(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(51) 。Int. Cl. [6]
G06T 9/00

(45) 공고일자 2001년07월12일
(11) 등록번호 10-0294926
(24) 등록일자 2001년04월23일

(21) 출원번호 10-1999-0009528
(22) 출원일자 1999년03월20일

(65) 공개번호 특2000-0016837
(43) 공개일자 2000년03월25일

(30) 우선권주장 1019980035419 1998년08월29일 대한민국(KR)

(73) 특허권자
삼성전자 주식회사
윤종용
경기 수원시 팔달구 매탄3동 416

(72) 발명자
한만진
서울특별시서초구서초동1495-1현대빌라트602호
김성진
경기도수원시팔달구매탄4동삼성1차아파트6동1009호
송문섭
경기도용인시기흥읍농서리산14-1
장의선
경기도수원시팔달구영통동신나무실쌍용아파트542동904호
정석윤
서울특별시서초구잠원동52-2신반포13차아파트328동601호
서양석
서울특별시송파구풍납동219미성아파트3동501호

(74) 대리인
이영필
권석흠
이상용

심사관 : 강해성

# (54) 점진적인 삼차원 메쉬 정보의 부호화/복호화 방법 및 장치

요약

본 발명은 점진적인 삼차원 메쉬 정보의 부호화/복호화 방법 및 장치를 개시한다. 본 발명의 바람직한 일실시예에 따른 점진적인 삼차원 메쉬 정보의 부호화 방법은, (a) 입력된 삼차원 메쉬를 복수개의 부분 메쉬들로 재구성하는 단계; (b) 복수개의 부분 메쉬들을 각각 부호화하는 단계; 및 (c) 복수개의 부호화된 부분 메쉬들을 압축된 비트 스트림으로

통합하여 전송하는 단계를 구비하며, 점진적인 삼차원 메쉬 정보의 부호화 방법에 의해 부호화되어 전송된 비트 스트림을 점진적으로 복호화하는 방법은, (a) 전송된 비트 스트림을 복수개의 부호화된 부분 메쉬들로 나누는 단계; (b) 복수개의 부호화된 부분 메쉬들을 각각 복호화하는 단계; 및 (c) 복수개의 복호화된 부분 메쉬들을 합성하여 삼차원 메쉬를 복원하는 단계를 구비한다.

## 대표도

도 3

## 명세서

### 도면의 간단한 설명

도 1은 종래의 삼차원 메쉬 정보 부호화/복호화 방식을 개념적으로 나타낸 블럭도이다.

도 2는 본 발명에 의한 점진적인 삼차원 메쉬 정보의 구조를 개념적으로 표현한 도면이다.

도 3은 본 발명에 의한 점진적인 삼차원 메쉬 정보 부호화/복호화 방식을 개념적으로 나타낸 바람직한 일실시예에 따른 블럭도이다.

도 4는 도 3에 도시된 삼차원 데이타 분석기의 바람직한 실시예에 따른 블럭도이다.

도 5는 본 발명에 의한 점진적인 삼차원 메쉬 정보 부호화/복호화 방식을 개념적으로 나타낸 바람직한 다른 실시예에 따른 블럭도이다.

도 6은 도 5에 삼차원 MOL 합성기가 더 포함된 블럭도이다.

도 7은 본 발명에 의한 점진적인 삼차원 메쉬 정보 부호화/복호화 방식을 개념적으로 나타낸 바람직한 또다른 실시예에 따른 블럭도이다.

도 8은 도 7에 삼차원 데이터 합성기가 더 포함된 블럭도이다.

도 9는 본 발명의 이해를 돕기위해 하나의 단계별 메쉬(MOL)로 구성된 간단한 삼차원 메쉬(MO)의 일예를 나타내는 도면이다.

도 10은 도 9에 도시된 단계별 메쉬(MOL)를 세개로 나눈 부분 메쉬(MCOM0~MCOM2)의 일예를 나타내는 도면이다.

도 11 (a)~(c)는 단계별 메쉬(MOL)를 부분 메쉬들(MCOM0~MCOM2)로 나눌때 각 부분 메쉬에 포함되는 정보의 예를 설명하기 위한 도면이다.

도 12 (a)~(c)는 두개의 부분 메쉬 사이에 공유되는 정보를 처리하는 방법의 예를 설명하기 위한 도면이다.

### 발명의 상세한 설명

#### 발명의 목적

##### 발명이 속하는 기술 및 그 분야의 종래기술

본 발명은 삼차원 메쉬 정보 부호화/복호화에 관한 것으로서, 특히 MPEG-4 SNHC (Synthetic and Natural Hybrid Coding) 분야와 VRML(Virtual Reality Modeling Language)등에서 사용되고 있는 삼차원 메쉬(Mesh) 정보에 대한 점진적인 부호화/복호화 방법 및 그 장치에 관한 것이다.

삼차원 메쉬로 이루어지는 삼차원 객체의 전송에서는 메쉬 데이터의 효율적인 부호화 뿐만 아니라, 전송되는 메쉬 데이터의 점진적인 복원(progressive reconstruction)이 중요한 필요조건으로 인식되고 있다. 점진적인 복원에 의해, 전송도중 선로상의 오류로 메쉬 데이터에 손실이 발생하였을 경우 이미 전송된 메쉬 데이터만을 가지고도 일부 복원이 가능하게 됨으로써, 재전송되어야 할 메쉬 데이터의 양이 최소화된다. 점진적인 복원 기법은 이러한 선로 오류에 강한 특징을 가지므로 향후의 무선 통신 또는 저전송률 통신 등에서 효과적으로 사용될 수 있는 기술로 전망되고 있다.

도 1은 종래의 삼차원 메쉬 정보 부호화/복호화 장치를 개념적으로 나타낸 블록도이다. 도 1에 의하면, 부호화부(101)는 연결정보 부호화기(102), 위치 부호화기(103) 및 엔트로피 부호화기(104)로 구성되며, 복호화부(112)는 엔트로피 복호화기(106), 연결정보 복호화기(107) 및 위치 복호화기(108)로 구성된다.

MPEG에서 사용하는 종래의 삼차원 메쉬 데이터의 압축 방식을 도 1을 참조하여 설명하면 다음과 같다. 부호화부(101)로 입력된 삼차원 메쉬 데이터(100)는 연결정보와 위치정보로 나누어져 각각 연결정보 부호화기(102)와 위치 부호화기(103)에 의해 부호화된다. 이때, 꼭지점 구조에 대한 정보(105)는 연결정보 부호화기(102)에서 위치 부호화기(103)로 전달된다. 연결정보 부호화기(102)와 위치 부호화기(103)에 의해 압축된 정보는 다시 엔트로피 부호화기(104)를 통해 압축된 비트 스트림(111)으로 변환된다.

압축된 비트 스트림(111)은 복호화부(112)로 입력되어 다음과 같이 복원된다. 압축된 비트 스트림(111)은 엔트로피 복호화기(106)를 거쳐 연결정보 및 위치정보로 나누어져 각각 연결정보 복호화기(107)와 위치 복호화기(108)에 의해 복호화된다. 부호화부(101)에서와 비슷하게 꼭지점 구조에 대한 정보(109)는 연결정보 복호화기(107)에서 위치 복호화기(108)로 전달된다. 복호화된 연결정보와 복호화된 위치정보를 이용하여 삼차원 메쉬(110)를 복원할 수 있다.

도 1에 도시된 바와 같이, 하나의 삼차원 메쉬는 통신 선로상에서 압축된 비트 스트림의 형태로 전송된다. 그런데, 종래의 방식은 엔트로피 부호화기를 사용하기 때문에 통신 선로상에서 발생할 수 있는 전송 에러에 취약하다는 결함을 지닌다.

정리하면, 종래의 삼차원 메쉬 데이터에 대한 부호화는 메쉬 데이터 전체를 단위로 부호화하기 때문에, 부호화된 데이터의 전송시에 전체의 비트 스트림 (bit stream)을 모두 전송받기 전에는 그 일부를 복원하는 것이 거의 불가능하였다. 또한, 전송시에 발생하는 통신 선로상의 에러로 인해 극히 일부의 데이터가 손실되더라도 메쉬 데이타 전체를 다시 전송받아야 하는 비효율적인 문제점이 있었다. 그 예로서, 현재 MPEG-4 SNHC 삼차원 메쉬 부호화에 기술로 채택되어 있는 IBM사에서 제안한 부호화 방식을 들 수 있다(ISO/IEC JTC1/SC29/WG11 MPEG98/W2301,'MPEG-4 SNHC Verification Model 9.0).

## 발명이 이루고자 하는 기술적 과제

본 발명이 이루고자하는 과제는, 모델을 부분별로 처리되도록 재구성함으로써 부분별 구분 및 부분별 복원이 가능하며, 이에 따라 점진적인 영상 재현과 전송 에러에 대한 대응능력이 뛰어난, 점진적인 삼차원 메쉬 정보의 부호화/복호화 방법 및 그 장치를 제공하는데 있다.

본 발명이 이루고자하는 다른 과제는, 모델을 서로 독립된 단계별 메쉬 또는 부분 메쉬로 분리함으로써 독립적으로 부호화 및 복호화가 가능하며, 이에 따라 점진적인 영상 재현과 전송 에러에 대한 대응능력이 뛰어난, 점진적인 삼차원 메쉬 정보의 부호화/복호화 방법 및 그 장치를 제공하는데 있다.

## 발명의 구성 및 작용

상기 과제를 이루기 위하여, 본 발명의 바람직한 일실시예에 따른 점진적인 삼차원 메쉬 정보의 부호화 방법은, (a) 삼차원 메쉬를 복수개의 부분 메쉬들로 재구성하는 단계; (b) 복수개의 부분 메쉬들을 각각 부호화하는 단계; 및 (c) 복수개의 부호화된 부분 메쉬들을 압축된 비트 스트림으로 통합하여 전송하는 단계를 구비한다.

또한, 상기 과제를 이루기 위하여, 점진적인 삼차원 메쉬 정보의 부호화 방법에 의해 부호화되어 전송된 비트 스트림을 점진적으로 복호화하는 방법은, (a) 상기 전송된 비트 스트림을 복수개의 부호화된 부분 메쉬들로 나누는 단계; (b) 상기 복수개의 부호화된 부분 메쉬들을 각각 복호화하는 단계; 및 (c) 복수개의 복호화된 부분 메쉬들을 합성하여 삼차원 메쉬를 복원하는 단계를 구비한다.

또한, 상기 과제를 이루기 위하여, 본 발명의 바람직한 일실시예에 따른 점진적인 삼차원 메쉬 정보의 부호화 장치는, 삼차원 메쉬를 복수개의 부분 메쉬들로 재구성하는 삼차원 데이터 분석부; 복수개의 부분 메쉬들을 각각 부호화하는 복수개의 부분별 부호화부들; 및 복수개의 부호화된 부분 메쉬들을 압축된 비트 스트림으로 통합하여 전송하는 멀티플렉서를 구비한다.

또한, 상기 과제를 이루기 위하여, 점진적인 삼차원 메쉬 정보의 부호화 장치에 의해 부호화되어 전송된 비트 스트림을 점진적으로 복호화하는 장치는, 전송된 비트 스트림을 복수개의 부호화된 부분 메쉬들로 분류하는 디멀티플렉서; 복수개의 부호화된 부분 메쉬들을 각각 복호화하는 복수개의 부분별 복호화부들; 및 복수개의 복호화된 부분 메쉬들을 합성하여 삼차원 메쉬를 복원하는 삼차원 데이터 합성부를 구비한다.

상기 다른 과제를 이루기 위하여, 본 발명의 바람직한 다른 실시예에 따른 점진적인 삼차원 메쉬 정보의 부호화/복호화 방법은, (a) 삼차원 메쉬로부터 서로 독립된 하나 이상의 단계별 메쉬들을 추출하는 단계; (b) 상기 하나 이상의 단계별 메쉬들을 각각 서로 독립적으로 부호화하고, 서로 독립적으로 전송하는 단계; 및 (c) 서로 독립적으로 부호화하여 전송된 하나 이상의 단계별 메쉬들을 각각 복호화하여 서로 독립된 하나 이상의 단계별 메쉬들을 얻는 단계를 구비하며, (d) 서로 독립된 단계별 메쉬들을 모으고, 중복되는 정보를 제거하여 원래의 삼차원 메쉬로 복원하는 단계를 더 포함한다.

또한, 상기 다른 과제를 이루기 위하여, 본 발명의 바람직한 다른 실시예에 따른 점진적인 삼차원 메쉬 정보의 부호화/복호화 장치는, 삼차원 메쉬를 입력하고, 입력된 삼차원 메쉬로부터 서로 독립된 하나 이상의 단계별 메쉬들을 추출하는 삼차원 단계별 메쉬 분석부; 상기 하나 이상의 단계별 메쉬들을 각각 서로 독립적으로 부호화하고, 서로 독립적으로 전송하는 하나 이상의 단계별 메쉬 부호화부들; 및 서로 독립적으로 부호화하여 전송된 하나 이상의 단계별 메쉬들을 각각 복호화하여 서로 독립된 하나 이상의 단계별 메쉬들을 얻는 하나 이상의 단계별 메쉬 복호화부들을 구비하며, 서로 독립된 하나 이상의 단계별 메쉬들을 모으고, 중복되는 정보를 제거하여 원래의 삼차원 메쉬로 복원하는 삼차원 단계별 메쉬 합성부를 더 포함한다.

또한, 상기 다른 과제를 이루기 위하여, 본 발명의 바람직한 또다른 실시예에 따른 점진적인 삼차원 메쉬 정보의 부호화/복호화 방법은, (a) 삼차원 메쉬로부터 하나 이상의 단계별 메쉬를 추출하고, 서로 독립된 복수개의 부분 메쉬들로 분할하는 단계; (b) 복수개의 부분 메쉬들을 각각 서로 독립적으로 부호화하고, 서로 독립적으로 전송하는 단계; 및 (c) 서로 독립적으로 부호화하여 전송된 복수개의 부분 메쉬들을 각각 복호화하여 서로 독립된 복수개의 부분 메쉬들을 얻는 단계를 구비하며, (d) 서로 독립된 부분 메쉬들을 모으고, 인접한 부분 메쉬들간에 중복되는 정보를 제거하여 원래

의 삼차원 메쉬로 복원하는 단계를 더 포함한다.

또한, 상기 다른 과제를 이루기 위하여, 본 발명의 바람직한 또다른 실시예에 따른 점진적인 삼차원 메쉬 정보의 부호화/복호화 장치는, 삼차원 메쉬를 입력하여 하나 이상의 단계별 메쉬들을 추출하고, 서로 독립된 복수개의 부분 메쉬들로 분할하는 삼차원 데이터 분석부; 복수개의 단계별 메쉬들을 각각 서로 독립적으로 부호화하고, 서로 독립적으로 전송하는 복수개의 부분 메쉬 부호화부들; 및 서로 독립적으로 부호화하여 전송된 복수개의 부분 메쉬들을 각각 복호화하여 서로 독립된 복수개의 부분 메쉬들을 얻는 복수개의 부분 메쉬 복호화부들을 구비하며, 서로 독립된 복수개의 부분 메쉬들을 모으고, 인접한 부분 메쉬들간에 중복되는 정보를 제거하여 원래의 삼차원 메쉬로 복원하는 삼차원 데이터 합성부를 더 포함한다.

이하, 본 발명에 의한 점진적인 삼차원 메쉬 정보의 부호화/복호화 방법 및 장치를 첨부한 도면을 참조하여 다음과 같이 설명한다.

도 2는 본 발명에 의한 점진적인 삼차원 메쉬 정보의 구조를 개념적으로 표현한 도면이다.먼저 점진적인 삼차원 메쉬의 처리를 위하여 본 발명에서는 새로운 메쉬 구조를 도 2에 도시된 바와 같이 제안한다. 도 2에 의하면, 삼차원 메쉬(Mesh Object : MO)는 메쉬의 정보를 여러 단계로 분류한 단계별 메쉬(Mech Object Layer : MOL)로 구성될 수 있다. 이때, 각 MOL은 하나 또는 그 이상의 부분 메쉬(MCOM:Mesh COMponent)를 포함하게 된다. 하나의 MCOM은 그 자체의 복원에 필요한 연결정보(Connectivity information), 위치정보(Geometry information), 그리고 기타 정보로서 화상정보(Photometry information) 등을 포함한다. 즉, MO는 부호화할 삼차원 메쉬 객체 단위로 정의되며, 이들 메쉬 정보의 다양한 화질과 기능 측면에 따라 여러 단계로 분류하고, 이 분류된 각 단계를 단계별 메쉬(MOL)로 정의한다. 아울러 위상기하학적 서져리 방법을 이용하여 하나의 삼차원 메쉬 객체를 상호 연결성이 없는 여러개의 독립된 메쉬 정보(즉, 연결성분)들로 구성하였을 때, 부호화할 데이터의 크기나 그 외의 특성에 따라 상호 독립된 메쉬 정보를 통합하거나 분할시켜 구성한 것을 부분 메쉬로 정의한다.

도 3에 의하면, 본 발명에 의한 점진적인 삼차원 메쉬 정보 부호화/복호화 장치를 개념적으로 나타낸 바람직한 일실시예는 부호화부(200) 및 복호화부(209)로 구성된다. 그리고, 부호화부(200)는 삼차원 데이터 분석기(201), 복수개의 부분별(Component) 부호화기1~N(202) 및 멀티플렉서(MUX)(204)로 구성되며, 복호화부(209)는 디멀티플렉서(DMUX)(205), 복수개의 부분별 복호화기1~N(206) 및 삼차원 데이터 합성기(208)로 구성된다.

도 3을 참조하면, 먼저 삼차원 메쉬(MO)(100)가 삼차원 데이터 분석기(201)에서 복수개의 부분 메쉬(MCOM)들로 재구성되고, 복수개의 부분 메쉬(MCOM)들은 각각 복수개의 부분별 부호화기1~N(202)으로 입력된다. 이때, 부분 메쉬(MCOM)들은 여러개가 모여서 하나의 단계별 메쉬(MOL)를 이룰 수 있다. 각각의 부분 메쉬(MCOM)는 해당하는 부분별 부호화기(202)를 통하여 압축되고, 각각의 압축된 비트 스트림은 MUX(204)에 의해 통합되어 전송된다. 이때 이미 수행된 한 단계별 메쉬(MOL) 또는 부분 메쉬(MCOM)의 부분별 부호화기에서 사용된 정보(203)는 아직 수행되지 않은 부분별 부호화기에서 이용될 수 있다. 예컨대, 상위 부분별 부호화기인 부분별 부호화기1에서 사용된 정보(203)는 하위 부분별 부호화기인 부분별 부호화기2로 전송되어 이용될 수 있다.

복호화부(209)로 전송된 압축된 비트 스트림은 DEMUX(205)를 통해 각 단계별 메쉬(MOL)들로 분류되고, 단계별 메쉬(MOL)들은 다시 부분 메쉬(MCOM)들로 나누어지고, 각각의 부분 메쉬(MCOM)들은 복수개의 부분별 복호화기1~N(206)을 통해 복호화된다. 복호화부(209)에서도 이미 수행된 부분별 복호화기에서 생성된 정보(207)는 아직 수행되지 않은 부분별 복호화기에서 재사용된다. 예컨대, 상위 부분별 복호화기인 부분별 복호화기1에서 생성된 정보(207)는 하위 부분별 복호화기인 부분별 복호화기2로 전송되어 재사용될 수 있다. 단계별로 복호화된 부분 메쉬(MOL)

들은 삼차원 데이터 합성기(208)에 의해 삼차원 메쉬(110)로 복원된다.

삼차원 메쉬(MO)를 우선 하나 이상의 단계별 메쉬(MOL)들로 구성한 다음, 이들을 각각 부분 메쉬(MCOM)들로 나누는 경우에, 삼차원 데이터 분석기(201)는 도 4와 같이 구성될 수 있다.

도 4에 의하면, 도 3에 도시된 삼차원 데이타 분석기(201)의 바람직한 실시예는 삼차원 단계별 메쉬(MOL) 분석기(301) 및 복수개의 부분 메쉬(MCOM) 분석기1~n(303)로 구성된다.

삼차원 메쉬(MO)(100)가 삼차원 데이터 분석기(300)로 입력되면, 우선 삼차원 MOL 분석기(301)에서는 입력된 삼차원 메쉬(MO)에서 각 단계별 메쉬들(MOL1~MOLn)(302)을 추출한 다음, 추출된 각 단계별 메쉬(MOL1~MOLn)(302)는 부분 메쉬(MCOM) 분석기1~n(303)를 통해 부분 메쉬들(304)로 나누어져서 출력된다. 각 부분 메쉬(304)는 부분별 부호화기1-1 내지 부분별 부호화기1-m, 부분별 부호화기2-1 내지 부분별 부호화기2-m,... 중 대응하는 부분별 부호화기로 입력된다.

각 부분메쉬(MCOM) 분석기는 다른 부분메쉬(MCOM) 분석기에서 사용된 정보를 사용하며 각 부분별 부호화기는 동일한 부분메쉬(MCOM) 분석기에 대응하는 다른 부분별 부호화기에서 사용된 정보를 사용하는데, 이때 서로 다른 단계별 메쉬 사이(305)에서는 임의의 부분메쉬(MCOM) 분석기의 정보를 다른 부분메쉬(MCOM) 분석기에서 사용하지 않을 수도 있다. 이 경우에, 독립적인 단계별 메쉬 정보 부호화/복호화 방식은 도 5와 같이 구성될 수 있다.

도 5에 의하면, 본 발명에 의한 점진적인 삼차원 메쉬 정보 부호화/복호화 방식의 바람직한 다른 실시예는 삼차원 단계별 메쉬(MOL) 분석기(401), 복수개의 독립적 부호화기/복호화기1~n(403)로 구성된다.

도 5에 도시된 실시예에서는 임의의 부호화기에서 생성된 정보를 다른 부호화기가 사용하지 않는다. 즉, 삼차원 메쉬(MO)(100)가 삼차원 MOL 분석기(401)를 통하여 단계별 메쉬들(MOL1~n)(402)로 분리된 다음, 각각 독립적인 부호화기/복호화기1~n(403)을 통해 압축, 전송, 복호화되어 복원된다. 여기서 사용되는 부호화기/복호화기는 각각 MCOM 분석기를 포함하여 단계별 메쉬를 다시 부분 메쉬들로 나누어 각각 부호화 및 복호화할 수 있다. 각 복호화기를 통해 복호화된 각 정보(404)는 서로 독립된 단계별 메쉬 데이터이기 때문에, 이들을 단순히 모으면 삼차원 메쉬(405)가 복원된다. 그러나, 이러한 경우 삼차원 MOL 분석기(401)에서 삼차원 메쉬(MO)(100)를 단계별 메쉬들(MOL1~n)(402)로 나누면서 생성된 부가적인 정보도 복원된 삼차원 메쉬(405)에 포함된다.

도 6에 의하면, 도 5에 도시된 바람직한 다른 실시예는 삼차원 MOL 합성기(406)를 더 포함하여 구성된다. 복원된 삼차원 메쉬(405)에 포함된 부가적인 정보를 제거하여 원래의 삼차원 메쉬(M0)(100)와 동일하게 복원된 삼차원 메쉬(110)를 얻으려면, 삼차원 MOL 합성기(406)를 추가하면 된다. 삼차원 MOL 합성기(406)는 각 복호화기에 의해 복호화된 정보(404)을 모으고 중복되는 정보를 제거함으로써, 원래의 삼차원 메쉬(110)를 복원한다.

한편, 도 4에서 각 부분별 부호화기는 다른 부분별 부호화기에서 사용된 정보를 사용하는데, 이때 부분 메쉬 사이(306)에서도 이미 수행된 부분별 부호화기의 정보를 아직 수행되지 않은 부분별 부호화기에서 사용하지 않을 수도 있다. 이 경우에, 독립적인 부분 메쉬 정보 부호화/복호화 방식은 도 7과 같이 구성될 수 있다.

도 7에 의하면, 본 발명에 의한 점진적인 삼차원 메쉬 정보 부호화/복호화 방식의 바람직한 또다른 실시예는 삼차원 데이터 분석기(501), 복수개의 독립적 부분 메쉬(MCOM) 부호화기/복호화기1~n(503)로 구성된다.

도 7에 도시된 실시예에서는 이미 수행된 부호화기의 정보를 아직 수행되지 않은 부호화기에서 사용하지 않는다. 즉, 삼차원 메쉬(MO)(100)가 삼차원 MOL 분석기 및 MCOM 분석기를 포함하는 삼차원 데이터 분석기(501)를 통하여

부분 메쉬들(MCOM1~n)(502)로 재구성된 다음, 각각 독립적으로 MCOM 부호화기/복호화기1~n(503)을 통해 압축, 전송, 복호화되어 복원된다. 각 복호화기를 통해 복호화된 각 정보(504)는 서로 독립된 부분 메쉬 데이터이기 때문에, 이들을 단순히 모으면 삼차원 메쉬(505)가 복원된다. 그러나, 삼차원 데이터 분석기(501)에서 하나의 단계별 메쉬를 복수개의 독립적인 부분 메쉬로 나눌 경우에, 각 부분 메쉬는 경계 부분의 정보(예컨대, 엣지, 점의 좌표 및 점 단위의 속성등)를 각각 공유하고 있다. 따라서, 복원된 삼차원 메쉬(505)에는 중복되는 정보가 많이 존재한다.

도 8에 의하면, 도 7에 도시된 바람직한 또다른 실시예는 삼차원 데이터 합성기(506)를 더 포함하여 구성된다. 복원된 삼차원 메쉬(505)에 포함된 중복된 정보를 제거하여 부호화 효율을 높이기 위해서, 도 8과 같이 삼차원 데이터 합성기(506)가 추가된다. 삼차원 데이터 합성기(506)는 각 복호화기에 의해 복호화된 정보(504)를 모으고 중복되는 정보를 제거함으로써, 원래의 삼차원 메쉬(110)를 복원한다.

도 9는 본 발명의 이해를 돕기위해 하나의 단계별 메쉬(MOL)로 구성된 간단한 삼차원 메쉬(MO)를 예시한 것이며, 도 10은 도 9에 도시된 단계별 메쉬(MOL)를 세개로 나눈 부분 메쉬(MCOM0~MCOM2)들을 예시한 것이다. 도 10에서 서로 독립적인 부분 메쉬들간의 경계 부분은 굵은 선으로 표시되어 있다. 이 경계 부분에 해당하는 정보(예컨대, 엣지, 점의 좌표 및 점 단위의 속성등)는 앞선 부분 메쉬에 공유되어 있다. 따라서, 뒤의 부분 메쉬는 앞선 부분 메쉬의 복호화/부호화 과정에서 생성된 정보를 일부 또는 모두 재사용하여 복호화/부호화할 수 있다.

도 11 (a)~(c)는 단계별 메쉬(MOL)를 부분 메쉬들(MCOM0~MCOM2)로 나눌때 각 부분 메쉬에 포함되는 정보의 예를 설명하기 위한 도면이다.

도 11 (a)는 단계별 메쉬(MOL)를 부분 메쉬들(MCOM0~MCOM2)로 나눌때 부분 메쉬들간의 경계 부분에 있는 정보가 공유되는 것을 나타낸다. 경계 부분에 있는 정보중에서 일부 또는 모든 정보가 복사되어 서로 인접한 부분 메쉬가 동일한 정보를 각각 보유할 수 있다. 도 11 (b)는 인접한 부분 메쉬간에 공유되는 정보를 각각 갖을 경우에 부분 메쉬들(MCOM0~MCOM2)을 나타낸다. 이와 같이 각 부분 메쉬마다 공유하는 정보의 중복을 허용하면, 모든 부분 메쉬들이 독립적으로 처리될 수 있지만, 압축 효율이 떨어진다.

한편, 도 11 (c)는 인접한 부분 메쉬간에 공유되는 정보를 먼저 부호화될 부분 메쉬에서만 갖을 경우에 부분 메쉬들(MCOM0~MCOM2)을 나타낸다. 이미 부호화된 부분 메쉬와 공유하는 정보가 있는 경우에, 이미 부호화된 부분 메쉬에 의해 생성된 정보가 아직 부호화되지 않은 부분 메쉬를 위해 사용되기 때문에 중복되는 정보가 없지만, 각 부분 메쉬는 독립적으로 처리되지 못한다.

도 12 (a)~(c)는 두개의 부분 메쉬 사이에 공유되는 정보를 처리하는 방법의 예를 설명하기 위한 도면이다.

도 12 (a)는 두개의 부분 메쉬가 서로 독립적으로 나누어진 예를 나타낸다. 여기서, 두개의 부분 메쉬는 동일한 변, 점 및 점의 속성들을 공유하고 있다. 이러한 정보가 양측에 모두 복사되어 두개의 부분 메쉬는 분리된 후에 각각 독립적으로 처리된다. 이때, 처리과정에서 각 부분 메쉬에 복사된 정보는 각각 다른 정보로 인식되기 때문에, 복호화부에서 각각 복호화된 각 정보를 모아서 단계별 메쉬를 복원시키면, 원래의 단계별 메쉬의 형태가 변경되며, 결과 파일의 크기가 증가하게 된다. 도 5 또는 도 7를 참조하면, 도 12 (a)와 같이 서로 독립적인 부분 메쉬들은 서로 독립적인 부호화기/복호화기에 의해 독립적으로 처리된다.

도 12 (b)는 각 부분 메쉬를 완전히 분리하지 않고, 공유되는 정보중 일부 정보만 중복하여 부호화하는 예를 나타낸다. 앞선 부분 메쉬는 공유되는 정보를 전부 갖고 있지만, 다음 부분 메쉬는 공유되는 정보중 일부로서 앞선 부분 메쉬와

동일한 점에 대한 정보만 갖고 있다. 이 경우에 공유된 동일한 점이 두개 이상의 부분 메쉬에서 정의되어 있지만, 이것이 동일한 점으로 인식되기 때문에 복원후에 원래의 단계별 메쉬의 형태가 유지된다. 또한, 이전 부분 메쉬가 복원이 안된 경우에는 다음 부분 메쉬에 의해 경계 부분의 복원이 가능하다. 도 3을 참조하면, 이미 수행된 부분별 부호화기에서 생성된 정보가 아직 수행되지 않은 부분별 부호화기에 이용될 때, 그 정보는 공유되는 정보 중 일부이다. 아직 수행되지 않은 부분별 부호화기는 이미 수행된 부분별 부호화기와 공유된 동일한 점에 대한 정보를 중복하여 부호화한다.

도 12 (c)는 앞선 부분 메쉬에만 공유되는 정보가 있고, 다음 부분 메쉬에서는 이전 부분 메쉬에서 생성된 정보를 모두 사용하는 예를 나타낸다. 이 경우에, 중복하여 부호화하는 정보가 없기 때문에 압축 효율은 증가하지만, 각 부분 메쉬가 독립성이 없기 때문에 이전 부분 메쉬가 복원되지 않은 경우에는 다음 부분 메쉬에 의해 경계 부분의 복원이 불가능하다. 도 3을 참조하면, 이미 수행된 부분별 부호화기에서 생성된 정보가 아직 수행되지 않은 부분별 부호화기에 이용될 때, 그 정보는 공유되는 정보 전부이다. 아직 수행되지 않은 부분별 부호화기는 이미 수행된 부분별 부호화기와 공유된 모든 정보를 중복하여 복호화하지 않는다.

이하에서 본 발명에 의한 점진적인 삼차원 메쉬 정보의 부호화를 구현한 ISO/IEC JTC1/SC29/WG11 MPEG-4 SNHC 삼차원 메쉬 부호화의 문법표를 표시한다.

▶ 3D_Mesh_Object

MO_start_code : 이것은 동기 목적으로 사용되는 고유한 16비트 코드이다. 이 코드의 값은 항상 '0000 0000 0010 0000'이다.

▶ 3D_Mesh_Object_Layer

MOL_start_code : 이것은 동기 목적으로 사용되는 고유한 16비트 코드이다. 이 코드의 값은 항상 '0000 0000 0011 0000'이다.

mol_id : 이 8비트의 부호없는 정수는 단계별 메쉬(mesh object layer:MOL)에 대한 유일한 식별자를 나타낸다. 값 0은 베이스 단계(base layer)를 나타내고, 0보다 큰 값은 정밀 단계(refinement layer)를 나타낸다. 3D_Mesh_Object_Header 뒤의 최초의 3D_Mesh_Object_Layer는 mol_id="0이어야" 하고, 동일한 3D_Mesh_Object내에서 차후의 3D_Mesh_Object_Layer들은 mol_id0이어야 한다.

N_Vertices는 3차원 메쉬의 현재 해상도에서의 꼭지점들의 개수이다. 계산을 줄이기 위해 사용된다.

N_Triangles는 3차원 메쉬의 현재 해상도에서의 삼각형들의 개수이다. 계산을 줄이기 위해 사용된다.

N_Edges는 3차원 메쉬의 현재 해상도에서의 에지(edge)들의 개수이다. 계산을 줄이기 위해 사용된다.

▶ 3D_Mesh_Object_Base_Layer

MOBL_start_code : 이것은 동기 목적으로 사용되는 고유한 16비트 코드이다. 이 코드는 항상 '0000 0000 0011 0001'이다.

mobl_id : 이 8비트의 부호없는 정수는 부분 메쉬(mesh object component:MCOM)에 대한 유일한 식별자를 나타낸다.

last_component : 이 불값(boolean value)은 복호화할 연결성분(connected component)들이 더 있는지 여부를 나타낸다. 만약 last_component가 참(true)이면, 마지막 성분은 복호화되었다. 그렇지 않으면, 복호화할 성분들이 더 있다. 이 필드는 산술적으로 부호화된다.

▶ 3D_Mesh_Object_Header

ccw : 이 불값은 복호화되는 페이스(face)들의 꼭지점 순서가 역시계방향의 순서를 따르는지 여부를 나타낸다.

convex : 이 불값은 모델이 볼록한지 여부를 나타낸다.

solid : 이 불값은 모델이 견고한지 여부를 나타낸다.

creaseAngle : 이 6비트의 부호없는 정수는 크리스 각(crease angle)인지 여부를 나타낸다.

▶ coord_header

coord_binding : 이 2비트 부호없는 정수는 3차원 메쉬에 대한 꼭지점 좌표의 결합을 나타낸다. 유일하고 허용되는 값은 '01'이다.

coord_bbox : 이 불값은 기하정보(geometry)에 대하여 바운딩 상자(bounding box)가 제공되는지 여부를 나타낸다. 바운딩 상자가 제공되지 않으면, 디폴트가 사용된다.

coord_xmin, coord_ymin, coord_zmin : 이 부동소숫점 값들은 기하정보가 놓인 바운딩 상자의 하위 좌측 구석을 나타낸다.

coord_size : 이 부동소숫점 값은 바운딩 상자의 크기를 나타낸다.

coord_quant : 이 5비트의 부호없는 정수는 기하정보에 대한 양자화 스텝(quantization step)을 나타낸다.

coord_pred_type : 이 2비트 부호없는 정수는 메쉬의 꼭지점 좌표들을 재생하기 위해 사용되는 예측의 형태를 나타낸다.

[표 1]

| coord_pred_type | 예측 형태 |
|---|---|
| 00 | 예측 없음 |
| 01 | 무효 |
| 10 | 평행사변형 예측 |
| 11 | 다각형 예측 |

coord_nlambda : 이 2비트 부호없는 정수는 기하정보를 예측하기 위해 사용되는 선조(ancestor)들의 개수를 나타낸다. coord_nlambda에 대해 허용가능한 값들은 1과 3이다. 표 2는 normal_pred_type의 기능으로서 허용가능한 값들을 보여준다.

[표 2]

| coord_pred_type | coord_nlambda |
|---|---|
| 00 | 1 |
| 01 | 무효 |
| 10 | 3 |
| 11 | 1 |

coord_lambda : 이 부호없는 정수는 예측을 위한 선조들에 주어지는 가중치를 나타낸다. 이 필드에서 사용되는 비트들의 수는 coord_quant+3과 같다.

▶ normal_header

normal_binding : 이 2비트 부호없는 정수는 3차원 메쉬에 대한 노멀(normal)들의 결합을 나타낸다. 허용가능한 값들은 표 3에 기술된다.

[표 3]

| normal_binding | 결합 |
|---|---|
| 00 | 어떤 노멀도 부호화되지 않는다 |
| 01 | 하나의 노멀이 꼭지점당 부호화된다 |
| 10 | 하나의 노멀이 페이스당 부호화된다 |
| 11 | 하나의 노멀이 코너당 부호화된다 |

normal_bbox : 이 불값은 항상 거짓('0')이어야 한다.

normal_quant : 이 5비트의 부호없는 정수는 노멀에 대해 사용되는 양자화 스텝(quantization step)을 나타낸다.

normal_pred_type : 이 2비트 부호없는 정수는 어떻게 노멀 값들이 예측되는지를 나타낸다.

[표 4]

| normal_pred_type | 예측 형태 |
|---|---|
| 00 | 예측 없음 |
| 01 | 트리 예측 |
| 10 | 평행사변형 예측 |
| 11 | 무효 |

[표 5]

| normal_binding | normal_pred_type |
|---|---|
| 00 | 무시됨 |
| 01 | 10 |
| 10 | 01 |
| 11 | 01 |

normal_nlambda : 이 2비트 부호없는 정수는 노멀들을 예측하기 위해 사용되는 선조(ancestor)들의 개수를 나타낸다. normal_nlambda에 대해 허용가능한 값들은 1과 3이다. 표 6은 normal_pred_type의 기능으로서 허용가능한 값들을 보여준다.

[표 6]

| normal_pred_type | normal_nlambda |
|---|---|
| 00 | 1 |
| 01 | 1 |
| 10 | 3 |
| 11 | 1 |

normal_lambda : 이 부호없는 정수는 예측을 위한 선조들에 주어지는 가중치를 나타낸다. 이 필드에서 사용되는 비트들의 수는 normal_quant+3과 같다.

▶ color_header

color_binding : 이 2비트 부호없는 정수는 3차원 메쉬에 대한 색상(color)들의 결합을 나타낸다. 허용가능한 값들은 표 7에 기술된다.

[표 7]

| color_binding | 결합 |
|---|---|
| 00 | 어떤 색상도 부호화되지 않는다 |
| 01 | 하나의 색상이 꼭지점당 부호화된다 |
| 10 | 하나의 색상이 페이스당 부호화된다 |
| 11 | 하나의 색상이 코너당 부호화된다 |

color_bbox : 이 불값은 색상들에 대해 바운딩 상자가 주어지는지 여부를 나타낸다.

color_rmin, color_gmin, color_bmin : 이 부동소숫점 값들은 알.지.비(RGB) 공간 내의 바운딩 상자의 하위 좌측 코너의 위치를 나타낸다.

color_size : 이 부동소숫점 값은 색상 바운딩 상자의 크기를 나타낸다.

color_quant : 이 5비트의 부호없는 정수는 색상에 대해 사용되는 양자화 스텝(quantization step)을 나타낸다.

color_pred_type : 이 2비트 부호없는 정수는 어떻게 색상들이 예측되는지를 나타낸다.

[표 8]

| color_pred_type | 예측 형태 |
|---|---|
| 00 | 예측 없음 |
| 01 | 트리 예측 |
| 10 | 평행사변형 예측 |
| 11 | 무효 |

[표 9]

| color_binding | color_pred_type |
|---|---|
| 00 | 무시됨 |
| 01 | 10 |
| 10 | 01 |
| 11 | 01 |

color_nlambda : 이 2비트 부호없는 정수는 색상들을 예측하기 위해 사용되는 선조(ancestor)들의 개수를 나타낸다. color_nlambda에 대해 허용가능한 값들은 1과 3이다. 표 10는 normal_pred_type의 기능으로서 허용가능한 값들을 보여준다.

[표 10]

| color_pred_type | color_nlambda |
|---|---|
| 00 | 1 |
| 01 | 1 |
| 10 | 3 |
| 11 | 1 |

color_lambda : 이 부호없는 정수는 예측을 위한 선조들에 주어지는 가중치를 나타낸다. 이 필드에서 사용되는 비트들의 수는 color_quant+3과 같다.

▶ texCoord_header

texCoord_binding : 이 2비트 부호없는 정수는 3차원 메쉬에 대한 텍스쳐(texture)들의 결합을 나타낸다. 허용가능한 값들은 표 11에 기술된다.

[표 11]

| texCoord_binding | 설명 |
|---|---|
| 00 | 어떤 텍스처도 부호화되지 않는다 |
| 01 | 하나의 텍스처가 꼭지점당 부호화된다 |
| 10 | 하나의 텍스처가 페이스당 부호화된다 |
| 11 | 하나의 텍스처가 코너당 부호화된다 |

texCoord_bbox : 이 불값은 텍스쳐들에 대해 바운딩 상자가 주어지는지 여부를 나타낸다.

texCoord_umin, texCoord_vmin : 이 부동소숫점 값들은 2차원 공간 내의 바운딩 상자의 하위 좌측 코너의 위치를 나타낸다.

texCoord_size : 이 부동소숫점 값은 텍스쳐 바운딩 상자의 크기를 나타낸다.

texCoord_quant : 이 5비트의 부호없는 정수는 텍스쳐에 대해 사용되는 양자화 스텝(quantization step)을 나타낸다.

texCoord_pred_type : 이 2비트 부호없는 정수는 texCoord_binding이 '01'이면 항상 '10'이고, 그렇지 않은 경우에는 '01'이다.

texCoord_nlambda : 이 2비트 부호없는 정수는 텍스쳐들을 예측하기 위해 사용되는 선조(ancestor)들의 개수를 나타낸다. texCoord_nlambda에 대해 허용가능한 값들은 1과 3이다. 표 12는 texCoord_pred_type의 기능으로서 허용가능한 값들을 보여준다.

[표 12]

| texCoord_pred_type | texCoord_nlambda |
|---|---|
| 00 | 1 |
| 01 | 1 |
| 10 | 3 |
| 11 | 1 |

texCoord_lambda : 이 부호없는 정수는 예측을 위한 선조들에 주어지는 가중치를 나타낸다. 이 필드에서 사용되는 비트들의 수는 texCoord_quant+3과 같다.

▶ Cgd_header

N_Proj_surface_Spheres는 사출된 표면구들(Projected Surface Spheres)의 개수이다. 전형적으로, 이 개수는 1과 같다.

x_coord_Center_Point는 사출된 표면구의 중심점(전형적으로, 객체의 무게 중심점)의 x좌표이다.

y_coord_Center_Point는 사출된 표면구의 중심점(전형적으로, 객체의 무게 중심점)의 y좌표이다.

z_coord_Center_Point는 사출된 표면구의 중심점(전형적으로, 객체의 무게 중심점)의 z좌표이다.

Normalized_Screen_Distance_Factor는 사출된 표면구의 반경과 비교하여 가상 스크린이 어디에 위치하는지를 나타낸다. 사출된 표면구의 중심점과 가상 스크린 사이의 거리는 Radius/(Normalized_Screen_Distance_Factor+1)과 같다. Radius는 각 사출된 표면구에 대하여 기술되나, Normalized_Screen_Distance_Factor는 단 한번 기술된다.

Radius는 사출된 표면구의 반경이다.

Min_Proj_Surface는 대응하는 사출된 표면구에 관한 최소의 사출된 표면값이다. 이 값은 종종(그러나, 반드시 그러한 것은 아니다) Proj_Surface 값들 중 하나와 동일하다.

N_Proj_Points는 사출된 표면이 전송될 사출된 표면구상의 점들의 개수이다. 다른 모든 점들에 대하여, 사출된 표면은 선형 보간에 의해 결정된다. N_Proj_Points는 전형적으로 최초의 사출된 표면구에 대해서는 작고(예들들면, 20), 추가적인 사출된 표면구들에 대해서는 매우 작다(예를들면, 3).

Sphere_Point_Coord는 8면체 내의 점 위치에 대한 인덱스이다.

Proj_Surface는 Sphere_Point_Coord에 의해 기술된 점 내의 사출된 표면이다.

▶ 꼭지점_그래프

vg_simple : 이 불값은 현재의 꼭지점 그래프가 간단한지 여부를 지정한다. 간단한 꼭지점 그래프는 어떠한 루프로 포함하지 않는다. 이 필드는 산술적으로 부호화된다.

vg_last : 이 불 플래그는 현재의 런이 현재의 브랜칭 꼭지점으로부터 시작하는 마지막 런인지 여부를 나타낸다. 이 필드는 각 브랜칭 꼭지점의 첫 번째 런, 즉, skip_last 변수가 참(true)일 때에는 부호화되지 않는다. 현재의 꼭지점 런에 대하여 vg_last의 값이 부호화되지 않은 경우에는 거짓(false)인 것으로 간주된다. 이 필드는 산술 부호화된다.

vg_forward_run : 이 불 플래그는 현재의 런이 새로운 런인지 여부를 나타낸다. 만약 그것이 새로운 런이 아니라면, 그것은 그래프 내의 루프를 나타내는 미리 방문된(traversed) 런이어야 한다. 이 필드는 산술 부호화된다.

vg_loop_index : 이 부호없는 정수는 현재의 루프가 연결된 런에 대한 인덱스를 나타낸다. 그것의 단일(unary)의 표현(표 13 참조)은 산술 부호화된다. 만약 변수 openloops가 vg_loop_index와 같다면, 그 단일의 표현에서 뒤따르는 '1'은 삭제된다.

[표 13]

| vg_loop_index | 단일의 표현 |
|---|---|
| 0 | 1 |
| 1 | 01 |
| 2 | 001 |
| 3 | 0001 |
| 4 | 00001 |
| 5 | 000001 |
| 6 | 0000001 |
| ... | |
| openloop-1 | openloop-1 0's |

vg_run_length : 이 부호없는 정수는 현재의 꼭지점 런의 길이를 나타낸다. 그것의 단일의 표현(표 14 참조)은 산술 부호화된다.

[표 14]

| vg_run_length | 단일의 표현 |
|---|---|
| 1 | 1 |
| 2 | 01 |
| 3 | 001 |
| 4 | 0001 |
| 5 | 00001 |
| 6 | 000001 |
| 7 | 0000001 |
| 8 | 00000001 |
| n | n-1 0들 다음에 1이 따른다 |

vg_leaf : 이 불 플래그는 현재의 런의 마지막 꼭지점이 립(leaf) 꼭지점인지 여부를 나타낸다. 만약 그것이 립 꼭지점이 아니라면, 그것은 브랜칭(분기) 꼭지점이다. 이 필드는 산술 부호화된다.

vg_loop : 이 불 플래그는 현재의 런의 립이 루프를 나타내며 그 그래프의 브랜칭 꼭지점에 연결되었는지 여부를 나타낸다. 이 필드는 산술 부호화된다.

▶ 삼각형_tree

tt_run_length : 이 부호없는 정수는 현재의 삼각형 런의 길이를 나타낸다. 그것의 단일의 표현(표 15 참조)은 산술 부호화된다.

[표 15]

| vg_run_length | 단일의 표현 |
|---|---|
| 1 | 1 |
| 2 | 01 |
| 3 | 001 |
| 4 | 0001 |
| 5 | 00001 |
| 6 | 000001 |
| 7 | 0000001 |
| 8 | 00000001 |
| n | n-1 0들 다음에 1이 따른다 |

tt_leaf : 이 불 플래그는 현재의 런의 마지막 삼각형이 립(leaf) 삼각형인지 여부를 나타낸다. 만약 그것이 립 삼각형이 아니라면, 그것은 브랜칭 삼각형이다. 이 필드는 산술 부호화된다.

triangulated : 이 불 값은 현재의 성분이 삼각형들만 포함하는지 여부를 나타낸다. 이 필드는 산술 부호화된다.

marching_triangle : 이 불 값은 삼각형 트리에서의 삼각형의 위치에 의해 결정된다. 삼각형이 립 또는 브랜칭이면 marching_triangle="0이고," 그렇지 않으면 marching_triangle="1이다.

marching_pattern : 이 불 플래그는 삼각형 런 내부의 에지(edge)들의 마칭(marching) 패턴을 나타낸다. 0은 왼쪽으로부터의 마치(march)를 나타내고, 1은 오른쪽으로부터의 마치(march)를 나타낸다. 이 필드는 산술 부호화된다.

polygon_edge : 이 불 플래그는 현재 삼각형의 베이스가 3차원 메쉬 객체의 재생할 때 유지되어야 하는 에지인지 여부를 나타낸다. 만약 현재 삼각형의 베이스가 유지되지 않는다면, 그것은 버려진다. 이 필드는 산술 부호화된다.

▶ 삼각형

coord_bit : 이 불 값은 기하정보 비트의 값을 나타낸다. 이 필드는 산술 부호화된다.

coord_heading_bit : 이 불 값은 헤딩 기하정보 비트의 값을 나타낸다. 이 필드는 산술 부호화된다.

coord_sign_bit : 이 불 값은 기하정보 샘플의 부호를 나타낸다. 이 필드는 산술부호화된다.

coord_trailing_bit : 이 불 값은 트레일링 기하정보 비트의 값을 나타낸다. 이 필드는 산술부호화된다.

normal_bit : 이 불 값은 노멀(normal) 비트의 값을 나타낸다. 이 필드는 산술부호화된다.

normal_heading_bit : 이 불 값은 헤딩 노멀 비트의 값을 나타낸다. 이 필드는 산술부호화된다.

normal_sign_bit : 이 불 값은 노멀 샘플의 부호를 나타낸다. 이 필드는 산술부호화된다.

normal_trailing_bit : 이 불 값은 트레일링 노멀 비트의 값을 나타낸다. 이 필드는 산술부호화된다.

color_bit : 이 불 값은 색상 비트의 값을 나타낸다. 이 필드는 산술부호화된다.

color_heading_bit : 이 불 값은 헤딩 색상 비트의 값을 나타낸다. 이 필드는 산술부호화된다.

color_sign_bit : 이 불 값은 색상 샘플의 부호를 나타낸다. 이 필드는 산술부호화된다.

color_trailing_bit : 이 불 값은 트레일링 색상 비트의 값을 나타낸다. 이 필드는 산술부호화된다.

texCoord_bit : 이 불 값은 텍스쳐 비트의 값을 나타낸다. 이 필드는 산술부호화된다.

texCoord_heading_bit : 이 불 값은 헤딩 텍스쳐 비트의 값을 나타낸다. 이 필드는 산술부호화된다.

texCoord_sign_bit : 이 불 값은 텍스쳐 샘플의 부호를 나타낸다. 이 필드는 산술부호화된다.

texCoord_trailing_bit : 이 불 값은 트레일링 텍스쳐 비트의 값을 나타낸다. 이 필드는 산술부호화된다.

▶ 3D_Mesh_Object_Forest_Split

MOFS_start_code : 이것은 동기 목적으로 사용되는 고유한 16비트 코드이다. 이 코드의 값은 항상 '0000 0000 001 1 0010'이다.

mofs_id : 이 8비트 부호없는 정수는 포리스트(forest) 분할 성분에 대한 유일한 식별자이다.

pre_smoothing : 이 불 값은 현재의 포리스트 분할 동작이 총체적으로 꼭지점 위치들을 예측하기 위한 사전 평활화 단계(pre - smoothing step)을 사용하는지 여부를 나타낸다.

pre_smoothing_n : 이 정수 값은 사전 평활화 필터(pre - smoothing filter)의 반복횟수를 나타낸다.

pre_smoothing_lambda : 이 부동소숫점 값은 사전 평활화 필터(pre - smoothing filter)의 첫 번째 매개변수이다.

pre_smoothing_mu : 이 부동소숫점 값은 사전 평활화 필터(pre - smoothing filter)의 두 번째 매개변수이다.

post_smoothing : 이 불 값은 현재의 포리스트 분할 동작이 양자화 가공물을 제거하기 위한 사후 평활화 단계(post - smoothing step)을 사용하는지 여부를 나타낸다.

post_smoothing_n : 이 정수 값은 사후 평활화 필터의 반복횟수를 나타낸다.

post_smoothing_lambda : 이 부동소숫점 값은 사후 평활화 필터의 첫 번째 매개변수이다.

post_smoothing_mu : 이 부동소숫점 값은 사후 평활화 필터의 두 번째 매개변수이다.

sharp_edges : 이 불 값은 평활화 불연속 에지들(smoothing discontinuity edges)을 표시하는 데이터가 비트스트림에 포함되었는지 여부를 나타낸다. 만약 sharp_edges = ="0이면," 어떤 에지도 평활화 불연속 에지(smoothing discontinuity edge)로서 표시되어 있지 않다. 만약 평활화 불연속 에지들(smoothing discontinuity edges)이 표시되어 있다면, 사전 평활화 필터(pre - smoothing filter)들 및 사후 평활화 필터(post - smoothing filter)들은 이들을 고려한다.

fixed_vertices : 이 불 값은 평활화 과정동안 움직이지 않는 데이터가 비트스트림에 포함되었는지 여부를 나타낸다. 만약 fixed_vertices = ="0이면," 어떤 꼭지점도 움직이는 것이 허용되지 않는다. 만약 고정된 꼭지점들이 표시되어 있다면, 사전 평활화 필터(pre - smoothing filter)들 및 사후 평활화 필터(post - smoothing filter)들은 이들을 고려한다.

edge_mark : 이 불 값은 대응하는 에지가 평활화 불연속 에지(smoothing discontinuity edge)로 표시되어 있는지 여부를 나타낸다.

vertext_mark : 이 불 값은 대응하는 꼭지점이 고정된 꼭지점인지 아닌지 여부를 나타낸다.

tree_edge : 이 불 값은 에지가 지금까지 만들어진 포리스트에 추가되어야 하는지 여부를 나타낸다.

other_update : 이 불 값은 꼭지점 좌표에 대한 갱신과 포리스트의 어떠한 트리에도 부수하지 않는 페이스(face)들에 관련된 특성들이 비트스트림에 따르는지 여부를 나타낸다.

[표 16]

3D_Mesh_Object

| | | |
|---|---|---|
| 3D_Mesh_Object () { | | |
| **3D_MO_start_code** | 16 | uimsbf |
| 3D_Mesh_Object_Header() | | |
| Do { | | |
| 3D_Mesh_Object_Layer() | | |
| } while (nextbits_bytealigned() == **3D_MOL_start_code**) | | |
| } | | |

3D_Mesh_Object_Header

| | | |
|---|---|---|
| 3D_Mesh_Object_Header() { | | |
| **Ccw** | 1 | blsbf |
| **Convex** | 1 | blsbf |
| **Solid** | 1 | blsbf |
| **CreaseAngle** | 6 | uimsbf |
| Coord_header() | | |
| Normal_header() | | |
| Color_header() | | |
| TexCoord_header() | | |
| **cgd_data** | 1 | blsbf |
| if (**cgd_data** == 1) | | |
| cgd_header() | | |
| } | | |

3D_Mesh_Object_Layer

| | | |
|---|---|---|
| 3D_Mesh_Object_Layer () { | | |
| **3D_MOL_start_code** | 16 | uimsbf |
| **Mol_id** | 8 | uimsbf |
| if(**cgd_data**==1) { | | |
| **N_Vertices** | 24 | uimsbf |
| **N_Triangles** | 24 | uimsbf |
| **N_Edges** | 24 | uimsbf |
| } | | |
| if (**mol_id** == '00000000') | | |
| 3D_Mesh_Object_Base_Layer() | | |
| else | | |
| 3D_Mesh_Object_Forest_Split() | | |
| } | | |

3D_Mesh_Object_Base_Layer

[표 17]

| | | |
|---|---|---|
| 3D_Mesh_Object_Base_Layer() | | |
| do { | | |
| **3D_MOBL_start_code** | 16 | uimsbf |
| **mobl_id** | 8 | uimsbf |
| start_qcoder() | | |
| do { | | |
| connected_component() | | |
| **last_component** | | bac |
| } while (!**last_component**) | | |
| | | |
| } while (nextbits_bytealigned() == **3D_MOBL_start_code**) | | |
| } | | |

coord_header

| | | |
|---|---|---|
| coord_header() { | | |
| **coord_binding** | 2 | uimsbf |
| **coord_bbox** | 1 | blsbf |
| if (**coord_bbox**) { | | |
| **coord_xmin** | 32 | ieeefp |
| **coord_ymin** | 32 | ieeefp |
| **coord_zmin** | 32 | ieeefp |
| **coord_size** | 32 | ieeefp |
| } | | |
| **coord_quant** | 5 | uimsbf |
| **coord_pred_type** | 2 | uimsbf |
| if (**coord_pred_type**=='10') { | | |
| **coord_nlambda** | 2 | uimsbf |
| for (i=1; i<**coord_nlambda**; i++) | | |
| **coord_lambda** | **coord_quant**+3 | |
| } | | |
| } | | |

normal_header

| | | |
|---|---|---|
| normal_header() { | | |
| **normal_binding** | 2 | uimsbf |
| if (**normal_binding** != '00') { | | |
| **normal_bbox** | 1 | blsbf |
| **normal_quant** | 5 | uimsbf |
| **normal_pred_type** | 2 | uimsbf |
| if (**normal_pred_type**=='10') { | | |
| **normal_nlambda** | 2 | uimsbf |
| for (i=1; i<**normal_nlambda**; i++) | | |
| **normal_lambda** | **normal_quant**+3 | |
| } | | |
| } | | |
| } | | |

color_header

[표 18]

| color_header() { | | |
|---|---|---|
| **color_binding** | 2 | uimsbf |
| if (**color_binding** != '00') { | | |
| **color_bbox** | 1 | blsbf |
| if (**color_bbox**) { | | |
| **color_rmin** | 32 | ieeefp |
| **color_gmin** | 32 | ieeefp |
| **color_bmin** | 32 | ieeefp |
| **color_size** | 32 | ieeefp |
| } | | |
| **color_quant** | 5 | uimsbf |
| **color_pred_type** | 2 | uimsbf |
| if (**color_pred_type**=='10') { | | |
| **color_nlambda** | 2 | uimsbf |
| for (i=1; i<**color_nlambda**; i++) | | |
| **color_lambda** | **color_quant**+3 | |
| } | | |
| } | | |
| } | | |

texCoord_header

| texCoord_header() { | | |
|---|---|---|
| **texCoord_binding** | 2 | uimsbf |
| if (**texCoord_binding** != '00') { | | |
| **texCoord_bbox** | 1 | blsbf |
| if (**texCoord_bbox**) { | | |
| **texCoord_umin** | 32 | ieeefp |
| **texCoord_vmin** | 32 | ieeefp |
| **texCoord_size** | 32 | ieeefp |
| } | | |
| **texCoord_quant** | 5 | uimsbf |
| **texCoord_pred_type** | 2 | uimsbf |
| if (**texCoord_pred_type**=='10') { | | |
| **texCoord_nlambda** | 2 | uimsbf |
| for (i=1; i<**texCoord_nlambda**; i++) | | |
| **texCoord_lambda** | **texCoord_quant**+3 | |
| } | | |
| } | | |
| } | | |

cgd_header

[표 19]

| | | |
|---|---|---|
| cgd_header() { | | |
| **N_Proj_Surface_Spheres** | 4 | uimsbf |
| if (**N_Proj_Surface_Spheres** <> 0) { | | |
| **x_coord_Center_Point** | 32 | ieeefl |
| **y_coord_Center_Point** | 32 | ieeefl |
| **z_coord_Center_Point** | 32 | ieeefl |
| **Normalized_Screen_Distance_Factor** | 8 | uimsbf |
| for (i=1;i<= **N_Proj_Surface_Spheres**;i++) { | | |
| **Radius** | 32 | ieeefl |
| **Min_Proj_Surface** | 32 | ieeefl |
| **N_Proj_Points** | 8 | uimsbf |
| for (j=1;j<=N_Proj_Points;j++) { | | |
| **Sphere_Point_Coord** | 11 | uimsbf |
| **Proj_Surface** | 32 | ieeefl |
| } | | |
| } | | |
| } | | |
| } | | |

connected_component

| | | |
|---|---|---|
| connected_component() { | | |
| vertex_graph() | | |
| triangle_tree() | | |
| triangle_data() | | |
| } | | |

vertex_graph

[표 20]

| | | |
|---|---|---|
| vertex_graph() { | | |
| **vg_simple** | 0-16 | bac |
| depth = 0 | | |
| skip_last = 0 | | |
| openloops = 0 | | |
| do { | | |
| do { | | |
| if (!skip_last) { | | |
| **vg_last** | 0-16 | bac |
| if (openloops > 0) { | | |
| **vg_forward_run** | 0- | bac |
| if (!**vg_forward_run**) { | | . |
| openloops-- | | |
| if (openloops > 0) | | |
| **vg_loop_index** | 0- | uac |
| break | | |
| } | | |
| } | | |
| } | | |
| **vg_run_length** | 0- | uac |
| **vg_leaf** | 0-16 | bac |
| if (**vg_leaf** && !**vg_simple**) { | | |
| **vg_loop** | 0-16 | bac |
| if (**vg_loop**) | | |
| openloops++ | | |
| } | | |
| } while (0) | | |
| if (**vg_leaf** == (**vg_last** & !skip_last)) | | |
| if (**vg_last** & !skip_last) | | |
| depth-- | | |
| else | | |
| depth++ | | |
| skip_last = !**vg_leaf** | | |
| } while (depth >= 0) | | |
| } | | |

triangle_tree

| | | |
|---|---|---|
| triangle_tree() { | | |
| depth = 0 | | |
| ntriangles = 0 | | |
| do { | | |
| **tt_run_length** | 0-16 | bac |
| ntriangles += **tt_run_length** | | |
| **tt_leaf** | 0-16 | bac |
| if (**tt_leaf**) | | |
| depth-- | | |
| else | | |
| depth++ | | |
| } while (depth >= 0) | | |
| } | | |

[표 21]

triangle_data

| | | |
|---|---|---|
| triangle_data() { | | |
| **triangulated** | 0-16 | bac |
| root_triangle() | | |
| for (i=1; i<ntriangles; i++) | | |
| triangle() | | |
| } | | |

root_triangle

| | | |
|---|---|---|
| root_triangle() { | | |
| if (**marching_triangle**) | | |
| **marching_pattern** | 0-16 | bac |
| root_coord() | | |
| root_normal() | | |
| root_color() | | |
| root_texCoord() | | |
| } | | |

| | | |
|---|---|---|
| root_coord() { | | |
| root_coord_sample() | | |
| coord_sample() | | |
| coord_sample() | | |
| } | | |

| | | |
|---|---|---|
| root_normal() { | | |
| if (**normal_binding** != `00') { | | |
| root_normal_sample() | | |
| if (**normal_binding** != `10') { | | |
| normal_sample() | | |
| normal_sample() | | |
| } | | |
| } | | |
| } | | |

| | | |
|---|---|---|
| root_color() { | | |
| if (**color_binding** != `00') { | | |
| root_color_sample() | | |
| if (**color_binding** != `10') { | | |
| color_sample() | | |
| color_sample() | | |
| } | | |
| } | | |
| } | | |

[표 22]

| | | |
|---|---|---|
| root_texCoord() { | | |
| if (**texCoord_binding** != '00') { | | |
| root_texCoord_sample() | | |
| texCoord_sample() | | |
| texCoord_sample() | | |
| } | | |
| } | | |

triangle

| | | |
|---|---|---|
| triangle() { | | |
| if (**marching_triangle**) | | |
| **marching_pattern** | 0-16 | bac |
| if (!**triangulated**) | | |
| **polygon_edge** | 0-16 | bac |
| coord() | | |
| normal() | | |
| color() | | |
| texCoord() | | |
| } | | |

| | | |
|---|---|---|
| coord() { | | |
| if (!visited) | | |
| coord_sample() | | |
| } | | |

| | | |
|---|---|---|
| normal() { | | |
| if (**normal_binding** == '01') { | | |
| if (!visited) | | |
| normal_sample() | | |
| } | | |
| else if (**normal_binding** == '10') { | | |
| if (**triangulated** \|\| **polygon_edge**) | | |
| normal_sample() | | |
| } | | |
| else if (**normal_binding** == '11') { | | |
| if (**triangulated** \|\| **polygon_edge**) { | | |
| normal_sample() | | |
| normal_sample() | | |
| } | | |
| normal_sample() | | |
| } | | |
| } | | |

[표 23]

| color() { | | |
|---|---|---|
| if (**color_binding** == '01') { | | |
| if (!visited) | | |
| color_sample() | | |
| } | | |
| else if (**color_binding** == '10') { | | |
| if (**triangulated** \|\| **polygon_edge**) | | |
| color_sample() | | |
| } | | |
| else if (**color_binding** == '11') { | | |
| if (**triangulated** \|\| **polygon_edge**) { | | |
| color_sample() | | |
| color_sample() | | |
| } | | |
| color_sample() | | |
| } | | |
| } | | |

| texCoord() { | | |
|---|---|---|
| if (**texCoord_binding** == '01') { | | |
| if (!visited) | | |
| texCoord_sample() | | |
| } | | |
| else if (**texCoord_binding** == '11') { | | |
| if (**triangulated** \|\| **polygon_edge**) { | | |
| texCoord_sample() | | |
| texCoord_sample() | | |
| } | | |
| texCoord_sample() | | |
| } | | |
| } | | |

| coord_root_sample() { | | |
|---|---|---|
| for (i=0; i<3; i++) | | |
| for (j=0; j<**coord_quant**; j++) | | |
| **coord_bit** | 0-1 | bac |
| } | | |

| normal_root_sample() { | | |
|---|---|---|
| for (i=0; i<1; i++) | | |
| for (j=0; j<**normal_quant**; j++) | | |
| **normal_bit** | 0-1 | bac |
| } | | |

| color_root_sample() { | | |
|---|---|---|
| for (i=0; i<3; i++) | | |
| for (j=0; j<**color_quant**; j++) | | |
| **color_bit** | 0-1 | bac |
| } | | |

[표 24]

| | | |
|---|---|---|
| texCoord_root_sample() { | | |
| for (i=0; i<2; i++) | | |
| for (j=0; j<**texCoord_quant**; j++) | | |
| **texCoord_bit** | 0-1 | bac |
| } | | |

| | | |
|---|---|---|
| coord_sample() { | | |
| for (i=0; i<3; i++) { | | |
| j=0 | | |
| do { | | |
| **coord_leading_bit** | 0-16 | bac |
| j++ | | |
| } while (j<**coord_quant** && !**coord_leading_bit**) | | |
| if (**coord_leading_bit**) { | | |
| **coord_sign_bit** | 0-1 | bac |
| do { | | |
| **coord_trailing_bit** | | |
| } while (j<**coord_quant**) | | |
| } | | |
| } | | |
| } | | |

| | | |
|---|---|---|
| normal_sample() { | | |
| for (i=0; i<1; i++) { | | |
| j=0 | | |
| do { | | |
| **normal_leading_bit** | 0-16 | bac |
| j++ | | |
| } while (j<**normal_quant** && !**normal_leading_bit**) | | |
| if (**normal_leading_bit**) { | | |
| **normal_sign_bit** | 0-1 | bac |
| do { | | |
| **normal_trailing_bit** | | |
| } while (j<**normal_quant**) | | |
| } | | |
| } | | |
| } | | |

[표 25]

| | | |
|---|---|---|
| color_sample() { | | |
| for (i=0; i<3; i++) { | | |
| j=0 | | |
| do { | | |
| **color_leading_bit** | 0-16 | bac |
| j++ | | |
| } while (j<**color_quant** && !**color_leading_bit**) | | |
| if (**color_leading_bit**) { | | |
| **color_sign_bit** | 0-1 | bac |
| do { | | |
| **color_trailing_bit** | | |
| } while (j<**color_quant**) | | |
| } | | |
| } | | |
| } | | |

| | | |
|---|---|---|
| texCoord_sample() { | | |
| for (i=0; i<2; i++) { | | |
| j=0 | | |
| do { | | |
| **texCoord_leading_bit** | 0-16 | bac |
| j++ | | |
| } while (j<**texCoord_quant** && !**texCoord_leading_bit**) | | |
| if (**texCoord_leading_bit**) { | | |
| **texCoord_sign_bit** | 0-1 | bac |
| do { | | |
| **texCoord_trailing_bit** | | |
| } while (j<**texCoord_quant**) | | |
| } | | |
| } | | |
| } | | |

3D_Mesh_Object_Forest_Split

[표 26]

| 3D_Mesh_Object_Forest_Split () { | | |
|---|---|---|
| do { | | |
| **3D_MOFS_start_code** | 16 | uimsbf |
| **mofs_id** | 8 | uimsbf |
| **pre_smoothing** | 1 | blsbf |
| if(**pre_smoothing**) | | |
| pre_smoothing_parameters() | | |
| **post_smoothing** | 1 | blsbf |
| if(**post_smoothing**) | | |
| post_smoothing_parameters() | | |
| start_qcoder() | | |
| **sharp_edges** | 1 | blsbf |
| if(**sharp_edges**) | | |
| edge_marks() | | |
| **fixed_vertices** | 1 | blsbf |
| if(**fixed_vertices**) | | |
| vertex_marks() | | |
| for each connected component { | | |
| fs_pre_update() | | |
| fs_post_update() | | |
| } | | |
| | | |
| } while (nextbits_bytealigned() == **3D_MOFS_start_code**) | | |
| } | | |

| pre_smoothing_parameters() { | | |
|---|---|---|
| **pre_smoothing_n** | 8 | uimsbf |
| **pre_smoothing_lambda** | 32 | ieeefl |
| **pre_smoothing_mu** | 32 | ieeefl |
| } | | |

| post_smoothing_parameters() { | | |
|---|---|---|
| **post_smoothing_n** | 8 | uimsbf |
| **post_smoothing_lambda** | 32 | ieeefl |
| **post_smoothing_mu** | 32 | ieeefl |
| } | | |

| edge_marks () { | | |
|---|---|---|
| for each edge | | |
| **edge_mark** | 1 | bac |
| } | | |

| vertex_marks () { | | |
|---|---|---|
| for each vertex | | |
| **vertex_mark** | 1 | bac |
| } | | |

[표 27]

| | | |
|---|---|---|
| fs_pre_update() { | | |
| forest() | | |
| for each tree in forest { | | |
| triangle_tree() | | |
| for each vertex in tree loop | | |
| visited = 1 | | |
| triangle_data() | | |
| } | | |
| } | | |

| | | |
|---|---|---|
| forest () { | | |
| for each edge | | |
| if (creates no loop in forest) | | |
| **tree_edge** | 1 | bac |
| } | | |

| | | |
|---|---|---|
| fs_post_update() { | | |
| for each tree in forest { | | |
| for each vertex in tree loop | | |
| visited = 0 | | |
| tree_loop_property_update() | | |
| } | | |
| **other_update** | 1 | blsbf |
| if(**other_update**) | | |
| other_property_update() | | |
| } | | |

| | | |
|---|---|---|
| tree_loop_property_update () { | | |
| for each triangle incident to tree { | | |
| coord_update() | | |
| normal_update() | | |
| color_update() | | |
| texCoord_update() | | |
| } | | |
| }. | | |

| | | |
|---|---|---|
| other_property_update() { | | |
| for each triangle not incident to any tree in forest { | | |
| coord_update() | | |
| normal_update() | | |
| color_update() | | |
| texCoord_update() | | |
| } | | |
| } | | |

| | | |
|---|---|---|
| coord_update () { | | |
| if (!visited) | | |
| coord_sample() | | |
| } | | |

[표 28]

```
normal_update () {
        if (normal_binding == '01') {
                if (!visited)
                        normal_sample()
        }
        else if (normal_binding == '10') {
                normal_sample()
        }
        else if (normal_binding == '11') {
                if (1st corner adjacent to tree)
                        normal_sample()
                if (2nd corner adjacent to tree)
                        normal_sample()
                if (3rd corner adjacent to tree)
                        normal_sample()
        }
}
```

```
color_update () {
        if (color_binding == '01') {
                if (!visited)
                        color_sample()
        }
        else if (color_binding == '10') {
                color_sample()
        }
        else if (color_binding == '11') {
                if (1st corner adjacent to tree)
                        color_sample()
                if (2nd corner adjacent to tree)
                        color_sample()
                if (3rd corner adjacent to tree)
                        color_sample()
        }
}
```

```
texCoord_update () {
        if (texCoord_binding == '01') {
                if (!visited)
                        texCoord_sample()
        }
        else if (texCoord_binding == '11') {
                if (1st corner adjacent to tree)
                        texCoord_sample()
                if (2nd corner adjacent to tree)
                        texCoord_sample()
                if (3rd corner adjacent to tree)
                        texCoord_sample()
        }
}
```

## 발명의 효과

이상에서 설명한 바와 같이, 본 발명에 의한 점진적인 삼차원 메쉬 정보의 부호화/복호화 방법 및 장치는, 부분 메쉬(MCOM)로 처리되는 데이터의 구조를 가짐으로써 압축된 비트 스트림 상에서도 부분별 구분이 가능하고, 이에 따라 복호화부는 부분별로 복호화가 끝나는 대로 각 부분들을 용이하게 재현할 수 있다. 이것은 종래의 삼차원 메쉬 정보의 부호화 방식과 비교하여 점진적인 영상 재현과 전송에러에 대한 대응능력을 보강할 수 있다. 또한, 모델을 우선 단계별 메쉬(MOL) 또는 부분 메쉬(MCOM)로 구분하면, 이들을 독립적으로 부호화 및 복호화할 수 있으며, 이러한 구조는 구현하기도 간단하고, 필요에 따라 중복되는 정보들도 간단하게 제거할 수 있도록 구현할 수 있다.

## (57) 청구의 범위

### 청구항 1.

(a) 삼차원 메쉬를 복수개의 부분 메쉬들로 재구성하는 단계;

(b) 상기 복수개의 부분 메쉬들을 각각 부호화하는 단계; 및

(c) 복수개의 부호화된 부분 메쉬들을 압축된 비트 스트림으로 통합하여 전송하는 단계를 구비하는 것을 특징으로 하는 점진적인 삼차원 메쉬 정보의 부호화 방법.

### 청구항 2.

제1항에 있어서, 상기 복수개의 부분 메쉬들은 각각

그 자체의 복원에 필요한 정보로서 적어도 연결정보, 위치정보 및 화상정보를 각각 포함하는 것을 특징으로 하는 점진적인 삼차원 메쉬 정보의 부호화 방법.

### 청구항 3.

제1항에 있어서, 상기 (a) 단계는

(a.1) 삼차원 메쉬로부터 하나 이상의 단계별 메쉬들을 추출하는 소단계; 및

(a.2) 상기 하나 이상의 단계별 메쉬들을 각각 복수개의 부분 메쉬들로 분할하는 소단계를 구비하는 것을 특징으로 하는 점진적인 삼차원 메쉬 정보의 부호화 방법.

### 청구항 4.

제1항에 있어서, 상기 (b) 단계는

상기 복수개의 부분 메쉬들을 각각 부호화하되, 이미 수행된 부분 메쉬의 부호화 과정에서 생성된 정보를 아직 수행되지 않은 부분 메쉬의 부호화 과정에서 재사용하는 것을 특징으로 하는 점진적인 삼차원 메쉬 정보의 부호화 방법.

### 청구항 5.

점진적인 삼차원 메쉬 정보의 부호화 방법에 의해 부호화되어 전송된 비트 스트림을 점진적으로 복호화하는 방법에 있어서,

(a) 상기 전송된 비트 스트림을 복수개의 부호화된 부분 메쉬들로 나누는 단계;

(b) 상기 복수개의 부호화된 부분 메쉬들을 각각 복호화하는 단계; 및

(c) 복수개의 복호화된 부분 메쉬들을 합성하여 삼차원 메쉬를 복원하는 단계를 구비하는 것을 특징으로 하는 점진적인 삼차원 메쉬 정보의 복호화 방법.

청구항 6.

제5항에 있어서, 상기 (a) 단계는

(a.1) 상기 전송된 비트 스트림을 하나 이상의 복호화된 단계별 메쉬들로 분류하는 소단계; 및

(a.2) 상기 하나 이상의 복호화된 단계별 메쉬들을 각각 복수개의 부분 메쉬들로 나누는 소단계를 구비하는 것을 특징으로 하는 점진적인 삼차원 메쉬 정보의 복호화 방법.

청구항 7.

제5항에 있어서, 상기 (b) 단계는

상기 복수개의 부호화된 부분 메쉬들을 각각 복호화하되, 이미 수행된 부분 메쉬의 복호화 과정에서 생성된 정보를 아직 수행되지 않은 부분 메쉬의 복호화 과정에서 재사용하는 것을 특징으로 하는 점진적인 메쉬 정보의 복호화 방법.

청구항 8.

점진적인 삼차원 메쉬 정보의 부호화/복호화 방법에 있어서,

(a) 삼차원 메쉬로부터 서로 독립된 하나 이상의 단계별 메쉬들을 추출하는 단계;

(b) 상기 하나 이상의 단계별 메쉬들을 각각 서로 독립적으로 부호화하고, 서로 독립적으로 전송하는 단계; 및

(c) 서로 독립적으로 부호화하여 전송된 하나 이상의 단계별 메쉬들을 각각 복호화하여 서로 독립된 하나 이상의 단계별 메쉬들을 얻는 단계를 구비하는 것을 특징으로 하는 점진적인 삼차원 메쉬 정보의 부호화/복호화 방법.

청구항 9.

제8항에 있어서, 상기 (c) 단계 후에,

(d) 서로 독립된 단계별 메쉬들을 모으고, 중복되는 정보를 제거하여 원래의 삼차원 메쉬로 복원하는 단계를 더 포함하는 것을 특징으로 하는 점진적인 삼차원 메쉬 정보의 부호화/복호화 방법.

청구항 10.

점진적인 삼차원 메쉬 정보의 부호화/복호화 방법에 있어서,

(a) 삼차원 메쉬로부터 하나 이상의 단계별 메쉬를 추출하고, 서로 독립된 복수개의 부분 메쉬들로 분할하는 단계;

(b) 상기 복수개의 부분 메쉬들을 각각 서로 독립적으로 부호화하고, 서로 독립적으로 전송하는 단계; 및

(c) 서로 독립적으로 부호화하여 전송된 복수개의 부분 메쉬들을 각각 복호화하여 서로 독립된 복수개의 부분 메쉬들을 얻는 단계를 구비하는 것을 특징으로 하는 점진적인 삼차원 메쉬 정보의 부호화/복호화 방법.

청구항 11.

제10항에 있어서, 상기 (c) 단계 후에,

(d) 서로 독립된 부분 메쉬들을 모으고, 인접한 부분 메쉬들간에 중복되는 정보를 제거하여 원래의 삼차원 메쉬로 복원하는 단계를 더 포함하는 것을 특징으로 하는 점진적인 삼차원 메쉬 정보의 부호화/복호화 방법.

청구항 12.

삼차원 메쉬를 입력하고, 복수개의 부분 메쉬들로 재구성하는 삼차원 데이터 분석부;

상기 복수개의 부분 메쉬들을 각각 부호화하는 복수개의 부분별 부호화부들; 및

복수개의 부호화된 부분 메쉬들을 압축된 비트 스트림으로 통합하여 전송하는 멀티플렉서를 구비하는 것을 특징으로 하는 점진적인 삼차원 메쉬 정보의 부호화 장치.

청구항 13.

제12항에 있어서, 상기 삼차원 데이터 분석부는

입력된 삼차원 메쉬로부터 하나 이상의 단계별 메쉬들을 추출하는 삼차원 단계별 메쉬 분석기; 및

상기 하나 이상의 단계별 메쉬들을 각각 복수개의 부분 메쉬들로 분할하는 복수개의 부분 메쉬 분석기들을 구비하는 것을 특징으로 하는 점진적인 삼차원 메쉬 정보의 부호화 장치.

청구항 14.

제12항에 있어서, 상기 복수개의 부분별 부호화부들이 상기 복수개의 부분 메쉬들을 각각 부호화할때, 아직 부호화를 수행하지 않은 부분별 부호화부는 이미 부호화를 수행한 부분별 부호화부에서 생성된 부호화 정보를 사용하여 부호화하는 것을 특징으로 하는 점진적인 삼차원 메쉬 정보의 부호화 장치.

청구항 15.

점진적인 삼차원 메쉬 정보의 부호화 장치에 의해 부호화되어 전송된 비트 스트림을 점진적으로 복호화하는 장치에 있어서,

상기 전송된 비트 스트림을 복수개의 부호화된 부분 메쉬들로 분류하는 디멀티플렉서;

상기 복수개의 부호화된 부분 메쉬들을 각각 복호화하는 복수개의 부분별 복호화부들; 및

복수개의 복호화된 부분 메쉬들을 합성하여 삼차원 메쉬를 복원하는 삼차원 데이터 합성부를 포함하는 것을 특징으로 하는 점진적인 삼차원 메쉬 정보의 복호화 장치.

청구항 16.

제15항에 있어서, 상기 복수개의 부분별 복호화부들이 상기 복수개의 부분 메쉬들을 각각 복호화할때, 아직 복호화를 수행하지 않은 부분별 복호화부는 이미 복호화를 수행한 부분별 복호화부에서 생성된 복호화 정보를 사용하여 복호화하는 것을 특징으로 하는 점진적인 삼차원 메쉬 정보의 복호화 장치.

청구항 17.

삼차원 메쉬를 입력하고, 입력된 삼차원 메쉬로부터 서로 독립된 [복수개의] 하나 이상의 단계별 메쉬들을 추출하는 삼차원 단계별 메쉬 분석부;

상기 하나 이상의 단계별 메쉬들을 각각 서로 독립적으로 부호화하고, 서로 독립적으로 전송하는 하나 이상의 단계별 메쉬 부호화부들; 및

서로 독립적으로 부호화하여 전송된 하나 이상의 단계별 메쉬들을 각각 복호화하여 서로 독립된 하나 이상의 단계별 메쉬들을 얻는 하나 이상의 단계별 메쉬 복호화부들을 구비하는 것을 특징으로 하는 점진적인 삼차원 메쉬 정보의 부호화/복호화 장치.

청구항 18.

제17항에 있어서, 상기 서로 독립된 하나 이상의 단계별 메쉬들을 모으고, 중복되는 정보를 제거하여 원래의 삼차원 메쉬로 복원하는 삼차원 단계별 메쉬 합성부를 더 포함하는 것을 특징으로 하는 독립적이고 점진적인 삼차원 메쉬 정보의 부호화/복호화 장치.

청구항 19.

삼차원 메쉬를 입력하여 하나 이상의 단계별 메쉬들을 추출하고, 서로 독립된 복수개의 부분 메쉬들로 분할하는 삼차원 데이터 분석부;

상기 복수개의 단계별 메쉬들을 각각 서로 독립적으로 부호화하고, 서로 독립적으로 전송하는 복수개의 부분 메쉬 부호화부들; 및

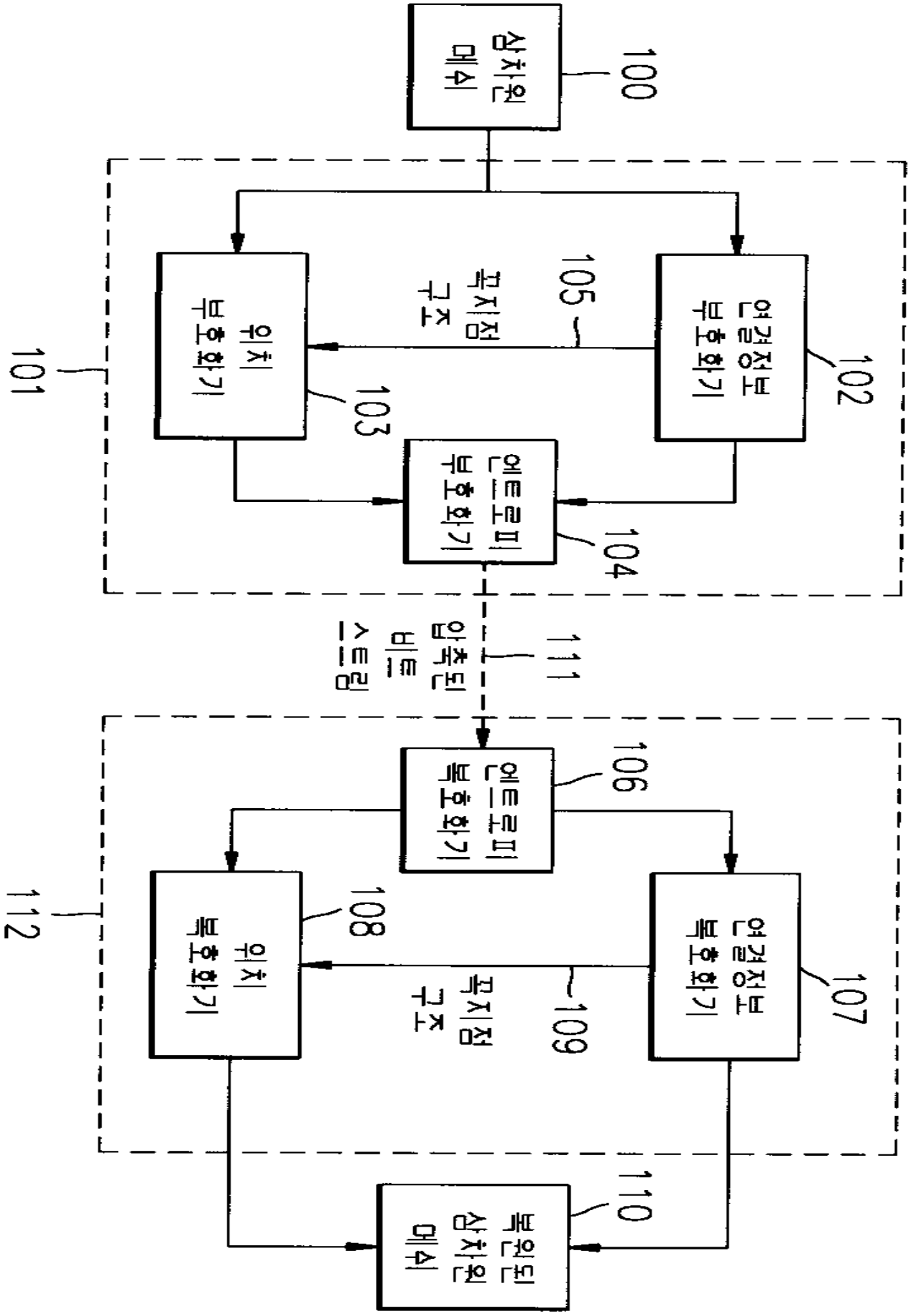
서로 독립적으로 부호화하여 전송된 복수개의 부분 메쉬들을 각각 복호화하여 서로 독립된 복수개의 부분 메쉬들을 얻는 복수개의 부분 메쉬 복호화부들을 구비하는 것을 특징으로 하는 점진적인 삼차원 메쉬 정보의 부호화/복호화 장치.

청구항 20.

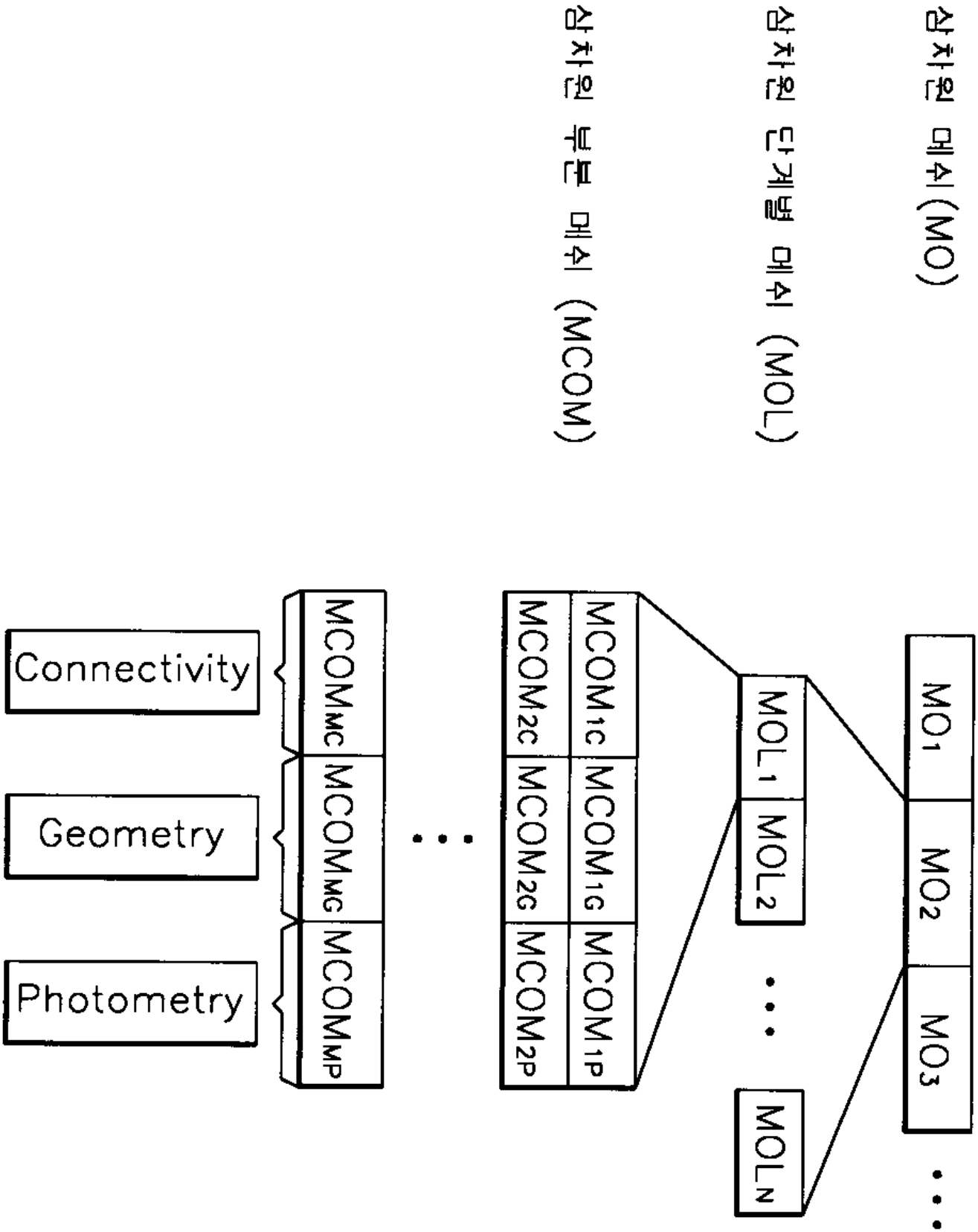
제19항에 있어서, 상기 서로 독립된 복수개의 부분 메쉬들을 모으고, 인접한 부분 메쉬들간에 중복되는 정보를 제거하여 원래의 삼차원 메쉬로 복원하는 삼차원 데이터 합성부를 더 포함하는 것을 특징으로 하는 독립적이고 점진적인 삼차원 메쉬 정보의 부호화/복호화 장치.

도면

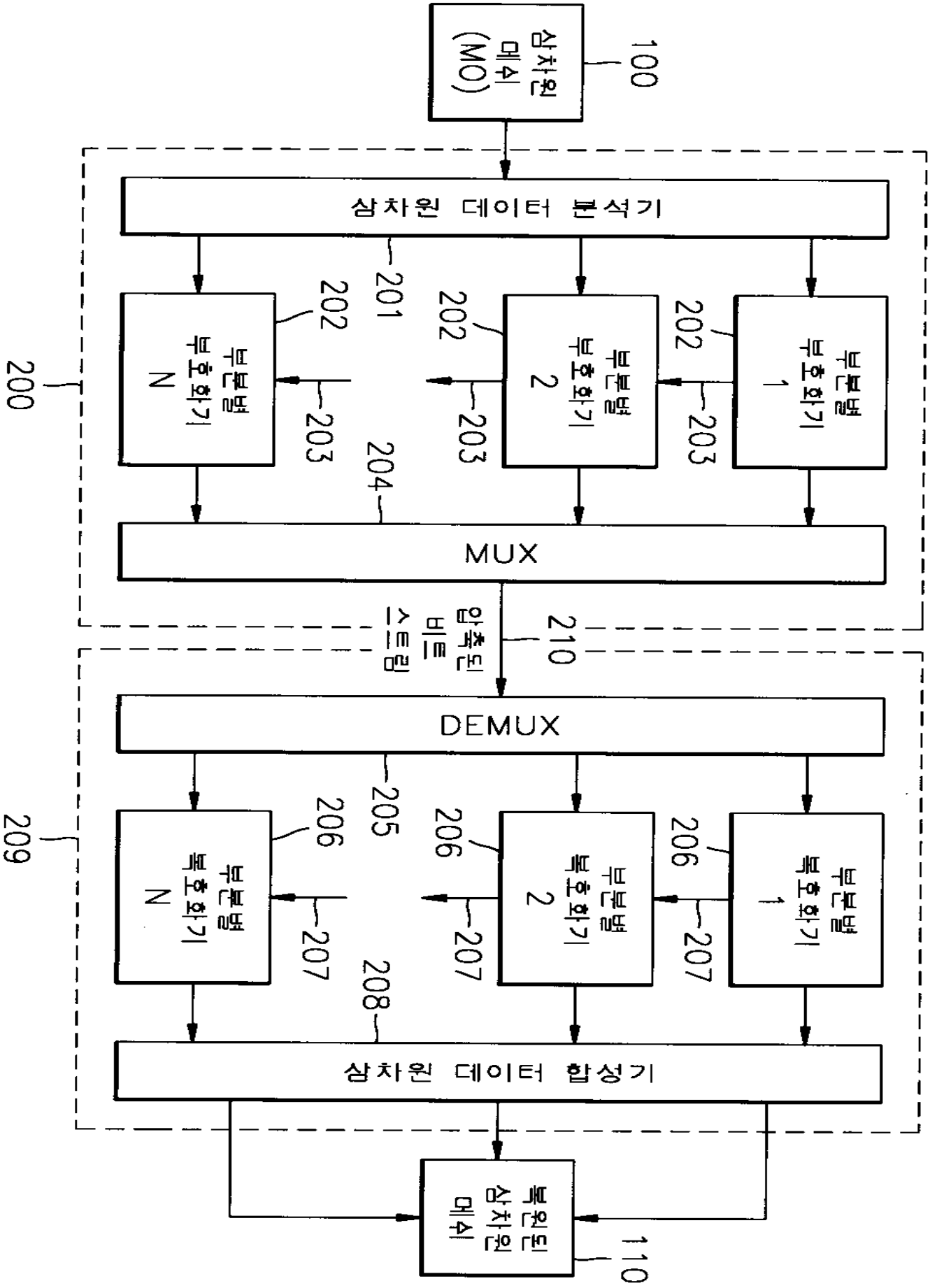
도면 1


삼차원 메쉬
100
101
위치 부호화기
103
연결정보 부호화기
102
꼭지점 구조
105
엣트로피 부호화기
104
압축된 비트 스트림
111
엣트로피 부호화기
106
112
위치 부호화기
108
연결정보 부호화기
107
꼭지점 구조
109
복원된 삼차원 메쉬
110

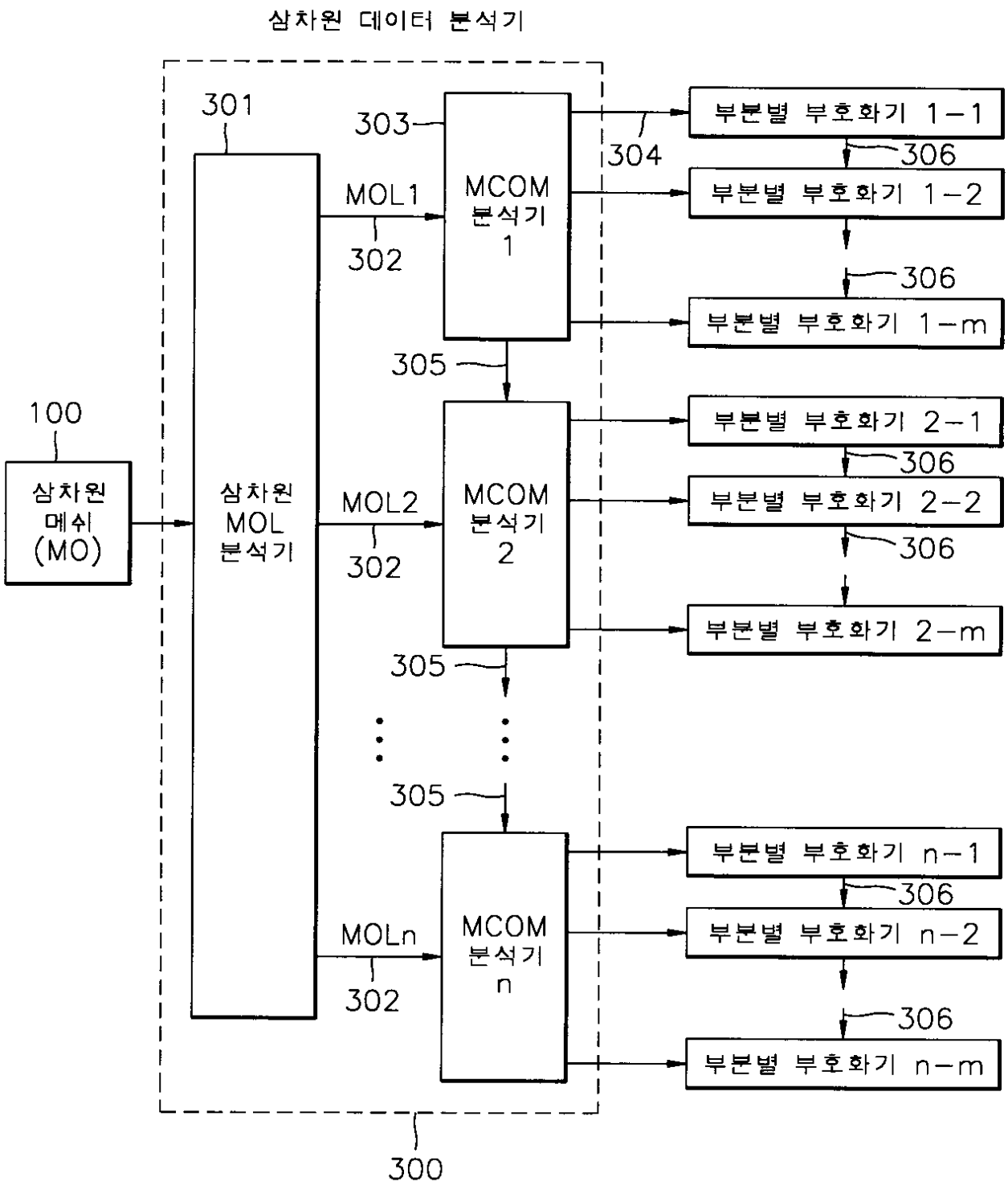
도면 2


삼차원 메쉬(MO)
삼차원 단계별 메쉬 (MOL)
삼차원 부분 메쉬 (MCOM)
MO1
MO2
MO3
MOLN
MOL1
MOL2
MCOM1C
MCOM1G
MCOM1P
MCOM2C
MCOM2G
MCOM2P
MCOMMC
MCOMMG
MCOMMP
Connectivity
Geometry
Photometry

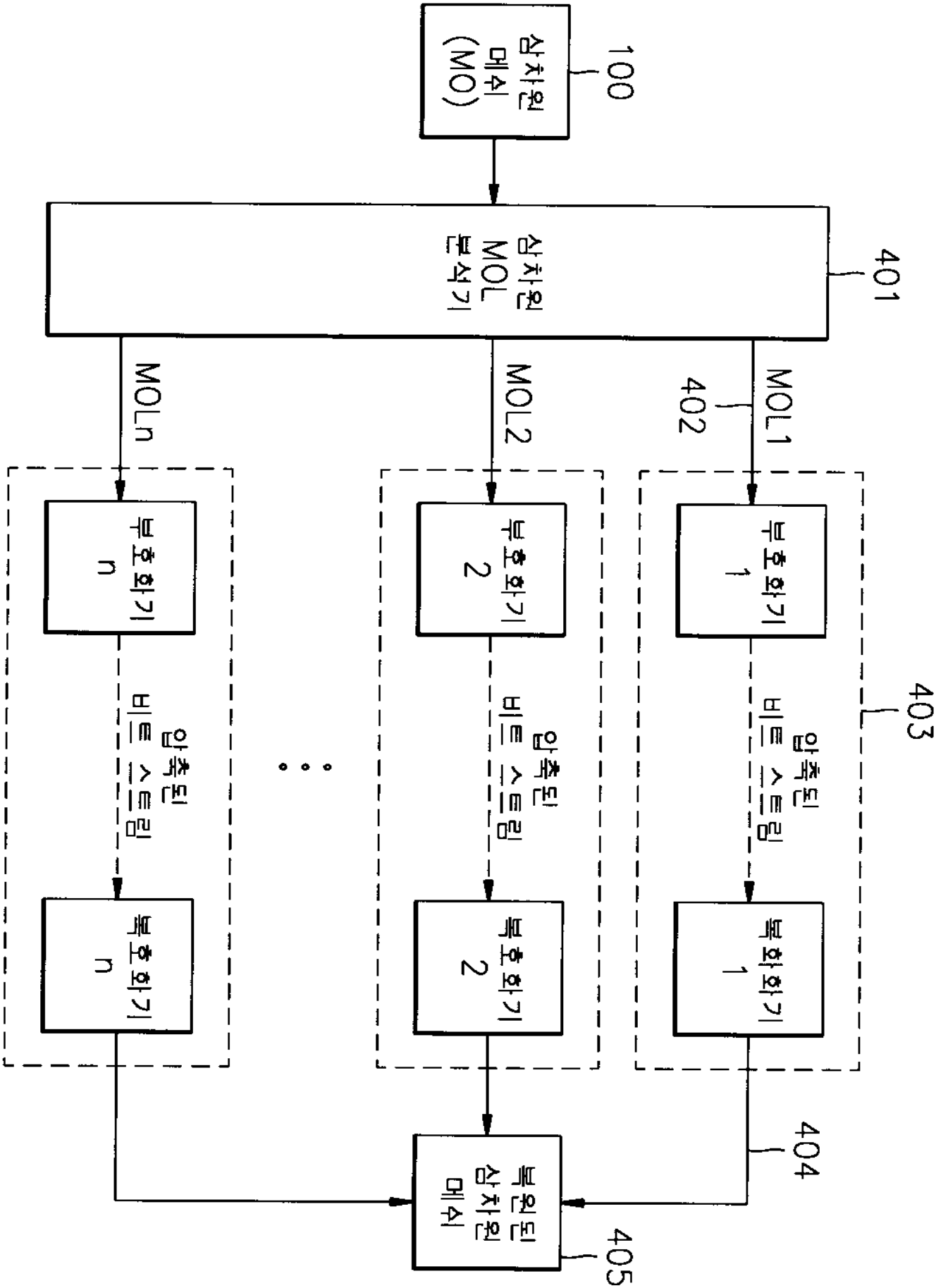
도면 3


삼차원 메쉬 (MO)
100
삼차원 데이터 분석기
200
201
202
203
204
양자화된 버텍스 스트림
210
MUX
DEMUX
209
205
206
207
208
삼차원 데이터 합성기
복원된 삼차원 메쉬
110

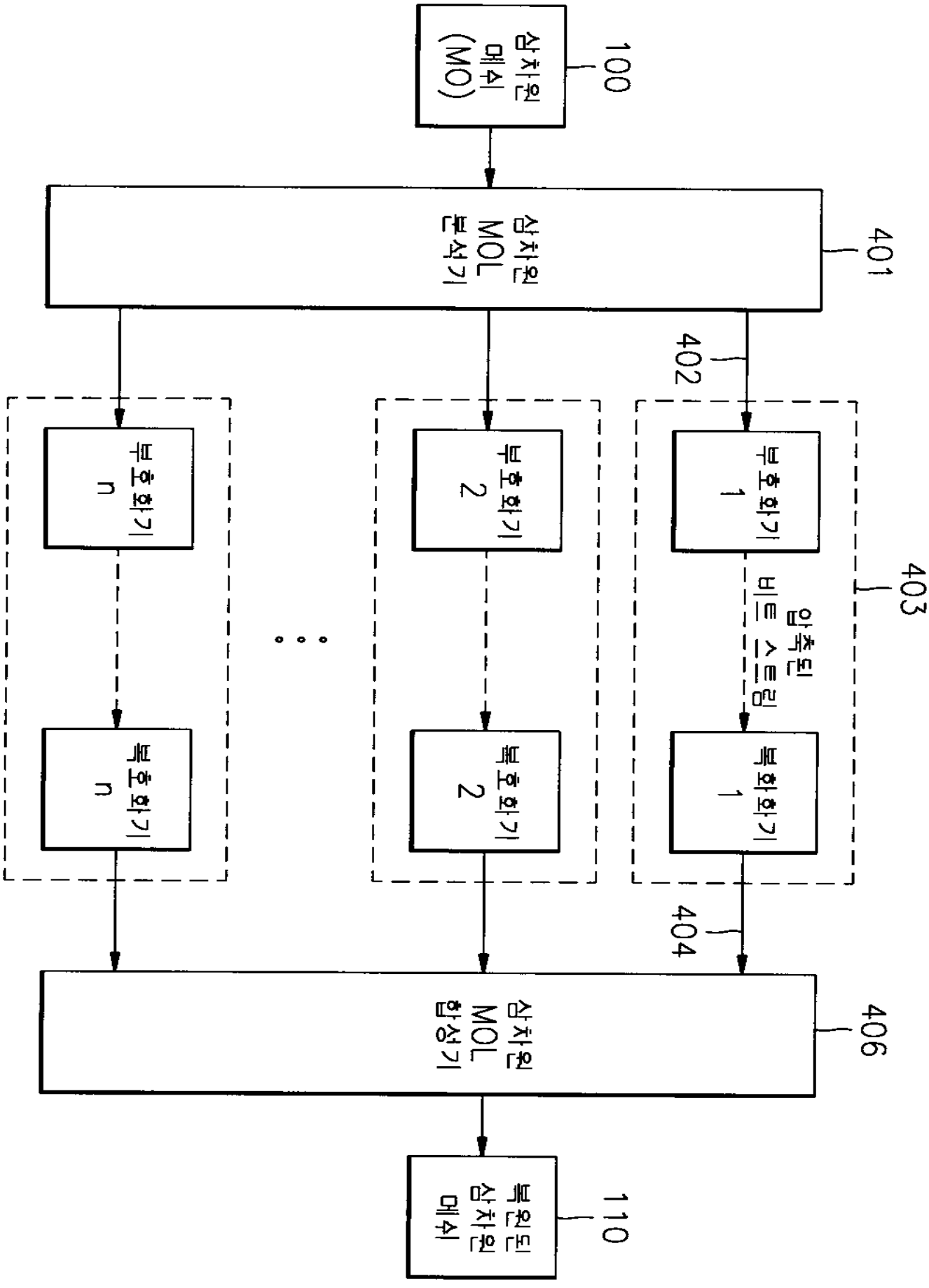
도면 4


삼차원 데이터 분석기
100
삼차원 메쉬 (MO)
301
삼차원 MOL 분석기
303
MOL1
302
MCOM 분석기 1
304
305
MOL2
302
MCOM 분석기 2
305
MOLn
302
MCOM 분석기 n
305
306
부분별 부호화기 1-1
부분별 부호화기 1-2
부분별 부호화기 1-m
부분별 부호화기 2-1
부분별 부호화기 2-2
부분별 부호화기 2-m
부분별 부호화기 n-1
부분별 부호화기 n-2
부분별 부호화기 n-m
300

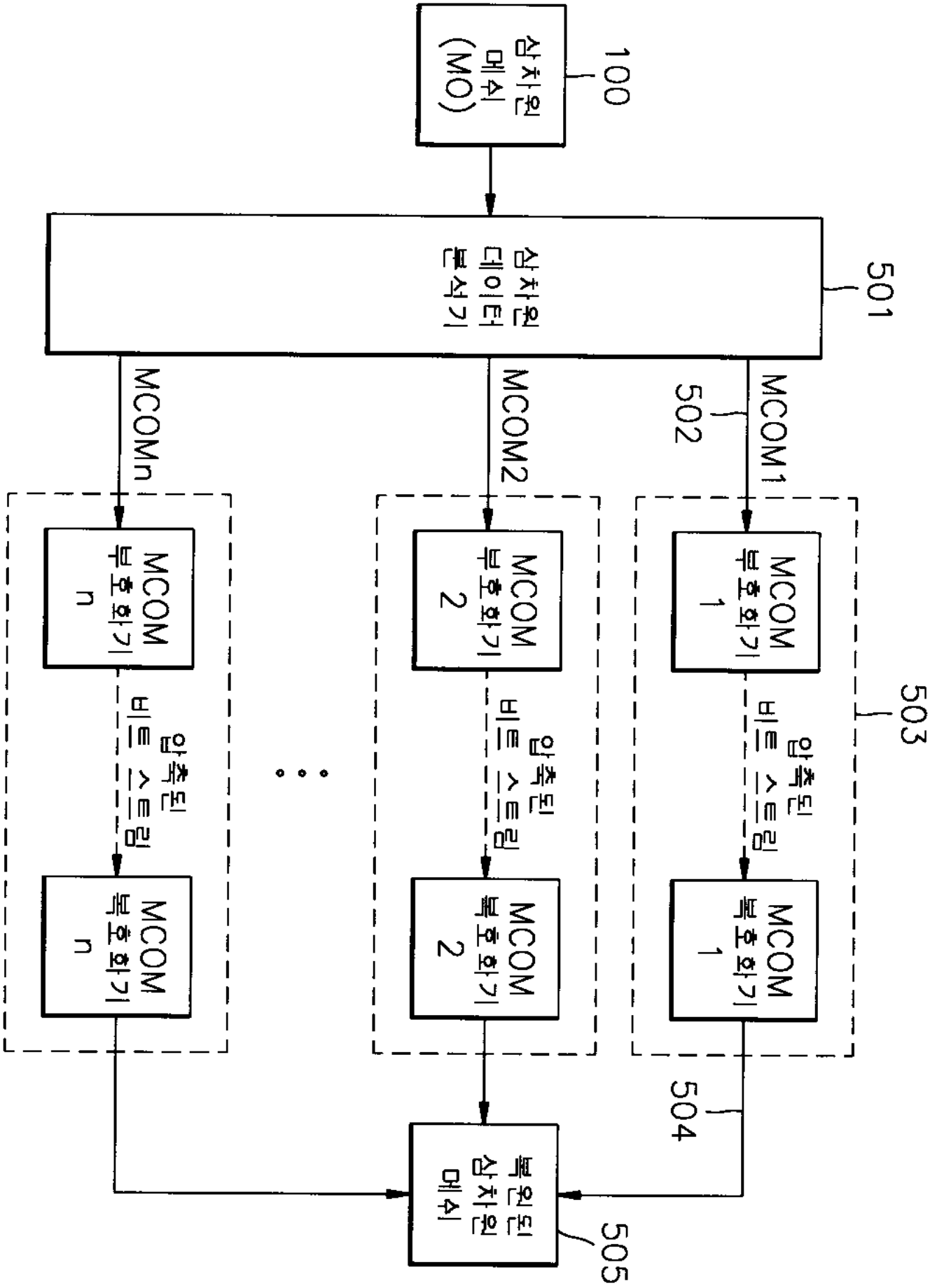
도면 5


삼차원 메쉬 (MO)
100
삼차원 MOL 분석기
401
MOL1
MOL2
MOLn
402
403
부호화기 1
부호화기 2
부호화기 n
압축된 비트 스트림
복호화기 1
복호화기 2
복호화기 n
404
복원된 삼차원 메쉬
405

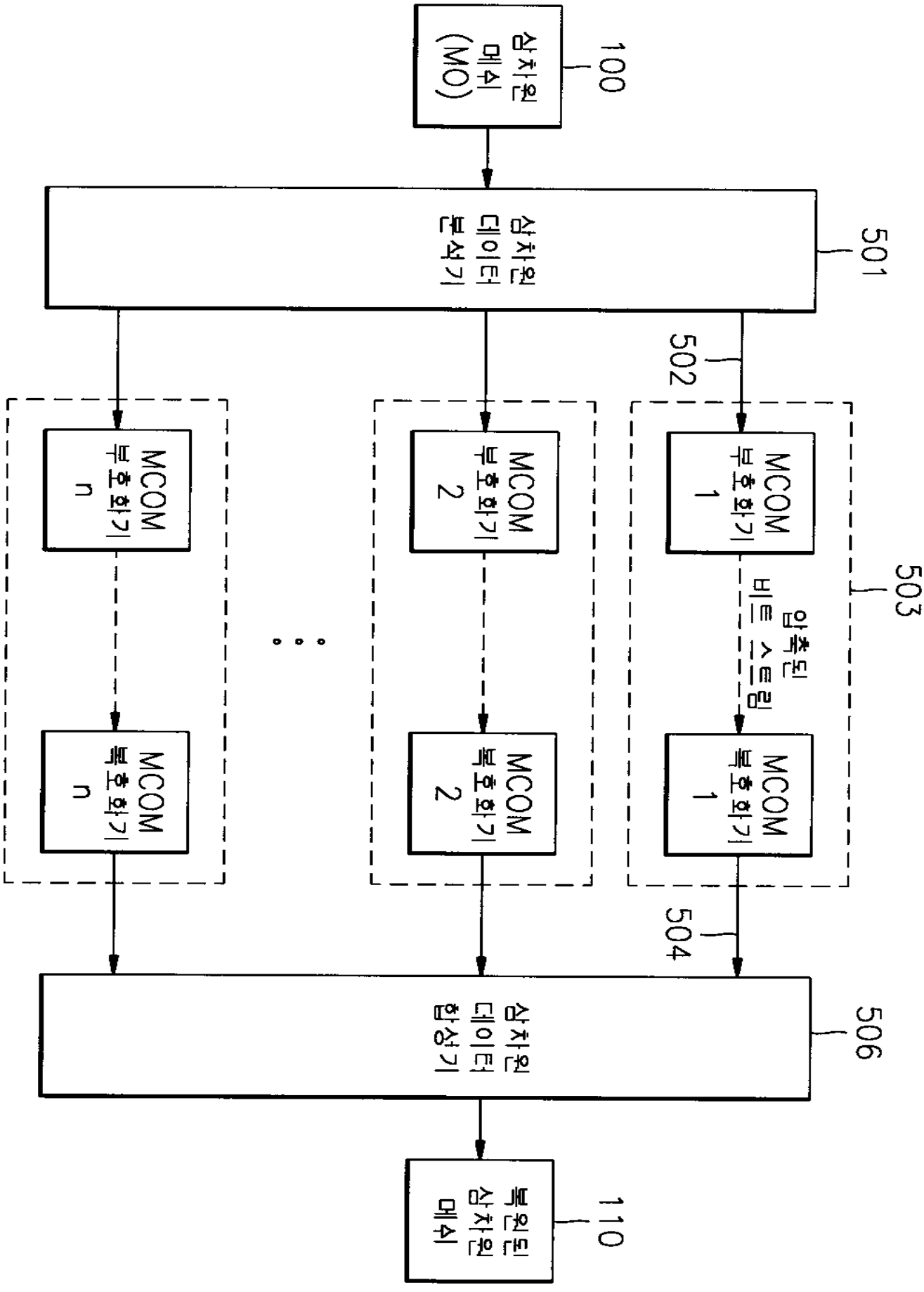
도면 6


삼차원 메쉬 (MO)
100
삼차원 MOL 분석기
401
402
403
부호화기 1
부호화기 2
부호화기 n
압축된 비트 스트림
복호화기 1
복호화기 2
복호화기 n
404
삼차원 MOL 합성기
406
복원된 삼차원 메쉬
110

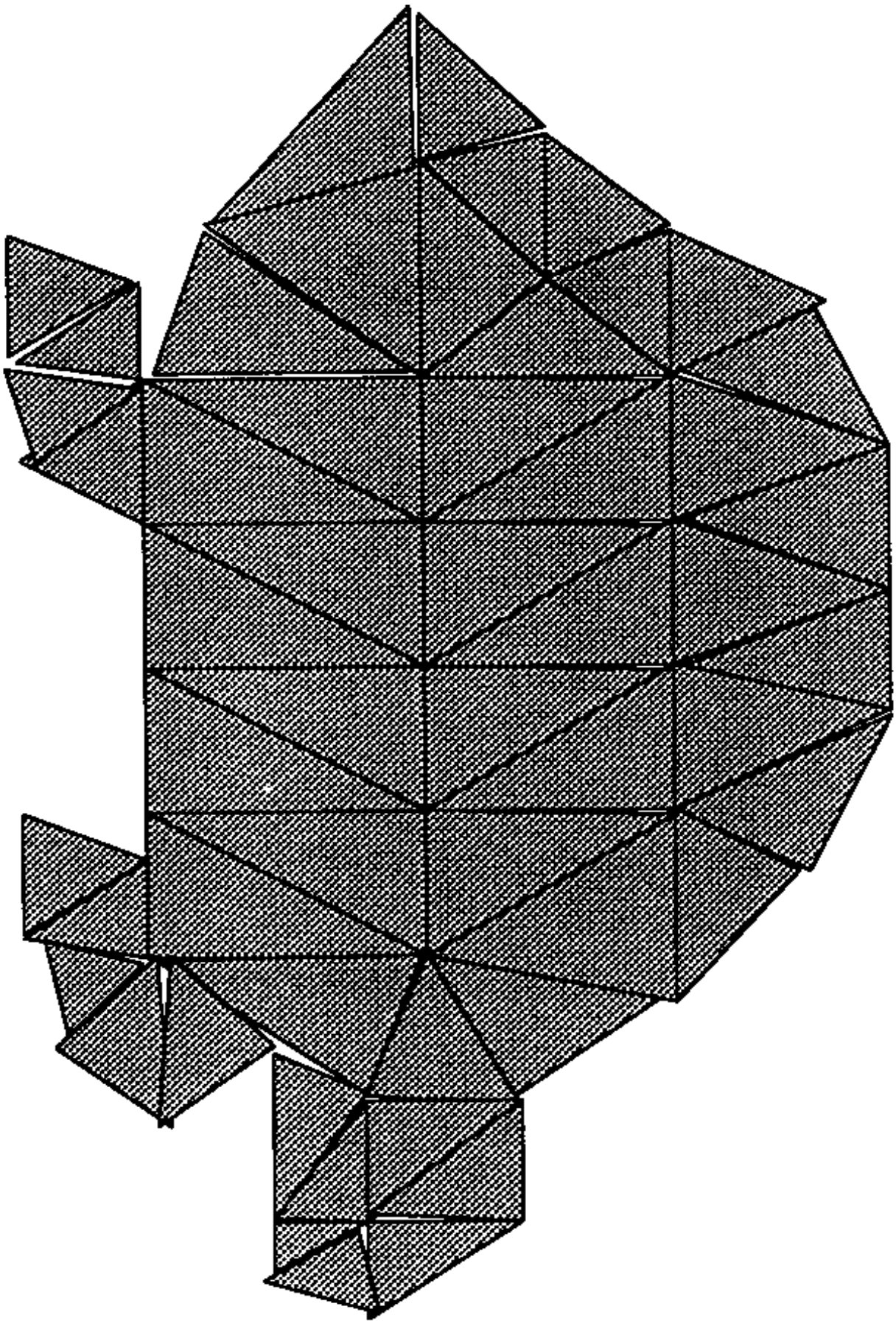
도면 7


삼차원 메쉬 (MO)
100
삼차원 데이터 분석기
501
502
MCOM1
MCOM2
MCOMn
MCOM 부호화기 1
MCOM 부호화기 2
MCOM 부호화기 n
압축된 비트 스트림
503
MCOM 복호화기 1
MCOM 복호화기 2
MCOM 복호화기 n
504
복원된 삼차원 메쉬
505

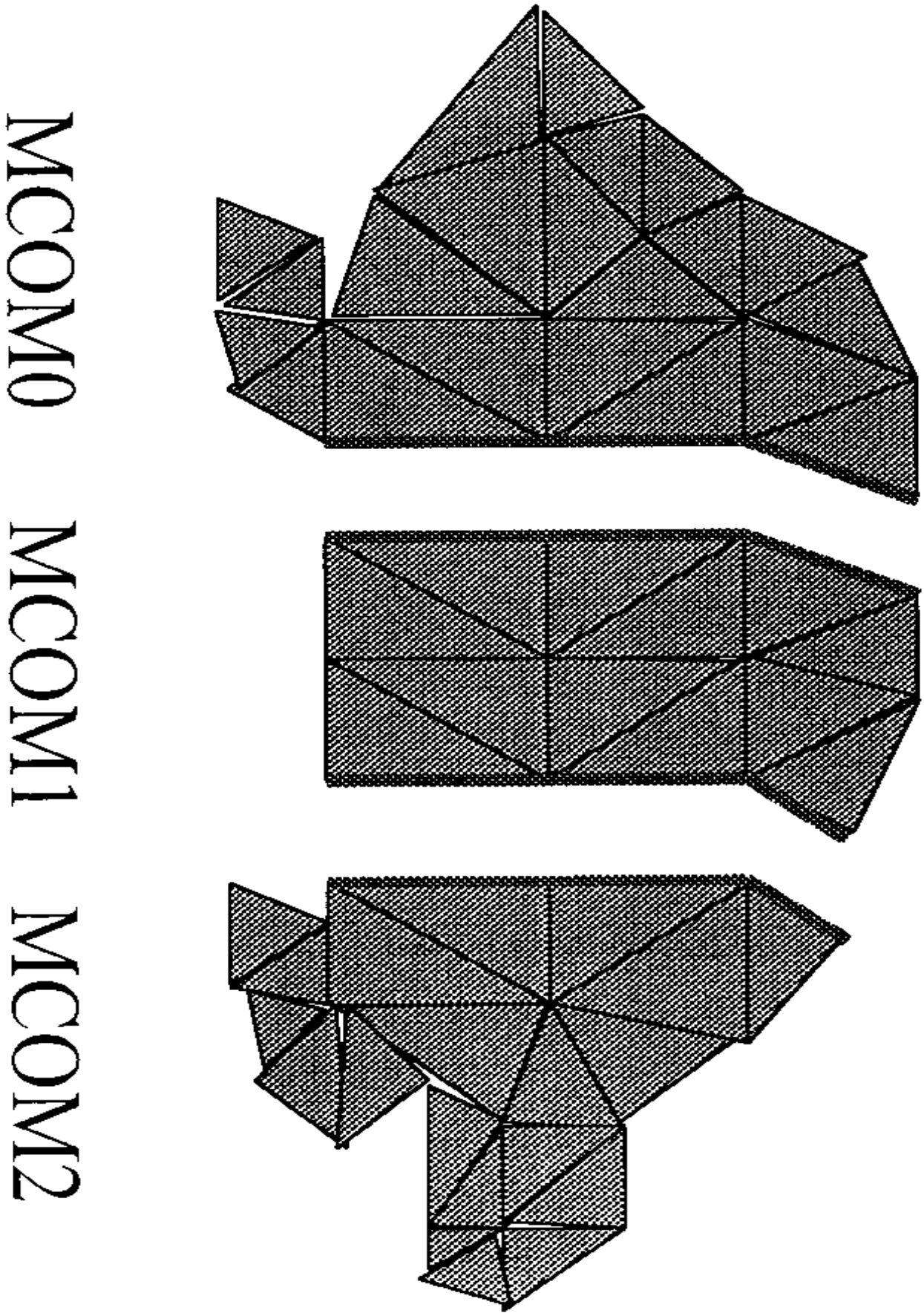
도면 8


삼차원 메쉬 (MO)
100
삼차원 데이터 분석기
501
502
MCOM 부호화기 1
MCOM 부호화기 2
MCOM 부호화기 n
503
압축된 비트 스트림
MCOM 복호화기 1
MCOM 복호화기 2
MCOM 복호화기 n
504
삼차원 데이터 합성기
506
복원된 삼차원 메쉬
110

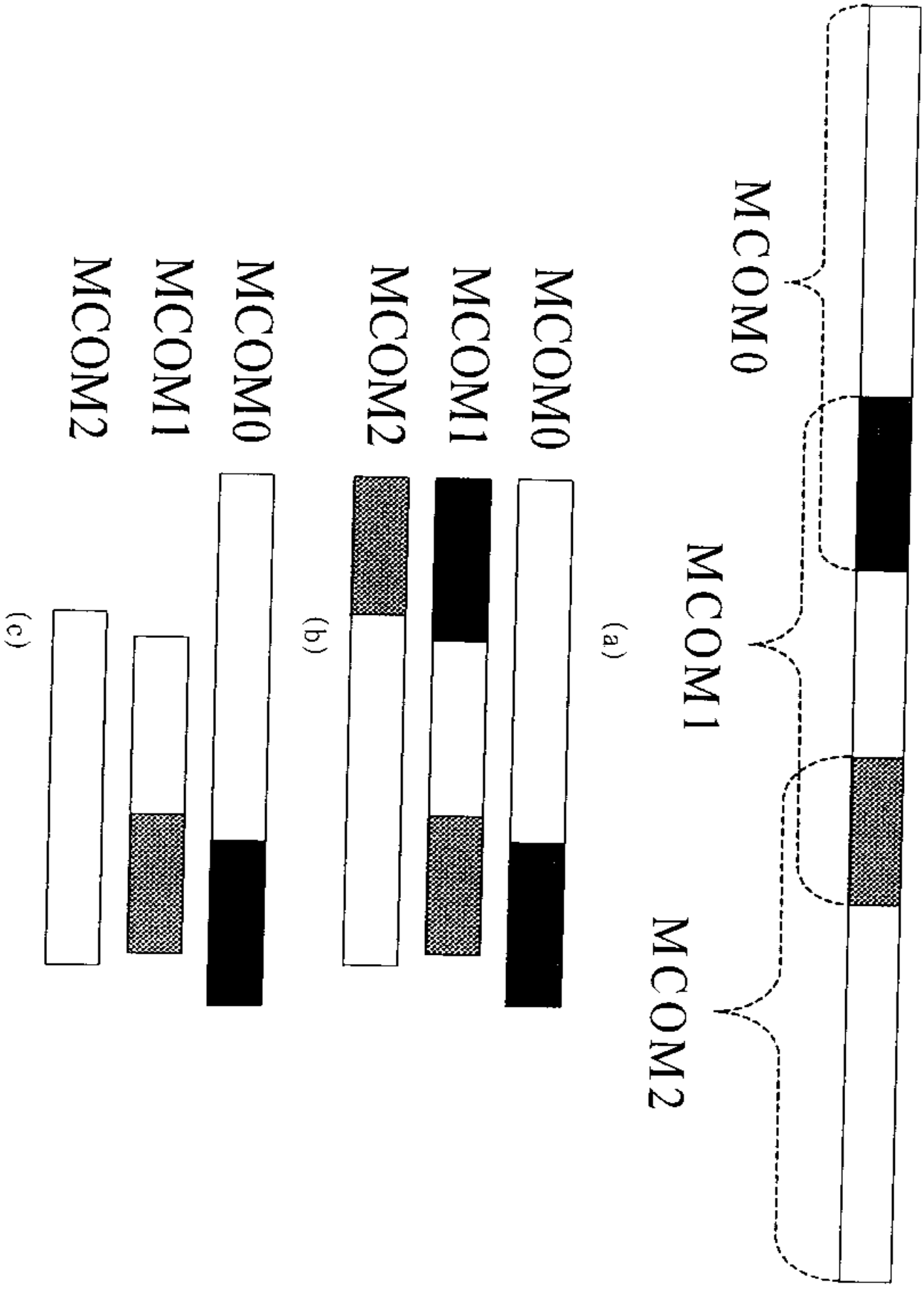
도면 9



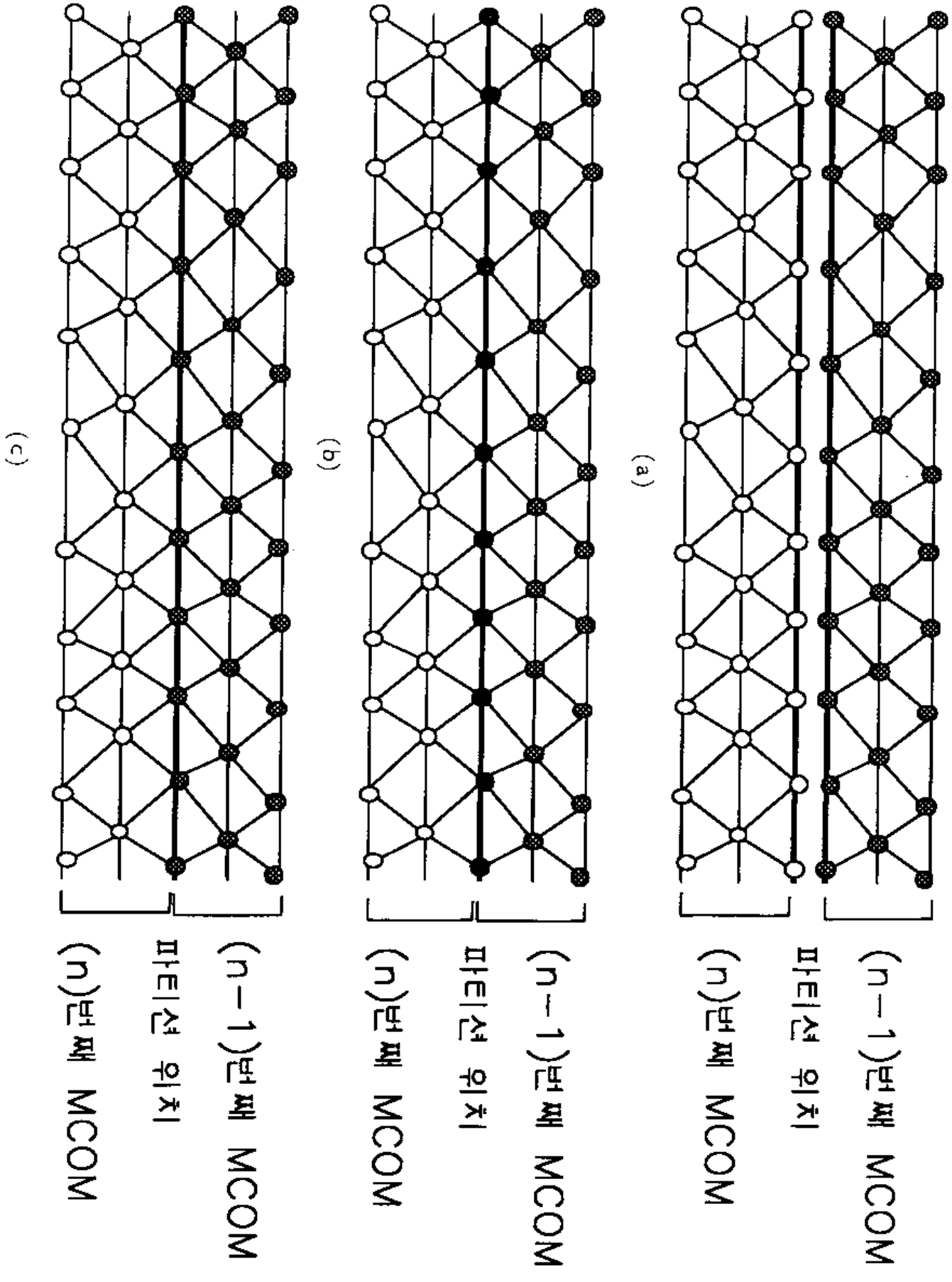
도면 10



도면 11



도면 12


(a)
(n-1)번째 MCOM
파티션 위치
(n)번째 MCOM
(b)
(n-1)번째 MCOM
파티션 위치
(n)번째 MCOM
(c)
(n-1)번째 MCOM
파티션 위치
(n)번째 MCOM