DOI: 10.3745/KIPSTB.2008.15-B.6.501

부분 곡률을 이용한 개선된 스네이크 알고리즘

이 정 호[†] · 최 완 석[††] · 장 종 환[†††]

요 약

기존 스네이크 알고리즘은 에너지 함수의 정의에 의해 복잡한 객체의 윤곽을 추출하는데 어려움이 있고, GVF 방법은 에너지 맵 계산 시간이 많이 소요되는 문제점이 있다. 본 논문에서는 빠르고, 복잡한 객체의 윤곽을 잘 추출하는 방법을 제안한다. 객체 윤곽의 복잡도는 곡률로 정의하여 곡률 값이 임계치 이상이면 스네이크 포인트를 추가하여 객체의 윤곽을 추출하였다. 다수의 복잡한 영상에 실험을 통해 계산속도 및 윤곽 추출 성능을 개선하는 결과를 보여준다.

키워드 : 스네이크 알고리즘, 부분 곡률, 객체 윤곽 검출

An Improved Snake Algorithm Using Local Curvature

Jungho Lee[†] · Wansok Choi[††] · Jongwhan Jang[†††]

ABSTRACT

The classical snake algorithm has a problem in detecting the boundary of an object with deep concavities. While the GVF method can successfully detect boundary concavities, it consumes a lot of time computing the energy map. In this paper, we propose an algorithm to reduce the computation time and improve performance in detecting the boundary of an object with high concavity. We define the degree of complexity of object boundary as the local curvature. If the value of the local curvature is greater than a threshold value, new snake points are added. Simulation results on several different test images show that our method performs well in detecting object boundary and requires less computation time.

Keywords : Snake Algorithm, Local Curvature, Detection of Object Boundary

1. 서 론

최근 들어 IT기술의 급속한 발달과 멀티미디어 기술이 급격히 발달함에 따라 정보 및 멀티미디어 서비스는 현재 우리 생활 속에서 없어서는 안 될 정도의 영향력을 가지고 있으며 누구나 쉽게 사용하고 있다.[1] 이러한 서비스를 제공하기 위해 객체추출 연구를 집중적으로 하고 있는데 크게 영역 기반 기법과 윤곽선 기반 기법으로 나누어진다.[2,3]

윤곽선 기반 기법 중 대표적인 방법이 능동윤곽모델(Active Contour Model)인데 Kass에 의해 처음으로 제안되었으며 스네이크(Snake) 알고리즘이라고 불리며 정의된 에너지 함수에 의하여 최소화하는 과정을 통하여 객체의 윤곽을 추출한다.[3] 스네이크 알고리즘은 스네이크 점을 연결한 스네이크 등고선이 폐곡선을 이루어 객체를 분할하는 것이다. 이 방법은 Concavity가 없는 간단한 객체에서는 성능이 우수하지만 높은 Concavity가 있는 복잡한 객체는 성능이 저하되는 단점이 있다. Xu는 Optical Flow를 이용한 GVF (Gradient Vector Flow) 방법을 제안하였다. 이 방법은 복잡한 객체의 윤곽을 추출할 수 있지만 에너지 맵 계산시간이 많이 소요되는 문제점이 있다.[7,8] 본 논문에서는 이러한 문제점을 해결하기 위하여 부분 곡률의 특성을 이용하여 계산시간 및 성능을 개선하는 알고리즘을 한다.

본 논문의 구성은 2장에서는 기존 스네이크 알고리즘을 설명하고, 3장에서는 제안하는 새로운 알고리즘의 설명한다. 4장에서는 제안하는 알고리즘을 구현하여 기존 방법과 성능을 비교 분석하고, 5장에서는 결론을 기술한다.

2. 스네이크 알고리즘

스네이크 알고리즘은 일반적으로 능동윤곽모델(Active Contour Model)로서 Kass에 의해 처음 제안되었다.[3] 이 방법은 추출하고자 하는 객체의 주위에 스네이크 포인트를 설정하고 정의된 에너지 함수를 이용하여 스네이크 포인트를 반

† 준 회 원 : 배재대학교 정보통신공학과 석사과정
†† 준 회 원 : 배재대학교 정보통신공학과 석사
††† 종신회원 : 배재대학교 정보통신공학과 교수(교신저자)

논문접수 : 2008년 10월 7일
수 정 일 : 1차 2008년 11월 5일
심사완료 : 2008년 11월 6일

복적으로 이동시키면서 에너지 함수를 최소화시키는 과정을 통해 객체의 윤곽을 추출하는 방법이다. Williams는 각 스네이크 포인트마다 가중치를 두어 속도를 계산한 Greedy Snake를 제안하였으며 식 (1)과 같이 표현한다.[4]

$$E_{snake}(v)=\sum_{i=0}^{N-1}(E_{internal}(v_i)+E_{external}(v_i)) \quad (1)$$

에너지함수는 내부 에너지($E_{internal}$) 함수와 외부 에너지($E_{external}$) 함수의 합으로 표현된다. 스네이크 포인트 v_i는 $v_i=(x_i, y_i)$ 라고 정의한다. x_i 및 y_i는 각각 i번째의 x좌표 및 y좌표이다. 스네이크 포인트 수가 N개 이면 v_0부터 v_{N-1}까지 된다. 스네이크 포인트는 스네이크의 등고선을 만들면서 반복 수행하여 객체의 윤곽을 추출한다. 내부 에너지 함수는 스네이크 포인트들 간의 간격을 결정하는 Continuity 에너지 함수와 스네이크 포인트의 이동을 결정하는 Curvature 에너지 함수의 합이다.[4-8] 외부 에너지 함수는 영상의 강도나 명암 값으로 이루어진 영상의 기울기를 사용하며 스네이크 포인트를 관심객체의 윤곽에 놓이도록 한다.

3. 제안한 알고리즘

곡률은 곡선이 휨 정도를 나타내는 변화율이다. 복잡한 객체의 윤곽은 일반적으로 큰 곡률 값을 가지며, 복잡한 윤곽을 추출하기 위해서는 스네이크 포인트의 수가 단순한 윤곽을 추출할 때 보다 많이 필요하다. 제안한 알고리즘은 객체 윤곽의 복잡도를 부분 곡률을 사용하여 정의한다. 부분 곡률의 임계값을 설정하여 임계값 보다 크면 스네이크 포인트를 새로 삽입하여 복잡한 객체의 윤곽을 효율적으로 추출한다.

3.1 곡률의 임계값 및 스네이크 포인트 삽입

(그림 1)은 객체 윤곽의 곡률 값을 설명한다. (b)는 (a)보다는 높은 곡률 값을 갖는다. (a)의 경우는 객체 윤곽을 3개의 스네이크 포인트로 추출할 수 있지만, (b)의 경우는 3개의 스네이크 포인트로는 객체의 윤곽을 정확히 추출할 수 없고 새로운 스네이크 포인트(v_{n_1}, v_{n_2})를 삽입하여야 한다.

연속적으로는 곡률(k_a)는 2차 미분으로 식 (2)와 같이 정의한다.

$$k_a \approx \left|\frac{d^2y}{dx^2}\right| \quad (2)$$

이산적인 곡률(k_d)은 세 개의 스네이크 포인트 v_{i-1}, v_i, v_{i+1}을 이용하여 식 (6)로 정의한다.

$$\vec{A}=v_{i-1}-v_i, \quad \vec{B}=v_{i+1}-v_i \quad (3)$$

$$\vec{A}=a_1x+a_2y, \quad \vec{B}=b_1x+b_2y \quad (4)$$

$$\vec{A}\cdot\vec{B}=\|\vec{A}\|\,\|\vec{B}\|\cos\theta \quad (5)$$

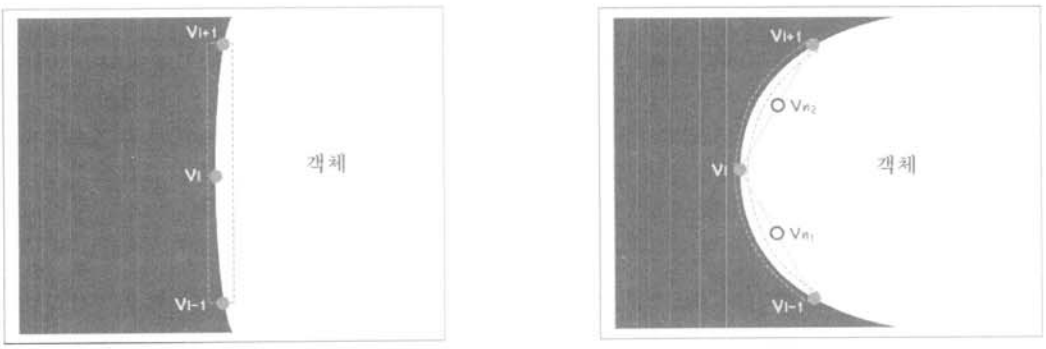
$$k_d \approx \cos\theta=\frac{\vec{A}\cdot\vec{B}}{\|\vec{A}\|\,\|\vec{B}\|}=\frac{a_1b_1+a_2b_2}{\sqrt{a_1^2+a_2^2}\sqrt{b_1^2+b_2^2}} \quad (6)$$

•는 두 벡터의 내적을 나타내며, ‖ ‖는 벡터 길이(Norm)를 나타낸다.

곡률 값이 크면 곡선의 휨 정도가 크고, 객체의 윤곽이 복잡하다는 것을 뜻한다. 객체의 윤곽을 정확히 추출하기 위해서는 곡률이 임계값보다 크면 새로운 스네이크 포인트(v_{n_1}, v_{n_2})를 삽입한다.

$$k_d > Th \quad (7)$$

새로운 스네이크 포인트는 (그림 1)(b)와 같이 $v_{n_1}=v_{i-1}+v_i/2$, $v_{n_2}=v_i+v_{i+1}/2$ 인 2개의 스네이크 포인트를 삽입한다. 삽입된 스네이크 포인트는 식 (7)을 만족할 때 까지 반복 수행한다.



(a) 낮은 곡률 (b) 높은 곡률

(그림 1) 객체 윤곽의 곡률 값 및 새로운 스네이크 포인트(v_{n_1}, v_{n_2}) 삽입

3.2 스네이크 포인트 이동

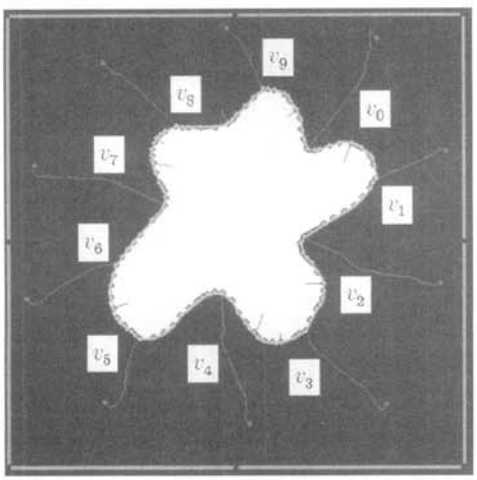
기존 스네이크 알고리즘은 스네이크 포인트를 객체 밖에서 안쪽으로 한쪽 방향으로 이동하는 문제점이 있다. 스네이크 포인트를 밖에서 안쪽으로만 이동하면 스네이크 포인트가 객체 윤곽을 통과하여 안쪽에 위치하면 객체 윤곽을 추출할 수 없다. 이러한 문제를 해결하기 위해 Kim은 종법선 벡터를 이용하여 스네이크 포인트의 이동 방향을 결정한다.[9] 스네이크 포인트가 객체 윤곽 안에 있더라도 종법선 벡터를 계산하여 스네이크 포인트를 밖으로 이동하여 스네이크 포인트가 객체의 윤곽에 이동한다.

4. 실 험

실험은 Intel Core2 CPU 1.86GHz, 메모리 DDR2 2GByte, 윈도우 XP환경에서 Visual C++6.0을 이용하여 구현하였고 몇 개의 이진 실험영상에 실험하였다.

4.1 삽입된 스네이크 포인트 수

(그림 2)에 주어진 실험 영상은 초기 스네이크 포인트의 수(N_i)는 $v_0 ... v_9$인 10개로 설정하였다. <표 1>은 초기 스네이크 포인트의 간격의 평균 곡률 값과 삽입된 스네이크 포인트 수를 보여 준다. 평균 곡률 값이 크면 삽입된 스네이크 포인트 수가 많이 삽입된 것을 보여 준다. 삽입된 스네이크 포인트 수는 객체의 윤곽의 길이에 관계가 없고 윤곽의 휨 정도가 크면 많이 삽입하게 된다.



(그림 2) N_i = 10인 실험영상

〈표 1〉 초기 스네이크 포인트 구간의 평균 곡률 값과 삽입된 포인트 수

구간	평균 곡률 값	삽입된 스네이크 포인트 수
0~1	-0.521	10
1~2	-0.942	7
2~3	-0.601	11
3~4	-0.512	10
4~5	-0.918	9
5~6	-0.791	8
6~7	-0.995	5
7~8	-0.469	12
8~9	-0.743	9
9~0	-0.897	9

4.2 스네이크 포인트의 수(N_i)에 대한 성능 분석

스네이크 포인트의 수(N_i)가 많으면 에너지 함수의 반복 연산 수가 증가하여 소요 시간이 오래 걸린다. 또한 스네이크 포인트 수가 적으면 소요시간은 적지만 객체의 윤곽을 정확하게 추출할 수 없다. 그러므로 기존의 스네이크 알고리즘은 포인트의 수(N_i)에 많은 영향을 받는다. 포인트의 수(N_i)를 변경하여 성능을 분석하였다. 성능 평가는 RSD (Relative Shape Distortion)을 이용하였고 식 (8)에 주어진다.[8]

$$RSD(R) = \left(\sum_{(x,y)\in f} R_{original}(x,y) \oplus R_{estimation}(x,y) \right) / \sum_{(x,y)\in f} R_{original}(x,y) \quad (8)$$

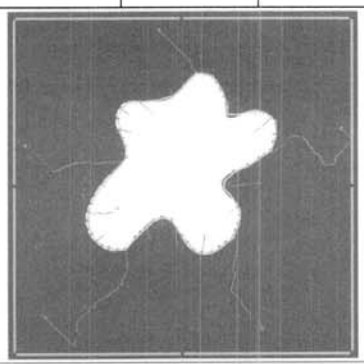
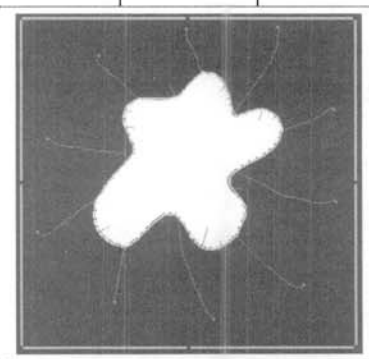
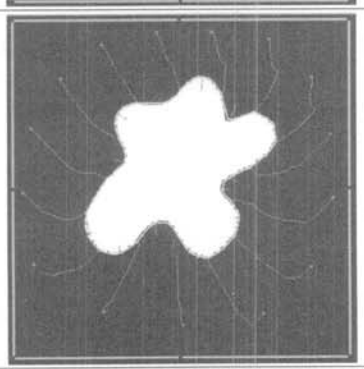
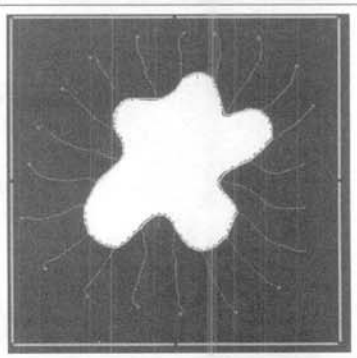
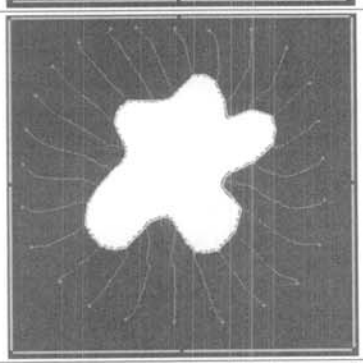
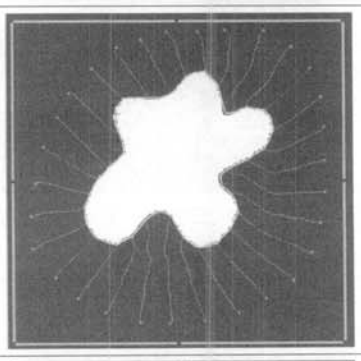
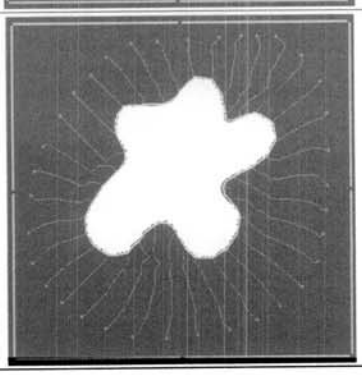
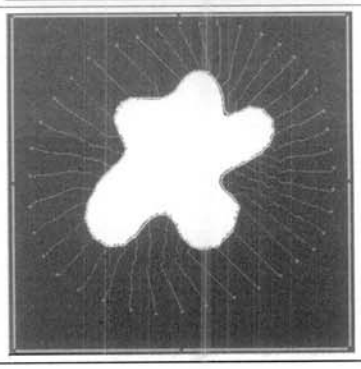
$R_{original}(x,y)$는 객체의 윤곽의 실제 좌표이고, $R_{estimation}(x,y)$는 실험한 객체의 좌표이다.

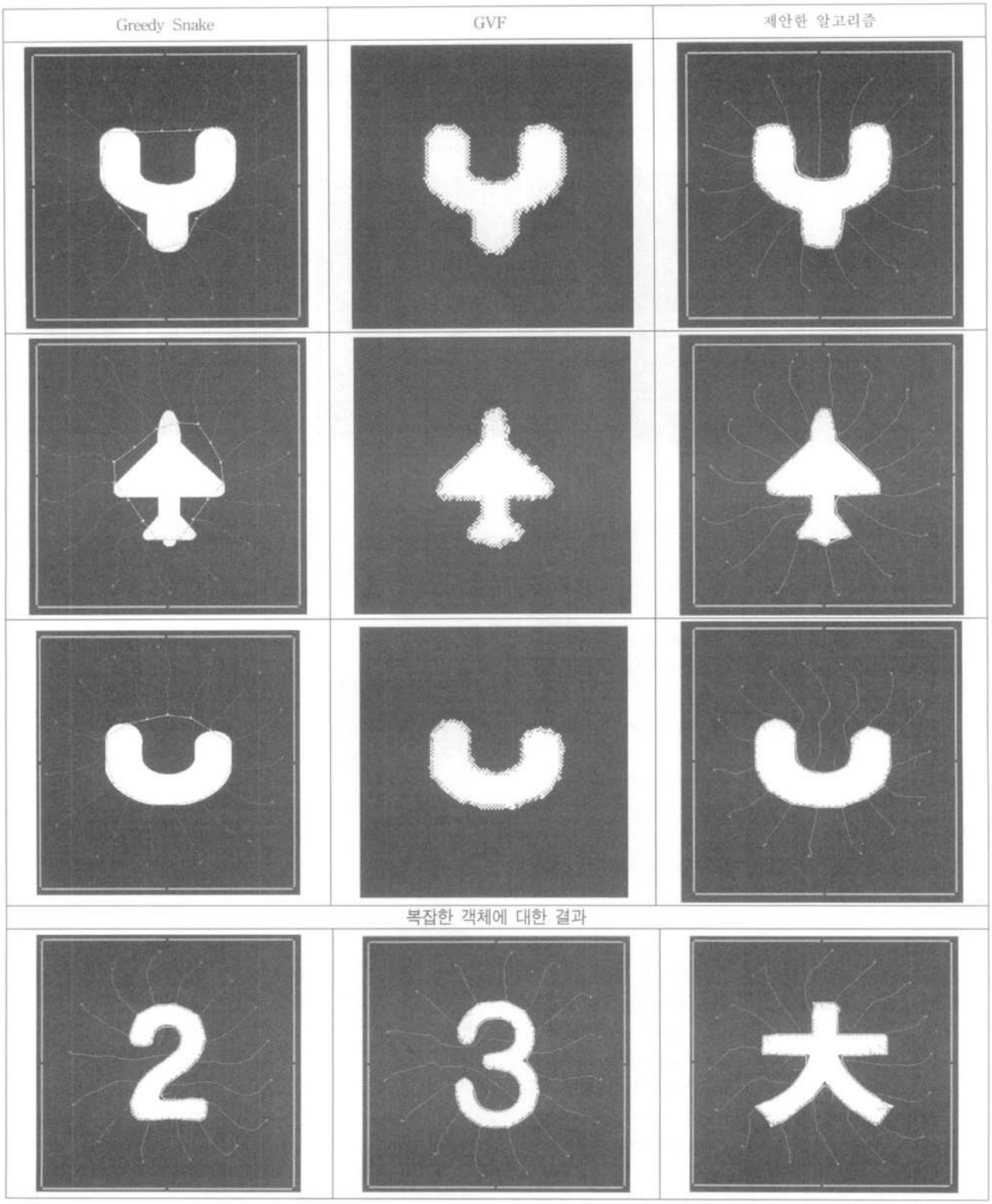
실험 결과는 <표 2>에 주어진다. N_f는 최종 스네이크 포인트 수이다. T_i, T_a, T_t는 각각 초기 스네이크 포인트가 객체의 윤곽에 수렴한 시간, 삽입된 스네이크 포인트가 객체의 윤곽에 수렴한 시간, 객체의 윤곽을 추출하기 위한 총 시간을 나타낸다. 포인트의 수(N_i)가 크면 T_i가 커진다. 포인트의 수(N_i)가 20 이상이면 T_i는 증가하지만 RSD는 거의 일정하다. 그러므로 포인트의 수(N_i)가 20 이상인 스네이크 포인트가 필요하지 않다. 포인트의 수(N_i)가 15이면 객체의 윤곽을 잘 추출할 수 있는 최적의 수이다. T_a는 객체의 윤곽을 수렴하기 위한 반복 수행 시간이 필요 없기 때문에 포인트의 수(N_i)에 관계없이 거의 일정한 시간이 걸린다. 실험에서는 포인트의 수(N_i)가 10이상이면 T_a는 약 0.06s의 거의 일정한 값을 나타낸다. 그러므로 T_t는 포인트의 수(N_i)에 가장 영향을 많이 받는다.

4.3 성능 분석 비교

<표 3>에서 제안한 알고리즘은 Greedy Snake와 GVF 방법과 비교하여 성능을 분석하였다. 포인트의 수(N_i)는 15개로 설정하였다. Greedy Snake은 T_t는 적지만 굴곡이 심한 윤곽을 추출하지 못하는 문제점이 있다. GVF는 RSD는 적지만 T_t가 커서 실시간 응용에 적용하기에는 문제점이 있다. 제안하는 알고리즘은 Greedy Snake보다는 T_t가 약간 크지만 성능은 GVF와 거의 같고, 굴곡이 심한 윤곽도 잘 추출할 수 있다. (그림 3)은 여러 영상에 대해 성능을 분석 비교하였다. 제안한 알고리즘은 여러 복잡한 영상에 대해서도 객체를 효율적으로 추출하는 것을 보여 준다.

〈표 2〉 N_i에 대한 성능 분석

N_i	5	10	15	20	25	30	35	40
N_f	70	100	94	107	112	99	102	104
T_i	0.007s	0.012s	0.016s	0.019s	0.023s	0.29s	0.32s	0.37s
T_a	0.054s	0.069s	0.063s	0.068s	0.069s	0.64s	0.61s	0.59s
T_t	0.061s	0.081s	0.079s	0.087s	0.092s	0.93s	0.93s	0.96s
RSD	283/26569 =0.001	135/26569 =0.005	0.004	0.004	0.003	0.003	0.003	0.003
결과영상	$N_i = 5$				$N_i = 10$			
	$N_i = 15$				$N_i = 20$			
	$N_i = 25$				$N_i = 30$			
	$N_i = 35$				$N_i = 40$			

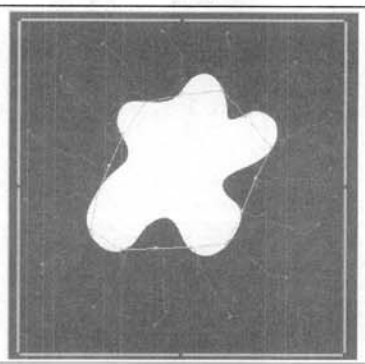
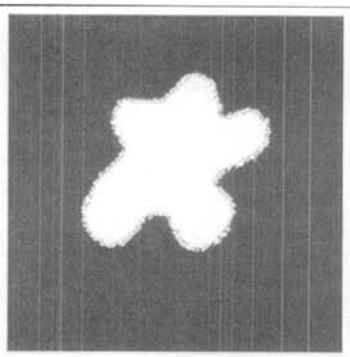
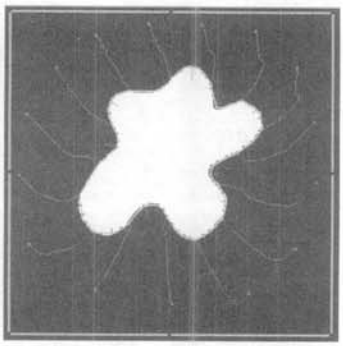


(그림 3) 여러 영상에 대한 성능 분석 비교와 복잡한 객체에 대한 실험 결과

5. 결 론

부분 곡률을 이용하여 굴곡이 심한 객체의 윤곽을 효율적으로 추출하는 알고리즘을 제안하였다. 스네이크 포인트의 수(N_i)는 속도 및 성능에 변수지만 일정 수 이상이면 거의 영향을 주지 않는 것을 보여 준다. 향후에는 초기 스네이크 포인트를 객체 밖에서 설정하는 것보다 객체의 윤곽을 잘 추출할 수 있는 위치에 설정하는 것에 대한 연구가 필요하다.

〈표 3〉 성능 분석 비교 ($N_i = 15$)

	Greedy Snake	GVF	제안한 알고리즘
T_t	0.011s	18.05s	0.079s
RSD	6536/26569=0.246	0/26569=0.000	132/26569=0.004
결과영상			

참 고 문 헌

[1] M. Bais, J. Cosmas, C.Dosch, A. Engelsberg, A. Erk, P. S. Hansen, P.Healey, G.K. Klungsoeyr, R. Mies, J.R. Ohm, Y.Parker, A. Pearmain, L. Pedersen, A. Sandvancd, R.Schafer, P.Schoonjans and P. Stammnitz, "Customized Television: Standards Compliant Advanced Digital Television," *IEEE, Trans. Brood.*. Vol.48, No.2, pp.151-158, June, 2002.

[2] D. H. Ballard and C. Brown, *Computer Vision*, Prentice-hall, 1982.

[3] M. Kass, A. Witkin and D. Terzopoulos. "Snake: Active Contour Models." *Int'l J. Computer Vision*, Vol.1, No.4, pp.321-331, 1987.

[4] D. J. Williams and M. Shah, "A Fast Algorithm for Active Contours." *Computer Vision*, Third International Conference on (1990), pp.592-595, 1990.

[5] K. M. Lam and H. Yan, "Fast Greedy Algorithm for Active Contours," *Electron Lett.*, Vol.30, pp.21-23, 1994.

[6] Xu, Chenyang and Jerry L. Prince. "Gradient Vector Flow : A New External Force for Snakes," In *Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition (CVPR '97)*, pp.66-71. IEEE Computer Sociey, 1997.

[7] Xu, Chenyang and Jerry L. Prince. "Snakes, Shapes, and Gradient Vector Flow," *IEEE Trans. on Image Processing*, Vol.7, No.3, pp.359-369, March, 1998.

[8] 김신형, 전병태, 장종환, "스테레오 영상에서 변이 정보를 결합한 새로운 스네이크 알고리즘," *한국통신학회 논문지*, 제11회 A, 제11호, pp.266-275. 2003.

[9] 김신형, 장종환, "오목한 윤곽을 갖는 객체에서 스네이크 기반의 윤곽선 검출방법," *한국정보처리학회 논문지*,제13회-B권, 제4호, pp.361-368, 2006.

[10] S. H. Kim, J. W. Jang and J. H. Choi. "Object Segmentation Algorithm Using Snakes in Stereo Images," *Optical Engineering*, Vol.45, No.3, pp.037005, Mar., 2006.

[11] K. W. Sum and P.Y.S. Cheung, "Boundary vector field for parametric active contours," *Pattern Recognition*, Vol.40, Issue 6, pp.1635-1645, Jun., 2007.

이 정 호

e-mail : style0428@naver.com

2008년 배재대학교 정보통신공학과(공학사)

2009년~현 재 배재대학교 정보통신공학과 석사과정

관심분야 : 영상처리, 컴퓨터 비전, 멀티미디어

최 완 석

e-mail : comber27@hotmail.com

2006년 배재대학교 정보통신공학과(공학사)

2008년 배재대학교 정보통신공학과(공학석사)

관심분야 : 영상처리, 컴퓨터 비전, 멀티미디어

장 종 환

e-mail : jangjw@pcu.ac.kr

1979년 한양대학교 전자통신공학과(공학사)

1986년 North Carolina 주립대학교 전기 및 컴퓨터 공학과(공학석사)

1990년 North Carolina 주립대학교 전기 및 전자 컴퓨터 공학과(공학박사)

1990년~현 재 배재대학교 정보통신공학과 교수

관심분야 : 영상처리, 멀티미디어 검색, 비디오 편집, 컴퓨터 비전