

단순한 도면으로부터 선분 확장을 이용한 아크 분할 기법 개발

이호근, 정성태
원광대학교 컴퓨터공학과

Development of an Arc Segmentation Technique based on Line Segment Expansion from Simple Drawing

Ho-Geun Lee, Sung-Tae Jung
Dept. of Computer Engineering, wonkwang University

요약

본 논문에서는 직선과 곡선으로 구성된 단순한 도면으로부터 곡선을 검출한 다음에 곡선을 원형 아크로 분할하는 방법을 제안한다. 본 논문의 방법에서는 먼저 선의 중심점을 찾은 다음에 연결된 중심점을 추적하여 선분을 검출한다. 그 다음에는 선분의 양 끝에서 선분의 방향을 이용하여 이웃한 선분을 검출하여 선분을 확장해 나간다. 선분을 확장한 다음에는 직선을 제거하고 곡선만 남긴 다음에 재귀적 분할 방법을 이용하여 곡선을 아크들의 집합으로 분할한다. 본 논문에서는 기존의 벡터화 소프트웨어와 벡터 기반 아크 분할 방법과 비교 실험을 수행하였다. 실험 결과에 의하면 본 논문에서 제안된 방법이 기존의 방법에 비하여 교차점을 가지는 곡선에 대하여 보다 정확하게 아크로 분할하였다.

1. 서론

컴퓨터 비전 기술의 발전에 따라 도면의 인식에 대한 연구가 활발하게 진행되고 있다. 도면은 일반적으로 복잡한 자연 영상에 비하여 단순한 형태의 직선과 곡선으로 구성된 영상으로 인공적인 의미를 내포하고 있다. 도면은 용도 및 특장에 따라 전자 회로도, 논리 회로도, 전기 회로도, 기계 설계도, 건축 설계도, 공정도, 부품도, 도로망도, 지적도 및 지형도 등 여러 가지 형식으로 광범위한 분야에서 이용되고 있다. 근래에 들어서는 복잡하고 다양한 도면의 효율적인 설계 및 유지, 보수를 위하여 CAD(Computer Aided Design) 시스템이 보편화되고 있는 추세이다. 그러나 CAD 시스템 사용 이전에 수작업으로 생성된 도면들은 모두 종이 형태로 존재하고 있다. 따라서 CAD 시스템의 효율적인 사용을 위해서 기존의 중요한 도면들을 CAD 시스템에 입력해야 하는데, 이를 수작업으로 하는 것은 많은 노력과 시간을 요하는 일이므로 도면 인식을 통하여 자동으로 CAD 시스템에 입력하는 방법이 요구되어 왔다. 기존 방법들은 곡선을 여러 개의 직선의 집합으로 근사화 하는 경우가 대부분이다. 일반적으로 곡선에 대한 수식은 매우 복잡하여 곡선 전체에 대한 수식을 구하는 것은 어려우므로 곡선을 원형 아크의 집합으로 표현하는 방법이 많이 사용되고 있다.

기존의 아크 분할 방법 중에서 초기의 방법들[1]은 허프(Hough) 변환을 사용하였다. x 축과 y 축으로 이루어진 2 차원 공간의 영상을 어떤 다른 매개 변수의 공간으로 변환하여 기하학적 도형을 표현하는데 쓰이는 방법이다. 즉, 2 차원 영상을 직선을 나타내는 매개 변수 공간으로 변환하여 영상에 존재하는 직선을 검출하거나, 원을 나타내는 매개 변수 공간으로 변환하여 영상에 존재하는 원을 검출하는 방법이다. 그러나 허프 변환에 의한 방법은 많은 계산량을 요구하는 문제를 가지고 있다.

참고 문헌 [2]에서는 세선화(thinning) 과정을 통하여 선들을 두께가 1인 픽셀 리스트로 변환한 다음

이를 아크의 집합으로 분할하는 방법을 제안하였다. 이 방법은 독립된 곡선의 픽셀 리스트의 경우에는 아크들의 집합으로 잘 분할하지만 선들이 서로 교차하여 구성된 복잡한 픽셀 리스트의 경우에는 세선화 과정의 문제로 인하여 아크 분할에 오류가 발생한다.

참고문헌 [2,3,4]에서는 벡터 기반 아크 분할 방법들이 제안되었다. 이 방법들에서는 먼저 참고 문헌 [5]과 같은 벡터화 과정을 통하여 곡선을 직선 선분의 집합으로 벡터화 한다. 직선 선분으로부터 아크를 추출하므로 수행속도가 빠르나 곡선을 직선 선분의 집합으로 근사화 시키므로 아크의 정확도가 떨어질 수 있다는 문제점을 가지고 있다.

본 논문에서는 정확한 아크 검출을 위하여 픽셀 정보로부터 직접 아크를 검출하는 방법을 제안한다. 픽셀 값으로부터 선의 중심점을 찾은 다음에 연결된 중심점을 추적하여 선분을 검출 후 선분의 양 끝에서 선분의 방향을 이용하여 이웃한 선분을 검출하여 선분을 확장해 나간다. 직선을 제거하고 곡선만 남긴 다음에 참고문헌 [2]의 재귀적 분할 방법을 이용하여 곡선을 아크들의 집합으로 분할한다. 본 논문에서는 기존의 벡터화 소프트웨어와 벡터 기반 아크 분할 방법과 비교 실험을 수행하였다. 실험 결과에 의하면 기존의 방법에 비하여 교차점을 가지는 곡선에 대하여 보다 정확하게 아크로 분할하였다.

2. 아크 분할

본 논문의 아크 분할 과정은 그림 1과 같다. 먼저 256 명암도를 갖는 흑백 영상으로 스캔된 도면을 읽어 들인다. 임계치보다 큰 값을 갖는 픽셀들의 값을 배경색으로 변환하고 고립된 픽셀들을 제거함으로써 잡음을 제거한 후, 선의 수직 두께와 수평 두께를 이용하여 선의 중심점을 검출한다. 본 논문에서는 선이 교차하는 지역에서 중심점을 검출하지 않으므로 검출된 선분들은 선이 교차하는 곳에서 서로 단절된다. 따라서 선분을 검출한 다음에는 선분의 끝점에서 탐침선을 이용하여 연결된 선분을 확장한다. 연결된

선분을 모두 찾은 다음에는 선이 굽은 정도를 측정하여 선분이 직선인지 곡선인지를 판별한다. 발견된 곡선에 대해서는 재귀적 방법으로 곡선을 두 개의 곡선으로 분할해가면서 일치되는 원형 아크를 찾아나간다.

그림 1. 제안된 방법의 순서도

2.1 잡음 제거

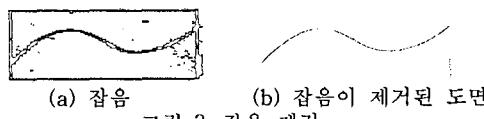
그림 2 (c)와 같이 스캔된 도면을 확대해서 각 픽셀의 값을 살펴보면 다양한 명암값이 존재하는 것을 알 수 있다.



(a) 원래 도면 (b) 스캔된 도면 (c) 확대된 선

그림 2. 스캔된 곡선 예

선 주변에서 배경색에 유사한 명암도를 갖는 픽셀들이 존재하게 된다. 이들은 잡음으로 간주하여 제거 해주어야 한다. 그림 3에는 임계치를 $230 (=256 \times 0.9)$ 이상의 값을 모두 배경색으로 처리한 결과가 나타나 있다. 임계치를 이용하여 잡음을 제거한 다음에는 픽셀 주위의 8개 픽셀이 모두 배경색인 고립된 픽셀을 찾아서 제거한다.



(a) 잡음

(b) 잡음이 제거된 도면

그림 3. 잡음 제거

기존의 많은 백터화 방법이나 아크 분할 방법에서는 256 명암도를 사용하지 않고 임계치를 이용하여 두 가지 명암도를 갖는 이진화된 도면을 사용하였는데, 본 논문에서는 선의 중심 위치를 사용하므로 보다 정확한 선의 중심을 찾기 위하여 256 명암도를 사용하였다. 그림 2 (c)에 나타나 있듯이 선의 중심에 가까울수록 명암 값이 더 작으므로 이를 이용하여 보다 정확한 선의 중심 위치를 구할 수 있는 것이다.

2.2 선의 중심 검출

정확한 선의 중심은 그림 4 (a)와 같이 선의 진행 방향에 대한 선의 양변으로부터 같은 거리에 위치한다. 그러나 이와 같은 정확한 중심을 찾는 것은 많은 계산량을 필요로 하므로 그림 4 (b)와 같이 선 내부에서 수평선과 수직선을 교차하도록 그었을 경우에 수평선과 수직선의 길이를 비교하여 길이가 짧은 수평 또는 수직선의 중심을 선의 중심으로 근사화하였다.



(a) 정확한 선의 중심

(b) 근사화된 선의 중심

그림 4. 선의 중심

그림 5 (a)와 같이 선 내부의 점에서 수직선과 수

평선을 그었을 경우에 각 선의 길이가 그 점에서의 수직 두께와 수평 두께가 된다. 올바른 선의 중심 검출을 위해서, MIN_{LW} 는 최소 임계값으로서 그림 5(b)와 같이 수평선이거나 수직선일 경우에 수직 두께나 수평 두께가 최소가 되므로 이때의 두께로 정의된다. MAX_{LW} 는 선이 45도로 기울었을 경우에 수평 두께나 수직 두께가 최대가 되므로 이때의 두께로 정의된다.



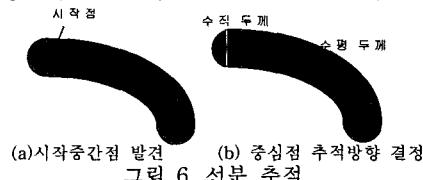
(a) 수직/수평 두께 (b) 임계값 (c) 선의 교차 (d) 선의 끝 영역

그림 5. 선의 두께

수평 두께와 수직 두께 모두 임계값 MAX_{LW} 보다 큰 영역은 선이 교차하는 영역으로 간주하여 선의 중심을 검출하지 않도록 하였다. 수평 두께와 수직 두께 모두 MAX_{LW} 보다 큰 영역의 예가 그림 5(c)에 나타나 있다. 선이 교차하는 영역에서는 교차하는 각각의 선을 분리해야만 선의 중심을 알 수 있으므로 현재 단계에서는 선의 중심을 찾을 수 없는 것이다. 또한 수평 두께나 수직 두께 중의 하나가 임계값 MIN_{LW} 보다 작은 경우에도 선의 중심을 검출하지 않았다. 이런 경우는 그림 5(d)와 같이 선의 끝 영역에서 발생한다.

2.3 선분 검출

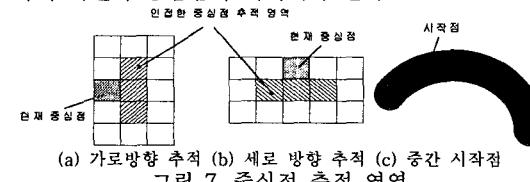
중심점 추적을 위해서는 먼저 시작 중간점을 발견한다. 그림 6(a)에 나타나 있는 곡선의 경우에는 좌측 상단의 중간점이 시작점으로 선택된다.



(a) 시작중간점 발견 (b) 중심점 추적방향 결정

그림 6. 선분 추적

시작점을 발견한 다음에는 그림 8(b)와 같이 수직 두께가 수평 두께보다 작은 경우에는 가로 방향으로 중심점을 추적한다. 추적 방향을 결정한 다음에는 해당 방향으로 인접한 중심점을 추적한다. 인접한 중심점이 위치해야 할 영역이 그림 7에 나타나 있다. 가로 방향의 선분을 추적하는 경우에는 그림 7(a)와 같이 현재 중심점의 바로 오른쪽 픽셀이나 그 위 또는 아래 픽셀에 중심점이 위치해야 한다.



(a) 가로방향 추적 (b) 세로방향 추적 (c) 중간 시작점

그림 7. 중심점 추적 영역

그림 7과 같이 설정된 인접한 중심점 추적 영역에 중심점이 없거나 두 개 이상인 경우에는 중심점 추적이 종료되고 현재까지 추적된 중심점들이 선분으로 결정된다. 또한 수직 두께와 수평 두께를 비교

하여 현재의 추적 방향에 모순되는 경우에는 추적을 종료한다. 이러한 경우는 선분의 방향이 변하는 위치에서 발생한다. 가로 방향 선분의 경우에는 그림 7(c)와 같이 선분의 중간에 있는 점이 시작점으로 선택될 수 있다. 이러한 경우에는 추적을 좌측과 우측 두 방향으로 하였다. 세로 방향 선분은 가장 위에 위치한 중간점이 시작점으로 선택되므로 항상 아래 방향으로만 추적을 하면 된다. 또한 최대 선두께의 2배 보다 작은 길이를 갖는 선분은 버리도록 하였다.

2.4 선분 확장

선분의 확장은 그림 8와 같이 선의 진행 각도 α 를 구한 다음에 그 각도를 중심각으로 하여 양방향으로 일정 각도 θ 만큼의 영역을 탐색 영역으로 설정한다.

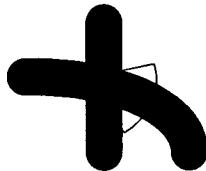


그림 8. 선분 확장을 위한 탐색 영역 설정

선의 끝에서부터 순서대로 세 개의 중심점을 $(x_1, y_1), (x_2, y_2), (x_3, y_3)$ 라 할 때 선의 진행 각도 α 는 식 (1)과 같이 구할 수 있다.

$$\alpha = \arctan\left(\frac{y_1 - y_3}{x_1 - x_3}\right) \quad \text{식(1)}$$

식(1)에서 $(y_1 - y_3)/(x_1 - x_3)$ 은 점 (x_2, y_2) 에서의 접선의 기울기를 근사화한 값이다. 점 (x_2, y_2) 에서의 정확한 접선의 기울기는 직선의 방정식이나 곡선의 방정식이 주어지는 경우에 (x_2, y_2) 에서의 미분값으로 구할 수 있다. 그러나 스캔된 도면에서 선에 대한 방정식이 주어지지 않으므로 미분을 근사화하는 방법을 사용한 것이다. 식 (1)에서 $\arctan(\frac{b}{a})$

함수는 $\tan()$ 함수의 역함수로서 원점에서 좌표 (a, b) 까지의 선과 x 축이 이루는 각도를 계산한다. 이 각도는 -180° 에서 180° 사이의 값을 가지는데, 양수 각도는 x 축과 시계 반대 방향을 이루는 각이며, 음수 각도는 시계 방향의 각이다.

식 (1)에서 $(y_1 - y_3)/(x_1 - x_3)$ 은 미분 값을 근사화하는 것으로서 어느 정도의 오차를 내포하게 된다. 또한, 선분 검출 단계에서 구한 중심점의 좌표는 정수 값을 갖는데, 식 (1)에서 정수 좌표를 사용하면 이로 인한 오차가 추가적으로 발생할 수 있다. 따라서 본 논문에서는 픽셀의 명암도를 이용하여 보다 정확한 중심점의 위치를 사용하였다.

181				
	228	100	24	
		179	48	18
			81	
	189	97	24	48
			154	
	105	18	8	108
	56	18	75	181
	16	130		
	154			

(a) 확대 장면
(b) 각 픽셀의 명암값

그림 15. 선의 일부를 확대한 모습

오차를 줄이기 위하여 먼저 선의 중심 좌표를 실

수값을 갖도록 다시 계산한다. 이와 같이 하는 이유는 선의 두께가 짙수인 경우에 정수 좌표로는 중심 위치를 정확하게 표현할 수 없기 때문이다. 그 다음에는 중심점을 기준으로 양쪽에 대하여 각각의 명암값 평균을 구한 다음에 양쪽의 값에 비례하도록 하여 중심점의 위치를 이동시켰다. 중심점의 양쪽의 픽셀 값의 평균값이 I_a 와 I_b 라 할 때 이동 크기는 식 (2)와 같이 계산하였다. 식 (2)에서 $1/2$ 을 곱한 이유는 최대 변동 폭을 0.5 이내로 제한하기 위해서이다.

$$k = \frac{|I_a - I_b|}{255} \times \frac{1}{2} \quad \text{식 (2)}$$

선의 진행 방향 α 를 구한 다음에는 검사 영역 내에서 가장 기울기가 완만하게 변화하는 선분을 선택하여 합병한다.



그림 9. 탐침선 선분 검사

그림 9에서 탐침선을 설정하기 위해서 식 (3)과 같이 탐침선의 끝점의 좌표를 구한다. 여기에서 (src_x, src_y) 는 선분의 끝 점 좌표이고 r 은 탐침선의 길이이며 $angle$ 은 현재의 탐침선의 각도이다.

$$dst_x = src_x + r \times \cos(angle) \quad \text{식 (3)}$$

$$dst_y = src_y + r \times \sin(angle)$$

식 (3)과 같이 탐침선의 끝 점 (dst_x, dst_y) 를 구한 다음에는 (src_x, src_y) 와 (dst_x, dst_y) 를 잇는 직선의 방정식을 구하고 중간점의 좌표들을 계산하면서 (src_x, src_y) 로부터 (dst_x, dst_y) 로 선분을 추적해간다. 연결된 선분이 여러 개인 경우에 그 중에서 현재 선분과 연속도가 가장 높은 선분과 합병하였다.

2.5 선분 분할

곡선의 분할을 수행하기 위해 검출된 선분 중에서 곡선 선분만을 남기고 직선 선분은 제거한 후, 각 선분에 대하여 재귀적 함수를 호출하여 이용하여 곡선 선분을 원형 아크의 집합으로 분할한다.

직선 선분과 곡선 선분을 구별하기 위해서 선분의 중심점에서의 기울기 변화와 함께 그림 10과 같이 선분의 양 끝점을 잇는 직선과 선분의 중심점 사이의 거리 정보를 이용하였다. 선분이 정확한 직선 선분이라면 선분의 각 중심점으로부터 직선으로의 거리의 합이 0이 된다.



그림 10. 점과 직선 거리(직선과 곡선 구별)

선분의 끝점이 (sx, sy) 와 (dx, dy) 이고 선분 상의 한 점이 (x, y) 라고 할 때에 (sx, sy) 와 (dx, dy) 를 잇는 직선과 점 (x, y) 와의 거리는 식 (4)와 같다. 검출된 선분의 모든 점에 대하여 식 (4)와 같이 거리를 구한 다음에 그 합이 작은 경우에는 직선으로 간

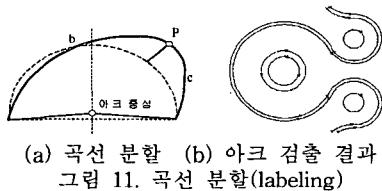
주하였다.

$$d = \frac{|a \times x + b \times y + c|}{\sqrt{a^2 + b^2}}, \quad \text{식 (4)}$$

$$a = \frac{dy - sy}{dx - sx}, \quad b = -1, \quad c = sy - a \times sx$$

곡선을 원형 아크들의 집합으로 분할은 참고 문헌 [2]의 알고리즘을 수정한 것이다. 그럼 11과 같이 세그먼트의 양 끝점을 지나는 원형 아크를 찾는다. 곡선의 양 끝점을 잇는 아크는 무수히 많이 존재할 수 있는데, 여기에서는 양 끝점을 잇는 직선의 중간점에서 법선을 그었을 경우에 그 법선 위에 아크의 중심이 존재하는 것으로 가정한다. 아크 중심이 법선 위에 있다고 가정하더라도 아크 중심의 위치에 따라 여러 개의 아크가 존재할 수 있는데, 그 중에서 곡선의 점들과의 오차가 가장 작게 되는 아크를 사용한다. 본 논문에서는 오차 계산의 정확성을 높이기 위하여 중심점 검출에 의해 구해진 중심점 대신에 중심점 검출 전의 원래의 픽셀 값을 사용하여 오차 계산을 수행하였다.

원형 아크와 중심점 사이의 거리의 최대값 max_deviations를 이용하여 아크로부터 가장 멀리 벗어나 있는 곡선 상의 점을 분할 기점 split_position로 설정한다. 그럼 11(a)에서는 점 P로부터 곡선의 양 끝점을 잇는 아크로의 거리가 최대가 된다. 원형 아크를 발견한 다음에는 세그먼트의 길이가 3픽셀이 하이거나 max_deviations 값이 3보다 작으면 더 이상 세그먼트를 분할하지 않았다. 그렇지 않은 경우에는 split_position 위치에서 세그먼트를 두개로 분할하여 두 세그먼트 각각에 대하여 수행하였다. 분할된 두 세그먼트의 오차 중에서 큰 값이 현재의 세그먼트의 오차보다 작으면 현재의 세그먼트를 두 개의 세그먼트로 분할하는 것이 좋다는 의미하므로 분할을 실시한다. 그림 11(b)은 아크 검출의 최종 결과이다.



(a) 곡선 분할 (b) 아크 검출 결과
그림 11. 곡선 분할(labeling)

3. 실험 결과

본 논문에서 제안된 곡선 분할 방법은 PC의 윈도우즈 운영체제 환경 하에서 Visual C++을 사용하여 구현되었다. 구현된 방법과 기존 방법의 성능을 비교하기 위하여 상업용으로 판매되고 있는 벡터화 소프트웨어 RasterVect[9]와 ProVec[10]을 사용하였다. 또한 Rosin[3]에 의해 개발된 아크분할 방법과 기존의 벡터화 기반 곡선 분할 프로그램인 MDUS[5]를 사용하였다. 그림 12에는 곡선에 교차하는 선들이 존재하는 보다 복잡한 도면에 대한 곡선 분할 결과가 나타나 있다. RasterVect의 경우에는 교차점을 통과하는 아크를 잘 검출하고 있으나 아크의 일부분에 대해서는 아크로 인식하지 못하고 직선으로 인식되었다. ProVec의 경우에는 교차점이 없는 아크는 잘 인식되었지만 교차점에서 아크가 단절되는 결과를 보여주고 있다. Rosin[3]의 방법에서는 세선화 과정을 통하여 선을 두께가 1인 선으로 변환한 다음에 아크를 추출하는데, 세선화 과정의 문제로 인하여 교차점에서 선이 단절되는 문제를 가지고 있음을 알 수 있

다. MDUS는 곡선을 먼저 직선의 집합으로 근사화한 다음에 직선들의 집합을 아크로 맵핑하는 방법을 사용한다. 이 방법에서는 도면을 직선의 집합으로 근사화할 때에 교차점을 잘 통과하는 방법을 고안하여 사용하고 있어서 교차점에 대한 처리를 잘 하고 있지만 직선으로 근사화할 때에 누적된 오차로 인하여 일부 아크들이 부정확하게 검출되었다. 이와 같이 교차점을 가지는 경우에는 기존의 방법들이 정확한 아크를 검출하는데 문제점을 보이고 있는 반면에 본 논문의 방법은 교차점을 가지는 도면에 대해서도 아크를 잘 검출하는 것으로 평가된다.



(a) 본 논문 (b)RasterVect (c)ProVec (d)Rosin (e)MDUS
그림 12. 교차점을 가지는 곡선에 대한 아크 분할

4. 결론

본 논문에서는 직선과 곡선으로 구성된 도면에서 곡선을 검출한 다음 곡선을 아크로 분할하는 방법을 제안하였다. 본 논문에서는 먼저 선분을 검출하고 선분의 양 끝점에서 선분의 방향에 따라 선분을 검출하고 확장하는 방법을 사용함으로써 교차점을 효과적으로 통과하는 방법을 제안하였다. 실험결과에 따르면 본 논문에서 제안된 방법은 기존의 방법에 비하여 교차점을 잘 처리할 수 있었으며 아크 분할이 보다 정확하였다. 그러나, 직선이나 곡선이 외에 복잡한 패턴을 포함하는 면이 존재하거나 곡선이 문자나 기호등과 아주 복잡한 형태로 교차한다든지 하는 경우에는 기존의 방법 뿐만 아니라 본 논문의 방법도 아크를 정확하게 검출하지 못할 수 있다. 향후 이러한 문제를 극복할 수 있도록 하기 위하여 통계적 방법이나 지능형 아크 분할 방법에 대한 연구가 필요한 것으로 사료된다. 또한 본 논문에서는 아크 검출 결과를 사람의 눈으로 판별하는 방법을 사용했는데, 아크 검출 결과를 보다 객관적으로 비교할 수 있는 방법에 대한 연구가 필요한 것으로 사료된다.

참고문헌

- [1]D.H. Hallard, "Generalizing the Hough Transform to Detect Arbitrary Shapes", Pattern Recognition, Vol. 13, No. 2, pp. 111-122, 1981
- [2]P.L. Rosin and G.A.W. West, "Segmentation of edges into arcs and lines", Image and Vision Computing, Vol. 7, pp. 109-114, 1989
- [3]P.L. Rosin and G.A.W. West, "Non-parametric Segmentation of Curves into Various Representations", IEEE Transactions on Pattern Analysis and machine Intelligence, Vol. 17, No. 12, Dec. 1995
- [4]D. Dori, "Vector-Based Arc Segmentation in the Machine Drawing Understanding System Environment", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 17, No. 11, pp. 1057-1068, Nov. 1995
- [5]W. Liu and D. Dori, "Sparse Pixel Tracking: A Fast Vectorization Algorithm Applied to Engineering Drawings", Proceeding of International Conference on Pattern Recognition, Vol. 3, pp. 808-812, 1996
- [6] <http://www.rastervect.com>
- [7] <http://www.skmconsulting.com/spatial/products/index.cfm>