쾌속조형 시스템을 위한 3차원 기하학적 형상인 STL의 디지털 워터마킹

김기석[†] · 천인국[††]

요 약

본 논문은 쾌속조형(rapid prototyping) 시스템에서 사용되며 3D 기하학적 형상을 가지는 STL 파일에 워터마크를 삽입하고 추출하는 방법에 관한 연구이다. 제안된 알고리즘은 3D 형상의 왜곡이 없도록 하기 위해, 패싯의 법선 영역과 꼭지점 영역에 워터마크를 삽입한다. 워터마크 비트들은 법선의 위치와 꼭지점의 순서 정보를 이용하여 삽입된다. 제안된 알고리즘은 패싯의 저장 순서에 대한 종속성이 없으며, 워터마크의 비가시성(invisibility)을 충족한다. 제안된 알고리즘으로 3D 기하학적 형상에 워터마크를 삽입하고 추출하는 실험 결과들은 STL로 표현된 3D 원형상에 영향을 주지 않고 워터마크의 삽입과 추출이 가능함을 보여준다.

A Digital Watermarking of 3D Geometric Model STL for Rapid Prototyping System

Kim, Ki-Seok[†] and Chun, In-Gook[††]

ABSTRACT

In this paper, a new watermarking algorithm for STL files which contains 3D geometric information as triangular facets is proposed. STL files are widely used in rapid prototyping industry as a standard interchange format. The proposed algorithm inserts multi-bit watermark information into the surface normal vector and vertex description area of STL file without distorting the original 3D geometric information. According to the watermark bits, the position of normal vector and the direction of vertex sequence are modulated. The proposed algorithm is robust to the attack of changing the order of the triangular meshes. In addition, the invisibility requirement is also satisfied. Experiment results show that the proposed algorithm can encode and decode watermark bits into the various STL files without any distortion of 3D shape.

Key words: 3D watermarking, rapid prototyping, STL

1. 서 론

최근에 통신망의 발달로 정보 교환이 신속하게 이루어지고 있고, 멀티미디어 자료의 사용이 증가되고 있다. 하지만, 디지털 자료는 복제와 조작이 용이하고 복제된 자료는 원본과 동일하다는 부작용이 따른다. 이런 까닭에 디지털 워터마킹 기술을 이용하여 텍스트 문서, 2D 이미지, 동영상, 음악 등과 같은 디지털 컨텐츠에 저작권 정보를 삽입하려는 많은 연구가 진행되어 왔다. 하지만, 최근 산업 현장에서 많이 사용되고 있는 RP 시스템의 3D 형상 모델인 STL에 워터마크를 삽입하려는 연구는 미비한 실정이다.

RP 시스템은 절삭 도구(레이저, 칼 등)를 사용하여 재료를 절단하고 적층하여 시제품(prototype)을 제작하는 시스템이다. RP 시스템으로 시제품(prototype)을 제작하는 목적은 제품 개발 초기 단계에서 설계상

접수일 : 2002년 4월 24일, 완료일 : 2002년 9월 17일
본 연구는 한국 소프트웨어 진흥원의 ITRC 사업에 의해 수행된 것임.
[†] 준회원, 순천향대학교 대학원 전산학과 박사과정
[††] 정회원, 순천향대학교 공과대학 정보기술공학부 부교수

의 오류나 부적합한 요인을 조기에 발견하는 것이다[1]. 이러한 RP 시스템에서의 자료 교환 표준이 바로 STL 파일이며 STL 파일로 표현되는 3D 기하학적 형상 모델은 시제품 제작뿐만 아니라 SF 영화, 3D 애니메이션 등에 이용할 수도 있다. 이러한 응용을 위해, 실물을 3D 스캐닝하거나 CAD 시스템으로 설계하여 STL 파일을 제작하는 작업에 상당한 비용이 발생한다. 그림 1은 RP 시스템이 레이저를 사용하여 다종 재료의 각 단면을 절단한 다음, 절단된 단면들을 적층하여 시제품을 완성하는 것을 보여준다[2-4].

3D 형상에 워터마크를 삽입하는 기존의 연구들은 RP 시스템에서 사용하기에는 부적합한 알고리즘이다. Xiaoyang[5], François[6], Ryutarou[7]가 제안한 알고리즘은 3D 형상에 워터마크를 삽입하며 주로 3D 형상에 HVS (human visual system)로 감지할 수 없는 잡음을 첨가하는 기법이다. RP 시스템은 시제품을 만들어 제품 생산 이전에 검증하는 목적으로 산업 현장에서 많이 사용되기 때문에 높은 정밀도(accuracy)가 필요하다. 뿐만 아니라, 실제 제작될 시제품의 크기를 워터마킹 시점에서 미리 예측할 수 없으므로 기존 알고리즘으로 워터마킹할 경우, 3D 형상이 왜곡될 가능성과 정밀도에 문제가 발생할 가능성이 높다. 기존 알고리즘을 RP에 적용할 때 발생하는 제약점에 관해서는 2장에서 논한다.

STL (standard transfer language) 파일은 패싯으로 구성되며, 패싯은 삼각형을 나타내는 3개의 꼭지점과, 패싯 평면의 앞면과 뒷면을 나타내는 법선 영역으로 구성된다. 본 연구에서 제안하는 알고리즘은 패싯의 법선 영역(normal domain)과 꼭지점 영역(vertex domain)에 동시에 워터마크를 삽입한다. 제안한 알고리즘에서는 워터마크 비트에 따라 법선의 시작 위치가 달라지며, 이와 동시에, 꼭지점의 저장 순서를 변경하여 꼭지점 영역에도 워터마크를 삽입한다. 패싯의 법선 영역이나 꼭지점 영역에 워터마크를 삽입할 경우, 3D 형상에 어떠한 왜곡도 발생하지 않을 뿐만 아니라, 제3자의 단순한 공격에 대하여 강인성을 가진다. 제안된 알고리즘을 사용하면 3D 형상의 왜곡이 전혀 없으므로 정밀도에 문제가 생기지 않는다. 3D 형상의 특성 때문에 법선과 패싯은 밀접한 관계를 가진다. 제안된 알고리즘으로 삽입된 워터마크를 제거하기 위해서는, STL의 두 영역(법선과 꼭지점)을 고려한 알고리즘으로 제거하여야 한다. 두 영역을 고려하지 않은 워터마크 제거 알고리즘으로 공격을 받게 될 경우, STL 3D 원형상이 손상되어 디지털 컨텐츠로서의 가치가 손상된다.

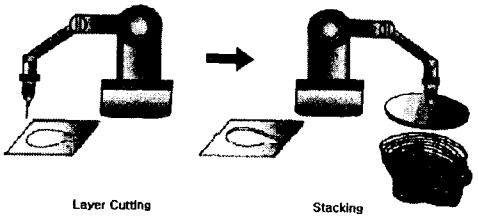


그림 1. 레이저로 절단한 후 적층하는 시스템

본 논문의 구성은 다음과 같다. 2장에서는 3D 형상에서의 기존 워터마킹 기법을 살펴보며, 워터마크가 삽입될 STL과 패싯의 구조에 대해서는 3장에서 고찰한다. 4장에서는 제안된 워터마크 삽입/추출 알고리즘에 대해 살펴보고, 5장에서 제안된 알고리즘의 성능을 평가한다. 마지막 6장은 결론이다.

2. 3D 형상 워터마킹을 위한 기존의 방법

지금까지 연구되어온 3D 형상에서의 워터마킹 알고리즘은 애니메이션, 시뮬레이션, 게임 등과 같은 시스템에서 사용하여 단지 저작권을 보호하는 기능을 수행하기에는 충분하다. 하지만, 3D 형상을 실물로 재현하는 RP 시스템에서는, 실물로 제작될 시제품의 크기를 알 수 없으므로 기존 알고리즘을 적용할 경우, 정밀도 측면에서 문제가 발생한다. RP 시스템에 사용 가능한 워터마킹 알고리즘은 실물의 형상에 어떠한 왜곡도 발생시켜서는 안 된다. 생산 현장에서 사용되는 부품일 경우에는 더욱 큰 정밀도를 가져야 한다[10].

Xiaoyang의 알고리즘을 STL에 적용할 경우, 패싯의 저장 순서를 변경하는 공격에는 강인성을 가진다. 이 알고리즘은 한 비트의 워터마크 삽입을 위해 주위에 있는 3개 이상의 메쉬(mesh)들의 꼭지점을 변경하고 새로운 메쉬를 추가해야하는 알고리즘이다. 이런 까닭에 너무 과다한 오버헤드가 발생할 뿐만 아니라, 한 비트의 워터마크 삽입에 너무 많은 메쉬를 사용함으로서 삽입될 워터마크 크기(비트 수)에 제한을 받을 수도 있다.

François의 알고리즘은 메쉬를 이루는 세 꼭지점 중에서 임의로 한 꼭지점을 선택한 후, 나머지 두 꼭지점이 이루는 선분에 수선을 그렸을 때의 위치를 가지

고 양자화하는 알고리즘으로 HVS로 감지할 수 없을 만큼 꼭지점을 이동시켜 양자화 한다. 따라서 이 알고리즘은 비가시성은 만족할 수 있겠지만, 원래의 3D 형상이 변경되므로 RP 시스템에 적용할 경우 정밀도에 문제가 생길 수 있으며, 또 메쉬의 저장 순서를 변경하는 기본적인 공격에도 워터마크가 제거된다.

Ryutarou의 알고리즘은 3D 형상에서 표면(surface)이 비교적 완만한 위치를 탐색한 후, 그 위치에 워터마크를 삽입한다. 탐색된 위치에 저작권자의 로고나 마크를 가시적으로 나타내기 위해 메쉬(패싯)들을 추가하여 삽입한다. 이런 방식의 워터마크를 제거하기 위해서는 점(vertex)을 결정하여 메쉬들을 새로이 생성하고 기존의 메쉬를 제거하는 공격을 생각할 수 있으나, 일반인의 이러한 공격을 가정하기에는 무리가 있으므로 높은 강인성을 갖는 알고리즘이다. 하지만, Ryutarou의 알고리즘을 RP 시스템에 적용할 경우, 부가적인 메쉬의 추가로 인하여 전체 메쉬를 STL 표준[10]에 부합하도록 변경해야 하는 오버헤드가 발생한다. 예를 들면, 그림 3과 같은 형태가 되어서는 안 된다. 게다가, 많은 3D 형상이 워터마크를 추가할 수 있는 완만한 곡면을 가지고 있지 않다. 따라서, 표준 STL 3D 형상에 일반적으로 적용할 수 없을 뿐만 아니라, 삽입된 메쉬가 원형상을 왜곡시키므로 정밀도에 문제를 일으킬 가능성이 높다.

Benedens[8]의 알고리즘은 3D 형상의 법선 영역에 워터마크를 삽입한다. 3D 형상을 구성하는 중심을 기준으로 각 메쉬의 법선을 계산한다. 법선벡터 값을 위치를 이용하여 워터마크 비트를 결정하는 알고리즘이다. 이런 까닭에 Benedens의 알고리즘은 워터마크의 비가시성을 만족하며, 3D 형상의 변형이 전혀 없으므로 RP 시스템에서 사용할 수도 있다. 하지만, 법선벡터를 모두 변경하는 공격이나 꼭지점 저장 순서변경 공격, 메쉬 저장 순서를 변경하는 공격에는 워터마크가 손쉽게 제거되는 단점을 가지고 있다.

Praun[9]의 알고리즘은 가우스 분포(Gaussian Distribution)에서 추출한 실수 형의 계수(coefficient) 값에 워터마크를 삽입하는 까닭에 외부의 공격에 강인한 특성을 갖는다. 3D 형상에서 메쉬들은 저장 위치가 고정되어있지 않으므로 워터마크를 삽입한 후, 워터마크를 추출하기 위해서는 메쉬들의 저장 위치가 변경되지 않아야 한다. Praun의 알고리즘은 메쉬들의 저장 순서에 종속성을 제거하기 위해 메쉬 연결(mesh connectivity)을 재구성한다. Praun의 알고리즘은 주파수 영역에 워터마크를 삽입함으로 여러 가지 다양한 공격에 강인성을 가진다. 하지만, Praun의 알고리즘은 3D 형상을 결정하는 꼭지점 좌표를 변형하므로 HVS로 감지할 수 없는 형상의 변형이 발생한다. 따라서, 높은 정밀도를 필요로 하는 RP 시스템에 적용하기 어렵다.

그림 2는 STL 표준에 부합하는 형태이고, 그림 3은 STL 표준을 벗어난 형태이다. STL 표준에 의하면 한 패싯의 변에 다른 패싯의 꼭지점이 존재해서는 안 된다. RP 시스템은 레이어(layer)를 적층하여 시제품을 제작하므로 그림 3과 같은 패싯은 레이어(layer)에서의 절삭기의 쾌적(locus) 추적이 꼭지점 b와 c가 이루는 선분에서 중지된다[12]. 따라서 임의의 패싯을 삽입하여 워터마킹하는 Ryutarou의 알고리즘은 많은 오버헤드(overhead)가 발생할 뿐만 아니라, 그 알고리즘이 적용할 경우, 모든 패싯에 대하여 STL 표준에 부합하는지의 여부도 검증해야 한다.

지금까지 기존에 연구된 3D 워터마킹 알고리즘을 RP 시스템의 STL 형식에 적용할 경우에 발생할 수 있는 문제점들을 살펴보았다. 따라서, STL에 워터마킹을 적용하기 위해서는 새로운 패러다임(paradigm)이 필요하다. 본 논문은 RP 시스템에 적용할 수 있는 새로운 워터마킹 알고리즘과 그것의 제약점과 특성들에 대한 연구이다.

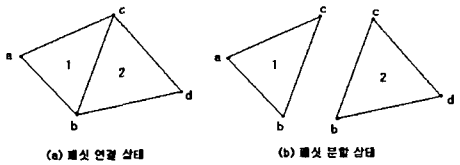


그림 2. 표준에 부합하는 STL

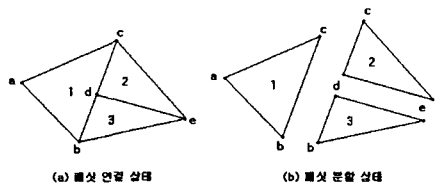


그림 3. 표준에 부합하지 않는 STL

3. STL의 구조와 특징

3.1 STL의 구조

STL로 구성된 3D 형상은 그림 4에서와 같이 패싯

이라 불리는 삼각형들로 구성된다. 그림 4와 같이, STL 파일은 아스키(ASCII)와 이진(binary) 형식 두 가지가 있다.

그림 4와 같이 STL은 3D 형상을 구성하는 수많은 패싯들로 구성되며 각 패싯에 대한 정보를 가지고 있다. 두 형식에서 가장 큰 차이점은, 이진 STL 형식은 전체 패싯의 개수에 대한 정보를 가지고 있다는 점이다. 아스키 STL 형식에는 패싯의 개수에 대한 정보가 없으므로 워터마킹 시스템에서 파악해야 한다.

아래의 그림 5는 STL 3D 형상 모델의 화면 표시 방식을 나타낸다.

3.2 STL의 메쉬(패싯)의 구조

본 절에서는 3D 형상을 구성하는 STL의 패싯에 대하여 알아본다. 그림 6을 통해 알 수 있듯이, 하나의 패싯에는 법선벡터(normal vector)를 나타내는 한 개의 좌표와 패싯의 꼭지점(V_1, V_2, V_3)을 나타내는 세 개의 좌표가 있다. 각 좌표마다 x, y, z의 좌표 값을 저장해야 하므로 총 12개의 좌표 값이 필요하며, 이 값들은 모두 실수이다.

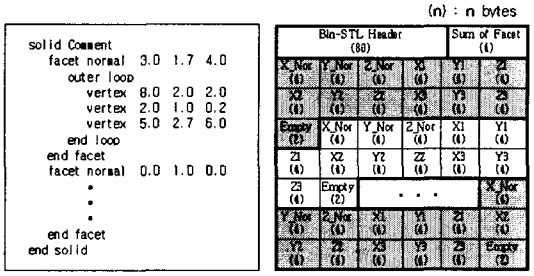


그림 4. STL의 구조

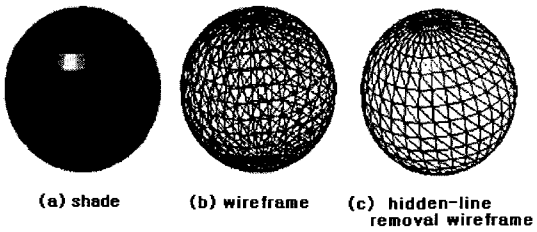


그림 5. STL 3D 형상 (sphere.stl)

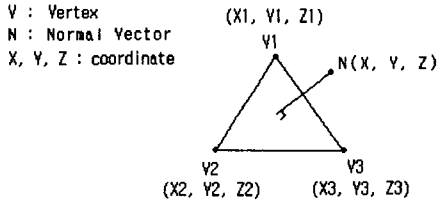


그림 6. STL의 패싯

3.3 STL의 법선벡터

STL에서 패싯으로 표현되는 3D 평면은 관찰자의 시점(viewpoint)에 따라 앞면과 뒷면, 윗면과 아랫면, 또는 안쪽이나 바깥쪽이 된다. 3D 형상의 표면을 표시하는 경우에는 항상 바깥쪽을 화면 표시해야 하므로 패싯의 표면(surface)에 관한 정보가 필요하다. 이 표면에 관한 정보가 패싯의 법선이다. STL 형식에서 법선의 정의는 패싯 평면에서 시작하여 수직으로 평면 위에 있는 임의의 한 점이다. 이 법선 좌표 값에 오류가 있을 경우, STL 표준에 맞춰 제작된 시스템에서는 3D 형상이 정상적으로 표시되지 않는다.

3.4 STL의 워터마크 삽입시의 제약점

2D 영상의 경우 변형된 한 픽셀이 전체 영상에 미치는 영향이 작지만, 3D 형상의 경우에는 꼭지점 좌표가 변경될 경우, 원래의 3D 형상에 왜곡을 가져와 RP 시스템에서 사용하지 못할 수도 있다. STL 3D 형상 데이터에는 한 꼭지점을 공유하는 패싯이 반드시 3개 이상이다. 따라서 임의로 한 꼭지점의 좌표만을 변경할 경우, 3D 형상에 공백이 생기게 된다. 이런 공백은 RP 시스템의 단면정보 추출기(slicer)가 레이저의 궤적을 탐색하는 과정에서 문제를 일으킨다[12].

STL 형식에서 하나의 패싯에는 워터마크 삽입에 사용될 수 있는 두 개의 영역이 있다. 첫 번째는 꼭지점 좌표를 저장하는 영역이고 두 번째는 법선 좌표를 저장하는 영역이다. 꼭지점 좌표 저장 영역의 수치를 변경할 경우에는 3D 형상에 왜곡이 발생하여 3D 원형상이 변형될 가능성이 높다. 실수(float)로 저장되는 꼭지점 좌표에 소수점 이하의 아주 작은 수치를 가감하여 양자화하는 방법을 생각할 수도 있으나, 시제품으로 제작될 크기를 알 수 없으므로 원형상이 왜곡되고 정밀도가 떨어질 가능성이 있다.

3D STL 워터마킹 시스템을 구현할 경우, 주의해야 할 사항은 패싯의 저장 순서에 종속적이어서는 안 된다는 점이다. 보통 한 패싯에 한 비트의 정보를 저장하게 되는데, 패싯의 저장 순서가 뒤바뀌는 경우, 삽입된

워터마크가 모두 사라지게 된다. 본 연구에서 제안하는 워터마크 삽입 알고리즘은 패싯의 저장순서에 종속성(dependency)이 없다.

4. STL에서의 워터마크 삽입과 추출

4.1 삽입 알고리즘의 설계와 구현

본 논문에서는 STL의 패싯 오류를 찾아 자동으로 정정하는 검증기(verifier)를 구현하여 워터마크 삽입 이전에 STL 파일을 검증하도록 설계하였다. 구현된 시스템에 입력된 STL 파일은 검증기를 통해 검증된 후, 정렬기(sorter)에서 패싯의 크기(면적)에 따라 오름차순으로 정렬된다.

RP 시스템에서 실물로 제작되어 산업현장에서 사용되는 STL은 높은 정밀도를 요구한다. STL이 이런 특수한 목적으로 사용되는 까닭에 꼭지점의 좌표를 변경하는 공격은 STL의 유용성을 떨어뜨린다. 기존의 3D 워터마킹 알고리즘을 이용하여 STL에 워터마크를 삽입할 경우, 정밀도가 낮아지거나 패싯(메쉬)의 저장 순서를 변경하는 간단한 공격에도 워터마크가 제거되는 문제점을 가진다. 이러한 문제점을 해결하기 위해 본 논문에서 제안한 알고리즘은 패싯의 면적에 따라 패싯을 정렬한 다음, 난수 발생기에서 발생된 수에 해당하는 순서의 패싯에 워터마크를 삽입한다. 오름차순으로 정렬한 후, 워터마크를 삽입함으로서 패싯의 저장순서를 변경하는 공격에 강인성을 갖도록 하였다. 즉, 워터마크가 패싯의 저장 순서에 종속적이지 않다.

정렬된 패싯은 사용자의 키 값으로 생성된 난수들과 함께 워터마크 인코더로 전달된다. 먼저 꼭지점 영역에 워터마크를 삽입한 후, 법선 영역에 워터마크를 삽입한다. 워터마크를 삽입한 후에는 입력시의 패싯 저장 순으로 복원하여 출력한다. 그림 7은 제안된 알고리즘으로 구현한 시스템의 블록 다이아그램(block diagram)이다.

그림 7의 블록 다이아그램처럼 본 논문에서 제안하는 워터마크 삽입 알고리즘은 법선 영역과 꼭지점 영역에 워터마크를 삽입한다. 워터마크 삽입은 반드시 꼭지점 영역부터 수행해야 한다. 만약, 수행 순서가 뒤바뀌어 법선 영역부터 워터마킹하면, 꼭지점 영역에 워터마크를 삽입하는 동안, 법선 영역에 삽입했던 워터마크가 사라진다.

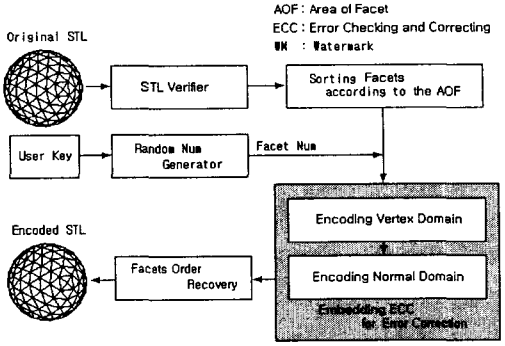


그림 7. 워터마크 삽입 블록 다이아그램

구현된 시스템은 외부의 공격으로 몇 비트의 워터마크가 손상되어도 워터마크를 추출할 수 있도록 하기 위해, ECC(error checking and correcting) 코드를 워터마크와 함께 삽입한다.

4.1.1 워터마크 비트를 삽입할 패싯의 선택

일반적으로 STL 3D 형상은 수백에서 수만 개의 패싯들로 구성된다. 모든 패싯에 반복적으로 워터마크를 삽입할 수도 있지만, 패싯의 저장 순서를 변경하는 공격에 대하여 강인성을 갖도록 하기 위하여 전체 패싯을 크기(면적)에 따라 오름차순으로 정렬한다.

정렬한 후에는 사용자의 키 값을 초기값(seed)으로 하여 난수를 발생시킨다. 발생되는 난수 값(N)의 범위는 "$1 \leq N \leq$ 전체 패싯의 수"를 만족하는 정수이다. 발생된 난수에 해당하는 크기 순의 패싯에 워터마크가 삽입되므로 발생되는 난수는 전체 패싯의 개수보다 작아야하며 중복해서 발생되면 안 된다. 발생된 난수들을 패싯의 면적 순으로 정렬했을 때의 순서로 간주하여 해당 패싯에 워터마크를 삽입한다.

제안된 알고리즘은 패싯의 면적 순으로 정렬한 다음, 워터마크를 삽입하였으므로 패싯의 저장 순서에 종속성이 없다. 워터마크 삽입을 수행한 후에는 패싯을 원래의 저장 순서로 복원하여 출력한다.

4.1.2 꼭지점 영역의 워터마킹

3D 형상을 나타내는 세 꼭지점의 좌표를 변경할 경우에는 형상에 왜곡이 발생하게 된다. 이 때문에, 본 연구에서는 세 꼭지점의 좌표가 아니라, STL에 저장되는 꼭지점의 순서로 워터마크 비트를 표현하였다.

패싯을 바라보는 시점을 기준으로 꼭지점의 저장 순서가 반시계방향(CCW, count clockwise winding)

인 경우, 표면(바깥쪽)이고 시계방향(CW, clockwise winding)인 경우, 내면(안쪽)이다. 워터마크 비트가 '1'인 경우에는 패싯을 이루는 세 꼭지점의 저장 순서가 반시계 방향이 되도록 하고, 워터마크 비트가 '0'인 경우에는 시계 방향이 되도록 변경하였다. 그림 8은 꼭지점 영역의 워터마크 삽입 알고리즘이다.

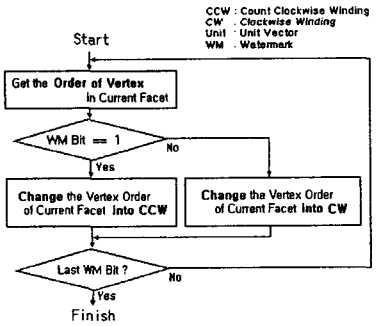


그림 8. 꼭지점 영역의 워터마크 삽입 알고리즘

4.1.3 법선 영역의 워터마킹

STL의 패싯에서 법선은 패싯 평면(표면)에서 수직으로 패싯 평면 위에 존재하는 임의의 점이다. STL에서는 이 법선으로 3D 형상의 표면을 구분한다. 그림 9는 패싯 평면상에 존재하는 법선들의 예이다. 그림 9에서와 같이 패싯 평면상에 많은 법선이 존재하므로 워터마크 비트에 따라 그림 10과 같은 알고리즘으로 워터마크를 삽입한다.

STL 표준에서 법선벡터는 패싯 평면상에 존재하는 임의의 점이다. 표준에서는 법선벡터의 크기나 위치에 관한 제약이 없다[10]. 본 논문에서는 임의의 법선벡터의 시작 위치를 사용하여 워터마크를 삽입한다. 이와 같이 법선벡터를 STL 표준에 맞춰 워터마크 삽입에 사용할 경우, STL을 이용하는 다른 프로세스에서 아무런 문제없이 동작한다. 법선벡터의 크기는

법선벡터를 구할 때, 단위벡터를 계산하므로, 패싯 평면을 기준으로 법선벡터는 항상 1보다 작은 임의의 실수 값을 갖는다.

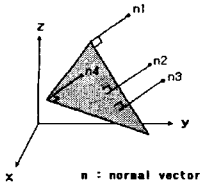


그림 9. 패싯 평면에 존재하는 법선들

그림 10과 같이 제안된 워터마크 삽입 알고리즘을 사용하기 위해서는 패싯 평면의 단위벡터를 구해야 한다. 그림 11에 있는 오른손 법칙을 사용하여 단위벡터 V_3을 얻는다[11].

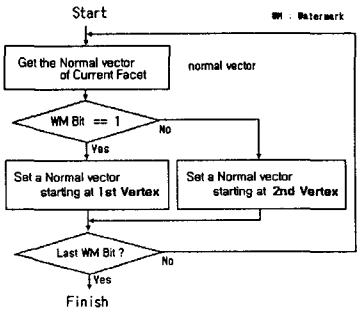


그림 10. 법선 영역의 워터마크 삽입 알고리즘

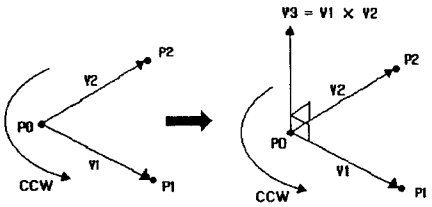


그림 11. 새 점으로 법선 벡터 생성

그림 11에서, 세 점 P_0, P_1, P_2는 패싯의 세 꼭지점을 나타내며, V_3은 P_0을 중심으로 P_1, P_2로 그려지는 패싯에서 생성되는 벡터를 나타낸다. 두 벡터 V_1과 V_2의 외적 V_3은 으로 얻을 수 있다. 두 벡터 V_1과 V_2의

$$\overrightarrow{V_3} = \overrightarrow{V_1} \times \overrightarrow{V_2} \tag{1}$$

외적은 두 벡터에 수직인 단위벡터(unit vector)와 같은 방향이다. 식 (1)에서 구한 벡터는 원점을 기준으로 생성되는 단위벡터이므로 워터마크 비트가 '1'인 경우에는 첫 번째 꼭지점에서 시작되는 법선 벡터로 변경하고, 워터마크 비트가 '0'인 경우에는 두 번째 꼭지점의 시작되는 법선 벡터로 변경한다. 즉, 워터마크 비트에 따라 법선 벡터의 시작 위치를 다르게 하여 워터마크를 삽입한다.

4.2 추출 알고리즘의 설계와 구현

그림 12는 제안된 알고리즘으로 구현된 워터마크 추출기의 설계를 나타내는 블록 다이아그램이다.

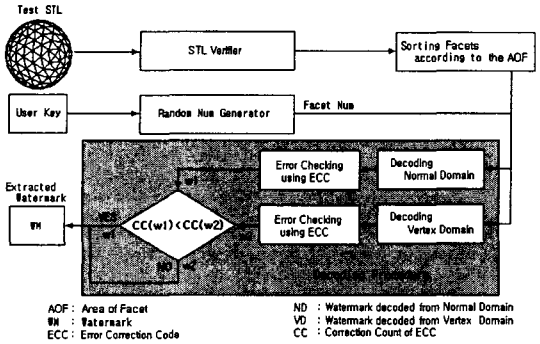


그림 12. 워터마크 추출 블록 다이아그램

하나의 패싯에는 법선과 꼭지점 영역에 워터마크 비트가 삽입된다. 외부의 공격에 하나의 워터마크 비트가 제거되어도 다른 영역의 워터마크 비트는 남아 있도록 설계하였다. 두 가지 방법으로 삽입된 워터마크와 ECC 코드를 추출한다. 두 영역에서 추출된 워터마크를 ECC 코드를 이용하여 오류를 정정한다. 이때, 오류를 정정하는 횟수를 카운트하여 정정 횟수가 작은 방법으로 추출된 워터마크를 디코딩의 결과로 출력한다.

4.2.1 워터마크 비트가 삽입된 패싯의 선별

앞 절에서 워터마크가 삽입될 패싯을 선별하여 워터마크를 삽입했으므로, 워터마크를 추출할 경우에도 워터마크가 삽입된 패싯을 검출한 후, 워터마크 추출을 수행한다. 삽입 패싯을 선정하는데 사용했던 키 값으로 난수 발생기를 초기화한 후, 삽입된 워터마크 비트의 개수와 ECC 코드의 비트 크기만큼 난수를 발생시킨다. 그런 다음, 패싯을 면적 순으로 정렬한다. 발생된 난수를 정렬된 패싯의 크기 순서로 간주하여 난수의 크기에 해당하는 패싯을 선별한다.

4.2.2 꼭지점 영역에서의 워터마크 추출

꼭지점 영역에서 워터마크를 추출하기 위해서 STL 파일에 저장되어 있던 법선 좌표를 기준으로 저장된 꼭지점들이 시계 방향인지 반시계 방향인지 판별한다. 반시계 방향(CCW)인 경우, 워터마크 비트는 '1'이고, 시계 방향(CW)인 경우, 워터마크 비트는 '0'이다. 그림 13은 꼭지점에서의 워터마크 추출 알고리즘이다.

4.2.3 법선 영역에서의 워터마크 추출

법선 영역에서 워터마크를 추출하기 위해, 패싯 평면의 단위 벡터를 계산한다. 이 값을 첫 번째 꼭지점에서 시작하는 벡터로 변경한 후, 좌표 값을 얻는다.

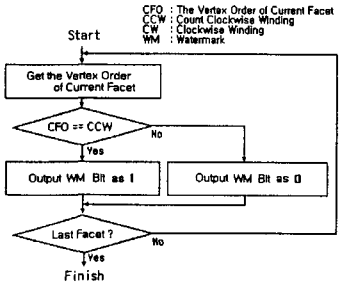


그림 13. 꼭지점 영역에서의 워터마크 추출

이동시킨 좌표 값이 실제 STL 파일에 저장되어 있는 법선 좌표와 일치하면 워터마크 비트는 '1'이고, 그렇지 않으면 '0'이다. 그림 14는 법선에서 워터마크를 추출하는 알고리즘이다.

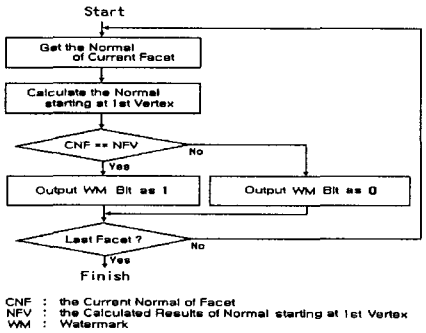


그림 14. 법선 영역에서의 워터마크 추출

5. 성능 평가 및 분석

제안된 알고리즘에서는 워터마크 삽입과 추출 이전에, 모든 패싯을 면적에 따라 오름차순으로 정렬한다. 워터마크는 작게는 수십 개의 비트에서 많게는 수백 개의 비트만으로 구성되므로 워터마크 삽입에 소요되는 대부분의 시간은 패싯을 면적 순으로 정렬하는 데 소요된다.

3D 공간에서 삼각 평면의 면적을 구하는 일은 많은 계산을 필요로 한다. 이런 이유로 제안된 알고리즘에서는 실제 면적을 구하지 않고 패싯을 구성하는 세변의 길이 중 가장 짧은 변을 키 값으로 하여 패싯을 정렬하였다.

워터마크를 삽입하거나 추출하는 경우, 각 삼각형마다 면적을 구해야 되고 삼각형 면적에 따라 패싯을

정렬해야 하므로 삽입 알고리즘과 추출 알고리즘의 시간 복잡도는 각각 $O(nlog_2n)$ 일 것으로 예상된다. 여기에서 n은 전체 패싯의 개수이다.

5.1 실험 환경

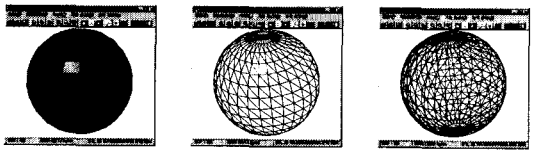
제안된 알고리즘으로 시스템을 구현하여 펜티엄IV PC (Windows XP) 환경에서 실험하였다. 시스템 구현에는 MS Visual C++ 6.0을 사용했고, 화면 표시에는 OpenGL 그래픽 라이브러리를 사용하였다. 워터마크 비트들은 입력된 텍스트 문자열에서 비트들을 추출하였다.

STL 파일에는 많은 오류(법선벡터 수치, 패싯 중복, 공백 등)가 있는 까닭에 워터마크 삽입과 추출에 여러 가지 문제를 일으켰다. 제안된 시스템에서는 검증기(verifier)를 구현하여 워터마크의 삽입 이전에 STL의 오류 유무를 검증하도록 구현하여 오류 발생률을 낮췄다.

5.2 실험 결과

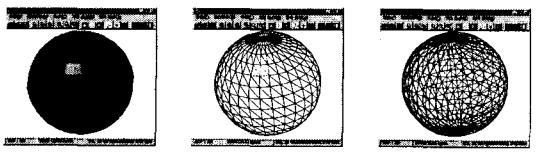
그림 15는 제안된 시스템에서 'sphere.stl' 파일에 워터마크를 삽입하기 직전의 STL 3D 형상이며, 그림 16은 워터마크를 삽입한 후의 3D 형상이다. 그림 15와 그림 16은 간단한 3D 형상 파일인 'sphere28.stl'을 3가지 화면 표시 모드로 디스플레이한 모습이다. 두 그림에서 보이는 것처럼 3D 형상에 시각적인 왜곡이 없을 뿐 아니라, 3D 형상 데이터의 왜곡도 전혀 없다.

그림 17은 제안된 알고리즘으로 "Image Processing Lab."이란 워터마크를 'sphere28.stl' 파일에 삽입한



ⓐ shade ⓑ wireframe ⓒ hidden wireframe

그림 15. 워터마크를 삽입하기 이전의 STL



ⓐ shade ⓑ wireframe ⓒ hidden wireframe

그림 16. 워터마크가 삽입된 STL

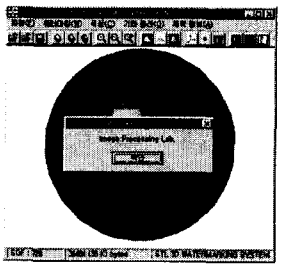


그림 17. 'sphere28.stl'에서 추출된 워터마크

후, 제안된 알고리즘으로 워터마크를 다시 추출해낸 결과를 보여주는 그림이다.

그림 18과 그림 19는 좀더 복잡한 3D 형상에 워터마크를 삽입하고 추출한 화면이다.

ⓐ shade ⓑ wireframe ⓒ hidden wireframe

그림 18. 워터마크가 삽입된 'cow.stl' 파일

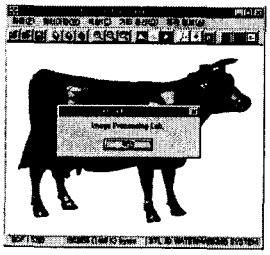


그림 19. 'cow.stl' 파일에서 워터마크 추출 결과

아래의 표 1은 기존의 알고리즘과 제안된 알고리즘을 RP 시스템에 사용될 STL 3D 형상에 적용하였을

표 1. 알고리즘들의 공격에 대한 강인성

알고리즘 \ 공격유형	패싯 저장 순서 변경	꼭지점 저장 순서 변경	꼭지점 좌표 변경	정밀도
Xiaoyang[5]	×	○	×	손실
François[6]	×	×	×	손실
Ryutarou[7]	○	○	○	손실
Benedens[8]	×	×	×	불변
E.Praun [9]	○	○	○	손실
Proposed	○	○	×	불변

때의 워터마크 강인성을 보여준다. 기존의 알고리즘들 중에서 Ryutarou[7]과 Praun[9]의 알고리즘은 다양한 공격에 대하여 높은 강인성을 보였으나 정밀도를 떨어뜨리는 까닭에 RP 시스템에 적용이 곤란한 알고리즘이다. 제안된 알고리즘은 꼭지점의 좌표 변경 공격에 대해서는 워터마크가 쉽게 제거되지만 3D 형상의 변형이 없는 까닭에 높은 정밀도를 유지할 수 있음을 알 수 있었다.

6. 결 론

기존의 3D 형상 모델 워터마킹 알고리즘을 STL 3D 형상에 적용했을 경우의 문제점들을 살펴보았다. 본 연구에서 STL 3D 형상에 워터마크를 삽입할 때 발생하는 문제점들을 개선한 워터마킹 알고리즘을 제안하였다. 제안된 알고리즘으로 3D 형상에 워터마크를 삽입하는 경우에는, 정밀도에 문제를 일으키는 왜곡이나 변형같은 문제가 전혀 발생하지 않았으며, STL에 워터마크를 삽입한 후에도, 워터마킹된 STL이 STL 표준에 부합했다. 따라서, 제안된 알고리즘은 RP를 위한 3D 형상의 워터마킹에 적합하였다.

기존 3D 형상 워터마크 알고리즘으로 3D 형상에 워터마킹할 경우에는 3D 형상을 구성하는 메쉬(패싯)의 순서를 변경하는 공격에 무력하다. 제안된 알고리즘은 패싯의 크기에 따라 오름차순으로 정렬한 후에 키 값으로 생성된 값에 해당하는 크기순의 패싯에 워터마크를 삽입함으로서, 패싯의 저장 순서를 변경하는 공격에 대하여 강인성을 보였다. 제안된 워터마크 알고리즘으로 법선과 꼭지점 영역에 삽입한 워터마크를 제거하기 위해, 무작위 변형 공격을 가할 경우, 3D 형상의 표면이 바뀌어 형상이 변형되고 디지털 컨텐츠로서의 가치가 사라진다. 따라서, 제안된 알고리즘은 외부의 무작위적인 변형 공격에 강인성을 보인다.

참 고 문 헌

[1] J. D. Cawley, A. H. Heuer, W. S. Newman, and B. B. Mathewson, "Computer-Aided Manufacturing of Laminated Engineering Materials," Am. Ceram. Soc. Bull., 75, 1996.

[2] E. A. Griffin, D. R. Mumm, and D. B. Marshall, "Rapid Prototyping of Functional Ceramic Composites," American Ceramic Society Bulletin, Vol. 75, No. 7, pp. 65, 1996.

[3] E. A. Griffin, J. Daufenbach, and S. McMillin, "Desktop Manufacturing : LOM vs Pressing," American Ceramic Society Bulletin, Vol. 73, No. 8, pp. 109, 1994.

[4] R. E. Mistler, "Tape Casting : The Basic Process for Meeting the Needs of the Electronics Industry," American Ceramic Society Bulletin, Vol. 69, No. 6, pp. 1022, 1990.

[5] Xiaoyang Mao, Makoto Shiba, and Atsumi Imamiya, "Watermarking 3D Geometric Models Through Triangle Subdivision," Proceedings of SPIE Vol. 4314, pp. 253~260, 2001.

[6] François Cayre, Benoit Macq, "Spatial watermarking of 3D triangle meshes," Proceedings of SPIE Vol. 4472, pp. 155~166, 2001.

[7] Ryutarou Ohbuchi, Hiroshi Masuda, and Masaki Aono, "Watermarking Three-Dimensional Polygonal Models Through Geometric and Topological Modifications," IEEE journal on selected areas in communications, vol. 16, No. 4, pp. 551~560, May 1998.

[8] Oliver Benedens, "Watermarking of 3D polygon based models with robustness against mesh simplification," Proceedings of SPIE: Security and Watermarking of Multimedia Contents, SPIE, pp.329~340, 1999.

[9] Emil Praun, Hugues Hoppe, and Adam Finkelstein, "Robust Mesh Watermarking," Siggraph 1999, Computer Graphics Proceedings, Addison Wesley Longman, LA., pp. 49~56, 1999.

[10] 3D Systems. Inc. "StereoLithography Interface Specification," Oct. 1988.

[11] Yong Zheng, "Enabling Computational Techniques For Tangential-Building Solid Freeform Fabrication," Doctoral Dissertation, Dept. of Electrical Engineering and Applied Physics, Case Western Reserve Univ., 1997.

[12] 김기석, 천인국, "RP를 위한 3D STL에서의 단면 정보 추출," 춘천멀티미디어 학술회의, pp. 455~459, 2001.

[13] E. A. Griffin, D. R. Mumm, and D. B. Marshall, "Rapid Prototyping of Functional Ceramic Composites," American Ceramic Society Bulletin, Vol. 75, No. 7, pp. 65, 1996.

[14] R. Ohbuchi, H. Masuda and M. Aono, "Watermarking Three-Dimensional Polygon Models," *Proceedings of the ACM Multimedia '97*, pp. 261-272, Seattle, Washington, USA, November, 1997.

[15] R. Ohbuchi, H. Masuda and M. Aono, "Watermarking Three-Dimensional Polygon Models Through Geometric and Topological Modifications," *IEEE Journal on Selected Areas in Communications*, pp. 551-559, 1998.

[16] O. Benedens, "Geometry-Based Watermarking of 3D Models," *IEEE Computer Graphics and Applications*, Vol. 19, No. 1, pp. 46-55, 1999.

김 기 석

1997년 2월 청운대학교 전자계산학과 (공학사)
1999년 2월 순천향대학교 전산학과 (공학석사)
1999년 3월~현재 순천향대학교 전산학과 박사과정

E-mail : paperlion@dreamwiz.com
관심분야 : 영상처리, 컴퓨터비전, 워터마킹, 3D 그래픽스, RP 시스템

천 인 국

1983년 2월 서울대학교 전자공학과 (공학사)
1985년 2월 KAIST 전기 및 전자공학과 (공학석사)
1985년~1988년 삼성전자 종합연구소 연구원
1993년 2월 KAIST 전기 및 전자공학과 (공학박사)
1993년 3월~현재 순천향대학교 정보기술공학부 교수
E-mail : chunik@sch.ac.kr
관심분야 : 영상처리, 컴퓨터비전, 워터마킹, RP 시스템, 3D 그래픽스, 인터넷 상품추천 시스템, 모바일 컴퓨팅