2차원 딜로니 삼각화를 이용한 3차원 메시 생성

최지훈[O] 윤종현 박종승
인천대학교 컴퓨터공학과
{jongbang,jhyoon,jong}@incheon..ac.kr

3D Mesh Creation using 2D Delaunay Triangulation of 3D Point Clouds

Ji-Hoon Choi[O] Jong-Hyun Yoon Jong-Seung Park
Department of Computer Science and Engineering, University of Incheon

요 약

본 논문에서는 3차원 점집합으로부터 3차원 메시를 생성하는 효율적인 기법을 소개한다. 대표적인 3차원 삼각화 방법으로 3차원 딜로니 삼각화 기법이 있으나 물체의 표면만을 고려한 메시 생성을 위한 방법으로 비효율적인 측면이 있다. 본 논문에서는 적은 계산량으로 물체의 표면 메시를 생성하는 기법을 소개한다. 물체의 각 영역을 분할하고 각 영역에 대해서 2차원 딜로니 삼각화를 적용하여 3차원 메시 구조를 얻는다. 3차원 점 집합에 대해 OBB(Oriented Bounding Box)를 계산하고 이를 기준으로 점 집합을 여러 분할 영역으로 나누고 각 부분 점 집합에 대해서 2차원 딜로니 삼각화를 실시한다. 각 2차원 삼각화 결과는 점전적으로 전체 메시에 병합된다. 또한 병합된 메시에서 잘못된 에지의 연결을 제거함으로써 객체의 삼각 분할 결과를 향상시킨다. 제안된 메시 생성 기법은 다양한 영상 기반 모델링 응용에서 효과적으로 적용될 수 있다.

Abstract

The 3D Delaunay triangulation is the most widely used method for the mesh creation via the triangulation of a 3D point cloud. However, the method involves a heavy computational cost and, hence, in many interactive applications, it is not appropriate for surface triangulation. In this paper, we propose an efficient triangulation method to create a surface mesh from a 3D point cloud. We divide a set of object points into multiple subsets and apply the 2D Delaunay triangulation to each subset. A given 3D point cloud is cut into slices with respect to the OBB(Oriented Bounding Box) of the point set. The 2D Delaunay triangulation is applied to each subset producing a partial triangulation. The sum of the partial triangulations constitutes the global mesh. As a postprocessing process, we eliminate false edges introduced in the split steps of the triangulation and improve the results. The proposed method can be effectively applied to various image-based modeling applications.

키워드 : 메시 생성, 딜로니 삼각화, 점집합, OBB
Keywords : Mesh creation, Delaunay triangulation, Point cloud, Oriented bounding box

1. 서론

최근 실사 영상들로부터 3차원 점들을 획득하는 연구가 활발히 진행되고 있다. 3차원 점들의 좌표값들이 추정되면 이들로부터 실제의 형상 메시를 생성해야 3차원 렌더링이 가능하다. 3차원 형상에 대한 메시 모델이 주어지면 다양한 기법으로 화려한 입체 표현이 가능해진다. 3차원 그래픽의 응용에는 정교한 3차원 객체들이 필요하고 이들 3차원 객체들의 생성 절차는 디자이너들의 많은 수작업의 결과를 통해서 완성된다. 물체의 표현은 다각형이 결합한 메시 형태가 일반적이다. 다각형의 형태는 일반적으로 삼각형이 사용된다. 하나의 물체는 적게는 수백 개에서 많게는 수십만 개 이상의 다각형을 포함한다. 다각형의 수는 표현의 정교한 정도에 비례하므로 가급적이면 많은 다각형으로의 구성이 바람직하다. 그러나 최근의 3차원 그래픽 분야의 추세인 실

시간 렌더링에서는 렌더링의 고속화가 중요하므로 비슷한 정교함을 가지면서 최소한의 다각형을 사용하는 접근이 가장 바람직하다. 본 연구에서의 목표는 주어진 3차원 점 집합(point cloud)으로부터 물체를 자연스럽게 표현하는 메시 구조를 생성하는 기법의 개발이다.

메시는 연결성(connectivity)에 따라서 세 종류로 나눠 볼 수 있다[1][2]. 먼저 구조적(structured) 메시는 모든 내부 점들이 위상적으로 닮은꼴이며 연결 정보를 배열 형태로 표현할 수 있다. 이에 비해서 비구조적(unstructured) 메시는 점들이 임의의 이웃 점을 가지는 형태이고 연결 정보의 표현을 위해서 고정된 배열을 사용할 수가 없다. 혼합형(hybrid) 메시는 전체적으로는 비구조적 패턴에서 결합된 작은 구조적 메시의 개수에 의해 형성되는 방법이다. 일반적으로 구조적 메시는 단순성과 쉬운 데이터 접근을 제공한다. 반면에 비구조적 메시는 더 형편이 좋은 메시 적응성과 복잡한 영역에 더 적합한 제공을 한다.

점 집합으로부터의 메시 구조 생성은 삼각화(triangulation) 기법을 통해 수행된다[3][4]. 대표적인 삼각분할의 방식에는 Greedy 방식과 딜로니(Delaunay) 방식이 있다[5][6]. Greedy 방식은 필요한 개수의 선분이 더해질 때까지 에지의 길이가 짧은 것부터 순서대로 더하면서 삼각 분할을 구성하는 방식이다. 즉 삼각화에서의 전체 에지 길이를 최소화함이 목표이다. 이 방법은 계산량과 메모리를 많이 요구하고 삼각화 결과도 그리 우수하지 못하다. 딜로니 삼각 분할 방법은 삼각형에 있어서 최소 각도를 최대화하는 기법이다. 즉 한쪽으로 길쭉한 삼각형을 배제시킨다. 삼각화 결과는 각 삼각형의 외접원 내에는 어떠한 점도 포함되지 않아야 한다.

본 논문에서는 3차원 점 집합을 입력으로 하여 3차원 메시를 생성하는 효율적인 기법을 제안한다. 3차원 점 집합은 레이저 스캐너 등의 3차원 스캐닝 장비를 사용하여 획득될 수 있다. 3차원 점 집합은 물체의 표면 상의 점들로만 구성된다고 가정한다. 즉 물체 내부의 구멍과 같이 외부에서 관찰될 수 없는 점들은 점 집합에 포함되어 있지 않다고 가정한다. 이러한 가정을 따르는 점 집합에 대해서는 표면의 국부적인 각 영역들을 2차원 삼각화를 통해 메시화할 수 있으므로 표면 상의 점들에 대한 효율적인 삼각화가 가능하다. 본 논문에서는 2차원 삼각분할의 개념을 확장시켜서 3차원 객체에 대한 효율적인 삼각분할 방법을 제안한다. 2차원 딜로니 삼각화에 기반하여 3차원 점 집합을 분할하고 각 부분 점 집합들에 삼각화를 적용한다. 각 부분 점 집합들에 대한 삼각화 결과들을 합쳐서 전체적인 메시를 생성한다. 각 삼각화 결과에 대해서 잘못된 에지를 검사 및 배제시키는 후처리 절차를 통해서 보다 정확한 메시 구조를 생성한다.

2장에서는 관련 연구, 3장에서는 제안된 3차원 삼각 분할 방법에 대해 설명하고, 4장에서는 삼각 분할 된 실험과정 및 결과에 대해 설명하며, 마지막으로 결론 및 향후 보완과제를 5장에서 설명한다.

2. 관련 연구

점 집합으로부터 메시를 생성하는 기법에 있어서 생성된 메시가 얼마나 물체를 잘 표현하는지가 중요하다[7][8]. 딜로니 삼각 분할 방법은 점의 배치에 관한 몇 가지 접근 방법을 가진다. 하나는 점들의 경계 영역에서 모든 점들을 포함하는 Steiner 점들을 추가시키는 방법이다. 삼각형에서 좋은 삼각 분할을 하기 위한 조건이 되는 것으로 둔각이 없고, 0도에 가까운 작은 각이 없다는 것을 들 수 있다. 딜로니 삼각 분할 방법[6]은 대표적으로 제한된(constrained) 삼각분할기법[9], 점진적(incremental) 방법[10], 분할정복(divide and conquer) 방법[11]으로 분류할 수 있다. 제한된 삼각 분할 방법[9]은 경계 에지들이 작은 여러 에지들로 분할되지 않도록 제약을 두는 기법이다. 점진적 방법[10]은 초기에 모든 점들을 포함하는 초기 삼각형을 지정한다. 새로운 점 p가 입력되었을 때 그 점을 포함하는 삼각형의 예전의 에지는 제거되고 새로운 점과 새로운 에지를 만들어 나간다. 이렇게 삼각 분할을 하게 되면 $O(n^2)$의 수행 시간이 걸린다. 분할 정복기법[11]은 하나의 점 집합에 대해서 선을 그려서 두 부분으로 나누는 재귀적인 방법이다. 딜로니 삼각화를 위한 분할정복 기법은 O(N logN)의 시간 복잡도를 가진다. 또한 최악의 경우에서 $O(N^2)$을 요구하는 반복적 기법도 제시되었다[12]. 딜로니 삼각분할을 기본으로 사면체 메쉬들을 생성하는 방법도 제시되었다[13].

에지 flipping 방법은 사변형들의 최대각을 최소화하기 위해 에지를 교환하는 방법이다[14]. 이 방법은 항상 정확하게 최적의 에지를 발견하지 못한다. 더 향상된 방법인 에지 삽입 방법이 있다[15]. 일반적으로 최악의 삼각 분할된 삼각형의 최악의 점은 흔한 에지 삽입의 후보이다. 이런 후보 점들이 충분히 좋은 형태로 이동하면서 구멍 난 부분에 에지를 추가하는 방법이다.

3. 2차원 삼각화를 통한 메시 생성

본 논문에서는 3차원 점 집합 데이터를 입력으로 OBB(Oriented Bounding Box)를 계산하고 이를 기준으로 삼각분할을 수행하는 방법을 제안한다. 삼각분할 과정에서 각 점마다 가장 가까이 있는 점까지의 거리들 중 가장 큰 값을 임계값으로 지정하고 삼각분할 과정에서 잘못 연결된 긴 에지를 제거하여 결과를 개선한다.

3.1 입력 데이터의 정렬

3차원 데이터에 대한 삼각분할의 입력 데이터로써 객체에 대한 3차원 점 집합 S를 사용한다. 먼저 입력된 점 집합 S에 대해서 OBB를 계산한다. 한 물체의 OBB는 3차원 바운딩 박스로 각 면에 대한 방향 벡터와 길이를 가진다. 즉 OBB의 중심점의 위치 벡터 ($\boldsymbol{b}_c$), 상자의 각 직교축의 방향 벡터 ($\boldsymbol{b}_u$, $\boldsymbol{b}_v$, $\boldsymbol{b}_w$), 상자의 각 직교축으로의 절반 길이 (l_u, l_v, l_w)의 인자로 표현된다. OBB의 각 축은 주성분 분석

(principal component analysis; PCA)를 통해서 세 개의 성분축(component axis)를 찾아서 설정한다. 첫번째 축인 u-축은 첫번째 성분축으로 설정하고 그 두번째 축인 v-축은 두번째 성분축으로 설정한다. 마지막 w-축은 u-축과 v-축의 외적으로 설정한다.

OBB가 구해지면 OBB의 u-,v-,w-좌표계가 3차원 z-,y-,x-좌표계와 일치하도록 변화를 계산하고 이를 적용시켜서 u-,v-,w-축이 z-,y-,x-축과 일치하도록 한다. 즉 z-축은 OBB의 물체의 가장 긴 축에 해당하고 x-축은 가장 짧은 축에 해당한다.

3.2 점 집합의 분할과 삼각화 결과의 병합

본 논문에서 제시하는 방법은 주어진 3차원 점 집합을 여러 부분 집합으로 분할하여 딜로니 삼각화를 수행하고 그 결과를 병합한다. 주어진 3차원 점 집합은 물체의 표면 상의 점들로만 구성되므로 여러 부분 점 집합들로 나누어 처리하게 되면 각 부분 점집합에 대한 2차원 삼각화가 가능하게 된다. 각 부분 점 집합을 한 평면에 투사하고 투사된 점들을 2차원 점들고 간주하고 2차원 삼각화를 수행하게 된다. 부분 점 집합들을 어떻게 결정할 지와 투사할 평면을 어떻게 결정할 지가 삼각화 결과에 영향을 미친다. 본 논문에서는 단순하면서 효율적인 점 집합들의 분할 기법과 투사 평면의 결정 기법을 제시한다.

전체 점 집합의 분할을 위해서 OBB의 축으로 정렬된 3차원 점 집합을 세 축에 대해서 4개씩의 등분으로 분할하여 12개의 분할 공간으로 나눈다. 각 축에 대한 4개의 분할 공간이 전체 공간에 해당하므로 다른 축의 4분할 공간들은 중복되는 부분들을 포함한다. 각 분할 공간에 대해서 순차적으로 2차원 딜로니 삼각화를 적용한다. 삼각화에 참여한 점들은 점 집합에서 제거되고 다음 처리할 분할 공간들에 포함되지 않는다.

먼저 x-축에 대해서 전체 공간을 4등분한다. OBB의 x-좌표의 범위를 4등분하여 4개의 점 집합을 계산한다. 바깥쪽 두 부분 점 집합에 대해서 먼저 2차원 딜로니 삼각화를 적용한다. 그 다음으로는 y-축에 대해서 전체 공간을 4등분한다. 4개의 부분 점 집합 중에서 바깥쪽 두 부분 점 집합에 대해서 2차원 딜로니 삼각화를 적용한다. 한번 딜로니 삼각화에 포함된 점은 이후의 처리에서 제외된다. 그 다음으로 z-축에 대해서 4등분하고 바깥쪽 두 부분 점 집합에 대해서 삼각화를 적용한다. 그 다음에는 아직 처리되지 않은 안쪽 두 부분 점 집합들에 대해서 x-,y-,z-축의 순으로 삼각화를 적용한다. 한번 삼각화된 점들을 이후의 처리에서 제외되므로 부분 점 집합의 크기가 점점 작아지게 된다. 실제로 12개의 부분 점 집합 중에서 최초의 2~4개의 부분 점 집합이 대부분의 점들을 포함하게 된다.

3.3 잘못된 에지의 제거

삼각화 수행으로 잘못된 에지들이 발생될 수 있다. 이러한 에지들은 모두 제거되어야 하므로 유효 에지의 기준을 설정해주도록 해야 한다. 우선 초기 점 집합으로부터 계산한 가장 가까운 이웃점까지의 거리를 사용한다. 가장 가까운 이웃점까지의 최대 허용 거리를 계산할 수 있으며, 이 거리보다 더 큰 거리를 가지는 삼각형들은 모두 제거하도록 한다.

삼각형 제거 시에 사용될 임계값을 초기화 단계에서 계산한다. 입력된 점 집합에서 각 점으로부터 최단거리에 위치한 점까지의 거리를 계산한다. 이때, n개의 점에 대해 점 p_i에서 가장 가까운 점까지의 Euclid 거리를 d_i라고 하자. 임계값은 다음과 같이 계산한다:

$$d_t = \max_i(d_i)$$

임계값은 삼각형의 에지가 잘못 연결되는 것을 제거하는데 사용된다. 임계값은 최소 거리의 최대값에 해당한다. 만약 한 삼각형이 임계값 d_t보다 더 긴 에지를 가지게 된다면 그 삼각형은 부드러운 면의 표현에 적합하지 않음을 의미한다.

3.4 전체적인 삼각화 수행 과정

많은 3차원 점 집합이 국부적으로 2차원 평면으로 투사할 경우에 겹치지 않도록 할 수 있으므로 분할을 통한 2차원 삼각화 기법이 유효함을 알 수 있다. 주어진 3차원 점 점 집합을 여러 부분 점 집합으로 나누고, 각각에 대하여 삼각 분할한 후 그 결과들을 병합한다. 우선 3차원 점 집합의 OBB를 구하고 OBB에 맞도록 세 축을 설정한다. OBB에 맞도록 설정된 새로운 좌표축을 각각 x-,y-,z-축이라 하자. 이제 각 좌표축에 대해서 4등분씩 분할한다. 먼저 x-축 방향으로 OBB를 4등분하고 가장 바깥쪽의 두 등분에 대해서 삼각화를 실시한다. 그 다음, y-축에 대해서 4등분하고 가장 바깥쪽의 두 등분에 대해서 삼각화를 실시한다. 그 다음, z-축에 대해서 동일한 방법으로 삼각화를 실시한다. 이제 남은 안쪽 등분들에 대해서 x-축, y-축, z-축의 순서로 삼각화를 실시한다. 그림 1은 각 등분을 보여준다. (a)에서 (f)까지는 순서대로 나열된 각 등분들이다.

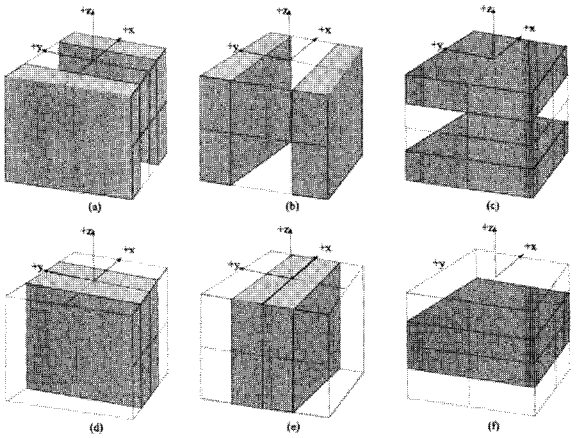

그림 1. 입력 점 집합의 OBB에 대한 공간 분할.

<제안하는 삼각화 알고리즘>

단계 1 (초기화 단계)

1. 3차원 점 집합 S의 주축(principal axis)을 구하고 그에 수직인 부축(minor axis)을 구함.
2. 주축과 부축을 각각 z-축, y-축으로 설정하고 이들과 직교하는 z-축을 설정하여 3차원 좌표계를 재설정함. 원점은 주축과 부축이 교차하는 지점으로 정함.
3. 새로운 좌표계에서의 OBB를 구함.

단계 2 (점 집합 S에 대한 분할 및 점진적 삼각화)

1. OBB의 12개 분할 공간에 대해서 2.2-2.4를 반복함.
2. 점 집합 S에서 분할 공간에 해당하는 점들을 선택함.
3. 선택된 점들에 대해서 2차원 삼각화를 실시함.
4. 부분적인 2차원 삼각화 결과를 전역 메시에 추가함. 추가된 점들은 점 집합 S에서 제거함.

단계 3 (잘못된 에지의 제거)

1. 임계값보다 더 큰 에지를 포함하는 삼각형들을 제거함.

그림 2. 분할을 통한 삼각화 기법의 수행 과정.

그림 2는 전체적인 삼각화 기법의 수행 절차를 나타낸다. 각 분할 공간 내의 점 집합에 대해서 2차원 삼각화를 실시하고 부분 데이터에 대한 삼각화 결과는 전역 메시에 점진적으로 더해진다. x-축에 대해서 분할한 4개의 부분 공간에 대해서는 yz-평면으로 투사할 때의 모습으로 2차원 삼각화를 수행한다. 이때 계산 과정에서는 실제 투사를 하지 않고 y-좌표와 z-좌표만을 이용하면 된다. 12개의 분할 공간의 적용 순서는 그림 1과 같이 (a)에서 (f)로의 순서이다. 각 그림에서 두 개씩의 분할공간을 포함하고 있는데 이들의 적용 순서는 무관하다. 부분적인 2차원 삼각화 결과는 점진적으로 전역 메시에 더해진다. 전역 메시에서 임계값보다 더 큰 에지를 포함하는 삼각형들을 잘못된 삼각형으로 간주하고 제거된다.

4. 실험 결과

그림 3. 실제 사람의 정면 모습에 대한 점 집합.

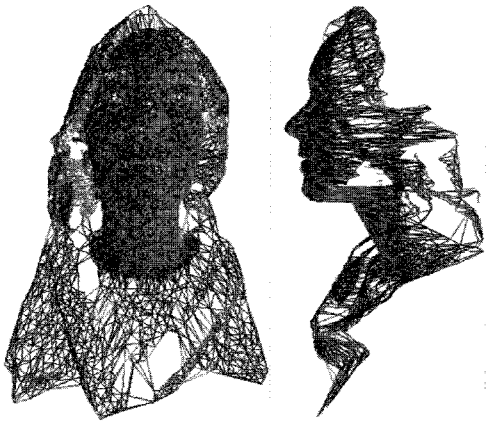

그림 4. 정면 모습 점 집합으로부터의 메시 생성.

제안한 알고리즘의 유효성 검증을 위해 다양한 3차원 점 집합을 사용하여 실험하였다. 먼저 2차원 딜로니 삼각화의 유효성 검증을 위해서 19,089개의 점들로 구성된 점집합을 삼각화하였다. 점집합의 형태가 그림 3에 있고 이를 삼각화한 결과가 그림 4에 있다. 얼굴과 목 중앙부분에 대한 점들은 정면 방향에 대해서 잘 정의된 곡면 상의 점들이나 옆머리나 목 뒤부분의 경우에는 바람직하지 못한 메시 결과가 생성된다.

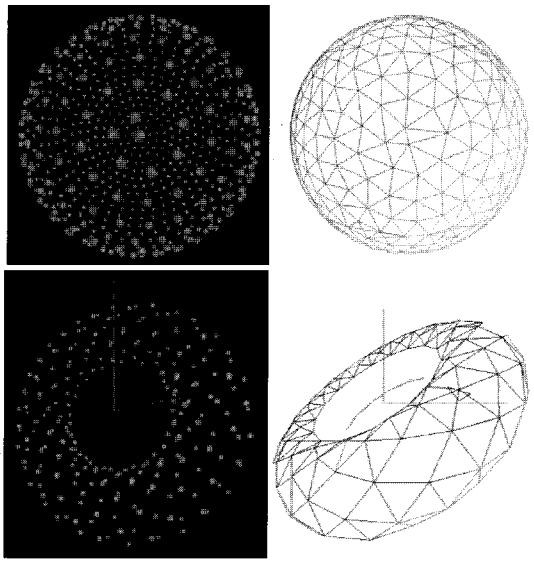

그림 5. 구와 토러스 표면의 점들에 대한 삼각화.

점집합을 다중으로 분할하고 각각의 삼각화 결과를 합하여 전체적인 삼각화를 생성하는 기법의 검증을 위해서 인위적인 데이터를 생성하여 실험하였다. 그림 5는 인위적으로 구(sphere)와 토러스(torus) 표면의 임의의 점들을 생성하고 이를 삼각화한 결과이다. 분할 간격 및 임계값에 영향을 받지 않으면서 올바른 메시가 생성되었다. 구나 토러스와 같

은 형태의 물체는 가장 전형적인 형태로 대부분의 점들이 그림 1의 (a)~(c)에 해당하는 분할 영역에 포함된다.

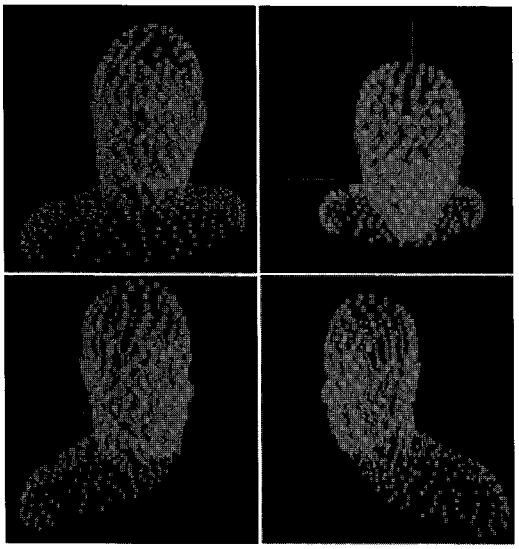

그림 6. 사람의 상반신을 나타내는 점 집합.

일반적인 실물 점 집합에 대해도 제안하는 삼각화를 적용하였다. 그림 6은 삼각화에 입력으로 사용된 점 집합을 나타내고 있다. 약 1,400개의 3차원 점으로 구성된 데이터로서 얼굴을 포함한 사람의 상반신을 나타낸다. 그림 7은 그림 6의 점 집합을 사용하여 삼각화를 한 결과이다. 두상과 어깨 사이의 형상의 굴곡으로부터 여러 잘못된 삼각형들이 생성되었다. 서로 멀리 떨어져 있는 점들이 에지가 연결됨으로써 잘못된 삼각형이 형성된 결과를 그림 7에서 확인할 수 있다. 그림 8은 그림 6의 실험결과와 같은 입력을 사용하고 잘못된 에지 연결을 제거하는 과정을 포함한 제안된 방법이 사용된 실험 결과이다. 그림 7과 그림 8을 비교하였을 때 머리 부분과 어깨 부분의 잘못된 연결된 에지가 제거된 모습을 확인할 수 있다.

그림 9은 뇌 구조를 나타내는 점 집합이다. 이를 입력으로 그림 10의 메시를 형성하였다. 잘못 연결된 에지가 없이 삼각화가 된 결과를 그림 10로부터 확인할 수 있다.

보다 복잡한 형태의 물체에 대한 점 집합에 대한 3차원 메시 생성을 실험하였다. 석고 흉상에 대한 점 집합이 그림 11에 있다. 이로부터 생성된 메시 구조가 그림 12에 있다. 점 집합에서의 정점의 수는 16,848개이다. 흉상의 뒷부분의 일부가 어색한 부분이 있으나 전체적으로 올바른 메시구조가 생성되었다.

분할을 통한 부분 삼각화 및 병합 절차로 항상 올바른 메시가 생성되도록 보장하는 것은 매우 어려운 일이다. 물체의 표면 구조가 매우 다양할 수 있으며 또한 입력되는 표면 점들의 분포가 일정하지 않기 때문이다. 빈 공간이 생겨서 표면이 불연속적으로 표현되거나 적절하지 못한 삼각화가 포함될 수도 있다. 따라서 생성된 메시는 필요에 따라서 후처리 단계에서 수정할 수 있도록 함이 바람직하다. 특히 불연속적으로 끊어진 메시 형태가 생성된다면 자연스럽게 이어주는 메시 스티징 기법의 도입이 바람직하다[16]. 제안한 기법은 대부분의 단순한 형태의 물체에 대해서 잘 동작된다. 오목과 볼록의 형태가 다양한 과도하게 혼합된 형태의 물체에 대해서는 생성된 메시의 일부 영역에서 잘못된 결과가 발견된다.

알고리즘의 처리 시간은 2차원 삼각화만 실시하므로 매우 빠르게 수행된다. 분할과 병합에 있어서의 처리 시간은 크게 늘어나지 않는다. 각 12개의 영역으로의 분할에 대해서는 각 축에 대해서 OBB를 4등분씩 하면 되고 병합은 새로운 삼각형들을 그대로 추가하면 된다. 잘못된 에지의 제거를 위해서는 모든 에지에 대해서 임계값과의 비교연산을 수행하면 된다. 전체를 한번만 2차원 삼각화하는 시간보다 약 2~3배의 수행 시간이 필요하나 3차원 삼각화와는 비교가 되지 않을 만큼 작은 시간이다. 500MB의 메모리를 가진 Pentium 4 2.4GHz PC 상에서 제안된 방법의 실행 속도를 측정하여 보았다. 그림 11의 점 집합은 모두 16,848개의 정점들로 구성되어 있다. 이 점 집합 입력에 대한 제안된 삼각화의 CPU 실행 시간은 약 2초 미만이다. 일반화된 3차원 딜로니 삼각화는 물체를 정사면체 형태의 입방체로 표현하므로 물체의 표면 메시의 표현에만 한정하는 제안 기법과는 비교 대상이 아니다. 그러나 볼륨 메시의 생성 후에 내부 면들을 제거하여 표면 메시를 얻을 수 있으므로 이를 측정하여 비교해볼 수 있다. CGAL 라이버러리[17]를 사용한 3차원 딜로니 삼각화를 적용한다면 약 40초의 실행 시간이 소요된다. 이는 제안된 기법의 실행 시간보다 매우 긴 시간으로 인터액티브한 응용 시스템에서는 수용하기 힘든 계산 시간이다.

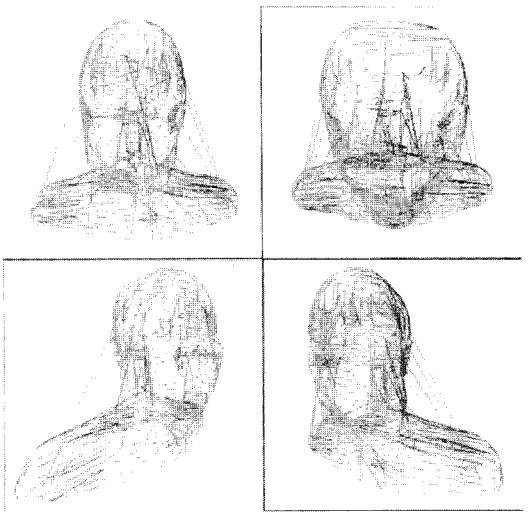

그림 7. 잘못된 에지를 포함한 분할을 통한 삼각화.

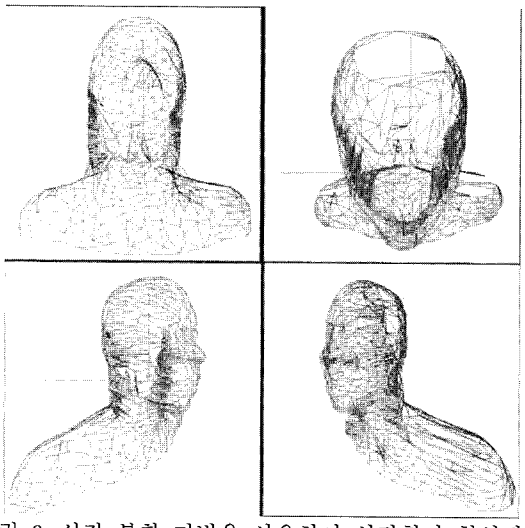

그림 8. 삼각 분할 기법을 사용하여 삼각형이 형성된 3 차원 객체의 모습.

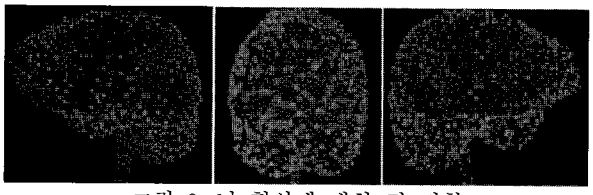

그림 9. 뇌 형상에 대한 점 집합.

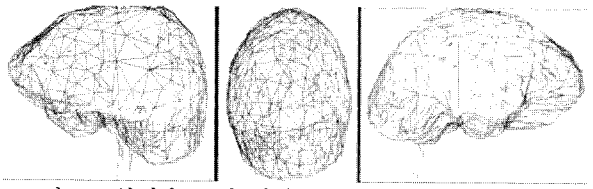

그림 10. 분할을 통한 삼각화 기법을 사용하여 생성한 메시의 모습.

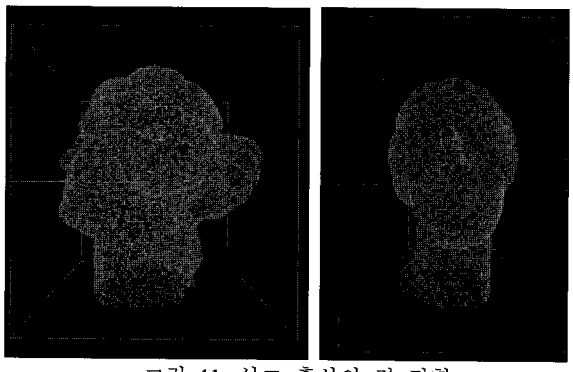

그림 11. 석고 흉상의 점 집합.

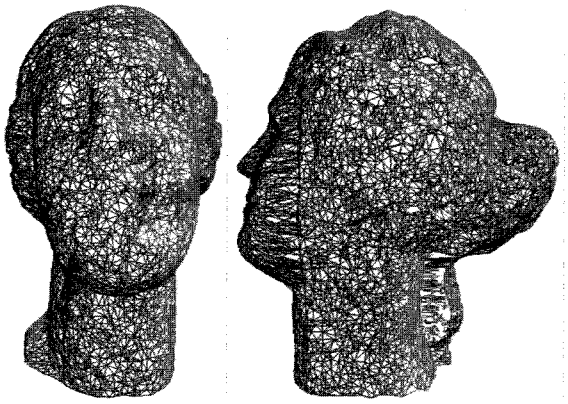

그림 12. 석고 흉상의 점 집합으로부터 생성된 메시.

5. 결론

본 논문에서는 3차원 객체를 표현하는 점 집합에 대하여 메시 구조를 형성할 수 있는 삼각 분할 기법을 제안하였다. 한 객체에 대하여 주축을 계산하고 이를 기준으로 부분적으로 삼각 분할하여 3차원 구조를 형성하였다. 부분 데이터에 대한 삼각 분할 기법으로는 딜로니 삼각 분할 기법을 확장한 방법을 사용하였다. 물체의 각 영역을 분할하고 각 영역에 대해서 2차원 딜로니 삼각화를 적용하여 3차원 메시 구조를 얻는다. 3차원 점 집합에 대해 OBB를 계산하고 이를 기준으로 점집합을 다양한 각도에서 자르고 각 부분 점집합에 대해서 2차원 딜로니 삼각화를 실시한다. 절단하는 각도의 간격이나 폭은 원래의 3차원 점집합에서의 가장 가까운 이웃점들까지의 평균 거리를 이용하여 결정하도록 하였다.

3차원 점집합으로부터 메쉬 생성 시에 잘못된 삼각형들이 다수 생성되는 결과가 빈번히 발생되지만 제안한 삼각화 기법은 잘못된 삼각형들의 생성을 억제시키고 또한 생성 후에 다시 제거하여서 올바른 삼각화가 이루어지도록 하였다. 표면이 비연속적으로 연결되고 오목과 볼록의 깊이 차가 큰 경우에는 자연스럽지 못한 메시가 생성되거나 빈 공간이 발생될 수도 있다. 이를 위해서 후처리 과정으로 수작업의 편집이 필요할 수도 있다. 후처리 과정에서는 삼각 분할 과정에서 잘못된 에지의 연결을 제거함으로써 객체의 삼각 분할 결과를 향상시킨다. 제안된 메시 생성 기법은 다양한 영상 기반 모델링 응용에서 효과적으로 적용될 수 있다.

제안한 방법은 영상으로부터 3차원 모델을 얻는 영상기반 모델링에 적용될 수 있다. 향후 연구과제로, 삼각 분할 후 메워지지 않은 영역에 대한 삼각화 방법에 대해 연구할 계획이다.

참고 문헌

[1] Y. Zheng, R.W. Lewis, D.T. Gethin, "Three-dimensional unstructured mesh generation," Computer Methods in Applied Mechanics and Engineering, Vol. 134, pp. 249-310,

1996.
[2] S.-H. Teng, C. W. Wong, "Unstructured Mesh Generation: Theory, Practice, and Perspectives," International Journal of Computatinal Geometry and Applications, Vol. 10, No. 3, pp. 227-266, 2000.
[3] M. Bern and P. Plassmann, "Mesh generation," Chapter 6 In J.-R. Sack and J. Urrutia, editors, *Handbook of Computational Geometry*. Elsevier Science, 1999.
[4] M. Bern and D. Eppstein, "Mesh Generation and Optimal Triangulation," *Computing and Euclidean Geometry* 2nd, pp. 47-123, 1995.
[5] S. A. Goldman, "A space efficient greedy triangulation algorithm," *Inform. Process. Lett.* 31, pp. 191-196, 1989.
[6] M. Varshosaz, H. Helali, D. Shojaee, "The Methods of Triangulation", Map Middle East 2005, 1st Annual Middle East Conference and Exhibition on Geospatial Information, Technology and Applications, Dubai, UAE, 2005.
[7] D. G. Kirkpatrick, "A note on Delaunay and optimal triangulations," *Inform. Process. Lett.* 10, pp. 127-128, 1990.
[8] M. Bern, D. Eppstein, and J. Gilbert, "Provably good mesh generation," *J. Comput. Syst. Sci.,* Vol. 48, pp. 384–409, 1994.
[9] L. Paul Chew, "Constraind Delaunay Triangulations," Algorithmica, Vol. 4, pp. 97-108, 1989.
[10] D. Lischinski, "Incremental Delaunay triangulation," *Graphics Gems IV*, pp. 47-59, 1994.
[11] R. A. Dwyer, "A simple divide-and-conquer algorithm for computing Delaunay triangulations is O(nloglogn) expected time," *Symposium on Computational Geometry*, pp. 276-284, 1986.
[12] D.T. Lee and B.J. Schacter, "Two Algorithms for Constructing a Delaunay Triangulation," *International Journal of Computer and Information Sciences*, Vol. 3, No. 9, pp. 219-242, 1980 .
[13] T.J. Baker, "Automatic mesh generation for complex three-dimensional regions using a constrained Delaunay triangulation," *Engineering with Computers*, Vol. 5, pp. 161-175, 1989.
[14] S. Hanke, T. Ottmann, S. Schuierer, "The edge-flipping distance of triangulations," *J. Universal Comput. Sci.* 2, pp. 570-579, 1996.
[15] M. Bern, H. Edelsbrunner, D. Eppstein, S. Mitchell and T.-S. Tan, "Edge-insertion for optimal triangulations," *Computational Geometry* 10, pp. 47-65, 1993.
[16] L.P. Kobbelt, M. Botsch, "An Interactive Approach to Point Cloud Triangulation," *Computer Graphics Forum,* Vol. 19(3), pp. 479-487, 2000.
[17] S. Pion, M. Teillaud, "3D Triangulations," CGAL User and Reference Manual (Release 3.3.1), 2007.