한국 CAD/CAM 학회 논문집
제 12권 제 2호 2007년 4월 pp. 109-117

학술논문

옥트리 인코딩을 이용한 법선 벡터의 압축

김용주*, 김재정**

Compression of Normal Vectors using Octree Encoding

Kim, Y. J.* and Kim, J. J.**

ABSTRACT

Three-dimensional mesh models have been widely used in various applications such as simulations, animations, and e-catalogs. In such applications the normal vectors of mesh models are used mainly for shading and take up the major portion of data size and transmission time over networks. Therefore a variety of techniques have been developed to compress them efficiently. In this paper, we propose the MOEC (Modified Octree Encoding Compression) algorithm, which allows multi-level compression ratios for 3D mesh models. In the algorithm, a modified octree has nodes representing their own positions and supporting a limited depth of the tree so that the normal vectors are compressed up to levels where the shading is visually indistinguishable. This approach provides efficiency in compressing normals with multi-level ratios, without additional encoding when changing in compression ratio is required.

Key words : 3D mesh model, Normal vector, Octree encoding

1. 서 론

동시 공학(Concurrent Engineering) 환경하에서는 Fig. 1에서 보는 것처럼 각 도메인들이 지리적으로 떨어져 있기 때문에 3D 모델 데이터 교환 시 네트워크를 통해 데이터를 주고 받는 일이 많아지고 있다.

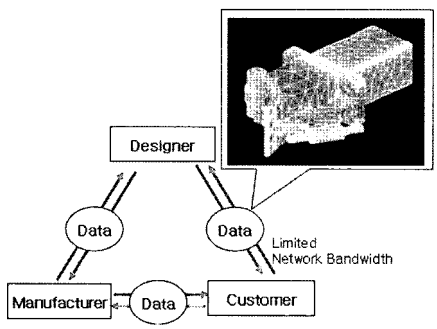
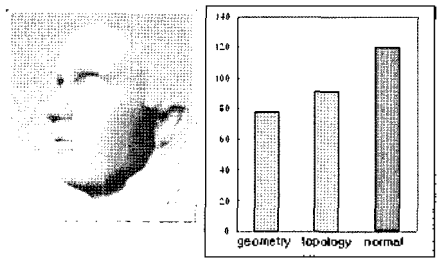


Fig. 1. 3D mesh data transmission through the network.

3D CAD 데이터에는 형상정보 외에도 많은 정보들이 포함될 수 있다. 그러나, 네트워크의 최대 전송 대역폭(bandwidth)이 한정되어 있으므로 해석과 가시화를 위해 좀더 가벼운 형태의 메쉬(mesh) 모델을 이용하고 있다. 메쉬 모델은 점 데이터의 좌표 값을 나타내는 지오메트리(Geometry)와 그 연관 관계를 인덱싱 하는 타폴로지(Topology), 법선 벡터(Normal Vector)의 정보로 구성된다. 지오메트리와 타폴로지의 경우에는 전송 속도를 높이기 위해 다양한 압축 알고리듬을 적용하여 용량이 줄어든 형태의 파일로 변환하여 전송하는 방법이 적용되고 있다. 법선 벡터의 경우 Fig. 2에서 보듯이 지오메트리나 타폴로지보다 파일에서 차지하는 용량이 더 크다. 따라서, 법선 벡터를 압축하여 전송하는 것이 시간을 줄이는 데에 효율적이며 최근에는 그에 관한 연구가 진행되어왔다[7-10].

일반적으로 법선 벡터 압축 알고리듬은 원래의 법선 벡터 값을 몇 개의 대표 값으로 클러스터링 하여 인덱스를 부여하는 방식으로 압축하기 때문에 원래 법선 벡터와 압축된 법선 벡터 사이에는 각도 편차(Angular Deviation)가 발생한다.

*교신저자, 현대기아자동차 남양연구소
**종신회원, 한양대학교 기계공학부
- 논문투고일: 2006. 02. 24
- 심사완료일: 2007. 03. 02



(a) Face model (b) Data size

Fig. 2. Sizes of geometry, topology and normal in 3D mesh model data.

이 각도편차의 크기가 법선 벡터의 압축률을 좌우한다. 법선 벡터의 압축률이 커질수록 파일의 용량은 더욱 줄어드는 이점이 있으나 음영처리(shading)를 할 때 시각적 왜곡도 비례하여 커지는 단점이 있다.

최근에는 3D 형상모델의 활용도가 높아짐에 따라 하나의 3D 모델이 여러 응용분야에 사용되는 일이 늘고 있다. Fig. 3에서 보는 바와 같이 하나의 3D 형상모델도 응용분야에 따라 렌더링에서 요구되는 시각적인 품질(quality)이 다르다. 형상 모델의 시각적인 품질을 높이는 데에는 여러 가지 요소가 관련되어 있다. 그 중, 법선 벡터만을 고려한다면, 애니메이션(animation)과 같이 섬세한 음영처리를 요하는 경우도 있으나 시뮬레이션(simulation)이나 전자 카탈로그(e-catalog)와 같은 응용분야에 사용되는 형상모델은 가시화를 할 때 형상 정보에 비해 상대적으로 음영처리의 중요성이 그리 크지 않다.

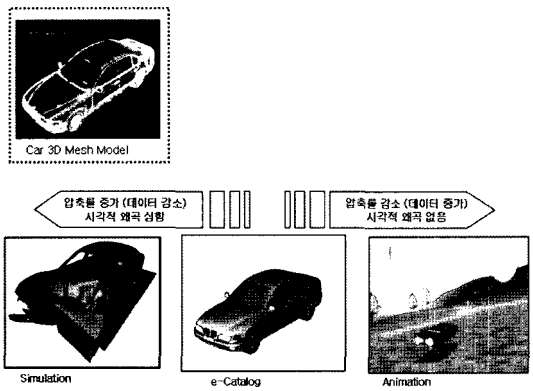


Fig. 3. Car model needed for various shading quality.

3D 형상모델이 각각의 응용분야에 적용되는 어플리케이션은 대개 지리적으로 떨어져 있다. 이러한 경우 네트워크의 대역폭을 고려하여 효율적으로 파일을 전송하는 일이 필요하다. 따라서 형상모델을 전송할 때 네트워크의 대역폭이 한정되어 있기 때문에 다양한 시각적인 품질(quality)이 요구되는 각각의 응용분야에 걸맞게 법선 벡터를 압축하여 용량이 적은 파일을 전송하는 것이 시간 절약에 유리하다.

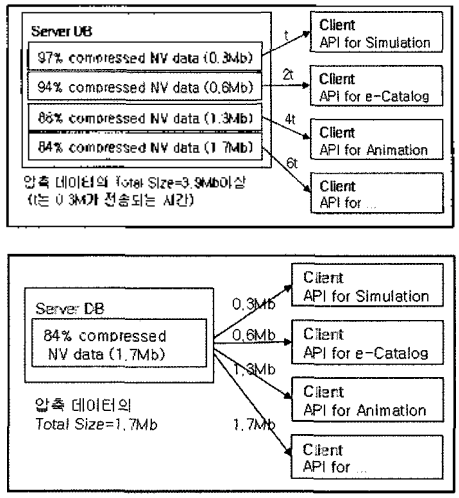


Fig. 4. Normal data transmission using conservative compression algorithm (up) and suggested compression algorithm (down).

만일 기존의 법선 벡터 압축 알고리듬을 이용하여 다양한 압축률을 가진 데이터 포맷을 생성할 경우에는 Fig. 4의 위에서와 같이 원본 데이터를 이용하여 각각의 압축률에 맞는 데이터 생성을 위해 추가적인 인코딩 작업을 해야 한다. 또한, 인코딩 후 파일을 저장할 데이터베이스에 각각의 압축된 데이터를 모두 저장하고 있어야 하므로 데이터베이스에 저장되는 데이터의 크기도 그만큼 늘어나게 된다. 이렇게 압축된 데이터를 각 어플리케이션으로 전송할 경우, 압축률에 따라 크게는 6배의 시간 차이가 나게 된다. 따라서, 전송 시간과 데이터 저장의 효율성을 고려한다면 하나의 압축 데이터로 다양한 압축률의 법선 벡터를 얻을 수 있는 포맷이 요구된다.

따라서, 본 연구에서는 멀티 레벨을 지원하는 법선 벡터 인코딩 알고리듬을 개발하였다. 본 논문에서는 다양한 압축률을 추가적인 인코딩 없이 추출하기 위해 수정된 옥드리 인코딩(Modified Octree Encoding) 방법을 제안하였고, 제안된 방법으로 법선 벡터의 멀티 레벨 압축 알고리듬을 개발하였다. 개발된 알고리듬은 시각적 왜곡을 구분할 수 없는 수준까지 법선 벡터의 다양한 압축률을 지원한다.

본 논문의 구성은 다음과 같다. 먼저 2장에서 법선 벡터에 대해 소개하고 이제까지의 관련 연구를 살펴

본 후, 3장에서는 수정된 옥트리 인코딩 압축(MOEC, Modified Octree Encoding Compression) 프로세스에 대해 설명한다. 4장에서는 기존의 옥트리 표현법 및 인코딩 방법과 본 논문에서 제안한 수정된 옥트리 인코딩 방법을 소개하고 수정된 옥트리 인코딩 압축 알고리듬을 이용한 법선 벡터의 압축과 시각적 왜곡을 구별할 수 없는 압축 레벨을 결정한다. 5장에서는 기존의 법선 벡터 압축 알고리듬을 이용하여 압축률을 변환할 때와 제안된 알고리듬을 이용하여 압축률을 변환할 때의 차이를 비교하였고 6장과 7장에서는 각각 적용 사례와 그에 따른 결론을 서술하였다.

2. 관련 연구

2.1 법선 벡터(Normal Vector)

법선 벡터(Normal Vector)는 IEEE 표준에 따라 float형(32 bits)으로 표현된 3개의 실수 값(x, y, z)으로 구성되어 있다. Fig. 5는 VRML 파일 내에서 법선 벡터를 표현하는 방법이며 Fig. 6은 파일 내에 표현된 법선 벡터 값들을 단위 구에 사상(Mapping)한 예이다.

```
Normal Normal {
    vector [
        0.47142   -0.81648   -0.33333 ---①
        0.47142    0.81648   -0.33333 ---②
       -0.94281    0.0       -0.33332 ---③
        0.0        0.0        1.0     ---④
        ]
}
```

Fig. 5. Normal vector expressions in VRML file.

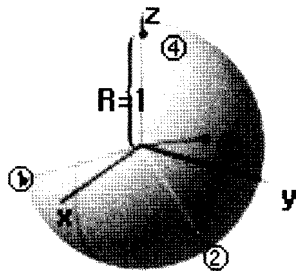
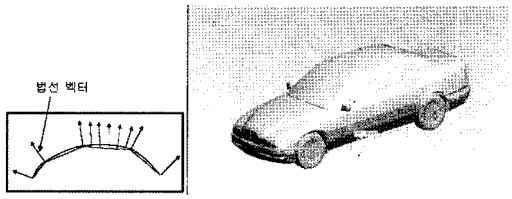


Fig. 6. Normal vectors on unit sphere.

이러한 법선 벡터는 주로 3D 형상 모델을 가시화할 때 보다 현실감 있게 표현하기 위한 방법인 음영처리(Shading)에 활용된다. Fig. 7의 (a)는 메쉬 모델의 점 데이터 위에 위치한 2D 단면에서의 법선 벡터를 나타낸 그림이며, (b)는 법선 벡터를 이용하여 자동차 메쉬 모델에 음영처리를 적용한 그림이다.



(a) (b)

Fig. 7. Shading model using normal vectors.

2.2 법선 벡터 압축에 관한 기존 연구

법선 벡터 압축에 관한 연구는 Deering[7]에 의해 처음 연구가 진행되었다. Deering은 Fig. 8에서와 같이 법선 벡터가 위치하는 단위 구를 8등분하고, 다시 8분구를 6개의 영역으로 분할하였다. 이렇게 분할된 각 영역에 대해 인덱스 값을 부여하여 그 영역에 속하는 법선 벡터를 부여된 인덱스 값으로 인코딩하는 방법을 제안하였다.

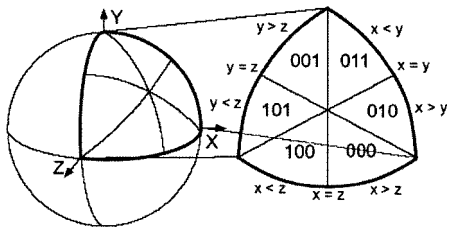


Fig. 8. Encoding of the six sextants of each octant of a sphere[7].

Cho[9]는 Fig. 9의 볼트 모델처럼 법선 벡터가 균일하게 분포되어 있지 않은 형상 모델 데이터의 경우에 K-means 알고리듬을 이용하여 법선 벡터를 클러스터링하고 클러스터링된 법선 벡터를 대표 법선 벡터(RNV: Representative Normal Vector)와 그 상대 값으로 인코딩하여 압축하는 방법을 제안하였다.

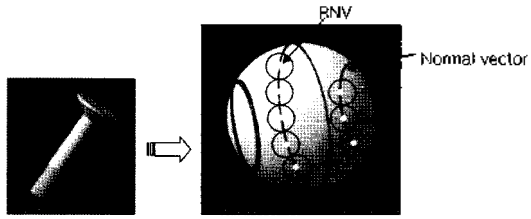


Fig. 9. Bolt model and its distribution of normal vectors.

Moon[10]은 법선 벡터를 압축할 때 시각적으로 왜곡을 구분할 수 없는 클러스터링 레벨을 결정하고 법선 벡터를 대표 법선 벡터와 그 차이 값으로 압축하

는 VULC(Visually Undistinguishable Lossy Compression) 알고리듬을 제안하였다.

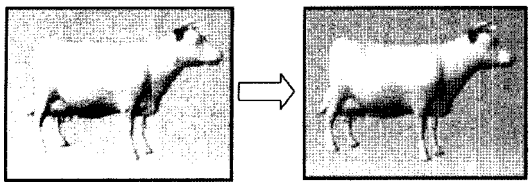


Fig. 10. Compare original cow model (left) and compression model using VULC (right).

3. 수정된 옥트리 인코딩 압축 프로세스

본 논문에서 제안하는 수정된 옥트리 인코딩 압축(MOEC: Modified Octree Encoding Compression) 프로세스는 Fig. 11과 같다. 먼저 x, y, z의 직교좌표로 저장된 법선 벡터 값을 수정된 옥트리 인코딩 방법으로 변환한다. 여기에서 시각적인 왜곡을 구분할 수 없는 압축 레벨을 만족하면 변환된 값으로 법선 벡터를 압축하고 그렇지 않은 경우에는 위의 과정을 반복적으로 수행한다.

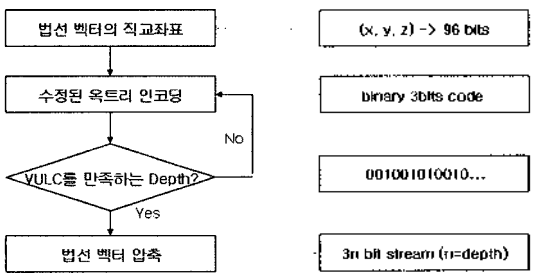


Fig. 11. MOEC process.

4. 수정된 옥트리 인코딩 방법

4.1 기존의 옥트리 표현법 및 인코딩 방법

옥트리 표현법(Octree Representation)은 Fig. 12의 (a)와 같이 주어진 형상에 대한 바운딩 박스(Bounding Box)를 생성하고 3D 공간 상에서 x, y, z축 방향으로 주어진 공간을 각각 이등분하여 분할된 육면체 내에 형상이 포함되면 가시화 대상으로 저장하는 데이터 표현방법이다[6].

옥트리는 x, y, z축의 각 방향에 대해 이등분한 8개의 서브 공간으로 분할되며 보다 세밀한 영역의 구분을 위해 재귀적(recursive)으로 공간을 분할하는 성질을 가지고 있다. 나뉘어진 영역은 8개의 서브 노드(node)를 가지는 트리 형태로 저장이 가능하다. 옥트리 표현법은 Fig. 12의 (b)처럼 주어진 바운딩 박스를 분할하는 수에 따라 depth를 나누게 된다.

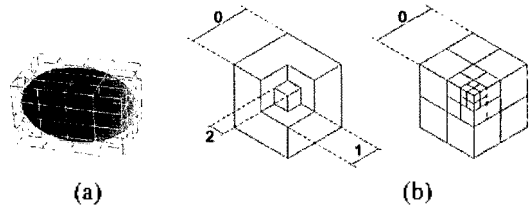
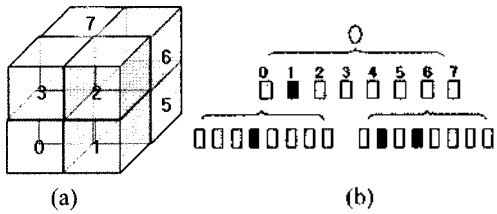


Fig. 12. Bounding Box[6] of octree and its depth.

표현된 옥트리를 인코딩하는 방법으로는 Tamminen[3]이 제안한 DF-representation이 가장 많이 쓰인다. 이 방법은 옥트리를 "깊이우선(depth-first)"으로 탐색하여 만나는 노드의 정보들을 순차적으로 저장하는 방법으로 각 노드는 "B", "W", 또는 "("로 표현된다. 여기에서 "B"는 Black을 "W"는 White를 "("는 Gray를 의미하며 Black 노드의 경우에는 그 영역이 꽉 차있음을 나타내고 White의 경우는 완전히 비어있음을, Gray 노드의 경우에는 부분적으로 차있음을 의미한다. 분할된 각 노드는 저장되며 Gray노드의 경우에는 추가적인 분할을 계속해야 함을 나타낸다.



DF-representation:
((WWWBWWWWBWWWW(WBWBWWWWW
(c)

Fig. 13. Octree and its DF-representation.

무한히 분할되는 것을 막기 위해 대개는 옥트리의 레벨을 정해준다. Fig. 13의 (a)와 (b)는 분할된 8개의 공간을 순차적으로 인덱스 값을 주는 방법과 그에 따른 옥트리의 구성을 보여준다. 위와 같은 옥트리를 DF-표현법을 이용하면 Fig. 13의 (c)와 같이 노드 전체를 표현할 수 있다.

4.2 수정된 옥트리 표현법 및 인코딩 방법

기존의 옥트리 인코딩 방법은 Fig. 13의 육면체에서 보듯이 8개의 노드를 반시계 나선 방향으로 순차적으

로 인덱싱을 한 후 그 노드 값에 공간을 어느 정도 차지하는 지에 대한 정보를 저장하게 되어 있다. 이 방법은 분할이 필요한 노드를 만날 때까지는 그 이전 노드의 정보도 가지고 있어야 한다. 따라서, 본 논문에서는 옥트리의 노드 값 자체에 위치 정보를 저장하도록 인코딩 방법을 수정하였다.

Fig. 14는 3D 공간 상의 옥트리를 한 차원 낮춰서 Quadtree로 표현한 그림이다. Quadtree의 4개의 노드 중 상위 노드의 자리를 바꾸어 이진수로 표현하면 Fig. 14의 오른쪽 그림과 같이 표현할 수 있다. 이렇게 위치를 바꾸어 옥트리를 이진코드(binary code)로 표현할 경우, 첫 번째 이진 코드에서 0은 왼쪽, 1은 오른쪽을, 두 번째 이진 코드에서 0은 아래쪽, 1은 위쪽으로 표현이 가능하다. 이렇게 표현할 경우 Quadtree에서는 노드 값에 2 bits로 위치정보를 줄 수 있으며 옥트리의 경우에는 깊이 방향을 더해 3 bits로 공간을 표현할 수 있다.

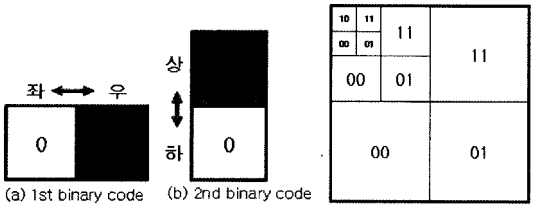


Fig. 14. Modified quadtree indexing.

Fig. 15의 (a)는 DF-representation을 이용한 옥트리 인코딩 방법이며, (b)는 수정된 옥트리 인코딩 방법을 이용하여 생성된 이진 코드들을 나타낸 그림이다. 기존의 방법은 주어진 공간에 점유하는 노드들 전체를

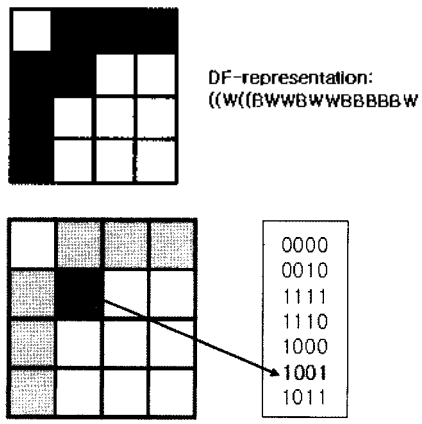


Fig. 15. (a) Conservative encoding (up) and (b)modified encoding in quadtree (down).

나타내기에는 좋으나 (b)처럼 공간 상의 임의의 노드 값 각각만을 지칭하는 데에는 수정된 옥트리 표현법이 더 효과적임을 알 수 있다.

4.3 MOEC 알고리듬을 이용한 법선 벡터의 압축

MOEC 알고리듬은 수정된 옥트리 인코딩 방법에 의해서 단위 구를 둘러싸는 바운딩 박스로부터 각 법선 벡터 값을 이진 인덱스 값으로 인코딩한다. 좀더 구체적으로 알고리듬을 설명하면, 먼저 주어진 법선 벡터의 x, y, z 값을 읽어들인 후, 각 축에 의해 이등분되는 영역 중 어느 곳에 속하는지를 비교하여 각각의 이진 코드를 생성한다. 한 번 분할될 때마다 depth가 증가하며 3 bits의 코드가 따라 붙게 되어, 사용자가 정한 한계 값을 만족하는 depth까지 3 bits 단위의 이진코드가 비트 스트림(bit stream)으로 생성된다. Fig. 16은 MOEC 알고리듬으로 법선 벡터를 압축하는 개략적인 코드이다.

```
struct octreeroot {
   double xmin, ymin, zmin;
   double xmax, ymax, zmax;
};

octreeroot r;
...

int depth;
for (depth=0; depth<LIMIT; depth++) {

//generate x-binary code
if (r.xmin<=vertex_x && vertex_x<=(r.xmax+r.xmin)/2) {
   bin_x=0; r.xmax=(r.xmax+r.xmin)/2;}
   else if (vertex_x<=r.xmax &&
            vertex_x>=(r.xmax+r.xmin)/2) {
   bin_x=1; r.xmin=(r.xmax+r.xmin)/2;}
else {cout<<"error! x = "<<vertex; break;}

//generate y-binary code
...
//generate z-binary code
...
}
```

Fig. 16. Pseudo code of MOEC algorithm.

이렇게 압축된 법선 벡터의 비트 스트림 값은 디코딩(decoding) 시 주어진 육면체의 중심에 사상하게 된다. Fig. 17의 (a)는 depth가 1인 경우의 상위 4개의 육면체의 중심에 사상된 법선 벡터들을 나타낸 그림이며, (b)는 depth가 2일 때 상위 4개의 육면체 중 좌측 전면의 육면체에 사상된 법선 벡터를 나타낸 그림이다.

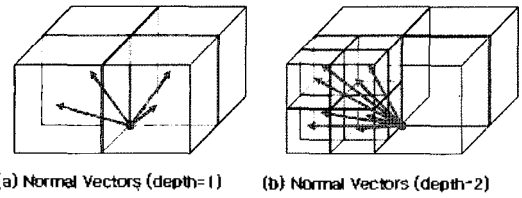


Fig. 17. Normal vectors at depth=1 and 2.

각 depth에 따라 여러 방향의 법선 벡터 압축이 가능한데 원래의 법선 벡터와 가장 각도편차가 작은 법선 벡터의 값을 추출하는 방법은 아래와 같다.

(1) 법선 벡터의 크기를 0부터 바운딩 박스에 접하는 곳까지 연장

(2) 법선 벡터와 만나는 육면체를 찾아서 그 중심점과의 각도 편차를 계산

(3) 해당하는 육면체 중 각도 편차가 가장 작은 값으로 인코딩

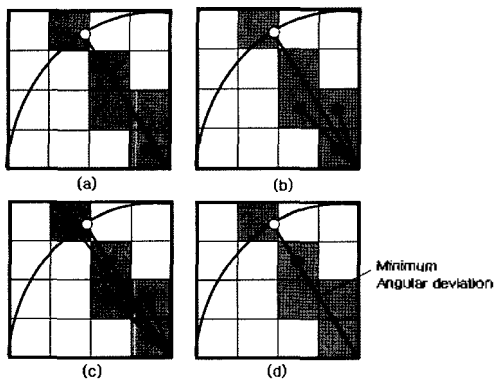


Fig. 18. Original normal vector and compressed normal vector.

Fig. 18에서 보듯이 원점에서 법선 벡터가 지나는 값 중, 법선 벡터의 스칼라(scalar) 값이 가장 작은 지점과 만나는 육면체부터 차례로 검색을 하여 그 중 각도편차가 가장 작은 값을 압축된 법선 벡터로 인코딩하게 된다. 각도편차를 구하는 식은 (1) 식과 같으며 이 때 벡터 $\vec{a}$ 는 원래의 법선 벡터를 $\vec{b}$ 는 압축된 법선 벡터를 나타내며 이를 통해 각도편차(ε)를 구할 수 있다.

$$\varepsilon = \cos^{-1}\left(\frac{\vec{a}\cdot\vec{b}}{|\vec{a}|\cdot|\vec{b}|}\right) \tag{1}$$

4.4 VULC를 만족하는 MOEC의 인코딩 depth 결정

법선 벡터와 압축된 법선 벡터 사이의 각도 편차(ε)가 0.01 radian(0.56 deg) 이하이면 원래 모델에 대한 왜곡을 시각적으로 구분할 수 없음이 입증되어 있다[7]. 따라서, 법선 벡터를 압축하는 경우에는 시각적인 왜곡을 구분할 수 없는 수준까지만 압축하는 것이 효율적이다. 이 절에서는 본 논문에서 제안한 MOEC 알고리듬을 이용하여 각도편차 값이 시각적 왜곡을 구분할 수 없는 수준의 옥트리 depth를 결정하는 방법에 대해 서술한다.

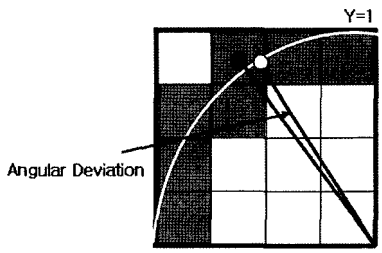


Fig. 19. Angular deviation.

MOEC로 압축된 법선 벡터는 옥트리의 각 depth에 해당하는 육면체의 중심점에 사영한 값을 가지게 되고 그 영역 안에서 다양한 각도 편차가 존재한다. 각 depth에서 생성된 법선 벡터 값들은 단위 구를 각 depth의 노드 수만큼 분할하게 된다. 이 때 각도편차는 2차원 상의 단위 원을 각 노드로 분할되는 법선 벡터들의 수로 나누어서 구하였다. 시각적인 왜곡을 구분할 수 없는 각도편차는 ε이고 그 때의 depth는 n이며 식 (2)를 통해서 그 값을 구할 수 있다.

$$\varepsilon = 2\pi \div \left(\frac{1}{2}\cdot 4^{n+1}\right) \tag{2}$$

Fig. 20의 (a)는 depth가 5일 때 옥트리 모든 노드의 중점들을 표현한 그림이며 (b)는 옥트리의 중심점에서 각 노드에 이르는 방향을 법선 벡터의 방향으로 한 단위 벡터 값들을 3D 공간 상에 가시화한 그림이다.

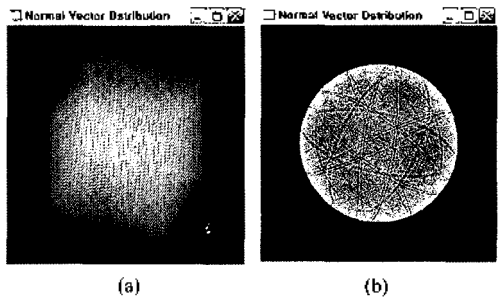


Fig. 20. Normal vector distribution.

Table 1은 각각의 depth에서 MOEC 알고리듬을 적용한 결과 얻어진 각도편차 값을 나타내고 있으며, depth가 5일 때 시각적 왜곡을 구별할 수 없는 수준을 만족하는 각도편차 값(ε = 0.003068)을 얻어냈다.

Table 1. Angular deviations of each depth

Depth	Angular dev.
1	0.785398
2	0.196349
3	0.049087
4	0.012271
5	0.003068
6	0.000767
7	0.000191

Depth가 5일 때 실제 옥트리의 분할된 육면체 수는 85개이므로 육면체 수에 해당하는 32768개로 분할된 법선 벡터 값의 추출이 가능하다. 원래의 법선 벡터가 x, y, z 좌표 값으로 표현된 96 bits의 데이터 값이므로 depth가 5일 때의 압축률은 식 (3)과 같이 나타낼 수 있으며 그때의 압축률은 84.4%이다. Fig. 20은 각 depth에 따른 각도편차의 변화를 보여주는 그래프이다.

$$\text{Compression Ratio}(\%) = \frac{3\text{bit} \times 5\text{depth}}{96\text{bits}} \times 100 = 84.4\% \quad (3)$$

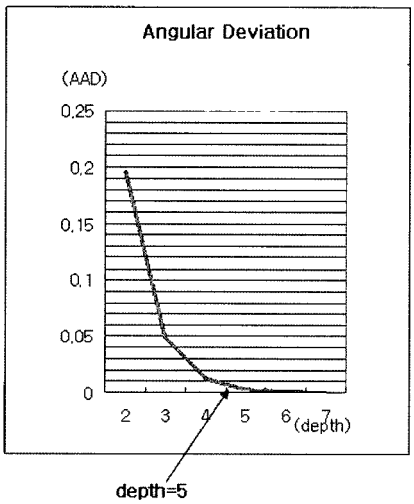


Fig. 21. Angular deviation graph.

Table 2는 각 depth에서의 압축률을 나타내고 있다. Depth가 증가할수록 각도편차는 줄어들게 되어 모델의 왜곡현상도 함께 줄어들게 되지만 대신 그만큼 비트수가 증가하기 때문에 데이터 사이즈가 증가하게 된다. Table 3은 법선 벡터 각각의 x, y, z 값과, MOEC를 이용하여 depth가 5가 될 때까지 이진 코드로 압축한 인코딩 값이다.

Table 2. Compression rates of each depth

depth	1	2	3	4	5	6
압축률	97	94	91	88	84.4	81.3

Table 3. x, y, z values and MOEC index values

X	Y	Z	Index Code
0.0830	0.554103	0.545485	000110110010011
0.108304	0.580984	0.510265	000110110010011
0.158741	0.594835	0.58393	000110110010011

5. 기존 압축방법과 압축률 변환 비교

본 논문에서 제안된 MOEC 알고리듬은 압축률의 변환 시에 기존의 압축 알고리듬에 비해 좋은 성능을 가지고 있다. Fig. 22에서 (a)는 압축 전의 법선 벡터라고 하면, (b)와 (c)는 각각 기존의 법선 벡터의 압축형과 MOEC 알고리듬을 이용한 법선 벡터의 압축형으로 표현된다. (b)의 경우는 주어진 영역을 분할하여 그에 해당하는 대표 법선 벡터(RNV)의 인덱스 값과 대표 법선 벡터와 압축된 법선 벡터의 차이만큼을 상대 값으로 저장한 두 영역으로 나눌 수 있다. (c)의 경우는 이진 코드가 비트 스트림으로 저장되어 있으며 3 bits씩 끊어서 읽으면 각 depth에서의 압축된 법선 벡터 값에 해당하게 된다.

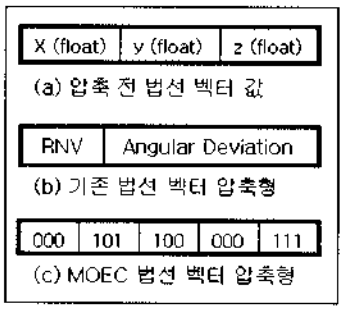


Fig. 22. Expressions of compressed normal vector.

Fig. 22에서 보듯이 기존의 법선 벡터 압축 알고리듬을 이용하여 압축률을 높이고자 할 경우에는 대표 법선 벡터와 그에 대한 상대 값을 새로 인코딩해야 하며 새롭게 인코딩된 데이터를 저장해야 하는 반면, 제

안된 MOEC 알고리듬을 이용한 경우에는 3 bits씩 나누어 저장할 경우에 압축률이 높은 형태의 데이터를 추가적인 인코딩 작업 없이 추출해 낼 수 있다.

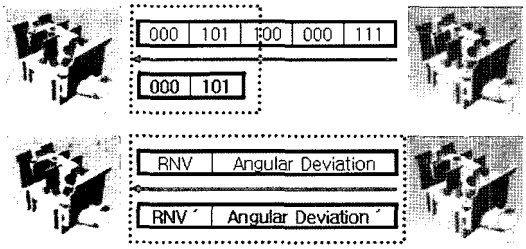


Fig. 23. Compression rate variation process using conservative compression algorithm (up) and MOEC algorithm (down).

각각의 압축률을 가진 데이터를 모두 저장하고 있지 않아도 되는 장점이 있으므로 실제 데이터가 차지하는 용량은 그만큼 줄어들게 된다. Fig. 23은 기존의 알고리듬(아래)과 제안된 알고리듬(위)을 이용하여 압축률이 높은 법선 벡터 압축 포맷으로 기존 포맷을 변환하고자 할 때의 변환 과정을 나타낸 그림이다.

Fig. 24는 각각 제안된 MOEC 알고리듬을 이용하여 다양한 레벨의 압축률을 지원하는 법선 벡터 값들을 추출해 내는 그림(위)과 기존의 압축 알고리듬을 이용할 시에 추가적인 인코딩 작업을 하게 되는 과정을 나타낸 그림(아래)이다.

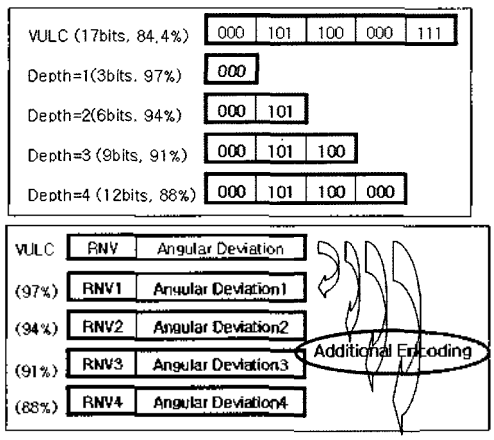


Fig. 24. Encoding process of MOEC data (up) and conservative compression data (down).

6. 적용 사례

본 논문에서는 3D 메쉬 모델 중에서 VRML 포맷을 사용하였고, 각각의 적용 사례는 MOEC 알고리듬으로 각 depth마다 가시화를 수행하였다. Fig. 25는 한 단면으로만 이루어진 얼굴 모델에 대한 적용 사례를, Fig. 26는 3D의 모든 면에서 법선 벡터가 고르게 분포하고 있는 소 모델에 대한 적용사례를 보여주고 있다. 이 두 모델에서는 각 레벨마다 왜곡의 정도가

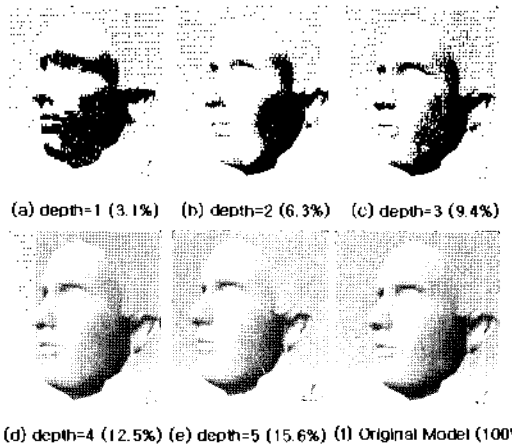


Fig. 25. Face model (10,920 bytes).

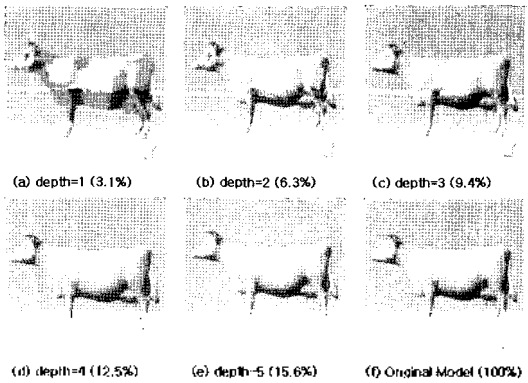


Fig. 26. Cow model (278,784 bytes).

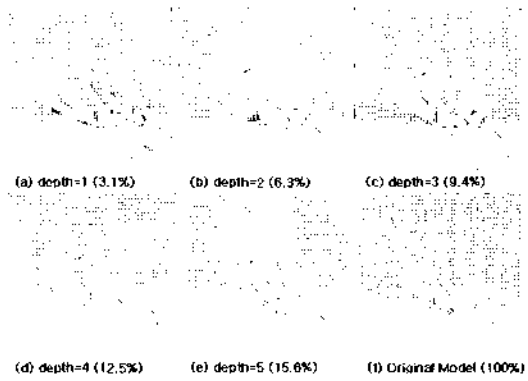


Fig. 27. Hand model (81,000 bytes).

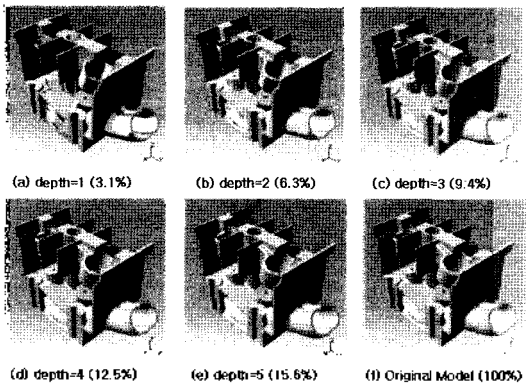


Fig. 28. Cylinder Head (1,243,680 bytes).

눈에 띄게 달라지는 반면 모델의 곡률 변화가 큰 Fig. 27의 손뼈 모델이나 Fig. 28과 같이 매쉬 수가 많은 CAD모델에서는 왜곡의 정도 차가 그리 크지 않았다.

7. 결 론

본 논문에서는 수정 옥트리 인코딩 방법을 이용하여 추가적인 인코딩없이 다양한 압축률을 가진 법선 벡터를 생성하는 알고리듬을 개발하였다. 시각적인 왜곡을 구별할 수 없는 수준까지 법선 벡터를 압축하기 위해 MOEC 인코딩 시의 옥트리 depth를 정하였고 그 때의 압축률이 84.4%(15 bits)로 이는 Deering의 방법(17 bits) 대비 11.8% 개선된 값이다. 법선 벡터의 압축률을 변환하는 경우 기존의 압축 알고리듬을 이용하면 클러스터링 레벨의 조정을 거쳐 추가적인 포맷 변환이 필요한 반면, MOEC 방법은 추가적인 인코딩없이 법선 벡터의 압축률 변환이 가능하다. 따라서, 특정한 응용 분야에 국한하여 사용될 수 있는 기존의 법선 벡터 압축 알고리듬에 비해 MOEC 알고리듬은 옥트리의 depth 값에 따라 디테일 정도가 다양한 법선 벡터 값을 추출할 수 있으므로 추가적인 인코딩 없이 압축률의 단계를 조정할 수 있는 유연성이 보장되어 효율적인 법선 벡터의 데이터 저장과 전송이 가능하다.

참고문헌

1. Meagher, D., "Geometric Modeling Using Octree Encoding", *Computer Graphics and Image Processiong*, Vol. 19, pp. 129-147, 1982.
2. Gargantini, I., "Linear Octrees for Fast Processing of Three-dimensional Objects", *Comput. Graph. Image Process*, Vol. 20, No. 4, pp. 365-374, Dec. 1982.
3. Tamminen, M., "Efficient Octree Conversion by Connectivity Labeling", Proceedings of the 11th Annual Conference on Computer Graphics and Interactive Techniques, 1984.
4. Brunet, P. and Navazo, I., "Solid Representation and Operation using Extended Octrees", *ACM Transactions on Graphics (TOG)*, Vol. 9, Issue 2, 1990.
5. Schmalstieg, D., An Octree-based Level of Detail Generator for VRML, Proceedings of the Second Symposium on Virtual Reality Modeling Language (1997).
6. Ryu, J. H., "Efficient Octree Encoding for Real Time Transmission of 3D Geometry Data Through Internet", *Society of CAD/CAM Engineers*, Vol. 7, No. 4, pp. 262-268, 2002.
7. Deering, M., Geometry Compression. In Proc. SIGGRAPH'95, pp. 13-15, 1995.
8. Taubin, G. and Rossign, J., "Geometry Compression through Topological Surgery", *ACM Transactions on GRAPHICS*, Vol. 17, No. 2, pp. 84-115, 1998.
9. Kim, D.-S., Cho, Y. S. and Kim, D. U., "The Compression of the Normal Vectors of 3D Mesh Models using Clustering", *Computational Science*, ICCS 2002, pp. 275-284, 2002.
10. Moon, H, S., The Compression of Normal Vectors for 3D Mesh Models (2003).

김 용 주

2002년 한양대학교 기계공학부 학사
2004년 한양대학교 기계설계학과 석사
2004년~현재 현대기아자동차 남양연구소 연구원
관심분야: CAD/CAE, 3D modeling, simplification of 3D mesh model, Multi-resolution modeling

김 재 정

1981년 한양대학교 정밀기계학과 학사
1983년 George Washington 공학석사
1989년 미국 MIT 공학박사
1989년~1991년 미국 IBM T.J. Watson 연구소 연구원
1991년~1993년 한국 IBM 소프트웨어 연구소 연구원
2002년~2004년 미국 NIST 객원 연구원
2003년 블란시 다쏘시스템 객원 연구원
1993년~현재 한양대학교 기계공학부 교수
관심분야: Geometric Modeling, CAD/CAM 응용, PLM 응용