(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2016-0002194
(43) 공개일자 2016년01월07일

(51) 국제특허분류(Int. Cl.)
H04N 19/597 (2014.01) *H04N 19/105* (2014.01)
(21) 출원번호 10-2014-0081190
(22) 출원일자 2014년06월30일
심사청구일자 없음

(71) 출원인
한국전자통신연구원
대전광역시 유성구 가정로 218 (가정동)
경희대학교 산학협력단
경기도 용인시 기흥구 덕영대로 1732 (서천동, 경희대학교 국제캠퍼스내)
(72) 발명자
방건
대전광역시 유성구 송강로42번길 61, 310-1003 (송강동, 송강청솔아파트)
이광순
대전광역시 유성구 진잠로149번길 30, 208동303호 (교촌동)
(뒷면에 계속)
(74) 대리인
인비전 특허법인

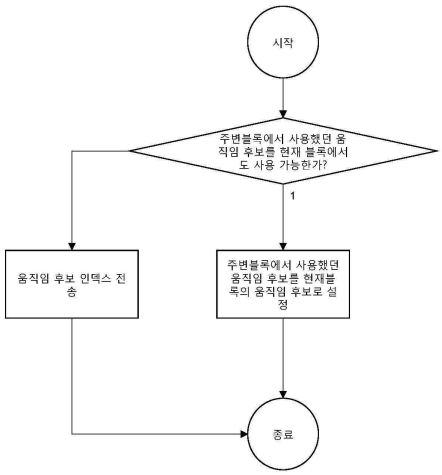
전체 청구항 수 : 총 4 항

(54) 발명의 명칭 **적응적 병합후보 선택방법 및 장치**

(57) 요 약

본 발명은 비디오의 부호화/복호화 방법 및 장치에 관한 것으로, 움직임 후보 인덱스를 전송하지 않고 적응적으로 선택하는 방법을 포함한다. 본 발명에 따르면, 비디오 부호화 효율을 증가시킬 수 있다.

대 표 도 - 도10



(72) 발명자

허남호

대전광역시 유성구 은구비남로 34, 801동1001호 (노은동, 열매마을8단지)

허영수

경기도 수원시 영통구 봉영로1492번길 20, B01호 (영통동)

박광훈

경기도 성남시 분당구 예원로 7, B동302호 (분당동, 동아빌라)

이 발명을 지원한 국가연구개발사업

과제고유번호	14-000-02-001
부처명	미래창조과학부
연구관리전문기관	정보통신기술진흥센터(IITP)
연구사업명	정보통신 방송연구개발사업
연구과제명	초고품질 콘텐츠 지원 UHD 실감방송/디지털시네마/사이니지 융합서비스 기술 개발
기 여 율	1/1
주관기관	한국전자통신연구원
연구기간	2014.03.01 ~ 2015.02.28

명 세 서

청구범위

청구항 1

움직임 후보 리스트 내의 후보 중 하나를 선택하여 현재 블록의 움직임 정보로 사용하는 방법에서, 주변블록이 사용했던 움직임 후보를 현재블록의 움직임 후보로 사용하고 움직임 후보 인덱스의 전송을 생략하는 방법

청구항 2

제 1 항에 있어서, 현재 블록과 공간적으로 인접한 주변블록에서 사용했던 움직임 후보를 활용하는 방법

청구항 3

제 1 항에 있어서, 주변블록으로서 현재 블록과 시간적으로 인접한 블록에서 사용했던 움직임 후보를 활용하는 방법

청구항 4

제 1 항에 있어서, 주변블록으로서 현재 블록과 시점 간으로 인접한 블록에서 사용했던 움직임 후보를 활용하는 방법

발명의 설명

기 술 분 야

[0001] 본 발명은 비디오의 부호화/복호화 방법 및 장치에 관한 것임.

배 경 기 술

[0002] 3차원 비디오는 3차원 입체 디스플레이 장치를 통해 사용자에게 실세계에서 보고 느끼는 것과 같은 입체감을 생생하게 제공한다. 이와 관련된 연구로써 ISO/IEC의 MPEG(Moving Picture Experts Group)과 ITU-T의 VCEG(Video Coding Experts Group)의 공동 표준화 그룹인 JCT-3V(The Joint Collaborative Team on 3D Video Coding Extension Development)에서 3차원 비디오 표준이 진행 중이다. 3차원 비디오 표준은 실제 영상과 그것의 깊이 정보 맵을 이용하여 스테레오스코픽 영상뿐만 아니라 오토스테레오스코픽 영상의 재생등을 지원할 수 있는 진보된 데이터 형식과 그에 관련된 기술에 대한 표준을 포함하고 있다.

발명의 내용

해결하려는 과제

[0003] 본 발명에서는 비디오 부호화 효율을 증가시키는 방법을 제안한다.

과제의 해결 수단

[0004] 본 발명은 움직임 후보 인덱스를 전송하지 않고 적응적으로 선택하는 방법을 포함함

발명의 효과

[0005] 본 발명은 비디오 부호화의 부호화 효율을 증가시킬 수 있다.

도면의 간단한 설명

[0006] 도 1. 3차원 비디오 시스템의 기본 구조와 데이터 형식

도 2. “balloons” 영상의 실제 영상과 깊이정보 맵 영상의 일 예: (a) 실제 영상, (b) 깊이정보 맵

도 3. CTU(CTU(LCU))를 CU단위로 분할하는 방법의 일 예

도 4. PU의 분할 구조의 일 예

도 5. 3D Video Codec에서 시점 간 예측 구조의 일예

도 6. 3D Video Encoder/Decoder의 실시 일예

도 7. 3D Video Codec의 예측 구조의 일 예

도 8. 병합 움직임 후보 리스트로 사용되는 현재 블록의 주변 블록들의 일예

도 9. 병합 움직임 후보가 리스트에 추가되는 기존의 과정

도 10. 제안하는 적응적 병합후보 선택방법

도 11. 제안하는 적응적 병합후보 선택방법의 일 예

발명을 실시하기 위한 구체적인 내용

[0007] 3차원 비디오는 3차원 입체 디스플레이 장치를 통해 사용자에게 실세계에서 보고 느끼는 것과 같은 입체감을 생생하게 제공한다. 이와 관련된 연구로써 ISO/IEC의 MPEG(Moving Picture Experts Group)과 ITU-T의 VCEG(Video Coding Experts Group)의 공동 표준화 그룹인 JCT-3V(The Joint Collaborative Team on 3D Video Coding Extension Development)에서 3차원 비디오 표준이 진행 중이다. 3차원 비디오 표준은 실제 영상과 그것의 깊이정보 맵을 이용하여 스테레오스코픽 영상뿐만 아니라 오토스테레오스코픽 영상의 재생등을 지원할 수 있는 진보된 데이터 형식과 그에 관련된 기술에 대한 표준을 포함하고 있다.

[0008] 3차원 비디오 표준에서 고려하고 있는 기본적인 3차원 비디오 시스템은 도 1과 같다. 송신 측에서는 스테레오 카메라, 깊이정보 카메라, 다시점 카메라, 2차원 영상을 3차원 영상으로 변환 등을 이용하여 N(N≥2) 시점의 영상 콘텐츠를 획득한다. 획득된 영상 콘텐츠에는 N 시점의 비디오 정보와 그것의 깊이정보 맵(Depth-map) 정보 그리고 카메라 관련 부가정보 등이 포함될 수 있다. N 시점의 영상 콘텐츠는 다시점 비디오 부호화 방법을 사용하여 압축되며, 압축된 비트스트림은 네트워크를 통해 단말로 전송된다. 수신 측에서는 전송받은 비트스트림을 다시점 비디오 복호화 방법을 사용하여 복호화하여 N 시점의 영상을 복원한다. 복원된 N 시점의 영상은 깊이정보 맵 기반 랜더링(DIBR; Depth-Image-Based Rendering) 과정에 의해 N 시점 이상의 가상시점 영상들을 생성한다. 생성된 N 시점 이상의 가상시점 영상들은 다양한 입체 디스플레이 장치에 맞게 재생되어 사용자에게 입체감이 있는 영상을 제공하게 된다.

[0009] 가상시점 영상을 생성하는데 사용되는 깊이정보 맵은 실세계에서 카메라와 실제 사물(object) 간의 거리(실사 영상과 동일한 해상도로 각 화소에 해당하는 깊이정보)를 일정한 비트수로 표현한 것이다. 깊이정보 맵을 생성하는 방법은 카메라를 이용하여 획득하는 방법과 실제 일반 영상(Texture image)을 이용하여 자동적으로 생성하는 방법으로 나뉠 수 있다. 카메라를 이용하여 획득하는 방법의 경우, 깊이정보 카메라가 일정한 거리 내에서만 작동된다는 문제점이 있다. 반면, 실제 일반 영상을 이용하여 자동적으로 생성하는 방법의 경우, 두 일반 영상 간의 시점 차이(disparity)를 이용하여 깊이정보 맵을 생성한다. 다시 말하면, 현재 시점에서의 임의의 한 화소를 주변 시점에서의 화소들과 비교하여 매칭이 가장 잘되는 부분의 화소를 찾고, 그 화소들간의 거리를 깊이정보로 표현하는 방법이다. 실제 일반 영상을 이용하여 자동적으로 생성된 깊이정보 맵의 예로써, 도 2는 국제 표준화 기구인 MPEG의 3차원 비디오 부호화 표준에서 사용 중인 “balloons” 영상(도 2 (a))과 그것의 깊이정보 맵(도 2 (b))을 보여주고 있다. 실제 도 2의 깊이정보 맵은 화면에 보이는 깊이 정보를 화소당 8비트로 표현한 것이다.

[0010] 실제 영상과 그것의 깊이정보 맵을 부호화하는 방법의 일예로, H.264/AVC(MPEG-4 Part 10 Advanced Video Coding)를 이용하여 부호화를 수행할 수 있으며, 또 다른 방법의 일 예로 MPEG(Moving Picture Experts Group)과 VCEG(Video Coding Experts Group)에서 공동으로 표준화를 진행한 HEVC(High Efficiency Video Coding) 국제 동영상 표준를 사용할 수 있다.

[0011] HEVC에서는 영상을 효율적으로 부호화하기 위해서 CU(Coding Unit: CU, 이하 ‘CU’라 함)로 부호화를 수행한다. 도 3는 영상을 부호화할 때 CTU(Coding Tree Unit; 이하 ‘CTU’라 함. 혹은 CTU(LCU)(Largest Coding Unit)라고도 할 수 있음) 내에서 CU를 분할하는 방법을 나타낸 그림이다.

[0012] 도 3와 같이 HEVC에서는 영상을 CTU(LCU) 단위로 순차적으로 분할한 후, CTU(LCU) 단위로 분할구조를 결정한다. 분할구조는 CTU(LCU) 내에서 영상을 효율적으로 부호화하기 위한 CU의 분포를 의미하는데, 이 분포는 CU를 크기가 가로 세로 절반씩 감소한 4개의 CU로 분할할지를 결정함으로써 결정할 수 있다. 분할된 CU는 같은 방식으로 가로 세로 절반씩 감소한 4개의 CU로 재귀적으로 분할할 수 있다. 이 때, CU의 분할은 미리 정의된 깊이까지 분할을 수행할 수 있는데, 깊이 정보(Depth)는 CU의 크기를 나타내는 정보로써 모든 CU에 저장되어 있다. 기본이 되는 CTU(LCU)의 깊이는 0이고, SCU의 깊이는 미리 정의된 최대 깊이이다. CTU(LCU)로부터 가로 세로 절반으로 분할을 수행할 때마다 CU의 깊이가 1씩 증가한다. 각각의 깊이 별로, 분할을 수행하지 않는 CU의 경우에는 2Nx2N의 크기로, 분할을 수행할 경우에는 NxN 크기의 CU 4개로 분할된다. N의 크기는 깊이가 1씩 증가할 때마다 절반으로 감소한다. 그림 5에서는 최소 깊이가 0인 CTU(LCU)의 크기가 64x64 화소이고, 최대 깊이가 3인 SCU(Smallest Coding Unit)의 크기가 8x8 화소인 예이다. 64x64 화소 CU(CTU(LCU))는 깊이 '0'으로 32x32 화소 CU는 깊이 '1'로 16x16 화소 CU는 깊이 '2'로 8x8 CU(SCU)는 깊이 '3'으로 표현된다. 또한 특정 CU를 분할할지에 대한 정보는 CU마다 1비트 정보인 분할 정보를 통해 표현한다. 이 분할 정보는 SCU를 제외한 모든 CU에 포함되어 있으며, CU를 분할하지 않을 경우에는 분할 정보에 '0'을 저장하고, 분할할 경우에는 '1'을 저장한다.

[0013] 이러한 CU는 부호화 단위로써, 하나의 CU단위로 부호화 모드를 가질 수 있다. 즉, CU 각각은 화면 내 부호화 (MODE_INTRA 혹은 INTRA라고 할 수도 있음) 모드 혹은 화면 간 부호화 (MODE_INTER 혹은 INTER라고 할 수도 있음) 모드로 나누어 질 수 있다. 화면 간 부호화 (MODE_INTER) 모드에는 MODE_INTER 모드와 MODE_SKIP (SKIP이라고 할 수도 있음) 모드로 구분될 수 있다.

[0014] PU(Prediction Unit)는 예측의 단위로서, 도 4과 같이 하나의 CU는 여러 개의 PU(Prediction Unit: PU, 이하 'PU'라 함)들로 분할되어 예측이 수행될 수 있다. 하나의 CU의 부호화 모드가 INTRA 라면, 그 CU내의 PU들은 모두 INTRA 로 부호화 되고, 하나의 CU의 부호화 모드가 INTER 라면, 그 CU내의 PU들은 모두 INTER 로 부호화된다. 실시 일예로, 하나의 CU가 INTRA 일 경우, PU의 분할 구조는 PU 2Nx2N과 PU NxN만 가질 수 있다. 실시 일예로, 하나의 CU가 INTER라면 PU는 도 4의 모든 분할 구조를 가질 수 있다.

[0015] 실제 영상과 그것의 깊이정보맵은 카메라 하나뿐만아니라 여러 개의 카메라에서 획득한 영상일 수 있다. 여러 개의 카메라에서 획득한 영상은 독립적으로 부호화될 수 있으며, 일반적인 2차원 비디오 부호화 코덱을 사용할 수 있다. 또한, 여러 개의 카메라에서 획득한 영상은 시점 간의 상관성이 존재하므로, 부호화 효율을 높이기 위하여 서로 다른 시점간 예측을 사용하여 영상들을 부호화할 수 있다. 일 실시 예로, 도 5은 카메라 3대에서 획득한 영상들에 대한 시점간 예측 구조의 일예를 나타낸다.

[0016] 도 5에서, 시점 1(View 1)은 시점 0(View 0)을 기준으로 왼쪽에 위치한 카메라에서 획득한영상이고, 시점 2(View 2)는 시점 0(View 0)을 기준으로 오른쪽에 위치한 카메라에서 획득한 영상이다. 또한, 시점 1(View 1)과 시점 2(View 2)는 시점 0(View 0)을 참조 영상으로 사용하여 시점 간 예측을 수행하며, 부호화 순서는 시점 1(View 1)과 시점 2(View 2)보다 시점 0(View 0)이 먼저 부호화되어야 한다. 이때, 시점 0(View 0)은 다른 시점과 상관없이 독립적으로 부호화될 수 있으므로 독립적인 시점(Independent View)이라고 한다. 반면, 시점 1(View 1)과 시점 2(View 2)는 시점 0(View 0)을 참조 영상으로 사용하므로 의존적인 시점(Dependent View)라고 한다. 독립적인 시점 영상은 일반적인 2차원 비디오 코덱을 사용하여 부호화 될 수 있다. 반면, 의존적인 시점 영상은 시점간 예측을 수행하여야 하므로, 시점간 예측 과정이 포함된 3차원 비디오 코덱을 사용하여 부호화할 수 있다.

[0017] 또한, 시점 1(View 1)과 시점 2(View 2)의 부호화 효율을 증가시키기 위하여 깊이정보 맵을 이용하여 부호화 할 수 있다. 일예로, 실제 영상과 그것의 깊이정보 맵을 부호화할 때, 서로 독립적으로 부호화/복호화할 수 있다. 또한 실제 영상과 깊이정보 맵을 부호화할 때, 도 6와 같이 서로 의존적으로 부호화/복호화 할 수 있다. 일 실시 예로, 이미 부호화된/복호화된 깊이정보맵을 이용하여 실제 영상을 부호화/복호화할 수 있으며, 반대로 이미 부호화된/복호화된 실제 영상을 이용하여 깊이정보맵을 부호화/복호화할 수 있다.

[0018] 일 실시 예로, 카메라 3대에서 획득한 실제 영상과 그것의 깊이정보맵을 부호화하기 위한 부호화 예측 구조를 도 7에 나타내었다. 도 7에서 3개의 실제 영상을 시점에 따라 T0, T1, T2로 나타내었고, 실제 영상과 동일한 위

치의 3개의 깊이정보맵을 시점에 따라 D0, D1, D2로 나타내었다. 여기서 T0와 D0는 시점 0(View 0)에서 획득한 영상이며, T1와 D1는 시점 1(View 1)에서 획득한 영상이며, T2와 D2는 시점 2(View 2)에서 획득한 영상이다. 각 픽쳐는 I(Intra Picture), P(Uni-prediction Picture), B(Bi-prediction Picture)로 부호화될 수 있다. 도 7에서 화살표는 예측 방향을 나타낸다. 즉, 실제 영상과 그것의 깊이정보맵은 서로 의존적으로 부호화/복호화된다.

[0019] 실제 영상에서 현재 블록의 움직임 정보(움직임 벡터만을 의미할 수 있으며, 혹은 움직임벡터와 참조영상 번호와 단방향 예측인지 양방향 예측인지 그리고 시점간 예측인지 시간적 예측인지 또다른 예측인지를 의미할 수도 있다.)를 유추하기 위한 방법은 크게 시간적 예측과 시점간 예측으로 나뉜다. 시간적 예측은 동일한 시점 내에서 시간적 상관성을 이용한 예측 방법이고, 시점간 예측은 인접한 시점에서 시점간 상관성을 이용한 예측 방법이다. 이러한 시간적 예측과 시점간 예측은 한 픽쳐에서 서로 혼용되어 사용될 수 있다.

[0020] 영상 부호화/복호화 시 움직임 정보의 부호화 방법 중 하나로 병합 움직임(merge) 방법을 이용한다. 이때 움직임 정보란, 움직임 벡터, 참조영상에 대한 인덱스 그리고 예측 방향(단방향, 양방향, 등) 중에서 적어도 하나를 포함하는 정보다. 예측 방향은 참조픽쳐목록(Reference Picture List; RefPicList)의 사용에 따라 크게 단방향 예측과 양방향 예측으로 나누어 질 수 있다. 단방향 예측에는 순방향 참조픽쳐목록 (LIST 0)을 사용한 순방향 예측(Pred_L0; Prediction L0)과 역방향 참조픽쳐목록 (LIST 1)을 사용한 역방향 예측(Pred_L1; Prediction L1)으로 구분된다. 또한, 양방향 예측(Pred_BI; Prediction BI)은 순방향 참조픽쳐목록 (LIST 0)와 역방향 참조픽쳐목록 (LIST 1)을 모두 사용하며, 순방향 예측과 역방향 예측이 모두 존재하는 것을 말할 수 있다. 또한, 순방향 참조픽쳐목록 (LIST 0)을 역방향 참조픽쳐목록 (LIST 1)에 복사하여 순방향 예측이 두개 존재하는 경우도 양방향 예측에 포함할 수 있다. 이러한 예측 방향 여부를 predFlagL0, predFlagL1를 사용하여 정의할 수 있다. 실시 일예로, 단방향 예측이고 순방향 예측일 경우 predFlagL0는 '1'이 되고 predFlagL1은 '0'이 될 수 있다. 또한, 단방향 예측이고 역방향 예측일 경우 predFlagL0는 '0'이 되고 predFlagL1은 '1'이 될 수 있다. 또한, 양방향 예측일 경우 predFlagL0는 '1'이 되고 predFlagL1은 '1'이 될 수 있다.

[0021] 병합 움직임은 부호화 유닛(Coding Unit: CU, 이하 'CU'라 함) 단위의 병합 움직임과 예측 유닛(Prediction Unit: PU, 이하 'PU'라 함) 단위의 병합 움직임이 가능하다. CU또는 PU(이하, 설명의 편의를 위해 '블록'이라 함) 단위로 병합 움직임을 수행하는 경우에는, 블록 파티션(partition)별로 병합 움직임을 수행할지에 관한 정보와 현재 블록에 인접한 주변 블록(현재 블록의 좌측 인접 블록, 현재 블록의 상측 인접 블록, 현재 블록의 시간적(temporral) 인접 블록 등) 중 어떤 블록과 병합 움직임을 할 것인지에 대한 정보를 전송할 필요가 있다.

[0022] 병합 움직임(merge) 후보 리스트(List)은 움직임 정보들이 저장된 리스트을 나타내며, 병합 움직임이 수행되기 전에 생성된다. 여기서 병합 움직임 후보 리스트에 저장되는 움직임 정보는 현재 블록에 인접한 주변 블록의 움직임 정보이거나 참조 영상에서 현재 블록에 대응되는(collocated) 블록의 움직임 정보일 수 있다. 또한 병합 움직임 후보 리스트에 저장되는 움직임 정보는 이미 병합 움직임 후보 리스트에 존재하는 움직임 정보들을 조합하여 만든 새로운 움직임 정보일 수 있다.

[0023] 병합 움직임 후보 리스트는 도 8의 주변 블록(A, B, C, D, E)과 그리고 동일 위치의 후보 블록(H(혹은 M))에 대하여 해당 블록의 움직임 정보가 현재 블록의 병합 움직임에 이용될 수 있는지를 판단하여, 이용 가능한 경우에는 해당 블록의 움직임 정보를 병합 움직임 후보 리스트에 입력할 수 있다. 또한, 각 주변 블록은 서로 동일한 움직임 정보를 가지는지 확인하여 동일한 움직임 정보일 경우, 해당 주변 블록의 움직임 정보는 병합 움직임 후보 리스트에 포함되지 않는다. 실시 일 예로, 도 8에서 X 블록에 대한 병합 움직임 후보 리스트를 생성할 때, 주변 블록 A가 사용가능하여 병합 움직임 후보 리스트에 포함된 후, 주변 블록 B는 주변 블록 A와 동일한 움직임 정보가 아닐 경우에만, 병합 움직임 후보 리스트에 포함될 수 있다. 동일한 방법으로 주변 블록 C는 주변 블록 B와 동일한 움직임 정보가 아닐 경우에만 병합 움직임 후보 리스트에 포함될 수 있다. 동일한 방법으로 주변 블록 D와 주변 블록 E에 적용될 수 있다. 여기서 동일한 움직임 정보라는 것은 움직임 벡터가 동일하고 동일한 참조픽쳐를 사용하고 동일한 예측 방향(단방향(정방향, 역방향), 양방향)을 사용함을 의미할 수 있다. 마지막으로 도 8에서 X 블록에 대한 병합 움직임 후보 리스트는 소정의 순서, 예컨대, A→B→C→D→E→H(혹은 M) 블록 순서로 리스트에 추가될 수 있다.

[0024] 비디오 부호화에서는 움직임 정보를 효율적으로 부호화하기 위하여, 주변의 움직임 정보를 이용할 수 있다. 주변의 움직임 정보로부터 유도 된 병합 움직임 후보들은 도 9와 같이 병합 움직임 후보 리스트에 삽입된다. 그 후, 가상으로 부호화를 시도하여 가장 부호화 효율이 좋은 후보를 최종적으로 선택한다. 선택 된 후보는 엔트로피 부호화를 이용하여 부호화 된다.

[0025] 이 때, 엔트로피 부호화의 효율을 증가시키기 위해서는 리스트의 첫 번째 위치에 들어가는 후보가 되도록 많이 선택되어 0번 인덱스가 부호화 되는 횟수가 많아야 한다. 따라서 후보들 간의 우선순위가 존재하게 되며, 통계적으로 가장 선택이 많이되는 순서대로 후보들을 리스트에 추가한다. 일 예로, 현재 3D-HEVC에서는 MPI(텍스처로부터 상속), IvMC(시점 간 움직임), A1(좌), B1(상), B0(우상), IvDC(시차), VSP(시점합성), A0(좌하), B2(좌상), ShiftIV(보정시차), Col(동일위치), Bi(양방향), Zero(0벡터) 후보가 나열 된 순서대로 우선적으로 병합 움직임 후보 리스트에 삽입 된다.

[0026] 표 1은 텍스처의 병합 움직임 후보 선택 비율의 일 예이다. 텍스처에서는 MPI 후보가 사용되지 않으므로 IvMC가 0번째 인덱스를 차지하며 가장 높은 비율로 선택된다. 78%의 후보가 0 값으로 엔트로피 부호화 되므로 부호화 효율이 좋은 편이다. 이후의 후보들 또한 정확하진 않지만 대체적으로 선택비율이 높은 순서대로 나열되어 있다.

표 1. 텍스처의 병합 움직임 후보 선택 비율의 일 예

[0027]

후보	비율	평균값
MPI	0%	0
IVMC	78%	0
A1	8%	0.9
B1	6%	1.4
B0	2%	2.1
IVDC	1%	2.3
VSP	0%	3.5
A0	0%	4.3
B2	0%	3
IVSHIFT	3%	4.2
COL	1%	4.6

[0028] 이와 같이 많은 종류의 병합 움직임 후보들이 존재하는데, 움직임 병합을 사용하는 블록을 부호화 시, 리스트 내에서 어떤 후보를 사용할지를 의미하는 후보 인덱스를 전송해야 하며, 이는 압축 비트스트림에서 많은 비중을 차지한다. 따라서 만일, 후보 인덱스를 적응적으로 선택함으로써 인덱스 전송을 생략할 수 있다면 비디오 부호화 효율을 증가시킬 수 있을 것이다.

[0029] 비디오 부호화에서 병합 후보리스트를 구성 시, 여러 종류의 후보들이 사용된다. 이 때, 높은 빈도로 사용되는 후보에 대해서는, 주변블록과 현재블록 간의 움직임 후보선택에 대한 상관성이 높다.

[0030] 본 발명에서는 주변블록이 특정 후보를 사용했을 시 현재 블록을 부호화 시, 움직임 후보 리스트 상에서 움직임 정보로 이용할 후보의 인덱스를 부호화 하지 않고, 주변블록에서 사용했던 후보를 현재블록의 움직임 후보로 사용하는 방법을 제안한다. 제안하는 방법의 개념도는 도 10과 같다. 제안하는 방법을 통하여 움직임 후보 인덱스를 전송하지 않아도 되므로 부호화 효율을 증가시킬 수 있다.

[0031] 도 11는 제안하는 방법을 3D-HEVC의 움직임 병합에 적용하는 일 예이다. 현재 블록이 기본시점 텍스처에 속해있

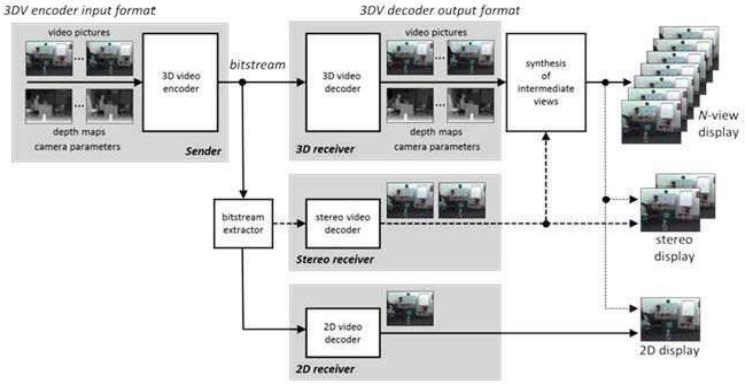
지 않고 아래 조건 중 하나를 만족한다면, 병합 움직임 후보를 부호화 하지 않는다.

[0032] - 조건 1: 현재 픽처가 깊이맵이 아니고, 현재 시점과 동일한 이전 픽처 내의 동일위치 블록이 시점 간 움직임 후보(IVMC, Inter-view merging candidate)를 사용했으며, 현재블록에서 시점 간 움직임 후보를 사용 가능함

[0033] - 조건 2: 현재 픽처가 깊이맵이고, 현재 픽처에 상응하는 텍스처 내의 동일위치 블록이 움직임 상속 후보(MPI, Motion Parameter Inheritance)를 사용했으며, 현재블록에서 움직임 상속 후보를 사용 가능함

도면

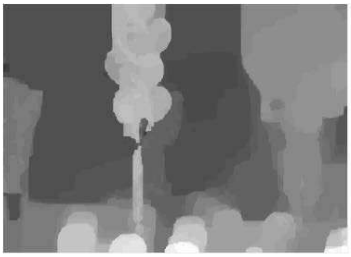
도면1



도면2

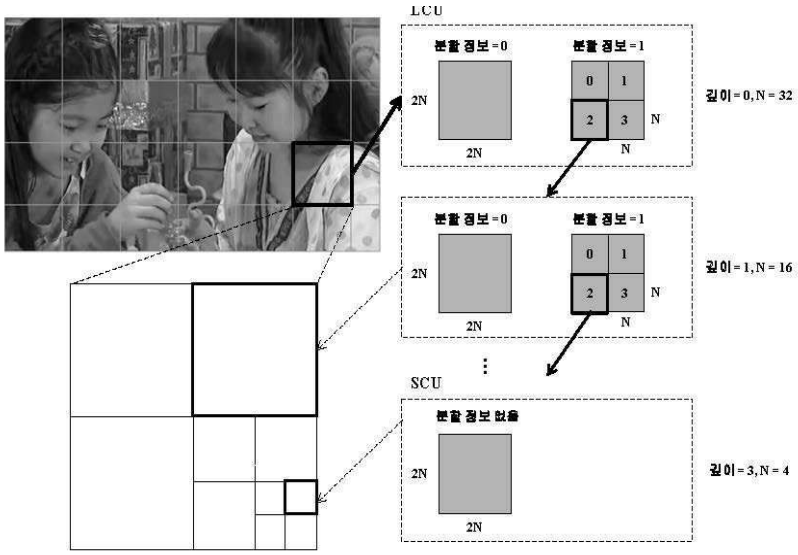


(a) 실제 영상

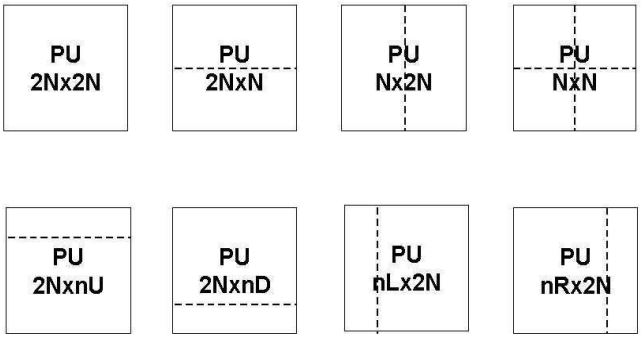


(b) 깊이정보 맵

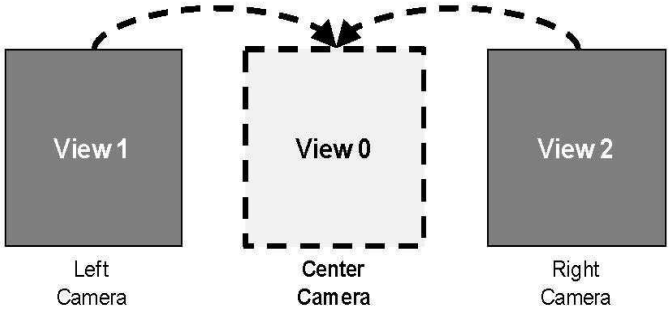
도면3


LCU
분할 정보 = 0
분할 정보 = 1
2N
N
깊이 = 0, N = 32
깊이 = 1, N = 16
SCU
분할 정보 없음
깊이 = 3, N = 4

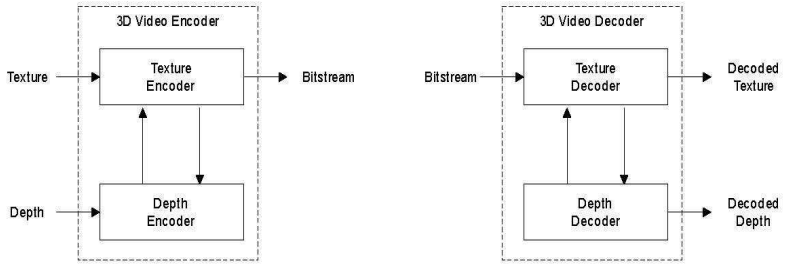
도면4


PU
2Nx2N
PU
2NxN
PU
Nx2N
PU
NxN
PU
2NxnU
PU
2NxnD
PU
nLx2N
PU
nRx2N

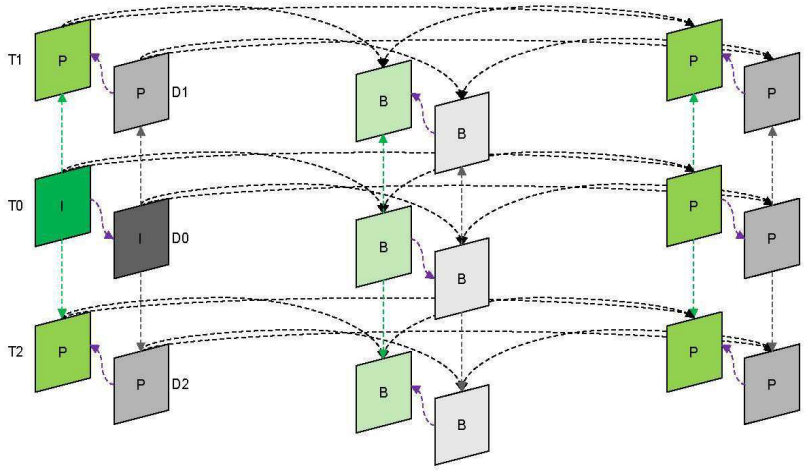
도면5


View 1
View 0
View 2
Left
Camera
Center
Camera
Right
Camera

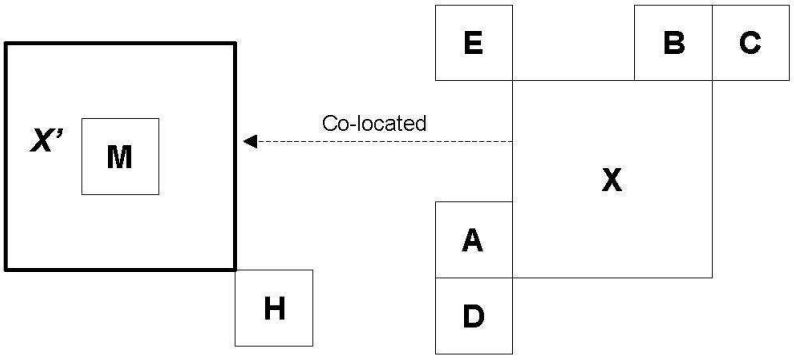
도면6


3D Video Encoder
Texture
Texture Encoder
Bitstream
Depth
Depth Encoder
3D Video Decoder
Bitstream
Texture Decoder
Decoded Texture
Depth Decoder
Decoded Depth

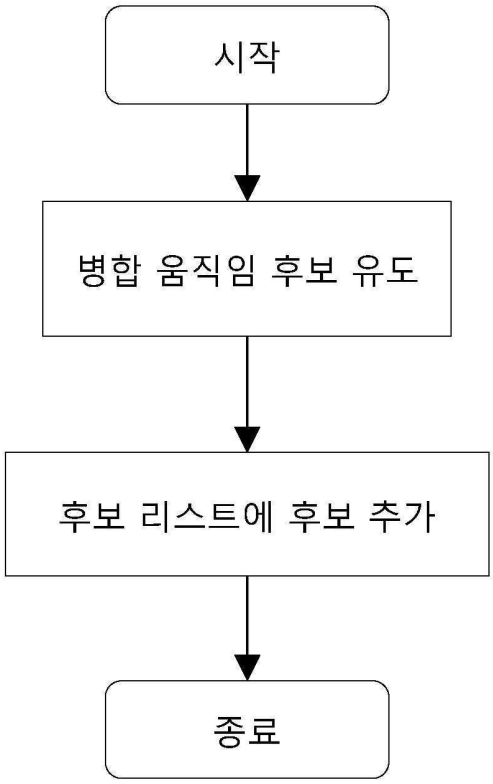
도면7


T1
T0
T2
P
D1
I
D0
D2
B

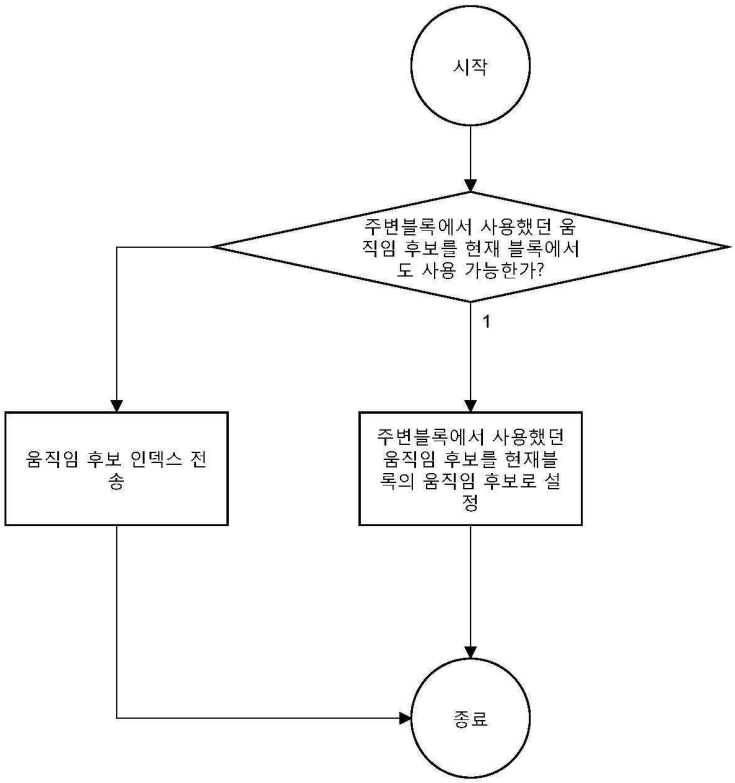
도면8


X'
M
H
Co-located
E
B
C
X
A
D

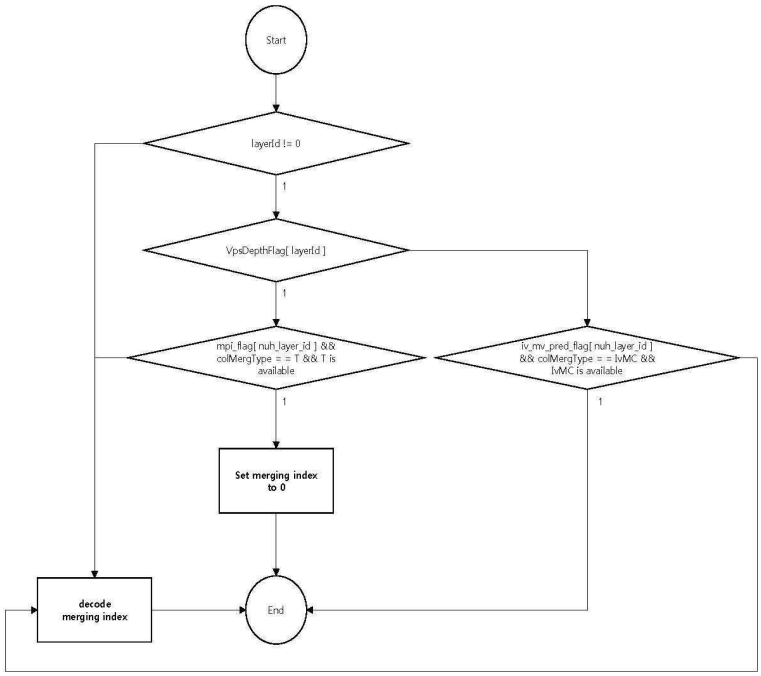
도면9



도면10


시작
주변블록에서 사용했던 움직임 후보를 현재 블록에서도 사용 가능한가?
1
움직임 후보 인덱스 전송
주변블록에서 사용했던 움직임 후보를 현재블록의 움직임 후보로 설정
종료

도면11


Start
layerId != 0
1
VpsDepthFlag[layerId]
1
mpi_flag[nuh_layer_id] && colMergType = = T && T is available
iv_mv_pred_flag[nuh_layer_id] && colMergType = = IvMC && IvMC is available
1
1
Set merging index to 0
decode merging index
End