위상변화가 자유로운 기하학적 스네이크

김행강 서용덕 정문열

서강대학교 영상대학원
미디어공학과
미디어랩

요 약

3차원 메쉬에서 특징을 추출하는 것은 메쉬 에디팅이나 메쉬 모핑 등의 여러 가지 메쉬 처리에 있어서 중요한 일이다. 특징을 추출하는 방법 중에서 사용자가 지정한 부근의 특징을 자동적으로 찾아주는 방법은 이미지 처리 분야에서는 오래 전부터 사용되어 왔는데 이미지 스네이크 알고리즘이 그것이다. 최근에는 그러한 이미지 스네이크 알고리즘이 3차원 메쉬에 적용되어 기하학적인 스네이크 알고리즘으로 탄생하였다. 본 논문은 기하학적 스네이크의 새로운 알고리즘을 제시하고, 찾고자 하는 특징의 모양에 따라 스네이크 곡선의 위상이 자유롭게 변화하는 기하학적 스네이크 모델을 제안한다. 본 논문에 사용된 알고리즘은 이미지 스네이크 알고리즘의 동적 프로그래밍 방법을 3차원 메쉬에 응용한 것으로 스네이크 포인트들이 메쉬의 에지를 따라 3차원 상에서 직접 이동을 하면서 에너지가 최소가 되는 지점을 찾아 가는 방식이다. 스네이크 곡선은 메쉬 상의 이웃한 정점들의 순차적인 연결선으로 이루어지며 찾고자 하는 특징의 모양과 크기에 따라 스네이크 포인트의 개수가 자동으로 조절된다. 또한 주변의 다른 스네이크 포인트와 만났을 때 합쳐지거나 반대로 여러 스네이크 곡선으로 나뉘어 질 수 있다.

제 1 절 서론

3차원 메쉬에서의 특징 추출은 여러 가지 메쉬 처리에 있어서 중요한 문제이다. 메쉬 에디팅이나 모핑분야, 그리고 메쉬 압축에 이르기까지 광범위하게 메쉬의 특징을 이용한 처리 기법이 연구되고 있다 [11, 5]. 예를 들어 메쉬의 노이즈 제거나 스무딩 과정에서 얼굴 메쉬의 주름과 같은 세부적인 특징까지 살리면서도 노이즈는 완전히 제거하는 여러 스무딩 기법이 사용된다. 이렇게 정의된 특징을 완전히 자동적으로 추출하는 것은 상당히 어려운 문제이다. 최근에는 컴퓨터 비전분야에서 특징을 자동적으로 추출하는 방법에 대한 연구가 다양하게 이루어지고 있으나 아직은 완전한 자동화는 시기상조라고 할 수 있다. 반면, 반자동의 사용자 기반 특징 추출 연구가 그에 대한 하나의 대안으로 대두되었는데, 사용자가 특징 주변을 지정해주면 지정해 준 부근의 특징을 자동으로 찾아주는 방식이다. 사용자 기반 특징 추출은 자동으로 특징을 추출하는 것보다 좀 더 손쉽게 특징을 찾아갈 수 있다. 또한 구현이 쉽다는 점 이외에도 사용자의 주관적인 의도를 효과적으로 살릴 수 있다는 장점으로 여러 가지 분야에서 특징 추출 방법의 하나로 자주 사용되고 있다.

본 논문에서는 메쉬에서 사용자 기반 특징 추출 기법인 기하학적인 스네이크의 새로운 모델을 제시한다. 본 논문에서 제시하는 기하학적인 스네이크는 이미지 스네이크에서와 같이 사용자가 최초의 초기 스네이크 포인트를 특징 부근에 지정해주면 스네이크 포인트들이 에너지를 최소화하는 방향으로 움직이면서 특징 근처에서 수렴하게 된다. 또한 위상 변화가 자유로운 이미지 스네이크 기법을 이용하여 스네이크 곡선의 만나는 것을 감지하여 자동으로 위상변화를 조절해준다.

이미지의 스네이크 기법과의 가장 큰 차이는 기하학적인 스네이크에서는 스네이크의 움직임이 메쉬의 표면위에서 이루어져야 한다는 것이다. 그리고 최종적으로 찾아야 하는 특징이 이미지에서는 픽셀의 집합이라고 한다면 메쉬에서는 메쉬의 정점이나 에지의 집합이다. 이를 위해서 본 논문에서는 스네이크 포인트들이 메쉬의 정점 위에만 위치할 수 있게 하여 메쉬 구조를 이미지에서의 픽셀 혹은 그리드처럼 사용할 수 있게 한다. 따라서 스네이크 포인트들은 메쉬의 정점과 에지로 연결되어 서로 이웃해서 위치하며 모든 스네이크 포인트들은 에지를 따라서만 이동할 수 있다.

본 논문은 다음과 같은 목차로 구성된다. 먼저 2장에서는 관련 연구를 살펴본다. 그리고 3장에서는 이미지 스네이크의 기본 개념과 에너지 함수 정의, 위상 변화의 조절에 대해 살펴본다. 4장에서는 본 논문에서 제시하는 기하학적 스네이크의 알고리즘을 구체적으로 설명한다. 5장에서는 결과 이미지를 나타내고, 6장에서 결론을 맺고 있다.

제 2 절 관련 연구

2.1 사용자 기반 특징 추출

스네이크는 사용자 기반의 특징 추출 알고리즘으로 명시적 방법과 내재적 방법으로 크게 구분된다. 명시적 방법은 매개변수 곡선을 이용하여 스네이크 포인트를 구성하고 각각의 스네이크 포인트가 이동하는 위치를 추적하여 특징을 추출하는 방법이다 [6]. 명시적 방법의 경우 구현이 용이하고 구속조건을 주기가 용이하다는 장점이 있다. 내재적 방법은 윤곽선의 움직임을 추정하는 것으로 특징의 기하학적인 흐름을 이용하여 특징을 추출하는 방법이다 [3, 12, 10]. 내재적 방법으로 특징을 찾는 경우 특징의 모양에 따라 스네이크 커브가 위상 변화에 자유로운 반면 폐곡선이 아닌 열린 커브에는 적용하기 어렵다.

2.2 기하학적 스네이크 모델

기하학적 스네이크는 [7]에 처음 제시되었는데, 이 모델은 3차원에서 사용자가 특징 주변의 포인트를 지정해주면 지정된 포인트들을 일정영역으로 나누고 각각의 영역을 2차원으로 매개변수화 시키고 이미지 스네이크 알고리즘을 적용시켜 원하는 스네이크 곡선을 얻는다. 그리고 다시 3차원으로 돌아가 메쉬의 면에 스네이크 곡선을 최종적으로 투영시키는 방법을 취하고 있다. 이 모델에서는 이미지의 RGB값 대신 메쉬상에서 노말의 변화값을 이용하여 메쉬 에너지를 정의하고 노말의 변화가 큰 부분의 특징을 찾아가는 방식을 취했다.

제 3 절 이미지 스네이크

3.1 에너지 함수

이미지 스네이크 알고리즘은 Kass에 의해서 1987년 처음 제안되어 지금까지 여러 가지 발전을 거듭해왔다 [6]. 일반적으로 이미지 스네이크는 다음의 에너지 함수로 표현된다.

$$E_{snake} = \int_0^1 (E_{spline}(v) + E_{image}(v))ds \quad (1)$$

스네이크 점들의 위치는 $v(s) = v(x(s), y(s))$ 의 매개변수 형태로 표현된다. 이때 매개변수 s는, $s \in [0,1]$ 의 값을 가진다. $E_{spline}(v)$은 스네이크의 내부 에너지이고 $E_{image}(v)$은 외부 에너지를 의미한다. 여기서 $E_{spline}(v)$ 과 $E_{image}(v)$는 다시 다음과 같이 정의한다.

$$E_{spline} = (\alpha(s)\|v_s\|^2 + \beta(s)\|v_{ss}\|^2)/2 \quad (2)$$

$$E_{image} = -|\bigtriangledown I(x,y)|^2 \quad (3)$$

E_{spline} 에너지는 $v(s)$ 의 일계 미분(v_s)과 이계 미분(v_{ss})을 포함한다. $\alpha(s)$와 $\beta(s)$는 가중치 값이다. 일계 미분의 경우 스네이크 점들간의 거리를 조절하는 인자가 된다. 이계 미분은 스네이크 곡선의 곡률값과 관계가 된다. $\bigtriangledown I$는 이미지의 픽셀값의 변화값이다. 스네이크 에너지 함수는 오일러 라그랑즈 방정식(Euler-Lagrangian Equation)으로 변환되어 다음의 행렬식을 통해 스네이크 점들이 움직인다.

$$\begin{aligned} Ax + f_x(x,y) &= 0 \\ Ay + f_y(x,y) &= 0 \end{aligned} \quad (4)$$

여기서 A는 $v(s)$의 유한 차분법으로 근사한 항이며, $f_x(x,y), f_y(x,y)$ 는 이미지의 에너지함수의 편미분으로 표현된 항이다. 행렬식을 풀면 최종적으로 스네이크 포인트 각각에서 에너지가 최소가 되는 2차원 방향이 다음의 반복식에 의해서 계산된다.

$$\begin{aligned} x_t &= (A+\gamma I)^{-1}(\gamma x_{t-1} - f_x(x_{t-1}, y_{t-1})) \\ y_t &= (A+\gamma I)^{-1}(\gamma y_{t-1} - f_y(x_{t-1}, y_{t-1})) \end{aligned} \quad (5)$$

따라서 스네이크 포인트들은 반복적으로 이동하게 된다.

3.2 동적 프로그래밍

동적 프로그래밍을 통해서 에너지의 최소값을 찾아가는 방법이 있다. 동적 프로그래밍은 스네이크 점들이 갈 수 있는 방향을 한정하고 한정된 방향에서의 에너지 값을 계산하여 에너지가 최소가 되는 방향을 선택하여 이동하는 방식이다. 스네이크 곡선을 분연속의 점들로 표현하고 각각의 스네이크 점이 이동할 수 있는 지점을 정의하여 각각의 지점으로 이동했을 때 전체 에너지를 계산한 후 최소가 되는 지점으로 최적화한다. 이를 식으로 표현하면 다음과 같다.

$$\begin{aligned} E_t(i,k) = \min_{0\le j\le N} \{ & E_t(i-1,j) \\ & + \frac{1}{2}\alpha_i(v_i \bigoplus k - v_{(i-1+n)} \bigoplus j)^2 \\ & + E_{ext}(v_i \bigoplus k_i)\} \end{aligned} \quad (6)$$

여기서 v_m은 스네이크의 m번째 점을 의미한다. $\bigoplus$은 모든 이웃점에 대한 연산을 의미한다. 위 식을 풀면 $E_{min}(t) = min_k E_t(n-1,k)$인 최적화된 에너지가 계산되고 그 방향으로 스네이크를 움직인다. 이 알고리듬의 계산 속도는 $O(n(m+1)^2)$이다. 여기서 n은 스네이크 포인트의 갯수이고 m은 이웃 픽셀의 갯수이다. 구체적인 것은 [2, 1]에서 설명하고 있다.

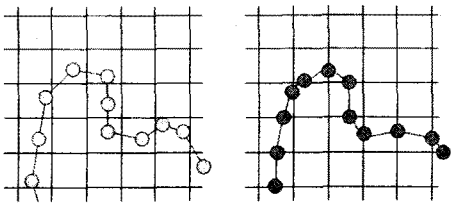


그림 1: 스네이크 점 개수를 조절하는 이미지 그리드 : 스네이크 포인트의 설정 처음의 스네이크 곡선(왼쪽) 이 사각형 그리드와 만나는 부분에 새로운 스네이크 포인트를 생성(오른쪽)하고 기존의 스네이크 포인트는 삭제한다.

스네이크 움직임을 찾는 다른 방법으로 그리디 알고리즘이 이용된다 [14]. 그리디 알고리즘의 경우에는 3개의 스네이크 점의 그룹으로 묶어진 영역에서 에너지 최소값을 찾고 스네이크 점의 위치를 갱신을 한다. 이후 그룹을 차례로 바꾸면서 이전의 스네이크 점의 갱신된 위치값을 이용하여 에너지를 반복적으로 계산하여 전체 에너지 최소값에 근접한다. 그리디 방법은 구속조건을 주기에 편하다는 장점이 있으며 계산 속도가 O(nm)으로 빠르다. 여기서 n은 스네이크 포인트의 개수이고 m은 주변 픽셀의 개수이다.

3.3 위상 조절

스네이크의 위상 변화를 자유롭게 하려면 앞에서 설명한 바와 같이 레벨셋 방법과 같은 내재적 방법이 일반적이다. 그러나 명시적 매개변수 방법에서도 자유로운 위상변화를 구현할 수 있는 여러 이론이 있다 [8, 4]. 명시적 방법에서 위상변화를 자유롭게 조절하기 위해서는 일반적으로 스네이크 포인트의 개수가 물체의 모양과 같은 특징에 따라 자유롭게 조절이 되어야 하고 포인트 사이의 충돌을 감지하여 스네이크 커브가 나뉘거나 합쳐질 수 있어야 한다. 이미지에서 이러한 조절은 이미지에 맞는 그리드를 사용함으로 이루어진다. 그림 1에서처럼 스네이크가 움직여 갈 때마다 스네이크 에지와 그리드가 만나는 지점을 새로운 스네이크 포인트로 구성하고 스네이크 곡선을 재구성한다. 작은 물체의 에지를 추출할 때에는 스네이크 포인트의 수가 작고 큰 물체에는 많은 포인트가 생겨서 스네이크 포인트 수의 자동 조절이 가능하다. 충돌을 감지할 때에는 한 그리드 내부에 스네이크 에지가 2개 이상 들어간 경우 충돌로 감지하고 매개변수의 인덱스를 조절해준다.

제 4 절 기하학적 스네이크

4.1 알고리듬

스네이크정점들은 다음의 가정을 항상 만족해야 한다.

- **가정 1 : 스네이크정점들은 메쉬의 정점(vertex)에 이웃하여 위치하고 에지(edge)로 연결되어 있다.**
- **가정 2 : 모든 스네이크정점들은 에지를 따라서만 이동할 수 있다.**

본 논문에서는 스네이크정점들이 메쉬의 정점 위에만 위치할 수 있게 하여 메쉬 구조를 이미지에서의 픽셀 혹은 그리드처

럼 사용할 수 있게 하였다. 이는 메쉬의 구조를 잘 살린 특징선을 얻을 수 있다는 장점이 있다. 스네이크 알고리듬은 다음과 같다. 처음에 사용자가 원하는 특징의 주변에 몇 개의 포인트를 입력해준다. 이 포인트들은 메쉬의 정점에 위치하게 된다. 그러면 시스템에서는 입력된 포인트를 연결하는 최단 에지 경로를 생성하게 되고 그 경로를 연결하는 매개 변수화된 메쉬의 정점이 스네이크의 초기 포인트가 된다. 스네이크 포인트의 이동 방향을 결정할 때에는 생성된 스네이크 포인트들의 현재 위치와 주변 메쉬 정점에서의 에너지 함수를 정의하고 스네이크 에너지가 최소가 되는 방향으로 스네이크 포인트들을 이동시킨다. 이 때 이전 단계에서 이웃하게 위치한 스네이크 포인트가 이동 후 겹쳐지는 부분은 하나의 포인트로 만들어주게 된다. 또, 이동 후 스네이크 포인트가 메쉬의 정점마다 이웃해 위치하지 않은 경우에는 포인트를 추가시켜 준다. 이러한 방법으로 스네이크 포인트의 개수를 조절해 주고 나면 다음으로 스네이크 포인트 간의 충돌을 체크한다. 충돌이 발생할 경우에는 스네이크 포인트들의 매개 변수 인덱스를 조절하여 스네이크 곡선을 나누거나 합쳐준다. 포인트 수의 조절과 충돌 체크는 루프를 반복해서 돌릴 때마다 수행해준다. 이러한 과정을 통하여 결과적으로 특징에서 수렴하는 스네이크 곡선을 얻을 수 있다.

4.2 에너지 함수

기하학적 스네이크에서 에너지는 다음과 같다. 이미지 스네이크와는 다르게 외부에너지로 메쉬에너지가 사용된다.

$$E_{snake} = \sum_{i=0}^{N} (E_{spline}(v_i) + E_{mesh}(v_i))ds \qquad (7)$$

이때 N은 스네이크 점의 개수이다. 스플라인 에너지의 경우 이미지의 스네이크 알고리즘의 스플라인 에너지를 그대로 사용한다. 스플라인 함수에서처럼 연속된 스네이크가 아닌 불연속된 정점로 스플라인 에너지를 표현한다. 따라서 이미지 스네이크에서는 유한 차분법으로 $v_s = v_i - v_{i-1}, v_{ss} = v_{i-1} - 2v_i + v_{i+1}$를 근사한다. 본 논문에서는 메쉬 에너지 함수를 다음과 같이 정의한다. 메쉬에서 곡률은 [9, 13]의 방법을 통해서 곡률을 구할 수 있다.

$$E_{mesh} = -|\kappa_G| \qquad (8)$$

또한 메쉬의 Gauss 곡률값에 가우시안 블러(bluring)을 적용하여 에너지 함수를 부드럽게 만들어 준 후 상한값을 주어 특징이 아닌 부분의 노이즈를 제거하는 효과를 얻을 수 있다.

4.3 스네이크의 움직임

처음에 스네이크 곡선은 이웃한 정점의 연속된 집합으로 초기화된다. 이는 사용자가 처음에 몇개의 스네이크 제어점을 찍어주면 시스템에서 자동으로 최단 연결 에지 경로를 찾아 내어 초기 스네이크 곡선을 구형하는 방식으로 한다. 그림 2과 같이 스네이크 곡선이 구성되면 이미지 스네이크의 동적프로그래밍을 통해서 스네이크의 움직임을 구할 수 있다. 여기서 이미지 스네이크와 메쉬 스네이크의 가장 큰 차이는 이미지에서는 검색하는 영역이 이웃 픽셀집합이라고 한다면 메쉬에서는 이웃 정점 집합이 된다는 것이다. 따라서 어떠한 스네이크 포인트 P 에서 이웃한 정점을 검색해보면 그림 2에서 나타내는 바와 같이 A,B 정점이 된다. 동적 프로그래밍에서는 A위치만을 검색해서 스네이크가 가야할 방향을 찾는다. 스네이크 포인트의 겹침을 최소로 하기 위해서 B 방향은 검색 방향에서 제외시킨다. 실험을 통해서 확인 해 본 결과 B 방향을 제외하는 것이 다음 장에서 설명할 스네이크 포인트의 겹침 제거에 효율적인 것을 확인하였다.

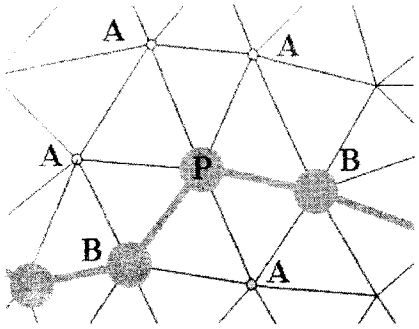


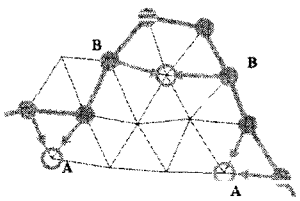
그림 2: 동적 프로그래밍에서의 스네이크 정점의 검색 방향 : 이웃 스네이크가 위치한 B 방향을 제외한 이웃 정점인 A 방향을 검색한다.

4.4 위상 조절

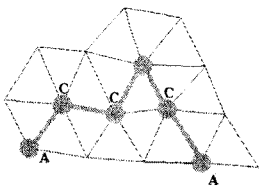
4.4.1 스네이크 정점의 수 조절

스네이크의 위상을 자동으로 조절하기 위해서는 우선 스네이크 포인트의 수를 자동으로 조절하는 단계가 필요하다. 이를 위해서 아래의 두단계가 필요하다.

- **MERGE** 스네이크 포인트 두 개가 하나의 메쉬 정점에서 만나는 경우 스네이크 포인트 하나를 삭제해준다. 이 과정을 'MERGE' 라고 부른다. 즉 겹치는 스네이크 포인트가 없도록 해준다.
- **INSERT** 다음으로 이웃한 스네이크 포이트가 에지로 연결되어 있지 않은 경우 사이에 스네이크 포인트를 채워 넣어 준다. 이를 통해 스네이크는 항상 이웃한 정점의 연속된 집합으로 구성될 수 있다.



(a)



(b)

그림 3: (a), (b) 순서로 진행하는 스네이크: 이웃하지 않은 스네이크 정점이 만나는 경우 에도 B 사이에 스네이크정점이 2개 이하이면 하나로 합쳐준다(MERGE). B 사이에 스네이크정점이 만약 3개 이상이면 겹침처리가 아닌 충돌로 처리한다. (b)에서는 스네이크 정점 사이에 빈 곳을 스네이크 정점C으로 추가시켜준다(INSERT).

바로 이웃한 스네이크 포인트가 겹치는 경우이외에도 스네이크 포인트를 삭제해 줄 수 있다. 그림 3에서 그 과정을 설명하고 있다. 그림 3의 (a)에서 보면 스네이크정점이 바로 이웃하지 않아도 한 정점에서 만나는 경우를 볼 수 있다. 이와 같은 경우 두 스네이크정점 B 사이에 몇개의 스네이크정점이 있는지를 조사한다. 충돌하는 스네이크 포인트 사이에 2개이하의 스네이크정점이 있으면 충돌하는 스네이크 포인트와 그 사이의 스네이크 포인트를 모두 제거하여 하나로 합쳐준다. 만약 3개이상의 스네이크 포인트가, 충돌하는 스네이크 포인트 사이에 존재하는 경우 이는 충돌로 처리하여 다음 4.4.2장에서 다룬다. 정점 A 사이에 스네이크 정점이 없는 경우에는 (그림 3(b))와 같이 스네이크정점 C 를 채워 넣어 준다. C는 최단 거리 정점이 추가된다. 프로그램상에서는 스네이크정점을 링크드리스트와 같은 자료구조에 넣고 INSERT와 MERGE를 자유롭게 구현한다. 이와 같은 과정을 통해 스네이크 커브는 메쉬 위의 이웃한 정점들로 연결되어 진행하게 되므로 추출하고자 하는 특징의 크기에 따라 자동으로 개수가 조절 될 수 있다. 이렇게 스네이크 점이 이웃한 메쉬의 정점마다 위치하게 되면 스네이크 점들 간의 충돌을 감지하는 것 또한 용이하다.

4.4.2 충돌 처리

스네이크 곡선의 위상이 변화하는 부분은 다음과 같은 알고리즘을 사용한다. 그림 4(a)에서와 같이 스네이크정점이 충돌을 하는 경우 A와 같은 하나의 에지를 이루는 경우가 발생한다. 또한 B와 같이 하나의 정점에서 만날 수도 있다. 이 경우 에지 A는 삭제하여 스네이크 정점의 매개변수 순서(index)를 조절하고 정점 B는 삭제하지 않고 서로 다른 스네이크 곡선이 되도록 기존의 스네이크 곡선의 인덱스를 조절한다. 만약 나누어진 스네이크 곡선안에 찾고자하는 특징이 존재하지 않는 경우 스네이크 곡선은 계속 줄어들어 사라질 수 도 있다. 이 같이 충돌이 동시에 일어나는 경우 스네이크 곡선의 인덱스를 한꺼번에 조절하지 않고, 하나의 충돌을 감지한 즉시 2개의 스네이크 곡선으로 나누고 다시 나누어진 스네이크 내에서 충돌이 있는가를 체크하여 충돌을 발견하면 다시 2개의 스네이크 곡선으로 나누는 작업을 수행한다. 이렇게 충돌을 체크하는 방법은 메쉬의 연결구조를 이용하기 때문에 이미지에서 보다 충돌 체크가 용이한 장점이 있다.

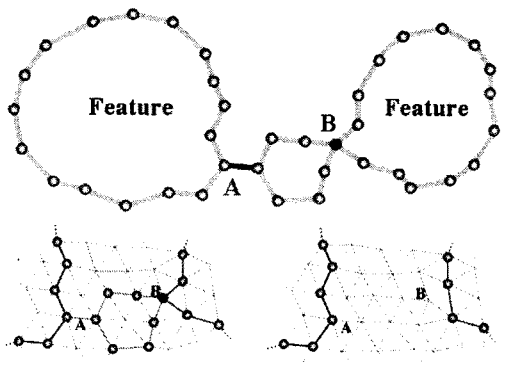
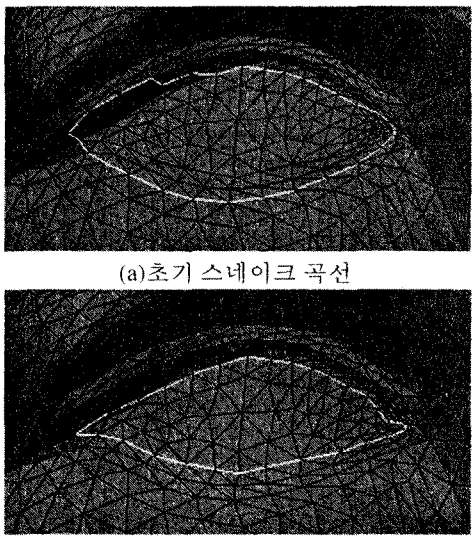


그림 4: 스네이크의 충돌과 나누어짐 : 하나의 스네이크 곡선이 A, B에서 충돌한다. 스네이크 곡선 내부에 특징이 존재하지 않는 경우 진화하여 스네이크 곡선이 줄어들다가 사라진다.

제 5 절 결과

먼저 본 실험은 삼각 메쉬를 대상으로 이루어졌다. 실험에 사용된 곡률은 전처리로 미리 구해둔다. 곡률을 계산하는데에는 수 초의 시간이 소요된다. 그림 5은 Greek 메쉬에서 실험한 결과이다. 초기 스네이크를 눈 주위에 대강 잡아주면 반복 6회를 수행하여 특징을 찾는다. 눈의 가장 왼쪽 스네이크 포인트는 반복 5회에서 고정 시켰다. 이미 찾은 특징점은 고정하여 특징을 더 용이하게 찾을 수 있다. 그림 6은 Skull 메쉬, Gorilla 메쉬에서 실험한 결과이다. 그림 6은 두 영역 모두 안장형 영역에서 $|\kappa_G|$가 최대가 되는 특징이다.



(a)초기 스네이크 곡선

(b)반복 6회로 찾은 특징

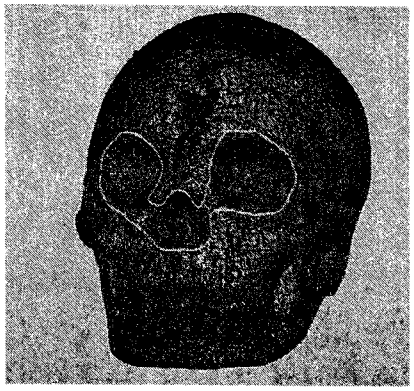
그림 5: Greek 메쉬의 눈 추출

제 6 절 Conclusion

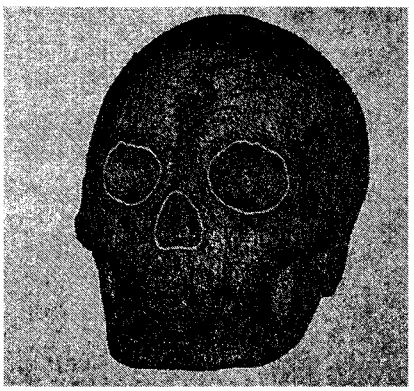
본 논문에서는 이미지 스네이크 알고리즘의 동적 프로그래밍 방법을 3차원 메쉬에 응용하여 스네이크 포인트들이 메쉬의 에지를 따라 3차원 상에서 직접 이동을 하면서 에너지가 최소가 되는 지점의 특징을 추출하였다. 또한 찾고자 하는 특징의 모양과 크기에 따라 스네이크 포인트의 개수가 자동으로 조절되며 주변의 다른 스네이크 포인트와 만났을 때 합쳐지거나 반대로 여러 스네이크 곡선으로 나뉘어 질 수 있다. 따라서 사용자가 여러 가지 특징을 찾을 때 각각의 특징 주변에 초기 입력 포인트를 주지 않고서도 한꺼번에 영역을 지정하여 여러 특징을 동시에 찾을 수 있는 장점이 있다. 또한 본 논문에서는 결과적으로 추출한 특징의 정보가 메쉬의 정점과 에지를 연결한 형태이므로 여러 가지 메쉬 처리 기법에 응용하기에 좋은 유용한 형태의 정보가 된다.

Acknowledgment

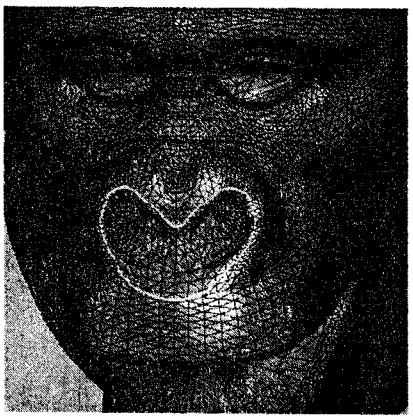
Skull 메쉬는 Headus(www.headus.com.au)에서 제작한 모델이다.



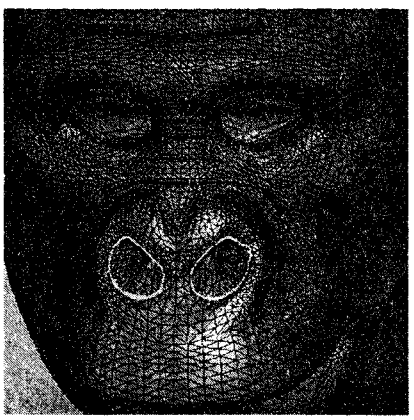
(a) 초기 스네이크 곡선.



(a) 초기 스네이크 곡선.



(c) 반복 12회 .



(c) 반복 8회 .

그림 6: 위상 변화가 조절되는 스네이크

참고 문헌

[1] A. Amini, S. Tehrani, and T. Weymouth. Using dynamic programming for minimizing the energy of active contours in the presence of hard constraints. In *Proc. ICCV*, pages 95–99, 1988.

[2] A. A. Amini, T. E. Weymouth, and R. C. Jain. Using dynamic programming for solving variational problem in vision. In *IEEE*, volume 12, September 1990.

[3] V. Caselles, R. Kimmel, and G. Sapiro. Geodesic active contours. In *Fifth ICCV*, June 1995.

[4] H. Delingette and J. Montagnat. New algorithms for controlling active contours shape and topology. In *Proc. ECCV '00*, pages 381–395, 2000.

[5] A. Hubeli and M. Gross. Multiresoultion feature extraction from unstructured meshes. In *12th IEEE Visualization 2001*, 2001.

[6] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *International journal of Computer Vision*, 1(4):321–331, July 1987.

[7] Y. J. Lee and S. Y. Lee. Geometric snake for triangular meshes. In *Proc. Eurographics '02*, volume 21, 2002.

[8] T. McInerney and D. Terzopoulos. Topologically adaptable snakes. In *Proc. ICCV '95*, pages 840–845, June 1995.

[9] M. Meyer, M. Desbrun, P. Schroder, and A.H. Barr. Discrete differential geometric operator for triangulated 2-manifolds. 2002.

[10] S. Osher and N. Paragios (eds). *Geometric Level Set Methods in Imaging, Vision and Graphics*. Springer Verlag, 2002.

[11] C. Rossl, L. Kobbelt, and H. P. Seidel. Extraction of feature lines on triangulated surface using morphological operators. In *Smart Graphics (AAAI Spring Symposium-00)*, 2000.

[12] J. A. Sethian. *Level Set Methods and Fast Marching Methods*. Cambridge University Press, 1999.

[13] G. Taubin. Estimating the tensor of curvature of a surface from a polyhedral approximation. In *Proc. ICCV '95*, pages 902–907, June 1995.

[14] D. J. Williams and M. Shah. A fast algorithm for active contours and curvature estimation. *CVGIP:Image understanding*, 55(1):14–26, January 1992.