파동분할 기반의 꼭지점 계보를 이용한 순차적 삼차원 메쉬 부호화

김태완, 안정환, 양창모, 호요성
광주과학기술원 정보통신공학과
광주광역시 북구 오룡동 1번지

Sequential 3D Mesh Coding using Vertex Pedigree based on Wave Partitioning

Tae-Wan Kim, Jeong-Hwan Ahn, Chang-Mo Yang and Yo-Sung Ho
Kwangju Institute of Science and Technology (K-JIST)
1 Oryong-dong, Puk-gu, Gwang-ju, Korea
E-mail: twkim@kjist.ac.kr

요약

본 논문에서는 파동분할(Wave Partitioning) 방식을 기반으로 꼭지점들간의 특징적인 관계(Vertex Pedigree)를 이용한 순차적(Sequential) 메쉬 부호화 방식을 제안한다. 파동분할 방식은 호수에 물방울이 퍼져 나가는 자연 원리를 이용하여 초기 삼각형의 주위에 삼각형을 덧붙여 가면서 하나의 SPB(Small Processing Block)을 만들어내는 방식이다. 이 방식을 이용하여 하나의 모델을 서로 독립적인 SPB로 분할하고, 각각의 SPB내에서 초기 삼각형을 중심으로 그것에 덧붙여진 삼각형에 의해 만들어진 원 또는 반원을 찾는다. 또한, 그 원주를 따라 순차적으로 꼭지점을 구하면 각각의 꼭지점들은 서로의 관계에 따라 일정한 패턴으로 늘어서게 되고, 이것을 이용하여 연결성 정보 없이 부가 정보만으로 모델을 순차적으로 무손실 부호화한다.

1. 서론

최근 VRML 기반의 삼차원 모델은 컴퓨터 애니메이션이나 스튜디오 그래픽 디자인 등 다양한 분야에서 사용되고 있다. 하지만 삼차원 모델은 저장과 전송에 있어서 많은 용량을 차지하기 때문에 인터넷과 같은 제한된 대역에서의 다양한 조작을 위해 효율적인 부호화 방식이 요구된다. 초기에는 삼차원 메쉬 모델의 단순화에 대한 연구가 활발히 이루어졌다. Hoppe[1]는 상당히 적은 양의 꼭지점과 삼각형으로 모델을 표현하기 위해 메쉬 최적화(Mesh Optimization)를 이용한 점진적 메쉬(Progressive Mesh) 부호화 방식을 제안 하였다. 최근에는 단순화보다 메쉬 압축에 대한 연구가 많이 진행되고 있다. Taubin은 메쉬의 꼭지점 트리(Vertex Tree)와 삼각형 트리(Triangle Tree)를 이용한 Topological Surgery[7] 방식을 제안하였다.

이러한 기존의 부호화 방식은 모델에 대한 정보를 단순화 또는 압축 부호화하는 데에만 초점을 두고 있다. 하지만 전송 또는 저장 과정에서 비트 오류가 발생하면 재생되는 모델에 심각한 영향을 미칠 수 있다. 제안한 메쉬 부호화 방식은 삼차원 모델이 분할된 후 부호화가 수행되므로, 국부적인 오류는 전체적인 모델에 영향을 미치지 않는다. 본 논문에서는 삼차원 모델의 저장과 전송에 있어서 부호화 효율과 오류 내성의 성질을 높이기 위해 파동 분할 방식(Wave Partitioning)을 기반으로 한 향상된 순차적 메쉬 부호화 방식을 제안한다[8].

2. 메쉬 부호화

점진적 메쉬(Progressive Mesh) 부호화는, 그림 1과 같이, 낮은 해상도를 가진 모델에 부가적인 정보를 추가하여 원래 모델로 복원되는 과정이다. 반면, 순차적 메쉬(Sequential Mesh) 부호화는, 그림 2와 같이, 완전한 해상도를 가진 부분들이 연속적으로 덧붙여져 원래 모델로 복원되는 과정이다.

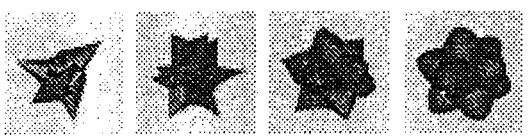


그림 1. 점진적 메쉬

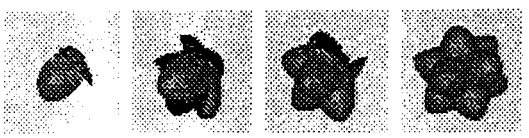


그림 2. 순차적 메쉬

3. 점진적 메쉬 부호화

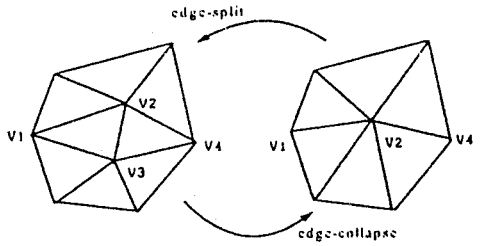


그림 3. Edge-Collapse와 Edge-Split

그림 3에 보인 것처럼, 인접한 두 개의 꼭지점 v2와 v3를 Edge-Collapse[1] 방법을 이용하여 하나의 꼭지점으로 통합한다. 이 과정에서 인접한 두 면인 (v1, v2, v3)와 (v2, v3, v4)는 없어진다. 따라서 초기 메쉬를 n번의 연속적인 Edge-Collapse 방법으로 단순화한다.

$$(\hat{M} = M^n) \xrightarrow{ecol_{n-1}} \cdots\cdots \xrightarrow{ecol_1} M^1 \xrightarrow{ecol_0} M^0$$

Edge-Split[1] 과정은 Edge-Collapse 과정의 역 과정으로 하나의 꼭지점 v2에 또 다른 꼭지점 v3가 추가되어 새로운 삼각형 (v1, v2, v3)와 (v2, v3, v4)가 만들어지고, 모델은 좀 더 상세히 표현된다.

그림 3의 Edge-Split 과정에서 Vertex-Split도 동시에 일어나므로, Vertex-Split은 Edge-Split과 동일한 의미로 사용된다. 즉, 임의의 모델은 단순화된 기본 모델과 n개의 Vertex-Split 변수인 $vsplit_i(v1_i, v2_i, v3_i, v4_i, A_i)$로 표현될 수 있다. 또한, n번의 Vertex-Split 과정을 통해 단순화된 모델은 완전히 복원된다. 이러한 표현 방식이 점진적인 부호화 방식이다. 여기서 A는 속성 정보를 의미하며, 색상 정보, 법선 벡터, 질감 정보 등을 포함한다.

$$M \xrightarrow{vsplit_0} M \xrightarrow{vsplit_1} \cdots\cdots \xrightarrow{vsplit_{n-1}} (M^n = \hat{M})$$

4. 순차적 메쉬 부호화

4.1 삼차원 모델의 파동분할 방법

본문에서 제안한 방법은 물방울이 물 위에 떨어져 원 또는 반원을 그리며 퍼져 나가는 파동 현상을 이용한다.

그림 4에 보인 것처럼, 임의의 초기 삼각형을 정하고, 이 삼각형의 세 모서리를 공유하는 삼각형 A를 화살표 방향으로 덧붙인다. 덧붙여진 삼각형 A의 모서리를 공유하는 삼각형 B를 화살표 방향으로 덧붙인다. 다시 덧붙여진 삼각형 B의 모서리를 공유하는 삼각형 C를 화살표 방향으로 덧붙인다.

덧붙여지는 삼각형의 순서에 관계없이 이러한 과정을 원하는 수의 삼각형을 가지는 SPB(Small Processing Block)을 얻을 때까지 반복적으로 수행한다. 분할이 삼각형의 개수를 기준으로 이루어지기 때문에 꼭지점의 개수를 이용한 분할에서 발생할 수 있는 독립된 SPB 사이의 삼각형 복원은 고려하지 않아도 된다.

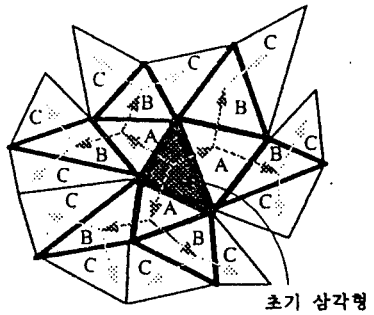


그림 4. 파동분할을 이용한 SPB 형성

그림 5는 모델을 1/2의 두 부분으로 나누고, 다시 각각을 1/2로 분할한 것이다. 어느 특정 부분만을 분할하는 경우, 초기 삼각형은 그 부분의 임의의 자리에 위치할 수 있다. 반면, 전체 모델을 부호화할 경우에는 초기 삼각형은 각 SPB의 최정점 또는 최저점에 위치하게 된다.

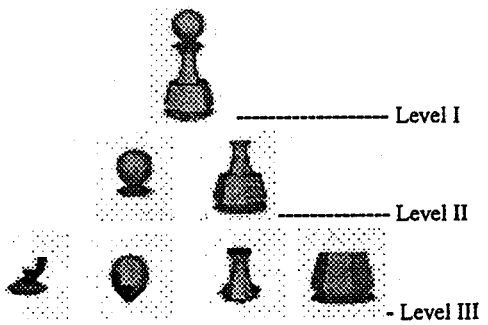


그림 5. 분할된 PAWN.

모델을 분할한 후에 부호화 효율을 높이기 위해 접선 정제 과정(Smoothing Scheme[5])을 적용한다. 본 논문에서는 기존의 접선 정제 과정을 제안한 알고리즘에 맞게 향상시켜 적용한다.

그림 6은 일반적인 접선 정제 과정을 나타낸 것이고, 그림 7은 향상된 접선 정제 과정을 적용한 결과 나타나는 SPB이다.

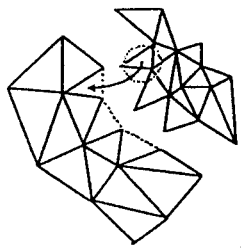
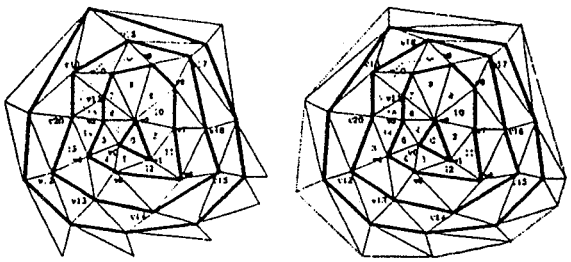


그림 6. 기존의 접선 정제 과정



(a) 적용 전 (b) 적용 후

그림 7. 향상된 접선 정제 과정

기존의 접선 정제 과정은 모델을 분할할 때 단순히 경계 부분을 매끄럽게 하는데 그 목적이 있다. 하지만 향상된 접선 정제 과정에서는 제안한 알고리즘의 성능 향상을 위해 분할된 SPB 내의 마지막 Rank가 완전한 원을 이루도록 삼각형을 추가 시켜 준다. 그림 7에 점선으로 이루어진 삼각형이 이에 해당한다. 따라서 떨어짐을 표현하는 추가적인 부가정보 없이 복부호가 가능해지고 부호화 효율이 높아진다.

4.2 과동분할을 이용한 순차적 부호화

과동분할 방법에 의해 모델이 분할되면, 분할된 각각의 SPB 내에서 초기 삼각형의 접선과 초기 삼각형에 덧붙여진 삼각형의 외부 접선으로 만들어지는 원 또는 반원의 원주를 따라 시계방향 또는 반시계 방향으로 꼭지점을 순차적으로 구한다.

하나의 원주를 따라 꼭지점이 구해지면, 그 원주의 바깥에 있는 다음 원주를 따라 꼭지점이 연속적으로 구해진다. 하나의 원주를 Rank로 정의하면, 각 꼭지점은 해당하는 Rank에 포함된다. 각 Rank에 포함된 꼭지점의 개수는 각 꼭지점의 모 꼭지점과 부호화 변수와 함께 부호화된 후 꼭지점 정보와 같이 전송된다. 복호화 부분에서는 전송된 꼭지점 스트림을 각 Rank 내에서의 개수로 분할하여 사다리 모양의 배열을 만들고, 이것을 기반으로 복호화를 수행한다. 일반적으로 분할하고자 하는 부분의 중심 또는 가장자리에 위치한 임의의 삼각형을 초기 삼각형으로 정한다. 분할된 메쉬는 초기 삼각형의 위치에 따라 원 SPB와 반원 SPB로 나누어진다.

원 SPB는 초기 삼각형이 SPB 중심에 위치하는 경우로, 초기 삼각형 주위의 삼각형이 초기 삼각형에 덧붙여져 원 형태를 이룬다. 반면, 반원 SPB는 초기 삼각형이 SPB 가장자리에 위치하는 경우로 초기 삼각형 주위의 삼각형이 초기 삼각형에 넷붙여져 반원 형태를 이룬다.

4.2.1. 원 SPB

그림 8에서 굵은 선은 하나의 Rank를 나타낸다. v0가 v1~v5에 의해 둘러싸여 있을 때, v0를 꼭지점 v3~v5의 어미 꼭지점(Mother Vertex)으로 정의하고, v3~v5를 v0의 아들 꼭지점(Son Vertex)으로 정의한다.

꼭지점 v1과 v2는 v0와 같은 Rank에 포함되기 때문에 v1~v2는 v0의 아들 꼭지점이 되지 않는다. 따라서, v1은 v6~v7의 어미 꼭지점이 되고, v2는 v8~v11의 어미 꼭지점이 된다. 초기 삼각형의 세 꼭지점 v0, v1, 그리고 v2로 이루어진 폐곡선을 Rank I이라 한다.

Rank I에 포함된 초기 삼각형의 꼭지점을 제외한 임의의 Rank내에서의 꼭지점 스트림은 이전 Rank인 (Rank-1)에 속하는 어미 꼭지점을 중심으로 반 시계방향으로 회전하면서 얻어진다. 그러면 Rank I에 놓여진 꼭지점을 어미 꼭지점으로 하여 얻어진 새로운 꼭지점 v3~v11의 집합이 Rank II가 된다.

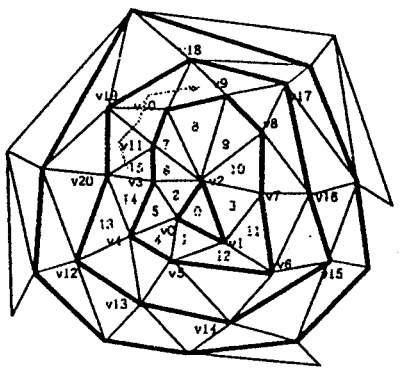


그림 8. 분할된 원 SPB

그림 9는 그림 8의 원 SPB에 대해 각각의 Rank에 포함되는 꼭지점 스트림을 일렬로 나열한 그림이다. Rank내에서의 꼭지점과 이전 Rank인 (Rank-1)에 포함되어 있는 어미 꼭지점과의 관계는 진한 직선을 이용하여 나타내었다. 하나의 Rank는 원 모양으로 폐곡선을 이루기 때문에 Rank 마지막에 초기 꼭지점을 한번 더 넣어 폐곡선임을 나타냈다. 삼각형 내의 번호는 그림 8에 나타난 삼각형과 대응되는 삼각형을 나타낸다.

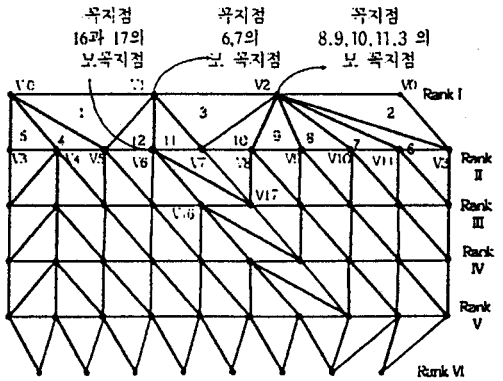


그림 9. 원 SPB의 순차적 꼭지점 배열

4.2.2 반원 SPB

그림 10에서 보는 바와 같이, v4와 v5, v11과 v12를 공유하는 삼각형이 없기 때문에 Rank I를 제외한 하나

의 Rank는 반원 형태를 이룬다. 원 SPB와 같은 원리를 이용하여 v0은 v3~v4의 어미 꼭지점이 되고, v1는 v5~v6의 어미 꼭지점이 된다. 그리고 v2는 v7~v8의 어미 꼭지점이 된다. 따라서 Rank I에는 v0~v2가 포함되고, Rank II에는 v3~v8이 포함된다. 연속해서 Rank III에는 v16~v17가 포함된다.

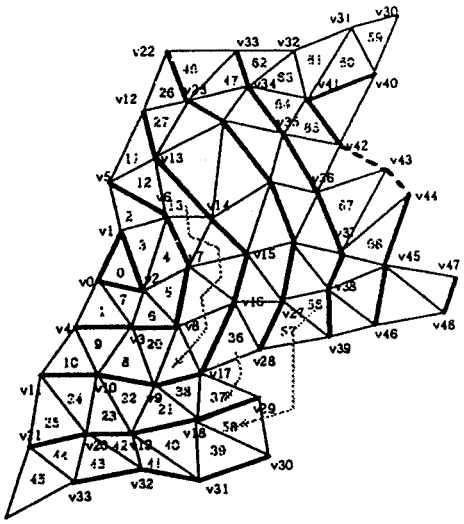


그림 10. 분할된 반원 SPB

그림 11은 그림 10의 반원 SPB에 있어서 각각의 Rank에 포함되는 꼭지점 스트림을 일렬로 나열한 그림이다. v4와 v5 사이와 v11와 v12 사이에 연결성 정보가 없는 것은, 그림 10에서 보는 바와 같이, v4와 v5 그리고 v11와 v12를 공유하는 삼각형이 없는 경우이다. 이처럼 메쉬가 두 부분으로 나누어지는 경우는 1비트의 변수로 구별한다. 분할된 반원 SPB 내에서의 첫 꼭지점은 가장 자리에 위치한 꼭지점 v5, v12 또는 v22가 될 필요는 없다. 왜냐하면, 위에서 설명한 두 부분으로 나누어지는 경우를 나타내는 1비트가 이러한 경우를 대비한 여유 정보이기 때문이다.

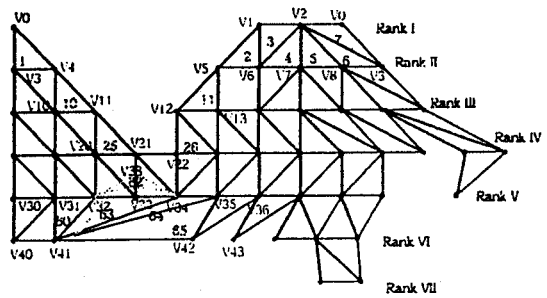


그림 11. 반원 SPB의 순차적 꼭지점 배열

4.3. 어미 꼭지점을 이용한 순차적 복호화

부호화 과정에서 어미 꼭지점은 어미 꼭지점이 포함되는 Rank내에서의 위치로 부호화된다. 즉, 표 2에서 보는 바와 같이, v42의 어미 꼭지점은 Rank V에 포함되어 있는 꼭지점 v35가 된다. v35는 Rank V내에서 0에서부터 6번째 해당하는 꼭지점이므로, v42의 어미 꼭지점은 5로 부호화된다. 따라서, 꼭지점 v43의 어미 꼭지점은 같은 방법으로 6으로 부호화 되고, 이들 꼭지점 v42와 v43의 어미 꼭지점의 차분은 1이 된다.

현재의 꼭지점을 Cv, 부호화 과정에서 Cv 다음에 구해진 꼭지점을 Nv, 그리고 Cv와 Nv의 어미 꼭지점을 각각 Mv_1과 Mv_2로 정의하면, 이들의 관계는 표 1과 표 2처럼 정리된다.

표 1. 그림 9에서 꼭지점 사이의 관계

Cv	Nv	Mv_1	Mv_2	$\lvert Mv_1-Mv_2\rvert$
v6	v7	v1	v1	0
v7	v8	v1	v2	1
v8	v9	v2	v2	0

표 2. 그림 11에서 꼭지점 사이의 관계

Cv	Nv	Mv_1	Mv_2	$\lvert Mv_1-Mv_2\rvert$
v40	v41	v30	v31	1
v41	v42	v31	v35	4
v42	v43	v35	v36	1

Mv_1과 Mv_2의 차분은 크게 세 가지 유형으로 분류된다.

<유형 1> : $|Mv_1 - Mv_2| = 0$

이 경우에는 Rank내에서의 Cv, Nv 그리고 (Rank-1)에서의 Mv_1으로 이루어진 삼각형을 구한다. 그림 11에서 꼭지점 v3와 꼭지점 v4의 어미 꼭지점 사이의 차분은 0이 된다. 이것은 유형 1에 해당되므로 Rank내에 있는 Cv인 v3와 Nv인 v4 그리고 (Rank-1)에서의 Mv_1인 v0를 이용하여 삼각형을 만든다.

<유형 2> : $|Mv_1 - Mv_2| = 1$

이 경우에도 유형 1과 마찬가지로 꼭지점 Cv와 꼭지점 Nv 그리고 Mv_1을 이용하여 삼각형을 만든다. 하지만, 이 경우에는 어미 꼭지점의 차이가 1이기 때문에, (Rank-1)상에 있는 두 꼭지점 Mv_1과 Mv_2 그리고 Cv로 이루어지는 삼각형을 추가로 구해야 한다.

예를 들어, 그림 9에서 v5와 v6의 Mv의 차분은 1이며 유형 2에 속하게 된다. 이 경우 유형 1의 경우처럼 삼각형 (v5, v6, v1)을 만든다. 그 다음 두 꼭지점의 어미 꼭지점들과 현재의 꼭지점으로 이루어지는 삼각형 (v0, v1, v5)를 추가로 만들어야 한다.

<유형 3> : $|Mv_1 - Mv_2| \geq 2$

이 경우는 반원 형태로 만들어진 SPB의 양쪽 두 부분이 서로 접하는 경우와 완전한 원이나 반원을 이루지 못하는 경우 발생한다. Cv, Nv, Mv_1, Mv_2 그리고 Mv_1과 Mv_2사이의 꼭지점들로 삼각형이 만들어진다. 만들어지는 삼각형의 유형은 각각 1비트의 변수로 정의되고, 각 꼭지점마다 적용된다.

그림 11에서 Cv인 v41과 Nv인 v42 그리고 이것들의 어미 꼭지점에 해당하는 두 꼭지점 v31과 v35, 그리고 어미 꼭지점 사이의 꼭지점인 v32~v34로 이루어지는 삼각형을 구한다. 즉 삼각형(v31, v32, v41), (v32, v34, v41), (v41, v34, v35), (v41, v35, v42), 그리고 (v32, v33, v34)가 만들어진다. 좀 더 구체적으로 설명하면, 차분이 2이상일 경우에는 일종의 Table에 의한 약속된 유형을 부호화하여 전송하게 된다. 본 논문에서는, 그림 12에서와 같이, 특정의 유형을 2개로 정의하였다.

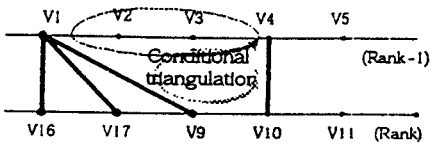


그림 12. 2이상인 Mv 차분

유형 I은 Cv가 v9, Nv가 v10인 경우 이들의 아들 꼭지점과 그 사이에 있는 꼭지점만을 이용하여 삼각형을 만드는 경우이다.

유형 I에 해당하는 꼭지점들로 만들어 질 수 있는 모든 삼각형에 대해 각각의 인덱스를 부여하고 그 인덱스를 부호화하여 전송한다. 같은 방법으로 유형 II는 Cv와 Nv, 이들의 자 꼭지점들 사이의 꼭지점만으로 삼각형이 만들어지는 경우이다.

그림 12에 해당하는 모든 경우의 삼각형과 그 인덱스는 표 3과 같다.

표 3. 유형과 인덱스

	삼각형(인덱스)
유형 I	(v9,v1,v2) : 0, (v9,v1,v3) : 1, (v9,v1,v4) : 2, (v9,v2,v3) : 3, (v9,v2,v4) : 4, (v9,v3,v4) : 5, (v9,v10,v4) : 6
유형 II	(v1,v2,v3) : 0, (v1,v2,v4) : 1, (v2, v3, v4) : 2

<유형 4>: 복합 구성 객체(Multiple Components)

만약 모델이 여러 개의 객체들로 이루어져 있는 경우는, 그림 13에서처럼, 분리된 각각을 하나의 독립된 객체로 간주하여 개별적으로 부호화한다.

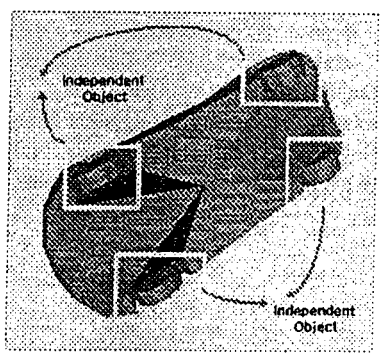


그림 13. 복합 구성 객체

4.4 부호화 알고리즘 흐름도

그림 14는 부호화 알고리즘의 흐름도를 나타낸 것이다. 원 모델에 초기 과정으로 과동 분할이 적용되고, 모든 처리는 독립된 SPB내에서 이루어진다.

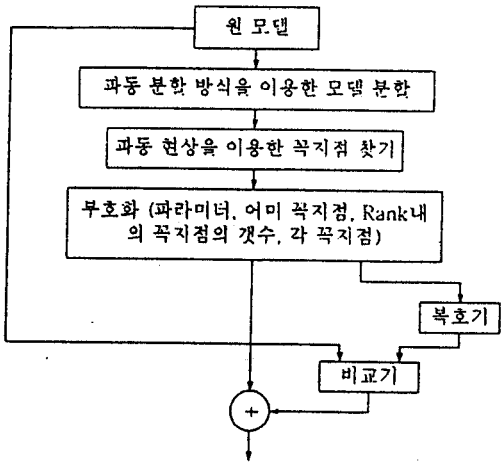


그림 14. 부호화 흐름도

만약 분할이 이루어지지 않는다면 하나의 SPB가 모델 전체로 구성된다. 꼭지점은 독립된 SPB내에서 원 또는 반원을 그리며 구해진다. 이렇게 구해진 꼭지점들은 각 꼭지점의 어미 꼭지점과 Rank내의 꼭지점의 개수, 분리된 SPB를 나타내는 1비트 태그(tag), 그리고 어미 꼭지점의 차분이 2 이상일 경우 발생하는 특별한 유형을 나타내는 인덱스와 함께 부호화되어 전송된다. 여기서 유의해야 할 점은, 부호화기의 마지막 부분에 복호기를 설치하는 것이다. 이것은 복호화부에서 찾지 못하는 삼각형의 연결성 정보가 존재할 경우를 대비한 것으로 이들에 대한 정보는 부호화를 거치지 않고, 다른 정보들과 같이 전송된다.

5. 실험 및 결과

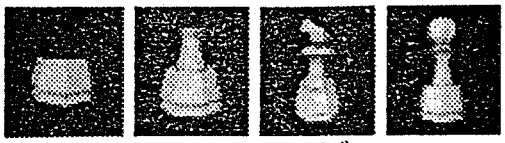
본 논문에서는 삼차원 모델을 4개로 분할하여 부호화하였다. 분할된 SPB의 크기는 초기 메쉬에 덧붙여지는 삼각형의 개수를 기준으로 정하였다. 하지만 향상된 접선 정제 과정을 통해 분할된 SPB내의 삼각형의 수는 조금 달라질 수 있다. 그리고 꼭지점 정보의 압축을 배제하고, 연결성 정보만을 부호화하는데 초점을 두었다.

그림 15는 본 알고리즘을 PAWN 모델, MUSHROOM 모델, 그리고 KING 모델에 적용하여 삼차원 모델이 순차적으로 복호화되는 과정을 나타내고 있다.

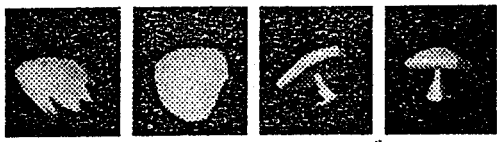
TRICERATOPS 모델과 BEETHOVEN 모델은 복합 구성 객체에 해당하는 경우로 원 모델이 각각 24개와 17개의 독립된 객체로 나누어져 있기 때문에 원래의 모델에 충실하게 가장 큰 부분에 대해서 순차적 부호화를 수행하였다. 단, 부호화 효율은 전체 모델에 대한 것이다.

표 4는 주어진 꼭지점의 개수와 주어진 연결성 정

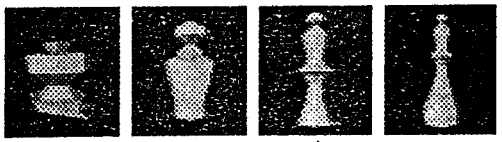
보를 차지하는 모델에 대한 제안한 알고리즘의 효율을 나타낸다. 표 5에서는 2500개 이상의 꼭지점을 가진 모델에 대한 IBM[7] 알고리즘과 제안한 알고리즘의 효율을 비교 분석하였다.



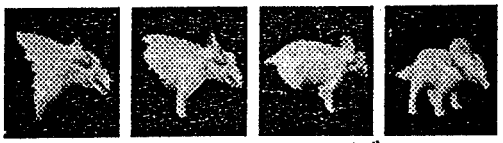
(a) PAWN 모델



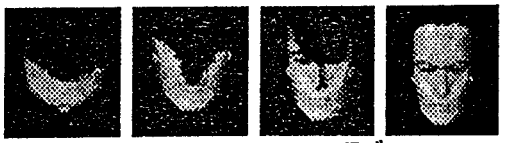
(b) MUSHROOM 모델



(c) KING 모델



(d) TRICERATOPS 모델



(e) BEETHOVEN 모델

그림 15. 순차적 복호화

표 4. 부호화 결과

	꼭지점 (개)	원 모델 (Bytes)	부호화 효율 (bits/vertex)
PAWN	154	1824	1.4
KING	314	3744	1.17
MUSHROOM	226	2688	1.6
SHAPE	2562	30720	0.95
TRICERATOPS	3100	33960	3.31
BEETHOVEN	2845	30168	3.32

표 5. 결과 비교 (bits/vertex)

	제안된 알고리즘	IBM 방식
SHAPE	0.95	2.2
TRICERATOPS	3.31	4.3
BEETHOVEN	3.32	4.8

6. 결론

본 논문에서는 오류 내성과 연결성 정보의 효율적인 압축을 위해 파동분할 방식에 기반하여 꼭지점들 간의 특징적인 관계를 이용하는 순차적으로 메쉬를 압축 부호화하는 방식을 제안하였다.

제안한 방식에서는 삼차원 모델의 저장과 전송을 위해 기하학 정보와 연결성 정보를 모두 전송하는 방식에서 벗어나, 기하학 정보와 그것의 부가 정보만을 이용하여 삼차원 모델을 부호화하였다. 제안한 방식을 이용하면 하나의 모델에 있어서 원하는 임의의 부분을 독립적으로 부호화할 수 있다.

따라서 각 SPB간의 오류 전달을 막을 수 있고, 오류가 발생한 SPB만을 재처리하면 전체 모델을 다시 부호화하는 번거로움을 피할 수 있다. 또한 파동 분할 방식에 기반을 두고, 인접한 꼭지점의 어미 꼭지점들 간의 차분을 이용한 부호화 방식을 통해 연결성 정보에 대해 꼭지점 당 약 1.5 ~ 3.0비트 정도의 높은 부호화 효율을 얻을 수 있었다.

감사의 글

본 연구는 광주과학기술원(K-JIST) 초고속광네트워크연구센터(UFON)를 통한 한국과학재단 우수연구센터(ERC)와 교육부 두뇌한국21(BK21) 정보기술사업단의 지원에 의한 것입니다.

참고 문헌

[1] H. Hoppe, "Progressive Meshes," *SIGGRAPH, pp.* 99-108, Aug. 1996

[2] M. Garland and P. S. Heckbert, "Surface Simplification using Quadric Error Metrics", *SIGGRAPH, pp.* 209-216, Aug. 1997

[3] G.K. Wallace, "The JPEG Still Picture Compression Standard," *Communication of the ACM,* 34(4), *pp.* 31-44, April 1991

[4] 송문섭, 안정환, 김성진, 한만진, 호요성, "삼차원 모델의 점진적인 렌더링과 오류 강인을 위한 데이터 분할 방법(CODAP)," 전자공학회 추계 학술 대회, *pp.* 1089-1092, 11월 2000

[5] Z. Yan, S. Kumar, J. Li and C.C. Jay Kuo, "Robust Coding of 3D Graphic Models using Mesh Segmentation and Data Partitioning," *Proc. IEEE Int. Conf. Image Processing*, October 1999

[6] M. Deering, "Geometric compression," *Computer Graphics Pro., pp.* 13-20, Aug. 1995

[7] G. Taubin and J. Rossignac, "Geometry compression through topological surgery," *ACM Trans. Graphics*, vol. 17, 1998

[8] 김태완, 안정환, 임동근, 호요성, "파동 분할 방식의 순차적 삼차원 메쉬 압축 부호화," 신호처리 합동 학술대회, *pp.* 125-128, 9월 2001

[9] W.J. Schroeder and J.A. Zarge, "Decimation of Triangle Meshes," *SIGGRAPH, pp.* 65-70, July 1992