3차 베지에 곡면에 대한 Bisector Surface의 생성
(Bisector Surface Construction for Two Bicubic Bezier Surfaces)

지도교수 : 김명수

이 논문을 공학학사 학위 논문으로 제출함.

2018년 12월 27일

서울대학교 인문대학
고 고 미 술 사 학 과
손 상 현

2019년 2월

초록

본 논문에서는 두 개의 3차 베지에 곡면(Bezier Surface)이 주어졌을 때, 그 중 하나의 곡면이 다른 곡면에 대해 가지는 bisector surface를 효율적으로 구하는 방법을 제시한다. Bisector는 medial axis, medial surface와 깊은 연관을 맺고 있고, 이는 보로노이 다이어그램(Voronoi diagram)을 구성하는 기초가 된다는 점에서 중요하다. Bisector의 토폴로지(Topology)를 결정하는 점들을 얻기 위해서 주곡면에서 여러 정점들을 샘플링하고, 각 정점에 접하는 maximal sphere를 찾는다. 이 과정에서 충돌 감지(Collision Detection) 기법을 사용하여 sphere의 크기를 오차 범위 내로 조정해나간다. 효율적인 충돌 감지를 수행하기 위해 각 곡면을 감싸는 Bounding Volume Hierarchy(BVH) 를 사용하여 계산한다. 전체 과정을 가속화하기 위하여 거리 계산 과정을 개선하고, 인접 정보(Locality)를 활용하며, GPU를 사용한다.

주요어: Bisector, Medial surface, 보로노이 다이어그램, 베지에 곡면

학번: 2012-10242

목차

그림 목차

제 1 장 서론

1.1 연구의 배경 및 목적

Bisector란 평면 혹은 공간 상에서 두 물체가 주어졌을 때, 그들로부터의 최단 거리가 동일한 점들의 집합을 의미한다. Bisector는 경우에 따라 곡선일 수도 있고 곡면일 수도 있다. 이러한 bisector는 medial axis, 그리고 medial surface와 긴밀하게 연관되어 있다. [4]

평면 상에서 medial axis 는 어떤 형태(Pattern)가 주어졌을 때, 그 형태와의 최단 거리를 산출하는 정점이 2개 이상인 점들의 집합을 의미한다. 따라서, medial axis 는 해당 형태의 대칭 구조를 나타낸다고 할 수 있다. [2] Medial surface 란 medial axis 를 3차원 공간으로 확장한 개념이다. 즉 medial surface 는 3차원 공간에서 주어진 특정 형태에 대해 최단 거리를 형성하는 정점이 2개 이상인 점들의 집합을 가리킨다.

Medial axis 와 medial surface 는 모두 보로노이 다이어그램(Voronoi diagram)을 구성한다는 점에서 중요하다. Medial axis와 medial surface 는 보로노이 다이어그램을 구성하는 보로노이 셀(Voronoi cell)의 경계(Boundary)가 된다. 보로노이 다이어그램이란 공간에 특정 지점들이 주어졌을 때, 각 지점까지의 거리가 가장 가까운 점들의 집합들로 공간을 분할한 것을 의미한다. 보로노이 다이어그램은 여러 가지 유용한 수학적 특성을 가지고 있고, 여러 종류의 공학 문제들을 해결하는데 강력한 도구가 될 수 있다는 점에서 중요하다. [1]

Bisector는 서로 다른 두 물체로부터의 최단 거리에 대해 다루지만, medial axis와 medial surface는 단일한 형태의 내적 구조에 대해 다룬다는 점이 다르다.

하지만 bisector를 구하는 작업은 medial axis와 medial surface, 더 나아가 보로노이 다이어그램을 구하는 기초 작업이 될 수 있다. 지금까지 평면 상에서 이러한 기하학적 구조를 수치적으로 계산하려는 시도는 많이 이루어져 왔다. 하지만 3차원 공간 상에서는 그러한 시도가 평면에 비해 많이 이루어지지 않았다. 따라서 본 논문에서는 이를 위한 기초 작업으로 2개의 3차 베지에 곡면에 대한 bisector를 구하도록 한다.

1.2 관련 연구

보로노이 다이어그램과 medial axis의 정의, 그리고 수치적으로 계산하는 방법에 관해서는 Aurenhammer에 의해 잘 정리된 바 있다. [1]

Medial axis와 bisector를 구하는 방법은 다양하게 제시되었다. 그 중 하나는 implicit bisector function을 이용하여 찾는 것이다. [4] [5] [6] 이 방법은 medial axis나 bisector 상의 한 점과 그에 대응하는 형태 상의 두 정점이 맺는 관계를 이용하여 수식을 세우고 그 해를 찾는다.

다른 방법은 maximal touching disk 를 이용하는 방법이다. Maximal touching disk 는 medial axis 를 정의하는 여러 방법 중 하나에 해당한다. 특히 medial axis 상의 임의의 한 점의 위치와 그에 대응하는 maximal touching disk 의 반지름이 주어졌을 때, 그것은 medial axis transform 이라 불리며 원래 형태를 완벽하게 묘사한다. [2] [3] Maximal touching disk 를 이용하여 평면 상의 자유형상곡선 (Planar freeform curves)에 대해 medial axis 와 보로노이 다이어그램을 빠른 시간 내에 계산해 낸 연구도 있다. [8]

본 논문에서는 maximal disk 를 이용하여 3차원 공간에서의 medial surface 를 찾는다. 공간이 3차원으로 확장하여 maximal disk 역시 maximal sphere 로 칭한다. 다음 장에서 이 방법에 관해 자세히 논의한다.

1.3 논문의 구성

본 논문은 다음과 같이 구성된다. 2.1장에서는 베지에 곡면과 bisector, 그리고 maximal sphere의 정의에 대해 알아본다. 2.2장에서는 maximal sphere를 통해 bisector surface를 찾는 방법에 대해서 논한다. 2.3장에서는 maximal sphere를 수치적으로 계산하는 알고리즘을 설명한다. 2.4장에서는 앞의 알고리즘을 가속화하기 위한 방법들에 대해서 논하고, 2.5장에서는 실험 결과를 제시한다. 이후 3장에서는 결론을 내리고 향후 계획을 제시한다.

제 2 장 본론

2.1 예비지식

이 장에서는 베지에 곡면과 bisector, 그리고 maximal sphere의 정의에 대해서 논한다.

2.1.1 베지에 곡면

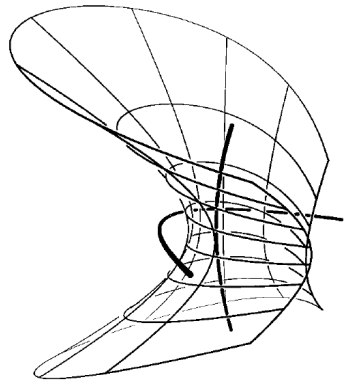
양의 정수인 차수 n, m의 베지에 곡면 $S(u,v)$는 $(n+1)(m+1)$개의 제어점 $\mathbf{p}_{i,j}$로 다음과 같이 정의된다.

$$\mathbf{S}(u,v) = \sum_{i=0}^{n}\sum_{j=0}^{m} B_i^n(u)B_j^m(v)\mathbf{p}_{i,j}, \qquad 0 \le u,v \le 1 \tag{2.1}$$

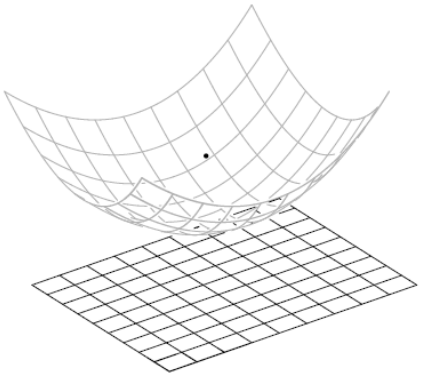
따라서 3차 베지에 곡면은 16개의 제어점으로 정의된다. 제어점이 잘 주어질 경우, 3차 베지에 곡면은 어디에서나 부드러우며 경계를 제외한 모든 점에서 잘 정의된 법선 벡터를 가진다. 제어점을 변형함으로써 곡면의 모양을 쉽게 변형할 수도 있다. 이와 같이 좋은 특성을 지녔기에 본 논문에서는 3차 베지에 곡면을 이용하여 medial surface를 구하도록 한다.

2.1.2 Bisector

3차원 공간에 주어진 2개의 물체에 대한 bisector는 해당 물체들로부터의 최단 거리가 같은 점들의 집합을 의미한다. 공간 상에서 형성되는 bisector는 흔히 surface의 형태를 띄고, 다음과 같이 다양한 물체들의 경우에 적용할 수 있다. [4] [5]

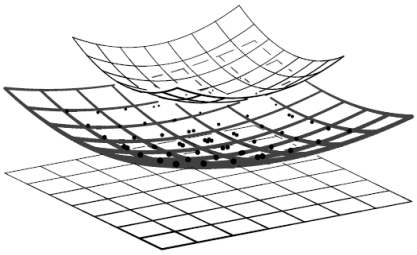


(a) 이차 베지에 곡선 사이의 bisector

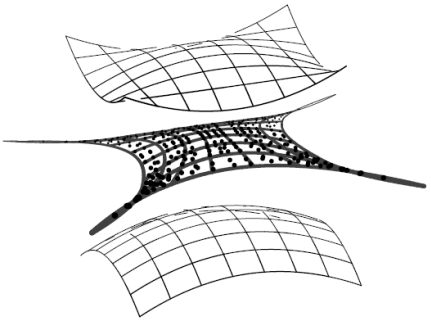


(b) 정점과 평면 사이의 bisector

본 논문에서는 3차 베지에 곡면 2개에 대한 bisector surface를 구한다. 기존의 연구(surfsurfbisector)에서 구한 두 곡면 사이의 bisector surface는 다음과 같다. [6]



(a) 볼록 곡면과 평면 사이의 bisector



(b) 자유형상곡면 사이의 bisector

위의 결과는 두 곡면에 대해 bisector surface 상의 한 점이 만족해야 하는 constraint equation을 세우고 그것의 해를 찾아서 구한 것이다. 하지만 본 논문에서는 다음에 살펴볼 maximal sphere를 이용하여 bisector surface를 구하고자 한다.

2.1.3 Maximal sphere

Maximal sphere는 평면 상의 maximal disk의 3차원 공간에 대한 확장이다. 평면 상에서 maximal disk는 medial axis를 정의하는 방법 중 하나로 사용된다. 정의는 다음과 같이 이루어진다.

Medial axis를 구하는 평면 상의 domain을 Ω, 중점이 p이고 반지름이 r인 원을 $\mathbb{B}_r(p)$라 하자. 이 때, $\mathbb{D}(\Omega)$를 Ω에 포함되는 모든 원의 집합이라 가정한다. 동시에 Ω의 core, 즉 CORE(Ω)라는 집합을 정의한다. CORE(Ω)는 $\mathbb{D}(\Omega)$의 부분집합으로, 그것에 속하는 $\mathbb{B}_r(p)$는 다음과 같은 성질을 만족한다.

$$\text{If } \mathbb{B}_r(p) \in CORE(\Omega),\ \text{every } \mathbb{B}_s(q) \text{ in } \mathbb{D}(\Omega) \text{ such that } \mathbb{B}_r(p) \subset \mathbb{B}_s(q) \text{ equals } \mathbb{B}_r(p). \tag{2.2}$$

이 때 CORE(Ω)에 속하는 원 $\mathbb{B}_r(p)$을 maximal disk라 한다. Domain Ω의 medial axis는 CORE(Ω)에 속하는 모든 maximal disk의 중점들의 집합들로 정의된다. [3]

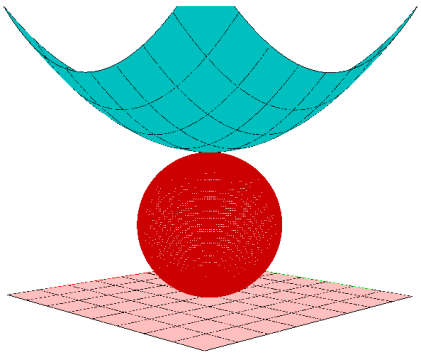
Medial disk와 마찬가지로, maximal sphere는 3차원 공간에서 medial surface를 구할 때 사용할 수 있다. 이를 응용하여, bisector surface를 구하기 위해 maximal sphere를 사용한다. 다음 절에서부터 Maximal sphere를 사용하여 bisector surface를 구하는 과정을 자세히 설명한다.

2.2 Maximal sphere와 bisector surface

2개의 3차 베지에 곡면 A, B가 주어졌다고 가정한다. 앞으로의 논의에서 주 곡면은 A라고 두자. A상의 한 점 p를 선택했을 때, p에서는 법선 벡터 $\mathbb{N}(t)$가 잘 정의된다. 점 P에서의 법선 $\mathbb{L}(t)$는 점 p의 위치와 $\mathbb{N}(t)$에 의해 결정된다. 이 때 점 p에 접하는 sphere $\mathbb{B}_r(q)$의 중심 q는 $\mathbb{L}(t)$ 상에 위치하며, p까지의 거리와 동일한

길이의 반지름 r을 갖는다. 이러한 $\mathbb{B}_r(q)$가 다른 곡면 B와 외부의 한 점에서 접하는 경우, $\mathbb{B}_r(q)$를 p에서의 maximal sphere라고 한다. 이렇게 구한 maximal sphere의 중점을 이용하여 bisector surface의 토폴로지를 결정할 수 있다.

본 논문에서는 bisector surface의 토폴로지를 결정하기 위해 베지에 곡면 A의 domain을 각각 u 방향으로 1024등분, v 방향으로 1024등분 한 뒤 점들을 샘플링한다. 따라서 샘플링되는 총 정점의 개수는 2^{20}개이다. 각각의 정점에서 maximal sphere를 찾게 되는데, 그 과정에서 충돌 감지를 수행한다. 정점 p에서 충돌 감지를 수행한 결과 충돌이 발생하지 않았다면 sphere의 반지름은 커져야 한다. 반대로, 충돌이 발생했다면 sphere의 반지름은 작아져야 한다. 이와 같은 과정을 반복하여 반지름의 길이가 오차 범위 내로 수렴하게 된다면 그 때의 sphere를 maximal sphere라고 할 수 있다.



(a) 평면 상의 한 점에서의 maximal sphere

2.3 Maximal sphere의 수치적 계산

Maximal sphere을 수치적으로 찾기 위해 충돌 감지를 수행한다. 충돌 감지는 각 베지에 곡면에 대해 Bounding Volume Hierarchy(BVH)를 만들어서 수행한다. Filip에 의하면, 직사각형 domain $R \subset \mathbb{R}^2$상에서 정의된 C^2 함수 $\mathbf{f} : R \to \mathbb{R}^3$를

사각형으로 근사할 때 오차는 domain의 크기에 의존한다. [7] 3차 베지에 곡면은 조건을 만족하므로 이를 이용하여 BVH를 쉽게 구성할 수 있다.

이제 maximal sphere을 수치적으로 구하는 순차 알고리즘을 다음과 같이 제시한다.

1. 곡면 A와 A상의 점 P에 대해, 임의의 반지름 r을 가정하고 sphere S를 생성한다.

2. S가 다른 곡면 B와 충돌하는지 검사한다.

3-1. 만약 충돌한다면, r을 줄인다.

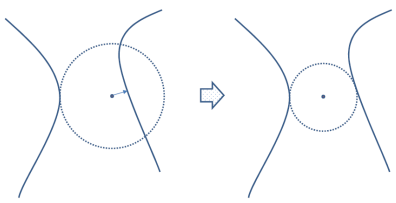
3-2. 만약 충돌하지 않는다면, r을 늘린다.

4. 만약 r이 오차범위 내에 있다면 그것을 반환한다. 그렇지 않다면 2로 돌아간다. (본 논문에서는 오차 범위를 10^{-4}로 두었다.)

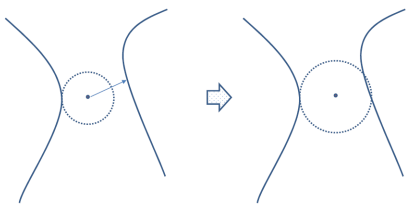
알고리즘의 세부적인 구현을 살펴보도록 한다.

2.3.1 새로운 반지름의 계산

위 알고리즘에서 반지름이 바뀌어야 하는 경우는 sphere S가 충돌하는 경우, 충돌하지 않는 경우의 두 가지 경우이다. 이를 단순화하여 2차원 평면 상에서 확인하면 다음과 같다.



(a) Sphere S가 충돌하는 경우



(b) Sphere S가 충돌하지 않는 경우

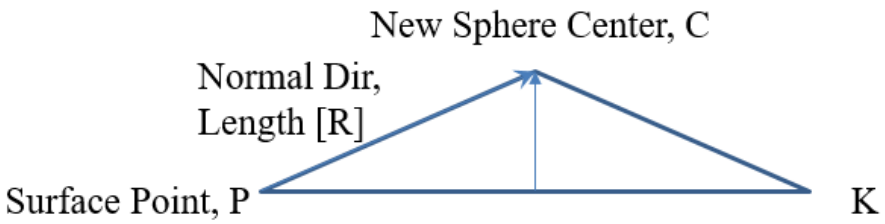
하지만, 두 경우 모두 새로운 반지름을 계산하는데 동일한 알고리즘을 사용

할 수 있다. BVH를 사용하여 충돌 검사를 수행하면서 새로운 반지름을 계산하기 위한 기준이 될 최우선 노드 N을 다음과 같이 설정할 수 있다.

1. 충돌하는 경우 : 충돌하는 leaf node들 중 제일 가까운 노드를 N으로 선택한다.

2. 충돌하지 않는 경우 : 충돌하는 노드들 중 제일 깊은(deepest) 노드를 선택한다. 같은 깊이의 노드들 중에서는 제일 가까운 노드를 선택한다. 해당 노드의 자식 노드들 중, 제일 가까운 노드를 N으로 선택한다. 기존에 충돌하는 노드가 아예 없다면 root node를 N으로 선택한다.

이와 같이 N을 선택할 때, N 상에서 현재 원의 중심으로부터 제일 가까운 점 K를 선택할 수 있다. 이제 sphere S가 점 K에서 노드 N과 닿을 수 있도록 새로운 반지름을 계산한다. 이는 간단한 삼각함수 식을 이용하여 구할 수 있다. 이를 도식화하면 다음과 같다.



(a) 새로운 반지름의 계산

$\theta = \angle CPK$, $\mathbf{v}_1 =$ P에서의 길이가 1인 법선 벡터, $\mathbf{v}_2 = K - P$ 일 때 새로운 반지름 r은 다음과 같이 정해진다.

$$r = \frac{|\mathbf{v}_2|}{2\cos\theta} \tag{2.3}$$

이렇게 새로운 반지름을 정하면 각각의 경우에 대해 다음과 같이 반지름이 변하게 된다.

1. 충돌하는 경우 : 앞서 최우선 노드를 충돌하는 leaf node들 중에서 선택하였다. 따라서, 새로운 sphere는 최우선 노드와 여전히 만난다. 하지만 이전 최단거리를 형성했던 점을 반드시 지나므로 새로운 반지름은 이전에 비해 작거나 같다.

2. 충돌하지 않는 경우 : 최우선 노드가 이전의 충돌하는 노드들 중 제일 깊은 노드의 자식 노드였으므로, 이전에는 충돌하지 않았던 노드에 해당한다. 이를 기준으로 반지름을 계산하면 최우선 노드와 충돌하므로 새로운 반지름은 이전의 반지름에 비해 크다.

따라서 충돌하는 경우에는 반지름이 이전에 비해 작거나 같아지며, 충돌하지 않는 경우 반지름은 이전보다 커진다. 이와 같이 동일한 알고리즘을 사용하여 서로 다른 경우에 대해 반지름을 알맞게 조절할 수 있다.

2.3.2 Touching sphere의 특성

동일한 정점에 접하는 두 sphere S_1, S_2의 반지름을 각각 r_1, r_2라 하자. 이 때, $r_1 \leq r_2$이면 $S_1 \subseteq S_2$임이 자명하다. 따라서, 곡면 B상의 어떤 bounding volume $\mathbf{b}$가 S_1과 만난다면, $\mathbf{b}$는 S_2와도 만난다. 반대로, $\mathbf{b}$가 S_2와 만나지 않는다면, 그것은 S_1과도 만나지 않는다.

이를 이용하면, 접촉원이 충분히 커져서 leaf node들을 만난 이후에는 충돌 감지를 위해 BVH를 root부터 검사할 필요가 없게 된다. 즉, 접촉원 S가 반지름 r_1일 때 leaf node N만을 만났다고 가정하자. 이 때, 앞서 살펴본 반지름 계산 알고리즘에 의하면 새로운 반지름 r_2는 r_1에 비해 작거나 같다. 따라서, 반지름이 r_2일 때 S는 r_1일 때 만났던 K 외에는 다른 leaf node와 만날 수 없다. 이로 인해 충돌 감지는 K만을 대상으로 하면 된다. 만약 r_2일 때 S가 아무런 leaf node와도 만나지 않는다 하더라도, 우리가 찾는 반지름 R은 r_1보다 작아져야 하므로 K 노드 이외의 노드를 대상으로 충돌 감지를 할 필요가 없다.

이를 고려하여, 앞서 제시한 알고리즘을 두 개의 단계로 나눌 수 있다.

1. Leaf 노드와 만나기 전 : Leaf 노드와 만날 때까지 위의 알고리즘을 이용, 반지름을 증가시킨다.

2. Leaf 노드와 만난 후 : 만난 Leaf 노드의 집합을 유지/축소하면서 반지름을 감소시킨다. 만약 반지름을 어느 정도 감소시켰을 때 leaf node의 집합이 이전에 비해 작다면, 그것을 새로운 leaf node 집합으로 간주하여 알고리즘을 계속한다. 반지름의 길이가 오차 범위 내에 들어오면 그것을 반환한다.

본 논문에서 구현한 알고리즘은 이와 같이 두 단계로 구성하여 가속화 하였다.

2.4 가속화 방안

지금까지 살펴본 알고리즘을 가속화하기 위한 방법은 다음과 같다.

2.4.1 거리 계산 과정 개선

본 논문에서 구현하고 있는 알고리즘은 충돌 감지에 기반하고 있고, 충돌 감지의 핵심은 거리 계산이다. 따라서 거리 계산의 속도가 전체 알고리즘의 속도를 결정한다. 이를 고려할 때 BVH를 구성하는 bounding volume의 형태를 바꿔서 가속화할 수 있다.

이전에는 Tetrahedron Swept Sphere(TSS)를 사용하여 bounding volume을 구성하였다. TSS와 한 점 사이의 거리는 사면체와 한 점 사이의 거리를 구함으로써 얻는다. 이 때 GJK 알고리즘을 사용하여 거리를 빠른 시간 내에 구할 수 있다.

하지만 TSS 대신 Rectangle Swept Sphere(RSS)를 사용하여 bounding volume을 구성할 수 있다. 베지에 곡면을 이루는 작은 곡면들을 사각형으로 근사한 후 RSS로 만들 수 있다. RSS와 한 점 사이의 거리는 2개의 삼각형과 한 점 사이의 거리를 구함으로써 얻는다. 이 때 GJK 알고리즘 대신 삼각형과 한 점 사이의

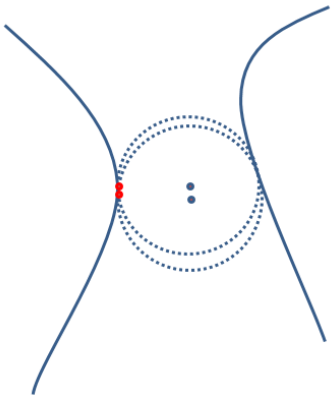
거리를 구하는 단순한 알고리즘을 사용함으로써 속도를 가속화할 수 있다.

이와 같이 구현한 결과 이전에 비해 거리 계산에서 2배 정도의 속도 향상을 확인하였다.

2.4.2 인접 정보 활용

본 논문의 알고리즘은 하나의 베지에 곡면 상에서 2^{20}개의 정점을 선택하여 bisector surface의 토폴로지를 결정하는 데에 활용한다. 지금까지는 각 정점에서 maximal sphere를 구할 때 반지름의 초기값을 임의의 값으로 설정하였다. 그러나 인접하여 위치한 정점들의 경우, 그 중 한 정점의 maximal sphere S를 구했을 때 다른 정점들의 maximal sphere는 S를 이용하여 빠르게 구할 수 있다.

평면 상에서 간단한 예시를 다음과 같이 도식화 할 수 있다.



(a) 인접한 정점에서의 maximal sphere

Maximal sphere를 찾는 정점들이 가까울수록 maximal sphere의 크기, 그리고 다른 곡면 B와 만나는 위치가 비슷하다. 이 정보를 각 정점마다 저장하여 활용할 수 있다. 실험 결과, 약 3배 정도의 속도 향상을 확인하였다.

2.4.3 GPU 가속화

본 논문의 알고리즘은 2^{20} 개의 정점을 선택하여 계산한다. 이는 방대한 계산량을 요구한다. 하지만 위에서 제시한 알고리즘은 모든 정점에 대해 동일하게 작동하여 예외 처리의 필요성이 크지 않다. 이와 같은 점을 고려하여 본 논문에서는 CPU 대신 GPU를 사용하여 속도를 가속화하였다.

GPU를 사용하기 위해 CUDA를 이용하여 위 알고리즘을 설계하였다. 하지만 이와 같이 GPU만 사용하여 알고리즘을 구현한 결과, 앞서 살펴본 인접 정보를 이용하는 데에 어려움을 겪었다. 따라서 본 논문에서는 일단 인접 정보 이용을 잠시 보류하고 GPU 알고리즘을 구현하였다. 시뮬레이션 결과는 GPU를 사용하여 계산한 결과이다.

2.5 시뮬레이션 결과

본 논문은 두 베지에 곡면들이 어떤 형태로 배치되어 있어도 bisector surface를 안정적으로 구할 수 있도록 알고리즘을 구현하였다. 따라서 베지에 곡면들을 다양한 형태로 배치하여 bisector surface를 계산하였다. 실험 결과는 기존의 소프트웨어(IRIT)에서 구한 bisector surface의 토폴로지와 비교해보았다. 또한 곡면이 여러 형태로 변형될 때 시간을 측정하였다.

2.5.1 토폴로지 비교

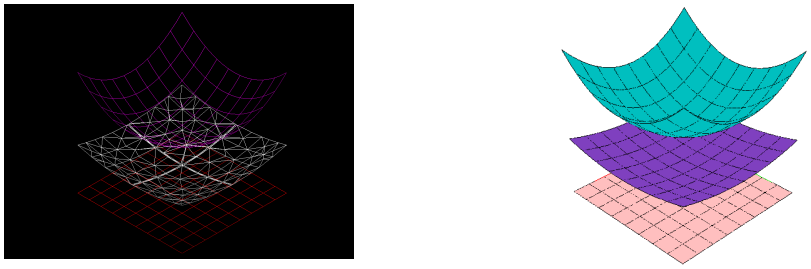


그림 2.7: 토폴로지 1 . IRIT(좌), 논문(우)

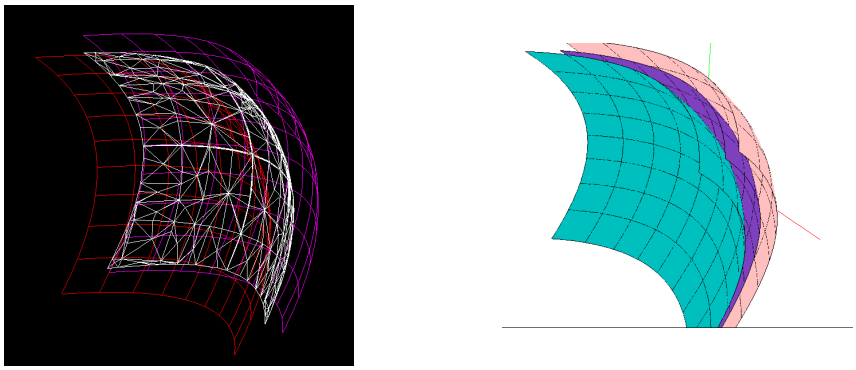


그림 2.8: 토폴로지 2 : IRIT(좌), 논문(우)

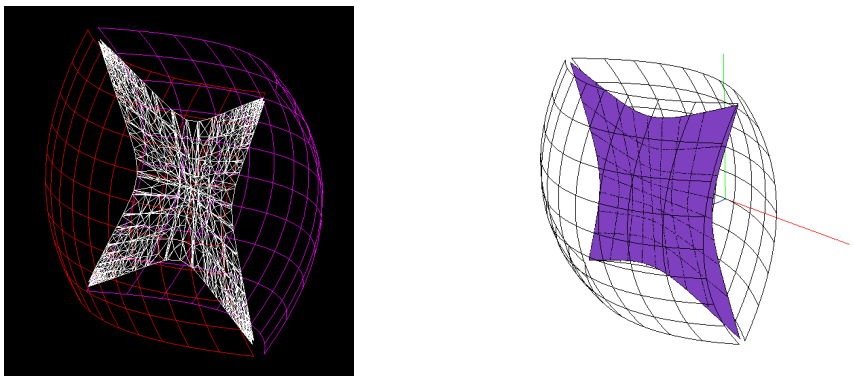


그림 2.9: 토폴로지 3 . IRIT(좌), 논문(우)

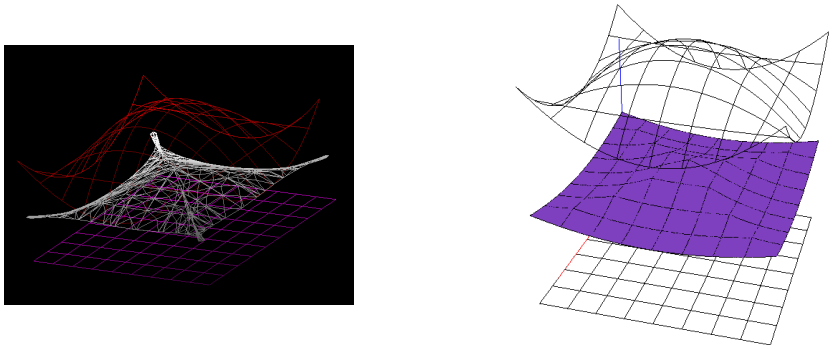


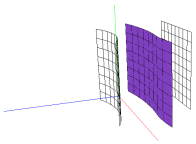
그림 2.10: 토폴로지 4 . IRIT(좌), 논문(우)

위와 같은 실험 결과들을 통해 볼 때, 기존의 소프트웨어와 유사한 형태의 bisector surface를 확인할 수 있다.

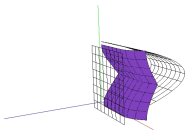
단 토폴로지 3, 4의 경우 기존에 비해 논문의 bisector surface 크기가 조금 더 큰 것을 확인할 수 있다. 이는 bisector surface를 계산하는 알고리즘의 차이에 기인한다. 기존 소프트웨어는 bisector surface를 계산할 때 sphere의 두 접점이 모두 각 곡면에서 boundary 상에 없다고 가정한다. 하지만 본 논문에서는 접점이 boundary 상에 있어도 구할 수 있도록 구현하였다. 따라서 본 논문에서 얻은 bisector surface가 기존의 것을 포함하는 곡면임이 자명하다.

2.5.2 시간 측정

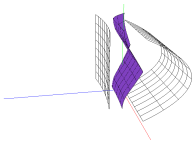
시간 측정도 다양한 상황을 가정하고 수행하였다. 기본적인 상황 3개와 복잡한 상황 2개에 대해 실험하였고, 그 결과는 다음과 같다.



(a) Base cases 1



(b) Base case 2

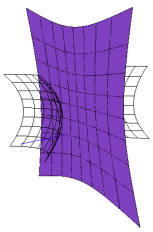


(c) Base case 3

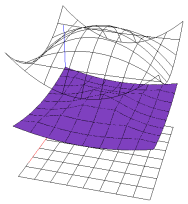
그림 2.11: Base cases

	Case 1	Case 2	Case 3
Gross time	16ms	38ms	48ms
Algorithm time	3ms	25ms	35ms

기본적인 상황에서는 실시간으로 알고리즘이 작동하는 것을 확인할 수 있다. 하지만, GPU를 사용하면서 데이터 이동에 걸리는 시간 등 기타 시간까지 고려하면 전체 시간은 더 느린 것을 확인할 수 있다. 두 곡면이 함께 움직이는 등 상황이 복잡해질 수록 더 많은 시간이 걸리는 양상도 보인다.



(a) Complex case 1



(b) Complex case 2

그림 2.12: Complex cases

	Case 1	Case 2
Gross time	137ms	280ms
Algorithm time	120ms	260ms

상황이 복잡해질 수록 시간이 많이 걸리는 것을 확인할 수 있다. 이와 같이 속도가 급격하게 느려지는 이유는 다음과 같다.

1. 베지에 곡면의 형태가 복잡해질 수록, 그것의 BVH는 커진다. 이는 안정적인 오차 수준을 유지하기 위해 곡면을 계속 분할하면서 생기는 결과이다. 따라서 BVH로 충돌 감지하는 데에 걸리는 시간이 길어진다.

2. GPU에서는 CPU만큼 유연한 자료 구조를 활용할 수 없다. 따라서 알고리즘을 활용하는 데 CPU보다 큰 제약이 생긴다. 이 한계는 베지에 곡면의 형태가 복잡해질수록 성능을 크게 저하시킨다.

제 3 장 결론

보로노이 다이어그램과 medial axis는 다양한 분야에서 그 중요성을 인정받고 있다. 평면 상에서 이러한 기하학적 구조를 구하기 위한 시도는 많이 이루어졌다. 하지만 3차원 공간에서의 보로노이 다이어그램과 medial surface를 구하기 위한 연구는 그에 비하면 많이 이루어지지 않았다.

본 논문에서는 3차원 공간 상에서 위와 같은 기하학적 구조를 구하기 위한 시도의 일환으로 2개의 3차 베지에 곡면에 대한 bisector surface를 효율적으로 구하는 알고리즘을 구현하였다. 이 알고리즘은 하나의 곡면을 주곡면으로 삼아 여러 개의 정점을 선택한 후, 각 정점에서 maximal sphere를 찾는다. 그리고 maximal sphere의 중점을 이용하여 bisector surface의 토폴로지를 결정한다. 알고리즘을 구현한 결과 많은 경우 안정적으로 bisector surface를 구할 수 있었고, 다양한 기법을 통해 속도를 향상시킬 수 있었다. 추후에는 알고리즘을 개선하여 보로노이 다이어그램과 medial surface를 구하기 위해 연구할 예정이다.

참고문헌

[1] Franz Aurenhammer. Voronoi diagrams—a survey of a fundamental geometric data structure. *ACM Computing Surveys (CSUR)*, 23(3):345–405, 1991.

[2] Harry Blum. A transformation for extracting new descriptors of shape. *Models for Perception of Speech and Visual Forms, 1967*, pages 362–380, 1967.

[3] Hyeong In Choi, Sung Woo Choi, Hwan Pyo Moon, and Nam-Sook Wee. New algorithm for medial axis transform of plane domain. *Graphical Models and Image Processing*, 59(6):463–483, 1997.

[4] Gershon Elber and Myung-So Kim. The bisector surface of rational space curves. *ACM Trans. Graph.*, 17(1):32–49, January 1998.

[5] Gershon Elber and Myung-Soo Kim. Computing rational bisectors. *IEEE Computer Graphics and Applications*, (6):76–81, 1999.

[6] Gershon Elber and Myung-Soo Kim. A computational model for nonrational bisector surfaces: Curve-surface and surface-surface bisectors. In *Geometric Modeling and Processing 2000. Theory and Applications. Proceedings*, pages 364–372. IEEE, 2000.

[7] Daniel Filip, Robert Magedson, and Robert Markot. Surface algorithms using bounds on derivatives. *Computer Aided Geometric Design*, 3(4):295–311, 1986.

[8] Jaewook Lee, Yong-Jun Kim, Myung-Soo Kim, and Gershon Elber. Efficient voronoi diagram construction for planar freeform spiral curves. *Computer Aided Geometric Design*, 43:131–142, 2016.