이화여자대학교 대학원
2013학년도
석사학위 청구논문

구체기반의 광선투사를 이용한 점집합의 부호거리 계산방법

컴퓨터공학과
신 혜 진
2014

구체기반의 광선투사를 이용한 점집합의 부호거리 계산방법

이 논문을 석사학위 논문으로 제출함

2013년 12월

이화여자대학교 대학원

컴퓨터공학과 신 혜 진

신 혜 진의 석사학위 논문을 인준함

지도교수 김 영 준 _____

심사위원 조 동 섭 _____

이 상 호 _____

김 영 준 _____

이화여자대학교 대학원

목 차

그 림 목 차

표 목 차

논 문 개 요

최근 들어 복잡한 점 집합을 다루는 장비의 사용이 증가하는 추세에 따라, 점집합을 재구성하여 출력하는 방법에 대한 필요성이 높아지고 있다. 재구성을 위하여 주로 사용되는 방법 중 하나는 점집합에 대한 부호거리장을 구하는 것이다. 하지만 점집합은 내·외부 영역 판정이 불가능하므로 부호거리를 직접 계산할 수 없으며, 영역 판정에 필요한 곡면을 추출하기 위해서는 복잡한 연산을 필요로 한다. 또한, 점집합에 데이터가 존재하지 않는 결함 영역이 있을 경우 다양한 계산 오류의 원인이 될 수 있으므로 이를 반드시 수정하여야 한다.

본 논문에서는 구체기반의 광선투사를 사용하여 점집합에 대해 부호거리를 구하는 방법을 제시한다. 제시하는 방법은 가상의 표면을 정의하여 점집합의 내·외부 영역을 판정하는 비교적 단순한 연산을 통하여 부호거리를 구할 수 있다. 점집합의 내·외부 영역을 판단하기 위해서 점집합을 이용하여 다수의 구체를 형성하고, 가상의 표면을 정의한다. 이후 점집합의 방향으로 광선을 투사하여 서로 다른 두 표면과 충돌할 경우 검사 지점을 점집합의 외부에 있다고 판단한다. 그렇지 않을 경우, 검사 지점은 점집합의 내부에 있다고 판단한다. 본 논문에서는 광선의 방향으로 존재하는 표면의 개수에 대한 판정을 위하여 광선과 충돌하는 구체를 활용하였다. 광선과 충돌하는 구체 중 가장 가까운 거리에 위치한 구체와 가장 먼 거리에 위치한 구체를 찾아 그 위치와 정점 법선 벡터를 비교하는 것으로 두 구체가 같은 표면 상에 위치하는 지 판별할 수 있다. 또한, 구체의 반지름을 증가시키는 것으로 구멍이 존재하지 않는 가상의 표면을

정의하여, 점집합의 결함 영역으로 인하여 발생할 수 있는 계산 오류를 방지하였다. 거리계산 및 충돌검사의 속도 향상을 위하여 구체로 구성된 경계 볼륨 계층구조를 구축하였으며, 정확한 부호거리를 구하기 위하여 부호와 거리를 각자 계산하는 방법을 채택하였다.

부호거리의 집합인 부호거리장은 점집합을 삼각형 메쉬 모델로 재구성하는 과정에 사용할 수 있다. 본 논문에서 제안하는 알고리즘의 정확성을 확인하기 위하여 부호거리장을 구축하고 이를 가시화하였으며, 마칭 큐브 알고리즘을 사용하여 부호거리장으로부터 메쉬 재구성 결과를 구현하였다. 실험 결과, 결함 영역이 존재하는 점집합 데이터를 원본으로 한 실험에서도 구멍이 발생하지 않는 고품질의 메쉬 모델을 얻을 수 있음을 확인하였다.

I. 서 론

A. 연구배경

1. 점집합

점집합(point set)은 레이저 스캐너 등의 장비에서 추출된 3차원 기하 정보를 가지는 자료구조이다. 점집합은 점에 대한 정보만을 가지고 있으므로 대량의 정보를 포함하는 모델을 비교적 작은 용량으로 저장할 수 있다는 특징을 가진다. 하지만 점집합은 별도의 위상 정보를 가지고 있지 않기 때문에, 이를 일반적인 목적으로 직접 사용하기는 어렵다[1]. 예컨대, 가시화 목적으로 사용하고자 한다면 표면 정보를 가지는 삼각형 메쉬와 같은 3차원 형상으로 재구성해야 한다.

점집합을 3차원 형상으로 표현하기 위해서는, 점과 점 사이의 여백 공간을 자료의 특성에 따라 해석할 필요가 있다. 대표적인 방법으로 삼각분할법(triangulation)을 사용하여 삼각형 메쉬를 구성하는 연구가 진행되었으며, MLS surface 기법 및 곡면요소(surfle) 등을 통하여 메쉬를 이용하지 않고 부드러운 곡면을 구성하는 방법 등이 연구되었다[2][3][4][5]. 이러한 방법들은 점집합에 존재하는 잡음을 제거하거나 교정하여 부드러운 곡면을 만들 수 있는 것으로 알려져 있다. 다만, 점집합 내에 점이 존재하지 않는 결함 영역이 있다면 해당 영역에 대한 곡면을 표현할 경우 기존의 방법들로는 결함을 보정하기 쉽지 않다[6]. 따라서 이러한 결함 영역을 효과적으로 수정할 수 있는 방법이 필요하다.

2. 부호거리

부호거리(signed distance field)는 특정한 점으로부터 다각형 모델 표면까지의 최단 거리로, 해당 점이 모델의 내부 혹은 외부에 위치하는 지를 알 수 있는 척도 중 하나이다[7]. 어떤 점이 다각형 모델의 내·외부에 있는지를 알기 위해서는, 일반적으로 해당 점이 모델을 구성하는 다각형으로 둘러싸여(surround) 있는지를 확인하는 방법을 사용할 수 있다. 그 중 가장 널리 알려진 방법은 짝수-홀수법(even-odd rule)이다[8]. 어떠한 점으로부터 다각형 모델의 방향으로 광선을 투사하여 충돌하는 다각형의 개수가 짝수일 경우 해당 점은 다각형으로 둘러싸여 있지 않은 것으로, 모델의 외부에 존재한다고 판단한다. 그렇지 않다면 해당 점은 모델의 내부에 존재하는 것으로 판단한다. 그림 1-1의 (a)에서, 점 a로부터 광선을 투사하였을 때 해당 광선은 두 개의 다각형, 즉 짝수 개의 다각형과 충돌하므로 점 a는 모델 외부에 존재하는 것이다. 그림 1-1의 (b)는 다각형 모델 내부에 존재하는 점의 경우를 나타내고 있다. 점 b에서 투사된 광선은 한 개의 다각형과 충돌하므로 점 b는 모델의 내부에 존재한다고 판단할 수 있다.

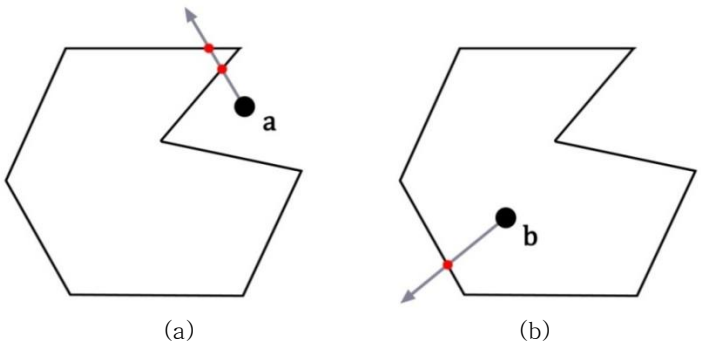


그림 1-1 짝수-홀수법을 사용한 다각형 모델의 내·외부 판단

(a) 점이 다각형 모델의 외부에 위치하는 경우 (b) 점이 다각형 모델의 내부에 위치하는 경우

해당 점이 다각형 모델 외부에 존재할 경우 부호거리는 양수가 되며, 내부에 존재할 경우 음수가 된다. 만약 특정 지점의 부호거리가 0이거나, 혹은 0에 가까운 경우 해당 지점은 곡면이 통과하는 지점이라고 할 수 있다.

일반적으로 다각형 모델을 구성하는 정점의 위치와 표면의 정점 벡터를 이용하여 부호거리를 계산한다[7]. 다만, 다각형 모델의 경우 모델을 구성하는 삼각형의 교차(self-intersection), 삼각형의 부재로 인하여 발생하는 구멍(hole) 등의 결함 요소가 존재할 수 있다. 이는 모델의 내부 및 외부 지점을 정확하게 판별할 수 없게 하기 때문에, 입력 데이터의 결함 요소를 고려하여 부호거리를 구하는 것이 중요하다.

부호거리의 집합인 부호거리장(signed distance field)은 공간을 3차원 그리드 형태로 분할하여, 분할된 공간인 각 셀의 정점들이 해당 정점으로부터 다각형 모델의 표면까지 이르는 부호거리를 저장하는 형태를 가진다[7]. 이를 이용하여 다각형 집합의 내부 및 외부 영역을 알 수 있기 때문에 메쉬 재구성(mesh reconstruction)에 많이 이용되고 있다.

하지만 점집합의 경우 점 사이에 별다른 연결 관계가 존재하지 않으므로 점집합에 대하여 바로 부호거리를 구하는 것은 불가능하며, 점집합의 내부와 외부지점을 판별할 수 있어야 한다. 대표적인 판별법으로 각 점의 정점 법선 벡터를 사용하여 추정 부호거리를 구하고, 해당 값으로부터 점집합의 제로셋(zero set)을 정의하는 방법이 제안되었다[9]. 또한, 점집합으로부터 외곽선을 추출하여 이를 내부 및 외부 지점 판별에 이용하는 방법 등이 존재한다[10].

한편 다각형 모델의 결함요소와 마찬가지로, 점집합에 데이터가 전혀 존재하지 않

는 결함 영역이 있을 경우 점집합의 내부 및 외부 지점을 정확히 판단할 수 없는 지점이 발생하고, 이는 부호거리의 정확한 계산을 저해하는 요인이 된다[6]. 이러한 문제를 수정하기 위해서 정점 벡터를 이용하여 결함 영역을 보정하고 내부 및 외부 위치를 판단하는 기법 등이 제시되었으나, 점집합 전체에 대한 복잡한 연산을 거쳐야 하는 단점을 가지고 있다[11].

B. 연구 목적 및 내용

본 연구에서는 점이 존재하지 않는 결함 영역을 가진 점집합에서 비교적 단순한 연산을 거쳐 부호거리를 구하는 것을 목적으로 한다. 이를 위하여, 점집합을 사용하여 다수의 구체를 생성하고 이를 가상의 표면으로 가정하여 점집합의 내·외부 위치를 판별하는 방법을 제시한다. 그림 1-2는 주어진 점 P_0 이 점집합의 내·외부 위치인지를 판별하는 과정을 나타내고 있다. (a)와 같은 경우 점집합에 별도의 표면이 정의되어 있지 않아 점 P_0의 위치를 정의할 수 없으므로, (b)와 같이 점집합을 사용하여 구체를 형성하고 가상의 표면을 정의한다. (c)와 같이 검사 지점으로부터 점집합의 방향으로 광선을 투사하여, 광선의 방향으로 얼마나 많은 가상의 표면이 존재하는 지를 검사한다. (d)는 광선을 투사하여 충돌하는 구체를 검출하는 과정을 나타내고 있다. 광선은 점집합의 방향으로 투사되어 가상의 표면을 구성하는 구체와 충돌하므로, 충돌하는 구체들의 위치 및 중심점의 법선 벡터 방향을 조사하는 것으로 광선과 충돌하는 가상 표면의 개수를 파악할 수 있다.

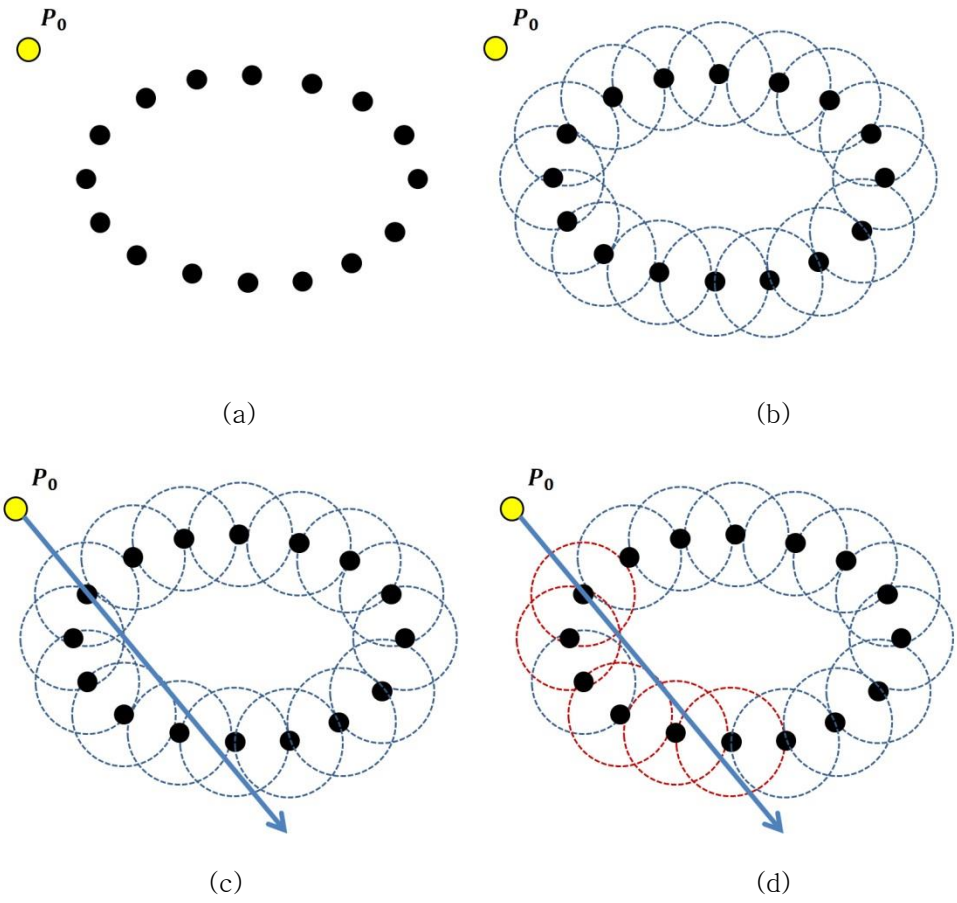


그림 1-2 점집합의 내·외부 위치를 판별하는 과정

(a) 점집합 원본 데이터 (b) 점집합으로부터 구체를 생성한 형태

(c) 주어진 점으로부터 광선을 투사 (d) 광선과 충돌하는 구체의 검출

제안하는 방법은 검사 지점으로부터 가장 가까운 구체와 가장 멀리 위치한 구체를 비교하여 이들이 같은 표면상에 존재하는 구체인지를 확인한다. 만일 두 구체가 일정한 거리 내에 위치하고, 중심점의 법선 벡터의 방향이 서로 같다면 구체들이 같은 표면 상에 위치하는 것이고, 이는 광선과 하나의 가상 표면만이 만나는 것을 의미하므

로 검사 지점이 점집합의 내부에 있다고 할 수 있다. 만약 그렇지 않다면, 검사 지점의 부근에는 두 개의 서로 다른 표면이 존재하는 것으로, 검사 지점이 점집합의 외부에 있다는 것과 같은 의미를 가진다.

특정한 지점으로부터 점집합까지의 부호거리를 계산하기 위해서는 해당 지점과 점집합 사이의 최단 거리를 계산하고, 해당 지점이 점집합의 내·외부에 존재하는 지를 검사하여 그 결과에 따라 거리 값에 부호를 붙이면 된다. 제시한 방법은 비교적 단순한 연산을 거쳐 부호거리를 계산할 수 있다는 장점을 가진다. 또한 점집합의 결함 영역 및 조밀한 분포에 대응하는 정확한 부호거리 계산이 가능하다. 본 연구에서는 광선과 충돌한 구체들을 전부 조사하지 않고 검사 지점으로부터 가장 가까운 구체와 가장 멀리 위치한 구체만을 비교함으로써 연산 과정을 단순화하였으며, 점을 감싸는 구체의 반지름을 조절하는 것으로 결함 영역이 존재하는 점집합에 대해서도 정확한 내·외부 지점의 판정이 가능함을 확인하였다. 두 구체의 정점 법선 벡터를 비교하는 과정을 추가하여 조밀하게 분포된 점집합의 경우에도 정확한 내·외부 지점의 판정이 가능하다.

제안한 부호거리 계산 방법의 정확도를 검증하기 위하여, 부호거리장을 계산하여 이를 가시화하였다. 또한, 부호거리장을 입력 데이터로 하여 마칭 큐브(Marching cube) 알고리즘을 사용한 메쉬 재구성을 실험하였다[12]. 실험 결과, 결함 영역을 가지지 않은 점집합과 결함영역을 가진 점집합 모두 정확한 부호거리장 계산이 가능하였으며, 높은 품질의 메쉬 재구성 결과를 얻을 수 있었다. 특히 결함 영역을 가진 점집합의 경우 점을 감싸는 구체의 반지름을 크게 하는 것으로 결함 영역의 수정이 이루어진 부호거리장의 계산과 매끄러운 곡면의 메쉬 재구성이 가능한 것을 확인하였다.

본 논문의 구성은 다음과 같다. 2장에서는 점집합을 다루는 방법과 다양한 입력 데이터에서 부호거리를 구하는 방법에 대한 관련 연구를 살펴본다. 3장에서는 점집합을 입력으로 받아, 구체를 이용하여 가상의 표면을 형성하고 내부 및 외부 지점을 판정하여 부호거리를 구하는 과정에 대하여 설명한다. 4장에서는 3장의 내용을 토대로 계산한 부호거리장의 가시화 결과 및 마칭 큐브 알고리즘을 사용한 메쉬 재구성 결과를 제시한다. 마지막으로 5장에서는 본 연구의 성과 및 의의를 논하고, 향후 연구에 대한 방향을 기술한다.

Ⅱ. 관련 연구

A. 점집합

점집합은 점의 위치, 정점 법선 벡터 등 점에 관한 기하 정보만을 포함하고 있기 때문에, 효과적인 사용을 위해서는 점집합을 연결하는 삼각형 메쉬 혹은 곡면 등을 생성하여 3차원의 형상으로 나타낼 필요가 있다[1]. 일반적으로 점집합 전체를 연결하는 삼각형 메쉬를 구성하는 방법이 널리 알려져 있으며, 보편적으로 사용되어 왔다[2]. 한편, 점집합으로부터 매끄러운 곡면을 추출하기 위하여 점과 점 사이에 지역적인 표면을 정의하는 방법들이 제시되어 왔다. Levin은 MLS Surface 기법을 통하여 점집합에서 지역적 평면을 정의하고 이를 통하여 매끄러운 곡면을 계산하는 방법을 제시하였다[3]. Alexa *et al.*은 Levin의 방식을 이용하여 점집합에서 메쉬를 사용하지 않는 곡면을 구성하는 알고리즘을 발표하였다[5]. Hanspeter *et al.*은 점과 정점 법선 벡터로 이루어진 곡면요소인 Surfel을 제안하여, 점집합으로부터 곡면을 도출하였다[4]. 해당 연구들은 점집합 내의 잡음을 교정하여 매끄러운 곡면을 구성할 수 있다는 장점을 가지고 있으나, 점집합 내에 점이 존재하지 않는 결함 영역이 있을 경우 해당 결함 영역을 수정하기 쉽지 않다는 단점을 가진다. 이러한 이유에서, 매끄러운 곡면 구성을 위해서는 결함 영역에 대한 수정이 반드시 필요하다. 모델의 전체적인 형태로부터 알맞은 영역을 찾아 복사하여 결함 영역에 붙여넣는 것으로 결함을 가진 점집합의 외관정보를 수정하는 방법이 발표되었다[6]. 다만, 제안된 방법은 점집합의 추출 밀도(sampling density)에 따라 영향을 받을 수 있다.

점집합은 적은 용량으로 고해상도의 모델 정보를 저장할 수 있다는 장점을 가지고 있으나 별도의 기하 정보를 포함하지 않기 때문에, 점집합을 활용하기 위해서는 점집합의 특성에 대한 고려가 필요하다. 가령 점과 점 사이에는 빈 공간이 존재하므로, 광선을 쏘아 활용하거나 특정 지점과 점집합 사이의 최단 거리를 빠르게 검출하기 위해서 점집합으로부터 별도의 데이터를 생성할 필요가 있다. 이러한 문제를 해결하기 위해, 점집합을 사용하여 경계 구(bounding sphere) 및 경계 볼륨 계층구조(bounding volume hierarchy)를 생성하여 활용하는 방법들이 제시되었다. Bart Adams *et al.*은 점집합을 사용하여 다수의 경계 구를 생성하고 이에 대한 광선 추적(ray tracing)기법을 제안하여, 별도의 표면이 정의되지 않은 점집합에 대해서도 자연스러운 조명을 사용한 렌더링이 가능하게 하였다[13]. 점집합과 광선과의 충돌 계산을 위하여 임의로 정의한 표면 부근의 점들을 중심으로 경계 구를 형성하였으며, 광선과 구체가 충돌하는 지점을 검출하는 과정을 통하여 점집합에 광선추적법을 적용하였다. Lee and Kim은 점집합에 대한 햅틱 인터랙션 지점(haptic interaction point)을 찾기 위하여 PSS(Point Swept Sphere) 기반의 경계 볼륨 계층구조를 형성하였으며, 경계 볼륨 계층구조를 사용하여 검사 지점으로부터 점집합 상에 존재하는 가장 가까운 점을 빠르게 검출하는 방법을 제안하였다[14].

B. 부호거리

부호거리는 컴퓨터 그래픽스 및 기하 모델링 분야에서 널리 사용되고 있는 자료구조이다. 부호거리장은 특정 지점이 모델의 표면 내·외부에 있는지를 측정하는 부호

거리의 집합을 뜻하며, 모션 플래닝(motion planning), 다물체동력학해석(multi-body dynamics), 변형체(deformable objects) 등에 사용가능하며, 대표적으로 메쉬 재구성(mesh reconstruction)에 유용하게 사용되어 왔다[15][16][17][18]. 그 중, 부호거리장을 이용하여 점집합을 메쉬 모델로 재구성하는 방법은 점집합의 사용이 증가함에 따라 더욱 주목 받고 있는 추세이다.

다각형 모델에 구멍, 겹침, 불규칙한 배열 등의 결함 요소가 존재할 경우 재구성 과정에서 모델의 표면 및 내·외부 지점을 정확하게 판별할 수 없기 때문에 입력 데이터의 특성을 고려하여 부호거리를 구하는 것이 중요하다. 다각형의 결함 요소에 대응하는 부호거리를 구하는 방법으로는 J. Andreas가 제시한 방법이 널리 알려져 있다. 제안된 방법은 삼각형 메쉬 모델에서 삼각형의 정점 및 각 정점의 법선 벡터를 이용하여 부호거리를 구하는 것이다[19]. 이후 연속성을 가지지 않은 삼각형 메쉬에 대해 주변 면의 법선 벡터 및 인접면과의 각도를 이용하여 Angle Weighted Pseudonormal을 구하고, 이를 부호 거리 계산에 활용하는 방법이 발표되었다[20].

별도의 표면이 정의되지 않은 점군으로부터 부호거리를 계산하는 방법은 Hoppe가 먼저 제안하였다[9]. 점군에는 별도의 표면이나 경계가 존재하지 않으므로, 점군을 구성하는 점의 정점 법선 벡터를 이용하여 부호를 결정하였다. P. Mullen *et al.*은 거친 점집합으로부터 부호가 없는 거리장(unsigned distance field)을 먼저 구한 다음, 모델의 외곽선을 뜻하는 ε-band를 구하여 거리에 대한 부호를 따로 계산하는 방법을 제안하였다[21]. 제안된 방법은 모델의 외곽 형태를 구한 다음 여러 검사 지점에서 외곽 형태를 향하여 다수의 광선을 투사하고, 광선과 외곽 형태선이 충돌하는 횟수를

측정하여 해당 검사 지점이 모델의 내부에 있는지를 측정하는 방식이다. 다각형 모델에서 내·외부 지점을 찾기 위해 사용하는 짝수-홀수법과 마찬가지로, 광선과 외곽 형태선의 충돌 횟수가 짝수인 경우 해당 지점은 점집합의 외부에 있으며, 홀수인 경우 해당 지점은 점집합의 내부에 있다고 판단한다. 제안된 방식의 경우 점 집합이 가진 구멍, 외부자(outlier) 등에 대응하는 정확한 부호거리장의 계산이 가능하다고 알려져 있으나, 점 집합 전체의 밀도, 연결성 등의 특성을 고려하여 외곽선을 구성하는 복잡한 연산이 필요하다. F. Calakli는 점집합 내의 점들이 가진 정점 법선 벡터를 전이시키는 것으로 점집합으로부터 매끄러운 곡면을 추출하고 결함 영역을 보정하는 방법을 제시하였다[11]. 다만, 제시된 방법은 법선 벡터 전이를 위하여 각 점들이 가진 정점 법선 벡터 간의 연속성을 검사하는 과정을 요구한다.

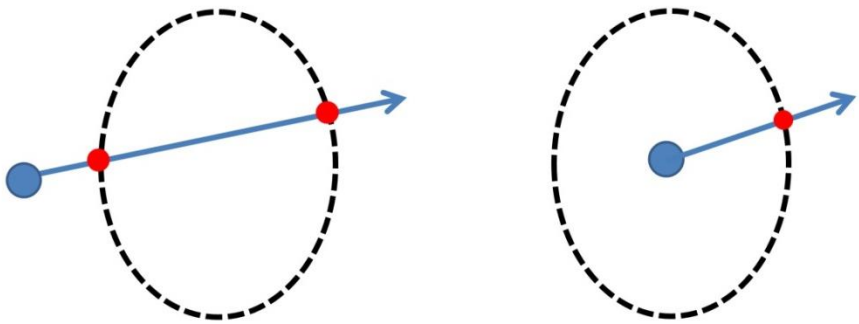
Ⅲ. 본 론

A. 점집합에서의 부호 판단

특정 지점에서 점집합에 대한 부호거리를 계산하기 위해서는 우선 해당 지점이 점집합의 내부 혹은 외부에 있는지를 확인하는 과정을 거쳐야 한다. 다각형 모델의 경우 표면을 기준으로 내부 및 외부 지점을 판별할 수 있으나, 점집합에는 내부 및 외부를 판별할 수 있는 명시적 표면이 존재하지 않으므로 위치 판별을 가능케 하는 별도의 기준을 마련할 필요가 있다. 일반적으로 점집합은 각 점의 위치 정보 및 정점 법선 벡터 정보를 포함하고 있으므로 해당 정보들을 이용하여 점집합 전체를 잇는 곡면을 추출하고, 추출한 곡면을 기준으로 점짐합에 대해 정확한 내·외부 지점을 판별할 수 있음이 알려져 있다[4][11]. 그러나 이와 같은 과정은 점집합 전체에 대한 복잡한 연산이 필요하다는 문제를 가지고 있다.

본 연구에서는 점집합의 내·외부 위치를 보다 간단한 연산을 통하여 판별하기 위해 점집합 내의 각 점이 가진 위치 정보와 정점 법선 벡터 정보를 활용하였다. 주어진 점으로부터 광선을 투사하여, 광선의 방향에 존재하는 표면의 개수를 세는 것으로 점집합에 대한 내부 및 외부 지점을 판단하였으며 그 결과에 따라 부호거리를 계산하였다. 부호거리를 계산하는 과정에서 표면의 개수를 세기 위하여 표면 부근에 위치한 점들의 거리 및 정점 법선 벡터의 방향을 사용하였으며, 이에 대한 자세한 활용 방법은 C절에서 설명하도록 한다. 그림 3-1은 점집합에서 내부 및 외부를 판단하는 과정으로, 특정한 검사지점에서 광선을 쏘아서 점선으로 그려진 가상의 표면과의 충돌을

검사하는 상황을 나타내고 있다. (a)에서, 화살표로 그려진 광선의 방향으로 서로 다른 두 개의 표면이 존재하고 있다면, 검사 지점은 가상으로 정의된 표면의 외부에 존재하는 것이다. 이 경우 검사 지점으로부터의 부호거리는 양수가 된다. 그렇지 않을 경우 가상의 표면 내부에 있다고 판단할 수 있다. (b)는 광선의 방향으로 하나의 표면만이 존재하는 경우이다. 이 때, 검사 지점의 부호거리는 음수가 된다.



(a) (b)

그림 3-1 점집합에서의 부호 판단

(a) 검사 지점이 표면의 외부에 존재하는 경우 (b) 검사 지점이 표면의 내부에 존재하는 경우

본 연구에서는 점집합 내 점들의 위치 및 분포를 이용하여 구체를 생성하는 것으로 점집합에 가상의 표면을 정의하였으며, 내부 및 외부 위치 판별을 위하여 검사 지점으로부터 점집합의 방향으로 광선을 투사하였다. 투사한 광선에는 점집합을 이용하여 생성한 구체들이 충돌하게 된다. 이후 광선에 충돌한 구체들이 서로 같은 표면에 존재하는 지를 검사하는 과정을 통하여 검사 지점이 점집합의 내부에 존재하는 지의 여부를 판별하였다. 또한, 검사 과정에서 표면 부근에 존재하는 점들의 거리 및 정점 법선 벡터를 활용하는 것으로 점집합에 대한 정확한 내·외부 위치 판단을 할 수 있음을 확인하였다.

B. 구체 생성을 통한 가상의 표면정의

별도의 표면이 정의되지 않은 점집합에서 정확한 내부 및 외부 위치 판별을 위하여, 가상의 표면을 먼저 정의할 필요가 있다.

제시하는 알고리즘은 점집합으로부터 가상의 표면을 정의하기 위하여, 정점 법선 벡터를 가진 점집합을 입력으로 받아 적절한 크기를 가진 다수의 구체를 생성한다. 각 점은 구체의 중심점이 되며, 각 점으로부터 가까운 위치에 존재하는 다른 점까지의 거리가 구체의 반지름이 된다. 하나의 구체는 하나의 점만을 중심점으로 포함하므로, 최종적으로 생성된 구체의 수는 점집합 내의 점의 개수와 동일하다.

본 논문에서는 점집합을 이용하여 다수의 구체를 생성하였으며, 이를 가상의 표면을 구성하는 요소로 정의하였다. 이를 이용하여, 검사 지점으로부터 점 집합의 방향으로 광선을 투사하고 충돌한 구체들을 통해 검사 지점의 부호를 결정할 수 있다. 해당 과정에 대해서는 본론의 C장에서 자세히 설명하도록 한다.

또한, 부호거리 계산을 위한 최단거리 검출 및 충돌검사를 빠른 시간 안에 수행하기 위하여 점집합으로부터 생성한 구체를 하향식으로 분할하고 경계 볼륨 계층구조를 구성하였다.

1. 구체 생성 및 반지름 선택

본 연구에서는 점 데이터가 존재하지 않는 결함 영역을 수정하기 위하여 점집합으로부터 다수의 구체를 생성하여, 해당 구체들을 가상의 표면을 구성하는 요소로 정의하였다. 만약 결함 영역으로 인하여 가상 표면에 구멍이 발생한다면, 이를 이용하여

점집합의 내·외부 위치를 정확히 판별하는 과정에서 오차가 발생할 수 있다. 그러므로 가상 표면을 정의하는 과정에서 의도하지 않은 구멍이 생기는 것을 방지할 필요가 있으며, 이를 위하여 가상의 표면을 정의하는 요소로 구체를 사용하였다. 다수의 구체를 생성하여 가상의 표면을 정의할 경우, 구체들의 반지름을 조절하는 것으로 가상 표면에 구멍이 발생하는 현상을 방지할 수 있다. 점집합을 사용하여 구체를 생성하는 과정은 다음과 같다.

집합 내의 점을 중심점으로 정하고, 점으로부터 가까운 다른 점과의 거리를 반지름으로 하여 다수의 구체를 생성한다. 구체의 반지름은 구체마다 모두 다를 수 있으며, 구체는 점 집합 내의 개수와 동일하게 생성된다. 점집합의 내부 및 외부 위치를 판별하는 과정에서 필요한 최단거리 검출 및 충돌검사를 빠른 시간 내에 수행하기 위해서, 구체의 위치와 반지름에 따라 경계 볼륨 노드 및 계층구조를 생성한다.

점집합에 데이터가 존재하지 않는 결함 영역이 존재할 경우, 구멍이 발생하여 내부 및 외부 위치가 정확하게 판별되지 않을 수 있다. 제시하는 알고리즘은 이를 방지하기 위하여 집합 내의 점들을 중심으로 적응적 반지름을 가진 구체를 생성하여, 이를 경계 볼륨 계층구조로 연결한다. 점 집합 내의 점들은 모두 구체의 중심점이 되며, 구체의 반지름은 각 점과 가까운 다른 점까지의 거리이다.

본 논문에서는 각 점과 k번째로 가까운 다른 점까지의 거리를 구체의 반지름으로 지정하여 사용하였다. 구체의 반지름은 각각의 중심점으로부터 k번째로 가까운 다른 점까지의 거리이므로 구체마다 서로 다른 반지름을 가진다는 특징이 존재한다. 구체의 반지름을 정하기 위해서는 먼저 각 점으로부터 k번째로 가까운 점의 위치를 구할

필요가 있는데, 점집합 내의 모든 점에 대한 각각의 근접점을 찾아야 하므로 각 점간의 거리를 비교하는 일반적인 방법을 사용할 경우 계산에 많은 시간이 소요된다는 문제가 존재한다. 이러한 문제를 해결하기 위하여, 점집합을 3개의 축을 사용하여 분할하고 각 점들을 거리에 따라 분류하여 저장하는 kd-tree를 이용하였다. k-dimensional tree라고도 불리우는 kd-tree는 공간을 초평면(hyperplane)을 사용하여 계속적으로 분할하는 트리 형태의 자료구조이다[21]. 트리를 생성하는 과정에서 점들은 각자의 위치에 따라 노드에 저장되므로 어떤 점으로부터 k번째로 가까운 점을 빠르게 찾을 수 있는데, 이를 k-Nearest neighbor search 기법이라 부른다. 공간을 분할하고 kd-tree를 형성하는 과정에서 각 점들은 위치에 따라 특정 노드로 배치된다. 따라서 어떤 점으로부터 k번째로 가까운 점을 찾고 싶을 때, kd-tree를 사용하면 빠르고 효율적인 검색이 가능하다. kd-tree를 이용한 k-nearest neighbor search 기법은 n개의 데이터를 사용할 시 평균적으로 O(log$n$)의 시간 복잡도 내에서 검색이 가능한 것으로 알려져 있다[21]. k-nearest neighbor search 기법을 활용하여 최근접점을 탐색하는 과정에서 k=0일 경우의 최근접점은 자기 자신이 되며, k=1일 경우의 최근접점은 현재 점과 가장 가까운 다른 점이 된다. 현재 점으로부터 k번째로 가까운 다른 점까지의 거리를 반지름으로 지정하므로, k의 값이 커질수록 점을 감싸는 구체의 반지름 역시 커진다.

충분히 큰 반지름을 가진 구체들은 서로 중첩하여 구멍이 존재하지 않는 가상의 표면을 형성한다. 그림 3-2는 구체를 사용하여 점집합에 가상의 표면을 형성한 상태를 나타낸다. 그림 속의 점들은 타원 형태를 이루는 점집합을 간략히 나타낸 것이며 구체가 형성된 모양은 점집합에서 내부 및 외부 위치를 판별하기 위한 가상의 표면 모

양으로, 점집합이 실제 3차원으로 형상화 된 형태와는 같지 않다. 부호거리를 정확하게 계산하였을 경우 부호거리를 사용하여 점집합으로부터 추출할 수 있는 실제 곡면의 형태는 그림 3-3에서 각 점을 연결하는 점선과 같이 나타나 있다. 이와 같이 점집합을 연결하는 곡면의 형태를 구하기 위하여, 그림 3-2와 같이 점집합을 사용하여 가상의 표면을 정의하고 부호거리를 계산하는 과정이 필요하다.

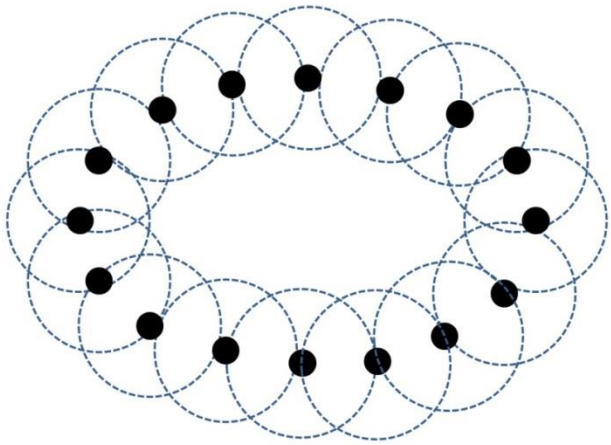


그림 3-2 다수의 구체가 형성하는 가상의 표면

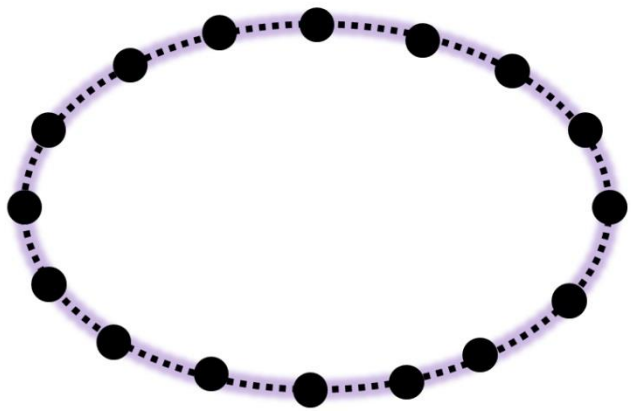


그림 3-3 점집합으로부터 추출되는 실제 곡면의 형태

점집합을 사용하여 구체를 생성할 때, 점집합에 대한 정확한 내·외부 판단을 위하여 점집합의 가상 표면에 빈 공간이 생기지 않도록 할 필요가 있다. 빈 공간이 생길 경우 광선을 이용한 내·외부 판별 시에 오차가 발생할 수 있기 때문이다. 그림 3-4는 점집합의 점 분포 특성으로 인하여 결함 영역이 발생한 상황을 나타내고 있다. 만약 어떤 지점으로부터 투사된 광선이 붉은 색으로 표시된 영역을 지날 경우 어떤 구체와도 충돌하지 않는다. 때문에 그림 3-4의 점집합은 그림 3-2의 점집합과 유사한 분포 형태를 가지고 있음에도 불구하고 특정 지점에 대한 점집합의 내·외부를 잘못 판정할 가능성이 존재한다. 이는 해당 점집합이 점이 균일하게 분포되지 않은 결함 영역을 가지고 있기 때문이다.

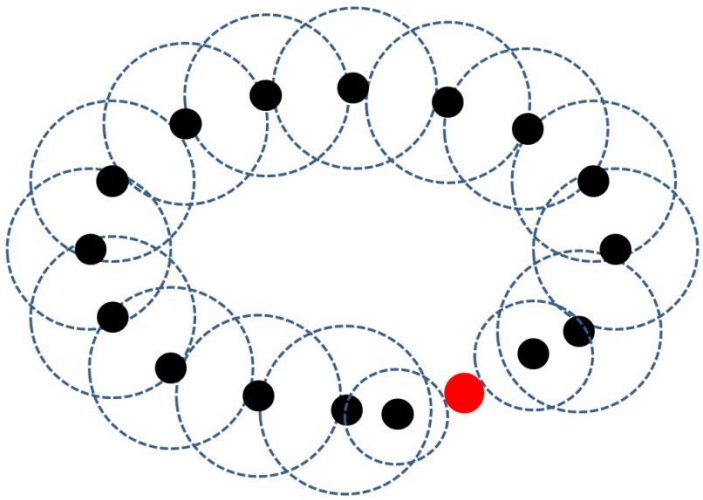


그림 3-4 결함 영역을 가진 점집합

잘못된 판정을 방지하기 위하여 결함 영역을 가진 점집합의 경우 k값을 임의로 증가시켜야 한다. 구체의 반지름을 크게 하여 구체가 점 집합을 충분히 덮을 수 있도록

하는 것이다. 구체의 반지름은 현재 점으로부터 k번째로 가까운 점과의 거리를 통하여 산정되므로, 각 구체의 반지름은 주변의 점 분포에 따라 그 크기가 달라질 수 있다.

2. 구체기반의 BVH 생성

본 연구에서는 검사 지점과 점집합간의 빠른 거리계산 및 충돌검사를 위하여 SSV(Swept Sphere Volumes) 생성을 위한 중심점과 구체를 중심 요소로 사용하는 PSS(Point Swept Sphere)와 유사한 구조를 경계 볼륨 노드로 채택한 계층구조를 사용하였다.

SSV란 일정한 반지름을 가진 구체와 중심 요소가 되는 도형의 민코우스키 합으로 이루어진 경계 볼륨(bounding volume)을 뜻한다[22]. 일반적으로 SSV는 그 형태에 따라 다음의 세 가지로 분류할 수 있다.

PSS(Point Swept Sphere) : 점을 중심 요소로 삼고 구체의 형태를 띄고 있는 BV로, 하나의 중심점과 구체의 반지름으로 표현할 수 있다. 저장 공간이 적고 충돌 검사에 빠르다는 장점을 가진다.

LSS(Line Swept Sphere) : LSS는 선(line segment)을 중심 요소로 가진다. LSS를 채택한 BV는 구체를 길게 늘려 끝이 둥근 기둥의 모양을 가지고 있으며, 중심점 및 구체의 반지름, 그리고 중심점을 지나는 선으로 표현된다.

RSS(Rectangle Swept Sphere) : RSS의 중심 요소는 3차원 상의 사각형이다. 끝이

둥근 상자와 같은 형태의 BV를 가지며, 사각형 및 중심점, 그리고 반지름으로 표현된다. 거리 계산에 가장 빠른 형태의 SSV로 알려져 있다.

그림 3-5는 왼쪽부터 차례대로 PSS, LSS, RSS의 경계 볼륨 형태를 나타내고 있다. 해당 그림은 Larsen *et al.*의 논문으로부터 발췌하였다[22].

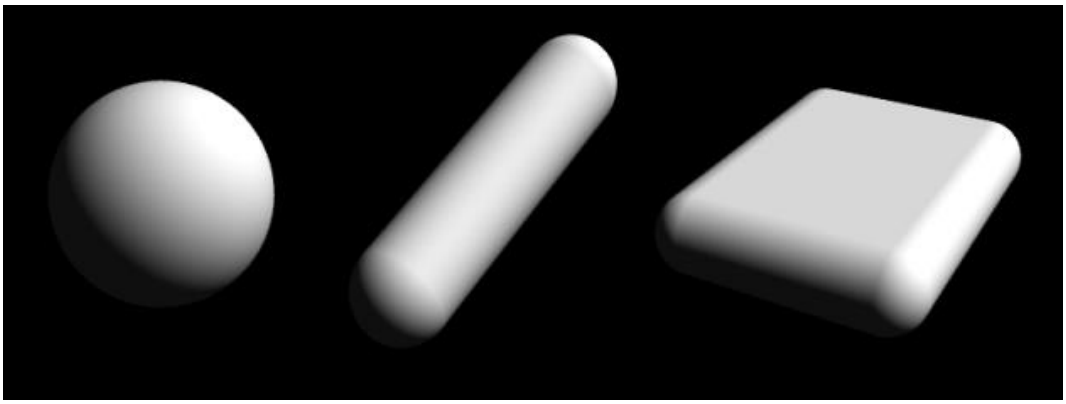


그림 3-5 SSV의 종류 및 형태 [22]

본 연구에서는 중심점과 구체의 형태를 가지고 있는 경계 볼륨인 PSS와 유사한 형태를 채택하여, 구체를 기반으로 한 경계 볼륨 계층구조를 사용하였다. PSS는 충돌검사에 가장 빠른 것으로 알려져 있으며, 중심점 및 구체를 이루는 반지름의 정보만을 필요로 하기 때문에 세 가지의 SSV 중 가장 적은 저장 공간만을 필요로 한다. PSS는 단말 노드(leaf node)인 하나의 구 안에 다수의 점을 포함할 수 있으며, 각 구체가 어떠한 점들을 포함할 것인지에 대한 계산을 필요로 한다. 본 연구에서 사용하는 경계 볼륨은 이와 같은 불필요한 계산 과정을 줄이기 위하여, 단말 노드인 하나의 구가 하나의 점만을 중심점으로 포함하도록 하였다. 즉, 경계 볼륨 계층구조의 단말 노드인 구체의 개수는 점집합 내의 정점의 개수와 동일하다.

경계 볼륨 계층구조의 형성 순서는 하향식으로, 근 노드(root node)가 가장 먼저 생성되고 단말 노드가 가장 마지막으로 생성된다. 근 노드의 중심점의 위치는 모든 구체의 중심점의 평균값을 이용하여 계산할 수 있으며, 근 노드의 중심점에서 가장 먼 거리에 위치하는 구체를 찾아 해당 구체를 완전히 감쌀 수 있는 거리를 반지름으로 채택한다. 근 노드는 모든 점집합 및 점을 감싸는 구체들을 포함하며, 단말 노드는 하나의 점과 해당 점을 중심으로 한 구체를 포함한다.

각 점들을 중심점으로 삼아 생성된 구체들은 중심점을 기준으로 한 구체의 위치 및 반지름의 크기에 따라 경계 볼륨 노드가 된다. 구체들의 중심점으로부터 경계 볼륨의 중심점이 되는 위치를 계산하고, 노드에 해당하는 구체들을 감쌀 수 있는 가장 작은 반지름을 찾아 새 자식 노드를 형성하는 과정을 거친다. 경계 볼륨 노드의 반지름을 찾는 과정은 최소 포함 구 문제(minimum covering sphere problem)를 통하여 해결할 수 있다[23]. 경계 볼륨 노드는 해당 영역의 구체 및 중심점을 전부 포함하므로, 노드의 반지름을 계산하는 과정에서 각 구체의 반지름을 반드시 고려해야 한다. 근 노드의 생성 이후 구체들의 중심점을 기준으로 공간을 이분할하여 재귀적으로 자식 노드를 형성한다. 마지막으로 하나의 구체 및 중심점만이 남게되면 남은 하나의 구체와 중심점은 자동으로 단말 노드가 된다. 단말 노드는 별도의 계산을 할 필요 없이 구체가 가진 중심점 및 구체의 반지름을 그대로 사용한다. 모든 단말 노드의 생성이 끝나면 최종적으로 트리 구조인 경계 볼륨 계층구조가 완성된다.

C. 부호거리의 계산

주어진 한 점으로부터 점집합까지의 부호거리를 구하는 과정은 다음과 같다.

검사지점으로부터 점집합 상에 존재하는 가장 가까운 점(최근접점)을 구한다. 검사지점으로부터 최근접점으로 향하는 벡터와 최근접점이 가진 정점 법선 벡터의 방향을 비교하여, 점집합에 가까운 방향으로 광선을 투사한다. 광선과 점집합 간의 충돌검사를 실시하여 광선과 충돌하는 구체를 전부 구한다. 다음으로 광선과 충돌한 구체들이 같은 표면 상에 위치하는 지를 알기 위하여, 광선과 충돌한 구체를 검사지점으로부터 가까운 거리 순으로 정렬하여 가장 가까운 구체와 가장 멀리 있는 구체 사이의 거리를 비교한다. 두 구체의 거리에 따라 광선의 방향으로 존재하는 표면의 수가 결정되고, 따라서 현재 검사지점이 점집합의 내부에 위치하는 지의 여부를 판별할 수 있다. 최종적으로 검사 지점으로부터 투사한 광선의 방향으로 최근접점까지의 거리를 측정하고, 점집합의 내부 및 외부 위치 판별 결과에 따라 부호를 계산하여 거리값에 붙이는 것으로 부호거리를 구할 수 있다.

1. 최근접점 검출

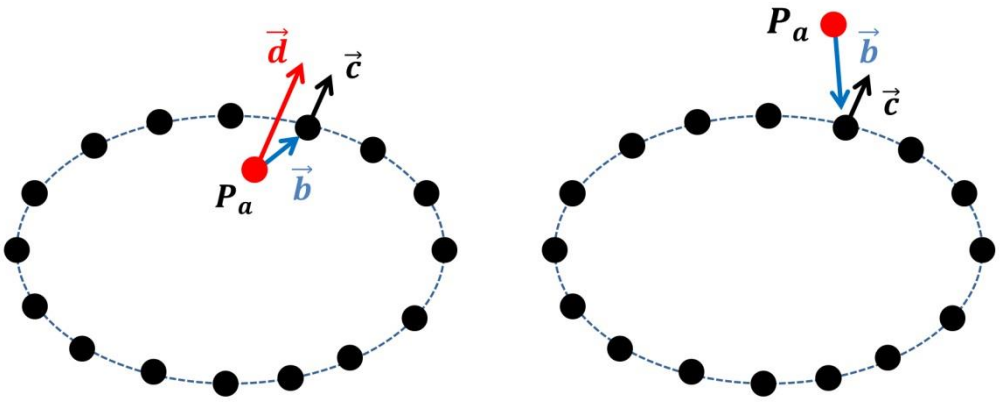
부호거리를 구하기 위해서는 우선 검사지점으로부터 점집합 상에 존재하는 가장 가까운 점, 즉 최근접점을 구해야 한다. 이는 집합 내의 최근접점을 이용하여 거리 및 부호를 계산하기 위함이다. 최근접점을 찾기 위해서는 검사 지점과 점집합을 사용하여 생성한 구체들간의 거리를 검사하여 그 중 가장 짧은 거리를 찾고, 그 거리에 위치하는 구체 및 중심점을 찾는 과정이 필요하다. 경계 볼륨 계층구조를 순회하는 과

정에서, 충돌 가능성이 없는 노드의 경우 탐색을 중지하고 충돌 가능성이 존재하는 노드의 경우에만 탐색을 계속하므로 효율적인 거리 탐색이 가능하다.

2. 광선투사 및 충돌검사

투사할 광선의 방향을 정하기 위하여, 집합 내 최근접점을 검출하고 검사지점으로부터 최근접점으로 향하는 벡터와 최근접점이 가진 정점 법선 벡터의 방향이 같은지의 여부를 검사한다.

점집합 내의 결함 영역으로 인하여 부호거리 계산 시 일어날 수 있는 계산 오류를 방지하기 위하여 만약 검사지점으로부터 최근접점으로 향하는 벡터와 최근접점이 가진 정점 법선 벡터의 방향이 같을 경우, 투사할 광선의 방향은 검사지점으로부터 최근접점의 정점 법선 벡터 방향과 같게 하였다. 그렇지 않다면, 검사지점으로부터 최근접점으로 향하는 벡터의 방향을 광선의 방향으로 지정하였다. 그림 3-6은 투사될 광선의 방향을 결정하는 과정을 나타내고 있다. (a)의 경우, 검사지점인 점 P_a로부터 최근접점으로 향하는 벡터 $\vec{b}$ 의 방향이 최근접점의 정점 법선 벡터인 $\vec{c}$의 방향과 같으므로, 최근접점의 정점 법선 벡터와 동일한 방향으로 광선을 투사하게 된다. 이 때 광선의 방향은 $\vec{d}$가 된다.



(a) (b)

그림 3-6 투사될 광선의 방향 결정

(a) 두 벡터의 방향이 서로 같은 경우 (b) 두 벡터의 방향이 서로 다른 경우

그림 3-6 (b) 의 경우, 검사지점인 점 P_a로부터 최근접점으로 향하는 벡터 $\vec{b}$의 방향과 최근접점의 정점 법선 벡터 $\vec{c}$의 방향이 서로 다르므로, 검사지점으로부터 최근접점으로 향하는 방향으로 광선을 투사한다. 검사지점으로부터 최근접점으로 향하는 방향 벡터 및 광선의 방향은 $\vec{b}$와 같이 나타낼 수 있다.

위와 같은 방법으로 정한 방향을 따라 광선을 투사하여, 광선과 점집합 사이 근접질의를 이용한 충돌검사를 실시한다. 광선과 경계 볼륨 계층구조와의 충돌검사 역시 최단거리 검사와 같은 방식으로 충돌 가능성이 존재하는 경계 볼륨 노드만을 탐색하나, 최단 거리에 위치한 구체를 찾아낸 경우 더 이상의 탐색을 진행하지 않는 최단거리 검사와 달리 충돌검사는 광선과 충돌하는 첫 구체를 검출한 이후에도 탐색을 계속하여 광선과 충돌하는 모든 구체를 찾아낸다. 경계 볼륨 노드 및 단말 노드인 구체

는 모두 하나의 중심점과 반지름을 가진 구체라는 공통점을 가지고 있기 때문에, 모든 충돌검사는 광선과 구체간의 충돌검사법을 사용하여 이루어진다.

충돌검사가 종료되면 광선과 충돌하는 구체의 중심점 및 각 구체의 반지름을 알 수 있다. 각 구체의 중심점들은 광선이 지나는 방향 혹은 그 방향에 가깝게 존재하는 점집합 내의 점들을 뜻한다.

3. 표면판정을 통한 부호정의

특정 지점이 점집합의 내부 혹은 외부인지를 판별하기 위해서는, 지점으로부터 광선을 쏘아 광선의 방향으로 얼마나 많은 가상의 표면이 존재하는 지를 확인하는 과정이 필요하다. 점집합을 이용하여 생성한 구체들이 가상의 표면을 구성하고 있으므로, 충돌검사를 통하여 얻은 구체들이 서로 같은 표면에 있는지를 검사하는 과정이 필요하다. 구체들이 서로 같은 표면에 존재한다면, 검사 지점으로부터 광선의 방향으로는 하나의 표면만이 존재한다고 할 수 있다. 만약 그렇지 않다면, 검사 지점으로부터 광선의 방향으로 두 개의 표면이 존재하는 것이다. 전자의 경우 검사 지점이 가상 표면의 내부에 위치한다고 판단할 수 있으며, 후자의 경우는 검사 지점이 가상 표면의 외부에 위치한다고 판단할 수 있다. 검사 지점이 가상 표면의 내부 혹은 외부에 있음을 알면, 그 결과에 따라 거리 값에 부호를 붙여 최종적으로 부호거리를 구할 수 있게 된다.

본 연구에서는 두 구체가 일정한 거리 안에 놓여 있거나 두 구체 내 중심점의 정점

법선 벡터 방향들이 동일할 때, 두 구체가 서로 같은 평면 위에 존재한다고 판단하였다. 이를 알기 위하여 광선과 충돌한 모든 구체를 검사하는 대신, 검사 지점으로부터 가장 가까운 구체와 가장 먼 구체를 비교하여 검사하였다. 그림 3-7은 검사 지점으로부터 광선을 쏘아 충돌하는 구체를 검출하는 과정을 나타내고 있다. 그림 상에서는 검사 지점인 점 P_a로부터 점집합 내의 최근접점인 점 P_b를 향하여 광선을 투사하는 상황을 가정하였으며, 광선의 방향은 $\vec{c}$와 같다. 광선과 충돌하는 구체는 구 S_a, 구 S_b, 구 S_c와 같다. 이 때 검사 지점 P_a와 가장 가까운 구체는 구 S_a이며. 가장 먼 구체는 구 S_c이다.

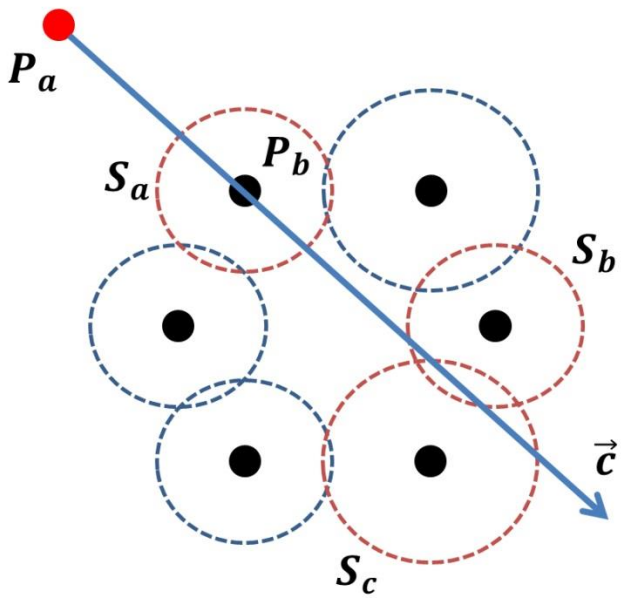


그림 3-7 광선과 충돌하는 구체

가장 가까운 구체와 가장 먼 구체가 같은 표면 상에 있다고 판단되면, 나머지 구체들도 모두 같은 표면 상에 존재한다고 할 수 있다. 만약 두 구체가 같은 표면 상에 존재하지 않는다고 판단될 경우, 검사 지점으로부터 광선의 방향으로 서로 다른 다수의 표면이 존재한다고 할 수 있다. 구체적인 검사 방법은 다음과 같다.

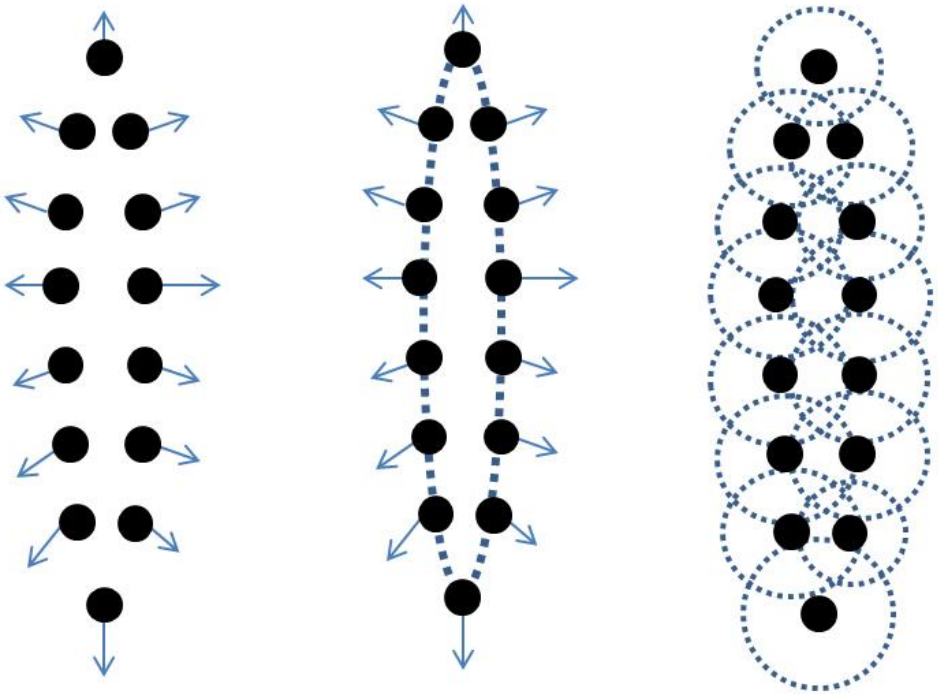
충돌검사를 통하여 얻은 구체들 중에서, 검사 지점으로부터 가장 가까이 위치한 구체와 가장 멀리한 구체를 찾고 두 구체의 거리를 비교한다. 두 구체의 거리는 가장 가까운 구체의 중심점 및 광선의 방향을 사용하여 초평면을 형성한 다음, 해당 초평면으로부터 가장 먼 구체의 중심점까지의 수직 거리를 구하는 것으로 계산할 수 있다. 수직 거리를 구하는 자세한 방법은 다음 절에서 자세히 설명하도록 한다. 두 구체간의 거리가 가장 가까운 구체의 반지름보다 크다면, 두 구체는 충분히 먼 거리에 위치하고 있으며 같은 표면 상에 있다고 볼 수 없다. 만약 두 구체간의 거리 D가 첫 번째 구체의 반지름 R_{first} 보다 크다면, 두 구체는 다른 표면 상에 존재한다고 판단한다. 그렇지 않다면, 두 구체의 중심점이 가진 정점 법선 벡터의 방향을 비교한다. 만약 같은 방향이라면, 두 구체는 같은 표면 상에 존재한다고 판단한다. 그렇지 않을 경우, 두 구체는 서로 다른 표면 상에 존재한다.

실험 결과, 비교 대상이 되는 거리 D와 가장 가까운 구체의 반지름 R_{first}을 비교할 경우 $D > R_{first} * 1.05$ 와 같은 값을 사용하였을 때 가장 정확한 결과를 얻을 수 있었다.

상기 언급한 방법으로 계산한 거리가 첫 번째 구체의 반지름보다 작을 때 두 구체는 같은 표면 상에 있다고 할 수 있다. 두 구체가 같은 평면 상에 존재한다는 것은

검사 지점이 모델의 내부에 있다는 것을 뜻하고, 서로 다른 평면 상에 존재한다는 것은 검사 지점이 모델의 외부에 존재한다는 의미를 가진다. 이러한 판정 방법은 다각형 모델을 대상으로 내부 및 외부 지점을 알기 위해 흔히 사용되는 짝수-홀수법을 점집합의 내·외부 판단에 적용한 결과이다. 제안하는 알고리즘은 충돌하는 구체들의 개수와 거리를 전부 집계하지는 않으며, 가장 가까운 구체와 가장 먼 구체만을 선택하여 두 구체가 같은 표면에 존재하는 지의 여부만을 검사한다.

점이 비교적 조밀하게 분포된 점집합에 대해서도 정확한 부호거리를 계산할 수 있도록, 두 구체의 거리가 일정 거리보다 가까울 경우 정점 법선 벡터의 방향을 비교하는 방법을 사용하였다. 좁은 폭으로 분포되어 있는 점집합의 경우 두 구체의 거리를 비교하는 방법 만을 사용하여 내·외부를 판별할 시 거의 대부분의 지점이 모델의 내부로 판단되어 표면이 제대로 정의되지 않는 경우가 발생하였다. 그림 3-8은 점이 긴 타원의 형태로 조밀하게 분포된 점집합의 예시를 나타내었다. (a)는 가까운 거리 안에 많은 점이 위치해 있으나 각 점의 정점 법선 벡터의 방향이 각기 다른 점집합의 예시를 나타내고 있다. 점집합의 곡면은 대개 정점 법선 벡터의 방향을 따라 형성되기 때문에, (a)의 점집합으로부터 생성되는 곡면은 (b)에서 각 점을 잇는 점선과 같은 형태를 가질 것이라고 예측할 수 있다. 한편, (a)의 점집합을 사용하여 형성된 구체는 (c)와 같은 형태를 가진다. (c)를 통해 알 수 있듯이 대다수의 구체가 중첩되거나 매우 가까운 거리 안에 위치하고 있으므로, 이러한 상태에서 구체들의 거리만을 고려하여 표면을 판별할 경우 거의 대부분의 점들을 같은 표면 상에 존재한다고 판단하기 쉽다.



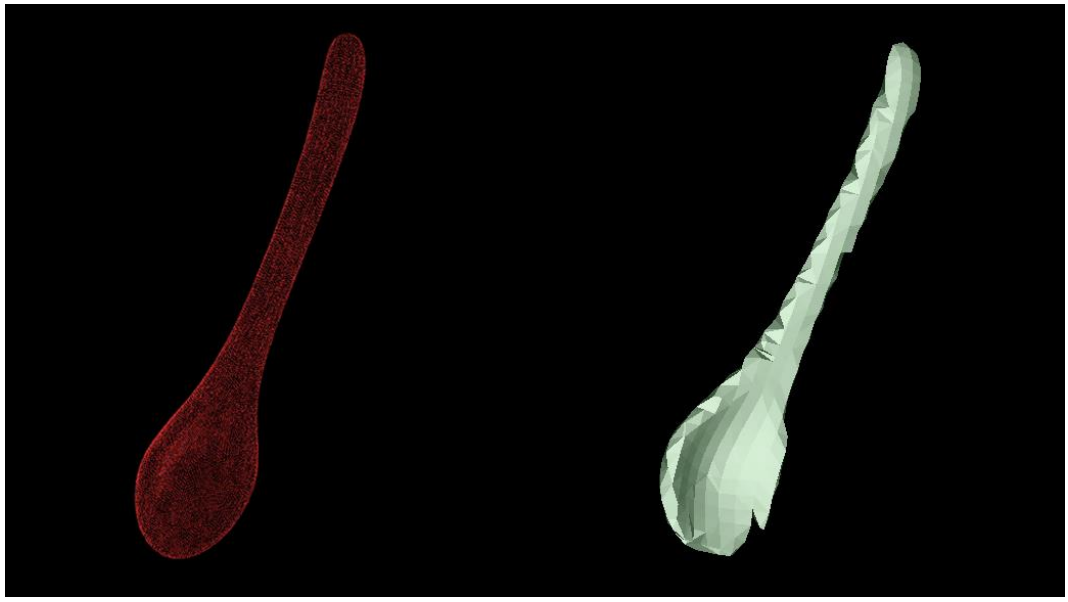
(a) (b) (c)

그림 3-8 조밀한 분포의 점집합

(a) 점집합 원본 (b) 곡면 생성이 예상되는 위치 (c) 점집합을 사용하여 구체를 형성한 경우

이와 같은 문제를 해결하기 위하여, 조밀한 분포의 점 집합에 대응하기 위하여 정점 법선 벡터의 방향을 비교하는 방법을 추가로 사용하였다. 단, 두 구체의 거리가 충분히 멀다고 판단될 경우 정점 법선 벡터의 방향을 비교하는 절차 없이 두 구체는 서로 다른 표면에 있다는 결정을 내린다. 그렇지 않을 경우, 구멍이 발생하여 모델을 제대로 표현할 수 없게 되는 경우가 발생한다. 그림 3-9의 (b)는 (a) 와 같은 점집합을 원본으로 사용하여, 정점 법선 벡터의 방향을 고려하지 않은 방법으로 부호거리장을

계산하고 메쉬 재구성을 실행하여 얻은 결과이다. 그림상에서 나타나 있듯이 재구성된 메쉬 모델은 매끈한 표면을 갖지 못하고, 군데군데 구멍이 나 있음을 알 수 있다. 부호거리장을 구하는 과정에서 정점 법선 벡터의 방향을 비교하지 않았기 때문에 점들이 조밀하게 분포된 점집합에 대해서 정확한 계산 결과를 얻지 못한 것이 (b)와 같은 현상의 원인이다.



(a) (b)

그림 3-9 잘못 계산된 부호거리장의 메쉬 재구성 결과

(a) 점집합 원본 데이터 (b) 메쉬 재구성 결과

4. 부호거리 계산

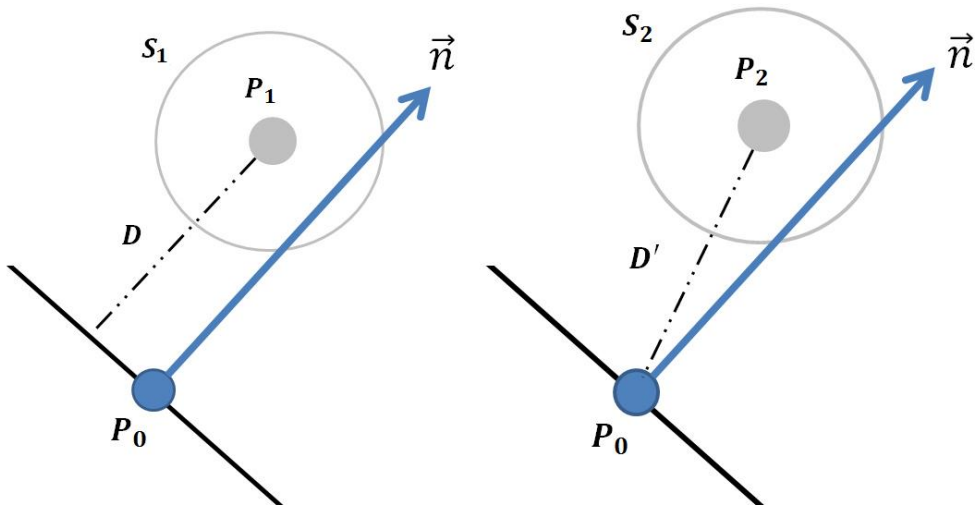
검사 지점으로부터 최근접점까지의 거리는 부호 없는 거리(unsigned distance)이다. 해당 지점이 점집합의 내·외부인지를 판단하고 그 결과에 따라 부호를 구하였으므로, 해당 지점과 최근접점간의 거리값을 측정하여 부호를 붙이면 부호거리를 계산할 수 있다. 거리를 측정하기 위해서는 검사지점을 기준으로 하여 하나의 초평면을 만들고, 투사된 광선의 방향으로 지역 평면으로부터 점집합상의 최근접점까지의 수직 거리를 구한다. 이는 점집합의 결함 영역으로 인하여 발생할 수 있는 구멍 등에 대비하여 비교적 정확한 부호거리를 계산하기 위함이다. 제안하는 거리 측정 방식은 입력 데이터인 점 집합에 결함 영역이 존재할 경우 이를 수정한 점 집합에서 부호거리를 측정하는 것과 비슷한 효과를 가진다.

어떤 지점 $P_0 = (\mathrm{x}, \mathrm{y}, \mathrm{z})$ 로부터 투사된 광선의 방향 벡터가 $\vec{\mathrm{n}} = (\mathrm{a}, \mathrm{b}, \mathrm{c})$ 이고, $\vec{\mathrm{n}} \cdot P_0 = -\mathrm{d}$ 일때, 광선의 방향 벡터 및 출발 지점이 형성하는 초평면의 식은 $\mathrm{ax} + \mathrm{by} + \mathrm{cz} + \mathrm{d} = 0$ 이다. 한편, 해당 초평면으로부터 어떤 점 $P_1 = (x_1, y_1, z_1)$ 까지의 수직 거리는 다음과 같은 수식으로 나타낼 수 있다.

$$D = \frac{|\,\mathrm{a}x_1 + by_1 + cz_1 + d\,|}{\sqrt{a^2 + b^2 + c^2}} \quad \cdots\cdots\cdots \text{(수식 3-1)}$$

그림 3-10 (a)는 초평면을 형성하고 구체의 중심점으로부터 거리를 측정하는 과정을 묘사하고 있다. 광선의 출발지점은 P_0이며, P_0에서 화살표 방향으로 광선을 투사하는 상황을 가정하였다. 이 때 광선의 방향 벡터는 $\vec{n}$과 같다. 출발지점의 좌표 및 광선의 방향 벡터를 이용하여 평면을 형성하며, 광선과 충돌하는 구체 S_1의 중심점인 P_1과

평면의 수직 거리를 계산한 결과를 D로 나타낼 수 있다.



(a) (b)

그림 3-10 초평면으로부터의 거리 측정

(a) 초평면 수직 거리 D (b) 유클리드 거리 D'

그림 3-10 (b)와 같이 유클리드 거리를 사용하여 검사지점 P_0으로부터 최근접점 P_2까지의 거리 D' 를 구하고 그 결과를 부호거리 계산에 반영할 경우 점집합 분포의 불균형함으로 인하여 메쉬 재구성 시 고르지 못한 표면을 얻을 수 있다. 이를 보정하기 위하여 광선의 방향 벡터와 벡터의 출발점인 검사지점의 위치를 사용하여, 벡터와 검사지점이 형성하는 초평면으로부터 점집합 내의 최근접점까지의 수직 거리 D를 구하였다. 그림 3-12는 그림 3-11과 같이 구체 형태로 분포된 점집합을 원본 데이터로 하여, 각자 다른 방법을 사용하여 계산한 부호거리장으로 메쉬 재구성을 실행한 결과를 나타내고 있다. 그림 3-12 (a)의 경우 재구성한 메쉬 모델이 구체의 형태를 띄고 있으나, 표면이 매우 거칠게 구성되어 있는 것을 알 수 있다. 이는 점집합의 고르지

못한 분포 및 결함 영역으로 인하여, 유클리드 거리를 사용하여 거리를 측정할 경우 정확한 부호거리장을 계산하기 어렵기 때문이다. 반면 (b)의 경우 초평면 수직 거리를 사용하여 부호거리장을 계산하였기 때문에, 재구성된 메쉬 모델의 표면이 매끄럽게 구성되어 있는 것을 알 수 있다.

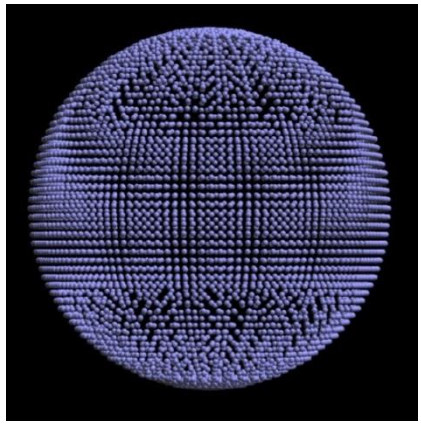
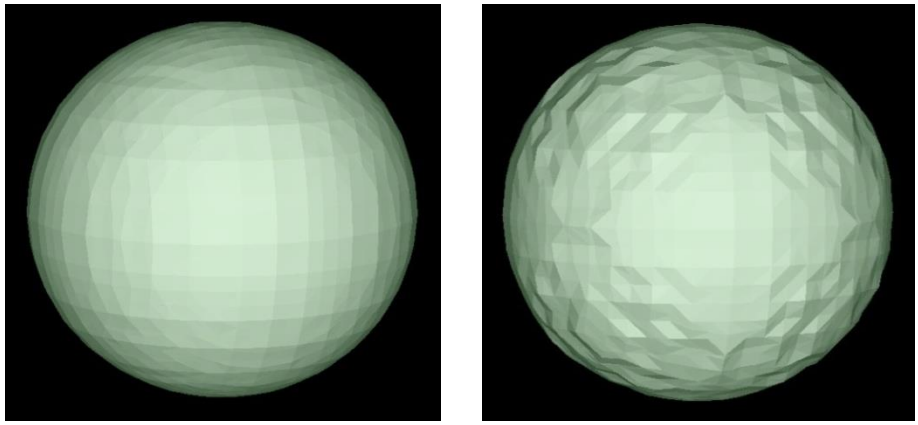


그림 3-11 점집합 원본 데이터



(a) (b)

그림 3-12 다른 거리 계산 방법을 통하여 얻은 메쉬 재구성 결과

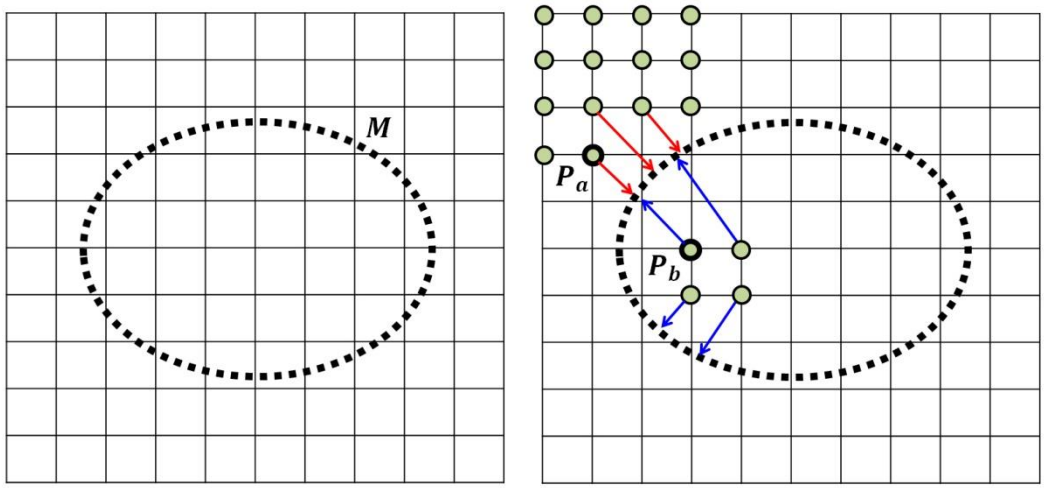
(a) 초평면 수직 거리 (b) 유클리드 거리

Ⅳ. 구현 결과

A. 부호거리장(signed distance field)

본 연구에서 제안한 방법을 바탕으로 한 부호거리장의 계산은 C++ 언어를 사용하여 구현하였다. 각 점을 감싸는 구체의 반지름을 구하기 위하여, PCL(Point Cloud Library)을 사용하여 점집합으로부터 kd-tree를 구축하여 k-nearest search에 사용하였다. PQP(Proximity Query Package) 라이브러리를 변형하여 최단거리 계산과 충돌검사를 위한 경계 볼륨 계층구조를 작성하고 근접질의 계산을 실시하였으며, 부호거리 장의 가시화를 위하여 OpenGL 라이브러리를 사용하였다. 상기 언급한 모든 과정은 Intel Core i7 3.07GHz CPU, 8GB memory, GTX 680 및 Windows 7을 사용하는 시스템에서 Visual Studio 2010을 이용하여 구현하였다.

부호거리장을 계산하기 위해서는 점집합 주변의 공간을 3차원 그리드를 형성하여 분할하고 크기가 같은 다수의 셀을 각 셀의 정점으로부터 부호거리를 구해야 한다. 그림 4-1은 부호거리장을 계산하는 방법을 나타내고 있다. (a)는 점집합 M의 주변 공간을 그리드를 사용하여 분할한 결과이다. 그림에서는 간략한 묘사를 위하여 3차원 그리드를 사용한 공간 분할의 예시를 2차원으로 표현하였다. 그림의 (b)는 공간을 분할하여 얻은 각 셀의 정점에서 점집합 M에 대한 부호거리를 구하는 과정을 나타내고 있다. 그림에서 점 P_a의 부호거리는 양수이며, 점 P_b에서의 부호거리는 음수이다. 모든 셀의 정점에 각 점으로부터 점집합 M에 대한 부호거리를 구하여 저장하면 부호거리장의 계산이 종료된다.



(a) (b)

그림 4-1 부호거리장의 계산 예시

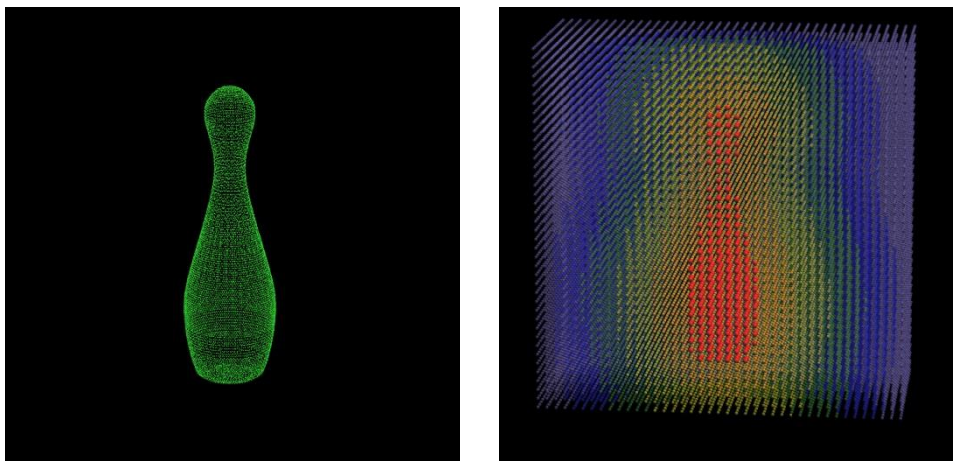
(a) 그리드를 사용한 점집합 주변의 공간 분할 (b) 점집합에 대한 부호 거리 계산

실험에 사용된 각 점집합에 대한 정보 및 실험결과는 표 4-1과 같다. k는 구체의 반지름을 찾기 위하여 사용된 값을 의미한다. 부호거리장 계산에 소요된 시간을 함께 기재하였다.

표 4-1 부호거리장 실험에 사용된 점집합 정보

비교정보 / 점집합	정점의 개수	공간 분할 수	k	소요시간 (초)
Pin	12320	40^3	5	59.296
Sphere	5366	30^3	10	19.931

그림 4-2는 Pin 점집합 및 해당 점집합에 대하여 계산된 부호거리장의 결과를 나타내고 있다. Pin 점집합은 (a)와 같이 점이 볼링핀의 형태와 흡사하게 분포되어 있는 데이터이다. (b)는 Pin 점집합에 대하여 부호거리장을 계산하고 이를 가시화한 결과를 나타내고 있다. 붉은색 점 부분은 점집합의 내부 지점이며, 다른 색의 점으로 표시된 부분은 모두 점집합의 외부 지점이다. 점집합으로부터 외부 위치에 있는 점의 경우 점집합으로부터 거리가 가까운 순서대로 노란색 – 초록색 – 푸른색 – 보라색으로 표기된다. 부호거리장의 가시화 결과로 미루어 볼 때 메쉬 재구성 시 붉은색 점과 노란색 점의 경계 부분에서 곡면이 형성될 것임을 추측할 수 있다.



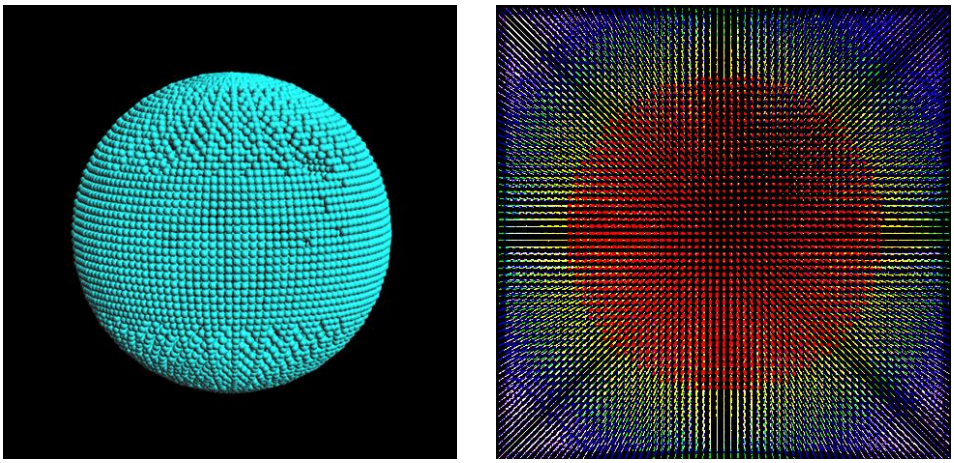
(a) (b)

그림 4-2 Pin 점집합 및 가시화된 부호거리장

(a) 점집합 원본 데이터 (b) 부호거리장의 가시화 결과

그림 4-3은 Sphere 점집합 및 해당 점집합에 대하여 계산된 부호거리장의 결과를

나타내고 있다. Sphere 점집합은 (a)와 같이 점들이 구의 형태와 흡사하게 분포된 데이터이다. 부호거리장의 가시화 결과는 그림 4-2에 나타난 Pin 모델에 대한 실험 결과와 동일한 방법으로 표시되었다.



(a) (b)

그림 4-3 Sphere 점집합 및 가시화된 부호거리장

(a) 점집합 원본 데이터 (b) 부호거리장의 가시화 결과

B. 메쉬 재구성(mesh reconstruction)

계산한 부호거리장의 정확성을 시험하기 위하여 마칭 큐브 알고리즘을 사용한 메쉬 재구성을 실험하였다[12]. 재구성 실험은 상기의 부호거리장을 계산하는 경우와 동일한 하드웨어 및 소프트웨어 환경에서 이루어졌으며, Paul Bourke가 마칭 큐브 알고리즘에 기반하여 제작하고 공개한 소스 코드를 사용하였다[24]. 해당 소스 코드는 C++ 언어로 구현 및 수정되었으며, 재구성 된 메쉬를 화면에 출력하기 위하여

OpenGL 및 WxWidget 라이브러리를 사용하였다.

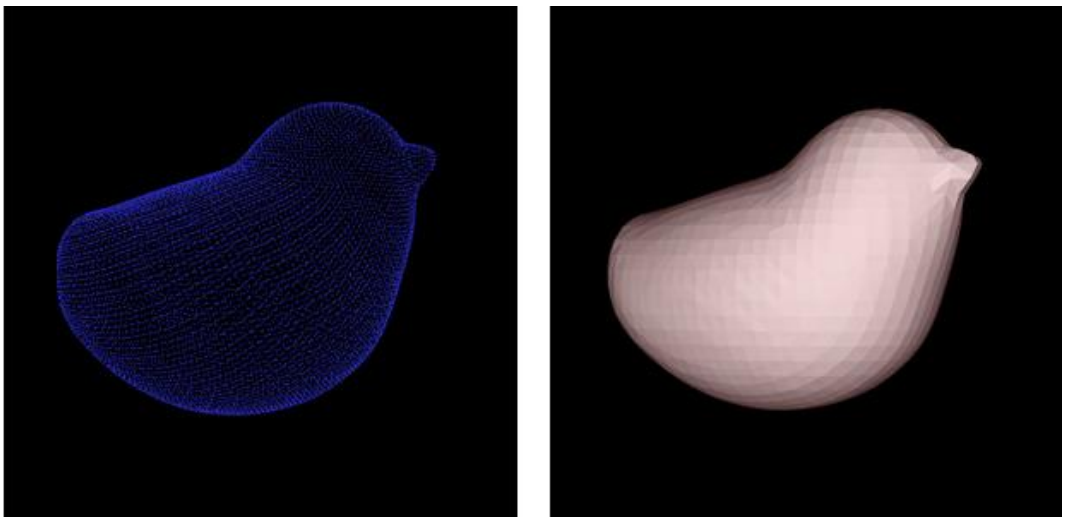
실험에 사용된 각 점집합에 대한 정보 및 실험결과는 표 4-2와 같다. k는 구체의 반지름을 찾기 위하여 사용된 값을 의미한다. 부호거리장 계산에 소요된 시간을 함께 기재하였다.

표 4-2 메쉬 재구성 실험에 사용된 점집합 정보

비교정보 / 점집합	정점의 개수	공간 분할 수	k	소요시간 (초)
Bird	12779	50^3	4	183.64
Bowl	11499	80^3	4	688.024
Pin	12320	40^3	5	59.296
Spoon	21297	150^3	5	2589.976
Sphere	5366	30^3	10	19.931
Pumpkin	15688	60^3	21	439.038

다음은 일반적인 구조의 점집합으로부터 부호거리장을 계산하여 메쉬 모델을 재구성한 결과이다. 그림 4-4는 Bird 모델에 대해 부호거리장 계산 및 메쉬 재구성을 실행한 결과를 나타내고 있다. 실험에 사용한 원본 데이터는 (a)와 같이 새와 흡사한 형

태로 점이 분포된 점집합이며, (b)의 메쉬 재구성 결과를 통하여 매끄럽게 구성된 메쉬 모델을 얻을 수 있음을 알 수 있다.

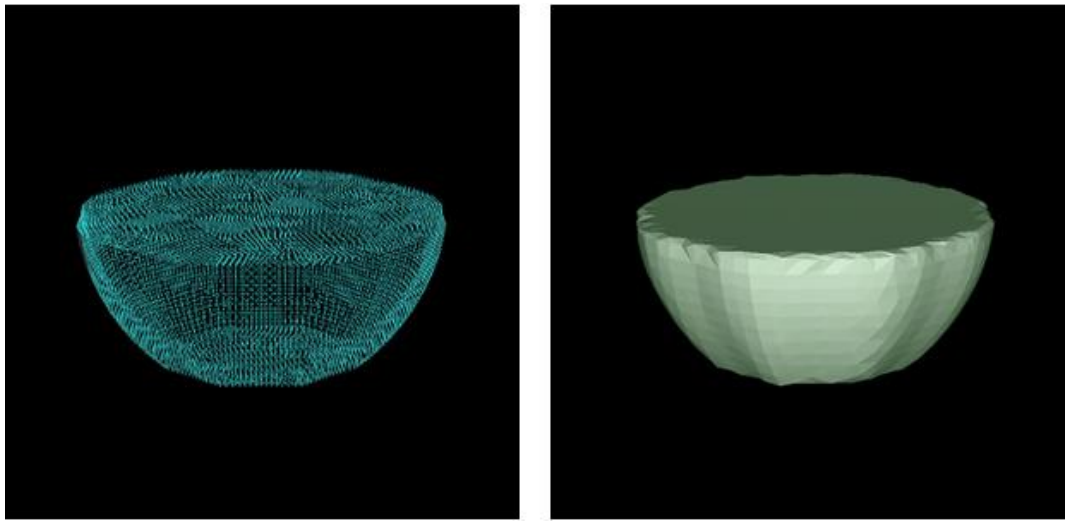


(a) (b)

그림 4-4 Bird 점집합 및 메쉬 재구성 결과

(a) 점집합 원본 데이터 (b) 메쉬 재구성 결과

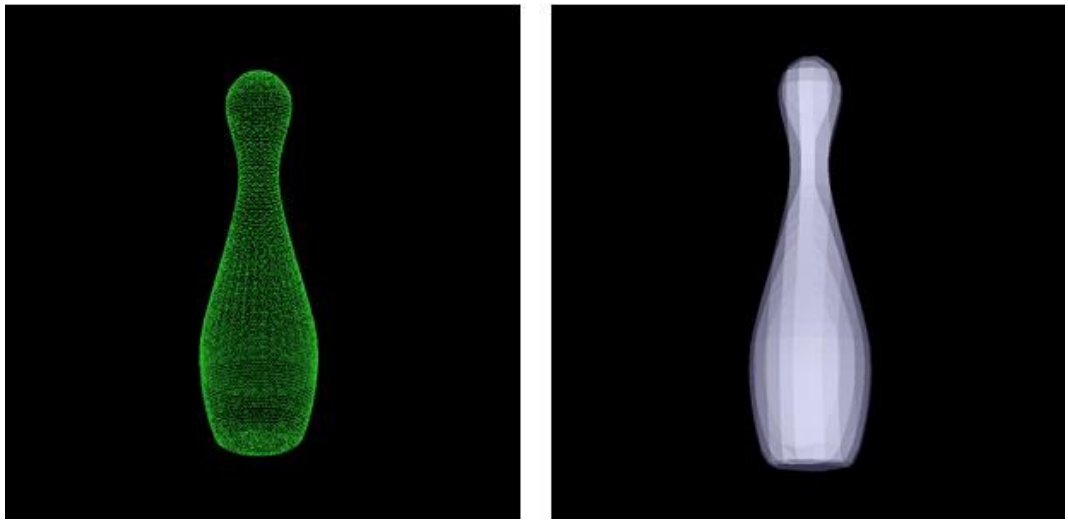
그림 4-5는 그릇과 유사한 형태로 점이 분포된 Bowl 점집합, 그림 4-6은 볼링핀 형태로 점이 분포된 Pin 점집합, 그림 4-7은 숟가락과 유사한 형태로 점이 분포된 Spoon 점집합의 원본 데이터 및 메쉬 재구성 결과를 나타내고 있다. Bird 점집합에 대한 실험 결과와 마찬가지로 고품질의 메쉬 모델을 얻을 수 있었다.



(a) (b)

그림 4-5 Bowl 점집합 및 메쉬 재구성 결과

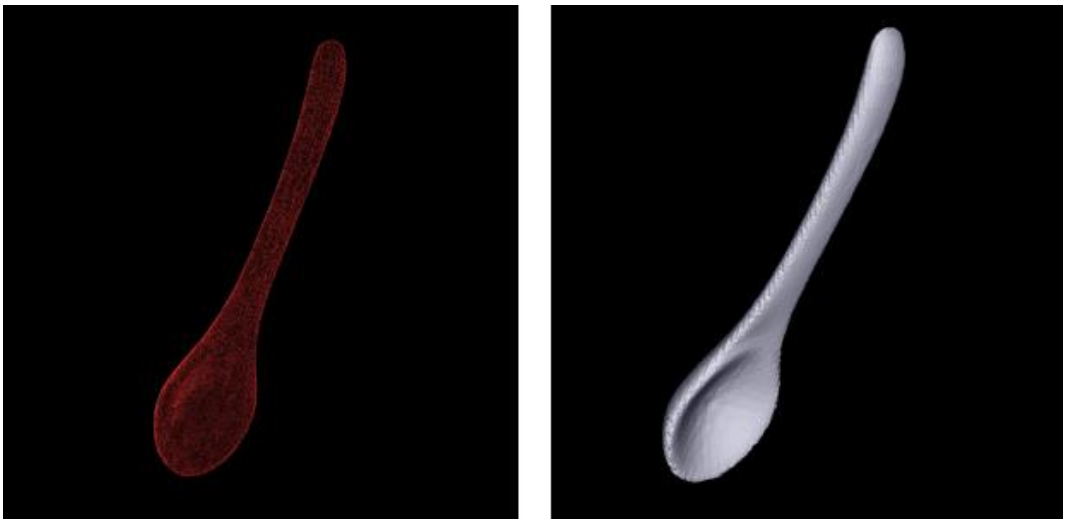
(a) 점집합 원본 데이터 (b) 메쉬 재구성 결과



(a) (b)

그림 4-6 Pin 점집합 및 메쉬 재구성 결과

(a) 점집합 원본 데이터 (b) 메쉬 재구성 결과



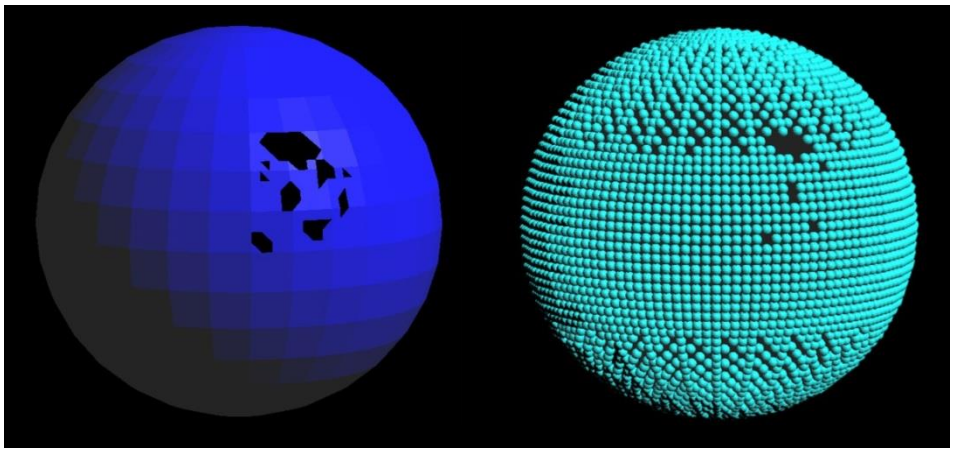
(a) (b)

그림 4-7 Spoon 점집합 및 메쉬 재구성 결과

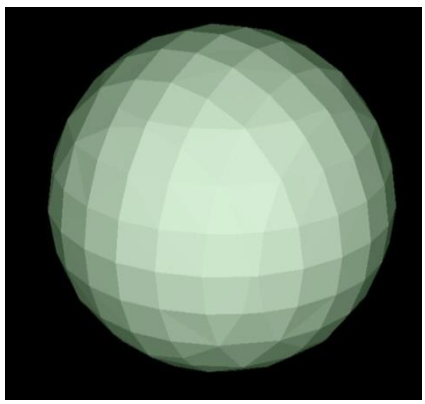
(a) 점집합 원본 데이터 (b) 메쉬 재구성 결과

또한, 구체의 반지름과 관련된 k의 값을 조절함으로써 결함 영역이 존재하는 점집합에 대한 부호거리장의 안정적인 계산이 가능함을 확인하였다. 그림 4-8과 그림 4-9는 구멍을 가진 다각형 집합(Polygon soup)으로부터 정점 정보만을 추출하여 점집합을 형성한 후, 이에 대한 부호거리장을 계산하여 메쉬 재구성을 실행한 결과물이다. Sphere 모델은 구체의 형태를 가진 다각형 집합이다. 다각형 집합의 원래 형태는 그림 4-8의 (a)와 같으며, 다각형 집합에서 추출한 점집합의 분포도를 (b)에서 나타내고 있다. 특히, 구체의 우측 부분에 점이 존재하지 않는 결함 영역이 있다는 것을 (b)를 통하여 확인할 수 있다. 그림 4-8의 (c)는 (b)와 같은 데이터를 입력으로 받아 부호거리장을 계산하고 재구성한 메쉬 모델을 렌더링한 결과이다. (b)에 존재하는 결함

영역이 (c)에서는 수정되어 구멍이 없는 메쉬 모델로 재구성 된 것을 알 수 있다. 이는 Sphere 모델에 대한 부호거리장을 계산하는 과정에서 구체를 이용한 점집합의 결함 영역 수정이 잘 이루어졌음을 의미한다.



(a) (b)

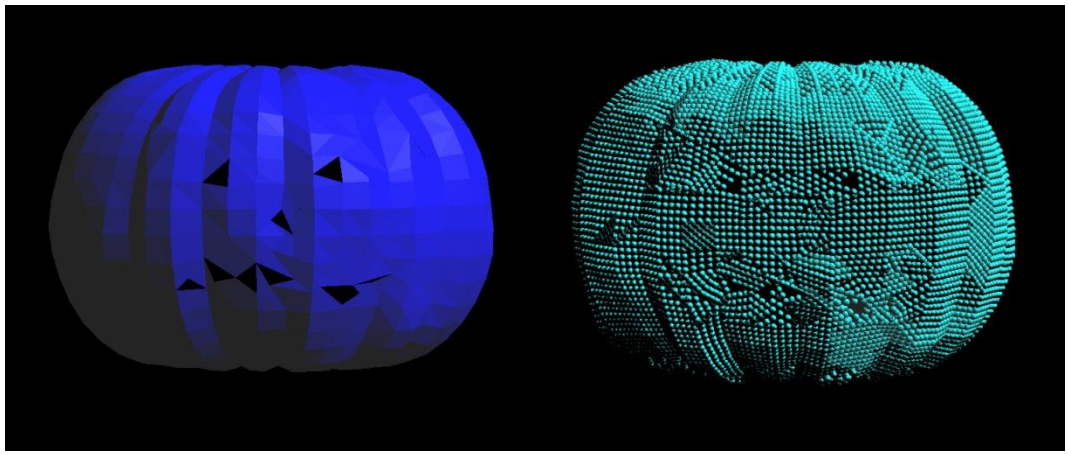


(c)

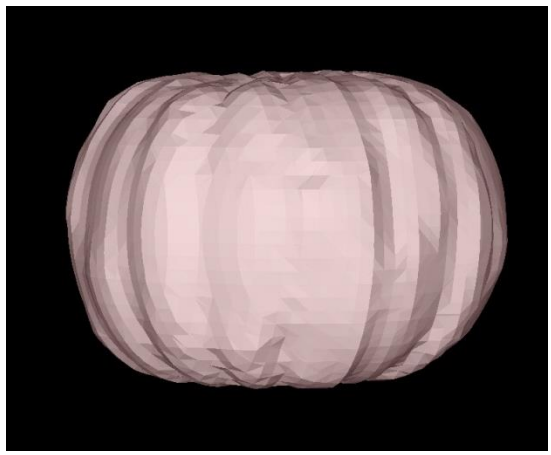
그림 4-8 Sphere 모델의 메쉬 재구성 결과

(a) 다각형 집합 데이터 (b) 추출된 점집합 데이터 (c) 메쉬 재구성 결과

결함을 가진 Pumpkin 모델에 대해서도 같은 실험을 시행하였다. Pumpkin 모델은 호박 형태를 한 다각형 집합이며, 중앙부에 결함 영역을 가지고 있다. 그림 4-9의 (a)는 다각형 집합의 원래 형태를 나타내며, 다각형 집합으로부터 추출한 점집합의 형태는 (b)와 같이 나타낼 수 있다. 그림 (c)에서 볼 수 있듯이, (b)에 존재하는 결함 영역이 수정되어 구멍이 존재하지 않는 메쉬 모델로 재구성되었음을 알 수 있다.



(a) (b)



(c)

그림 4-9 Pumpkin 모델의 메쉬 재구성 결과

(a) 다각형 집합 데이터 (b) 추출된 점집합 데이터 (c) 메쉬 재구성 결과

V. 결 론

A. 연구의 내용 및 의의

본 논문에서는 광선을 투사하여 점의 위치 정보 및 정점 법선 벡터 정보를 포함한 점집합으로부터 부호거리를 비교적 단순한 방법으로 계산하는 알고리즘을 제시하였다. 점집합으로부터 구체를 형성하여 가상의 표면을 정의하고, 검사 지점으로부터의 광선과 충돌하는 구체들이 같은 표면 상에 존재하는지를 검사하는 방법을 이용하여 부호거리를 계산할 수 있다. 점집합 상의 점을 중심으로 형성한 구체는 가상의 표면을 구성하는 요소가 되어 이를 기준으로 점집합의 내·외부 지점을 판정하는 데 사용할 수 있다. 또한, 구체의 반지름을 조절하여 점집합에 결함 영역이 존재하는 경우에도 이를 수정한 부호거리를 계산할 수 있게 하였다. 본 연구는 경계 볼륨 계층구조를 채택하여 최단 거리 계산과 충돌 검사를 빠르게 수행할 수 있으며, 점집합의 등위면이나 외곽선을 별도로 계산하는 과정 없이 구체 간의 거리 및 정점 법선 벡터를 비교하는 비교적 간단한 연산을 통하여 부호거리를 계산할 수 있다는 장점을 가진다.

본 연구는 점집합의 결함 영역을 수정하기 위하여 각 점을 중심으로 형성되는 구체의 최적 반지름을 자동으로 계산할 수 없다는 한계점을 가지고 있다. 결함 영역을 가진 점집합으로부터 구멍이 발생하지 않는 메쉬 모델을 재구성하기 위해서는, 부호거리장을 계산하기 위한 구체의 최적 크기를 직접 찾아야 한다. 또한, 검사지점으로부터 가장 가까운 구체와 가장 먼 구체간의 거리를 계산하는 단순한 연산을 사용하기 때문에 복잡한 구조를 가진 점집합의 경우 정확한 부호거리의 계산이 불가능하다는 단점

을 가진다.

B. 향후 연구

본 연구는 부호거리장의 계산을 위해 점집합의 분포에 따라 필요한 구체의 반지름 크기를 직접 찾아야 한다는 점을 극복하기 위하여, 최적의 반지름 크기 및 이를 자동으로 구하는 방법에 대한 추가적인 연구가 필요하다.

또한 충돌한 구체 중 검사지점으로부터 가장 가까운 구체와 가장 먼 구체의 거리 및 정점 법선 벡터를 비교하는 과정에서, 복잡한 분포 형태를 가진 점집합에 대해 정확한 내·외부 지점 판정이 불가능하다는 단점이 존재한다. 즉, 제시한 알고리즘은 비교적 단순한 분포 형태를 가지는 점집합에 대해서만 적용 가능하다는 한계를 가진다. 따라서, 점집합의 분포 형태에 상관없이 정확한 부호거리를 계산할 수 있는 방법에 대한 추가 연구가 필요할 것이다.

참고 문헌

[1] 박세연, “쾌속조형을 위한 점집합 기반의 단면곡선 생성 알고리즘,” 학위논문(석사), 산업공학과, 한국과학기술원, 2003.

[2] D. T. Lee, B. J. Schachter, “Two algorithms for constructing a Delaunay triangulation,” International Journal of Computer & Information Sciences, Volume 9, Issue 3, pp 219-242, 1980.

[3] David Levin, "Mesh-independent surface interpolation," Geometric modeling for scientific visualization, pp. 37-49, 2004.

[4] Hanspeter Pfister, Matthias Zwicker, Jeroen van Baar, Markus Gross, "Surfels: Surface elements as rendering primitives," Proceedings of the 27th annual conference on Computer graphics and interactive techniques, pp. 335-342, 2000.

[5] Marc Alexa, Johannes Behr, Daniel Cohen-Or, Shachar Fleishman, David Levin, and Claudio T. Silva, "Computing and rendering point set surfaces," IEEE Transactions on Visualization and Computer Graphics, Volume 9, Issue 1, pp. 3-15, 2003.

[6] 박세연, “점집합 모델의 곡면완성을 위한 결함 수정 및 형상곡선 추출에 관한 연구,” 학위논문(박사), 산업및시스템공학과, 한국과학기술원, 2009.

[7] Kenny Erleben, Henrik Dohlmann, 2013, “Signed Distance Fields Using Single-Pass GPU Scan Conversion of Tetrahedra,” GPU Gems 3[Online], https://developer.nvidia.com/content/gpu-gems-3.

[8] Ivan E. Sutherland, Robert F. Sproull, and Robert A. Schumacker, “A Characterization of Ten Hidden-Surface Algorithms,” ACM Comput. Surv, Volume 6, Issue 1, pp. 1-55, 1974.

[9] Hugues Hoppe, Tony DeRose, Tom Duchamp, John McDonald, and Werner Stuetzle, “Surface reconstruction from unorganized points,” Proceedings of the 19th annual conference on Computer graphics and interactive techniques, pp. 71-78, 1992.

[10] Patrick Mullen, Fernando de Goes, Mathieu Desbrun, David Cohen-Steiner, Pierre Alliez, "Signing the unsigned: Robust surface reconstruction from raw pointsets," Computer Graphics Forum. Volume 29, No. 5, pp. 1733-1741, 2010.

[11] Fatih Calakli, Gabriel Taubin, "SSD: Smooth signed distance surface reconstruction," Computer Graphics Forum, Volume 30, No. 7, pp. 1993-2002, 2011.

[12] William E. Lorensen, Harvey E. Cline, “Marching cubes: A high resolution 3D surface construction algorithm,” Proceedings of the 14th annual conference on Computer graphics and interactive techniques, pp. 163-169, 1987.

[13] Bart Adams, Richard Keiser, Mark Pauly, Leonidas J. Guibas, Markus Gross, Philip Dutré, "Efficient Raytracing of Deforming Point-Sampled Surfaces," Computer Graphics Forum, Volume 24, No. 3, pp. 677-684, 2005.

[14] Jae-Kyu Lee, Young J. Kim, "Haptic rendering of point set surfaces," EuroHaptics Conference, 2007 and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems. World Haptics 2007. Second Joint. IEEE, 2007.

[15] Kenneth E. Hoff, III, John Keyser, Ming Lin, Dinesh Manocha, Tim Culver, "Fast computation of generalized Voronoi diagrams using graphics hardware," Proceedings of the 26th annual conference on Computer graphics and interactive techniques, pp. 277-286, 1999.

[16] Eran Guendelman, Robert Bridson, and Ronald Fedkiw, "Nonconvex rigid bodies with stacking," ACM Transactions on Graphics, Volume 22, Issue 3, pp. 871-878, 2003.

[17] Susan Fisher, Ming C. Lin. "Deformed distance fields for simulation of non-penetrating flexible bodies," Computer Animation and Simulation 2001, pp. 99-111, 2001.

[18] Neil Molino, Robert Bridson, Joseph Teran, Ronald Fedkiw, "A Crystalline, Red Green Strategy for Meshing Highly Deformable Objects with

Tetrahedra," In 12th Int. Meshing Roundtable, pp. 103-114, 2003.

[19] J. Andreas Bærentzen, Henrik Aanæs, "Generating signed distance fields from triangle meshes," Informatics and Mathematical Modeling, Technical University of Denmark, DTU, 2002.

[20] J. Andreas Bærentzen, Henrik Aanæs, "Signed distance computation using the angle weighted pseudonormal," IEEE Transactions on Visualization and Computer Graphics, Volume 11, Issue 3, pp. 243-253, 2005.

[21] Jon Louis Bentley, "Multidimensional binary search trees used for associative searching," Communications of the ACM, Volume 18, Issue 9, pp. 509-517, 1975.

[22] Eric Larsen, Stefan Gottschalk, Ming C. Lin, Dinesh Manocha, “Fast proximity queries with swept sphere volumes,” Technical Report TR99-018, Department of Computer Science, University of North Carolina, 1999.

[23] D. Jack Elzinga and Donald W. Hearn, “The minimum covering sphere problem,” Management Science, Volume 19, No. 1, pp. 96-104, 1972.

[24] Paul Bourke, 2013, “Polygonising a scalar field,” [Online], http://paulbourke.net/geometry/polygonise/.

ABSTRACT

Signed Distance Calculation for Point Sets using Ray Casting

Department of Computer Science and Engineering

The Graduate School of Ewha Womans University

Shin, Hye Jin

The point set is a very useful and lightweight data structure used in computer graphics, since it includes only point geometry data. Since points do not have topology information, a point set should be reconstructed to a mesh model to be applicable to graphical applications. We often calculate signed distance fields to reconstruct a mesh model from a point set.

For a point set, signed distance cannot be directly calculated since we cannot determine where is inside or outside from a point set. Many studies suggested defining an implicit surface in order to define the sign for a point set. However, most of them are complicated and involving.

If there are holes in a point set, we have to correct them since they can induce numerical errors to calculate signed distance. For these reasons, we present a method to calculate signed distance for a point set using ray casting to correct holes with simple calculation. We define being inside and outside by creating spheres from the point set and shooting rays. From the ray-sphere intersection test, we know the number of surfaces that exist to the direction of a ray. We assume a point is outside of a point set if a ray collides with two surfaces. Otherwise, the point is inside of a point set. We use a set of spheres to correct the holes and to define an implicit surface from the point set. Also, we use a bounding volume hierarchy structure for fast distance calculation and collision detection.

Proposed algorithm is using simple calculation to calculate distance and sign, between a specified query point and a point set. We do not need to extract the isosurface or contour line to define being inside or outside from a point set. Also, we can define the sign correctly from a point set with missing data.

Our algorithm can be used to calculate a signed distance field. We calculate and visualize the signed distance field. Also, we reconstruct mesh models from the signed distance fields using Marching cube algorithm. The reconstructed surfaces are smooth without having holes.

감사의 글

저에게 공부할 수 있는 기회를 허락하시고, 2년 동안 지도해주신 김영준 교수님께 진심으로 감사를 드립니다. 여러모로 부족한 학생이었지만 연구실에서 생활하는 동안 교수님의 지식과 열정으로부터 많은 것을 배우고 성장할 수 있었습니다. 논문 지도에 힘써주신 조동섭 교수님, 이상호 교수님께도 깊은 감사의 말씀을 드립니다.

학부 과정 및 대학원 과정에서 많은 가르침을 주시고, 더 넓은 시야를 갖도록 도와주신 이화여대 컴퓨터공학과 교수님들께 감사드립니다.

연구실의 영은 언니와 Afroza에게는 연구실 생활 동안 여러 가지로 도움을 받았습니다. 덕분에 즐거운 연구실 생활을 할 수 있었습니다. 진심을 담아 감사의 뜻을 전합니다. 또한 다양한 지식을 가르쳐주시고 저의 질문에 친절하게 응답해주신 Xinyu Zhang 박사님, Fu-chang Liu 박사님, 배명수 박사님, 이택희 박사님, 윤민철 박사님과 Yi Li에게 감사드립니다.

학부생 시절, 부족한 저에게 더욱 많은 공부를 할 수 있도록 기꺼이 도움을 주신 한독미디어대학원대학교의 최유주 교수님께 감사의 말씀을 올리고 싶습니다. 아울러 학문에 대한 뜻을 계속하는 데 동기가 되어 준 친구들에게 고마움을 전합니다.

많은 분들의 호의와 격려로 논문을 완성할 수 있었습니다. 도움을 주신 모든 분들께 다시금 깊은 감사의 말씀을 드리고 싶습니다. 더욱 성장하여 제가 받았던 도움을 나눌 수 있는 사람이 될 수 있도록 노력하겠습니다.

2013년 12월

신혜진