(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2011년10월26일
(11) 등록번호 10-1076876
(24) 등록일자 2011년10월19일

(51) Int. Cl.
H04N 7/24 (2011.01) *H04N 13/00* (2006.01)
(21) 출원번호 10-2009-0088228
(22) 출원일자 2009년09월17일
심사청구일자 2009년09월17일
(65) 공개번호 10-2010-0102516
(43) 공개일자 2010년09월24일
(30) 우선권주장
1020090020917 2009년03월11일 대한민국(KR)
(56) 선행기술조사문헌
KR1020100083980 A
KR1020100004038 A
KR1020070029793 A
KR1020100083957 A
전체 청구항 수 : 총 10 항

(73) 특허권자
경희대학교 산학협력단
경기도 용인시 기흥구 서천동 1 경희대학교 국제캠퍼스내
(72) 발명자
박광훈
경기도 성남시 분당구 분당동 45번지, 동아빌라 B동-302호
김경용
전라남도 영광군 영광읍 남천리 349-2
(74) 대리인
박진석, 특허법인 신지, 유경열

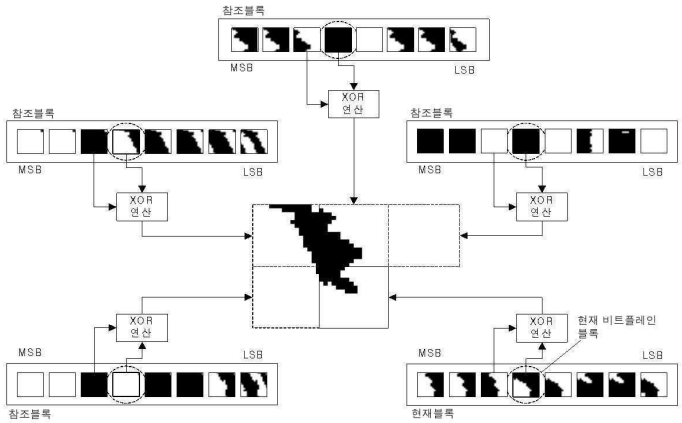
심사관 : 조우연

(54) 블록기반 깊이정보 맵의 코딩 방법과 장치, 및 이를 이용한 3차원 비디오 코딩 방법

(57) 요 약

블록기반 깊이정보 맵의 코딩 방법과 장치, 및 이를 이용한 3차원 비디오 코딩 방법을 제공한다. 본 발명의 일 실시예에 따른 깊이정보 맵의 디코딩 방법에서는 입력된 비트스트림에 대하여 소정 크기의 블록 단위로 비트플레인 디코딩 기법을 수행하여 깊이정보 맵을 재구성한다. 예를 들어, 디코딩된 코딩 모드 정보에 기초하여, 블록 단위로 비트플레인 디코딩 기법 또는 기존의 디씨티 기반 디코딩 기법을 선택하여 디코딩을 수행할 수 있다. 그리고 비트플레인 디코딩 기법은 비트플레인 블록 단위로 적응적으로 배타적 논리합(XOR) 연산을 수행하는 과정을 포함할 수 있다. 예를 들어, 비트스트림으로부터 디코딩된 XOR 연산 정보의 값에 따라서 비트플레인 블록 단위로 적응적으로 XOR 연산을 수행할 수 있다.

대 표 도 - 도16



이 발명을 지원한 국가연구개발사업
과제고유번호 R0A-2005-000-10061-0
부처명 한국과학재단
연구관리전문기관
연구사업명 국가지정연구실사업
연구과제명 실감방송을 위한 유니버셜 스케일러블 비디오 코딩기술
기여율
주관기관 경희대학교산학협력단
연구기간 2005년 04월 01일 ~ 2010년 03월 31일

특허청구의 범위

청구항 1

삭제

청구항 2

삭제

청구항 3

삭제

청구항 4

삭제

청구항 5

삭제

청구항 6

삭제

청구항 7

삭제

청구항 8

삭제

청구항 9

삭제

청구항 10

삭제

청구항 11

삭제

청구항 12

삭제

청구항 13

삭제

청구항 14

삭제

청구항 15

삭제

청구항 16

삭제

청구항 17

삭제

청구항 18

삭제

청구항 19

삭제

청구항 20

삭제

청구항 21

삭제

청구항 22

삭제

청구항 23

삭제

청구항 24

삭제

청구항 25

삭제

청구항 26

N비트로 표현되는 깊이정보 맵 블록을 디코딩하는 방법에 있어서,

각 비트플레인 블록에 대한 블록 모드 정보에 따라서 디코딩을 수행하여 n(≤N)개의 비트플레인 블록을 복원하는 단계;

각 비트플레인 블록에 대한 배타적 논리합(XOR) 연산 정보에 따라서 상기 n개의 복원된 비트플레인 블록에 대하여 참조 비트플레인 블록과 적응적으로 XOR 연산을 수행하는 단계; 및

상기 XOR 연산이 수행되거나 및/또는 상기 XOR 연산이 수행되지 않은 상기 n개의 복원된 비트플레인 블록을 결합하는 단계를 포함하는 깊이정보 맵의 디코딩 방법.

청구항 27

제26항에 있어서,

n이 N보다 작은 경우에, 상기 결합된 비트플레인 블록의 각 픽셀마다 (N-n)개의 하위 레벨의 비트를 추가하여 N비트로 표현되는 상기 깊이정보 맵 블록을 재구성하기 위한 재구성 단계를 더 포함하는 깊이정보 맵의 디코딩 방법.

청구항 28

제27항에 있어서,

상기 블록 모드 정보는 'all_0 모드', 'all_1 모드', 및'이진영상 코딩 모드' 중에서 하나인 깊이정보 맵의 디

코딩 방법.

청구항 29

제27항에 있어서,

상기 블록 모드 정보는 'No Update Without MV 모드', 'all_0 모드', 'all_1 모드', 'No Update with MV 모드', 및 '이진영상 코딩 모드' 중에서 하나인 깊이정보 맵의 디코딩 방법.

청구항 30

제28항 또는 제29항에 있어서,

상기 이진영상 코딩 모드는 컨텍스트 기반 산술 인코딩(Context-base Arithmetic Encoding, CAE) 기법을 나타내는 깊이정보 맵의 디코딩 방법.

청구항 31

제30항에 있어서,

상기 CAE 기법을 수행하기 위한 참조 비트플레인은, 현재 깊이정보 맵 블록에 인접한 참조 깊이정보 맵 블록들 각각에 대하여 비트플레인 분리를 수행하여 생성된 M개의 비트플레인 블록들 중에서, 현재 비트플레인 블록과 동일한 레벨의 비트에 대한 비트플레인 블록에 대하여 적응적으로 XOR 연산을 수행하여 출력되는 비트플레인 블록들을 이용하여 생성되는 깊이정보 맵의 디코딩 방법.

청구항 32

제31항에 있어서,

상기 현재 비트플레인 블록의 XOR 연산 정보가 XOR 연산을 수행하는 것을 지시할 경우에만, 상기 참조 비트플레인을 생성하는 과정에서 XOR 연산을 수행하는 깊이정보 맵의 디코딩 방법.

청구항 33

제31항에 있어서,

상기 참조 깊이정보 맵 블록은 상기 재구성 단계에서 재구성된 깊이정보 맵 블록이거나 또는 디씨티 기반 디코딩 기법을 이용하여 재구성된 깊이정보 맵 블록인 깊이정보 맵의 디코딩 방법.

청구항 34

제27항 또는 제28항에 있어서,

상기 이진영상 코딩 모드는 런 길이 코딩 방법과 가변 길이 코딩 방법을 결합하는 기법, 컨텍스트 기반 적응적 이진 산술 코딩(Context-based Adaptive Binary Arithmetic Coding, CABAC) 기법, JBIG(Joint Bi-level Image processing Group) 기법, 및 쿼드 트리(Quad Tree) 기법 중에서 어느 하나의 기법을 나타내는 깊이정보 맵의 디코딩 방법.

청구항 35

제26항에 있어서,

상기 참조 비트플레인 블록은 현재 비트플레인 블록보다 한 단계 상위 레벨의 비트에 대한 재구성된 비트플레인 블록인 깊이정보 맵의 디코딩 방법.

청구항 36

삭제

청구항 37

삭제

청구항 38

삭제

청구항 39

삭제

청구항 40

삭제

청구항 41

삭제

청구항 42

삭제

청구항 43

삭제

청구항 44

삭제

명 세 서

발명의 상세한 설명

기 술 분 야

[0001] 본 발명은 비디오 코딩(video coding)에 관한 것으로, 보다 구체적으로 영상의 깊이정보 맵(depth map)을 인코딩(encoding)/디코딩(decoding)하는 방법과 장치, 및 이를 이용한 3차원 비디오 코딩 방법에 관한 것이다.

배 경 기 술

[0002] 실감 미디어에 대한 관심이 증가하면서 그에 대한 연구도 활발히 진행되고 있다. 실감 미디어란 공간과 시간이 제약된 가상의 환경에서도 실제 세계에서와 같이 보고 듣고 느낄 수 있도록 하는 미디어를 말한다. 사용자들은 실감 미디어를 통하여 가상의 환경에서도 실세계와 같은 현장감과 몰입감을 느낄 수가 있다. 실감 미디어는 방송, 통신 분야뿐만 아니라 광고, 전시, 교육, 의료 분야 등에 이르기까지 다양한 분야에 활용될 것으로 예상된다.

[0003] 실감 미디어의 일례로 다시점 영상(multi-view image)이 있다. 다시점 영상이란 동일 피사체에 대하여 다양한 시점(view points)에서 획득한 복수의 영상들을 가리킨다. 다시점 영상을 이용하면, 사용자는 다양한 각도에서 동일 피사체를 관찰할 수가 있다. 다시점 영상과 관련된 대표적인 비디오 코딩으로서, 다시점 비디오 코딩(Multi-view Video Coding, MVC)과 3차원 비디오 코딩(3D Video Coding)이 있다.

[0004] MVC는 복수(예컨대, 4개 또는 8개)의 카메라로부터 입력 받은 복수 시점의 영상들(Multiple View Images)을 효율적으로 인코딩/디코딩하기 위한 것이다. MVC는 다시점 영상을 획득하는데 사용된 카메라의 시점과 동일한 시점의 영상만을 사용자에게 제공한다. 따라서 MVC로 인코딩된 영상 데이터를 이용하여 보다 많은 시점의 영상을 제공하기 위해서는, 우선 보다 많은 카메라를 사용하여 영상을 촬영해야 한다. 하지만, 카메라의 개수를 증가시키면 그 만큼 영상 데이터의 양도 증가하는데, 현존하는 전달 미디어(방송, 통신, 저장 미디어 등)의 전송 대역폭이나 저장 용량에는 일정한 한계가 있기 때문에, MVC를 이용하여 사용자에게 제공할 수 있는 시점은 제약이 된다. 뿐만 아니라, MVC에서는 카메라의 시점과는 다른 시점 영상을 사용자에게 제공할 수 없다.

[0005] 3차원 비디오 코딩(3D video coding)은 MVC의 이러한 한계를 보완하기 위한 것이다. 3D 비디오 코딩은 영상을

획득하는데 사용된 카메라의 시점(예컨대, N개 시점)은 물론 그 이외의 시점 즉, 가상 시점의 영상을 생성하는 것을 지원한다. 이를 위하여 3D 비디오 코딩에서는 N개 시점의 영상에 추가하여 깊이정보 맵(depth map)도 함께 코딩한다. 가상 시점의 영상은 인접한 시점에 위치한 카메라에서 획득한 영상들과 그 깊이정보 맵들을 이용하여 뷰 보간(view interpolation)을 수행함으로써 생성할 수가 있다. 이러한 3D 비디오 코딩은 FTV(Free View-point Television) 시스템 등과 같은 3차원 디스플레이 시스템을 지원할 수 있다.

[0006] 도 1은 3D 비디오 코딩이 적용되는 FTV 시스템에서의 랜더링(rendering)의 일례를 보여 주는 도면이다. 도 1에 도시된 FTV 시스템은 N(N=5)개의 카메라를 사용하는 경우로서, 코딩 시에 N(N=5)개의 카메라에서 획득한 영상들(검정색 카메라로 표시된 시점의 영상들)의 이미지정보와 각 영상의 깊이정보를 부호화한다. 그리고 FTV 시스템은 부호화된 이미지정보와 깊이정보를 복호화한 다음, 복호화된 각 카메라 시점의 이미지정보와 깊이정보를 이용하여, 해당 카메라의 시점뿐만 아니라 해당 카메라의 시점과는 다른 시점의 영상들, 즉 부채꼴 모양의 음영으로 표시된 영역 내의 임의의 시점에서의 영상을 렌더링할 수 있다.

[0007] 이와 같이, FTV 시스템에 적용되는 3D 비디오 코딩에서는 사용자에게 제공하고자 하는 모든 시점의 영상들을 카메라로 획득할 필요가 없다. 따라서 가능한 많은 시점의 영상을 제공하는데 있어서, 3D 비디오 코딩은 MVC에 비하여 전송 대역폭이나 저장 용량의 제한으로부터 상대적으로 자유롭다. 그리고 3D 비디오 코딩을 이용할 경우에는, 사용자가 원하는 임의의 시점의 영상을 특별한 제한 없이 제공할 수가 있다.

[0008] 그런데, MVC에서는 입력 받은 시점의 영상의 이미지정보만을 인코딩/디코딩하는 과정만이 필요하다. 반면, 3D 비디오 코딩에서는 영상의 이미지정보 외에 깊이정보 맵도 인코딩/디코딩을 해야 한다. 즉, MVC와 비교했을 때, 3D 비디오 코딩에서는 깊이정보 맵을 생성하는 과정, 깊이정보 맵을 인코딩/디코딩하는 과정, 및 깊이정보 맵을 이용하여 가상의 시점의 영상을 생성하는 과정이 추가적으로 필요하다. 3D 비디오 코딩과 관련하여 현재 진행되고 있는 대부분의 연구들은 깊이정보 맵의 생성 과정과 가상의 시점의 영상의 생성 과정에 집중되고 있으며, 깊이정보 맵을 인코딩/디코딩 과정에 대해서는 많은 연구가 진행되고 있지 않다. 이것은 깊이정보 맵의 인코딩/디코딩 과정은 이미지정보(휘도나 색차 등의 정보)를 코딩하기 위한 기존의 기법들(이하, '이미지정보 코딩 기법'이라 한다)을 그대로 사용하면 충분하다는 인식에 근거한다.

[0009] 깊이정보 맵이란 현재 시점에서 카메라와 사물(object) 사이의 거리를 나타내기 위한 것이다. 카메라와 사물 사이의 거리는 사물의 위치, 즉 이미지 내에서 사물이 놓여 있는 공간적인 위치에 따라서 달라질 수 있다. 따라서 깊이정보 맵도 이미지정보와 마찬가지로 픽셀 단위로 표현할 수 있다. 예를 들어, 현재 영상의 이미지정보와 동일한 해상도로 상기 거리를 일정한 비트수로 표현함으로써, 깊이정보 맵을 생성할 수 있다.

[0010] 그런데, 사물까지의 실제 거리는 프레임에 따라서 그 변동 범위가 클 수가 있기 때문에, 각 픽셀에서의 사물까지의 거리, 즉 깊이정보는 절대적인 값이 아니라 상대적인 값으로 표현한다. 예를 들어, 동일 프레임 내에서의 최단 거리(Z_{near})와 최장 거리(Z_{far})를 구한 다음, 이를 이용한 상대적인 값으로 깊이정보를 나타낼 수 있다. 예를 들어, 깊이 정보를 8비트로 나타낼 경우에, 최단 거리(Z_{near})의 사물이 있는 픽셀은 '255'으로 나타내고, 최장 거리(Z_{far})의 사물이 있는 픽셀은 '0'로 나타내며, 또한 그 중간 거리의 사물이 있는 픽셀들의 경우에는 '0'과 '255' 사이의 소정의 값으로 비례적으로 나타낼 수 있다.

[0011] 이와 같이, 깊이정보 맵은 실제 사물과 카메라 사이의 거리에 기초한 거리정보를 표현한 것이다. 반면, 기존의 비디오 코딩에서 인코딩/디코딩의 대상이 되는 데이터들은 휘도나 색차 또는 RGB값 등과 같은 이미지정보이다. 동일한 피사체는 그 거리, 밝기, 색상 등이 일치할 수도 있다는 점에서 깊이정보와 이미지정보는 유사한 특성도 보이지만, 서로 간에 관련성이 낮은 경우도 많이 존재한다. 예를 들어, 휘도나 색차 등의 이미지정보가 전혀 다른 사물이나 또는 같은 사물의 다른 두 부분은 깊이정보는 유사할 수가 있다. 반대로, 휘도나 색차 등의 이미지정보는 유사하다고 하더라도 깊이정보는 서로 다를 수가 있다.

[0012] 그런데, 기존의 비디오 코딩 기법들은 이미지정보의 특성을 고려하여 효율적으로 압축하기 위하여 개발된 것이다. 이러한 기존의 비디오 코딩 기법들을 깊이정보 맵의 인코딩/디코딩에 그대로 적용하면, 이미지정보와는 다른 특성을 갖는 깊이정보맵을 효율적으로 인코딩하기가 쉽지 않다. 또한, 깊이정보 맵을 사용하여 카메라의 시점과는 다른 시점의 영상을 생성할 경우에는, 깊이정보 맵의 정확도가 이미지의 화질에 큰 영향을 미칠 수가 있다. 따라서 MVC와 대비되는 3D 비디오 코딩의 장점을 극대화하기 위해서는, 깊이정보 맵에 고유한 특성을 고려한 효율적인 깊이정보 맵의 인코딩/디코딩 기법이 개발될 필요가 있다.

발명의 내용

해결 하고자하는 과제

[0013] 본 발명이 해결하고자 하는 하나의 과제는 깊이정보 맵을 효율적으로 인코딩 및 디코딩하기 위한 방법과 장치, 및 이를 이용한 3차원 비디오 코딩 방법을 제공하는 것이다.

[0014] 본 발명이 해결하고자 하는 다른 하나의 과제는 깊이정보 맵 영상의 화질을 향상시킬 수 있는 깊이정보 맵의 인코딩/디코딩 방법과 장치, 및 이를 이용한 3차원 비디오 코딩 방법을 제공하는 것이다.

[0015] 본 발명이 해결하고자 하는 또 다른 하나의 과제는 이미지정보와는 다른 깊이정보에 고유한 특성을 고려한 깊이정보 맵의 인코딩/디코딩 방법과 장치, 및 이를 이용한 3차원 비디오 코딩 방법을 제공하는 것이다.

[0016] 본 발명이 해결하고자 하는 또 다른 하나의 과제는 3D 비디오 코딩에 있어서 깊이정보 맵의 코딩 효율을 향상시킴으로써 보간되는 시점 영상의 화질을 개선할 수 있는 깊이정보 맵의 인코딩/디코딩 방법과 장치, 및 이를 이용한 3차원 비디오 코딩 방법을 제공하는 것이다.

과제 해결수단

[0017] 상기한 과제를 해결하기 위한 본 발명의 일 실시예에 따른 깊이정보 맵의 디코딩 방법은 입력된 비트스트림에 대하여 소정 크기의 블록 단위로 비트플레인 디코딩 기법을 수행하여 깊이정보 맵을 재구성한다. 예를 들어, 상기 블록 단위의 비트플레인 디코딩 기법은 비트스트림으로부터 각 깊이정보 맵 블록에 대한 코딩 모드 정보를 디코딩하는 단계를 포함하고, 상기 디코딩된 코딩 모드 정보가 비트플레인 디코딩 기법을 지시할 경우에만 상기 비트플레인 디코딩 기법을 수행할 수 있다. 그리고 상기 비트플레인 디코딩 기법은 비트플레인 블록 단위로 적응적으로 배타적 논리합(XOR) 연산을 수행하는 XOR 연산 단계를 포함할 수 있다. 보다 구체적으로, 상기 깊이정보 맵의 비트스트림으로부터 XOR 연산 정보를 디코딩하는 단계를 포함하고, 상기 XOR 연산 단계는 상기 디코딩된 XOR 연산 정보의 값이 해당 비트플레인 블록에 대하여 XOR 연산을 수행하는 것을 지시할 경우에 수행될 수 있다. 이 경우에, 상기 XOR 연산 단계를 거친 비트플레인 블록 및/또는 상기 XOR 연산 단계를 거치지 않은 비트플레인 블록을 결합하여 깊이정보 맵 블록을 구성하는 단계를 더 포함할 수 있다.

[0018] 상기한 과제를 해결하기 위한 본 발명의 다른 실시예에 따른 깊이정보 맵의 디코딩 방법은, 입력된 비트스트림을 비트플레인 단위로 디코딩하여 복원한 비트플레인 블록에 대하여 참조 비트플레인 블록과 배타적 논리합(XOR) 연산을 수행하는 과정을 포함한다. 예를 들어, 상기 디코딩 방법은 상기 입력된 비트스트림으로부터 상기 비트플레인 블록을 복원하기 위한 비트플레인 디코딩 단계, 상기 복원된 비트플레인 블록에 대하여 적응적으로 배타적 논리합(XOR) 연산을 수행하기 위한 XOR 연산 단계, 및 상기 XOR 연산이 수행된 상기 복원된 비트플레인 블록 및/또는 상기 XOR 연산이 수행되지 않은 상기 복원된 비트플레인 블록을 결합하기 위한 비트플레인 결합 단계를 포함할 수 있다.

[0019] 상기한 과제를 해결하기 위한 본 발명의 또 다른 실시예에 따른 깊이정보 맵의 디코딩 방법은 비트스트림으로부터 각 깊이정보 맵 블록에 대한 코딩 모드 정보를 디코딩하는 단계, 및 상기 디코딩된 코딩 모드 정보에 따라서 블록 단위의 비트플레인 디코딩 기법 또는 블록 단위의 디씨티 기반 디코딩 기법을 이용하여 상기 깊이정보 맵을 재구성하는 깊이정보 맵의 재구성 단계를 포함한다.

[0020] 상기한 과제를 해결하기 위한 본 발명의 또 다른 실시예에 따른 깊이정보 맵의 디코딩 방법은 N비트로 표현되는 깊이정보 맵 블록을 디코딩하는 방법으로서, 각 비트플레인 블록에 대한 블록 모드 정보에 따라서 디코딩을 수행하여 n(≤N)개의 비트플레인 블록을 복원하는 단계, 각 비트플레인 블록에 대한 배타적 논리합(XOR) 연산 정보에 따라서 상기 n개의 복원된 비트플레인 블록에 대하여 참조 비트플레인 블록과 적응적으로 XOR 연산을 수행하는 단계, 및 상기 XOR 연산이 수행되거나 및/또는 상기 XOR 연산이 수행되지 않은 상기 n개의 복원된 비트플레인 블록을 결합하는 단계를 포함한다.

[0021] 상기한 과제를 해결하기 위한 본 발명의 또 다른 실시예에 따른 깊이정보 맵의 디코딩 방법은 입력되는 비트스트림으로부터 블록 모드 정보를 디코딩하는 단계 및 상기 블록 모드 정보에 따라서 동일 레벨 블록 디코딩 기법 또는 이진영상 디코딩 기법을 이용하여 현재 비트플레인 블록을 복원하는 단계를 포함한다.

효 과

[0022] 본 발명의 실시예에 의하면, 3D 비디오 코딩에 있어서 깊이정보 맵의 코딩 효율을 향상시킬 수가 있다. 보다 구체적으로, 깊이정보 맵을 이용하여 복원된 영상, 특히 객체 경계 부근에서의 깊이정보 맵의 화질을 향상시키고 또는 비트율을 감소시킬 수가 있다. 따라서 본 발명의 실시예에 따른 깊이정보 맵의 코딩 방법을 이용하면, 3D 비디오 코딩에서 카메라로 캡쳐된 원영상의 화질은 물론 이 원영상을 이용하여 보간되는 영상의 화질을 향상시킬 수가 있다.

발명의 실시를 위한 구체적인 내용

[0023] 이하, 첨부된 도면들을 참조하여 본 발명의 실시예를 상세하게 설명한다. 사용되는 용어들은 실시예에서의 기능을 고려하여 선택된 용어들로서, 그 용어의 의미는 사용자, 운용자의 의도 또는 관례 등에 따라 달라질 수 있다. 그러므로 후술하는 실시예들에서 사용된 용어의 의미는, 본 명세서에 구체적으로 정의된 경우에는 그 정의에 따르며, 구체적인 정의가 없는 경우는 당업자들이 일반적으로 인식하는 의미로 해석되어야 할 것이다.

[0024] 본 발명의 실시예에 따른 인코딩/디코딩 장치 및 방법을 설명하기에 앞서 우선 깊이정보 맵에 고유한 특성에 관하여 살펴보기로 한다.

[0025] 도 2는 실제 영상과 그것의 깊이정보 맵 영상의 일례를 보여 주는 것으로서, 도 2의 (a)는 실제 영상의 일례로서 MVC의 테스트 시퀀스의 하나인 'Breakdancers' 영상 시퀀스이고, 도 2의 (b)는 이 'Breakdancers' 영상 시퀀스의 깊이정보 맵 영상이다. 도 2의 (b)의 깊이정보 맵 영상은 각 픽셀의 깊이정보를 8비트로 나타낸 것으로, 카메라와 가까울수록 큰 값(밝은 값)으로 표현하였다.

[0026] 이러한 8비트로 표현되는 깊이정보 맵을 이용하여 각 픽셀에서의 실제 거리(Z)를 구하는 방법의 일례는 수학식 1과 같이 표현할 수 있다.

수학식 1

$$Z = Z_{far} + v \cdot \frac{Z_{near} - Z_{far}}{255} \quad \text{with } v \in [0, \ldots, 255]$$

[0027]

[0028] 여기서, v는 해당 픽셀에서의 깊이정보 값이고, Z_{far}와 Z_{near}는, MPEG-C Part 3에 정의되어 있는 파라미터로써, 각각 해당 영상에서 실제 거리가 가장 먼 부분까지의 거리와 가장 가까운 부분까지의 거리를 나타낸다. 따라서 깊이정보 맵에 표시되는 깊이정보의 값은, 영상에서 보여지는 실제 세계에서의 가장 먼 영역과 가장 가까운 영역 사이를 2^N (여기서, N은 깊이 정보맵을 표현하는 비트 수를 나타내며, 도 1의 (b)의 경우에는 N이 8이다) 등분하여 표현한 것이라고 할 수 있다.

[0029] 도 3a 및 도 3b는 각각 도 2의 영상에서 이미지정보의 레벨 분포와 깊이정보의 레벨 분포를 보여주기 위한 그래프이다. 도 3a는 도 2의 (a)의 실제 영상에서 각 픽셀의 이미지정보의 일례인 휘도, 즉 밝기를 표현한 3차원 그래프이고, 도 3b는 도 2의 (b)의 깊이정보 맵 영상에서 각 픽셀의 깊이정보의 레벨을 표현한 3차원 그래프이다. 도 3a 및 도 3b를 참조하면, 일반적으로 같은 영상에서 밝기의 변화는 다소 급격하지만 깊이정보 레벨의 변화는 상대적으로 완만하다는 것을 알 수 있다. 다만, 이미지정보와 깊이정보 레벨 사이의 이러한 특성 차이를 일반화하는 것은 일정한 한계가 있으며, 영상의 종류에 따라서 예외가 있을 수도 있다.

[0030] 도 4a 및 도 4b는 각각 이미지정보와는 다른 깊이정보에 고유한 특성의 일례를 보여 주기 위한 도면으로서, 도 4a는 객체의 경계를 포함하는 깊이정보 맵 블록을 3차원으로 표현한 깊이정보 그래프이고, 도 4b는 배경 부분에서의 깊이정보 맵 블록을 3차원으로 표현한 깊이정보 그래프이다. 도 4a를 참조하면, 객체의 경계를 포함하는 블록에서는 비록 객체의 경계 부분에서 깊이정보의 급격한 변화가 존재하기는 하지만, 객체의 내부나 객체의 외부(배경)에서는 서로 인접한 주변 픽셀들에서는 깊이정보가 거의 비슷한 값으로 분포하고 있다는 것을 알 수 있다. 반면, 도 4b를 참조하면, 객체 내부나 배경 부분만을 포함하는 블록에서는 픽셀들 각각의 깊이정보가 46~54 사이로 분포하여 큰 변화가 없는 것처럼 보이기는 하지만, 깊이정보는 픽셀의 위치에 따라서 약간의 차이가 존재한다는 것을 확인할 수 있다.

[0031] 깊이정보의 이러한 특성은 깊이정보 맵 영상을 비트플레인 단위로 분리하면 보다 분명하게 파악될 수 있다. 도 5의 (a)와 (b)는 각각 도 2의 (a)의 실제 영상과 도 2의 (b)의 깊이정보 맵 영상을 비트플레인 단위로 표현한 것으로서, 위쪽에서부터 각각 비트 7 또는 최상위 비트(Most Significant Bit, MSB), 비트 6(MSB-1), 비트 5(MSB-2), 비트 4(MSB-3), 비트 3(MSB-4), 비트 2(MSB-5), 비트 1(MSB-6), 및 비트 0 또는 최하위 비트(Least Significant Bit, LSB)에 대한 비트플레인들을 보여 준다. 전체적으로 볼 때, 깊이정보 맵 영상의 비트플레인(도 4의 (b))은 실제 영상(예컨대, 이미지정보 중에서 밝기로 표현된 영상)의 비트플레인(도 4의 (a))에 비하여 상당히 단조로운 형태를 보여 주는데, 이하 이에 대하여 보다 상세하게 설명한다.

[0032] 도 5의 (a)를 참조하면, 실제 영상(이미지정보)에서 최상위 비트(MSB)에 대한 비트플레인은 단조로운 형태를 갖지만, 하위 비트의 비트플레인으로 갈수록 상당히 복잡해지는 것을 알 수 있다. 특히, 최하위 비트(LSB)를 포함한 하위 세 개의 비트플레인들은 모두 백색 잡음(white noise)과 같은 형태로 상당히 복잡하다는 것을 알 수 있다. 그리고 도 5의 (b)를 참조하면, 깊이정보 맵 영상에서도 최상위 비트(MSB)로부터 최하위 비트(LSB)로 갈수록 비트플레인이 복잡해진다는 점에서 실제 영상의 비트플레인과 유사하기는 하나, 전체적으로 실제 영상의 비트플레인에 비하여 단조로운 형태를 보이는 것을 알 수 있다. 또한, 깊이정보 맵 영상에서는 일부 상위 비트의 비트플레인과 일부 하위 비트의 비트플레인 사이에서 형태적으로 상관관계가 나타나고 있는데, 이것은 도 5의 (c)를 통해서 분명하게 확인할 수 있다.

[0033] 도 5의 (c)는 깊이정보 맵 영상에 대하여 그레이코딩(gray coding)을 수행하여 비트플레인 단위로 도시한 것이다. 그레이코딩은 연속하는 값을 표현할 경우에 1-비트만을 변경하여 표현하는 변환 기법을 가리키는데, 그레이코드를 적용한다고 표현하거나 또는 비트플레인간 배타적 논리합(eXclusive OR, XOR) 연산을 수행한다고도 표현할 수 있다. 그레이코딩에서는 예를 들어, 현재 레벨보다 한 단계 하위의 비트에 대한 비트플레인과 XOR 연산을 수행하거나 또는 한 단계 상위의 비트에 대한 비트플레인과 XOR 연산을 수행함으로써, 현재 레벨의 비트에 대한 변환된 비트플레인 (또는 그레이코딩된 비트플레인 또는 XOR 연산된 비트플레인)을 표현할 수 있다. 이하에서는, 후자의 경우를 예로 들어서 설명하지만, 전자의 경우에도 후술하는 본 발명의 실시예가 동일하게 적용될 수 있다는 것은 당업자에게 자명하다.

[0034] 도 5의 (c)에서도 위쪽에서부터 각각 비트 7 또는 최상위 비트(MSB), 비트 6(MSB-1), 비트 5(MSB-2), 비트 4(MSB-3), 비트 3(MSB-4), 비트 2(MSB-5), 비트 1(MSB-6), 및 비트 0 또는 최하위 비트(LSB)에 대한 그레이코딩된 비트플레인을 보여 준다. 도 5의 (c)를 참조하면, 깊이정보 맵의 그레이코딩된 비트플레인들은 원래의 비트플레인(도 5의 (b) 참조)에 비하여 더욱 단조로운 형태를 보인다는 것을 알 수 있다.

[0035] 이러한 깊이정보 맵에 고유한 특성은 깊이정보 맵 영상을 블록 단위로 검토하면 보다 구체적으로 파악할 수 있다. 도 6의 (a)는 도 2의 (b)의 깊이정보 맵의 일부이고, 도 6의 (b), (c), 및 (d)는 각각 도 6의 (a)의 깊이정보 맵에서 임의로 선택된 블록(예컨대, 16ㅅ16 블록)에 대한 비트플레인 블록들(최상위 비트(MSB)플레인 블록부터 최하위 비트(LSB)플레인 블록까지)과 이 비트플레인 블록들을 그레이코딩한 비트플레인 블록들을 보여 주는 도면이다. 여기서, 선택된 블록들은 모두 객체 경계 부분 또는 객체 경계를 포함하는 블록들이지만, 반드시 여기에만 한정되는 것은 아니다.

[0036] 도 6의 (b), (c), 및 (d) 각각에서 선택된 비트플레인 블록들을 살펴보면, 각 비트플레인 블록의 이진영상(binary image)들의 전부 또는 일부는 서로 완전히 일치하거나 또는 서로 반전되게 일치되는 것을 알 수 있다. 이러한 깊이정보 맵의 특성은 반드시 모든 블록들에서 나타나는 것은 아니며 일부 블록들에서만 나타날 수 있으며, 특히 객체의 경계 부근의 블록들에서 빈번하게 발생할 수 있다. 이 경우에, 깊이정보 맵은 일부 블록들, 특히 객체의 경계 부근 또는 객체의 경계를 포함하는 깊이정보 맵 블록들은, 그 전부 또는 일부가 서로 완전히 일치하거나 또는 반전되게 일치할 가능성이 높다.

[0037] 이러한 깊이정보 맵의 특성은 비트플레인 블록에 그레이코딩(gray coding)을 수행하면 더욱 명확하게 확인할 수 있다. 도 6의 (b), (c), 및 (d) 각각에 도시되어 있는 바와 같이, 그레이코딩된 비트플레인 블록들(MSB플레인 블록을 제외한 다른 비트플레인 블록들) 중에서 블록 전체가 같은 값(0(0) 또는 255(1))을 갖는 블록들이 많이 생긴다. 이것은 전술한 깊이정보 맵의 특성으로 인하여, 현재 비트플레인의 이진영상이 그 상위 비트플레인의 이진영상과 완전히 일치할 경우에는 그레이코딩된 비트플레인 블록의 픽셀들은 모두 '0(0)'이 되고, 반전되게 일치할 경우에는 그레이코딩된 비트플레인 블록의 픽셀들은 모두 '255(1)'이 되기 때문이다.

[0038] 그런데, 깊이정보 맵 영상에 그레이코드를 적용하게 되면, 특정 비트에 대한 비트플레인의 이진영상(그레이코딩된 비트플레인의 이진영상)은 그레이코드를 적용하기 이전의 비트플레인보다 더욱 단조로운 형태가 되지만, 이와 반대의 경우도 발생할 수가 있다. 즉, 일부 비트에 대한 비트플레인의 이진영상은 그레이코딩된 비트플레인

이 더욱 복잡한 형태가 될 수도 있다. 이것은 해당 레벨의 비트에 대한 비트플레인 이진영상이 인접 레벨의 비트에 대한 비트플레인 이진영상과 형태적으로 상관관계가 거의 없기 때문이다.

[0039] 도 7은 해당 레벨의 비트에 대한 비트플레인이 인접 레벨의 비트에 대한 비트플레인과 상관관계가 거의 없는 경우의 일례를 보여 주기 위한 것이다. 도 7의 (a)는 도 6의 (b)에서 비트 2(MSB-5)에 대한 비트플레인 블록의 이진영상이고, 도 7의 (b)는 도 6의 (b)에서 비트 2(MSB-5)에 대한 그레이코딩된 비트플레이 블록의 이진영상이다. 도 7의 (a)와 (b)를 참조하면, 일부 비트에 대한 비트플레인의 경우에는 그레이코드를 적용하면 이진영상이 더욱 복잡해진다는 것을 알 수 있다.

[0040] 이상에서 설명한 깊이정보 맵의 고유한 특성을 정리하면 다음과 같다.

[0041] (1) 깊이정보 맵은 픽셀 간 변화가 상당히 완만하다. 깊이정보 맵을 비트플레인 단위로 분리하고 각각의 분리된 비트플레인 이진 영상을 실제 영상(이미지정보)에 대한 비트플레인 이진 영상과 비교해보면, 깊이정보 맵의 비트플레인 이진 영상은 이미지정보의 비트플레인 이진 영상에 비하여 상당히 단조로운 형태를 갖는다.

[0042] (2) 깊이정보 맵에 대한 블록 단위의 비트플레인 이진 영상에 있어서, 일부 블록들, 예컨대 해당 블록이 객체의 경계를 포함하거나 또는 경계에 인접한 경우에는, 일부 레벨의 비트플레인의 이진 영상들 사이에는 완전히 일치되거나 또는 반전되게 일치하는 경우가 자주 발생한다. 다만, 이 경우에도 모든 레벨의 비트플레인 이진 영상들 사이에서 이러한 상관관계가 나타나는 것은 아니며, 다른 레벨의 비트플레인 이진 영상은 그레이코딩을 수행하면 이진 영상이 더욱 복잡해질 수도 있다.

[0043] 이러한 깊이정보 맵의 특성에 기초하여, 본 발명의 일 실시예에 따른 깊이정보 맵의 코딩 방법에서는 다음과 같은 방법으로 깊이정보 맵에 대하여 인코딩/디코딩을 수행한다.

[0044] 우선, 깊이정보 맵의 코딩에 있어서 비트플레인 인코딩 기법을 선택적 또는 적응적으로 적용한다. 예를 들어, 깊이정보 맵 영상(프레임 또는 픽쳐)에서 일부 블록은 비트플레인 인코딩 기법을 적용하지만, 다른 일부 블록은 비트플레인 인코딩 기법이 아닌 이미지정보를 코딩하기 위한 기존의 기법(이하에서는 '비트플레인 인코딩(부호화) 기법'과의 구별을 위하여, 이를 '이산여현변환(Discrete Cosine Transform, DCT)기반 인코딩(부호화) 기법'이라 한다. 따라서 본 명세서에서 비록 'DCT기반 인코딩 기법'이라는 명칭을 사용한다고 하더라도, 이것이 DCT 과정을 필수적으로 포함하는 인코딩 기법에 한정되는 것으로 해석되어서는 안되며, 변환 과정(transform process)에서 다른 기법이 사용될 수도 있다.)을 이용하여 부호화를 수행할 수 있다. 이 경우에, DCT기반 인코딩 기법과 비트플레인 인코딩 기법의 코딩 효율을 비교한 다음, 블록마다 코딩 효율이 더 좋은 인코딩 기법을 선택적으로 적용하여 부호화를 수행할 수 있는데, 이에 대해서는 후술한다. 이러한 본 발명의 실시예가 깊이정보 맵 영상에 대하여 비트플레인 인코딩 기법만을 적용하는 것을 배제하지 않는다는 것은 자명하다.

[0045] 그리고 비트플레인 인코딩 기법을 적용하여 부호화를 수행하는 블록에 대해서는, 비트플레인에 따라서 XOR 연산(그레이코딩)을 적응적으로 수행한 후에 비트플레인을 부호화한다. 보다 구체적으로, 일부 레벨의 비트에 대한 비트플레인은 XOR 연산을 수행하지 않고 그대로 부호화를 수행하지만, 다른 일부 레벨의 비트에 대한 비트플레인은 XOR 연산을 먼저 수행한 이후에 부호화를 수행한다. 물론, 이러한 적응적 XOR 연산을 이용하는 비트플레인 인코딩 기법이 모든 레벨의 비트에 대한 비트플레인에 대해서 전혀 XOR 연산을 수행하지 않거나 또는 모두 XOR 연산을 수행하는 것을 배제하는 것이 아니라는 것은 자명하다.

[0046] 도 8은 본 발명의 일 실시예에 따른 적응적 XOR 연산을 이용하는 깊이정보 맵의 비트플레인들에 대한 비트플레인 인코딩 방법의 일례를 보여 주는 흐름도로서, 인트라 픽쳐(intra picture)에 대한 인코딩 방법의 일례이다.

[0047] 도 8을 참조하면, 우선 입력된 깊이정보 맵 블록에 대하여 비트플레인 분리를 수행한다(10). 수행 결과, MSB에 대한 비트플레인 블록부터 LSB에 대한 비트플레이 블록까지 복수(예컨대, 8개)의 비트플레인 블록이 구해진다.

[0048] 그리고 각각의 비트플레인 블록에 대하여 적응적으로 XOR 연산을 수행한다(11). 보다 구체적으로, 특정 레벨의 비트에 대한 비트플레인 블록에 대해서만 XOR 연산을 수행하며, 나머지 레벨의 비트에 대해서는 XOR 연산을 수행하지 않는다. 어떤 레벨의 비트에 대하여 XOR 연산을 수행할지는 코딩 효율을 고려하여 결정할 수 있다. 예를 들어, XOR 연산을 수행하지 않고 비트플레인 블록을 그대로 코딩할 경우에 발생하는 비트양과 XOR 연산을 수행한 후의 비트플레인 블록을 코딩할 경우에 발생되는 비트양을 비교한 다음, 후자의 경우에 발생되는 비트양이 전자의 경우보다 더 적은 경우에만 해당 레벨의 비트플레인 블록에 대해서 XOR 연산을 수행할 수 있다. 그리고 XOR 연산을 수행하는지 여부를 지시하는 정보(예컨대, 'XOR 플래그(xor_flag)'를 이용하여 나타낼 수 있는데, 이것은 예시적인 것이다)는 각 레벨의 비트플레인 블록에 대한 추가 정보로써 비트스트림에 포함되어 디코딩단

으로 전송된다.

[0049] 현재 비트플레인과 참조 비트플레인 사이에 XOR 연산을 수행한다는 것은, 두 비트플레인의 대응되는 픽셀들 사이에서 XOR 연산을 수행하는 것을 의미한다. 이 경우에, 참조 비트플레인으로는 현재 비트플레인의 바로 상위 레벨의 비트플레인 블록이 이용되거나 또는 바로 하위 레벨의 비트플레인 블록이 이용될 수 있다. 또는, 현재 비트플레인이외의 비트플레인 블록 중에서 선택된 특정 레벨의 비트에 대한 비트플레인 블록이 참조 비트플레인으로 이용되거나 또는 현재 비트플레인 블록과 특정한 여러 개의 비트플레인 블록들에 대하여 XOR 연산을 수행하여 합친 혼합 비트플레인 블록이 참조 비트플레인으로 이용될 수도 있다.

[0050] 계속해서, 본 발명의 실시예에 따른 적응적 XOR 연산을 이용한 비트플레인 코딩 방법에서는 XOR 연산이 수행되거나 또는 수행되지 않은 비트플레인 블록에 대해서 순차적으로 비트플레인 코딩을 수행한다. 예를 들어, MSB 비트플레인 블록부터 LSB 비트플레인 블록의 순으로 비트플레인 코딩을 수행할 수 있다. 이 경우에, 모든 비트플레인 블록에 대하여 비트플레인 코딩을 수행할 필요는 없으며, 발생 비트양의 조절이 필요한 경우에는 MSB 비트플레인 블록부터 상위 일부의 비트에 대한 비트플레인 블록에 대해서만 비트플레인 코딩을 수행하거나 또는 각각의 비트플레인 블록에 대하여 다운샘플링을 수행한 다음에 비트플레인 코딩을 수행할 수도 있다. 그러나 본 실시예가 여기에만 한정되는 것은 아니며, 단계 10을 수행한 이후에 임의로 상위 일부의 비트에 대한 비트플레인 블록에 대해서만 단계 11 이후의 과정을 수행함으로써, 비트플레인 코딩으로 발생되는 비트양을 조절할 수도 있다.

[0051] 인트라 픽처에 대한 비트플레인 코딩 과정에서는, 예를 들어 해당 비트플레인 블록 내의 모든 이진영상 값이 동일한지에 따라서 적응적으로 부호화를 수행할 수 있다. 이를 위하여, 먼저 비트플레인 블록 내의 모든 이진영상 값이 동일한지를 판단한다(12). 판단 결과, 블록 내의 모든 이진영상 값이 동일한 경우, 예컨대 모두 '0(0)'이거나 또는 모두 '255(1)'인 경우에는, 이를 지시하는 비트플레인의 '블록 모드 정보(block mode information)'(예컨대, 'all_0' 모드 또는 'all_1' 모드)를 지시하는 값만을 부호화한다(13). 반면, 블록 내의 모든 이진영상 값이 동일하지 않은 경우에는, 현재 비트플레인 블록을 소정의 '이진영상 압축 방법', 예컨대 인트라 컨텍스트 기반 산술 인코딩(Intra Context-based Arithmetic Encoding, intraCAE) 기법(이에 대해서는 후술한다)으로 부호화를 수행할 수 있다(14). 그리고 이 경우에는 현재 비트플레인 블록의 모드는 'intraCAE 모드'로 설정할 수 있다.

[0052] 도 9는 본 발명의 다른 실시예에 따른 적응적 XOR 연산을 이용하는 깊이정보 맵의 비트플레인들에 대한 비트플레인 인코딩 방법의 일례를 보여 주는 흐름도로서, 인터 픽처(inter picture)에 대한 인코딩 방법의 일례이다.

[0053] 도 9를 참조하면, 도 8을 참조하여 설명한 것(단계 10 및 11)과 마찬가지로, 입력된 깊이정보 맵 블록에 대하여 비트플레인 분리를 수행한다(20). 수행 결과, MSB에 대한 비트플레인 블록부터 LSB에 대한 비트플레이 블록까지 복수(예컨대, 8개)의 비트플레인 블록이 구해진다. 그리고 비트플레인 블록들의 전부 또는 일부에 대하여 적응적으로 XOR 연산을 수행한다(21). 어떤 레벨의 비트에 대하여 XOR 연산을 수행할지는 코딩 효율을 고려하여 결정할 수 있다. 그리고 XOR 연산을 수행하는지 여부를 지시하는 정보(예컨대, 'XOR 플래그(xor_flag)'를 이용하여 나타낼 수 있는데, 이것은 예시적인 것이다)는 각 레벨의 비트플레인 블록에 대한 추가 정보로써 비트스트림에 포함되어 디코딩단으로 전송된다.

[0054] 계속해서, XOR 연산이 수행되거나 또는 수행되지 않은 비트플레인 블록에 대해서 순차적으로 비트플레인 코딩을 수행한다. 예를 들어, MSB 비트플레인 블록부터 LSB 비트플레인 블록의 순으로 비트플레인 코딩을 수행할 수 있다. 이 경우에, 모든 비트플레인 블록에 대하여 비트플레인 코딩을 수행할 필요는 없으며, 발생 비트양의 조절이 필요한 경우에는 MSB 비트플레인 블록부터 상위 일부의 비트에 대한 비트플레인 블록에 대해서만 비트플레인 코딩을 수행하거나 또는 각각의 비트플레인 블록에 대하여 다운샘플링을 수행한 다음에 비트플레인 코딩을 수행할 수도 있다.

[0055] 인터 픽처에 대한 비트플레인 코딩 과정에서는 다음과 같은 방법을 이용하여 효율적으로 부호화를 수행할 수 있다. 다만, 후술하는 단계 22부터 30까지의 과정은 예시적인 것이며, 본 실시예가 여기에만 한정되는 것은 아니다. 예를 들어, 도 9에 도시된 흐름도에서 움직임 벡터의 예측값(MVp)을 이용하여 모드를 결정하는 단계(22)는 생략되고, 단계 21 이후에 바로 단계 24가 수행될 수도 있다.

[0056] 먼저, 현재 비트플레인 블록의 움직임 벡터(MV)의 예측값(MVp)에 대응하는 참조 비트플레인 블록과 현재 비트플레인 블록 사이의 오차(Aerr)를 구한 다음, 위 오차(Aerr)가 소정의 허용 오차(Berr) 범위 이내인지를 판단한다(S22). 현재 비트플레인 블록의 움직임 벡터(MV)의 예측값(MVp)를 구하는 방법에는 아무런 제한이 없다. 그리고

허용 오차(Berr)의 값에도 특별한 제한은 없다. 판단 결과, 오차(Aerr)가 허용 오차(Berr)와 동일하거나 또는 이보다 작은 경우에는, 해당 비트플레인 블록에 대해서는 이를 지시하는 소정의 블록 모드(예컨대, 'No Update Without MV'모드)로 설정하고(23), 이 블록 모드 정보('No Update Without MV'모드를 지시하는 소정의 값)만을 부호화한다(31).

[0057] 반면, 단계 23에서의 판단 결과, 오차(Aerr)가 허용 오차(Berr) 보다 큰 경우에는, 현재 비트플레인 블록 내의 모든 이진영상 값이 동일한지에 따라서 적응적으로 부호화를 수행할 수 있다. 이를 위하여, 비트플레인 블록 내의 모든 이진영상 값이 동일한지를 판단한다(24). 판단 결과, 블록 내의 모든 이진영상 값이 동일한 경우, 예컨대 모두 '0(0)'이거나 또는 모두 '255(1)'인 경우에는, 해당 비트플레인 블록에 대해서는 이를 지시하는 소정의 블록 모드(예컨대, 'all_0' 모드 또는 'all_1' 모드)로 설정하고(25), 이 블록 모드 정보('all_0'모드 또는 'all_1' 모드를 지시하는 소정의 값)만을 부호화한다(31).

[0058] 반면, 단계 24에서의 판단 결과, 현재 비트플레인 블록 내의 모든 픽셀값이 동일하지 않은 경우에는, 현재 비트플레인 블록에 대한 움직임 예측을 수행한다(26). 그리고 움직임 예측 단계(26)에서 계산한 움직임 벡터(MV)에 대응하는 참조 비트플레인 블록과 현재 비트플레인 블록 사이의 오차(Cerr)를 구한 다음, 이 오차(Cerr)가 소정의 허용 오차(Berr) 범위 이내인지를 판단한다(27). 본 실시예에서는 허용 오차(Berr)로 단계 22에서 사용한 허용 오차와 같은 값을 사용하지만, 이것은 예시적인 것이다. 판단 결과, 오차(Cerr)가 허용 오차(Berr)와 동일하거나 또는 이보다 작은 경우에는, 해당 비트플레인 블록에 대해서는 이를 지시하는 소정의 블록 모드(예컨대, 'No Update With MV'모드)로 설정하고(28), 이 블록 모드 정보('No Update With MV'모드를 지시하는 소정의 값)와 움직임 벡터(MV)만을 부호화한다(30).

[0059] 반면, 오차(Cerr)가 허용 오차(Berr) 보다 큰 경우에는, 현재 비트플레인 블록에 대한 오차(Cerr)에 대하여 소정의 '이진영상 압축 방법', 예컨대 콘텍스트 기반 산술 인코딩(CAE) 기법으로 부호화를 수행할 수 있다(29). CAE 인코딩 기법을 이용하여 부호화를 할 경우에, 인트라 CAE(intraCAE) 기법과 인터 CAE(interCAE) 기법 중에서 어느 하나의 기법만을 이용하거나 또는 이 두 가지의 CAE 기법 중에서 코딩 효율이 더 좋은 CAE 기법을 이용하여 적응적으로 부호화를 수행할 수도 있다. 후자의 경우, 예를 들어 인트라 CAE 기법과 인터 CAE 기법을 이용하여 각각 부호화를 수행한 다음, 부호화의 결과 발생하는 비트량이 더 적은 CAE 기법을 이용하여 부호화를 수행하며, 현재 비트플레인 블록의 블록 모드로는 'intraCAE' 모드 또는 'interCAE' 모드로 설정할 수 있다.

[0060] 다음으로 전술한 깊이정보 맵의 특성을 이용하는 본 발명의 일 실시예에 따른 깊이정보 맵의 부호화 장치와 이 부호화 장치에서의 부호화 방법에 관해서 설명한다.

[0061] 도 10은 본 발명의 일 실시예에 따른 깊이정보 맵의 부호화를 위한 장치, 즉 깊이정보 맵 인코더(depth map encoder)의 구성을 보여 주는 블록도이다. 본 실시예에 따른 깊이정보 맵 인코더(DCT 기반 인코딩 유닛은 물론 비트플레인 인코딩 유닛도 해당된다)에서는 입력된 깊이정보 맵을 소정 크기(M×N)의 블록 단위(M×N 블록)로 부호화를 수행한다. 여기서, M×N 블록의 크기에 대해서 특별한 제한은 없으며, 예컨대 16×16 블록의 크기를 갖는 매크로블록이나 그보다 더 큰 크기의 블록이나 또는 더 작은 크기의 블록(예컨대, 8×8 블록이나 4×4 블록 등)일 수 있다.

[0062] 도 10을 참조하면, 본 실시예에 따른 깊이정보 맵 인코더(100)는 DCT 기반 인코딩 유닛(A) 및 비트플레인 인코딩 유닛(B)을 포함한다. DCT 기반 인코딩 유닛(A)은 이미지정보를 부호화하는 기법과 동일한 부호화 기법으로 깊이정보 맵을 블록 단위로 부호화하기 장치의 일례이다. 도시된 바와 같이, DCT 기반 인코딩 유닛(A)은 H.264/AVC(Advanced Video Coding)의 인코딩 유닛과 동일한 구성을 가질 수 있지만, 실시예가 여기에만 한정되는 것은 아니다. 예를 들어, DCT 기반 인코딩 유닛(A)은 H.264/AVC 이외에도 MPEG-1, MPEG-2, MPEG-4 Part 2 Visual, VC(Video Coding)-1 등에서의 인코딩 유닛, H.264/AVC 인코딩 유닛을 응용한 MVC나 SVC(Scalable Video Coding)의 인코딩 유닛, 또는 현재 만들어져 있거나 또는 장래에 만들어질 새로운 동영상 코딩 기법에 따른 인코딩 유닛이 될 수도 있다.

[0063] 반면, 비트플레인 인코딩 유닛(B)은 깊이정보 맵 블록을 비트플레인 블록들로 분리하고, 분리된 각 비트플레인 블록에 대하여 XOR 연산을 적응적으로 적용한 다음, 비트플레인 블록 단위로 부호화를 수행하기 위한 장치의 일례이다. 따라서 본 발명의 실시예가 도시된 비트플레인 인코딩 유닛(B)의 구성으로 한정되는 것으로 해석되어서는 안된다. 예를 들어, 비트플레인 인코딩 유닛(B)의 일부 구성요소(예컨대, 비트율 조절부(142))는 임의적인 구성요소이거나 또는 다른 구성요소에 통합될 수도 있다. 그리고 비트플레인 인코딩 유닛(B)을 구성하는 각각의

구성요소들도 관점에 따라서 다른 블록으로 대체, 통합, 또는 분리될 수도 있다. 이러한 비트플레인 인코딩 유닛(B)의 구체적인 구성과 그 부호화 방법에 대해서는 후술한다.

[0064] 본 실시예에 의하면, 깊이정보 맵 인코더(100)는 DCT 기반 인코딩 유닛(A)과 비트플레인 인코딩 유닛(B) 중에서 부호화 효율이 더 좋은 인코딩 유닛으로 부호화한 다음, 부호화된 데이터를 비트스트림으로 출력한다. 이 비트스트림에는 부호화된 깊이정보 맵 데이터와 함께 이 데이터가 어떤 인코딩 유닛에서 부호화된 것인지를 지시하는 정보(이하, '코딩 모드 정보(coding mode information)'라 하며, 예컨대 bitplane_coding_flag를 이용하여 나타낼 수 있다)도 포함된다. 여기서, 코딩 모드 정보는 부호화된 깊이정보 맵 데이터가 DCT 기반 코딩 모드로 부호화되었다는 것을 지시하는 정보이거나 또는 비트플레인 코딩 모드 부호화되었다는 것을 지시하는 정보인데, 이를 표현하는 방법에는 특별한 제한이 없다. 예를 들어, bitplane_coding_flag를 이용하여 코딩 모드 정보를 표현하는 경우에, 그 값이 '0'이면 DCT 기반 코딩 모드를 지시하고, '1'이면 비트플레인 코딩 모드를 지시할 수 있으며, 반대의 경우도 가능하다.

[0065] 따라서 본 실시예에 따른 깊이정보 맵 인코더(100)는 이러한 코딩 모드를 선택하기 위한 유닛(전기회로 또는 소프트웨어 프로그램 등과 같은 모드 선택 유닛)을 포함한다. 도 11은 깊이정보 맵 인코더(100)의 모드 선택 유닛(도시하지 않음)에서 DCT 기반 코딩(도 10의 A 유닛에서의 인코딩) 모드와 비트플레인 코딩(도 10의 B 유닛에서의 인코딩) 모드 중에서 코딩 효율이 더 좋은 모드를 선택하는 방법의 일례를 보여 주는 흐름도이다.

[0066] 도 11을 참조하면, 깊이정보 맵 인코더(100)는 우선 입력되는 M×N 블록에 대하여 DCT 기반 코딩 모드로 인코딩을 할 경우의 코스트($Cost_A$)와 비트플레인 코딩 모드로 인코딩을 할 경우의 코스트($Cost_B$)를 각각 계산한다(40). 여기서, 코스트($Cost_A$, $Cost_B$)를 계산하는 방법은 특별한 제한이 없다. 예를 들어, H.264/AVC에서 사용 중인 '율-왜곡 최적화 기법(Rate-Distortion(RD) optimization technique)'을 사용하여 코스트($Cost_A$, $Cost_B$)를 계산할 수 있으며, 이에 따른 수식은 수학식 2와 같다.

수학식 2

[0067] $$Cost_i = SSD_i + \lambda_{MODE} \cdot R_i, \quad \lambda_{MODE} = 0.85 \cdot 2^{(QP-12)/3}$$

[0068] 여기서, SSD_i(Sum of Squared Differences)는 원본 영상과 재구성된 영상간의 예측 오차의 제곱의 합을 의미하고, λ_{MODE}는 양자화 파라미터(Quantization Parameter, QP)를 이용하여 생성된 값으로서 매크로블록에 대한 라그랑지 상수(Lagrange constant)를 나타내며, R_i는 해당하는 매크로블록 모드로 실제 코딩을 수행하였을 때 발생되는 비트의 양을 나타낸다.

[0069] 그리고 단계 40에서 계산된 DCT 기반 코딩 모드 코스트($Cost_A$)와 비트플레인 코딩 모드 코스트($Cost_B$)를 비교한다(41). 비교 결과, 전자($Cost_A$)가 더 작은 경우에는 DCT 기반 코딩 모드를 선택하고(42), 반대로 후자($Cost_B$)가 더 작은 경우에는 비트플레인 코딩 모드를 선택한다(43). 따라서 DCT 기반 코딩 모드 코스트($Cost_A$)가 비트플레인 코딩 모드 코스트($Cost_B$)보다 더 작은 경우에는, 입력되는 M×N 블록(또는 부호화의 단위가 되는 소정 크기의 블록)의 코딩 모드 정보는 DCT 기반 코딩 모드가 되며, 비트플레인 코딩 모드 코스트($Cost_B$)가 더 작은 경우에는 입력되는 M×N 블록(또는 부호화의 단위가 되는 소정 크기의 블록)의 코딩 모드 정보는 비트플레인 코딩 모드가 된다. 전술한 바와 같이, 이러한 코딩 모드 정보는 해당 모드로 인코딩된 깊이정보 맵 데이터와 함께 비트스트림에 포함되어 출력된다.

[0070] 계속해서 도 10을 참조하면, 본 발명의 일 실시예에 따른 깊이정보 맵 인코더(100)는 DCT 기반 인코딩 유닛(A)의 일례로 H.264/AVC에 따른 인코딩 유닛을 포함한다. 이하에서는 DCT 기반 인코딩 유닛(A)에서의 부호화 과정의 일례로 H.264/AVC 인코딩 유닛에서의 부호화 과정에 대해서 설명하지만, 이것이 다른 동영상 코덱에 따른 인코딩 유닛이 DCT 기반 인코딩 유닛(A)이 될 수 있다는 것을 배제하는 것은 아니라는 것은 전술한 바와 같다.

[0071] DCT 기반 인코딩 유닛(A)으로 입력되는 데이터는 깊이정보 맵(예컨대, 영상 포맷 4:0:0)으로서, M×N 크기 블록, 예컨대 16×16 픽셀 크기의 매크로블록 단위로 입력될 수 있다. 그리고 DCT 기반 인코딩 유닛(A)에서는 인트라 모드(intra mode) 또는 인터 모드(inter mode)로 부호화가 수행된다. 전자의 경우에 DCT 기반 인코딩 유닛(A)의 내부 스위치가 인트라로 연결이 되지만, 후자의 경우에는 상기 스위치가 인터로 연결이 된다. H.264/AVC에 따른 DCT 기반 인코딩 유닛(A)에서는 우선 입력된 매크로블록에 대한 예측 블록을 생성한다. 그리

고 입력된 매크로블록과 예측 블록과의 차분(difference)을 구한 다음에 이 차분들로 이루어진 잔여 데이터(residual data)를 부호화하는데, 이를 간단히 설명하면 다음과 같다.

[0072] (1) 예측 블록을 생성하는 방법은 인트라 모드인지 또는 인터 모드인지에 따라서 다르다. 인트라 모드일 경우에, 인트라 예측부(106)는 현재 매크로블록에서 이미 코딩된 주변 픽셀의 값을 이용하여 공간적 예측을 수행함으로써 예측 블록을 생성한다. 반면, 인터 모드일 경우에는, 우선 움직임 예측부(102)는 영성 버퍼(130)의 참조영상 버퍼에 저장되어 있는 참조영상에서 현재 입력된 블록과 가장 매치가 잘 되는 영역을 찾아서 움직임 벡터(Motion Vector)를 구하는 움직임 예측 과정을 수행한다. 그리고 움직임 보상부(104)는 상기 움직임 벡터를 이용하여 참조영상 버퍼에 저장되어 있는 참조영상에서 예측 블록을 가져오는 움직임 보상 과정을 수행하여 예측 블록을 생성한다.

[0073] (2) 예측 블록이 생성되면, 제1 가산기(108)는 예측 블록과 현재 블록과의 차분(difference)을 구하여 잔여 블록(residual block)을 생성한다. 생성된 잔여 블록을 입력받은 변환부(110)는 변환(transform) 과정을 수행하여 변환 계수(transform coefficient)를 출력한다. 그리고 변환 계수는 양자화부(112)로 입력되며, 양자화부(112)는 양자화 파라미터(Quantization Parameter, QP)에 따라 양자화 과정을 수행하여 양자화된 계수(Quantitized Coefficient)를 출력한다. 양자화된 계수에 대해서는 엔트로피 코딩부(114)에서 확률 분포에 따른 엔트로피 코딩(entropy coding) 과정이 수행된다.

[0074] (3) 엔트로피 코딩부(114)에서 엔트로피 코딩을 수행하여 출력되는 부호화된 데이터는 멀티플렉서(multiplexer, 160)로 입력되며, 멀티플레서(160)는 다른 정보/데이터와 함께 부호화된 데이터를 비트스트림으로 출력한다.

[0075] (4) 한편, H.264/AVC에 따른 DCT 기반 인코딩 유닛(A)은 인코딩된 현재 영상을 다시 디코딩하여 영상 버퍼(130)에 저장한다. 이것은 현재 영상은 이후에 입력되는 영상을 인코딩하는데 있어서 참조영상으로 사용되어야 하기 때문이다. 이를 위하여, 역양자화부(116)는 양자화된 계수에 대하여 역양자화 과정을 수행하며, 역변환부(118)는 역양자화 과정의 결과로 생성된 역양자화 계수에 대하여 역변환 과정을 수행하여 잔여 블록을 생성한다. 생성된 잔여 블록은 제2 가산기(120)에서 예측 블록과 더해지며, 그 결과 재구성된 블록이 출력된다. 제2 가산기(120)로부터 출력되는 재구성된 블록은 인트라 예측부(106)에서의 인트라 예측 과정이나 또는 비트플레인 인코딩 유닛(B)에서 참조 블록으로 사용될 수 있다. 또한, 상기 재구성된 블록은 디블록킹 필터(122)로 입력되어서 인코딩 과정에서 발생한 블록킹 현상(blocking artifact)을 제거한 다음에, 영상 버퍼(130)의 재구성된 영상 버퍼에 저장된다.

[0076] 계속해서 도 10을 참조하면, 본 발명의 일 실시예에 따른 깊이정보 맵 인코더(100)는 적응적 XOR 연산을 이용하여 비트플레인 단위로 부호화를 수행하기 위한 비트플레인 인코딩 유닛(B)을 포함한다. 비트플레인 인코딩 유닛(B)으로 입력되는 데이터는 깊이정보 맵(예컨대, 영상 포맷 4:0:0)으로서, M×N 크기의 블록 예컨대, 16×16 픽셀의 크기를 갖는 매크로블록 단위로 입력될 수 있다. 비트플레인 인코딩 유닛(B)에서는 비트플레인 블록들로의 분리(144) 이후에 도 8 또는 도 9를 참조하여 전술한 적응적 XOR 연산을 이용하는 비트플레인 인코딩 방법에 따라서 부호화가 수행될 수 있으며, 또한 비트율 조절(142)을 위한 과정이 추가로 수행될 수 있다. 이하, 비트플레인 인코딩 유닛(B)에서의 절차에 관하여 설명한다.

[0077] (1) 우선, 비트플레인 인코딩 유닛(B)에서 데이터를 처리하는 단위는 다양하게 적용될 수 있다. 예를 들어, 입력되는 M×N 블록을 작은 블록으로 나누어서 서브 블록들로 구성한 다음 서브 블록별로 적응적으로 부호화를 수행할 수 있다. 또는, 입력되는 M×N 블록을 더 큰 블록으로 합친 다음에 부호화를 수행하거나 또는 이 합친 블록을 서브 블록들로 나눈 다음에 부호화를 수행할 수도 있다.

[0078] (2) 비트율 조절부(142)는 비트플레인 인코딩 유닛(B)로부터 출력되는 부호화된 데이터의 양, 즉 비트율을 조절하기 위한 것이다. 비트율 조절부(142)는 출력되는 부호화된 데이터의 양에 대한 조절이 필요한 경우에만 동작하므로, 임의적인 구성요소이다. 또는, 비트율 조절부(142)에서 별도로 비트율을 조절하지 않고, 비트플레인 인코딩부(148)에서 사용되는 부호화 알고리즘을 적절히 이용하거나 또는 비트플레인 인코딩부(148)에서의 부호화 시에 다운샘플링 등의 방법을 이용함으로써 비트율을 조절할 수도 있다.

[0079] 비트율 조절부(142)에서 비트율을 조절하는 방법은 특별한 제한이 없다. 예를 들어, 깊이정보 맵이 N비트로 표현되는 경우에, 비트율 조절부(142)는 부호화의 대상이 되는 비트플레인 블록을 일부 비트(예컨대, 상위 n(≤N)개의 비트)만으로 제한함으로써 비트율을 조절할 수 있다. 이를 위하여, 비트율 조절부(142)는 입력되는 N비트로 표현되는 깊이정보 맵 블록에서 하위 (N-n)비트를 삭제하고 상위 n비트로 표현되는 깊이정보 맵 블록을 출력할 수 있다.

[0080] 이 경우에, 부호화의 대상이 되는 비트플레인을 몇 개의 비트에 대한 것으로 할지는 임의로 결정되거나 또는 비트율에 영향을 미치는 다른 정보, 예컨대 DCT기반 인코딩 유닛(A)에서의 양자화 파리미터(QP)값을 기초로 하여 결정이 될 수도 있다. 그리고 비트율 조절부(142)는 삭제되는 하위 (N-n)비트에 대한 복원 정보(예컨대, 디코딩 시에 모두 '0(0)'으로 복원할 것인지 또는 모두 '255(1)'로 할 것인지에 관한 정보나 또는 다른 비트의 값을 이용하여 패딩(padding)하는 방법에 관한 정보 등)도 추가로 생성하여 멀티플렉서(160)로 출력할 수도 있다.

[0081] 도 12는 비트율 조절부(142)에서 양자화 파라미터(QP)에 기초하여 비트율을 결정하는 방법의 일례를 보여 주는 흐름도이다. 도 12에서 코딩할 비트플레인의 수를 결정하는 양자화 파라미터(QP)는 H.264/AVC에서 사용하고 있는 방법이 이용되었는데, 본 실시예가 여기에만 한정되는 것은 아니다.

[0082] 도 12를 참조하면, 우선 양자화 파라미터(QP)값이 소정의 값, 예컨대 22이하인지를 판단한다(50). 판단 결과, 양자화 파라미터(QP)의 값이 22이하인 경우에는, 비트플레인 코딩 유닛(B)은 전부 또는 일부의 비트플레인(예컨대, 상위 7개 비트플레인(MSB, MSB-1, MSB-2, MSB-3, MSB-4, MSB-5, MSB-6 비트플레인)) 블록만 코딩을 수행한다(51). 만일, 양자화 파라미터(QP)의 값이 22보다 큰 경우에는, 양자화 파라미터(QP)의 값이 소정의 값, 예컨대 32이하인지를 판단한다(52). 판단 결과, 양자화 파라미터(QP)의 값이 23보다 크고 32이하인 경우에는, 단계 31보다 적은 개수의 비트플레인(예컨대, 상위 6개 비트플레인(MSB, MSB-1, MSB-2, MSB-3, MSB-4, MSB-5 비트플레인)) 블록만 코딩을 수행한다(53). 반면, 양자화 파라미터(QP)의 값이 32보다 큰 경우에는, 단계 33보다 적은 개수의 비트플레인(예컨대, 상위 5개 비트플레인(MSB, MSB-1, MSB-2, MSB-3, MSB-4 비트플레인)) 블록만 코딩을 수행한다(54).

[0083] (2) 비트플레인 분리부(144)는 n비트로 표현되는 깊이정보 맵 블록을 입력 받아서, n개의 비트플레인 블록으로 분리한다. 그리고 비트플레인 분리부(144)는 분리된 n개의 비트플레인 블록을 XOR 연산부(146) 및/또는 비트플레인 인코딩부(148)로 출력한다. 도 10에서 비트플레인 분리부(144)는 비트율 조절부(142)와 분리되어 도시되어 있지만, 이것은 단지 설명의 편의를 위한 것이라는 점에 유의해야 한다. 예를 들어, 비트율 조절부(142)와 비트플레인 분리부(144)는 하나의 구성요소로 통합되거나 또는 입력되는 깊이정보 맵 블록을 비트플레인 분리부(144)에서 먼저 N개의 비트플레인 블록들로 분리한 후에, 비트율 조절부(142)에서 XOR 연산부(146) 및/또는 비트플레인 인코딩부(148)로 출력되는 비트플레인 블록의 개수를 n개로 한정할 수도 있다.

[0084] (3) XOR 연산부(146)는 비트플레인 분리부(144)로부터 출력되는 n개의 비트플레인 블록들 중에서 전부 또는 일부의 비트플레인 블록에 대해서만 XOR 연산을 수행한다. 예를 들어, 도 8 또는 도 9를 참조하여 설명한 바와 같이, n개의 비트플레인 블록들 중에서 어떤 비트플레인 블록에 대해서 XOR 연산을 수행할지는 원래의 비트플레인 블록을 코딩할 경우에 발생하는 비트양과 XOR 연산을 수행한 후에 비트플레인 블록을 코딩할 경우에 발생하는 비트양을 비교함으로써 결정할 수 있는데, 본 실시예가 여기에만 한정되는 것은 아니다. 그리고 XOR 연산부(146)는 해당 비트플레인 블록에 대하여 XOR 연산이 수행되었는지를 지시하는 소정의 정보(예컨대, 'XOR 플래그' 등과 같은 XOR 연산 정보)를 생성하여 멀티플렉서(160)로 전달하며, 이 정보는 비트스트림에 포함되어 디코딩단으로 전송된다.

[0085] 도 13은 XOR 연산부(146)에서 XOR 연산을 수행할지를 결정하는 방법의 일례를 보여 주는 흐름도이다. 도 13을 참조하면, 우선 현재 입력된 비트플레인 블록을 그대로 비트플레인 코딩하여 발생하는 비트양(Amount Of Bits, AOB_A)과 참조 비트플레인과 XOR 연산을 수행한 이후에 비트플레인 코딩하여 발생하는 비트양(AOB_B)을 구한다(60). 그리고 상기 두 개의 발생 비트양(AOB_A, AOB_B)을 비교한다(61). 비교 결과, 비트플레인 블록을 그대로 비트플레인 코딩하여 발생하는 비트양(Amount Of Bits, AOB_A)이 더 많은 경우에만, 해당 비트플레인 블록에 대하여 XOR 연산을 수행한다(62). 따라서 참조 비트플레인과 XOR 연산을 수행한 이후에 비트플레인 코딩하여 발생하는 비트양(AOB_B)이 더 많은 경우에는, 해당 비트플레인 블록은 XOR 연산이 수행되지 않고서 그대로 비트플레인 인코딩부(148)로 입력된다. 양자가 같은 경우에 XOR 연산을 수행할지 여부는 임의로 결정될 수 있다.

[0086] 그리고 XOR 연산부(146)에서 XOR 연산을 수행할 경우에, 한 단계 상위 레벨의 비트플레인 블록을 참조 비트플레인으로 사용하거나 또는 한 단계 하위 레벨의 비트플레인 블록을 참조 비트플레인으로 사용하는 등, 참조 비트플레인을 결정하는 방법에도 특별한 제한이 없다. 예를 들어, 참조 비트플레인이 한 단계 상위 레벨의 비트플레인 블록인 경우에, XOR 연산부(146)는 현재 코딩할 비트플레인 블록과 한 단계 상위 레벨의 비트플레인 블록(즉, 직전에 코딩된 비트플레인 블록) 사이에서 픽셀마다 XOR 연산을 수행할 수 있다.

[0087] 비트플레인 인코딩부(148)는 소정의 부호화 알고리즘을 이용하여 현재 비트플레인 블록에 대하여 부호화를 수행한다. 여기서, 현재 비트플레인 블록은 깊이정보 맵 블록에서 코딩을 수행할 비트플레인 블록을 의미하는데,

XOR 연산부(146)에서 XOR 연산이 수행된 비트플레인 블록이거나 또는 XOR 연산부(146)를 거치지 않고 비트플레인 분리부(144)로부터 바로 입력되는 비트플레인 블록일 수 있다. 비트플레인 인코딩부(148)는 현재 영상이 인트라 픽쳐인지 또는 인터 픽쳐인지에 따라서 다른 방법으로 부호화를 수행하거나 또는 인트라 픽쳐나 인터 픽쳐 등의 종류에 상관없이 어느 하나의 방법만을 이용하여 부호화를 수행할 수도 있다. 이하, 전자의 경우에 대해서 설명한다.

[0088] 만일, 현재 영상이 인트라 픽쳐라면 도 8를 참조하여 전술한 방법에 따라서 부호화를 수행할 수 있다. 보다 구체적으로, 비트플레인 인코딩부(148)는 현재 비트플레인 블록의 픽셀값이 모두 '0(0)'또는 '255(1)'인 경우와 '0(0)'과 '255(1)'이 섞여 있는 경우를 구분하여, 인코딩을 수행할 수 있다. 예를 들어, 전자의 경우에는 비트플레인 블록 모드 정보만을 코딩할 수 있다. 반면, 후자의 경우에는 비트플레인 인코딩부(148)는 이진 영상의 인코딩에 적합한 소정의 부호화 알고리즘에 따라서 현재 비트플레인 블록에 대한 인코딩을 수행할 수 있다. 비트플레인 인코딩부(148)에서 인코딩된 결과(인코딩된 데이터 및 비트플레인 블록 모드 정보)는 멀티플렉서(160)로 출력된다.

[0089] 도 14는 인트라 픽쳐를 위한 비트플레인 인코딩부(148)의 세부 구성의 일례를 보여 주는 블록도이다. 도 14의 비트플레인 인코딩부(148)는 도 8을 참조하여 설명한 인코딩 방법을 구현하기 위한 장치의 일례일 수 있다. 그리고 도 14에 따른 비트플레인 인코딩부(148)에서 인트라 픽쳐의 비트플레인 블록을 인코딩하는 방법은 국제 동영상 표준인 MPEG-4 Part-2 Visual(ISO/IEC 14496-2)의 이진 형상 코딩(Binary Shape Coding)에서 사용하는 방법을 응용한 것으로서, 이하에서는 이에 대해서 구체적으로 설명한다. 하지만, 비트플레인 인코딩부(148)의 구성이나 비트플레인 인코딩부(148)에서 사용하는 인트라 픽쳐에 대한 인코딩 알고리즘이 여기에만 한정되는 것으로 해석되어서는 안된다.

[0090] 도 14를 참조하면, 모드 결정부(1480)는 입력되는 현재 비트플레인 블록의 모드를 결정하기 위한 것이다. 예를 들어, 현재 비트플레인 블록의 픽셀값이 모두 '255(1)'이라면, 현재 비트플레인 블록의 모드는 이를 지시하는 소정의 블록 모드 정보, 예컨대 'all_1 모드'로 설정되고, 픽셀값이 모두 '0(0)'이라면 이를 지시하는 소정의 블록 모드 정보, 예컨대 'all_0 모드'로 설정될 수 있다. 이 경우에는 현재 비트플레인 블록에 대해서는 블록 모드 정보만이 인코딩되어 멀티플렉서(160)로 입력되어 비트스트림에 포함된다.

[0091] 그리고 현재 비트플레인 블록의 픽셀값이 모두 같은 값이 아닌 경우에는, 현재 비트플레인 블록의 블록 모드 정보는 해당 비트플레인 블록을 부호화하는데 사용되는 소정의 '이진영상 코딩 기법'을 지시하는 값으로 설정될 수 있다. 예를 들어, 컨텍스트 기반 산술 코딩(Context-based Arithmetic Encoding, CAE) 방법을 이용하여 '0(0)'과 '255(1)'이 섞여 있는 현재 비트플레인 블록을 코딩하는 경우에는, 현재 비트플레인 블록의 블록 모드 정보는 'intraCAE 모드'를 지시하는 값으로 설정될 수 있다. 이 경우에, 현재 비트플레인 블록은 CAE 인코딩부(1486)에서 CAE 방법을 사용하여 인코딩되며, CAE 인코딩된 결과(부호화된 데이터)는 블록 모드 정보와 함께 멀티플렉서(160)로 입력되어 비트스트림에 포함된다.

[0092] CAE 인코딩부(1486)에서는 공지된 CAE 인코딩 방법을 사용하여 부호화를 수행할 수 있다. 또는, CAE 인코딩부(1486)에서는 깊이정보 맵의 비트플레인의 특성을 고려하여 공지된 CAE 인코딩 방법을 적절히 응용하여 부호화를 수행할 수 있다. CAE 인코딩을 위해서는 참조 비트플레인이 필요한데, 상기 참조 비트플레인은 비트플레인 분리부(1482) 및/또는 적응적으로 XOR 연산부(1484)를 거친 참조영상의 비트플레인 블록으로부터 생성된다. 참조영상은 DCT 기반 인코딩 유닛(A, 도 10 참조)에서 디코딩되어 디블록킹 필터(122)로 입력되기 이전의 영상이거나 혹은 비트플레인 코딩 유닛(B, 도 10 참조)에서 디코딩된 영상일 수 있다. 그리고 이 참조영상은 비트플레인 분리부(1482)로 입력되어 블록 단위로 복수의 비트플레인 블록들로 분리된다. 또한, 분리된 비트플레인 블록들 중에서 현재 비트플레인 블록과 동일한 레벨의 비트플레인 블록은, 상기 현재 비트플레인 블록이 XOR 연산부(146)에서 XOR 연산이 수행된 블록인지에 따라서 XOR 연산부(1484)에서 적응적으로 XOR 연산이 수행된 다음, 참조 비트플레인을 구성하는 비트플레인 블록으로서 CAE 인코딩부(1486)로 입력된다.

[0093] 도 15는 인터 픽쳐를 위한 비트플레인 인코딩부(148)의 세부 구성의 일례를 보여 주는 블록도이다. 도 15의 비트플레인 인코딩부(148)는 도 9를 참조하여 설명한 인코딩 방법을 구현하기 위한 장치의 일례일 수 있다. 그리고 도 15에 따른 비트플레인 인코딩부(148)에서 인터 픽쳐의 비트플레인 블록을 인코딩하는 방법도 국제 동영상 표준인 MPEG-4 Part-2 Visual(ISO/IEC 14496-2)의 이진 형상 코딩(Binary Shape Coding)에서 사용하는 방법을 응용한 것으로서, 이하에서는 이에 대해서 구체적으로 설명한다. 하지만, 비트플레인 인코딩부(148)의 구성이나 비트플레인 인코딩부(148)에서 사용하는 인터 픽쳐에 대한 인코딩 알고리즘이 여기에만 한정되는 것으로 해석되어서는 안된다.

[0094] 도 15를 참조하면, 모드 결정부(148a)는 입력되는 현재 비트플레인 블록의 모드를 결정하기 위한 것이다. 예를 들어, 도 9를 참조하여 설명한 바와 같이, 비트플레인 분리부(148b)와 적응적으로 XOR 연산부(148c)를 거친 재구성된 영상 및/또는 참조 영상의 비트플레인 블록을 이용하여, 입력되는 현재 비트플레인 블록의 모드를 'No Update Without MV' 모드, 'all_1' 모드, 'all_0' 모드, 'No Update with MV' 모드, 'intraCAE 인코딩' 모드, 및/또는 'interCAE 인코딩' 모드 등으로 결정할 수 있다.

[0095] 만일 현재 비트플레인 블록의 모드가 'No Update Without MV' 모드, 'all_1' 모드, 또는 'all_0' 모드로 결정되는 경우에는, 블록 모드 정보만이 인코딩되어 멀티플렉서(160)로 입력될 수 있다. 이 경우에 모드 결정부(148a)에서 도 9의 단계 22 내지 단계 25에서 기술된 알고리즘을 수행하여, 현재 비트플레인 블록의 모드를 결정할 수 있다.

[0096] 그리고 현재 비트플레인 블록의 모드가 'No Update with MV' 모드로 결정되는 경우에는, 블록 모드 정보와 움직임 벡터(MV)가 인코딩되어 멀티플레서(160)로 입력될 수 있다. 이 경우에, 도 9의 단계 26은 움직임 예측부(148d)에서 수행되며, 단계 27과 단계 28은 움직임 보상부(148e)와 모드 결정부(148a)에서 수행될 수 있다.

[0097] 또한, 현재 비트플레인 블록의 모드가 'intraCAE 인코딩' 모드 또는 'interCAE 인코딩' 모드로 결정되는 경우에는, 블록 모드 정보는 인코딩되어 멀티플렉서(160)로 입력되지만, 현재 비트플레인 블록의 데이터는 CAE 인코딩부(148f)로 입력되어 인트라 CAE 방법 또는 인터 CAE 방법을 이용하여 부호화된 후에 멀티플렉서(160)로 입력될 수 있다. 이 경우에, 'intraCAE 인코딩' 모드 또는 'interCAE 인코딩' 모드로 결정하는 것은 모드 결정부(148a)에서 수행하거나 또는 CAE 인코딩부(148f)에서 수행할 수 있다. 그리고 CAE 인코딩부(148f)에서는 공지된 인트라 CAE 방법 또는 인터 CAE 방법을 그대로 사용하거나 또는 이를 적절이 응용하여 부호화를 수행할 수 있다. CAE 인코딩부(148f)에서 인트라 CAE 인코딩 및/또는 인터 CAE 인코딩을 수행하기 위해서는 참조 비트플레인이 필요한데, 상기 참조 비트플레인은 비트플레인 분리부(148b) 및/또는 적응적으로 XOR 연산부(148c)를 거친 재구성된 영상(또는 참조 영상)의 비트플레인 블록으로부터 생성된다.

[0098] 도 16은 도 14의 CAE 인코딩부(1486) 또는 도 15의 CAE 인코딩부(148f)에서 CAE 인코딩에서 이용하는 참조 비트플레인을 구성하는 방법의 일례를 보여 주기 위한 도면이다. 도 16에서 현재 비트플레인 블록은 가운데 5개의 블록들 중에서 실선으로 표시된 블록이다. 도 16의 우측 아래쪽을 참조하면, 상기 현재 비트플레인 블록은 MSB-3 비트플레인 블록으로서, XOR 연산부(146, 도 10 참조)에서 XOR 연산을 수행한 블록인 것을 알 수 있다. 그러나 현재 비트플레인 블록이 다른 레벨의 비트플레인 블록이거나 또는 XOR 연산을 수행하지 않은 블록인 경우에도, 도 16에 도시된 방법을 응용하여 참조 비트플레인을 구성할 수가 있다는 것은 당업자에게 자명하다.

[0099] 도 16에 도시된 바와 같이, CAE 인코딩부(1486 또는 148f)에서 MSB-3 비트플레인 블록에 대하여 CAE 인코딩을 수행하기 위해서는, 현재 블록에 인접한 블록들에서 현재 비트플레인 블록과 동일한 레벨(MSB-3 비트)의 참조 비트플레인(도 16에서 가운데 5개의 블록들 중에서 점선으로 표시된 4개의 블록)이 필요하다. 이를 위하여, 비트플레인 분리부(1482 또는 148b)는 참조영상에서 현재 블록에 인접한 블록, 즉 참조 블록들을 비트플레인 블록들로 분리한다. 그리고 코딩할 현재 비트플레인(MSB-3 비트플레인) 블록이 XOR 연산부(146)에서 XOR 연산이 수행된 블록이므로, 상기 참조블록의 MSB-3 비트플레인 블록에 대해서도 XOR 연산부(1484 또는 148c)에서 XOR 연산이 수행된다. 그리고 이와 같이, XOR 연산부(1484 또는 148c)에서 XOR 연산이 수행된 참조블록들 각각의 MSB-3 비트플레인 블록이 모여서 상기 참조 비트플레인을 구성하게 된다.

[0100] 이상에서는 비트플레인 인코딩부(148)에서 CAE 인코딩 알고리즘을 이용하여 비트플레인 블록들을 부호화하는 방법에 관하여 설명하였다. 하지만, 본 발명의 실시예가 여기에만 한정되는 것은 아니며, 비트플레인 인코딩부(148)는 다른 부호화 알고리즘을 이용하여 n개의 비트플레인 블록들을 부호화할 수도 있다. 예를 들어, 비트플레인 인코딩부(148)는 공지된 런 길이 코딩(Run Length Coding, RLC) 방법과 가변 길이 코딩(Variable Length Coding, VLC) 방법을 이용하거나, 또는 공지된 컨텍스트 기반 적응적 이진 산술 코딩(Context-based Adaptive Binary Arithmetic Coding, CABAC)을 응용하거나, 또는 공지된 JBIG(Joint Bi-level Image processing Group)의 방법이나 쿼드 트리(Quad Tree) 방법을 이용하여 비트플레인을 부호화할 수도 있다.

[0101] 계속해서 도 10을 참조하면, 비트플레인 인코딩부(148)로부터 출력되는 정보와 데이터는 멀티플렉서(160)를 거쳐서 비트스트림으로 출력되는 한편, 후속되는 깊이정보 맵의 인코딩을 위하여 비트플레인 블록 단위로 다시 디코딩이 된다. 이를 위하여, 비트플레인 인코딩부(148)의 출력은 비트플레인 디코딩부(150)로 입력되어서 XOR 연산부(152), 비트플레인 결합부(154), 및 깊이정보 맵 구성부(156)를 거쳐서 최종적으로 재구성된 깊이정보 맵 블록이 출력된다. 이하, 이를 보다 상세하게 설명한다.

[0102] 비트플레인 디코딩부(150)에서는 전술한 비트플레인 인코딩부(148)의 과정과는 반대의 과정이 수행된다. 보다 구체적으로, 비트플레인 디코딩부(150)는 비트플레인 인코딩부(148)로부터 출력되는 인코딩된 데이터 및/또는 모드 정보를 이용하여 비트플레인 블록을 재구성한다. 재구성된 비트플레인 블록은 인코딩 단계에서 XOR 연산이 수행된 비트플레인 블록이거나 또는 XOR 연산이 수행되지 않은 비트플레인 블록일 수 있는데, 전자는 XOR 연산부(152)로 출력되지만 후자는 비트플레인 결합부(154)로 출력된다.

[0103] 재구성된 비트플레인 블록을 생성할 수 있도록, 비트플레인 디코딩부(150)는 비트플레인 인코딩부(148)에 대응하는 구성을 갖는다.

[0104] 예를 들어, 비트플레인 인코딩부(148)가 도 14에 도시된 것과 같은 구성을 포함하는 경우에, 비트플레인 디코딩부(150)도, 도 17에 도시된 것과 같은, MPEG-4 Part-2 Visual(ISO/IEC 14496-2)의 인트라 픽쳐에 대한 이진 형상 디코딩법을 응용하는 구성을 포함할 수 있다. 도 17은 이러한 비트플레인 디코딩부(150)의 세부 구성의 일례를 보여 주는 블록도이다.

[0105] 도 17을 참조하면, 디코딩되어야 하는 비트플레인 블록의 모드가 'all_0 모드'이거나 또는 'all_1 모드'인 경우에는, 동일 레벨 블록 디코딩부(1500)가 블록 내의 모든 픽셀값이 '0(0)' 또는 '255(1)'인 비트플레인 블록을 생성한다. 반면, 디코딩되어야 하는 비트플레인 블록의 모드가 'intraCAE 모드'인 경우에는, CAE 디코딩부(1506)가 참조 비트플레인을 이용하여 이진 산술 디코딩(CAE 디코딩)을 수행함으로써 재구성된 비트플레인 블록을 출력한다. 상기 참조 비트플레인은 도 14를 참조하여 설명한 것과 동일한 과정을 거쳐서 만들어질 수 있다. 보다 구체적으로, 입력되는 참조영상은 비트플레인 분리부(1502)에서 복수의 비트플레인 블록들로 분리되며, 분리된 비트플레인 블록은 그대로 CAE 디코딩부(1506)에서 CAE 디코딩에서 이용하는 참조 비트플레인을 구성하거나 또는 상기 분리된 비트플레인 블록에 대하여 XOR 연산부(1504)에서 XOR 연산이 수행된 비트플레인 블록이 상기 참조 비트플레인을 구성할 수 있다. 이 경우에, 비트플레인 분리부(1502)에서 분리된 비트플레인 블록이 XOR 연산부(1504)를 거치는지 여부(즉, XOR 연산이 수행되는지 여부)는, 디코딩의 대상이 되는 현재 비트플레인 블록이 XOR 연산부(146)에서 XOR 연산이 수행된 블록인지에 따라서 결정된다.

[0106] 만일, 비트플레인 인코딩부(148)가 도 15에 도시된 것과 같은 구성을 포함하는 경우에, 비트플레인 디코딩부(150)도 MPEG-4 Part-2 Visual(ISO/IEC 14496-2)의 인터 픽쳐에 대한 이진 형상 디코딩법을 응용하는 구성을 포함할 수 있다. 이러한 비트플레인 디코딩부의 세부 구성의 일례에 대해서는 후술한다(도 19 참조).

[0107] 계속해서 도 10을 참조하면, XOR 연산부(152)는 재구성된 비트플레인 블록에 대하여 XOR 연산을 수행한다. 이 경우에, 모든 재구성된 비트플레인 블록에 대하여 XOR 연산을 수행하는 것이 아니라, XOR 연산부(146)에서 XOR 연산을 수행한 비트플레인 블록에 대해서만 XOR 연산을 수행한다. 현재의 재구성된 비트플레인 블록이 XOR 연산부(146)에서 XOR 연산이 수행된 블록인지는 이를 지시하는 XOR 연산 정보, 예컨대 XOR 플래그로부터 알 수 있다.

[0108] 그리고 비트플레인 결합부(154)는 재구성된 비트플레인 블록들로부터 n비트로 구성된 깊이정보 맵 블록을 복원한다. 상기 재구성된 비트플레인 블록은 비트플레인 디코딩부(150)로부터 직접 출력되는 재구성된 비트플레인 블록이거나 및/또는 XOR 연산부(152)에서 XOR 연산이 수행된 후에 출력되는 비트플레인 블록일 수 있다.

[0109] 또한, 깊이정보 맵 구성부(156)에서는 복원된 n비트의 깊이정보 맵 블록으로부터 N비트의 깊이정보 맵 블록을 생성한다. N비트의 깊이정보 맵 블록을 구성하기 위하여 n비트의 깊이정보 맵에 추가되는 하위 (N-n)비트에 대한 비트플레인 블록은, 픽셀값이 모두 '0(0)'으로 구성된 비트플레인 블록일 수 있다. 즉, 깊이정보 맵 블록을 구성하기 위하여 추가되는 각 픽셀의 하위 (N-n)비트는 모두 '0'의 값으로 함으로써, n비트의 깊이정보 맵 블록에서 N비트의 깊이정보 맵 블록을 복원할 수 있다. 이와 같이, 깊이정보 맵 구성부(156)에서 복원된 깊이정보 맵 블록은 DCT 기반 인코딩 유닛(A)에서 인트라 예측을 위한 참조영상으로 이용되거나, 디블록킹 필터부(122)로 입력되어 디블록킹 필터링 과정이 수행된 다음 영상버퍼(130)의 재구성된 영상 버퍼에 저장되거나, 및/또는 비트플레인 인코딩부(148) 및 비트플레인 디코딩부(150)에서 참조 비트플레인을 구성하기 위하여 이용될 수 있다.

[0110] 도 18은 본 발명의 일 실시예에 따른 깊이정보 맵의 복호화를 위한 장치, 즉 깊이정보 맵 디코더(depth map decoder) 구성을 보여 주는 블록도이다. 본 실시예에 따른 깊이정보 맵 디코더(200, DCT 기반 디코딩 유닛(D)은 물론 비트플레인 디코딩 유닛(C)도 해당된다)는 도 10을 참조하여 설명한 깊이정보 맵 인코더(100)에 대응한다. 따라서 깊이정보 맵 인코더(100)의 구성에 대응하여 깊이정보 맵 디코더(200)의 구성도 변경될 수 있다. 깊이정보 맵 디코더(200)에서도 소정 크기의 블록(M×N 블록) 단위로 복호화를 수행하는데, 여기서, M×N 블록의 크기

에 대해서 특별한 제한은 없다. 이하에서는 도 18을 참조하여, 본 발명의 일 실시예에 따른 블록 기반 깊이정보 맵의 복호화 장치 및 방법에 관하여 설명한다.

[0111] 도 18을 참조하면, 본 실시예에 따른 블록 기반 깊이정보 맵의 복호화 장치, 즉 깊이정보 맵 디코더(200)는 비트플레인 디코딩 유닛(C) 및 DCT 기반 디코딩 유닛(D)을 포함한다. 비트플레인 디코딩 유닛(C)은 도 10의 비트플레인 인코딩 유닛(B)에 대응하는 것으로서, 디멀티플렉서(demultiplexer, 210)로부터 입력되는 비트스트림으로부터 깊이정보 맵을 비트플레인 블록 단위로 재구성하고, 이에 대하여 적응적으로 XOR 연산을 수행하며, 또한 XOR 연산이 수행되거나 및/또는 XOR 연산이 수행되지 않은 재구성된 비트플레인 블록들을 결합함으로써 깊이정보 맵 블록을 재구성한다. 비트플레인 디코딩 유닛(C)의 구체적인 구성 및 동작에 대해서는 후술한다.

[0112] 반면, DCT 기반 디코딩 유닛(D)은 도 10의 DCT 기반 인코딩 유닛(A)에 대응하는 것으로서, 이미지정보를 복호화하는 기존의 기법과 동일한 복호화 기법으로 깊이정보 맵을 블록 단위로 디코딩한다. 이를 위하여, DCT 기반 디코딩 유닛(D)은 H.264/AVC(Advances Video Coding)의 디코딩 유닛과 동일한 구성을 가질 수 있다. 그러나 본 실시예가 여기에만 한정되는 것은 아니며, DCT 기반 인코딩 유닛(A)의 구성에 대응하여 DCT 기반 디코딩 유닛(D)도 MPEG-1, MPEG-2, MPEG-4 Part 2 Visual, VC(Video Coding)-1 등의 디코딩 유닛이나 장래에 규정될 새로운 동영상 코딩 기법에 따른 디코딩 유닛의 구성이 될 수도 있다.

[0113] 도 18을 참조하면, 깊이정보 맵 디코더(200)는 코딩 모드 정보에 따라서 비트플레인 디코딩 유닛(C)과 DCT 기반 디코딩 유닛(D) 중에서 어느 하나의 디코딩 유닛에서 복호화 과정을 수행함으로써 재구성된 깊이정보 맵 블록을 출력한다. 코딩 모드 정보는 비트스트림에 포함되어 있는데, 디멀티플렉서(demultiplexer, 210)는 입력되는 비트스트림에서 코딩 모드 정보를 먼저 디코딩한 다음, 이 코딩 모드 정보에 따라서 비트스트림을 비트플레인 디코딩 유닛(C) 또는 DCT 기반 디코딩 유닛(D)으로 분배한다. 예를 들어, 코딩 모드 정보가 '비트플레인 코딩'을 지시하는 경우에, 디멀티플렉서(210)로부터 출력되는 부호화된 데이터는 비트플레인 디코딩 유닛(C)에서 복호화 과정이 진행되는데, 그 구체적인 방법은 다음과 같다.

[0114] (1) 우선, 비트플레인 디코딩부(212)는 디멀티플렉서(210)로부터 부호화된 데이터를 입력 받아서 소정의 복호화 알고리즘을 수행함으로써, n개의 비트플레인 블록을 복원한다. 여기서, n개의 비트플레인 블록은 N개의 비트플레인 중에서 인코딩 단계에서 원하는 비트율을 얻기 위하여 실제 코딩된 비트플레인 블록의 개수를 가리킨다. 비트플레인 디코딩부(212)는 비트플레인 인코딩부(148, 도 10 참조)에 대응하는 구성을 가진다.

[0115] 예를 들어, 비트플레인 디코딩부(212)는 도 17에 도시된 비트플레인 디코딩부(150)와 동일한 구성의 디코딩부를 포함할 수 있다. 이러한 구성의 비트플레인 디코딩부(212)는 인트라 픽쳐에 대한 CAE 디코딩 기법 및/또는 동일 레벨 블록 디코딩 기법을 이용하여 인트라 픽쳐를 구성하는 블록의 비트플레인 블록을 복원할 수 있다. 또한, 비트플레인 디코딩부(212)는 인터 픽쳐를 구성하는 블록의 비트플레이 블록을 복원하기 위한 디코딩부도 포함할 수 있는데, 도 19는 인터 픽쳐를 위한 비트플레인 디코딩부(212)의 구성의 일례를 보여 주는 블록도이다.

[0116] 도 19를 참조하면, 수신된 비트스트림은 디멀티플렉서(210)에서 디멀티플렉싱되며, 그 결과 현재 비트플레임 블록의 모드 정보, 움직임 벡터(MV), CAE 인코딩된 비트스트림 등이 비트플레인 디코딩부(212)로 입력된다. 비트플레인 디코딩부(212)는 입력되는 데이터 중에서 모드 정보에 기초하여 디코딩을 수행하여, 비트플레인 블록들을 복원한다.

[0117] 만일, 모드 정보가 'No Update Without MV' 모드이면, 움직임 보상부(2126)는 움직임 벡터 예측값(MVp)을 움직임 벡터로 하여 움직임 보상을 수행하며, 그 결과로서 움직임 보상된 비트플레인 블록, 즉 움직임 벡터 예측값(MVp)에 대응하는 참조 비트플레인 블록을 복원된 비트플레인 블록으로 출력한다. 그리고 움직임 벡터 예측값(MVp)은 여러 가지 방법을 이용하여 결정될 수 있는데, 비트플레인 인코딩부(148, 도 10 참조)에서 움직임 벡터 예측값(MVp)을 결정할 때 사용한 방법과 같은 방법을 사용할 수 있다.

[0118] 만일 모드 정보가 'all_0 모드'또는 'all_1 모드'인 경우에는, 동일 레벨 블록 디코딩부(2120)가 블록 내의 모든 픽셀값이 '0(0)' 또는 '255(1)'인 비트플레인 블록을 복원하여 출력한다. 그리고 모드 정보가 'No Update with MV' 모드인 경우에는, 움직임 보상부(2126)는 디멀티플렉서(210)로부터 입력되는 움직임 벡터(MV)를 이용하여 움직임 보상을 수행하며, 그 결과로서 움직임 보상된 비트플레인 블록, 즉 움직임 벡터(MV)에 대응하는 참조 비트플레인 블록을 복원된 비트플레인 블록으로 출력한다. 또한, 모드 정보가 'intraCAE'모드 또는 'interCAE' 모드인 경우에는, CAE 디코딩부(2128)는 이진 산술 디코딩(예컨대, intraCAE 디코딩 또는 interCAE 디코딩)을 수행함으로써 재구성된 비트플레인 블록을 출력할 수 있다.

[0119] 움직임 보상이나 CAE 디코딩을 수행하기 위한 참조 비트플레인은 앞에서 설명에서 한 것과 동일한 과정을 거쳐

서 만들어질 수 있다. 보다 구체적으로, 입력되는 참조영상(또는 재구성된 영상)은 비트플레인 분리부(2122)에서 복수의 비트플레인 블록들로 분리된다. 또한, 분리된 비트플레인 블록들 중에서 현재 비트플레인 블록과 동일한 레벨의 비트플레인 블록은 XOR 연산을 수행하지 않고 그대로 움직임 보상부(2126) 또는 CAE 디코딩부(2128)로 입력되어 참조 비트플레인으로 이용되거나 또는 상기 비트플레인 블록에 대하여 XOR 연산부(2124)에서 XOR 연산이 수행된 비트플레인 블록이 참조 비트플레인으로 이용될 수 있다. 이 경우에, 비트플레인 분리부(2122)에서 분리된 비트플레인 블록이 XOR 연산부(2124)를 거치는지 여부(즉, XOR 연산이 수행되는지 여부)는, 디코딩의 대상이 되는 현재 비트플레인 블록이 인코딩 과정에서 XOR 연산이 수행된 후에 인코딩되었는지 또는 XOR 연산이 수행되지 않고서 인코딩되었는지(예컨대, XOR 플래그의 값)에 따라서 결정될 수 있다.

[0120] (2) 계속해서 도 18을 참조하면, XOR 연산부(214)는 비트플레인 디코딩부(212)에서 복원된 비트플레인 블록에 대하여 적응적으로 XOR 연산을 수행한다. 예를 들어, XOR 연산부(214)는 디코딩된 XOR 연산 정보, 예컨대 XOR 플래그의 값에 따라서 해당 비트플레인 블록에 XOR 연산을 수행하거나 또는 수행하지 않을 수 있다. XOR 연산부(214)는 현재 비트플레인 블록보다 상위 레벨의 비트에 대한 비트플레인 블록을 참조 비트플레인으로 이용하여 XOR 연산을 수행할 수 있는데, 여기에만 한정되는 것은 아니다. 즉, XOR 연산부(214)에서의 XOR 연산을 위한 참조 비트플레인은 XOR 연산부(146, 도 10 참조)에서 XOR 연산을 수행할 때 이용한 참조 비트플레인과 동일한 방법으로 결정될 수 있다.

[0121] (3) 비트플레인 결합부(216)는 비트플레인 디코딩부(212)로부터 출력되는 비트플레인 블록 및/또는 XOR 연산부(214)로부터 출력되는 비트플레인 블록을 결합하여, 각 픽셀이 n비트로 표현되는 깊이정보 맵 블록을 생성한다.

[0122] (4) 그리고 깊이정보 맵 구성부(218)는 비트플레인 결합부(216)로부터 출력되는 n비트의 깊이정보 맵 블록으로부터 실제 깊이정보 맵의 비트수(N개)의 깊이정보 맵 블록을 생성한다. 예를 들어, 깊이정보 맵 구성부(218)는 n비트의 깊이정보 맵 블록의 픽셀 각각에 (N-n)개의 하위 비트를 추가함으로써, 즉 (N-n)개의 비트플레인 블록을 하위 레벨의 비트에 대한 비트플레인 블록으로써 추가함으로써, N비트의 깊이정보 맵 블록을 구성할 수 있다. 이 경우에, 추가되는 하위 비트는 미리 결정된 임의의 값, 예컨대 모두 '0'이 될 수 있지만(즉, 추가되는 (N-n)개의 비트플레인 블록들 각각은 모든 픽셀의 값이 '0'이다), 여기에만 한정되는 것은 아니다.

[0123] 또는, 깊이정보 맵 구성부(218)에서는 비트스트림에 포함되어 있는 소정의 정보, 즉 추가되는 (N-n)개의 비트플레인 블록에 대한 정보를 이용할 수도 있다. 예를 들어, 상기 정보에 의하여 지시되는 내용에 따라서 추가되는 (N-n)개의 비트플레인 블록의 픽셀값을 모두 '0(0)'으로 하거나 또는 모두 '255(1)'로 하거나 또는 인접한 비트플레인 블록을 이용하는 패딩법 등의 방법으로 픽셀값을 결정할 수도 있다.

[0124] (5) 마지막으로, 깊이정보 맵 구성부(218)로부터 출력되는 N비트로 표현되는 깊이정보 맵 블록은 DCT 기반 디코딩 유닛(D)의 인트라 예측부(228)에서 인트라 예측을 위한 참조 블록으로 사용되거나 및/또는 비트플레인 디코딩 유닛(C)의 비트플레인 디코딩부(212)에서 CAE 디코딩을 위한 참조 블록으로 사용될 수 있다. 그리고 상기 N비트의 깊이정보 맵 블록은 디블록킹 필터부(232)를 거친 후에, 영상 버퍼(240)의 재구성된 영상 버퍼에 저장되거나 및/또는 재구성된 깊이정보 맵 블록으로서 깊이정보 맵 디코더(200)로부터 출력될 수 있다.

[0125] 계속해서 도 18을 참조하면, 코딩 모드 정보가 'DCT 기반 코딩'을 지시하는 경우에, 디멀티플레서(210)로부터 출력되는 부호화된 데이터는 DCT 기반 디코딩 유닛(D)에서 복호화 과정이 진행되는데, 그 구체적인 방법은 다음과 같다. DCT 기반 디코딩 유닛(D)에서는 먼저 예측 블록을 생성한 후에, 이 예측 블록을 입력받은 비트스트림을 디코딩하여 구한 잔여 블록과 더함으로써, 재구성된 깊이정보 맵 블록을 생성한다. 이를 간략히 설명하면 다음과 같다.

[0126] (1) 예측 블록의 생성은 현재 블록의 예측 모드에 따라서 수행된다. 예를 들어, 현재 블록의 예측 모드가 인트라 예측 모드인 경우에, 인트라 예측부(228)는 현재 블록의 이미 디코딩된 주변 픽셀값을 이용하여 공간적 예측을 수행함으로써 예측 블록을 생성한다. 반면, 현재 블록의 예측 모드가 인터 예측 모드인 경우에, 인터 예측부(230)는 움직임 벡터를 이용하여 영상 버퍼(240)의 참조 영상 버퍼에 저장되어 있는 참조 영상에서 해당 영역을 찾아서 움직임 보상을 수행함으로써 예측 블록을 생성한다.

[0127] (2) 엔트로피 디코딩부(220)는 디멀티플렉서(210)로부터 입력되는 비트스트림을 확률 분포에 따라 엔트로피 디코딩을 수행함으로써 양자화된 계수를 출력한다.

[0128] (3) 그리고 역양자화부(222)는 엔트로피 디코딩부(220)로부터 출력되는 양자화된 계수에 대하여 역양자화 과정을 수행하여 변환된 계수를 출력하며, 역변환부(224)는 상기 변환된 계수에 대하여 역변환 과정을 수행함으로써 잔여 블록을 생성한다. 가산기(226)는 역변환부(224)로부터 출력되는 잔여 블록과 인트라 예측부(228) 또는 움

직임 보상부(230)로부터 입력되는 예측 블록을 합산하여, 재구성된 블록을 출력한다. 상기 재구성된 블록은 비트플레인 디코딩 유닛(C)에서 비트플레인 디코딩을 수행하기 위한 참조 블록으로 입력되거나 및/또는 디블록킹 필터부(232)에서 디블록킹 필터링 과정을 수행한 다음에 영상 버퍼(240)의 재구성된 영상 버퍼에 저장되거나 및/또는 재구성된 깊이정보 맵 블록으로 출력된다.

[0129] 전술한 본 발명의 실시예에 따른 블록 기반 깊이정보 맵의 코딩 방법의 효과를 검증하기 위하여, H.264/AVC의 참조 소프트웨어인 JM(Joint Model) 13.2를 이용하여 본 발명의 실시예에 따른 방법과 기존의 방법(H.264/AVC에 따른 인코딩/디코딩 방법)에 따라서 실험(simulation)을 수행하였으며, 실험 조건은 표 1과 같다.

표 1

[0130]

해상도 및 프레임률	1024x768, 15Hz
프레임수	100 Frames
영상포맷	YUV 4:0:0
양자화 파라미터	22, 27, 32, 37
예측구조	I-P-P-P-
엔트로피 코딩 방법	CAVLC

[0131] 도 20과 도 21은 각각 H.264/AVC를 이용한 기존 방법(Anchor)과 본 발명의 실시예에 따른 방법(Proposed)에 대한 비트율 대비 PSNR(Peak Signal-to-Noise Ratio)의 율-왜곡 곡선(Rate-Distortion(RD) curve)을 나타내는 것으로, 도 20은 테스트 영상 시퀀스 중의 하나인 'Breakdancers' 영상 시퀀스에 대한 것이며, 도 21은 테스트 영상 시퀀스 중의 하나인 'Ballet' 영상 시퀀스에 대한 것이다. 도 20 및 도 21을 참조하면, 본 발명의 실시예에 따른 방법에 따른 결과가 기존 방법에 따른 결과보다 성능이 월등하게 향상된 것을 확인할 수 있다. 보다 구체적으로, "Breakdancers" 영상 시퀀스의 깊이정보 맵에서는 평균 PSNR의 향상을 나타내는 BD-PSNR 방법을 사용하여 성능을 비교하였을 때 약 1.2 dB가 향상되었고, 평균 비트율(average bit-rate)의 감소량을 나타내는 BD-rate 방법을 사용하여 성능을 비교하였을 때 비트량이 약 16.8% 감소하였다. 그리고 "Ballet" 영상 시퀀스의 깊이정보 맵에서는 BD-PSNR 방법을 사용하여 성능을 비교하였을 때 약 1.3dB가 향상되었고, BD-rate 방법을 사용하여 성능을 비교하였을 때 비트량이 약 18.7%가 감소하였다.

[0132] 도 22는 본 발명의 실시예에 따라서 블록 기반 깊이정보 맵의 코딩 방법을 사용한 경우의 화질을 보여 주기 위한 것으로서, 도 22의 (a)는 'Breakdancers' 영상 시퀀스의 일부를 보여 주는 원본 깊이정보 맵 영상이고, 도 22의 (b)는 (a)를 기존의 방법(H.264/AVC)으로 인코딩하고 디코딩하여 복원한 깊이정보 맵 영상이며, 도 22의 (c)는 (a)를 본 발명의 실시예에 따라서 인코딩하고 디코딩하여 복원한 깊이정보 맵 영상이다. 도 22를 참조하면, 기존의 방법을 그대로 사용하는 경우에는 복원된 영상, 특히 객체의 경계 부근에서 화질이 열화된다는 것을 쉽게 확인할 수 있다. 반면, 본 발명의 실시예에 의할 경우에는, 기존의 방법에 비하여 객체 경계 부분의 화질이 상당히 향상된다는 것을 확인할 수 있다.

[0133] 이상의 설명은 본 발명의 실시예에 불과할 뿐, 이 실시예에 의하여 본 발명의 기술 사상이 한정되는 것으로 해석되어서는 안된다. 본 발명의 기술 사상은 특허청구범위에 기재된 발명에 의해서만 특정되어야 한다. 따라서 본 발명의 기술 사상을 벗어나지 않는 범위에서 전술한 실시예는 다양한 형태로 변형되어 구현될 수 있다는 것은 당업자에게 자명하다.

도면의 간단한 설명

[0134] 도 1은 3D 비디오 코딩이 적용되는 FTV 시스템에서의 랜더링(rendering)의 일례를 보여 주는 도면이다.

[0135] 도 2의 (a)는 실제 영상의 일례인 MVC의 테스트 시퀀스의 하나인 'Breakdancers' 영상 시퀀스이고, 도 2의 (b)는 이 'Breakdancers' 영상 시퀀스의 깊이정보 맵 영상이다.

[0136] 도 3a 및 도 3b는 각각 도 2의 영상에서 이미지정보의 레벨 분포와 깊이정보의 레벨 분포를 보여주기 위한 그래프이다.

[0137] 도 4a는 객체의 경계를 포함하는 깊이정보 맵 블록에 대한 깊이정보 그래프이고, 도 4b는 배경 부분에서의 깊이

정보 맵 블록에 대한 깊이정보 그래프이다.

[0138] 도 5의 (a)와 (b)는 각각 도 2의 (a)의 실제 영상과 도 2의 (b)의 깊이정보 맵 영상을 비트플레인 단위로 표현한 도면이고, 도 5의 (c)는 (b)의 깊이정보 맵 영상에 대하여 그레이코딩을 수행하여 비트플레인 단위로 도시한 것이다.

[0139] 도 6의 (a)는 도 2의 (b)의 깊이정보 맵의 일부이고, 도 6의 (b), (c), 및 (d)는 각각 도 6의 (a)의 깊이정보 맵에서 임의로 선택된 블록과 이 비트플레인 블록들을 그레이코딩한 비트플레인 블록들을 보여 주는 도면이다.

[0140] 도 7은 인접 레벨의 비트에 대한 비트플레인과 상관관계가 거의 없는 비트플레인 블록의 일례를 보여 주기 위한 도면이다.

[0141] 도 8은 본 발명의 일 실시예에 따른 적응적 XOR 연산을 이용하는 깊이정보 맵의 비트플레인에 대한 인코딩 방법의 일례를 보여 주는 흐름도이다.

[0142] 도 9는 본 발명의 다른 실시예에 따른 인터 픽쳐에 대한 인코딩 방법의 일례를 보여 주는 흐름도이다.

[0143] 도 10은 본 발명의 일 실시예에 따른 깊이정보 맵 인코더의 구성을 보여 주는 블록도이다.

[0144] 도 11은 도 10에 도시된 깊이정보 맵 인코더에서 DCT 기반 코딩 모드와 비트플레인 코딩 모드 중에서 코딩 효율이 더 좋은 모드를 선택하는 방법의 일례를 보여 주는 흐름도이다.

[0145] 도 12는 도 10의 깊이정보 맵 인코더의 비트율 조절부에서 비트율을 결정하는 방법의 일례를 보여 주는 흐름도이다.

[0146] 도 13은 도 10의 깊이정보 맵 인코더의 XOR 연산부에서 XOR 연산을 수행할지를 결정하는 방법의 일례를 보여 주는 흐름도이다.

[0147] 도 14는 도 10의 깊이정보 맵 인코더에서 비트플레인 인코딩부의 세부 구성의 일례를 보여 주는 블록도이다.

[0148] 도 15는 인터 픽쳐를 위한 도 10의 비트플레인 인코딩부의 세부 구성의 일례를 보여 주는 블록도이다.

[0149] 도 16은 도 14 또는 도 15의 비트플레인 인코딩부의 CAE 인코딩부에서 참조 비트플레인을 구성하는 방법의 일례를 보여 주기 위한 도면이다.

[0150] 도 17은 도 10의 깊이정보 맵 인코더에서 비트플레인 디코딩부의 세부 구성의 일례를 보여 주는 블록도이다.

[0151] 도 18은 본 발명의 일 실시예에 따른 깊이정보 맵 디코더의 구성을 보여 주는 블록도이다.

[0152] 도 19는 인터 픽쳐를 위한 도 18의 비트플레인 디코딩부의 구성의 일례를 보여 주는 블록도이다.

[0153] 도 20은 H.264/AVC를 이용한 기존 방법(Anchor)과 본 발명의 실시예에 따른 방법(Proposed)에서 비트율 대비 PSNR(Peak Signal-to-Noise Ratio)의 율-왜곡 곡선(Rate-Distortion(RD) curve)을 보여 주는 실험 결과의 일례이다.

[0154] 도 21은 H.264/AVC를 이용한 기존 방법(Anchor)과 본 발명의 실시예에 따른 방법(Proposed)에서 비트율 대비 PSNR의 율-왜곡 곡선을 보여 주는 실험 결과의 다른 예이다.

[0155] 도 22의 (a)는 'Breakdancers' 영상 시퀀스의 일부를 보여 주는 원본 깊이정보 맵 영상이고, 도 22의 (b)는 (a)를 기존의 방법(H.264/AVC)으로 인코딩하고 디코딩하여 복원한 깊이정보 맵 영상이며, 도 22의 (c)는 (a)를 본 발명의 실시예에 따라서 인코딩하고 디코딩하여 복원한 깊이정보 맵 영상이다.

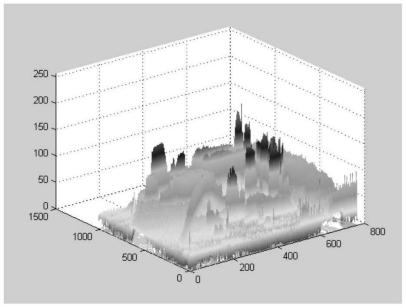
도면

도면1

도면2

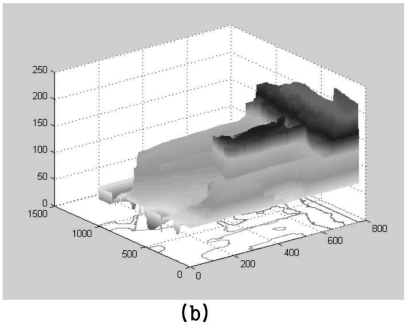
(a) (b)

도면3a



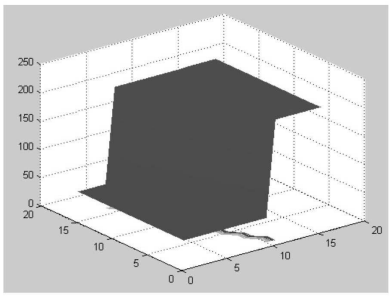
(a)

도면3b



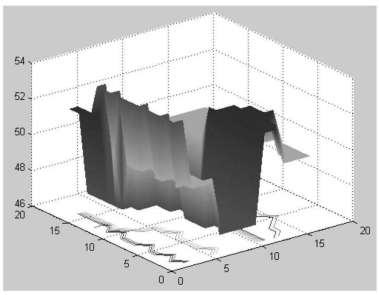
(b)

도면4a


250
200
150
100
50
0
20
15
10
5
0
0
5
10
15
20

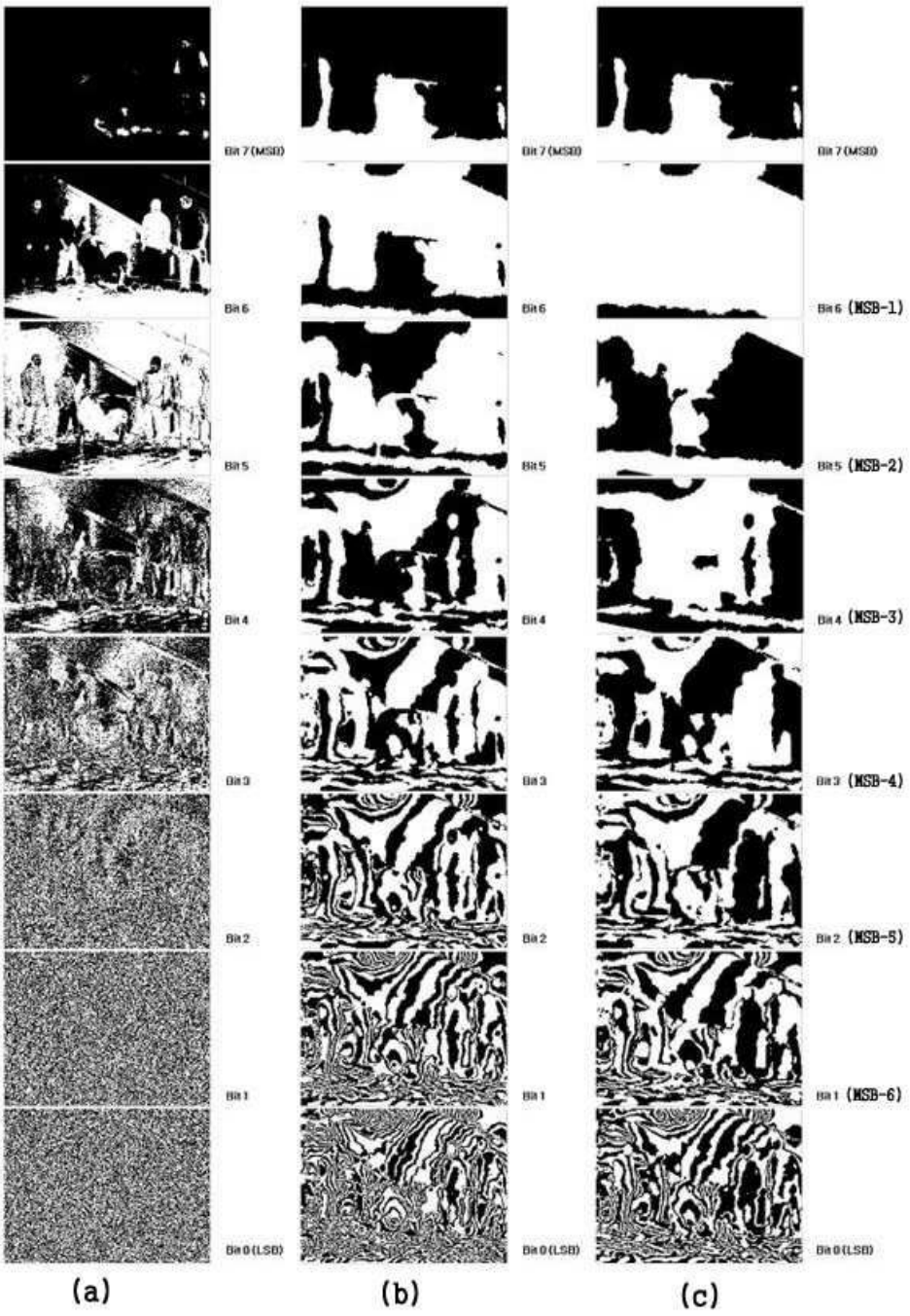
(a)

도면4b

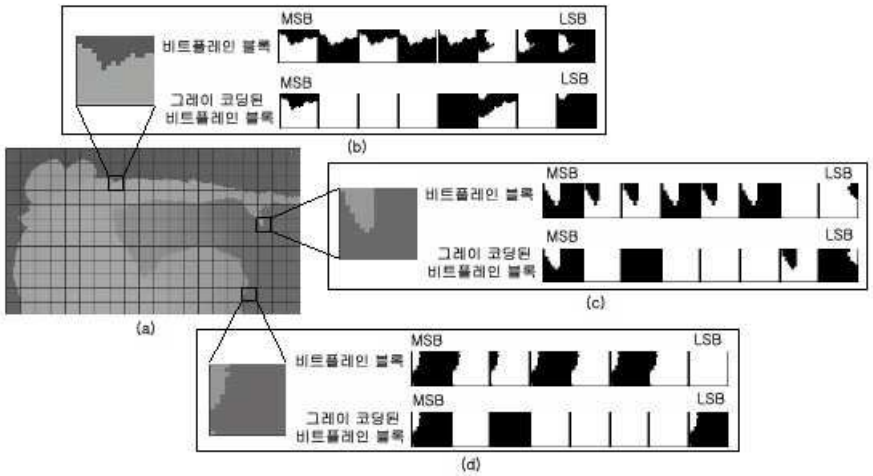

54
52
50
48
46
20
15
10
5
0
0
5
10
15
20

(b)

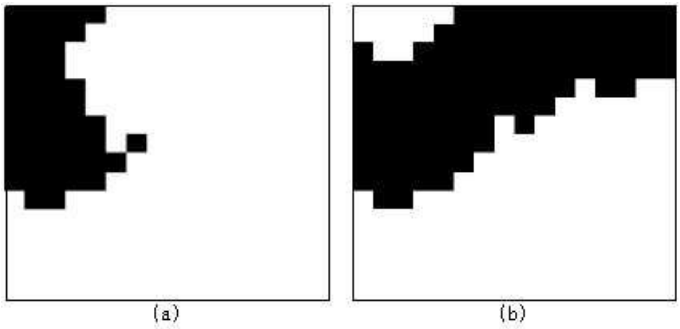
도면5


Bit 7 (MSB)
Bit 6
Bit 5
Bit 4
Bit 3
Bit 2
Bit 1
Bit 0 (LSB)
Bit 7 (MSB)
Bit 6
Bit 5
Bit 4
Bit 3
Bit 2
Bit 1
Bit 0 (LSB)
Bit 7 (MSB)
Bit 6 (MSB-1)
Bit 5 (MSB-2)
Bit 4 (MSB-3)
Bit 3 (MSB-4)
Bit 2 (MSB-5)
Bit 1 (MSB-6)
Bit 0 (LSB)
(a)
(b)
(c)

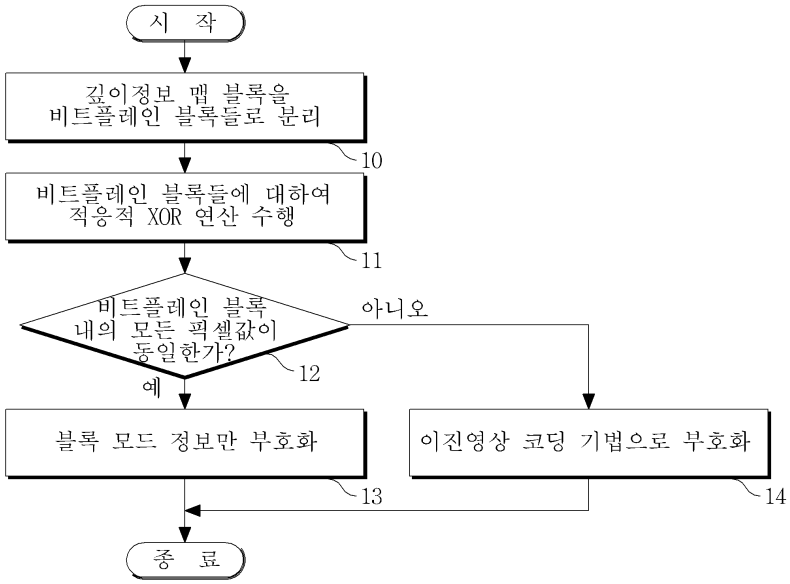
도면6


MSB
LSB
비트플레인 블록
그레이 코딩된 비트플레인 블록
(a)
(b)
(c)
(d)

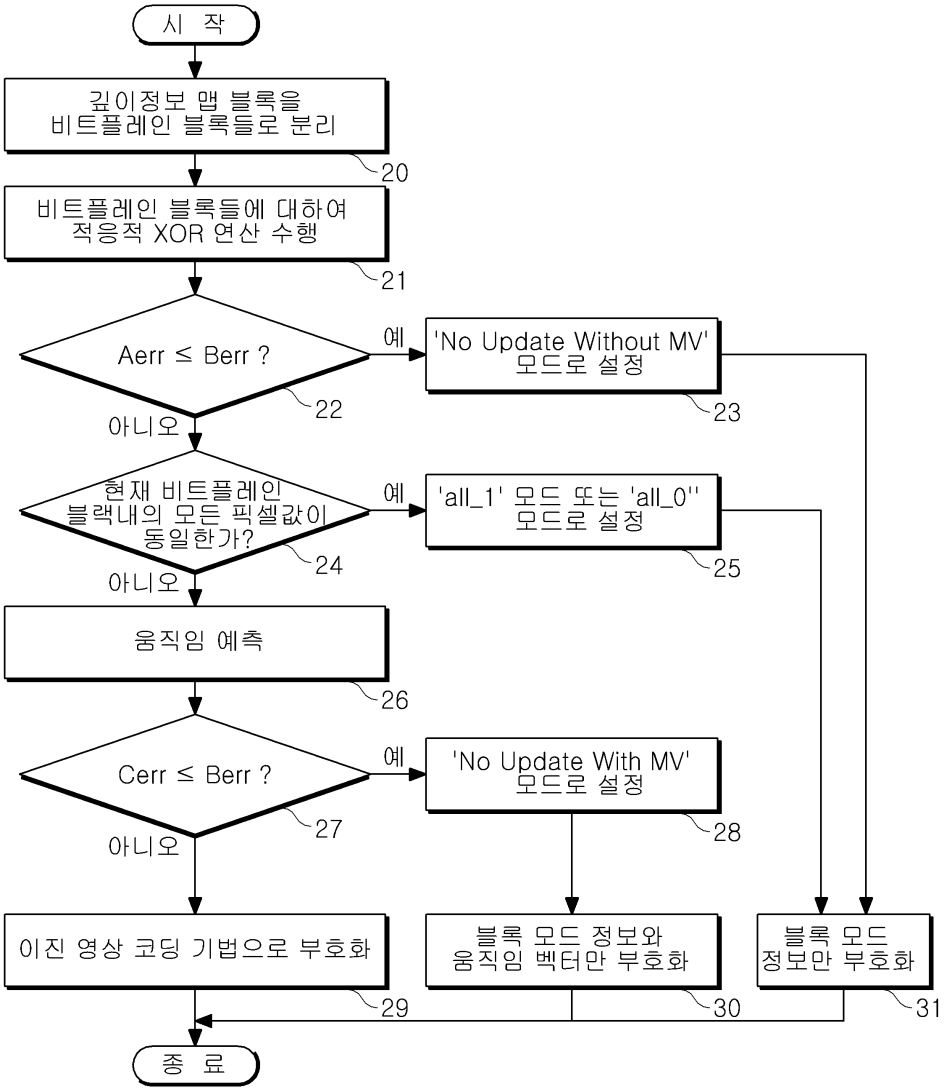
도면7


(a)
(b)

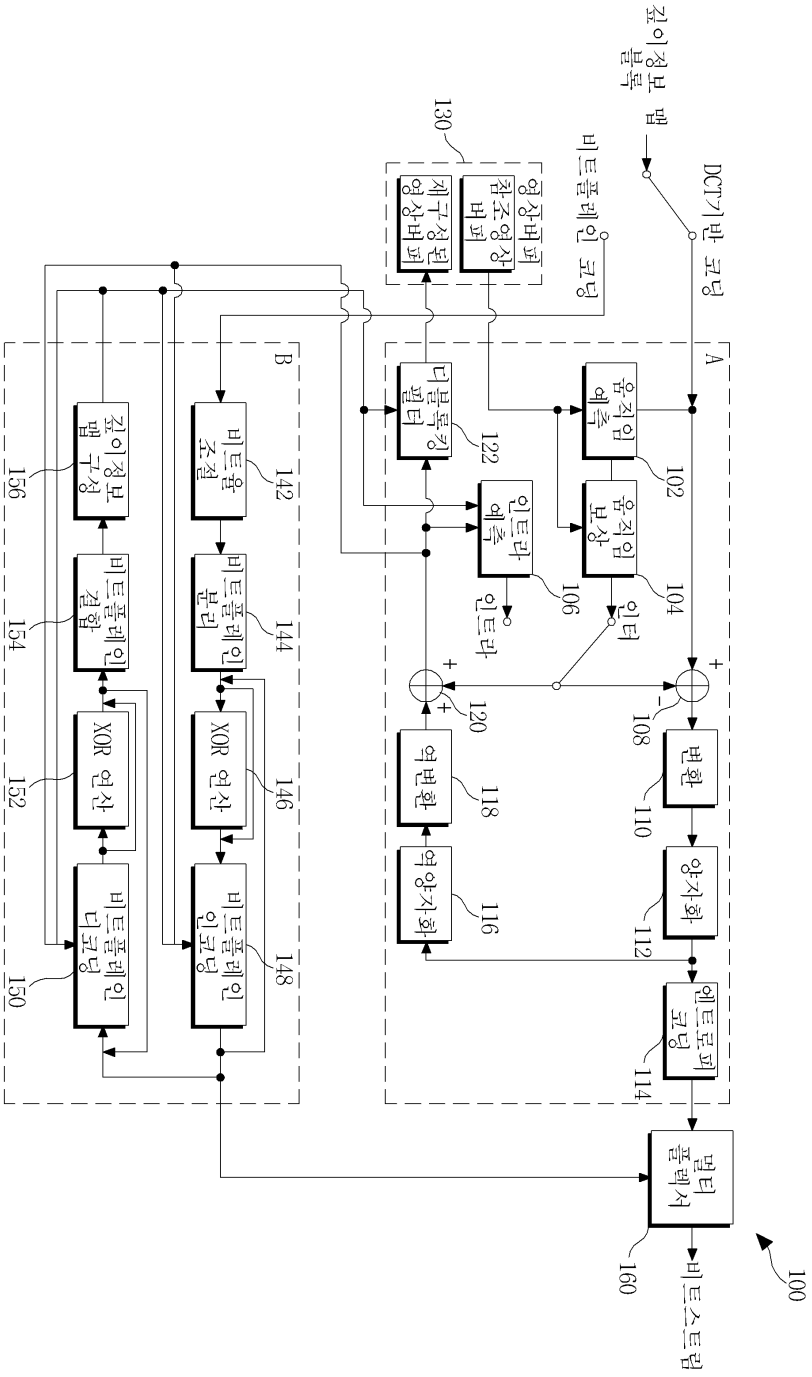
도면8


시 작
깊이정보 맵 블록을 비트플레인 블록들로 분리
10
비트플레인 블록들에 대하여 적응적 XOR 연산 수행
11
비트플레인 블록 내의 모든 픽셀값이 동일한가?
12
아니오
예
블록 모드 정보만 부호화
13
이진영상 코딩 기법으로 부호화
14
종 료

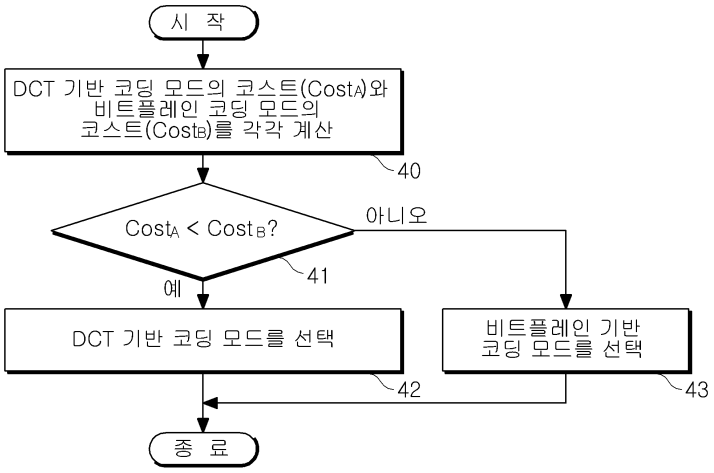
도면9


시 작
깊이정보 맵 블록을
비트플레인 블록들로 분리
20
비트플레인 블록들에 대하여
적응적 XOR 연산 수행
21
Aerr ≤ Berr ?
예
'No Update Without MV'
모드로 설정
22
23
아니오
현재 비트플레인
블랙내의 모든 픽셀값이
동일한가?
예
'all_1' 모드 또는 'all_0''
모드로 설정
24
25
아니오
움직임 예측
26
Cerr ≤ Berr ?
예
'No Update With MV'
모드로 설정
27
28
아니오
이진 영상 코딩 기법으로 부호화
블록 모드 정보와
움직임 벡터만 부호화
블록 모드
정보만 부호화
29
30
31
종 료

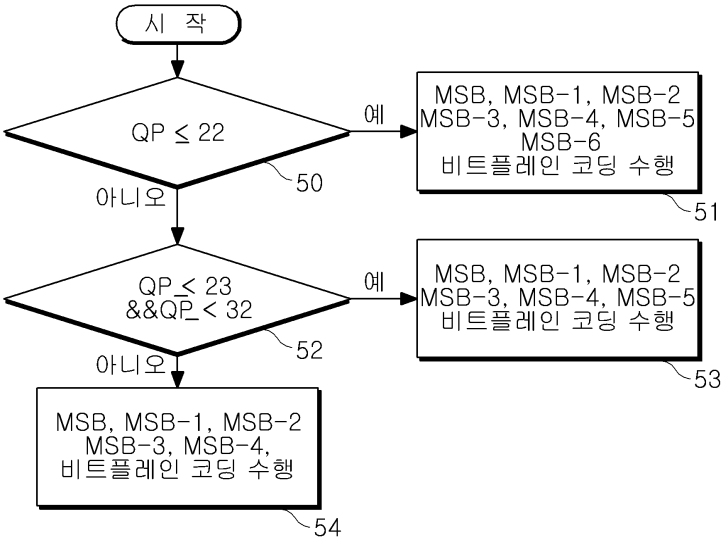
도면10


깊이정보 블록
맵
DCT기반 코딩
비트플레인 코딩
130
영상버퍼
참조영상 버퍼
재구성된 영상버퍼
A
B
움직임 예측
움직임 보상
디블록킹 필터
인트라 예측
인트라
인터
102
104
106
122
108
120
변환
역변환
양자화
역양자화
엔트로피 코딩
110
112
114
116
118
멀티플렉서
비트스트림
100
160
비트율 조절
비트플레인 분리
XOR 연산
비트플레인 인코딩
142
144
146
148
깊이정보 맵 구성
비트플레인 결합
XOR 연산
비트플레인 디코딩
156
154
152
150

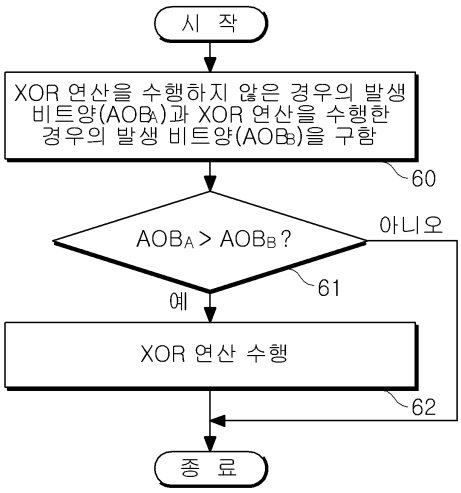
도면11


시 작
DCT 기반 코딩 모드의 코스트(Cost$_A$)와 비트플레인 코딩 모드의 코스트(Cost$_B$)를 각각 계산
40
Cost$_A$ < Cost$_B$?
41
아니오
예
DCT 기반 코딩 모드를 선택
42
비트플레인 기반 코딩 모드를 선택
43
종 료

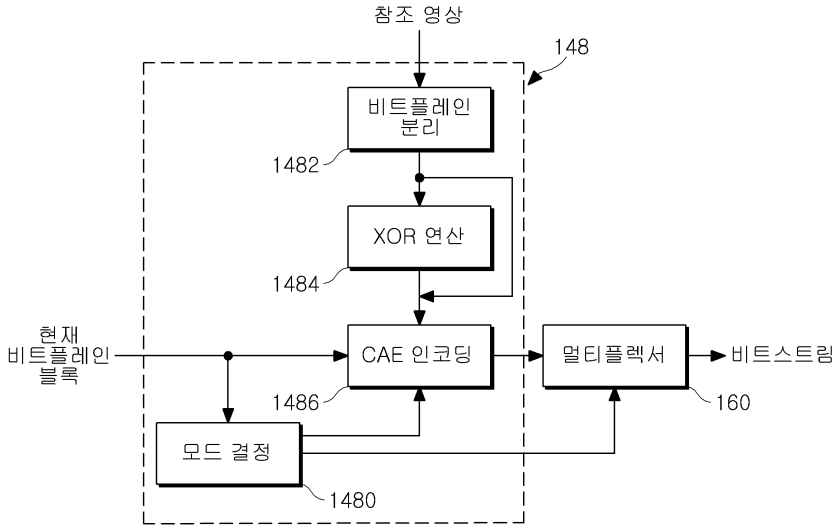
도면12


시 작
QP ≤ 22
50
예
아니오
MSB, MSB-1, MSB-2 MSB-3, MSB-4, MSB-5 MSB-6 비트플레인 코딩 수행
51
QP ≤ 23 &&QP < 32
52
예
아니오
MSB, MSB-1, MSB-2 MSB-3, MSB-4, MSB-5 비트플레인 코딩 수행
53
MSB, MSB-1, MSB-2 MSB-3, MSB-4, 비트플레인 코딩 수행
54

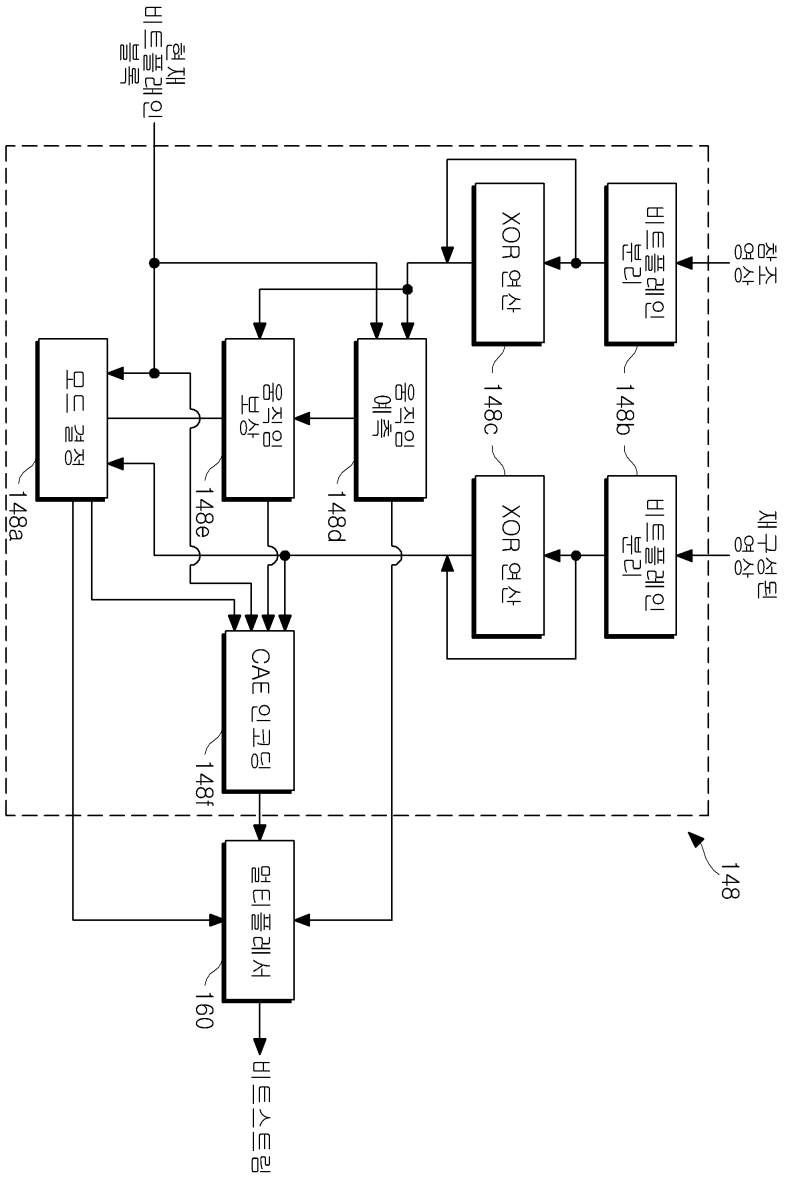
도면13


시 작
XOR 연산을 수행하지 않은 경우의 발생 비트양(AOB$_A$)과 XOR 연산을 수행한 경우의 발생 비트양(AOB$_B$)을 구함
60
AOB$_A$ > AOB$_B$?
아니오
61
예
XOR 연산 수행
62
종 료

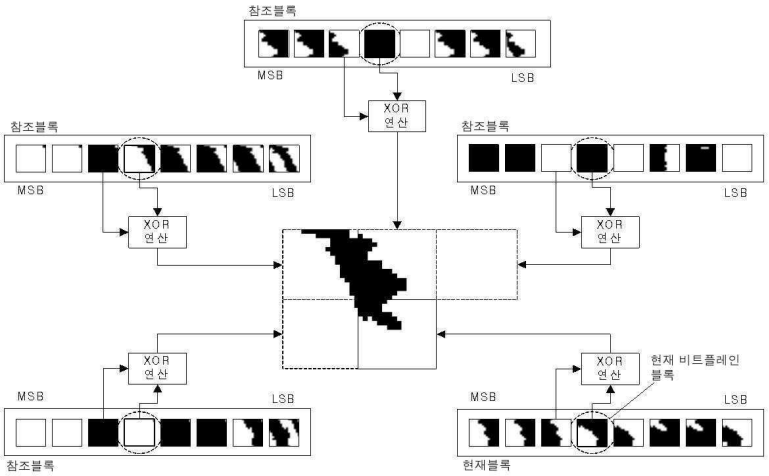
도면14


참조 영상
148
비트플레인 분리
1482
XOR 연산
1484
현재 비트플레인 블록
CAE 인코딩
멀티플렉서
비트스트림
160
1486
모드 결정
1480

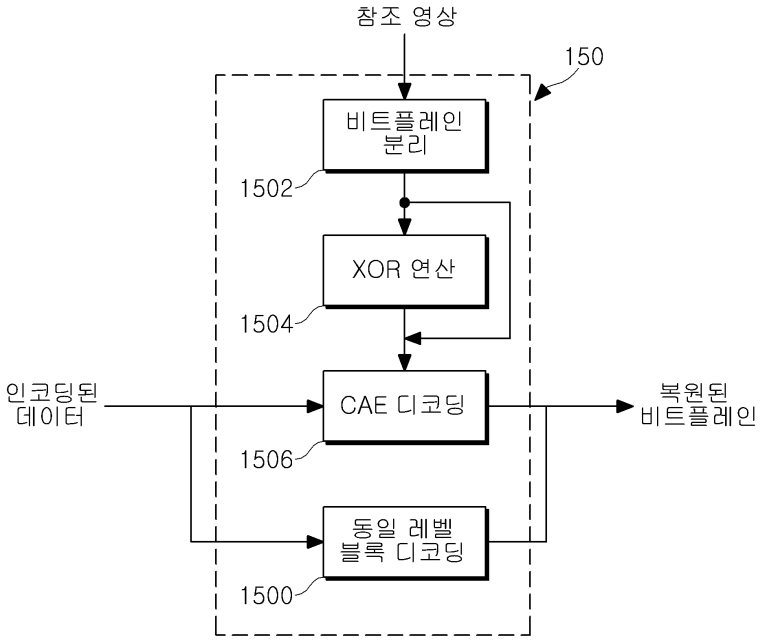
도면15


현재 비트플레인 블록
비트플레인 분리
참조 영상
XOR 연산
148b
148c
모드 결정
움직임 보상
움직임 예측
148a
148e
148d
재구성된 영상
비트플레인 분리
XOR 연산
CAE 인코딩
148f
148
멀티플렉서
160
비트스트림

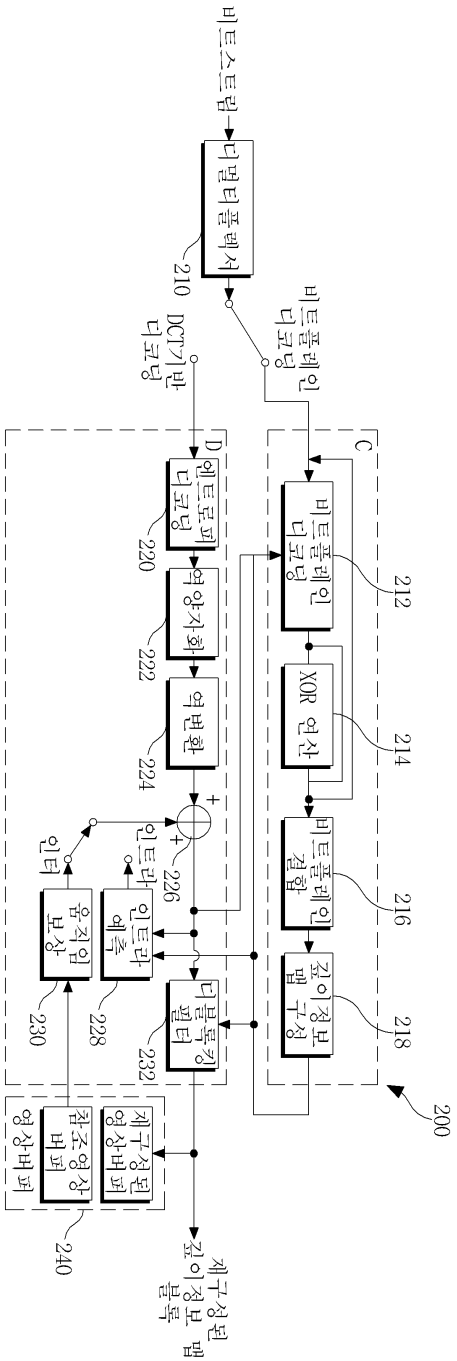
도면16


참조블록
MSB
LSB
XOR 연산
참조블록
MSB
LSB
XOR 연산
참조블록
MSB
LSB
XOR 연산
XOR 연산
현재 비트플레인 블록
MSB
LSB
참조블록
MSB
LSB
현재블록

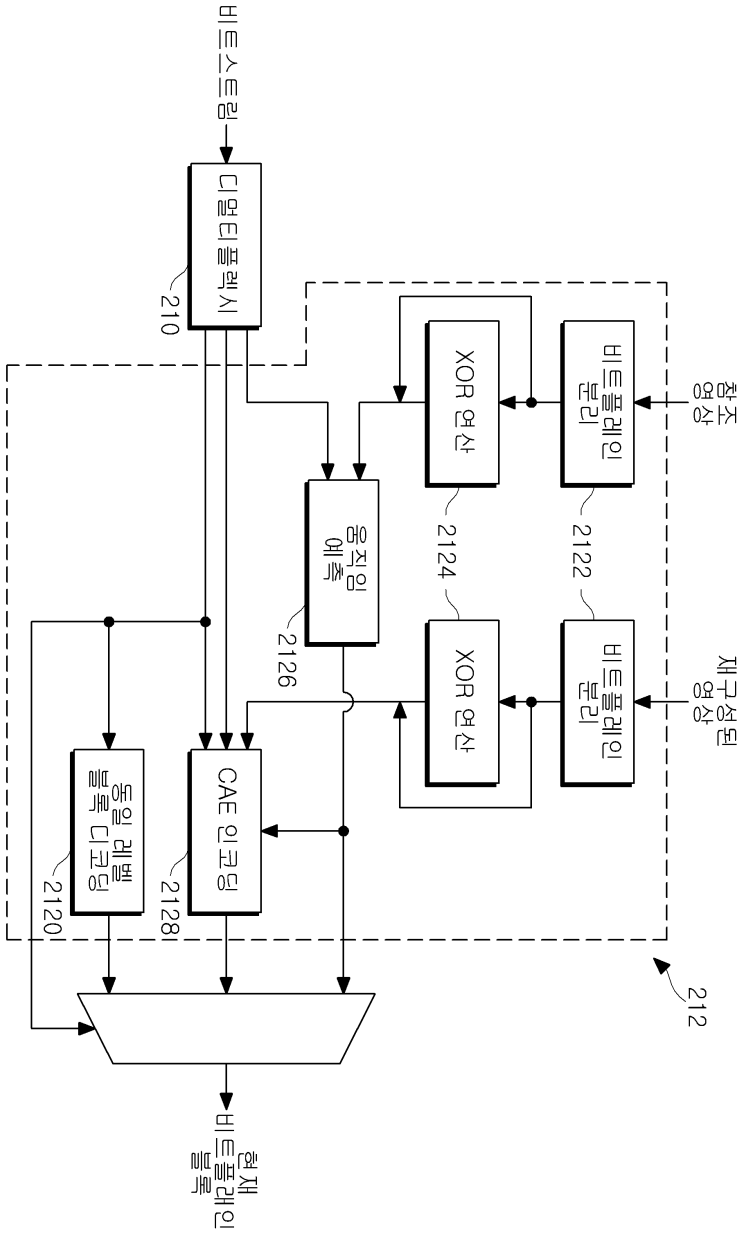
도면17


참조 영상
150
비트플레인 분리
1502
XOR 연산
1504
인코딩된 데이터
CAE 디코딩
1506
복원된 비트플레인
동일 레벨 블록 디코딩
1500

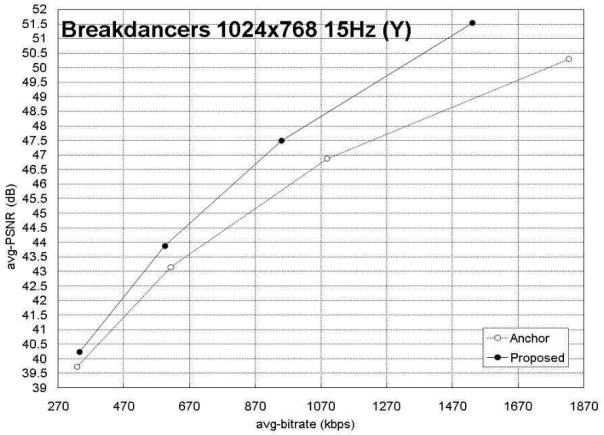
도면18


비트스트림
디멀티플렉서
210
DCT기반 디코딩
비트플레인 디코딩
D
엔트로피 디코딩
220
역양자화
222
역변환
224
+
+
226
인터
인트라 예측
228
움직임 보상
230
디블록킹 필터
232
C
비트플레인 디코딩
212
XOR 연산
214
비트플레인 결합
216
깊이정보 맵 구성
218
200
참조영상 버퍼
재구성된 영상 버퍼
영상 버퍼
240
재구성된 깊이정보 블록 맵

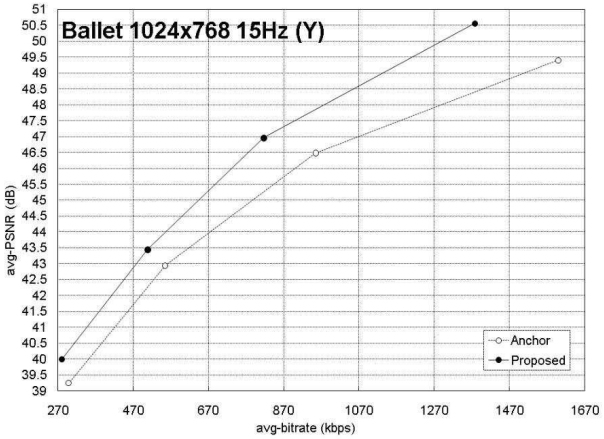
도면19


비트스트림
디멀티플렉시
210
XOR 연산
비트플레인 분리
참조 영상
움직임 예측
2124
2122
2126
XOR 연산
비트플레인 분리
재구성된 영상
동일 레벨 블록 디코딩
CAE 인코딩
2120
2128
212
현재 비트플레인 블록

도면20


Breakdancers 1024x768 15Hz (Y)
avg-PSNR (dB)
avg-bitrate (kbps)
Anchor
Proposed

도면21


Ballet 1024x768 15Hz (Y)
avg-PSNR (dB)
avg-bitrate (kbps)
Anchor
Proposed

도면22