버텍스 영역을 이용한 STL에서의 3차원 디지털 워터마킹

김기석 · 천인국

요 약

본 논문은 신속조형(RP: rapid prototyping) 시스템에서 사용되며 3D 기하학적 형상 모델을 가지고 있는 STL(standard transform language)에 워터마크를 삽입하는 방법에 관한 연구이다. 제안된 알고리즘은 3D 형상 데이터의 왜곡이 전혀 없이 패싯(facet, mesh)의 버텍스(vertex) 영역에 워터마크를 삽입한다. 기존의 알고리즘으로 STL에 워터마크를 삽입할 경우, 패싯의 저장순서를 변경하는 단순한 공격으로도 워터마크가 제거된다. 제안된 알고리즘은 패싯의 저장 순서 변경과 같은 공격에 대한 강인성을 가질 뿐만 아니라, 비가시성(invisibility)도 충족한다. 제안된 알고리즘으로 STL 3D 형상에 워터마크를 삽입하고 추출하는, 실험 결과들은 3D 원형상을 전혀 왜곡하지 않고 워터마크의 삽입과 추출이 가능함을 보여준다.

A 3D Watermarking on STL using Vertex domain

Gee-Seog Kim · In-Gook Chun

ABSTRACT

This paper is a research about method, that is used in Rapid Prototyping system, that inserts and extracts watermark in STL(standard transform language) that has a 3D geometrical model. The proposed algorithm inserts watermark in the vertex domain of STL facet without the distortion of 3D model. If we make use of a established algorithm for watermarking of STL, a watermark inserted to 3D model can be removed by simple attack that change order of facet. The proposed algorithm has robustness about these attack. Experiment results verify that the proposed algorithm, to encode and decode watermark in STL 3D geometrical model, doesn't distort a 3D model at all. And it shows that the proposed algorithm is available.

Key words : Rapid Prototyping, 3D Geometric Model, STL, Watermarking, Vertex

1. 서론

최근에 통신망의 발달로 정보 교환이 신속하게 이루어지고 있고, 멀티미디어 자료의 사용이 증가되고 있다. 하지만, 디지털 자료는 복제와 조작이 용이하고 복제된 자료는 원본과 동일하기 때문에 저작권에 다는 부작용이 따른다. 이런 까닭에 DRM(digital rights management)의 한 요소기술로서 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 모델(STL)에 워터마크를 삽입하려는 연구는 미비한 실정이다.

RP는 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 설계상의 오류나 부적합한 요인을 조기에 발견하는 것이다[1].

이러한 RP 시스템에서의 자료 교환 표준이 바로 STL 파일 형식이며 STL 파일로 표현되는 3D 기하학적 형상 모델은 시제품 제작뿐만 아니라 SF 영화, 3D 애니메이션 등에 이용할 수도 있다. 이러한 응용을 위해, 실물을 3D 스캐닝 하거나 CAD 시스템으로 설계하여 STL 파일을 제작하는 데에는 부가적인 비용이 발생한다.

3D 형상에 워터마크를 삽입하는 기존의 연구들은 RP 시스템에서 사용하기에는 부적합한 알고리즘이다. Xiaoyang[2], Ryutarou[3] 등이 제안한 알고리즘은 3D 형상에 워터마크를 삽입하지만, 3D 형상에 HVS(human visual system)로 감지할 수 없는 잡음을 첨가하는 기법이다. RP 시스템은 시제품을 만들어 제품 생산 이전의 검증을 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. 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 가상현실, 비디오 게임, 의료 영상, 애니메이션, 시뮬레이션 등과 같은 시스템에서 사용하여 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 시스템에서 사용되는 STL에 워터마크를 삽입하는 알고리즘은 실물의 형상에 어떠한 왜곡도 발생시켜서는 안 된다. 생산 현장에서 사용되는 부품일 경우에는 더욱 큰 정밀도를 가져야 한다[4].

Xiaoyang의 알고리즘을 STL에 적용할 경우, 패싯의 저장 순서를 변경하는 간단한 공격에는 강인성을 가진다. 이 알고리즘은 한 비트의 워터마크 삽입을 위해 주위에 있는 3개 이상의 메쉬(mesh)들의 버텍스를 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. 또한, 버텍스의 저장 순서를 변경하는 간단한 공격에 워터마크가 제거되므로, 3D STL 형상의 워터마킹에는 적합하지 않다.

François의 알고리즘은 메쉬를 이루는 세 버텍스 중에서 임의로 한 버텍스를 선택한 후, 나머지 두 버텍스와 이루는 선분에 수선을 그렸을 때의 위치로 워터마크를 삽입하는 알고리즘이다. 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. 하지만, Ryutarou의 알고리즘을 RP 시스템에 적용할 경우, 부가적인 메쉬의 추가로 인하여 전체 메쉬를 STL 표준[4]에 부합하도록 변경해야 하는 오버헤드가 발생한다(그림 1(a)와 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.

그림 1은 STL 표준을 벗어난 형태이다. STL 표준에 의하면 한 패싯의 변에 다른 패싯의 버텍스가 존재해서는 안 된다. RP 시스템은 레이어(layer)를 적층하여 시제품을 제작하므로 그림 1과 같은 패싯은 레이어(layer)에서의 레이저 궤적(laser locus) 추적이 버텍스 b와 c가 이루는 선분에서 중지된다[7]. 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. 본 연구에서는 RP 시스템에 적용 가능한 새로운 워터마킹 알고리즘을 제안한다.

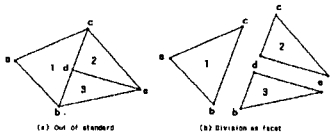
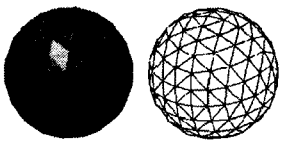


그림 1. STL 표준에 위배되는 경우

3. STL의 구조와 특징

3.1 STL의 구조

STL로 구성된 3D 형상 모델은 그림 3에서와 같이 패싯이라 불리는 삼각형들로 구성된다.



Ⓐ shade mode Ⓑ wire frame mode

그림 0. STL 3D 형상 (sphere.stl)

그림 3는 STL의 두 가지 파일 형식을 나타낸 것이다.

```
solid Comment
   facet normal  3.0  1.7  4.0
      outer loop
         vertex  8.0  2.0  2.0
         vertex  2.0  1.0  0.2
         vertex  5.0  2.7  6.0
      end loop
   end facet
   facet normal  0.0  1.0  0.0
      .
      .
      .
   end facet
end solid
```

Ⓐ ASCII STL

Bin-STL Header (80)					Sum of Facet (4)
X_Nor (4)	Y_Nor (4)	Z_Nor (4)	X1 (4)	Y1 (4)	Z1 (4)
X2 (4)	Y2 (4)	Z2 (4)	X3 (4)	Y3 (4)	Z3 (4)
Empty (2)	X_Nor (4)	Y_Nor (4)	Z_Nor (4)	X1 (4)	Y1 (4)
Z1 (4)	X2 (4)	Y2 (4)	Z2 (4)	X3 (4)	Y3 (4)
Z3 (4)	Empty (2)	• • •			X_Nor (4)
Y_Nor (4)	Z_Nor (4)	X1 (4)	Y1 (4)	Z1 (4)	X2 (4)
Y2 (4)	Z2 (4)	X3 (4)	Y3 (4)	Z3 (4)	Empty (2)

Ⓑ Binary STL

그림 3. STL의 구조

그림 3에서와 같이 STL은 3D 형상을 구성하는 수많은 패싯들에 대한 정보를 가진다. 두 형식에서 가장 큰 차이점은, 바이너리 STL은 전체 패싯의 개수에 대한 정보를 가지고 있다는 점이다. 아스키 형식에는 패싯의 개수에 대한 정보가 없으므로 워터마킹 시스템에서 파악해야 한다.

3.2 STL의 메쉬(패싯)의 구조

본 절에서는 3D 형상을 구성하는 STL의 패싯에 대하여 알아본다.

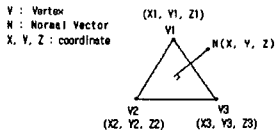


그림 4. STL의 패싯

그림 4에서와 같이 하나의 패싯은 3D 공간 좌표 3개와 1 개의 법선 좌표를 가진다. 각 좌표마다 x, y, z의 좌표 값을 저장해야 하므로 총 12개의 좌표 값이 필요하며, 이 값들은 모두 실수이다.

3.3 STL의 법선(normal)

STL에서 패싯으로 표현되는 3D 평면은 관찰자의 시점에 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 삽입 제약점

2D 영상의 경우 변형된 한 픽셀이 전체 영상에 미치는 영향이 작지만, 3D 형상의 경우에는 버텍스 좌표가 변경될 경우, 원래의 3D 형상에 왜곡을 가져와 RP 시스템에서 사용하지 못할 수도 있다. STL 3D 형상 데이터에는 한 버텍스를 공유하는 패싯이 반드시 3개 이상이다. 따라서 임의로 한 버텍스의 좌표만을 변경할 경우, 3D 형상에 공백(empty space)이 생기게 된다. 이런 공백은 RP 시스템의 단면정보 추출기(slicer)가 레이저의 패적을 탐색하는 과정에서 문제를 일으킨다.

버텍스 좌표 저장 영역의 수치를 변경할 경우에는 3D 형상에 왜곡이 발생하여 3D 원형상이 변형될 가능성이 높다. 실수(float)로 저장되는 버텍스 좌표에 소수점 이하의 아주 작은 수치를 가감하여 양자화 하는 방법을 생각할 수도 있으나, 시제품으로 제작될 크기를 알 수 없으므로 원형상이 왜곡될 가능성이 있다.

3D STL 워터마킹 시스템을 구현할 경우, 주의해야 할 사항은 패싯의 저장 순서에 종속적이어서는 안 된다는 점이다. 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

STL 패싯 자체적으로 가질 수 있는 오류를 찾아 자동으로 정정하는 검증기(verifier)를 구현하여 워터마크 삽입 이전에 STL 파일을 검증하도록 설계하였다. 그리고 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 ≤ N ≤ 전체 패싯의 수"를 만족하는 정수이다. 발생된 난수에 해당하는 크기 순의 패싯에 워터마크가 삽입되므로 발생되는 난수는 전체 패싯의 개수보다 작아야 하며 중복해서 발생되면 안된다. 그림 5는 본 논문에서 제안하는 알고리즘으로 구현된 시스템의 워터마크 삽입 다이아그램(diagram)이다.

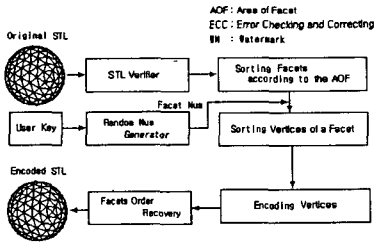


그림 5. 워터마크 삽입 다이아그램

4.1.1 워터마크 비트를 삽입할 패싯의 선택

일반적으로 STL 3D 형상은 수백에서 수만 개의 패싯들로 구성된다. 모든 패싯에 반복적으로 워터마크를 삽입할 수도 있지만, 제한된 알고리즘에서는 저작권자의 키 값을 입력받아 이 값으로 난수 발생기를 초기화 한다. 임의의 난수를 워터마크 비트의 개수만큼 생성한 다음, 발생된 난수들을 패싯의 면적 순으로 정렬했을 때의 순서로 간주하여 워터마크를 삽입한다.

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

4.1.2 버텍스 영역의 워터마킹

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

패싯을 바라보는 시점(viewpoint)을 기준으로 버텍스의 저장 순서가 반시계방향(CCW, countclock wise winding)인 경우, 표면(바깥쪽)이고 시계방향(CW, clockwise winding)인 경우, 내면(안쪽)이다. 워터마크 비트가 '1'인 경우에는 패싯을 이루는 세 버텍스의 저장 순서가 반시계 방향이 되도록 하고, 워터마크 비트가 '0'인 경우에는 시계 방향이 되도록 변경하였다. 그림 6은 버텍스 영역의 워터마크 삽입 알고리즘이다.

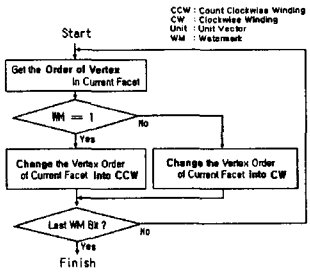


그림 6. 버텍스 영역의 워터마크 삽입 알고리즘

4.2 추출 알고리즘의 설계와 구현

그림 7은 제안된 알고리즘으로 구현된 시스템의 설계를 나타내는 다이아그램이다.

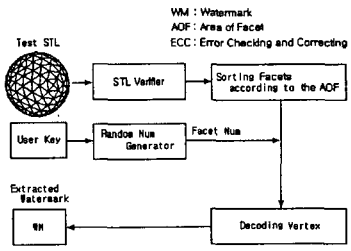


그림 7. 워터마크 추출 다이아그램

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 개수만큼 난수를 발생시킨다. 그런 다음, 패싯을 면적 순으로 정렬한다. 발생된 난수를 정렬된 패싯의 크기 순서로 간주하여 난수의 크기에 해당하는 패싯을 선별한다.

4.2.2 버텍스 영역에서의 워터마크 추출

버텍스 영역에서 워터마크를 추출하기 위해서 STL 파일에 저장되어 있던 법선 좌표를 기준으로 저장된 버텍스들이 시계 방향인지 반시계 방향인지 판별한다. 반시계 방향(CCW)인 경우, 워터마크 비트는 '1'이고, 시계 방향(CW)인 경우, 워터마크 비트는 '0'이다. 그림 8은 버텍스에서의 워터마크 추출 알고리즘이다.

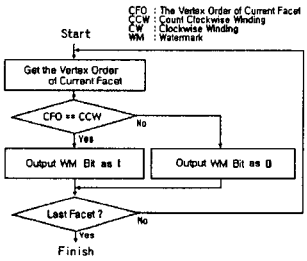


그림 8. 버텍스 영역에서의 워터마크 추출

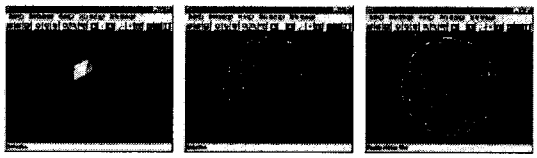
5. 성능 평가 및 분석

5.1 실험 환경

제안한 알고리즘으로 시스템을 구현하여 펜티엄Ⅳ PC (1.4GHz, 256M, Windows XP) 환경에서 실험하였다. 시스템 구현에는 MS Visual C++ 6.0을 사용했고, 화면 표시에는 OpenGL 그래픽 라이브러리를 사용하였다. 워터마크 시그널은 텍스트 문자열을 사용하였으며 입력된 문자열에서 비트들을 추출하여 패싯에 삽입하였다.

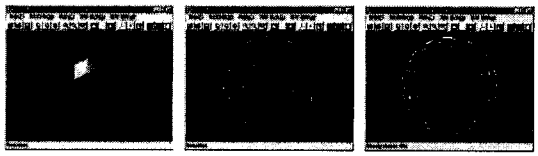
STL 파일에는 많은 오류(법선벡터 수치, 패싯 중복, 공백 등)가 있는 까닭에 워터마크 삽입과 추출에 여러 가지 문제를 일으켰다. 제안된 알고리즘으로 구현한 시스템에서는 검증기(verifier)를 구현하여 워터마크의 삽입 이전에 STL의 오류 유무를 검증하도록 구현하여 오류 발생률을 낮췄다.

그림 9는 제안된 알고리즘으로 구현된 워터마킹 시스템에서 'sphere.stl' 파일에 워터마크를 삽입하기 직전의 STL 3D 형상이며, 그림 10은 워터마크를 삽입한 후의 3D 형상이다. 'sphere.stl' 파일은 간단한 3D 형상이고 이 형상을 구성하는 패싯들의 크기가 거의 일정한 파일이다. 이런 형상에서는 제안된 알고리즘을 적용할 경우, 정상적으로 동작했다. 그림 9와 10에 2가지 모드로 화면 표시한 모습이다. 두 그림에서 보이는 것처럼 3D 형상에 시각적인 왜곡이 없을 뿐 아니라, 내부적인 3D 형상 데이터의 왜곡도 전혀 없다.



ⓐ shade ⓑ wire frame ⓒ hid-wire frame

그림 9. 워터마크 삽입 이전의 STL



ⓐ shade ⓑ wireframe ⓒ hid-wire frame

그림 10. 워터마크 삽입 이후의 STL

그림 11은 제안된 알고리즘으로 "Image Processing Lab. "이란 워터마크를 'sphere.stl' 파일에 삽입한 후, 제안된 알고리즘으로 워터마크를 다시 추출해낸 그림이다.

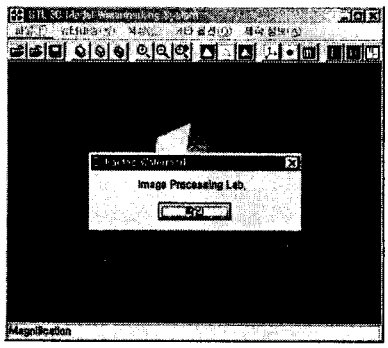


그림 11. 'sphere.stl'에서 추출된 워터마크

하지만, 여러 가지 크기의 패싯을 갖는 복잡한 3D 형상인 경우에는 정상적으로 찾지 못하는 경우가 발생한다. 그림 12에 보이는 3D 형상에 워터마크를 삽입하는 경우에는 워터마크를 완벽하게 검출하지 못했다. 워터마크 삽입을 시작하는 위치에 따라 검출이 완전하게 수행될 수도 있고 또 완전하게 검출하지 못할 수도 있었다. 워터마크 비트가 검출되지 않는 이유를 알아내기 위해 모든 패싯에 워터마크를 삽입한 후, 모든 패싯에서 워터마크를 검출해 보았다. 그림

13은 그림 12에 보이는 'cow.stl' 파일의 모든 패싯에 워터마크를 삽입한 후, 워터마크를 검출한 결과이다.

6. 결론

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 않았다. 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.

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

[3] 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

[4] 3D Systems. Inc. "StereoLithography Interface Specification", Oct. 1988