본 방법을 이용한 호추출 결과(그림 6(c))를 보여주고 있다(그림의 작은 네모는 breakpoint이고 작은 원 가운데의 점은 추출된 중심을 나타낸다). 그림 6(c)에서 c_1~c_8까지 반지름은 각각, 29.7, 32.5, 31.1, 34.7, 29.3, 30.4, 33.7 및 29.1이었고(단위는 1화소 크기), 같은 종류의 동전 중 c_2와 c_8의 반지름의 차이가 2.1로 가장 컸다. 대체로 오차의 한계이내에서 값의 동전크기의 오차가 생겼는데, 이는 양자화 오차 뿐만 아니라, 그림자에 대한 영향(특히



(a) 동전사진 (b) 추출된 경계선 (c) 경계선으로부터 호의 추출

그림 6 동전사진으로부터 호의 추출

그림에서 c_2의 경우)을 제거하지 않은 상태이므로 오차가 좀 더 커졌다고 할 수 있으며 실물에 비해 2화소 이내 오차범위에서 반지름 및 중심의 위치가 추출됨을 확인하였다.

한편, 그림 7은 허용오차 $e=0.5$로 두고 정밀하게 호를 추출하는 예를 보여 준다. 그림 7은 4개의 호를 연결하고 연속점에서 1차 미분 값의 변화가 없는 폐곡선을 64개의 점으로 양자화하여 만든 그림이다. 호 c_1, c_2, c_3 및 c_4의 실제 중심위치는 각각 (-100, 0), (0, -100), (100, 0) 및 (0, 100)이고, 반지름은 c_1과 c_3는 40이고, c_2와 c_4는 181.4이며, 그리고 4개의 호는 모두 90도의 각도를 가진다. 실험결과 추출한 호의 중심과 반지름은 각각, (-99.5, 0.1), (0,5, -98.6), (99.9, 0.0) 및 (0.0, 100.1)로 반지름은 40.5, 180.1, 40.1 및 181.4로 추출되었다.

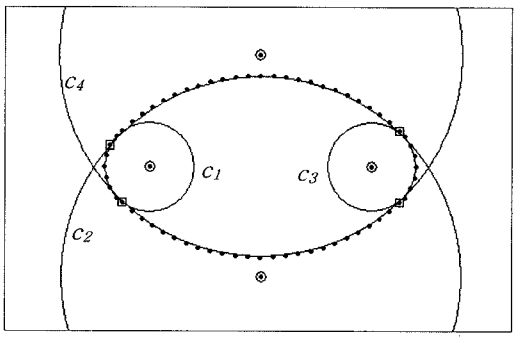


그림 7 부드럽게 연결된 폐곡선으로부터 호의 추출

3.2 처리속도

호 추출알고리즘 1의 처리속도는 Step (2-3) 또는 (2-4)에서 A_{Li+1}영역 또는 A_{Ui+1}영역을 구하기 위해 경계선의 교차점을 구하는 속도와 비례한다. 영역의 교차점은 최악의 경우 모두 $2i$개가 생길 수 있으며, 두개의 cone L_{i+1}와 U_{i+1}가 각각 A영역과 만나는 교차점을 구하기 위해서 최악의 경우 $4i$번의 경계선의 교차여부를 판정하기 위한 비교가 필요하므로 처리속도의 복잡도는 $O(n^2)$이 된다. 그러나 실제 적용에서 잡음 $N=2$이고, 원주점의 수(n)가 10개, 50개, 100개 및 500개인 호들을 각각 100개씩 만들어 수행한 결과(수행 CPU pentium IV, 1Ghz) 추출속도의 평균이 각각 0.47, 2.81, 6.25 및 44.84 msec 이었다. 이 결과는 원주점의 수 n에 대해서 속도가 n^2에 비례하지 않는 것을 나타내는데, 이것은 알고리즘 1에서 중심후보영역 A_i영역이 갱신될 때 영역경계선의 꼭지점 중 일부가 제거되고 제거된 꼭지점의 수만큼 연산회수가 줄어들기 때문이다. 따라서 실제 속도는 $O(n^2)$보다 작은 값을 가진다.

3.3 체인코드 또는 선분을 이용한 다른 방법들과의 비교

Rosin[12]과 Dosch[13]방법은 다각형 근사화 후 오차가 가장 큰 부분이 호를 나누는 기준이므로, 그림 8의 경우에 Rosin이 적용한 Landau방법[6]에 의한 원을 추출한 경우 그림 8의 호 c와 같고 이때 호로부터 가장 오차가 큰 디지털 곡선상의 점은 A또는 B근처의 점이 된다. 이때 그 오차의 크기가 임계값보다 작으면 그림 8의 원 c가 최종 추출결과가 되며 그렇지 않으면 점 A 또는 B를 기준으로 호가 나누어진다. 즉 나누어질 필요가 없는 호가 나누어지게 되므로 그림 7에서처럼 본방법과 같이 4개의 호를 추출하는 것은 불가능하다.

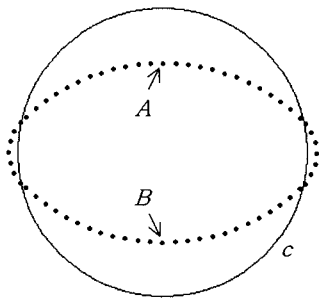


그림 8 Rosin의 방법을 적용했을 때 생기는 breakpoint (A와 B)들

한편 Dosch 방법도 Rosin 방법과 같은 원리이며, 다만 호의 중심을 구하기 위해 Dori 방법[4]을 이용하였고, 추출된 호의 거리오차를 다각형 근사화된 꼭지점과 비교하지 않고 원주점과 직접 비교 계산한 방법이므로 최대오차를 가지는 점의 위치 즉, 호가 분리되는 점의 위치만 약간 차이가 있을 뿐 결과는 거의 동일하다. 이 방법들은 호의 일부가 어떤 곡선과 연결되는가에 따라 breakpoint의 위치가 달라질 수 있으며 이것이 호의추출에 영향을 줄 수 있는데, 그것은 호에서 제외될 원주점들이 호의 후보로 포함되어 호를 결정하므로 이들 후보들로 인한 오류가 발생할 수 있다는 의미다. 그러나 본 방법도 정밀한 도형의 경우 보다 작은 허용오차의 설정이 필요하며 허용오차의 값이 커지면 breakpoint의 위치가 달라질 수 있다.

한편 경계선의 각도 변화를 이용한 Lim방법[14]의 경우는 곡률의 변화가 심한 그림 5의 경우는 여러 개의 작은 호로 나누어지기 쉽고, 그림 7의 경우에는 경계선 곡률의 변화가 매우 적어서 호를 분리하기 위한 곡률 임계값을 정하기 어려운 점이 있다. 잡음에 대한 곡률변화는 호로부터의 거리오차 변화보다 더 민감하므로 호의 오인식 가능성이 본 방법보다 더 높다고 할 수 있다. 특히 Lim 방법의 경우 곡률을 구하기 위해 scale factor를 미리 정해야 하고 구해진 곡률에 임계값을 적용하여 호를 분리하므로 호를 추출하기위해 scale factor와 곡률 임계값 등 두 가지에 대한 변수를 동시에 고려해야 하는 것이 단점이라 할 수 있다.

그리고, 위의 방법들이 호의 중심을 구하기 위해 사용한 Landau 방법[6] 또는 Dori[4]방법 등은 구해진 호와 주어진 곡선간의 허용오차를 보장하지 않는다.

4. 결 론

본 논문은 기하학적 방법으로 2차원 곡선인 경계선으로부터 호를 추출하는 방법을 제안하였다. 기존의 방법들이 경계선의 연결모양이나, 곡률변화에 민감하며, 추출된 호와 경계선간의 오차가 항상 일정값(허용오차)이내 있다는 보장이 없다. 반면에 제안된 방법은 경계선의 연결모양이나 곡률변화에 관계없이 2차원 곡선상의 점이 입력되는 즉시에 허용오차를 벗어나는 breakpoint 판정이 이루어진다. 따라서 breakpoint 뒤에 연결된 점들의 영향을 받지 않으며 추출된 호로부터의 주어진 호간의 거리가 항상 허용오차를 만족한다.

이에 대한 실험으로 경계선의 거리오차가 큰(곡률변화가 큰) 곡선, 여러 개의 원이 겹치는 실제 이미지로부터 추출한 경계선과 두 개 이상의 호가 부드럽게 연결된 곡선 등에 본 방법을 적용하여 유용성을 확인하였다.

본 방법은 2차원 곡선 내에서 물체를 인식하거나, 주어진 곡선과 추출된 호 간의 거리가 항상 허용오차 이내이므로 디지털 곡선의 호 근사화에 유용할 것으로 기대된다.

참 고 문 헌

[1] Pei, S. C. and Horng, J. H., "Optimum approximation of digital planar curves using circular arc," Pattern Recognition, Vol.29, No.3, pp.383-388, 1996.

[2] Yang, X., "Efficient circular arc interpolation based on active tolerance control," Commuter-Aided Design 34, pp.1037-1046, 2002.

[3] Qian, W. H. and Qian, K., "Optimising the four-arc approximation to ellipse," Computer Aided Geometric Design 18, pp.1-19. 2001.

[4] Dori, D. and Liu, W., "Stepwise recovery of arc segmentation in complex line environments," International Journal on Document Analysis and Recognition, pp.62-71, 1998.

[5] Song, J., Lyu, M. R. and Shijie C., "Effective multiresolution arc segmentation: algorithms and performance evaluation," Pattern Analysis and Machine Intelligence, IEEE Transaction on, Vol.26, No.11, 2004.

[6] Landau, U. M., "Estimation of a circular arc center and its radius," Computer Vision, Graphics, and Image processing 38, pp.317-326, 1987.

[7] Thomas, S. M. and Chan, Y. T., "A simple approach for the estimation of circular arc center and its radius," Computer Vision, Graphics, and Image processing 45, pp.362-370, 1989.

[8] Pei, S. C. and Horng, J. H., "Circular arc detection based on Hough transform," Pattern Recognition Letters 16, pp.615-625, 1995.

[9] Ho, C. H. and Chen, L. H., "A Fast ellipse/circle detector using geometric symmetry," Pattern Recognition, Vol.28, No.1, pp.117-124, 1995.

[10] Kim, H. S. and Kim, J. H., "A two-step circle detection algorithm from the intersecting chords," Pattern Recognition Letter 22, pp.627-636, 2001.

[11] Chju, S.H. and Liaw J.J., "An effective voting method for circle detection," Pattern Recognition Letter 26, pp.121-133, 2005.

[12] Rosin, P. L. and West, G. A. W., "Segmentation of edges into lines and arcs" Image and Vision Computing Vol.7, No.2, pp.109-114, 1989.

[13] Dosch, P., Masini, G. and Tombre, K., "Improving arc detection in graphics recognition," 15th International Conference on Pattern Recognition Vol.2, pp.2243-2246, 2000.

[14] Lim K. B., Xin K. and Hong G. S., "Detection and estimation of circular arc segments," Pattern Recognition Letters 16, pp.627-636, 1995.

류 승 필

1975년 3월~1979년 2월 서울대학교 전자공학과 학사. 1980년 7월~1993년 3월 한국원자력연구소 계측제어연구실 선임연구원. 1985년 3월~1987년 2월 충남대학교 전자공학과(석사). 1987년 3월~1991년 8월 충남대학교 전자공학과(박사). 1993년 3월~현재 세명대학교 컴퓨터학과 부교수. 관심분야는 패턴인식, 인공지능, 시뮬레이션