(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2017년03월21일
(11) 등록번호 10-1718642
(24) 등록일자 2017년03월15일

(51) 국제특허분류(Int. Cl.)
H04N 19/597 (2014.01) *H04N 13/00* (2016.01)
H04N 19/573 (2014.01) *H04N 19/70* (2014.01)
(52) CPC특허분류
H04N 19/597 (2015.01)
H04N 13/0011 (2013.01)
(21) 출원번호 10-2015-7021852
(22) 출원일자(국제) 2014년01월17일
심사청구일자 2016년04월05일
(85) 번역문제출일자 2015년08월12일
(65) 공개번호 10-2015-0106944
(43) 공개일자 2015년09월22일
(86) 국제출원번호 PCT/US2014/012047
(87) 국제공개번호 WO 2014/113669
국제공개일자 2014년07월24일
(30) 우선권주장
61/753,876 2013년01월17일 미국(US)
14/157,401 2014년01월16일 미국(US)
(56) 선행기술조사문헌
Multi-loop quality scalability based on high efficiency video coding, 2012 Picture coding symposium, pp. 445-448 (2012.05.07.)*
*는 심사관에 의하여 인용된 문헌

(73) 특허권자
퀄컴 인코포레이티드
미국 92121-1714 캘리포니아주 샌 디에고 모어하우스 드라이브 5775
(72) 발명자
천 잉
미국 92121-1714 캘리포니아주 샌디에고 모어하우스 드라이브 5775
강 제원
미국 92121-1714 캘리포니아주 샌디에고 모어하우스 드라이브 5775
장 리
미국 92121-1714 캘리포니아주 샌디에고 모어하우스 드라이브 5775
(74) 대리인
특허법인코리아나

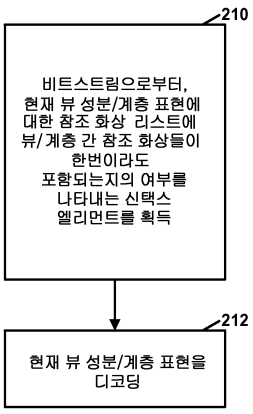
전체 청구항 수 : 총 25 항 심사관 : 장석환

(54) 발명의 명칭 **비디오 코딩에서 참조 화상 리스트에 대한 뷰 간 예측의 불가능**

(57) 요 약

비디오 코더가, 비트스트림에서, 현재 뷰 성분/계층 표현에 대한 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번이라도 포함되는지의 여부를 나타내는 신택스 엘리먼트를 시그널링한다. 비디오 디코더가, 비트스트림으로부터, 현재 뷰 성분/계층 표현에 대한 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번이라도 포함되는지의 여부를 나타내는 신택스 엘리먼트를 획득한다. 비디오 디코더는 현재 뷰 성분/계층 표현을 디코딩한다.

대 표 도 - 도8b



(52) CPC특허분류
H04N 13/0048 (2013.01)
H04N 19/573 (2015.01)
H04N 19/58 (2015.01)
H04N 19/70 (2015.01)

명 세 서

청구범위

청구항 1

비디오 데이터를 디코딩하는 방법으로서,

비트스트림에서의 비디오 파라미터 세트 (VPS) 로부터, 상기 VPS를 참조하는 임의의 코딩된 비디오 시퀀스 (CVS) 의 임의의 뷰 성분/계층 표현에 대한 제 2 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번이라도 포함되는지의 여부를 나타내는 신택스 엘리먼트를 획득하는 단계;

참조 화상 세트에서의 화상들에 기초하여, 상기 VPS를 참조하는 CVS의 현재 뷰 성분/계층 표현에 대한 제 1 참조 화상 리스트 및 제 2 참조 화상 리스트를 구축하는 단계로서, 적어도 하나의 뷰/계층 간 참조 화상은 상기 현재 뷰 성분/계층 표현에 대한 상기 제 1 참조 화상 리스트에 존재하는, 상기 현재 뷰 성분/계층 표현에 대한 제 1 참조 화상 리스트 및 제 2 참조 화상 리스트를 구축하는 단계; 및

상기 현재 뷰 성분/계층 표현을 디코딩하는 단계로서, 상기 VPS를 참조하는 임의의 CVS의 임의의 뷰 성분/계층 표현에 대한 상기 제 2 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번도 포함되지 않음을 상기 신택스 엘리먼트가 나타내는 경우, 상기 현재 뷰 성분/계층 표현은 뷰/계층 간 참조 화상들을 위하여 상기 현재 뷰 성분/계층 표현과 관련된 상기 제 2 참조 화상 리스트를 체크하지 않고 디코딩되는, 상기 현재 뷰 성분/계층 표현을 디코딩하는 단계를 포함하는, 비디오 데이터를 디코딩하는 방법.

청구항 2

제 1 항에 있어서,

상기 현재 뷰 성분/계층 표현을 디코딩하는 단계는 :

상기 현재 뷰 성분/계층 표현의 현재 블록에 대한 디스패리티 벡터를 결정하기 위하여 상기 현재 블록에 이웃하는 하나 이상의 블록들을 체크하는 디스패리티 벡터 도출 프로세스를 수행하는 단계를 포함하며, 그리고

상기 디스패리티 벡터 도출 프로세스를 수행하는 단계는 :

상기 VPS를 참조하는 임의의 코딩된 비디오 시퀀스의 임의의 뷰 성분/계층 표현에 대한 상기 제 2 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번도 포함되지 않음을 상기 신택스 엘리먼트가 나타내는 경우, 상기 현재 뷰 성분/계층 표현에 대한 상기 제 2 참조 화상 리스트에 대응하는 모션 정보를 체크하지 않는 단계를 포함하는, 비디오 데이터를 디코딩하는 방법.

청구항 3

제 1 항에 있어서,

상기 현재 뷰 성분/계층 표현은 상기 비트스트림에서 복수의 계층들에서의 특정 계층에 있으며,

상기 비디오 데이터를 디코딩하는 방법은 :

상기 복수의 계층들로부터의 각각의 개별 계층에 대해, 상기 개별 계층에 대한 개별 신택스 엘리먼트를 획득하는 단계로서, 상기 개별 신택스 엘리먼트는 상기 개별 계층에 있고 상기 VPS를 참조하는 임의의 CVS에 있는 뷰 성분들/계층 표현들에 대한 개별 제 2 참조 화상 리스트들에 뷰/계층 간 참조 화상들이 한번이라도 포함되는지의 여부를 나타내는, 상기 개별 계층에 대한 개별 신택스 엘리먼트를 획득하는 단계를 포함하는, 비디오 데이터를 디코딩하는 방법.

청구항 4

제 1 항에 있어서,

상기 비트스트림으로부터, 상기 현재 뷰 성분/계층 표현에 대한 상기 제 1 참조 화상 리스트를 수정하기 위한

참조 화상 리스트 수정 (reference picture list modification; RPLM) 신택스 엘리먼트들을 획득하는 단계를 더 포함하며,

상기 VPS를 참조하는 임의의 CVS의 임의의 뷰 성분/계층 표현에 대한 상기 제 2 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번도 포함되지 않음을 상기 신택스 엘리먼트가 나타내는 경우, 상기 RPLM 신택스 엘리먼트들의 각각은, 상기 VPS를 참조하는 코딩된 비디오 시퀀스들의 뷰 성분/계층 표현들에 대한 상기 제 2 참조 화상 리스트에 뷰/계층 간 참조 화상들이 포함될 수 있음을 상기 신택스 엘리먼트가 나타내는 경우보다 더 적은 비트들을 포함하는, 비디오 데이터를 디코딩하는 방법.

청구항 5

제 1 항에 있어서,

상기 신택스 엘리먼트는 제 1 신택스 엘리먼트이며,

상기 비디오 데이터를 디코딩하는 방법은 :

상기 비트스트림으로부터, 제 2 신택스 엘리먼트를 획득하는 단계로서, 상기 제 2 신택스 엘리먼트는 상기 현재 뷰 성분/계층 표현에 대한 상기 제 1 참조 화상 리스트에서의 뷰/계층 간 참조 화상들의 시작 포지션을 나타내는, 상기 제 2 신택스 엘리먼트를 획득하는 단계를 더 포함하는, 비디오 데이터를 디코딩하는 방법.

청구항 6

제 5 항에 있어서,

상기 VPS를 참조하는 코딩된 비디오 시퀀스들의 뷰 성분/계층 표현들에 대한 상기 제 2 참조 화상 리스트들에 뷰/계층 간 참조 화상들이 포함될 수 있음을 상기 제 1 신택스 엘리먼트가 나타내는 경우, 상기 비트스트림으로부터, 제 3 신택스 엘리먼트를 획득하는 단계로서, 상기 제 3 신택스 엘리먼트는 상기 현재 뷰 성분/계층 표현에 대한 상기 제 2 참조 화상 리스트에서의 뷰/계층 간 참조 화상들의 시작 포지션을 나타내는, 상기 제 3 신택스 엘리먼트를 획득하는 단계를 더 포함하는, 비디오 데이터를 디코딩하는 방법.

청구항 7

제 1 항에 있어서,

상기 현재 뷰 성분/계층 표현을 디코딩하는 단계는 :

상기 현재 뷰 성분/계층 표현의 현재 블록에 대한 디스패리티 벡터를 결정하기 위하여 상기 현재 블록에 이웃하는 하나 이상의 블록들을 체크하는 디스패리티 벡터 도출 프로세스를 수행하는 단계를 포함하며,

상기 디스패리티 벡터 도출 프로세스를 수행하는 단계는 :

상기 VPS를 참조하는 임의의 CVS의 임의의 뷰 성분/계층 표현에 대한 상기 제 2 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번도 포함되지 않음을 상기 신택스 엘리먼트가 나타내는 경우, 상기 현재 블록에 이웃하는 상기 하나 이상의 블록들의 각각에 대해 많아야 하나의 암시적 디스패리티 벡터를 저장하는 단계를 포함하는, 비디오 데이터를 디코딩하는 방법.

청구항 8

제 1 항에 있어서,

상기 현재 뷰 성분/계층 표현을 디코딩하는 단계는 :

상기 VPS를 참조하는 임의의 코딩된 비디오 시퀀스의 임의의 뷰 성분/계층 표현에 대한 상기 제 2 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번도 포함되지 않음을 상기 신택스 엘리먼트가 나타내는 경우, 후보 리스트에, 뷰/계층 간 참조 화상에 대응하는 후보를 포함시키지 않는 단계; 및

상기 후보 리스트에서의 특정 후보에 기초하여, 상기 현재 뷰 성분/계층 표현의 현재 블록에 대한 모션 벡터를 결정하는 단계를 포함하는, 비디오 데이터를 디코딩하는 방법.

청구항 9

제 1 항에 있어서,

상기 VPS를 참조하는 임의의 CVS의 임의의 뷰 성분/계층 표현에 대한 상기 제 2 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번도 포함되지 않음을 상기 신택스 엘리먼트가 나타내는 경우, 상기 현재 뷰 성분/계층 표현의 상기 제 2 참조 화상 리스트로부터의 참조 화상이 뷰/계층 간 참조 화상인지의 여부의 체킹을 피하는 단계; 및

상기 VPS를 참조하는 임의의 CVS의 임의의 뷰 성분/계층 표현에 대한 상기 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번도 포함되지 않음을 상기 신택스 엘리먼트가 나타내는 경우, 상기 현재 뷰 성분/계층 표현의 현재 코딩 유닛 (coding unit; CU) 의 예측 유닛 (prediction unit; PU) 이 상기 현재 뷰 성분/계층 표현에 대한 상기 제 2 참조 화상 리스트에서의 특정 참조 화상 내의 로케이션을 나타내는 모션 벡터를 갖는다면, 상기 특정 참조 화상의 유형을 체크하는 일 없이, 상기 현재 뷰 성분/계층 표현에 대한 상기 제 2 참조 화상 리스트에 대한 잔차 예측자 생성 프로세스를 가능하게 하는 단계를 더 포함하는, 비디오 데이터를 디코딩하는 방법.

청구항 10

제 1 항에 있어서,

상기 VPS를 참조하는 임의의 CVS의 임의의 뷰 성분/계층 표현에 대한 상기 제 2 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번도 포함되지 않음을 상기 신택스 엘리먼트가 나타내는 경우, 상기 현재 뷰 성분/계층 표현에 대한 상기 제 1 참조 화상 리스트에 삽입된 뷰/계층 간 참조 화상들만을 사용하여 뷰 합성 예측을 수행하는 단계; 및

상기 VPS를 참조하는 임의의 CVS의 임의의 뷰 성분/계층 표현에 대한 상기 제 2 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번도 포함되지 않음을 상기 신택스 엘리먼트가 나타내는 경우, 상기 현재 뷰 성분/계층 표현에 대한 상기 제 2 참조 화상 리스트의 초기 버전을 구축할 때 뷰/계층 간 참조 화상 세트 또는 뷰/계층 간 참조 화상들을 고려하지 않는 단계를 더 포함하는, 비디오 데이터를 디코딩하는 방법.

청구항 11

비디오 데이터를 인코딩하는 방법으로서,

참조 화상 세트에서의 화상들에 기초하여, 현재 뷰 성분/계층 표현에 대한 제 1 참조 화상 리스트 및 제 2 참조 화상 리스트를 구축하는 단계로서, 상기 현재 뷰 성분/계층 표현은 비디오 파라미터 세트 (VPS) 를 참조하는 코딩된 비디오 시퀀스 (CVS) 에 속하며, 적어도 하나의 뷰/계층 간 참조 화상은 상기 현재 뷰 성분/계층 표현에 대한 상기 제 1 참조 화상 리스트에 존재하는, 상기 현재 뷰 성분/계층 표현에 대한 제 1 참조 화상 리스트 및 제 2 참조 화상 리스트를 구축하는 단계;

비트스트림에서의 상기 VPS에서, 상기 VPS를 참조하는 임의의 CVS의 임의의 뷰 성분/계층 표현에 대한 제 2 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번이라도 포함되는지의 여부를 나타내는 신택스 엘리먼트를 시그널링하는 단계; 및

상기 현재 뷰 성분/계층 표현을 인코딩하는 단계로서, 상기 VPS를 참조하는 임의의 CVS의 임의의 뷰 성분/계층 표현에 대한 상기 제 2 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번도 포함되지 않음을 상기 신택스 엘리먼트가 나타내는 경우, 상기 현재 뷰 성분/계층 표현은 뷰/계층 간 참조 화상들을 위하여 상기 현재 뷰 성분/계층 표현과 관련된 상기 제 2 참조 화상 리스트를 체크하여 인코딩되지 않는, 상기 현재 뷰 성분/계층 표현을 인코딩하는 단계를 포함하는, 비디오 데이터를 인코딩하는 방법.

청구항 12

제 11 항에 있어서,

상기 현재 뷰 성분/계층 표현은 상기 비트스트림에서 복수의 계층들에서의 특정 계층에 있으며,

상기 비디오 데이터를 인코딩하는 방법은 :

상기 VPS에서, 상기 복수의 계층들로부터의 각각의 개별 계층에 대해, 상기 개별 계층에 대한 개별 신택스 엘리먼트를 시그널링하는 단계로서, 상기 개별 신택스 엘리먼트는 상기 개별 계층에 있고 상기 VPS를 참조하는 임의의 CVS에 있는 뷰 성분들/계층 표현들에 대한 개별 제 2 참조 화상 리스트들에 뷰/계층 간 참조 화상들이 한번이라도 포함되는지의 여부를 나타내는, 상기 개별 계층에 대한 개별 신택스 엘리먼트를 시그널링하는 단계를 포

함하는, 비디오 데이터를 인코딩하는 방법.

청구항 13

제 11 항에 있어서,

상기 비트스트림에서, 상기 현재 뷰 성분/계층 표현에 대한 상기 제 2 참조 화상 리스트를 수정하기 위한 참조 화상 리스트 수정 (reference picture list modification; RPLM) 신택스 엘리먼트들을 시그널링하는 단계를 더 포함하며,

상기 VPS를 참조하는 임의의 CVS의 임의의 뷰 성분/계층 표현에 대한 상기 제 2 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번도 포함되지 않음을 상기 신택스 엘리먼트가 나타내는 경우, 상기 RPLM 신택스 엘리먼트들의 각각은, 상기 VPS를 참조하는 임의의 CVS의 뷰 성분/계층 표현들에 대한 상기 제 2 참조 화상 리스트들에 뷰/계층 간 참조 화상들이 포함될 수 있음을 상기 신택스 엘리먼트가 나타내는 경우보다 더 적은 비트들을 포함하는, 비디오 데이터를 인코딩하는 방법.

청구항 14

제 11 항에 있어서,

상기 신택스 엘리먼트는 제 1 신택스 엘리먼트이며,

상기 비디오 데이터를 인코딩하는 방법은 :

상기 비트스트림에서, 제 2 신택스 엘리먼트를 시그널링하는 단계로서, 상기 제 2 신택스 엘리먼트는 상기 현재 뷰 성분/계층 표현에 대한 상기 제 1 참조 화상 리스트에서의 뷰/계층 간 참조 화상들의 시작 포지션을 나타내는, 상기 제 2 신택스 엘리먼트를 시그널링하는 단계를 더 포함하는, 비디오 데이터를 인코딩하는 방법.

청구항 15

비디오 디코딩 디바이스로서,

비일시적 저장 매체 및 상기 비일시적 저장 매체에 커플링된 하나 이상의 프로세서들을 포함하며,

상기 하나 이상의 프로세서들은 :

비트스트림에서의 비디오 파라미터 세트 (VPS) 로부터, 상기 VPS를 참조하는 임의의 코딩된 비디오 시퀀스 (CVS) 의 임의의 뷰 성분/계층 표현에 대한 제 2 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번이라도 포함되는지의 여부를 나타내는 신택스 엘리먼트를 획득하고;

참조 화상 세트에서의 화상들에 기초하여, 상기 VPS를 참조하는 CVS의 현재 뷰 성분/계층 표현에 대한 제 1 참조 화상 리스트 및 제 2 참조 화상 리스트를 구축하는 것으로서, 적어도 하나의 뷰/계층 간 참조 화상은 상기 현재 뷰 성분/계층 표현에 대한 상기 제 1 참조 화상 리스트에 존재하는, 상기 현재 뷰 성분/계층 표현에 대한 제 1 참조 화상 리스트 및 제 2 참조 화상 리스트를 구축하며; 그리고

상기 현재 뷰 성분/계층 표현을 디코딩하는 것으로서, 상기 VPS를 참조하는 임의의 CVS의 임의의 뷰 성분/계층 표현에 대한 상기 제 2 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번도 포함되지 않음을 상기 신택스 엘리먼트가 나타내는 경우, 상기 현재 뷰 성분/계층 표현은 뷰/계층 간 참조 화상들을 위하여 상기 현재 뷰 성분/계층 표현과 관련된 상기 제 2 참조 화상 리스트를 체크하지 않고 디코딩되는, 상기 현재 뷰 성분/계층 표현을 디코딩하도록 구성되는, 비디오 디코딩 디바이스.

청구항 16

제 15 항에 있어서,

상기 하나 이상의 프로세서들은 :

상기 현재 뷰 성분/계층 표현의 현재 블록에 대한 디스패리티 벡터를 결정하기 위하여 상기 현재 블록에 이웃하는 하나 이상의 블록들을 체크하는 디스패리티 벡터 도출 프로세스를 수행하도록 구성되며, 그리고

상기 하나 이상의 프로세서들은, 상기 디스패리티 벡터 도출 프로세스를 수행하는 것의 일부로서, 상기 VPS를 참조하는 임의의 CVS의 임의의 뷰 성분/계층 표현에 대한 상기 제 2 참조 화상 리스트에 뷰/계층 간 참조 화상

들이 한번도 포함되지 않음을 상기 신택스 엘리먼트가 나타내는 경우, 상기 하나 이상의 프로세서들이 상기 현재 뷰 성분/계층 표현에 대한 상기 제 2 참조 화상 리스트에 대응하는 모션 정보를 체크하지 않도록 구성되는, 비디오 디코딩 디바이스.

청구항 17

제 15 항에 있어서,

상기 현재 뷰 성분/계층 표현은 상기 비트스트림에서 복수의 계층들에서의 특정 계층에 있으며,

상기 하나 이상의 프로세서들은 :

상기 복수의 계층들로부터의 각각의 개별 계층에 대해, 상기 개별 계층에 대한 개별 신택스 엘리먼트를 획득하는 것으로서, 상기 개별 신택스 엘리먼트는 상기 개별 계층에 있고 상기 VPS를 참조하는 임의의 CVS에 있는 뷰 성분들/계층 표현들에 대한 개별 제 2 참조 화상 리스트들에 뷰/계층 간 참조 화상들이 한번이라도 포함되는지의 여부를 나타내는, 상기 개별 계층에 대한 개별 신택스 엘리먼트를 획득하도록 구성되는, 비디오 디코딩 디바이스.

청구항 18

제 15 항에 있어서,

상기 하나 이상의 프로세서들은 또한, 상기 비트스트림으로부터, 상기 현재 뷰 성분/계층 표현에 대한 상기 제 2 참조 화상 리스트를 수정하기 위한 참조 화상 리스트 수정 (reference picture list modification; RPLM) 신택스 엘리먼트들을 획득하도록 구성되며,

상기 VPS를 참조하는 임의의 CVS의 임의의 뷰 성분/계층 표현에 대한 상기 제 2 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번도 포함되지 않음을 상기 신택스 엘리먼트가 나타내는 경우, 상기 RPLM 신택스 엘리먼트들의 각각은, 상기 VPS를 참조하는 임의의 코딩된 비디오 시퀀스의 뷰 성분/계층 표현들에 대한 상기 제 2 참조 화상 리스트들에 뷰/계층 간 참조 화상들이 포함될 수 있음을 상기 신택스 엘리먼트가 나타내는 경우보다 더 적은 비트들을 포함하는, 비디오 디코딩 디바이스.

청구항 19

제 15 항에 있어서,

상기 신택스 엘리먼트는 제 1 신택스 엘리먼트이며,

상기 하나 이상의 프로세서들은 상기 비트스트림으로부터, 제 2 신택스 엘리먼트를 획득하는 것으로서, 상기 제 2 신택스 엘리먼트는 상기 현재 뷰 성분/계층 표현에 대한 상기 제 1 참조 화상 리스트에서의 뷰/계층 간 참조 화상들의 시작 포지션을 나타내는, 상기 제 2 신택스 엘리먼트를 획득하도록 구성되는, 비디오 디코딩 디바이스.

청구항 20

제 15 항에 있어서,

상기 하나 이상의 프로세서들은 :

상기 현재 뷰 성분/계층 표현의 현재 블록에 대한 디스패리티 벡터를 결정하기 위하여 상기 현재 블록에 이웃하는 하나 이상의 블록들을 체크하는 디스패리티 벡터 도출 프로세스를 수행하도록 구성되며, 그리고

상기 하나 이상의 프로세서들은, 상기 디스패리티 벡터 도출 프로세스를 수행하는 것의 일부로서, 상기 VPS를 참조하는 임의의 CVS의 임의의 뷰 성분/계층 표현에 대한 상기 제 2 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번도 포함되지 않음을 상기 신택스 엘리먼트가 나타내는 경우, 상기 하나 이상의 프로세서들이 상기 현재 블록에 이웃하는 상기 하나 이상의 블록들의 각각에 대해 많아야 하나의 암시적 디스패리티 벡터를 저장하도록 구성되는, 비디오 디코딩 디바이스.

청구항 21

제 15 항에 있어서,

상기 하나 이상의 프로세서들은 :

상기 VPS를 참조하는 임의의 코딩된 비디오 시퀀스의 임의의 뷰 성분/계층 표현에 대한 상기 제 2 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번도 포함되지 않음을 상기 신택스 엘리먼트가 나타내는 경우, 후보 리스트에, 뷰/계층 간 참조 화상에 대응하는 후보를 포함시키지 않으며; 그리고

상기 후보 리스트에서의 특정 후보에 기초하여, 상기 현재 뷰 성분/계층 표현의 현재 블록에 대한 모션 벡터를 결정하도록 구성되는, 비디오 디코딩 디바이스.

청구항 22

제 15 항에 있어서,

상기 하나 이상의 프로세서들은 :

상기 VPS를 참조하는 임의의 CVS의 임의의 뷰 성분/계층 표현에 대한 상기 제 2 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번도 포함되지 않음을 상기 신택스 엘리먼트가 나타내는 경우, 상기 현재 뷰 성분/계층 표현에 대한 상기 제 2 참조 화상 리스트로부터의 참조 화상이 뷰/계층 간 참조 화상인지의 여부의 체킹을 피하며; 그리고

상기 VPS를 참조하는 임의의 CVS의 임의의 뷰 성분/계층 표현에 대한 상기 제 2 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번도 포함되지 않음을 상기 신택스 엘리먼트가 나타내는 경우, 상기 현재 뷰 성분/계층 표현의 현재 코딩 유닛 (coding unit; CU) 의 예측 유닛 (prediction unit; PU) 이 상기 현재 뷰 성분/계층 표현에 대한 상기 제 2 참조 화상 리스트에서의 특정 참조 화상 내의 로케이션을 나타내는 모션 벡터를 갖는다면, 상기 특정 참조 화상의 유형을 체크하는 일 없이, 상기 현재 뷰 성분/계층 표현에 대한 상기 제 2 참조 화상 리스트에 대한 잔차 예측자 생성 프로세스를 가능하게 하도록 구성되는, 비디오 디코딩 디바이스.

청구항 23

제 15 항에 있어서,

상기 하나 이상의 프로세서들은 :

상기 VPS를 참조하는 임의의 CVS의 임의의 뷰 성분/계층 표현에 대한 상기 제 2 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번도 포함되지 않음을 상기 신택스 엘리먼트가 나타내는 경우, 상기 현재 뷰 성분/계층 표현에 대한 상기 제 1 참조 화상 리스트에 삽입된 뷰/계층 간 참조 화상들만을 사용하여 뷰 합성 예측을 수행하며; 그리고

상기 VPS를 참조하는 임의의 CVS의 임의의 뷰 성분/계층 표현에 대한 상기 제 2 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번도 포함되지 않음을 상기 신택스 엘리먼트가 나타내는 경우, 상기 현재 뷰 성분/계층 표현에 대한 상기 제 2 참조 화상 리스트의 초기 버전을 구축할 때 뷰/계층 간 참조 화상 세트 또는 뷰/계층 간 참조 화상들을 고려하지 않도록 구성되는, 비디오 디코딩 디바이스.

청구항 24

비디오 디코딩 디바이스로서,

비트스트림에서의 비디오 파라미터 세트 (VPS) 로부터, 상기 VPS를 참조하는 임의의 코딩된 비디오 시퀀스 (CVS) 의 임의의 뷰 성분/계층 표현에 대한 제 2 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번이라도 포함되는지의 여부를 나타내는 신택스 엘리먼트를 획득하는 수단;

참조 화상 세트에서의 화상들에 기초하여, 상기 VPS를 참조하는 CVS의 현재 뷰 성분/계층 표현에 대한 제 1 참조 화상 리스트 및 제 2 참조 화상 리스트를 구축하는 수단으로서, 적어도 하나의 뷰/계층 간 참조 화상은 상기 현재 뷰 성분/계층 표현에 대한 상기 제 1 참조 화상 리스트에 존재하는, 상기 현재 뷰 성분/계층 표현에 대한 제 1 참조 화상 리스트 및 제 2 참조 화상 리스트를 구축하는 수단; 및

상기 현재 뷰 성분/계층 표현을 디코딩하는 수단으로서, 상기 VPS를 참조하는 임의의 CVS의 임의의 뷰 성분/계층 표현에 대한 상기 제 2 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번도 포함되지 않음을 상기 신택스 엘리먼트가 나타내는 경우, 상기 현재 뷰 성분/계층 표현은 뷰/계층 간 참조 화상들을 위하여 상기 현재 뷰 성분/계층 표현과 관련된 상기 제 2 참조 화상 리스트를 체크하지 않고 디코딩되는, 상기 현재 뷰 성분/계층 표현

을 디코딩하는 수단을 포함하는, 비디오 디코딩 디바이스.

청구항 25

제 15 항에 있어서,

상기 비디오 디코딩 디바이스는 :

집적 회로; 및

마이크로프로세서

중 적어도 하나를 포함하는, 비디오 디코딩 디바이스.

청구항 26

삭제

청구항 27

삭제

발명의 설명

기 술 분 야

[0001] 본 출원은 2013년 1월 17일자로 출원된 미국 가특허 출원 제61/753,876호를 우선권 주장하며, 그 전체 내용은 참조로 본원에 통합된다.

[0002] 기술 분야

[0003] 본 개시물은 비디오 인코딩 및 디코딩에 관한 것이다.

배 경 기 술

[0004] 디지털 비디오 능력들은 디지털 텔레비전들, 디지털 직접 브로드캐스트 시스템들, 무선 브로드캐스트 시스템들, 개인 정보 단말기들 (PDA들), 랩톱 또는 데스크톱 컴퓨터들, 태블릿 컴퓨터들, e-북 리더들, 디지털 카메라들, 디지털 레코딩 디바이스들, 디지털 미디어 플레이어들, 비디오 게이밍 디바이스들, 비디오 게임 콘솔들, 셀룰러 또는 위성 무선 전화기들, 이른바 "스마트 폰들", 비디오 원격회의 디바이스들, 비디오 스트리밍 디바이스들 등을 포함한 넓은 범위의 디바이스들에 통합될 수 있다. 디지털 비디오 디바이스들은 MPEG-2, MPEG-4, ITU-T H.263, ITU-T H.264/MPEG-4, 파트 10, 고급 비디오 코딩 (Advanced Video Coding; AVC) 에 의해 규정된 표준들, 현재 개발중인 고 효율 비디오 코딩 (High Efficiency Video Coding; HEVC) 표준, 및 이러한 표준들의 확장본들에 기재된 것들과 같은 비디오 압축 기법들을 구현한다. 비디오 디바이스들은 이러한 비디오 압축 기법들을 구현하는 것에 의해 디지털 비디오 정보를 더 효율적으로 송신, 수신, 인코딩, 디코딩, 및/또는 저장할 수도 있다.

[0005] 비디오 압축 기법들은 공간적 (화상 내) 예측 및/또는 시간적 (화상 간) 예측을 수행하여 비디오 시퀀스들에 내재하는 리던던시를 감소시키거나 또는 제거한다. 블록 기반 비디오 코딩의 경우, 비디오 슬라이스 (즉, 비디오 프레임 또는 비디오 프레임의 부분) 가 블록들로 구획될 수도 있다. 화상의 인트라 코딩식 (intra-coded; I) 슬라이스에서의 블록들은 동일한 화상의 이웃 블록들에서의 참조 샘플들에 관한 공간적 예측을 이용하여 인코딩된다. 화상의 인터 코딩식 (inter-coded; P 또는 B) 슬라이스에서의 블록들은 동일한 화상의 이웃 블록들에서의 참조 샘플들에 관한 공간적 예측 또는 다른 참조 화상들에서의 참조 샘플들에 관한 시간적 예측을 이용할 수도 있다. 화상들은 프레임들이라고 지칭될 수도 있고, 참조 화상들은 참조 프레임들이라고 지칭될 수도 있다.

[0006] 공간적 또는 시간적 예측은 코딩될 블록에 대한 예측 블록을 초래한다. 잔차 데이터는 코딩될 원본 블록과 예측 블록 사이의 화소 차이들을 나타낸다. 인터 코딩식 블록이 예측 블록을 형성하는 참조 샘플들의 블록을 가리키는 모션 벡터에 따라 인코딩되고, 잔차 데이터는 코딩된 블록 및 예측 블록 사이의 차이를 나타낸다. 인트라 코딩식 블록이 인트라 코딩 모드 및 잔차 데이터에 따라 인코딩된다. 추가 압축을 위해, 잔차 데이터는 화소 도메인으로부터 변환 도메인으로 변환될 수도 있으며, 결과적으로 잔차 계수들이 생겨나며, 그 계

수들은 그 다음에 양자화될 수도 있다. 처음에는 2차원 어레이로 배열된 양자화된 계수들은, 계수들의 1차원 벡터를 생성하기 위하여 스캐닝될 수도 있고, 엔트로피 코딩이 더 많은 압축을 달성하기 위해 적용될 수도 있다.

[0007] 멀티뷰 코딩 비트스트림이, 예컨대, 다수의 관점들에서 뷰들을 인코딩함으로써 생성될 수도 있다. 멀티뷰 코딩 양태들을 이용하는 일부 3차원 (3D) 비디오 표준들이 개발되어 있다. 예를 들어, 상이한 뷰들이 3D 비디오를 지원하기 위해 좌안 및 우안 뷰들을 송신할 수도 있다. 대안으로, 일부 3D 비디오 코딩 프로세스들이 이른바 멀티뷰 플러스 깊이 (multi-view plus depth) 코딩을 적용할 수도 있다. 멀티뷰 플러스 깊이 코딩에서, 3D 비디오 비트스트림이 텍스처 뷰 성분들뿐만 아니라, 깊이 뷰 성분들도 포함할 수도 있다. 예를 들어, 각각의 뷰는 하나의 텍스처 뷰 성분과 하나의 깊이 뷰 성분을 포함할 수도 있다.

발명의 내용

과제의 해결 수단

[0008] 대체로, 본 개시물은 비디오 코딩, 이를테면 비디오 인코더들 및 디코더들의 분야에 관련한다. 일부 예들은 고 효율 비디오 코딩 (HEVC) 코덱을 이용한 2 개 이상의 뷰들의 코딩을 포함한 고급 코덱들에 기초한 멀티-뷰 비디오 코딩에 관련한다. 더 구체적으로는, 일부 예들에서, 비디오 디코더는, 비트스트림으로부터, 현재 뷰 성분/계층 표현에 대한 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번이라도 (ever) 포함되는지의 여부를 나타내는 신택스 엘리먼트를 획득할 수도 있다. 이러한 예들에서, 비디오 코더는 현재 뷰 성분/계층 표현을 디코딩할 수도 있다. 더욱이, 일부 예들은 디스패리티 벡터 생성에 관련한다.

[0009] 하나의 예에서, 본 개시물은 비디오 데이터를 디코딩하는 방법을 설명하는데, 그 방법은, 비트스트림으로부터, 현재 뷰 성분/계층 표현에 대한 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번이라도 포함되는지의 여부를 나타내는 신택스 엘리먼트를 획득하는 단계; 및 현재 뷰 성분/계층 표현을 디코딩하는 단계로서, 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번도 포함되지 않음을 신택스 엘리먼트가 나타내는 경우, 현재 뷰 성분/계층 표현은 참조 화상 리스트에서의 뷰/계층 간 참조 화상들의 사용 없이 디코딩되는, 상기 현재 뷰 성분/계층 표현을 디코딩하는 단계를 포함한다.

[0010] 다른 예에서, 본 개시물은 비디오 데이터를 인코딩하는 방법을 설명하는데, 그 방법은, 비트스트림에서, 현재 뷰 성분/계층 표현에 대한 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번이라도 포함되는지의 여부를 나타내는 신택스 엘리먼트를 시그널링하는 단계; 및 현재 뷰 성분/계층 표현을 인코딩하는 단계로서, 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번도 포함되지 않음을 신택스 엘리먼트가 나타내는 경우, 현재 뷰 성분/계층 표현은 참조 화상 리스트에서의 뷰/계층 간 참조 화상들을 사용하여 인코딩되지 않는, 상기 현재 뷰 성분/계층 표현을 인코딩하는 단계를 포함한다.

[0011] 다른 예에서, 본 개시물은 저장 매체 및 그 저장 매체에 커플링된 하나 이상의 프로세서들을 포함하는 비디오 디코딩 디바이스를 설명하는데, 하나 이상의 프로세서들은, 비트스트림으로부터, 현재 뷰 성분/계층 표현에 대한 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번이라도 포함되는지의 여부를 나타내는 신택스 엘리먼트를 획득하며; 그리고 현재 뷰 성분/계층 표현을 디코딩하는 것으로서, 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번도 포함되지 않음을 신택스 엘리먼트가 나타내는 경우, 현재 뷰 성분/계층 표현은 참조 화상 리스트에서의 뷰/계층 간 참조 화상들의 사용 없이 디코딩되는, 상기 현재 뷰 성분/계층 표현을 디코딩하도록 구성된다.

[0012] 다른 예에서, 본 개시물은 비디오 디코딩 디바이스를 설명하는데, 그 디바이스는, 비트스트림으로부터, 현재 뷰 성분/계층 표현에 대한 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번이라도 포함되는지의 여부를 나타내는 신택스 엘리먼트를 획득하는 수단; 및 현재 뷰 성분/계층 표현을 디코딩하는 수단으로서, 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번도 포함되지 않음을 신택스 엘리먼트가 나타내는 경우, 현재 뷰 성분/계층 표현은 참조 화상 리스트에서의 뷰/계층 간 참조 화상들의 사용 없이 디코딩되는, 상기 현재 뷰 성분/계층 표현을 디코딩하는 수단을 포함한다.

[0013] 다른 예에서, 본 개시물은 명령들을 저장하고 있는 컴퓨터 판독가능 데이터 저장 매체 (예컨대, 비일시적 컴퓨터 판독가능 데이터 저장 매체) 를 설명하는데, 그 명령들은, 실행되는 경우, 비디오 디코딩 디바이스로 하여금, 비트스트림으로부터, 현재 뷰 성분/계층 표현에 대한 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번이라도 포함되는지의 여부를 나타내는 신택스 엘리먼트를 획득하게 하며; 그리고 현재 뷰 성분/계층 표현을 디코딩하게 하는 것으로서, 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번도 포함되지 않음을 신택스 엘리

먼트가 나타내는 경우, 현재 뷰 성분/계층 표현은 참조 화상 리스트에서의 뷰/계층 간 참조 화상들의 사용 없이 디코딩되는, 상기 현재 뷰 성분/계층 표현을 디코딩하게 한다.

[0014] 다른 예에서, 본 개시물은 저장 매체 및 그 저장 매체에 커플링된 하나 이상의 프로세서들을 포함하는 비디오 인코딩 디바이스를 설명하는데, 하나 이상의 프로세서들은, 비트스트림에서, 현재 뷰 성분/계층 표현에 대한 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번이라도 포함되는지의 여부를 나타내는 신택스 엘리먼트를 시그널링하며; 그리고 현재 뷰 성분/계층 표현을 인코딩하는 것으로서, 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번도 포함되지 않음을 신택스 엘리먼트가 나타내는 경우, 현재 뷰 성분/계층 표현은 참조 화상 리스트에서의 뷰/계층 간 참조 화상들을 사용하여 인코딩되지 않는, 상기 현재 뷰 성분/계층 표현을 인코딩하도록 구성된다.

[0015] 다른 예에서, 본 개시물은 비디오 인코딩 디바이스를 설명하는데, 그 디바이스는, 비트스트림에서, 현재 뷰 성분/계층 표현에 대한 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번이라도 포함되는지의 여부를 나타내는 신택스 엘리먼트를 시그널링하는 수단; 및 현재 뷰 성분/계층 표현을 인코딩하는 수단으로서, 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번도 포함되지 않음을 신택스 엘리먼트가 나타내는 경우, 현재 뷰 성분/계층 표현은 참조 화상 리스트에서의 뷰/계층 간 참조 화상들을 사용하여 인코딩되지 않는, 상기 현재 뷰 성분/계층 표현을 인코딩하는 수단을 포함한다.

[0016] 다른 예에서, 본 개시물은 명령들을 저장하고 있는 컴퓨터 판독가능 데이터 저장 매체 (예컨대, 비일시적 컴퓨터 판독가능 데이터 저장 매체) 를 설명하는데, 그 명령들은, 실행되는 경우, 비디오 인코딩 디바이스로 하여금, 비트스트림에서, 현재 뷰 성분/계층 표현에 대한 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번이라도 포함되는지의 여부를 나타내는 신택스 엘리먼트를 시그널링하게 하며; 그리고 현재 뷰 성분/계층 표현을 인코딩하게 하는 것으로서, 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번도 포함되지 않음을 신택스 엘리먼트가 나타내는 경우, 현재 뷰 성분/계층 표현은 참조 화상 리스트에서의 뷰/계층 간 참조 화상들을 사용하여 인코딩되지 않는, 상기 현재 뷰 성분/계층 표현을 인코딩하게 한다.

[0017] 본 개시물의 하나 이상의 예들의 상세는 첨부 도면들 및 다음의 설명에서 언급된다. 다른 특징들, 목적들, 및 장점들은 상세한 설명, 도면들, 및 청구항들로부터 명확하게 될 것이다.

도면의 간단한 설명

[0018] 도 1은 본 개시물에서 설명되는 기법들을 이용할 수도 있는 일 예의 비디오 코딩 시스템을 도시하는 블록도이다.

도 2는 현재 PU에 관하여 예의 공간적으로 이웃하는 예측 유닛 (prediction unit; PU) 들을 도시하는 개념도이다.

도 3은 멀티-뷰 코딩을 위한 일 예의 예측 구조를 도시하는 개념도이다.

도 4는 멀티-뷰 및 3D 비디오 코딩에서 고급 잔차 예측의 일 예의 예측 구조를 도시하는 개념도이다.

도 5는 시간적 후보 화상의 대응하는 PU에서의 시간적 이웃들을 도시하는 개념도이다.

도 6은 본 개시물의 기법들을 구현하도록 구성되는 일 예의 비디오 인코더를 도시하는 블록도이다.

도 7은 본 개시물의 기법들을 구현하도록 구성되는 일 예의 비디오 디코더를 도시하는 블록도이다.

도 8a는 본 개시물의 일 예에 따른, 비디오 인코더의 동작을 도시하는 흐름도이다.

도 8b는 본 개시물의 일 예에 따른, 비디오 디코더의 동작을 도시하는 흐름도이다.

도 9는 본 개시물의 일 예에 따른, 슬라이스 헤더를 파싱하는 일 예의 동작을 도시하는 흐름도이다.

도 10은 본 개시물의 일 예에 따른, 이웃 블록 기반 디스패리티 벡터 (neighboring block based disparity vector; NBDV) 도출 프로세스를 도시하는 흐름도이다.

발명을 실시하기 위한 구체적인 내용

[0019] 고 효율 비디오 코딩 (HEVC) 이 새로 개발된 비디오 코딩 표준이다. HEVC와 다른 비디오 코딩 사양들 또는 표준들에서, 비디오 인코더가 각각의 화상에 대해 2 개까지의 참조 화상 리스트들을 생성할 수도 있다. 이

들 참조 화상 리스트들은 RefPicList0 및 RefPicList1이라고 지칭될 수도 있다. 특정 화상에 대한 참조 화상 리스트들은 특정 화상과는 상이한 시간 인스턴스들에서 발생하는 화상들을 포함할 수도 있다. 다르게 말하면, 참조 화상 리스트들은 시간적 참조 화상들을 포함할 수도 있다.

[0020] 비디오 인코더가 화상을 인코딩하는 경우, 비디오 코더는 화상에서 블록들의 인코딩된 표현들을 생성할 수도 있다. 비디오 인코더는 화상의 블록의 인코딩된 표현을 생성하기 위해 인트라 예측 또는 인터 예측을 사용할 수도 있다. 다르게 말하면, 비디오 인코더는 블록을 인코딩하기 위해 인트라 예측 또는 인터 예측을 사용할 수도 있다. 비디오 인코더가 현재 화상의 현재 블록을 인코딩하기 위해 인트라 예측을 사용하는 경우, 비디오 인코더는 현재 화상에서의 다른 샘플들에 기초하여 예측 블록을 생성할 수도 있다. 비디오 인코더가 현재 블록을 인코딩하기 위해 인터 예측을 사용하는 경우, 비디오 인코더는, 현재 화상의 RefPicList0에서의 참조 화상의 대응 블록에 기초하여, 현재 화상의 RefPicList1에서의 참조 화상의 대응 블록에 기초하여, 또는 현재 화상의 RefPicList0에서의 참조 화상의 제 1 대응 블록 및 현재 화상의 RefPicList1에서의 참조 화상의 제 2 대응 블록에 기초하여 현재 블록에 대한 예측 블록을 결정할 수도 있다. 비디오 인코더가 인트라 예측 또는 인터 예측을 사용하여 현재 블록을 인코딩하는지의 여부에 상관 없이, 비디오 인코더는 현재 블록의 원래의 콘텐츠 및 예측 블록 간의 차이를 결정할 수도 있다. 비디오 인코더는 결과적인 잔차 데이터를 변환 및 양자화할 수도 있다. 비디오 코더는, 비트스트림에, 변환된 및 양자화된 잔차 데이터를 나타내는 엔트로피 인코딩된 신택스 엘리먼트들을 포함시킬 수도 있다.

[0021] 비디오 디코더가 화상 (즉, 현재 화상) 을 디코딩하는 경우, 비디오 디코더는 현재 화상에 대해 동일한 참조 화상 리스트들을 생성할 수도 있다. 덧붙여서, 비디오 디코더는 현재 화상의 각각의 블록을 디코딩할 수도 있다. 비디오 디코더가 현재 화상의 현재 블록을 디코딩하고 현재 블록이 인터 예측을 사용하여 인코딩되었던 경우, 비디오 디코더는, 현재 화상의 RefPicList0 및/또는 RefPicList1에서의 참조 화상들의 대응 블록들에 기초하여, 현재 블록에 대한 예측 블록을 결정할 수도 있다. 비디오 디코더는 그 다음에 현재 블록에 대한 잔차 데이터에 그 예측 블록을 가산함으로써 현재 블록을 복원할 수도 있다.

[0022] MV-HEVC는 멀티-뷰 (MV) 코딩에 대한 HEVC의 확장본이다. 3D-HEVC는 3차원 (3D) 비디오 데이터에 대한 HEVC의 확장본이다. MV-HEVC와 3D-HEVC는 상이한 관점들로부터 동일한 장면의 다수의 뷰들을 제공한다. MV-HEVC와 3D-HEVC에서, 상이한 뷰들로부터의 복원된 화상들에 기초한 뷰 간 예측이 가능하게 될 수도 있다. MV-HEVC와 3D-HEVC에서, 현재 화상에 대한 참조 화상 리스트들은 뷰 간 참조 화상들 뿐만 아니라 시간적 참조 화상들을 포함할 수도 있다. 뷰 간 참조 화상들은 현재 화상과는 상이한 뷰들에 있을 수도 있다. 비디오 인코더들과 비디오 디코더들은 시간적 참조 화상들과 유사한 방식으로 참조 화상 리스트들에서의 뷰 간 참조 화상들을 사용할 수도 있다.

[0023] MV-HEVC와 3D-HEVC에서, 비디오 인코더들 및 비디오 디코더들에 의해 사용된 일부 코딩 도구들은 RefPicList0 및 RefPicList1 양쪽 모두가 뷰 간 참조 화상들을 포함한다고 가정한다. 예를 들면, 블록에 대한 디스패리티 벡터를 결정하는 코딩 도구들은 RefPicList0 및 RefPicList1 양쪽 모두가 뷰 간 참조 화상들을 포함한다고 가정할 수도 있다. 이 가정의 결과로서, 이러한 코딩 도구들의 복잡도는 증가할 수도 있는데, 코딩 도구들은 참조 화상 리스트들 양쪽 모두에서의 참조 화상들이 뷰 간 참조 화상들인지의 여부를 체크할 필요가 있을 수도 있기 때문이다. 더욱이, RefPicList1에서의 참조 화상이 뷰 간 참조 화상인지의 여부를 체크하는 것은 메모리에 대한 부가적인 읽기 및 쓰기 요청들을 초래할 수도 있으며, 이는 인코딩 및/또는 디코딩 프로세스를 느려지게 할 수도 있다. 그러나, RefPicList0 및 RefPicList1에 동일한 뷰 간 참조 화상들을 포함시키는 것은 임의의 코딩 이득을 생겨나지 못하게 할 수도 있다.

[0024] 이에 따라, 어떤 경우들에서는, 뷰 간 참조 화상들을 RefPicList0에는 포함시키지만 RefPicList1에는 포함시키지 않는 것이 유익할 수도 있다. 따라서, 본 개시물의 특정 예들에 따라, 비디오 인코더가 비트스트림에서 RefPicList1에 뷰 간 참조 화상들이 포함되지 않음을 시그널링하면, 비디오 디코더에 의해 사용되는 특정 코딩 도구들은 RefPicList1에서의 참조 화상들이 뷰 간 참조 화상들인지의 여부를 체크할 필요가 없다. 이는 메모리에 대한 읽기 및 쓰기 요청들의 수와 복잡도를 감소시킬 수도 있다. 더욱이, 비디오 인코더가 비트스트림에서 RefPicList1에 뷰 간 참조 화상들이 포함되지 않음을 시그널링하면, 비디오 인코더는 비트스트림에서 특정 신택스 엘리먼트들을 시그널링할 필요가 없을 수도 있다. 예를 들면, RefPicList1에 뷰 간 참조 화상들이 없다면, 비디오 인코더는 더 적은 비트들을 포함하는 참조 화상 리스트 수정 (reference picture list modification; RPLM) 신택스 엘리먼트들을 사용하여 RefPicList1에서의 참조 화상들의 순서를 수정하는 방법을 시그널링할 수도 있다.

[0025] 따라서, 본 개시물의 일 예에 따라, 비디오 인코더는, 비트스트림에서, 현재 화상에 대한 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번이라도 (ever) 포함되는지의 여부를 나타내는 신택스 엘리먼트를 시그널링할 수도 있다. 덧붙여서, 비디오 인코더는 현재 화상을 인코딩할 수도 있다. 마찬가지로, 비디오 디코더는, 비트스트림으로부터, 현재 화상에 대한 참조 화상 리스트에 뷰 간 참조 화상들이 한번이라도 포함되는지의 여부를 나타내는 신택스 엘리먼트를 획득할 수도 있다. 덧붙여서, 비디오 디코더는 현재 화상을 디코딩할 수도 있다. 본원에서 설명된 바와 같은, 본 개시물의 예들은 스케일러블 비디오 코딩 (scalable video coding; SVC), 뿐만 아니라 멀티-뷰 코딩 및 3DV 코딩에 적용할 수도 있다.

[0026] 도 1은 본 개시물의 기법들을 이용할 수도 있는 일 예의 비디오 코딩 시스템 (10) 을 도시하는 블록도이다. 본원에서 사용되는 바와 같이, 용어 "비디오 코더"는 비디오 인코더들 및 비디오 디코더들 양쪽 모두를 일반적으로 지칭한다. 본 개시물에서, "비디오 코딩" 또는 "코딩"이란 용어들은 비디오 인코딩 또는 비디오 디코딩을 일반적으로 지칭할 수도 있다.

[0027] 도 1에 도시된 바와 같이, 비디오 코딩 시스템 (10) 은 소스 디바이스 (12) 와 목적지 디바이스 (14) 를 구비한다. 소스 디바이스 (12) 는 인코딩된 비디오 데이터를 생성한다. 이에 따라, 소스 디바이스 (12) 는 비디오 인코딩 디바이스 또는 비디오 인코딩 장치라고 지칭될 수도 있다. 목적지 디바이스 (14) 는 소스 디바이스 (12) 에 의해 생성된 인코딩된 비디오 데이터를 디코딩할 수도 있다. 이에 따라, 목적지 디바이스 (14) 는 비디오 디코딩 디바이스 또는 비디오 디코딩 장치라고 지칭될 수도 있다. 소스 디바이스 (12) 와 목적지 디바이스 (14) 는 비디오 코딩 디바이스들 또는 비디오 코딩 장치들의 예들일 수도 있다.

[0028] 소스 디바이스 (12) 와 목적지 디바이스 (14) 는 데스크톱 컴퓨터들, 모바일 컴퓨팅 디바이스들, 노트북 (예컨대, 랩톱) 컴퓨터들, 태블릿 컴퓨터들, 셋톱 박스들, 이른바 "스마트" 폰들과 같은 전화기 핸드셋들, 텔레비전들, 카메라들, 디스플레이 디바이스들, 디지털 미디어 플레이어들, 비디오 게이밍 콘솔들, 차량내 컴퓨터들 등을 포함한 다양한 범위의 디바이스들을 포함할 수도 있다.

[0029] 목적지 디바이스 (14) 는 소스 디바이스 (12) 로부터의 인코딩된 비디오 데이터를 채널 (16) 을 통해 수신할 수도 있다. 채널 (16) 은 소스 디바이스 (12) 로부터 목적지 디바이스 (14) 로 인코딩된 비디오 데이터를 이동시킬 수 있는 하나 이상의 매체들 또는 디바이스들을 포함할 수도 있다. 하나의 예에서, 채널 (16) 은 소스 디바이스 (12) 가 인코딩된 비디오 데이터를 목적지 디바이스 (14) 로 직접 실시간으로 송신하는 것을 가능하게 하는 하나 이상의 통신 매체들을 포함할 수도 있다. 이 예에서, 소스 디바이스 (12) 는 인코딩된 비디오 데이터를 통신 표준, 이를테면 무선 통신 프로토콜에 따라 변조할 수도 있고, 변조된 비디오 데이터를 목적지 디바이스 (14) 로 송신할 수도 있다. 하나 이상의 통신 매체들은 무선 및/또는 유선 통신 매체들, 이를테면 무선 주파수 (RF) 스펙트럼 또는 하나 이상의 물리적 송신 라인들을 포함할 수도 있다. 하나 이상의 통신 매체들은 패킷 기반 네트워크, 이를테면 로컬 영역 네트워크, 광역 네트워크, 또는 글로벌 네트워크 (예컨대, 인터넷) 의 일부를 형성할 수도 있다. 하나 이상의 통신 매체들은 라우터들, 스위치들, 기지국들, 또는 소스 디바이스 (12) 로부터 목적지 디바이스 (14) 로의 통신을 용이하게 하는 다른 장비를 포함할 수도 있다.

[0030] 다른 예에서, 채널 (16) 은 소스 디바이스 (12) 에 의해 생성된 인코딩된 비디오 데이터를 저장하는 저장 매체를 포함할 수도 있다. 이 예에서, 목적지 디바이스 (14) 는 예컨대, 디스크 액세스 또는 카드 액세스를 통해 저장 매체에 액세스할 수도 있다. 저장 매체는 블루 레이 디스크들, DVD들, CD-ROM들, 플래시 메모리, 또는 인코딩된 비디오 데이터를 저장하기 위한 다른 적합한 디지털 저장 매체들과 같은 다양한 국소적으로 액세스되는 데이터 저장 매체들을 포함할 수도 있다.

[0031] 추가의 예에서, 채널 (16) 은 소스 디바이스 (12) 에 의해 생성된 인코딩된 비디오 데이터를 저장하는 파일 서버 또는 다른 중간 저장 디바이스를 포함할 수도 있다. 이 예에서, 목적지 디바이스 (14) 는 파일 서버 또는 다른 중간 저장 디바이스에 저장된 인코딩된 비디오 데이터를 스트리밍 또는 다운로드를 통해 액세스할 수도 있다. 파일 서버는 인코딩된 비디오 데이터를 저장하고 그 인코딩된 비디오 데이터를 목적지 디바이스 (14) 로 송신할 수 있는 유형의 서버일 수도 있다. 예의 파일 서버들은 웹 서버들 (예컨대, 웹사이트용), 파일 전송 프로토콜 (FTP) 서버들, 네트워크 부속 스토리지 (network attached storage; NAS) 디바이스들, 및 로컬 디스크 드라이브들을 포함한다.

[0032] 목적지 디바이스 (14) 는 표준 데이터 접속, 이를테면 인터넷 접속을 통해, 인코딩된 비디오 데이터에 액세스할 수도 있다. 예의 유형들의 데이터 접속들은 파일 서버 상에 저장된 인코딩된 비디오 데이터에 액세스하기에 적합한 무선 채널들 (예컨대, Wi-Fi 접속들), 유선 접속들 (예컨대, 디지털 가입자 회선 (digital subscriber line; DSL), 케이블 모뎀 등), 또는 양쪽 모두의 조합들을 포함할 수도 있다. 파일 서버로부터의 인코딩된

비디오 데이터의 송신은 스트리밍 송신, 다운로드 송신, 또는 양쪽 모두의 조합일 수도 있다.

[0033] 본 개시물의 기법들은 무선 애플리케이션들 또는 설정 (setting) 들로 제한되지 않는다. 그 기법들은, 다양한 멀티미디어 애플리케이션들, 이를테면, 공중경유 (over-the-air) 텔레비전 브로드캐스트들, 케이블 텔레비전 송신들, 위성 텔레비전 송신들, 예컨대, 인터넷을 통한 스트리밍 비디오 송신들, 데이터 저장 매체 상의 저장을 위한 비디오 데이터의 인코딩, 데이터 저장 매체 상에 저장된 비디오 데이터의 디코딩, 또는 다른 애플리케이션들의 지원 하에 비디오 코딩에 적용될 수도 있다. 일부 예들에서, 비디오 코딩 시스템 (10) 은 비디오 스트리밍, 비디오 플레이백, 비디오 브로드캐스팅, 및/또는 화상 통화와 같은 애플리케이션들을 지원하기 위해 단방향 또는 양방향 비디오 송신을 지원하도록 구성될 수도 있다.

[0034] 도 1은 단지 일 예이고 본 개시물의 기법들은 인코딩 및 디코딩 디바이스들 간의 임의의 데이터 통신을 반드시 포함하지는 않는 비디오 코딩 설정들 (예컨대, 비디오 인코딩 또는 비디오 디코딩) 에 적용될 수도 있다. 다른 예들에서, 데이터 (예컨대, 비디오 데이터) 는 로컬 메모리로부터 취출되며, 네트워크를 통해 스트리밍되는 등등이 된다. 비디오 인코딩 디바이스가 데이터 (예컨대, 비디오 데이터) 를 인코딩하고 메모리에 저장할 수도 있으며, 및/또는 비디오 디코딩 디바이스가 메모리로부터 데이터 (예컨대, 비디오 데이터) 를 취출하고 디코딩할 수도 있다. 많은 예들에서, 인코딩과 디코딩은, 서로 통신하지 않지만 단순히 메모리로의 데이터 (예컨대, 비디오 데이터) 를 인코딩하고 및/또는 메모리로부터 데이터 (예컨대, 비디오 데이터) 를 취출하고 디코딩하는 디바이스들에 의해 수행된다.

[0035] 도 1의 예에서, 소스 디바이스 (12) 는 비디오 소스 (18), 비디오 인코더 (20), 및 출력 인터페이스 (22) 를 구비한다. 일부 예들에서, 출력 인터페이스 (22) 는 변조기/복조기 (모뎀) 및/또는 송신기를 구비할 수도 있다. 비디오 소스 (18) 는 비디오 캡처 디바이스, 예컨대, 비디오 카메라, 이전에 캡처된 비디오 데이터를 포함한 비디오 아카이브, 비디오 콘텐츠 제공자로부터 비디오 데이터를 수신하는 비디오 피드 인터페이스, 및/또는 비디오 데이터를 생성하는 컴퓨터 그래픽 시스템, 또는 비디오 데이터의 이러한 소스들의 조합을 포함할 수도 있다.

[0036] 비디오 인코더 (20) 는 비디오 소스 (18) 로부터의 비디오 데이터를 인코딩할 수도 있다. 일부 예들에서, 소스 디바이스 (12) 는 출력 인터페이스 (22) 를 통해 목적지 디바이스 (14) 로 인코딩된 비디오 데이터를 직접 송신한다. 다른 예들에서, 인코딩된 비디오 데이터는 디코딩 및/또는 플레이백을 위한 목적지 디바이스 (14) 에 의한 나중의 액세스를 위해 저장 매체 또는 파일 서버 상에 또한 저장될 수도 있다.

[0037] 도 1의 예에서, 목적지 디바이스 (14) 는 입력 인터페이스 (28), 비디오 디코더 (30), 및 디스플레이 디바이스 (32) 를 구비한다. 일부 예들에서, 입력 인터페이스 (28) 는 수신기 및/또는 모뎀을 구비한다. 입력 인터페이스 (28) 는 채널 (16) 을 통해 인코딩된 비디오 데이터를 수신할 수도 있다. 비디오 디코더 (30) 는 인코딩된 비디오 데이터를 디코딩할 수도 있다. 디스플레이 디바이스 (32) 는 디코딩된 비디오 데이터를 디스플레이할 수도 있다. 디스플레이 디바이스 (32) 는 목적지 디바이스 (14) 와 통합될 수도 있고 또는 그것 외부에 있을 수도 있다. 디스플레이 디바이스 (32) 는 액정 디스플레이 (LCD), 플라즈마 디스플레이, 유기 발광 다이오드 (OLED) 디스플레이, 또는 다른 유형의 디스플레이 디바이스와 같은 다양한 디스플레이 디바이스들을 포함할 수도 있다.

[0038] 비디오 인코더 (20) 와 비디오 디코더 (30) 각각은 본원에서 설명된 기능들을 수행하기 위해 다양한 적합한 회로, 이를테면 하나 이상의 마이크로프로세서들, 범용 프로세서들, 디지털 신호 프로세서들 (DSP들), 주문형 집적회로들 (ASIC들), 필드 프로그램가능 게이트 어레이들 (FPGA들), 개별 로직, 하드웨어, 또는 그것들의 임의의 조합들 중 임의의 것으로서 구현될 수도 있다. 범용 프로세서가 마이크로프로세서일 수도 있지만, 대체예에서, 그 프로세서는 임의의 기존의 프로세서, 제어기, 마이크로제어기, 또는 상태 머신 (state machine) 일 수도 있다. 프로세서가 또한, 컴퓨팅 디바이스들의 조합, 예컨대, DSP 및 마이크로프로세서의 조합, 복수의 마이크로프로세서들, DSP 코어와 협력하는 하나 이상의 마이크로프로세서들, 또는 임의의 다른 이러한 구성으로서 구현될 수도 있다. 그 기법들이 부분적으로 소프트웨어에서 구현되면, 디바이스가 적합한 비일시적 컴퓨터 판독가능 저장 매체 내에 소프트웨어에 대한 명령을 저장할 수도 있고 하나 이상의 프로세서들을 사용하여 하드웨어에서 그 명령들을 실행하여 본 개시물의 기법들을 수행할 수도 있다. 전술한 바 (하드웨어, 소프트웨어, 하드웨어 및 소프트웨어의 조합 등을 포함) 중 임의의 것은 하나 이상의 프로세서들이라고 간주될 수도 있다. 비디오 인코더 (20) 및 비디오 디코더 (30) 의 각각은 하나 이상의 인코더들 또는 디코더들 내에 구비될 수도 있고, 그것들 중 어느 하나는 결합형 인코더/디코더 (CODEC) 의 일부로서 개별 디바이스 내에 통합될 수도 있다.

[0039] 본 개시물은 비디오 인코더 (20) 가 다른 디바이스, 이를테면 비디오 디코더 (30) 에 또는 비트스트림에서 특정 정보를 "시그널링하는 것" 을 일반적으로 참조할 수도 있다. 용어 "시그널링"은 일반적으로는 압축된 비디오 데이터를 디코딩하는데 사용되는 신택스 엘리먼트들 및/또는 다른 데이터의 통신을 말할 수도 있다. 이러한 통신은 실시간 또는 거의 실시간으로 일어날 수도 있다. 대안으로, 이러한 통신은, 인코딩 시에 신택스 엘리먼트들을 인코딩된 비트스트림에서 컴퓨터 판독가능 저장 매체에 저장하고 그 신택스 엘리먼트들이 이 매체에 저장된 후의 임의의 시간에 디코딩 디바이스에 의해 취출될 수도 있는 경우에 일어날 바와 같이 어떤 기간 (span of time) 에 걸쳐 일어날 수도 있다.

[0040] 일부 예들에서, 비디오 인코더 (20) 와 비디오 디코더 (30) 는, ISO/IEC MPEG-4 비주얼 그리고 SVC (Scalable Video Coding) 확장본, MVC (Multiview Video Coding) 확장본, MVC 기반 3DV 확장본을 포함한 ITU-T H.264 (또한 ISO/IEC MPEG-4 AVC로 알려짐) 와 같은 비디오 압축 표준에 따라 동작한다. 더욱이, H.264/AVC에 대한 3차원 비디오 (3DV) 코딩 확장본, 즉 AVC 기반 3DV를 생성하려는 지속적인 노력이 있다. H.264의 MVC 확장본의 공동 초안이 『"Advanced video coding for generic audiovisual services", ITU-T Recommendation H.264, Mar 2010』에 기재되어 있다. 다른 예들에서, 비디오 인코더 (20) 와 비디오 디코더 (30) 는 ITU-T H.261, ISO/IEC MPEG-1 비주얼, ITU-T H.262 또는 ISO/IEC MPEG-2 비주얼, 및 ITU-T H.263, ISO/IEC-4 비주얼에 따라 동작할 수도 있다.

[0041] 다른 예들에서, 비디오 인코더 (20) 와 비디오 인코더 (30) 는, ITU-T 비디오 코딩 전문가 그룹 (VCEG) 및 ISO/IEC 동 화상 전문가 그룹 (MPEG) 의 JCT-VC (Joint Collaboration Team on Video Coding) 에 의해 개발된 고 효율 비디오 코딩 (HEVC) 표준에 따라 동작할 수도 있다. "HEVC 규격 초안 8" 또는 "HEVC 기초 규격"이라고 지칭되는 HEVC 표준의 초안이, 『Bross et al., "High Efficiency Video Coding (HEVC) text specification draft 8", Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, 10th Meeting, Stockholm, Sweden, July 2012』에 기재되어 있다. 2014년 1월 9일 현재, HEVC 규격 초안 8은 http://phenix.int-evry.fr/jct/doc_end_user/documents/10_Stockholm/wg11/JCTVC-J1003-v8.zip으로부터 다운로드할 수 있다. SHVC라고 지칭되는 HEVC의 스케일러블 비디오 코딩 확장본이 개발되고 있다.

[0042] 더욱이, HEVC에 대한 멀티-뷰 코딩 및 3DV 확장본들을 생성하려는 노력이 지속되고 있다. 다르게 말하면, VCEG 및 MPEG의 JCT-3V (Joint Collaboration Team on 3D video Coding) 가 HEVC에 기초하여 3DV 표준을 개발하고 있는데, 표준화 노력의 부분이 HEVC에 기초한 멀티-뷰 비디오 코덱의 표준화 (MV-HEVC) 를 포함하고 다른 부분이 HEVC에 기초한 3D 비디오 코딩 (3D-HEVC) 을 포함한다. 비디오 인코더 (20) 와 비디오 디코더 (30) 는 HEVC 표준에 대한 이러한 확장본들에 따라 동작할 수도 있다. HEVC의 멀티-뷰 코딩 확장본은 MV-HEVC라고 지칭될 수도 있다. 『Gerhard Tech et al., "MV-HEVC Working Draft 1", JCT3V-A1004, Joint Collaborative Team on 3D Video Coding Extension Development of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11, 1st Meeting: Stockholm, SE, 16-20 July 2012』 (이후로는, "JCT3V-A1004" 또는 "MV-HEVC 규격 초안 1") 가, MV-HEVC에 대한 규격 초안을 제공한다. 『Gerhard Tech et al., "MV-HEVC Working Draft 2", JCT3V-B1004, Joint Collaborative Team on 3D Video Coding Extension Development of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11, 2nd Meeting: Shanghai, CN, 13-19 October 2012』 (이후로는, "MV-HEVC 규격 초안 2") 가, MV-HEVC에 대한 다른 규격 초안을 제공한다.

[0043] HEVC의 3DV 확장본은 3DV-HEVC라고 지칭될 수도 있다. 『Tech et al., "Draft of 3D-HEVC Test Model Description Draft", JCT3V-B1005, Joint Collaborative Team on 3D Video Coding Extension Development of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11, 1st Meeting: Stockholm, SE, 16-20 July 2012』 (이후로는, "3D-HEVC 테스트 모델 1") 가 3D-HEVC의 규격 초안뿐 아니라 참조 소프트웨어를 설명한다. 덧붙여서, 『Tech et al., "3D-HEVC Test Model Description Draft 2", JCT3V-B1005, Joint Collaborative Team on 3D Video Coding Extension Development of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11, 1st Meeting: Stockholm, SE, 16-20 July 2012』 (이후로는, "3D-HEVC 테스트 모델 설명 초안 2") 가 3D-HEVC의 규격 초안뿐 아니라 참조 소프트웨어를 설명한다. 2014년 1월 9일 현재, 3D-HEVC에 대한 참조 소프트웨어, 즉 3DV-HTM은 https://hevc.hhi.fraunhofer.de/svn/svn_3DVCSoftware/trunk로부터 입수가능하다.

[0044] HEVC는 모션 보상 루프를 제공한다. 대체로, HEVC의 모션 보상 루프는 H.264/AVC의 그것과 동일하다. 예를 들어, 현재 프레임 ($\hat{I}$) 의 복원은 역양자화된 계수들 (r) 더하기 시간적 예측 (P) 과 동일할 수도 있다:

[0045] $\hat{I} = r + P$.

[0046] 위의 수식에서, P는 P 프레임들에 대한 단방향 예측 또는 B 프레임들에 대한 양방향 예측을 나타낸다.

[0047] 그러나, HEVC에서의 모션 보상의 유닛은 이전의 비디오 코딩 표준들에서의 유닛과는 상이하다. 예를 들어, 이전의 비디오 코딩 표준들에서의 매크로블록의 개념은 HEVC에서 존재하지 않는다. 오히려, 매크로블록들은 일반 쿼드트리 체계에 기초하여 고도로 유연한 계층 구조에 의해 대체된다. 이 체계 내에서, 3 가지 유형들의 블록들, 즉, 코딩 유닛 (coding unit; CU) 들, 예측 유닛 (prediction unit; PU) 들, 및 변환 유닛 (transform unit; TU) 들이 정의된다. CU가 영역 분할의 기본 유닛이다. CU의 개념은 매크로블록의 개념과 유사하지만, CU가 최대 사이즈로 제한되지 않고 콘텐츠 적응성을 개선하기 위해 CU가 동일한 사이즈로 된 4 개의 CU들로의 재귀적 분할을 허용한다. PU가 인터/인트라 예측의 기본 유닛이고 PU가 불규칙한 이미지 패턴들을 효과적으로 코딩하기 위해 단일 PU 내에 다수의 임의적으로 형상된 구획들을 포함할 수도 있다. TU가 변환의 기본 유닛이다. CU의 TU들은 CU의 PU들로부터 독립적으로 정의될 수 있다. 그러나, TU의 사이즈가 TU가 속하는 CU로 제한된다. 3 개의 상이한 개념들로의 블록 구조의 이러한 분리는 각각이 그것의 역할에 따라 최적화되는 것을 허용할 수도 있는데, 이는 개선된 코딩 효율을 초래할 수도 있다.

[0048] HEVC 및 다른 비디오 코딩 사양들에서, 비디오 시퀀스가 일련의 화상들을 통상 포함한다. 화상들은 "프레임들"이라고 또한 지칭될 수도 있다. 화상이, S_L, S_{Cb} 및 S_{Cr}로 표시되는 3 개의 샘플 어레이들을 포함할 수도 있다. S_L은 루마 샘플들의 2차원 어레이 (즉, 블록) 이다. S_{Cb}는 Cb 크로미넌스 샘플들의 2차원 어레이이다. S_{Cr}은 Cr 크로미넌스 샘플들의 2차원 어레이이다. 크로미넌스 샘플들은 본원에서 "크로마" 샘플들이라고 또한 지칭될 수도 있다. 다른 경우들에서, 화상이 모노크롬일 수도 있고 루마 샘플들의 어레이만을 포함할 수도 있다.

[0049] 화상의 인코딩된 표현을 생성하기 위해, 비디오 인코더 (20) 는 코딩 트리 유닛들 (CTU들) 의 세트를 생성할 수도 있다. CTU들의 각각은, 루마 샘플들의 코딩 트리 블록, 크로마 샘플들의 2 개의 대응 코딩 트리 블록들, 및 코딩 트리 블록들의 샘플들을 코딩하는데 사용된 신택스 구조들을 포함할 수도 있다. 모노크롬 화상들 또는 3 개의 별개의 컬러 평면들을 갖는 화상들에서, CTU가 단일 코딩 트리 블록과 그 코딩 트리 블록의 샘플들을 코딩하는데 사용된 신택스 구조들을 포함할 수도 있다. 코딩 트리 블록이 샘플들의 NxN 블록일 수도 있다. CTU가 "트리 블록" 또는 "최대 코딩 유닛 (largest coding unit; LCU)"이라고 또한 지칭될 수도 있다. HEVC의 CTU들은 다른 표준들, 이를테면 H.264/AVC의 매크로블록들과 대체로 유사할 수도 있다. 그러나, CTU가 특정 사이즈로 반드시 제한되는 것은 아니고 하나 이상의 CU들을 포함할 수도 있다. 슬라이스가 래스터 스캔 순서로 연속하여 순서화된 정수 수의 CTU들을 포함할 수도 있다.

[0050] 코딩된 슬라이스가 슬라이스 헤더와 슬라이스 데이터를 포함할 수도 있다. 슬라이스의 슬라이스 헤더는 슬라이스에 관한 정보를 제공하는 신택스 엘리먼트들을 포함하는 신택스 구조일 수도 있다. 슬라이스 데이터는 슬라이스의 코딩된 CTU들을 포함할 수도 있다.

[0051] 본 개시물은 하나 이상의 샘플 블록들 및 그 하나 이상의 샘플 블록들을 코딩하는데 사용된 신택스 구조들을 지칭하기 위해 "비디오 유닛" 또는 "비디오 블록" 또는 "블록"이란 용어를 사용할 수도 있다. 비디오 유닛들의 예의 유형들은 CTU들, CU들, PU들, 변환 유닛들 (TU들), 매크로블록들, 매크로블록 구획들 등을 포함할 수도 있다. 일부 문맥들에서, PU들 또는 CU들의 논의는 매크로블록들 또는 매크로블록 구획들의 논의와 교환될 수도 있다.

[0052] 코딩된 CTU를 생성하기 위해, 비디오 인코더 (20) 는 CTU의 코딩 트리 블록들에 대해 쿼드트리 구획화를 재귀적으로 수행하여 코딩 트리 블록들을 코딩 블록들로 나눌 수도 있으며, 따라서 그 이름이 "코딩 트리 유닛들"이다. 코딩 블록이 샘플들의 NxN 블록이다. CU가, 루마 샘플 어레이, Cb 샘플 어레이 및 Cr 샘플 어레이를 갖는 화상의 루마 샘플들의 코딩 블록 및 크로마 샘플들의 2 개의 대응 코딩 블록들과, 그리고 그 코딩 블록들의 샘플들을 코딩하는데 사용된 신택스 구조들을 포함할 수도 있다. 모노크롬 화상들 또는 3 개의 별개의 컬러 평면들을 갖는 화상들에서, CU가 단일 코딩 블록과 그 코딩 블록의 샘플들을 코딩하는데 사용된 신택스 구조들을 포함할 수도 있다.

[0053] 비디오 인코더 (20) 는 CU의 코딩 블록을 하나 이상의 예측 블록들로 구획화할 수도 있다. 예측 블록이 동일한 예측이 적용되는 샘플들의 직사각형 (즉, 정사각형이거나 또는 정사각형이 아닌) 블록이다. CU의 PU가 루마 샘플들의 예측 블록, 크로마 샘플들의 2 개의 대응하는 예측 블록들, 및 그 예측 블록들을 예측하는데 사

용된 신택스 구조들을 포함할 수도 있다. 모노크롬 화상들 또는 3 개의 별개의 컬러 평면들을 갖는 화상들에서, PU가 단일 예측 블록과 그 예측 블록을 예측하는데 사용된 신택스 구조들을 포함할 수도 있다. 비디오 인코더 (20) 는 CU의 각각의 PU의 루마, Cb 및 Cr 예측 블록들에 대한 예측 루마, Cb 및 Cr 블록들을 생성할 수도 있다. 따라서, 본 개시물에서는, CU가 하나 이상의 PU들로 구획화될 것으로 말해질 수도 있다. 설명의 편의를 위해, 본 개시물은 간단히 PU의 사이즈로서 PU의 예측 블록의 사이즈를 지칭할 수도 있다.

[0054] 비디오 인코더 (20) 는 PU에 대한 예측 블록들을 생성하기 위해 인트라 예측 또는 인터 예측을 사용할 수도 있다. 비디오 인코더 (20) 가 PU의 예측 블록들을 생성하기 위해 인트라 예측을 사용하면, 비디오 인코더 (20) 는 그 PU와 연관된 화상의 샘플들에 기초하여 그 PU의 예측 블록들을 생성할 수도 있다. 본 개시물에서, "에 기초하여"라는 어구는 "에 적어도 부분적으로 기초하여"를 나타낼 수도 있다.

[0055] 비디오 인코더 (20) 가 PU의 예측 블록들을 생성하기 위해 인터 예측을 사용하면, 비디오 인코더 (20) 는 그 PU와 연관된 화상 이외의 하나 이상의 화상들의 디코딩된 샘플들에 기초하여 그 PU의 예측 블록들을 생성할 수도 있다. 인터 예측이 블록의 예측 블록들 (예컨대, PU) 을 생성하는데 사용되는 경우, 본 개시물은 그 블록을 "인터 코딩된" 또는 "인터 예측된" 것이라고 지칭할 수도 있다. 인터 예측은 단방향 (즉, 단예측 (uni-prediction)) 또는 양방향 (즉, 양예측 (bi-prediction)) 일 수도 있다. 단예측 또는 양예측을 수행하기 위해, 비디오 인코더 (20) 는 현재 화상에 대한 제 1 참조 화상 리스트 (RefPicList0) 및 제 2 참조 화상 리스트 (RefPicList1) 를 생성할 수도 있다. 참조 화상 리스트들의 각각은 하나 이상의 참조 화상들을 포함할 수도 있다. 참조 화상 리스트 (즉, 이용가능하면, RefPicList0 및 RefPicList1) 가 구축된 후, 참조 화상 리스트에 대한 참조 인덱스가 참조 화상 리스트에 포함된 임의의 참조 화상을 식별하는데 사용될 수 있다.

[0056] 단예측을 사용하는 경우, 비디오 인코더 (20) 는 참조 화상 내의 참조 로케이션을 결정하기 위해 RefPicList0 및 RefPicList1 중 어느 하나 또는 양쪽 모두에서 참조 화상들을 검색할 수도 있다. 더욱이, 단예측을 사용하는 경우, 비디오 인코더 (20) 는, 참조 로케이션에 대응하는 샘플들에 적어도 부분적으로 기초하여, PU에 대한 예측 블록들을 생성할 수도 있다. PU에 대한 예측 블록에서의 각각의 샘플은 참조 로케이션과 연관될 수도 있다. 일부 예들에서, PU에 대한 예측 블록에서의 샘플은, 그 샘플이 PU와 동일한 사이즈를 갖고 좌측상단 코너가 참조 로케이션인 샘플들의 블록 내에 있는 경우, 참조 로케이션과 연관될 수도 있다. 예측 블록에서의 각각의 샘플은 참조 화상의 실제 또는 보간된 샘플일 수도 있다. 더구나, 단예측을 사용하는 경우, 비디오 인코더 (20) 는 PU의 예측 블록 및 참조 로케이션 사이의 공간적 변위를 나타내는 단일 모션 벡터를 생성할 수도 있다. 그 모션 벡터는 PU의 예측 블록 및 참조 로케이션 사이의 수평 변위를 특정하는 수평 성분을 포함할 수도 있고, PU의 예측 블록 및 참조 로케이션 사이의 수직 변위를 특정하는 수직 성분을 포함할 수도 있다.

[0057] 양예측을 사용하여 PU를 인코딩하는 경우, 비디오 인코더 (20) 는 RefPicList0에서의 참조 화상의 제 1 참조 로케이션 및 RefPicList1에서의 참조 화상의 제 2 참조 로케이션을 결정할 수도 있다. 비디오 인코더 (20) 는 제 1 및 제 2 참조 로케이션들에 대응하는 샘플들에 적어도 부분적으로 기초하여, PU에 대한 예측 블록들을 생성할 수도 있다. 예측 블록에서의 각각의 샘플은 참조 블록들에서의 대응 샘플들의 가중된 평균일 수도 있다. 샘플들의 가중은 PU를 포함하는 화상으로부터의 참조 화상들의 시간적 거리들에 기초할 수도 있다. 더구나, 양예측을 사용하여 PU를 인코딩하는 경우, 비디오 인코더 (20) 는, PU의 예측 블록 및 제 1 참조 로케이션 사이의 공간적 변위를 나타내는 제 1 모션 벡터와 PU의 예측 블록 및 제 2 참조 로케이션 사이의 공간적 변위를 나타내는 제 2 모션 벡터를 생성할 수도 있다. 따라서, 비디오 인코더 (20) 가 PU에 대해 양 예측을 수행하는 경우, PU는 2 개의 모션 벡터들을 갖는다.

[0058] 비디오 인코더 (20) 가 PU의 예측 블록들을 생성하기 위해 인터 예측을 사용하면, 비디오 인코더 (20) 는 그 PU와 연관된 화상 이외의 하나 이상의 화상들의 샘플들에 기초하여 그 PU의 예측 블록들을 생성할 수도 있다. 예를 들면, 비디오 인코더 (20) 는 PU에 대해 단방향 인터 예측 (즉, 단예측) 또는 양방향 인터 예측 (즉, 양예측) 을 수행할 수도 있다.

[0059] 비디오 인코더 (20) 는 다양한 구획화 모드들에 따라 CU를 하나 이상의 PU들로 구획화할 수도 있다. 예를 들면, 인트라 예측이 CU의 PU들에 대해 예측 블록들을 생성하는데 사용되면, CU는 PART_2Nx2N 모드 또는 PART_NxN 모드에 따라 구획화될 수도 있다. PART_2Nx2N 모드에서, CU는 하나의 PU만을 갖는다. PART_NxN 모드에서, CU는 직사각형 예측 블록들을 갖는 4 개의 동일 사이즈로 된 PU들을 갖는다. 인터 예측이 CU의 PU들에 대한 예측 블록들을 생성하는데 사용되면, CU는 PART_2Nx2N 모드, PART_NxN 모드, PART_2NxN 모드, PART_Nx2N 모드, PART_2NxnU 모드, PART_2NxuD 모드, PART_nLx2N 모드, 또는 PART_nRx2N 모드에 따라 구

획화될 수도 있다. PART_2NxN 모드 및 PART_Nx2N 모드에서, CU는 직사각형 예측 블록들을 갖는 2 개의 동일 사이즈로 된 PU들로 구획화된다. PART_2NxnU 모드, PART_2NxuD 모드, PART_nLx2N 모드, 및 PART_nRx2N 모드의 각각에서, CU는 직사각형 예측 블록들을 갖는 2 개의 동일하지 않은 사이즈로 된 PU들로 구획화된다. 비대칭 구획화에서, CU의 하나의 방향은 구획화되지 않는 반면, 다른 방향은 25% 및 75%로 구획화된다. 25% 구획에 대응하는 CU의 부분은 "n" 다음에 "Up", "Down", "Left", 또는 "Right"의 표시가 뒤따르는 것에 의해 나타내어진다. 따라서, 예를 들어, "2NxnU"는 상단의 2Nx0.5N PU 및 하단의 2Nx1.5N PU로 수평으로 구획화되는 2Nx2N CU를 지칭한다.

[0060] 비디오 인코더 (20) 가 CU의 하나 이상의 PU들에 대해 하나 이상의 예측 블록들 (예컨대, 루마, Cb 및 Cr 예측 블록들) 을 생성한 후, 비디오 인코더 (20) 는 그 CU에 대해 하나 이상의 잔차 블록을 생성할 수도 있다. 예를 들면, 비디오 인코더 (20) 는 CU에 대한 루마 잔차 블록을 생성할 수도 있다. CU의 루마 잔차 블록에서의 각각의 샘플은 CU의 예측 루마 블록들 중 하나의 예측 루마 블록에서의 루마 샘플과 CU의 원래의 루마 코딩 블록에서의 대응하는 샘플 사이의 차이를 나타낸다. 덧붙여서, 비디오 인코더 (20) 는 CU에 대한 Cb 잔차 블록을 생성할 수도 있다. CU의 Cb 잔차 블록에서의 각각의 샘플은 CU의 예측 Cb 블록들 중 하나의 예측 Cb 블록에서의 Cb 샘플과 CU의 원래의 Cb 코딩 블록에서의 대응하는 샘플 사이의 차이를 나타낼 수도 있다. 비디오 인코더 (20) 는 CU에 대한 Cr 잔차 블록을 또한 생성할 수도 있다. CU의 Cr 잔차 블록에서의 각각의 샘플은 CU의 예측 Cr 블록들 중 하나의 예측 Cr 블록에서의 Cr 샘플과 CU의 원래의 Cr 코딩 블록에서의 대응하는 샘플 사이의 차이를 나타낼 수도 있다.

[0061] 더욱이, 비디오 인코더 (20) 는 쿼드트리 구획화를 사용하여 CU의 하나 이상의 잔차 블록 (예컨대, CU의 루마, Cb 및 Cr 잔차 블록들) 을 하나 이상의 변환 블록들 (예컨대, 루마, Cb 및 Cr 변환 블록들) 로 분해할 수도 있다. 동일한 변환이 적용되는 샘플들의 직사각형 (예컨대, 정사각형이거나 또는 정사각형이 아닌) 블록이 변환 블록이다. CU의 TU가 루마 샘플들의 변환 블록, 크로마 샘플들의 2 개의 대응하는 변환 블록들, 및 그 변환 블록 샘플들을 변환하는데 사용된 신택스 구조들을 포함할 수도 있다. 따라서, CU의 각각의 TU는 루마 변환 블록, Cb 변환 블록, 및 Cr 변환 블록과 연관될 수도 있다. TU와 연관된 루마 변환 블록은 CU의 루마 잔차 블록의 서브-블록일 수도 있다. Cb 변환 블록은 CU의 Cb 잔차 블록의 서브-블록일 수도 있다. Cr 변환 블록은 CU의 Cr 잔차 블록의 서브-블록일 수도 있다. 모노크롬 화상들 또는 3 개의 별개의 컬러 평면들을 갖는 화상들에서, TU가 단일 변환 블록과 그 변환 블록의 샘플들을 변환하는데 사용된 신택스 구조들을 포함할 수도 있다.

[0062] 비디오 인코더 (20) 는 하나 이상의 변환들을 TU의 변환 블록에 적용하여 TU에 대한 계수 블록을 생성할 수도 있다. 계수 블록이 변환 계수들의 2차원 어레이일 수도 있다. 예를 들어, 비디오 인코더 (20) 는 하나 이상의 변환들을 TU의 루마 변환 블록에 적용하여 그 TU에 대한 루마 계수 블록을 생성할 수도 있다. 변환 계수가 스칼라 양일 수도 있다. 비디오 인코더 (20) 는 하나 이상의 변환들을 TU의 Cb 변환 블록에 적용하여 TU에 대한 Cb 계수 블록을 생성할 수도 있다. 비디오 인코더 (20) 는 하나 이상의 변환들을 TU의 Cr 변환 블록에 적용하여 TU에 대한 Cr 계수 블록을 생성할 수도 있다.

[0063] 계수 블록 (예컨대, 루마 계수 블록, Cb 계수 블록 또는 Cr 계수 블록) 을 생성한 후, 비디오 인코더 (20) 는 그 계수 블록을 양자화할 수도 있다. 양자화는 변환 계수들이 그 변환 계수들을 표현하는데 사용된 데이터의 양을 가능한 한 줄이도록 양자화되어서, 추가의 압축을 제공하는 프로세스를 일반적으로 지칭한다. 비디오 인코더 (20) 가 계수 블록을 양자화한 후, 비디오 인코더 (20) 는 양자화된 변환 계수들을 나타내는 신택스 엘리먼트들을 엔트로피 인코딩할 수도 있다. 예를 들어, 비디오 인코더 (20) 는 양자화된 변환 계수들을 나타내는 신택스 엘리먼트들에 대해 콘텍스트 적응 이진 산술 코딩 (Context-Adaptive Binary Arithmetic Coding; CABAC) 을 수행할 수도 있다.

[0064] 비디오 인코더 (20) 는 코딩된 화상들의 표현 및 연관된 데이터를 형성하는 비트들의 시퀀스를 포함하는 비트스트림을 출력할 수도 있다. 다르게 말하면, 비디오 인코더 (20) 는 비디오 데이터의 인코딩된 표현을 포함하는 비트스트림을 생성할 수도 있다. 그 비트스트림은 네트워크 추상화 계층 (network abstraction layer; NAL) 유닛들의 시퀀스를 포함할 수도 있다. NAL 유닛이, NAL 유닛에서의 데이터의 유형의 표시 (indication) 와 에뮬레이션 방지 바이트들이 필요한대로 점재된 (interspersed) RBSP (raw byte sequence payload) 형태로 당해 데이터를 포함한 바이트들을 포함하는 신택스 구조이다. NAL 유닛들의 각각은 NAL 유닛 헤더를 포함하고 RBSP를 캡슐화한다. NAL 유닛 헤더는 NAL 유닛 유형 코드를 나타내는 신택스 엘리먼트를 포함할 수도 있다. NAL 유닛의 NAL 유닛 헤더에 의해 특정된 NAL 유닛 유형 코드는 NAL 유닛의 유형을 나타낸다. RBSP가 NAL 유닛 내에 캡슐화되는 정수 수의 바이트들을 포함하는 신택스 구조일 수도 있다.

일부 경우들에서, RBSP가 0 비트들을 포함한다.

[0065] 상이한 유형들의 NAL 유닛들이 상이한 유형들의 RBSP들을 캡슐화할 수도 있다. 예를 들어, 상이한 유형들의 NAL 유닛은 비디오 파라미터 세트들 (VPS들), 시퀀스 파라미터 세트들 (SPS들), 화상 파라미터 세트들 (PPS들), 코딩된 슬라이스들, 추가 향상 정보 (SEI) 등에 대해 상이한 RBSP들을 캡슐화할 수도 있다. 비디오 코딩 데이터에 대한 RBSP들 (파라미터 세트들 및 SEI 메시지들에 대한 RBSP과는 대조적임) 을 캡슐화하는 NAL 유닛들은, 비디오 코딩 계층 (video coding layer; VCL) NAL 유닛들이라고 지칭될 수도 있다.

[0066] HEVC에서, SPS들은 코딩된 비디오 시퀀스 (coded video sequence; CVS) 의 모든 슬라이스들에 적용되는 정보를 포함할 수도 있다. CVS가 화상들의 시퀀스를 포함할 수도 있다. HEVC에서, CVS가, IDR (instantaneous decoding refresh) 화상, 또는 BLA (broken link access) 화상, 또는 IDR 또는 BLA 화상이 아닌 모든 후속 화상들을 포함하는, 비트스트림에서의 첫 번째 화상인 CRA (clean random access) 화상으로부터 시작할 수도 있다. 다시 말하면, HEVC에서, CVS가, 비트스트림에서의 첫 번째 액세스 유닛인 CRA 액세스 유닛, IDR 액세스 유닛 또는 BLA 액세스 유닛과, 임의의 후속 IDR 또는 BLA 액세스 유닛 전까지를 포함하는 모든 후속 액세스 유닛들을 포함한 뒤따르는 0 이상의 비-IDR 및 비-BLA 액세스 유닛들로 디코딩 순서로 구성될 수도 있는 액세스 유닛들의 시퀀스를 포함할 수도 있다. HEVC에서, 액세스 유닛이 디코딩 순서에서 연속적이고 정확히 하나의 코딩된 화상을 포함하는 NAL 유닛들의 세트일 수도 있다. 코딩된 화상의 코딩된 슬라이스 NAL 유닛들에 더하여, 액세스 유닛은 코딩된 화상의 슬라이스들을 포함하지 않는 다른 NAL 유닛들을 또한 포함할 수도 있다. 액세스 유닛의 디코딩은 디코딩된 화상이 항상 생겨나게 한다.

[0067] VPS가 0 이상의 (예컨대, 하나 이상의) 전체 CVS들에 적용하는 신택스 엘리먼트들을 포함한 신택스 구조이다. 하나 이상의 SPS들은 SPS들이 액티브인 경우 동일한 VPS가 액티브임을 식별하는 신택스 엘리먼트들을 포함할 수도 있다. 따라서, VPS의 신택스 엘리먼트들은 SPS의 신택스 엘리먼트들보다 더 일반적으로 적용가능할 수도 있다. 0 이상의 코딩된 화상들에 적용되는 신택스 엘리먼트들을 포함하는 신택스 구조가 PPS이다. PPS는 그 PPS가 액티브인 경우 액티브가 되는 SPS를 식별하는 신택스 엘리먼트를 포함할 수도 있다. 슬라이스의 슬라이스 헤더가 그 슬라이스가 코딩되고 있는 경우 액티브인 PPS를 나타내는 신택스 엘리먼트를 포함할 수도 있다.

[0068] 비디오 디코더 (30) 는 비디오 인코더 (20) 에 의해 생성된 비트스트림을 수신할 수도 있다. 덧붙여서, 비디오 디코더 (30) 는 비트스트림으로부터 신택스 엘리먼트들을 획득하기 위해 그 비트스트림을 파싱할 수도 있다. 비디오 디코더 (30) 는 비트스트림으로부터 획득된 신택스 엘리먼트들에 적어도 부분적으로 기초하여 비디오 데이터의 화상들을 복원할 수도 있다. 비디오 데이터를 복원하는 프로세스는 비디오 인코더 (20) 에 의해 수행된 프로세스에 일반적으로 역일 수도 있다. 예를 들면, 비디오 디코더 (30) 는 현재 CU의 PU들에 대한 예측 블록들을 결정하기 위해 그 PU들의 모션 벡터들을 사용할 수도 있다. 덧붙여서, 비디오 디코더 (30) 는 현재 CU의 TU들과 연관된 계수 블록들을 역 양자화할 수도 있다. 비디오 디코더 (30) 는 현재 CU의 TU들과 연관된 변환 블록들을 복원하기 위해 계수 블록들에 대해 역 변환들을 수행할 수도 있다. 비디오 디코더 (30) 는 현재 CU의 PU들에 대한 예측 블록들의 샘플들을 현재 CU의 TU들의 변환 블록들의 대응하는 샘플들에 가산함으로써 현재 CU의 코딩 블록들을 복원할 수도 있다. 화상의 각각의 CU에 대한 코딩 블록들을 복원함으로써, 비디오 디코더 (30) 는 그 화상을 복원할 수도 있다.

[0069] 일부 예들에서, 비디오 인코더 (20) 는 병합/스킵 모드 또는 고급 모션 벡터 예측 (advanced motion vector prediction; AMVP) 모드를 사용하여 PU의 모션 정보를 시그널링할 수도 있다. 다르게 말하면, HEVC에서는, 모션 파라미터들의 예측을 위해 하나는 병합/스킵 모드이고 다른 하나는 AMVP인 2 개의 모드들이 있다. 모션 예측은 하나 이상의 다른 블록들의 모션 정보에 기초한 블록 (예컨대, PU) 의 모션 정보의 결정을 포함할 수도 있다. PU의 모션 정보 (즉, 모션 파라미터들) 는 PU의 모션 벡터(들), PU의 참조 인덱스(들), 및 하나 이상의 예측 방향 표시자들을 포함할 수도 있다.

[0070] 비디오 인코더 (20) 가 병합 모드를 사용하여 현재 PU의 모션 정보를 시그널링하는 경우, 비디오 인코더 (20) 는 병합 후보 리스트를 생성한다. 다르게 말하면, 비디오 인코더 (20) 는 모션 벡터 예측자 리스트 구축 프로세스를 수행할 수도 있다. 병합 후보 리스트는 현재 PU에 공간적으로 또는 시간적으로 이웃하는 PU들의 모션 정보를 나타내는 병합 후보들의 세트를 포함한다. 다시 말하면, 병합 모드에서, 후보들이 공간적 및 시간적 이웃 블록들로부터 형성될 수 있는 모션 파라미터들 (예컨대, 참조 인덱스들, 모션 벡터들 등) 의 후보 리스트가 구축될 수도 있다.

[0071] 더욱이, 병합 모드에서, 비디오 인코더 (20) 는 병합 후보 리스트로부터 병합 후보를 선택할 수도 있고 선택된

병합 후보에 의해 나타내어진 모션 정보를 현재 PU의 모션 정보로서 사용할 수도 있다. 비디오 인코더 (20) 는 선택된 병합 후보의 병합 후보 리스트에서의 포지션을 시그널링할 수도 있다. 예를 들면, 비디오 인코더 (20) 는 선택된 병합 후보의 후보 리스트 내의 포지션을 나타내는 인덱스 (즉, 병합 후보 인덱스) 를 송신함으로써 선택된 모션 벡터 파라미터들을 시그널링할 수도 있다. 비디오 디코더 (30) 는, 비트스트림으로부터, 후보 리스트에 대한 인덱스 (즉, 병합 후보 인덱스) 를 획득할 수도 있다. 덧붙여서, 비디오 디코더 (30) 는 동일한 병합 후보 리스트를 생성할 수도 있고, 병합 후보 인덱스에 기초하여, 선택된 병합 후보를 결정할 수도 있다. 비디오 디코더 (30) 는 그 다음에 선택된 병합 후보의 모션 정보를 사용하여 현재 PU에 대한 예측 블록들을 생성할 수도 있다. 다시 말하면, 비디오 디코더 (30) 는, 후보 리스트 인덱스에 적어도 부분적으로 기초하여, 후보 리스트에서 선택된 후보를 결정할 수도 있는데, 선택된 후보는 현재 PU에 대한 모션 벡터를 특정한다. 이런 식으로, 디코더 측에서, 일단 인덱스가 디코딩되면, 인덱스가 가리키는 대응 블록의 모든 모션 파라미터들은 현재 PU에 의해 인계될 수도 있다.

[0072] 스킵 모드는 병합 모드와 유사하다. 스킵 모드에서, 비디오 인코더 (20) 와 비디오 디코더 (30) 는 비디오 인코더 (20) 와 비디오 디코더 (30) 가 병합 모드에서 병합 후보 리스트를 사용하는 것과 동일한 방식으로 병합 후보 리스트를 생성하고 사용한다. 그러나, 비디오 인코더 (20) 가 스킵 모드를 사용하여 현재 PU의 모션 정보를 시그널링하는 경우, 비디오 인코더 (20) 는 현재 PU에 대해 임의의 잔차 데이터를 시그널링하지 않는다. 이에 따라, 비디오 디코더 (30) 는, 잔차 데이터의 사용 없이, 병합 후보 리스트에서의 선택된 후보의 모션 정보에 의해 나타내어진 참조 블록에 기초하여 PU에 대한 예측 블록을 결정할 수도 있다.

[0073] AMVP 모드는 비디오 인코더 (20) 가 후보 리스트를 생성할 수도 있고 후보 리스트로부터 후보를 선택할 수도 있다는 점에서 병합 모드와 유사하다. 그러나, 비디오 인코더 (20) 가 AMVP 모드를 사용하여 현재 PU의 RefPicListX (X는 0 또는 1 임) 모션 정보를 시그널링하는 경우, 비디오 인코더 (20) 는 현재 PU에 대한 RefPicListX 모션 벡터 예측자 (motion vector predictor; MVP) 신택스 엘리먼트 (예컨대, 플래그) 를 시그널링하는 것에 더하여 현재 PU에 대한 RefPicListX 모션 벡터 차이 (motion vector difference; MVD) 와 현재 PU에 대한 RefPicListX 참조 인덱스를 시그널링할 수도 있다. 현재 PU에 대한 RefPicListX MVP 신택스 엘리먼트는 AMVP 후보 리스트에서 선택된 AMVP 후보의 포지션을 나타낼 수도 있다. 현재 PU에 대한 RefPicListX MVD는 현재 PU의 RefPicListX 모션 벡터와 선택된 AMVP 후보의 모션 벡터 사이의 차이를 나타낼 수도 있다. 이런 식으로, 비디오 인코더 (20) 는 RefPicListX MVP 신택스 엘리먼트, RefPicListX 참조 인덱스 값, 및 RefPicListX MVD를 시그널링함으로써 현재 PU의 RefPicListX 모션 정보를 시그널링할 수도 있다. 다르게 말하면, 현재 PU에 대한 모션 벡터를 나타내는 비트스트림에서의 데이터는 참조 인덱스, 후보 리스트에 대한 인덱스, 및 MVD를 표현하는 데이터를 포함할 수도 있다. 따라서, 선택된 모션 벡터들은 후보 리스트에 대한 인덱스를 송신함으로써 시그널링될 수도 있다. 덧붙여서, 참조 인덱스 값들과 모션 벡터 차이들이 또한 시그널링될 수도 있다.

[0074] 더욱이, 현재 PU의 모션 정보가 AMVP 모드를 사용하여 시그널링되는 경우, 비디오 디코더 (30) 는, 비트스트림으로부터, 현재 PU에 대한 MVD와 MVP 신택스 엘리먼트를 획득할 수도 있다. 비디오 디코더 (30) 는 동일한 AMVP 후보 리스트를 생성할 수도 있고, MVP 신택스 엘리먼트에 기초하여, 선택된 AMVP 후보를 결정할 수도 있다. 다르게 말하면, AMVP에서, 각각의 모션 가정에 대한 모션 벡터 예측자들의 후보 리스트가 코딩된 참조 인덱스에 기초하여 도출된다. 앞서와 같이, 이 리스트는 동일한 참조 인덱스와 연관되는 이웃 블록들의 모션 벡터들뿐만 아니라 시간적 참조 화상에서 병치된 블록의 이웃 블록의 모션 파라미터들에 기초하여 도출되는 시간적 모션 벡터 예측자를 포함할 수도 있다. 비디오 디코더 (30) 는 MVD를 선택된 AMVP 후보에 의해 나타내어진 모션 벡터에 가산함으로써 현재 PU의 모션 벡터를 복구할 수도 있다. 다시 말하면, 비디오 디코더 (30) 는, 선택된 AMVP 후보 및 MVD에 의해 나타내어진 모션 벡터에 기초하여, 현재 PU의 모션 벡터를 결정할 수도 있다. 비디오 디코더 (30) 는 그 다음에 현재 PU의 복구된 모션 벡터 또는 모션 벡터들을 사용하여 현재 PU에 대한 예측 블록들을 생성할 수도 있다.

[0075] 비디오 코더가 현재 PU에 대한 병합 후보 리스트 또는 AMVP 후보 리스트를 생성하는 경우, 비디오 코더는 현재 PU에 공간적으로 이웃하는 로케이션들을 커버하는 PU들 (즉, 공간적으로 이웃하는 PU들) 의 모션 정보에 기초하여 하나 이상의 후보들을 도출할 수도 있고 비디오 코더는 현재 PU에 시간적으로 이웃하는 PU들의 모션 정보에 기초하여 하나 이상의 후보들을 도출할 수도 있다. 본 개시물에서, PU (또는 다른 유형의 블록) 는, 그 PU와 연관된 예측 블록 (또는 그 블록과 연관된 다른 유형의 샘플 블록) 이 로케이션을 포함하면, 그 로케이션을 "커버"한다고 말해질 수도 있다. 더욱이, 본 개시물에서, 제 1 PU의 예측 블록이 화상 내에서 제 2 PU의 예측 블록에 인접할 때, 제 1 PU는 제 2 PU와 공간적으로 이웃할 수도 있다. 후보 리스트는 동일한 참조 인덱

스와 연관되는 이웃 블록들의 모션 벡터들뿐만 아니라 시간적 참조 화상에서 블록의 모션 파라미터들 (즉, 모션 정보) 에 기초하여 도출되는 시간적 모션 벡터 예측자를 포함할 수도 있다.

[0076] 도 2는 현재 PU (40) 에 관하여 예의 공간적으로 이웃하는 PU들을 도시하는 개념도이다. 도 2의 예에서, 공간적으로 이웃하는 PU들은 A_0, A_1, B_0, B_1, 및 B_2로서 나타내어진 로케이션들을 커버하는 PU들일 수도 있다. 다르게 말하면, 현재 PU (40) 와 그것의 공간적 이웃 PU들 사이의 일 예의 관계가 도 2에 묘사되어 있다.

[0077] 공간적 이웃 PU들에 관해, 다음의 심볼들이 정의될 수도 있다:

[0078] - 루마 로케이션 (xP, yP) 이 현재 화상의 좌측상단 샘플에 관하여 현재 PU의 좌측상단 루마 샘플을 특정하는데 사용되며;

[0079] - 변수들 (nPSW 및 nPSH) 은 루마에 대한 PU의 폭 및 높이를 나타내며;

[0080] - 현재 화상의 좌측 상단 샘플을 기준으로 현재 PU N의 좌측 상단 루마 샘플은 (xN, yN) 이다.

[0081] (xN, yN) (여기서 N은 A_0, A_1, B_0, B_1 또는 B_2에 의해 대체된다) 은 각각 (xP - 1, yP + nPSH), (xP - 1, yP + nPSH - 1), (xP + nPSW, yP - 1), (xP + nPSW - 1, yP - 1) 또는 (xP - 1, yP - 1) 로서 정의된다.

[0082] 현재 PU에 시간적으로 이웃하는 PU (즉, 현재 PU와는 상이한 시간 인스턴스와 연관되는 PU) 의 모션 정보에 기초하는 병합 후보 리스트 또는 AMVP 후보 리스트에서의 후보는 TMVP라고 지칭될 수도 있다. TMVP가 HEVC의 코딩 효율을 개선하는데 사용될 수도 있고, 다른 코딩 도구들과는 달리, TMVP가 디코딩된 화상 버퍼에서의 화상 (예컨대, 참조 화상 리스트에서의 화상) 의 모션 벡터에 액세스할 있을 수도 있다.

[0083] TMVP를 결정하기 위해, 비디오 코더가 현재 PU와 병치되는 PU를 포함하는 참조 화상을 맨 먼저 식별할 수도 있다. 다르게 말하면, 비디오 코더는 이른바 "병치된 화상"을 식별할 수도 있다. 현재 화상의 현재 슬라이스가 B 슬라이스 (즉, 양 방향으로 인터 예측된 PU들을 포함하는 것이 허용된 슬라이스) 이면, 비디오 인코더 (20) 는, 슬라이스 헤더에서, 병치된 화상이 RefPicList0으로부터 유래하는지 또는 RefPicList1로부터 유래하는지를 나타내는 신택스 엘리먼트 (예컨대, collocated_from_l0_flag) 를 시그널링할 수도 있다. 다르게 말하면, TMVP들의 사용이 현재 슬라이스에 대해 가능하게 되고 현재 슬라이스가 B 슬라이스 (예컨대, 양 방향으로 인터 예측된 PU들을 포함하는 것이 허용된 슬라이스) 인 경우, 비디오 인코더 (20) 는 병치된 화상이 RefPicList0에 있는지 또는 RefPicList1에 있는지를 나타내는 신택스 엘리먼트 (예컨대, collocated_from_l0_flag) 를 슬라이스 헤더에서 시그널링할 수도 있다.

[0084] 슬라이스 헤더에서의 신택스 엘리먼트 (예컨대, collocated_ref_idx) 가 식별된 참조 화상 리스트에서의 병치된 화상을 나타낼 수도 있다. 따라서, 비디오 디코더 (30) 가 병치된 화상을 포함하는 참조 화상 리스트를 식별한 후, 비디오 디코더 (30) 는 슬라이스 헤더에서 시그널링될 수도 있는 collocated_ref_idx를 사용하여, 식별된 참조 화상 리스트에서의 병치된 화상을 식별할 수도 있다. 비디오 코더는 병치된 화상을 체크함으로써 병치된 PU를 식별할 수도 있다. TMVP는 병치된 PU의 우측하단 PU의 모션 정보 또는 병치된 PU의 중앙 PU의 모션 정보 중 어느 하나를 나타낼 수도 있다.

[0085] 비디오 코더가 시간적 참조 화상에서 TMVP의 모션 벡터를 식별하는 모션 벡터 후보 (예컨대, AMVP 후보 리스트의 병합 리스트에서의 후보) 를 생성하는 경우, 비디오 코더는 시간적 참조 화상의 (POC 값이 반영된) 시간적 로케이션에 기초하여 TMVP의 모션 벡터를 스케일링할 수도 있다. 다르게 말하면, 비디오 코더는 현재 화상과 참조 화상 간의 POC 거리에 기초하여 모션 벡터 후보의 모션 벡터를 스케일링할 수도 있다. 예를 들면, 비디오 코더가 제 1 화상과 제 2 화상 사이의 POC 거리에 기초하여 모션 벡터를 스케일링하는 경우, 비디오 코더는, 제 1 화상 및 제 2 화상의 POC 값들 간의 차이가 작은 경우보다 제 1 화상 및 제 2 화상의 POC 값들 간의 차이가 더 큰 경우에, 더 큰 양만큼 모션 벡터의 크기를 증가시킬 수도 있다.

[0086] TMVP로부터 도출된 시간적 병합 후보에 대한 모든 가능한 참조 화상 리스트들의 타겟 참조 인덱스는 항상 0으로 설정될 수도 있다. 타겟 참조 인덱스는 모션 보상을 위해 사용되는 참조 화상을 식별할 수도 있다. 그러나, AMVP의 경우, 모든 가능한 참조 화상들의 타겟 참조 인덱스는 디코딩된 참조 인덱스와 동일하게 설정된다. HEVC에서, SPS가 플래그 (예컨대, sps_temporal_mvp_enable_flag) 를 포함할 수도 있고 슬라이스 헤더는 sps_temporal_mvp_enable_flag가 1과 동일한 경우 플래그 (예컨대, pic_temporal_mvp_enable_flag) 를 포함할 수도 있다. pic_temporal_mvp_enable_flag 및 temporal_id 양쪽 모두가 특정 화상에 대해 0과 동일한 경우, 디코딩 순서에서 그 특정 화상 전의 화상들로부터의 모션 벡터는 특정 화상 또는 디코딩 순서에서

특정 화상 후의 화상의 디코딩에 있어서 TMVP로서 사용되지 않는다.

[0087] 본 개시물의 기법들은 MV-HEVC 및 3D-HEVC를 포함한 멀티-뷰 코딩 및/또는 3DV 표준들 및 규격들에 잠재적으로 적용가능하다. MV-HEVC에서, 하이-레벨 신택스 (high-level syntax; HLS) 변경들이 있을 수도 있어서, HEVC에서 CU 또는 PU 레벨에서의 모듈은 재설계될 필요가 없다. 이는 HEVC를 위해 구성된 모듈들이 MV-HEVC에 대해 재사용되는 것을 허용할 수도 있다. 3D-HEVC의 경우, CU 및/또는 PU 레벨에서의 코딩 도구들을 포함한 새로운 코딩 도구들은, 텍스처 뷰 및 깊이 뷰 양쪽 모두에 대해, 포함되고 지원될 수도 있다.

[0088] HEVC에 대한 상이한 코덱 확장본들 (예컨대, MV-HEVC, 3D-HEVC, SHVC 등) 은 HEVC에서 정의된 다양한 신택스 구조들에 대한 상이한 확장본들을 정의할 수도 있다. 신택스 구조들에 대한 확장본들은 코덱 확장본들에 특유한 신택스 엘리먼트들을 포함할 수도 있다. 예를 들어, MV-HEVC는 VPS들에 대한 확장본을 정의할 수도 있고 3D-HEVC는 VPS들에 대한 상이한 확장본을 정의할 수도 있다.

[0089] MV-HEVC와 3D-HEVC에서 정의된 것과 같은 멀티-뷰 코딩에서, 동일한 장면의 상이한 관점들로부터의 다수의 뷰들이 있을 수도 있다. 멀티-뷰 코딩 및 3DV 코딩의 맥락에서, "액세스 유닛"이라는 용어는 동일한 시간 인스턴스에 대응하는 화상들의 세트를 지칭하는데 사용된다. 구체적으로는, MV-HEVC 및 3D-HEVC에서, 액세스 유닛이, 디코딩 순서에서 연속적이고 하나 이상의 뷰 성분들로 이루어진 정확히 하나의 코딩된 화상을 포함하는 NAL 유닛들의 세트일 수도 있다. 코딩된 화상의 코딩된 슬라이스 NAL 유닛들에 더하여, 액세스 유닛은 코딩된 화상의 슬라이스들을 포함하지 않는 다른 NAL 유닛들을 또한 포함할 수도 있다. 일부 예들에서, 액세스 유닛의 디코딩은 하나 이상의 디코딩된 뷰 성분들로 이루어진 하나의 디코딩된 화상이 항상 생겨나게 한다. 따라서, 비디오 데이터는 시간이 지남에 따라 발생하는 일련의 액세스 유닛들로서 개념화될 수도 있다. "뷰 성분"이 단일 액세스 유닛에서의 뷰의 코딩된 표현일 수도 있다. 뷰 성분이 텍스처 뷰 성분 및 깊이 뷰 성분을 포함할 수도 있다. 본 개시물에서, "뷰"가 동일한 뷰 식별자와 연관된 뷰 성분들의 시퀀스를 지칭할 수도 있다.

[0090] 텍스처 뷰 성분 (즉, 텍스처 화상) 이 단일 액세스 유닛에서의 뷰의 텍스처의 코딩된 표현일 수도 있다. 텍스처 뷰가 뷰 순서 인덱스의 동일한 값과 연관된 텍스처 뷰 성분들의 시퀀스일 수도 있다. 뷰의 뷰 순서 인덱스가 그 뷰의 다른 뷰들에 대한 카메라 포지션을 나타낼 수도 있다. 깊이 뷰 성분 (즉, 깊이 화상) 이 단일 액세스 유닛에서의 뷰의 깊이의 코딩된 표현일 수도 있다. 깊이 뷰가 뷰 순서 인덱스의 동일한 값과 연관된 깊이 뷰 성분들의 시퀀스일 수도 있다.

[0091] 멀티-뷰 코딩, 3DV 코딩, 및 스케일러블 비디오 코딩에서, 비트스트림이 복수의 계층들을 가질 수도 있다. MV-HEVC 및 3D-HEVC에서 정의된 것과 같은 멀티-뷰 코딩과 3DV 코딩에서, 계층들은 상이한 뷰들에 대응할 수도 있다. 비디오 디코더 (예컨대, 비디오 디코더 (30)) 가 계층과 연관된 화상들을 임의의 다른 계층에서의 화상들에 대한 참조 없이 디코딩할 수 있다면, 그 뷰는 "기본 계층 (base layer)" (또는 "기본 뷰 (base view)")라고 지칭될 수도 있다. 계층의 디코딩이 하나 이상의 다른 계층들 (예컨대, 뷰들) 과 연관된 화상들의 디코딩에 의존한다면 그 계층은 비-기본 계층 (예컨대, 비-기본 뷰) 이라고 지칭될 수도 있다.

[0092] SVC에서, 기본 계층 이외의 계층들은 "향상 계층 (enhancement layer) 들"이라고 지칭될 수도 있고 비트스트림으로부터 디코딩된 비디오 데이터의 시각적 품질을 향상시키는 정보를 제공할 수도 있다. 스케일러블 비디오 코딩 (예컨대, SHVC) 에서, "계층 표현"이 단일 액세스 유닛에서의 공간적 계층의 코딩된 표현일 수도 있다. 설명의 편의를 위해, 본 개시물은 뷰 성분들 및/또는 계층 표현들을 "뷰 성분들/계층 표현들"이라고 지칭할 수도 있다.

[0093] 계층들을 구현하기 위해, NAL 유닛들의 헤더들은 nuh_reserved_zero_6bits 신택스 엘리먼트들을 포함할 수도 있다. 상이한 값들을 특정하는 nuh_reserved_zero_6bit 신택스 엘리먼트들을 갖는 NAL 유닛들은 비트스트림의 상이한 "계층들"에 속한다. 따라서, 멀티-뷰 코딩, 3DV, 또는 SVC에서, NAL 유닛의 nuh_reserved_zero_6bits 신택스 엘리먼트는 NAL 유닛의 계층 식별자 (즉, 계층 ID) 를 특정한다. 일부 예들에서, NAL 유닛이 멀티-뷰 코딩, 3DV 코딩, 또는 SVC에서의 기본 계층에 관련되면, 그 NAL 유닛의 nuh_reserved_zero_6bits 신택스 엘리먼트는 0과 동일하다. 비트스트림의 기본 계층에서의 데이터는 그 비트스트림의 임의의 다른 계층에서의 데이터에 대한 참조 없이 디코딩될 수도 있다. NAL 유닛이 멀티-뷰 코딩, 3DV, 또는 SVC에서의 기본 계층에 관련되지 않으면, nuh_reserved_zero_6bits 신택스 엘리먼트는 0이 아닌 값을 가질 수도 있다. 위에서 나타낸 바와 같이, 멀티-뷰 코딩 및 3DV 코딩에서, 비트스트림의 상이한 계층들은 상이한 뷰들에 대응할 수도 있다.

[0094] 더욱이, 계층 내의 일부 뷰 성분들/계층 표현들이 동일한 계층 내의 다른 뷰 성분들/계층 표현들에 대한 참조 없이 디코딩될 수도 있다. 따라서, 계층의 특정한 뷰 성분들/계층 표현들의 데이터를 캡슐화하는 NAL 유닛들은 그 계층에서의 다른 뷰 성분들/계층 표현들의 디코딩능력 (decodability) 에 영향을 미치는 일 없이 비트스트림으로부터 제거될 수도 있다. 이러한 뷰 성분들/계층 표현들의 데이터를 캡슐화하는 NAL 유닛들을 제거하는 것은 비트스트림의 프레임 레이트를 떨어뜨릴 수도 있다. 계층 내의 다른 뷰 성분들/계층 표현들에 대한 참조 없이 디코딩될 수도 있는 그 계층 내의 뷰 성분들/계층 표현들의 서브세트가 "서브-계층" 또는 "시간적 서브-계층"이라고 본원에서 지칭될 수도 있다.

[0095] NAL 유닛들은 NAL 유닛들의 시간적 식별자들을 특정하는 temporal_id 신택스 엘리먼트들을 포함할 수도 있다. NAL 유닛의 시간적 식별자는 그 NAL 유닛이 속하는 서브-계층을 식별한다. 따라서, 비트스트림의 각각의 서브-계층이 상이한 시간적 식별자를 가질 수도 있다. 일반적으로, 제 1 NAL 유닛의 시간적 식별자가 제 2 NAL 유닛의 시간적 식별자보다 작으면, 제 1 NAL 유닛에 의해 캡슐화된 데이터는 제 2 NAL 유닛에 의해 캡슐화된 데이터에 대한 참조 없이 디코딩될 수도 있다.

[0096] 멀티-뷰 코딩은 뷰 간 예측을 지원할 수도 있다. 뷰 간 예측은 H.264/AVC, HEVC, 또는 다른 비디오 코딩 사양들에서 사용된 인터 예측과 유사하고, 동일한 신택스 엘리먼트들을 사용할 수도 있다. 그러나, 비디오 코더가 현재 블록 (이를테면 매크로블록 또는 PU) 에 대해 뷰 간 예측을 수행하는 경우, 비디오 코더는, 참조 화상으로서, 현재 블록과 동일한 액세스 유닛에 있지만 상이한 뷰에 있는 화상을 사용할 수도 있다. 다르게 말하면, 멀티-뷰 코딩에서, 뷰들 간의 상관을 제거하기 위해 동일한 액세스 유닛의 (즉, 동일한 시간 인스턴스 내의) 상이한 뷰들에서 캡처된 화상들 중에서 뷰 간 예측이 수행된다. 그 반면, 기존의 인터 예측은 상이한 액세스 유닛들에서의 화상들만을 참조 화상들로서 사용한다.

[0097] 따라서, 비-기본 뷰에서 화상 (즉, 현재 화상) 을 코딩하는 경우, 비디오 코더 (이를테면 비디오 인코더 (20) 또는 비디오 디코더 (30)) 는 참조 화상 리스트에 뷰 간 참조 화상을 포함시킬 수도 있다. 뷰 간 참조 화상은 현재 화상과는 상이한 뷰에 그리고 현재 화상과 동일한 시간 인스턴스 (즉, 액세스 유닛) 에 있다. 비디오 코더는 뷰 간 참조 화상을 참조 화상 리스트의 임의의 포지션에 삽입할 수도 있다. 다르게 말하면, 뷰 간 예측으로 코딩된 화상이 다른 비-기본 뷰들의 뷰 간 예측을 위해 참조 화상 리스트에 추가될 수도 있다.

[0098] 도 3은 멀티-뷰 코딩을 위한 일 예의 예측 구조를 도시하는 개념도이다. 도 3의 멀티-뷰 예측 구조는 시간적 및 뷰 간 예측을 포함한다. 도 3의 예에서, 각각의 정사각형은 뷰 성분에 대응한다. 도 3의 예에서, 액세스 유닛들은 T0...T11로 라벨 표시되고 뷰들은 S0...S7로 라벨 표시된다. "I"로 라벨 표시된 정사각형들은 인트라 예측된 뷰 성분들이다. "P"로 라벨 표시된 정사각형들은 단방향으로 인터 예측된 뷰 성분들이다. "B" 및 "b"로 라벨 표시된 정사각형들은 양방향으로 인터 예측된 뷰 성분들이다. "b"로 라벨 표시된 정사각형들은 "B" 라벨 표시된 정사각형들을 참조 화상들로서 사용할 수도 있다. 제 1 정사각형에서부터 제 2 정사각형을 가리키는 화살표는, 제 1 정사각형이 인터 예측에서 제 2정사각형에 대한 참조 화상으로서 이용가능하다는 것을 나타낸다. 도 3에서 수직 화살표들에 의해 나타낸 바와 같이, 동일한 액세스 유닛의 상이한 뷰들에서의 뷰 성분들은 참조 화상들로서 이용가능할 수도 있다. 하나의 액세스 유닛의 하나의 뷰 성분의 동일한 액세스 유닛의 다른 뷰 성분에 대한 참조 화상으로서의 사용은 뷰 간 예측이라고 지칭될 수도 있다.

[0099] H.264/AVC의 MVC 확장본에서, 뷰 간 예측은 디스패리티 모션 보상에 의해 지원되는데, 이 디스패리티 모션 보상은 H.264/AVC 모션 보상의 신택스를 사용하지만, 상이한 뷰에서의 화상이 참조 화상으로서 사용되는 것을 허용한다. 2 개의 뷰들의 코딩은 H.264/AVC의 MVC 확장본에 의해 또한 지원될 수도 있다. H.264/AVC의 MVC 확장본의 장점들 중 하나는, MVC 인코더가 2 개를 초과하는 뷰들을 3D 비디오 입력으로서 사용할 수도 있고 MVC 디코더는 이러한 멀티뷰 표현을 디코딩할 수도 있다는 것이다. 결과적으로, MVC 디코더를 갖는 임의의 렌더러 (renderer) 는 2 개를 초과하는 뷰들을 갖는 3D 비디오 콘텐츠를 기대할 수도 있다.

[0100] 멀티-뷰 비디오 코딩의 맥락에서, 2 종류의 모션 벡터들이 있다. 한 종류의 모션 벡터는 시간적 참조 화상을 가리키는 정상적인 모션 벡터이다. 정상적인 시간적 모션 벡터에 대응하는 인터 예측의 유형은 모션 보상 예측 (motion-compensated prediction; MCP) 이라고 지칭될 수도 있다. 뷰 간 예측 참조 화상이 모션 보상을 위해 사용되는 경우, 대응하는 모션 벡터는 "디스패리티 모션 벡터"라고 지칭될 수도 있다. 다르게 말하면, 디스패리티 모션 벡터가 상이한 뷰에서의 화상 (즉, 디스패리티 참조 화상 또는 뷰 간 참조 화상) 을 가리킨다. 디스패리티 모션 벡터에 대응하는 인터 예측의 유형은 "디스패리티 보상 예측 (disparity-compensated prediction)" 또는 "DCP"라고 지칭될 수도 있다.

[0101] 스케일러블 비디오 코딩은 본 개시물에서 설명된 뷰 간 예측에 유사한 방식으로 계층 간 예측을 구현할 수도 있다. 대체로, 계층 간 예측은 현재 화상에 대한 것과는 다른 값의 nuh_layer_id를 갖는 참조 화상들의 데이터 엘리먼트들 (예컨대 샘플 값들 또는 모션 벡터들) 에 의존하는 방식의 예측이다. 따라서, 계층 간 예측에 사용되는 참조 화상이 "계층 간 참조 화상"이라고 지칭될 수도 있다. 설명의 편의를 위해, 본 개시물은 뷰 간 참조 화상들 및/또는 계층 간 참조 화상들을 "뷰/계층 간 참조 화상들"이라고 지칭할 수도 있다.

[0102] MV-HEVC와 3D-HEVC는 뷰 간 모션 예측 및 뷰 간 잔차 예측을 사용하여 코딩 효율을 개선할 수도 있다. 뷰 간 모션 예측에서, 비디오 코더가 현재 PU와는 상이한 뷰에서의 PU의 모션 정보에 기초하여 현재 PU의 모션 정보를 결정 (즉, 예측) 할 수도 있다. 다르게 말하면, 현재 PU의 대응 블록이 디스패리티 벡터에 의해 식별되고, 대응 블록의 모션 벡터들은 현재 PU의 AMVP 또는 병합 리스트의 부가적인 후보로서 사용될 수도 있다. 덧붙여서, 디스패리티 벡터는 디스패리티 모션 벡터로 변환되고 AMVP 또는 병합 리스트에 추가될 수도 있다. 뷰 간 잔차 예측에서, 비디오 코더가 현재 CU와는 상이한 뷰에서의 잔차 데이터에 기초하여 현재 CU의 잔차 블록들을 결정할 수도 있다. 다시 말하면, 뷰 간 잔차 예측에서, 현재 CU의 대응 블록들이 0이 아닌 잔차 화소들을 포함하면, 블록들의 잔차는 현재 CU의 잔차를 예측하는데 사용된다.

[0103] 뷰 간 모션 예측과 뷰 간 잔차 예측을 가능하게 하기 위해, 비디오 코더가 블록들 (예컨대, PU들, CU들 등) 에 대한 디스패리티 벡터들을 결정할 수도 있다. 대체로, 디스패리티 벡터가 2 개의 뷰들 간의 변위의 추정자 (estimator) 로서 사용된다. 비디오 코더가 뷰 간 모션 또는 잔차 예측을 위한 참조 블록을 다른 뷰에서 로케이팅하기 위해 블록에 대한 디스패리티 벡터를 사용할 수도 있거나, 또는 비디오 코더는 뷰 간 모션 예측을 위해 디스패리티 벡터를 디스패리티 모션 벡터로 변환할 수도 있다.

[0104] 『L. Zhang et al., "3D-CE5.h related: Advanced residual prediction for multiview coding", Joint Collaborative Team on 3D Video Coding Extension Development of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11, 2nd Meeting: Shanghai, CN, 13-19 Oct. 2012』, 문서 JCT3V-B0051 (이후로는, "JCT3V-B0051") 이, 뷰 간 잔차 예측의 코딩 효율을 더욱 개선하기 위해 고급 잔차 예측 (ARP) 법을 제안했다.

[0105] 도 4는 멀티-뷰 및 3D 비디오 코딩에서 ARP의 일 예의 예측 구조를 도시하는 개념도이다. 도 4는 4 개의 화상들, 즉, 현재 화상 (70), 시간적 참조 화상 (72), 디스패리티 참조 화상 (74), 및 시간적-디스패리티 참조 화상 (76) 을 포함한다. 현재 화상 (70) 은 뷰 V1과 연관되고 시간 인스턴스 T_j와 연관된다. 시간적 참조 화상 (72) 이 뷰 V1과 연관되고 시간 인스턴스 T_i와 연관된다. 디스패리티 참조 화상 (74) 이 뷰 V0와 연관되고 시간 인스턴스 T_j와 연관된다. 시간적-디스패리티 참조 화상 (76) 이 뷰 V0와 연관되고 시간 인스턴스 T_i와 연관된다.

[0106] 현재 화상 (70) 은 "D_c"로서 표시된 현재 PU를 포함한다. 다르게 말하면, D_c는 현재 뷰 (뷰 1) 에서의 현재 블록을 나타낸다. D_c는 시간적 참조 화상 (72) 내의 로케이션을 나타내는 시간적 모션 벡터 V_D를 갖는다. 비디오 인코더 (20) 는 시간적 모션 벡터 V_D에 의해 나타내어진 로케이션과 연관되는 화상 (72) 에서의 샘플들에 기초하여 시간적 참조 블록 D_r을 결정할 수도 있다. 따라서, D_r은 시간 T_i에서 동일한 뷰 (뷰 1) 로부터의 D_c의 시간적 예측 블록을 나타내고 V_D는 D_c 로부터 D_r까지의 모션을 나타낸다.

[0107] 더욱이, 비디오 인코더 (20) 는 D_c의 디스패리티 벡터에 의해 나타내어진 로케이션과 연관되는 디스패리티 참조 화상 (74) 에서의 샘플들에 기초하여 디스패리티 참조 블록 B_c를 결정할 수도 있다. 따라서, B_c는 참조 블록 (즉, 시간 T_j에서 참조 뷰 (뷰 0) 에서의 D_c 의 표현) 을 나타낸다. B_c의 좌측상단 포지션은 도출된 디스패리티 벡터와 D_c의 좌측상단 포지션을 가산함으로써 도출된 디스패리티 벡터로 계산될 수 있다. D_c와 B_c가 2 개의 상이한 뷰들에서의 동일한 대상의 투영들일 수도 있고, D_c와 B_c는 동일한 모션 정보를 공유해야 한다. 그러므로, 시간 T_i에서 뷰 0에서의 B_c의 시간적 예측 블록 B_r은 V_D의 모션 정보를 적용함으로써 B_c로부터 로케이팅될 수 있다.

[0108] 비디오 인코더 (20) 는 시간적-디스패리티 화상 (76) 에서 시간적-디스패리티 참조 블록 B_r (B_c의 예측 블록) 을 결정할 수도 있다. 위에서 나타낸 바와 같이, 시간적-디스패리티 화상 (76) 은 B_r과 동일한 뷰 (즉, 뷰 V0) 와 연관되고 D_r과 동일한 시간 인스턴스 (즉, 시간 인스턴스 T_i) 와 연관된다. 비디오 인코더 (20) 는 D_c의

모션 벡터 V_D에 의해 나타내어진 로케이션에서의 샘플들에 기초하여 B_r을 결정할 수도 있다. 따라서, B_r의 좌측상단 포지션은 B_c의 좌측상단 포지션에 모션 벡터 V_D를 가산함으로써 재사용 모션 벡터 V_D로 계산될 수 있다. B_c의 좌측상단 포지션은 D_c의 좌측상단 포지션과 디스패리티 벡터의 합과 동일할 수 있다. 따라서, B_r의 좌측상단 포지션은 D_c의 좌측상단 포지션의 좌표들, 디스패리티 벡터, 및 모션 벡터 V_D의 합과 동일할 수도 있다. 이런 식으로, 도 4에서 화살표 78에 의해 도시된 바와 같이, 비디오 인코더 (20) 는 B_r을 결정하기 위해 모션 벡터 V_D를 재사용할 수도 있다.

[0109] 더욱이, ARP에서, 제 1 잔차 블록에서의 각각의 샘플은 D_c에서의 샘플과 D_r의 대응하는 샘플 간의 차이를 나타낼 수도 있다. 제 1 잔차 블록은 D_c에 대한 원래의 잔차 블록이라고 지칭될 수도 있다. 제 2 잔차 블록에서의 각각의 샘플은 B_c에서의 샘플과 B_r에서의 대응하는 샘플 간의 차이를 나타낼 수도 있다. 제 2 잔차 블록은 "잔차 예측자"라고 지칭될 수도 있다. 비디오 인코더 (20) 가 B_r을 결정하기 위해 모션 벡터 V_D를 사용하기 때문에, 잔차 예측자는 B_c의 실제 잔차 데이터와는 상이할 수도 있다.

[0110] 비디오 인코더 (20) 가 잔차 예측자를 결정한 후, 비디오 인코더 (20) 는 잔차 예측자와 가중 계수를 곱할 수도 있다. 다르게 말하면, V_D의 모션 정보를 갖는 B_c의 잔차는 가중 계수가 곱해지고 현재 잔차에 대한 잔차 예측자로서 사용된다. 가중 계수는 0, 0.5, 또는 1과 동일할 수도 있다. 따라서, 3 개의 가중 계수들 (즉, 0, 0.5, 및 1) 이 ARP에서 사용될 수도 있다. 비디오 인코더 (20) 가 잔차 예측자와 가중 계수를 곱한 후, 잔차 예측자는 가중된 잔차 예측자라고 지칭될 수도 있다. 비디오 인코더 (20) 는, 최종 가중 계수로서, 현재 CU (즉, 현재 PU를 포함하는 CU) 에 대한 최소 레이트-왜곡 비용을 이끄는 가중 계수를 선택할 수도 있다. 비디오 인코더 (20) 는, CU 레벨에서, 비트스트림에, 가중 지수 (weighting index) 를 나타내는 데이터를 포함시킬 수도 있다. 가중 지수는 현재 CU에 대한 최종 가중 계수 (즉, 가중된 잔차 예측자를 생성하는데 사용되었던 가중 계수) 를 나타낼 수도 있다. 일부 예들에서, 0, 1, 및 2의 가중 지수들은 0, 1, 및 0.5의 가중 계수들에 각각 대응한다. 현재 CU에 대한 0의 가중 계수의 선택은 현재 CU의 PU들 중 임의의 것에 대해 ARP를 사용하지 않는 것과 동등하다.

[0111] 그 다음에 비디오 인코더 (20) 는 현재 PU에 대한 최종 잔차 블록을 결정할 수도 있다. 현재 PU에 대한 최종 잔차 블록에서의 각각의 샘플은 원래의 잔차 블록에서의 샘플과 가중된 잔차 예측자에서의 대응 샘플 간의 차이를 나타낼 수도 있다. 현재 CU (즉, 현재 PU를 포함한 CU) 의 잔차 블록이 현재 PU에 대한 최종 잔차 블록을, 만약 있다면, 현재 CU의 다른 PU들에 대한 잔차 블록들과 함께 포함할 수도 있다. 본 개시물의 다른 곳에서 설명되는 바와 같이, 비디오 인코더 (20) 는 현재 CU의 잔차 블록을 하나 이상의 변환 블록들 간에 구획화할 수도 있다. 변환 블록들의 각각은 현재 CU의 TU와 연관될 수도 있다. 각각의 변환 블록에 대해, 비디오 인코더 (20) 는 변환 계수 블록을 생성하기 위해 변환 블록에 하나 이상의 변환들을 적용할 수도 있다. 비디오 인코더 (20) 는, 비트스트림에, 변환 계수 블록의 양자화된 변환 계수들을 나타내는 데이터를 포함시킬 수도 있다.

[0112] 따라서, ARP에서, 2 개의 뷰들의 잔차들 간에 높은 상관을 보장하기 위해, 비디오 인코더 (20) 는 현재 PU의 모션을 참조 뷰 화상에서의 대응 블록에 적용하여, 뷰 간 잔차 예측을 위해 사용될 기본 뷰에서의 잔차를 생성할 수도 있다. 이런 식으로, 모션은 현재 PU와 참조 뷰에서의 대응 참조 블록에 대해 정렬된다. 더구나, 예측 에러가 추가로 감소되도록 적응적 가중 계수가 잔차 신호에 적용된다.

[0113] 현재 PU가 양예측되면, 비디오 인코더 (20) 는 유사한 프로세스를 수행할 수도 있다. 예를 들면, 비디오 인코더 (20) 는 현재 PU의 RefPicList0 및 RefPicList1 모션 벡터들 양쪽 모두를 사용하여 RefPicList0 및 RefPicList1 시간적-디스패리티 참조 블록들을 결정할 수도 있다. 비디오 인코더 (20) 는 RefPicList0 및 RefPicList1 시간적-디스패리티 참조 블록의 샘플들을 보간함으로써 예측 블록을 결정할 수도 있다. 비디오 인코더 (20) 는 이 예측 블록을 위에서 설명된 방식으로 사용할 수도 있다. 비디오 디코더 (30) 는 양 예측된 PU에 대해 ARP를 수행하는 경우 유사한 프로세스를 수행할 수도 있다. 예를 들면, 비디오 디코더 (30) 는 현재 PU의 예측 블록 및 가중된 잔차 예측자를 위에서 설명된 샘플 방식으로 결정할 수도 있다. 비디오 디코더 (30) 는 비트스트림에서 시그널링된 엘리먼트들에 기초하여 현재 PU의 최종 잔차 블록을 결정할 수도 있다. 비디오 디코더 (30) 는 그 다음에 현재 PU의 최종 잔차 블록, 현재 PU의 예측 블록, 및 가중된 잔차 예측자를 가산함으로써 현재 PU의 예측 블록을 복원할 수도 있다.

[0114] 일부 예들에서, 비디오 코더는 이웃 블록 기반 디스패리티 벡터 (Neighboring Blocks Based Disparity Vector; NBDV) 법을 사용하여 블록에 대한 디스패리티 벡터를 도출할 수도 있다. 3D-HEVC는 『L. Zhang et al., "3D-CE5.h: Disparity vector generation results", Joint Collaborative Team on 3D Video Coding Extension Development of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11, 1st Meeting: Stockholm, SE, 16-20 July 2012』, 문서 JCT3V-A0097 (이후로는, "JCT3V-A0097") 에서 제안된 NBDV 도출 프로세스를 먼저 채택했다. NBDV 도출 프로세스는 그 후 추가로 개정되었다. 예를 들면, 암시적 디스패리티 벡터 (implicit disparity vector; IDV) 들이 『Sung et al., "3D-CE5.h: Simplification of disparity vector derivation for HEVC-based 3D video coding", Joint Collaborative Team on 3D Video Coding Extension Development of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11, 1st Meeting: Stockholm, SE, 16-20 July 2012』, 문서 JCT3V-A0126 (이후로는, "JCT3V-A0126") 에서 단순화된 NBDV에 포함되었다. 더욱이, 『Kang et al., "3D-CE5.h related: Improvements for disparity vector derivation", Joint Collaborative Team on 3D Video Coding Extension Development of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11, 2nd Meeting: Shanghai, CN, 13-19 Oct. 2012』, 문서 JCT3V-B0047 (이후로는, "JCT3V-B0047") 에서, NBDV 도출 프로세스는 디코딩된 화상 버퍼에 저장된 IDV들을 제거함으로써 더욱 단순화되지만, 랜덤 액세스 화상 (random access picture; RAP) 화상 선택으로 코딩 이득의 측면에서 개선을 또한 제공한다.

[0115] NBDV 도출 프로세스는 공간적 및 시간적 이웃 블록들로부터의 디스패리티 모션 벡터들을 사용하여 현재 블록에 대한 디스패리티 벡터를 도출한다. 이웃 블록들 (예컨대, 현재 블록에 공간적으로 또는 시간적으로 이웃하는 블록들) 이 비디오 코딩에서 거의 동일한 모션 및 디스패리티 정보를 공유할 가능성이 있기 때문에, 현재 블록은 이웃 블록들에서의 모션 벡터 정보를 현재 블록에 대한 디스패리티 벡터의 예측자들로서 사용할 수 있다.

다르게 말하면, 이웃 블록들이 비디오 코딩에서 거의 동일한 모션/디스패리티 정보를 공유하기 때문에, 코딩 이득을 개선시키기 위해 현재 블록은 양호한 예측자로서 이웃 블록들에서의 모션 벡터 정보를 사용할 수 있다.

이 아이디어를 추종하여, NBDV 도출 프로세스는 상이한 뷰들에서의 디스패리티 벡터를 추정하기 위해 이웃 디스패리티 정보를 사용한다.

[0116] 비디오 코더가 현재 블록에 대한 디스패리티 벡터를 결정하기 위해 NBDV 도출 프로세스를 수행하는 경우, 비디오 코더는 이웃 블록들의 2 개의 세트들을 이용할 수도 있다. 하나의 세트는 공간적으로 이웃하는 블록들로부터이고 다른 세트는 시간적으로 이웃하는 블록들로부터이다. 다르게 말하면, 여러 공간적 및 시간적 이웃 블록들이 맨 먼저 정의된다. 비디오 코더는 그 다음에 현재 블록 및 이웃 블록 간의 상관의 우선순위에 의해 결정된 미리 정의된 순서로 이웃 블록들의 각각을 체크할 수도 있다. 일단 비디오 코더가 후보들 (즉, 이웃 블록들) 에서 디스패리티 모션 벡터를 찾아내면, 비디오 코더는 그 디스패리티 모션 벡터를 현재 블록에 대한 디스패리티 벡터로 변환할 수도 있다.

[0117] NBDV 도출 프로세스의 일부 버전들에서, 비디오 코더는 디스패리티 벡터 도출을 위해 5 개의 공간적 이웃 블록들을 사용한다. 예를 들면, 비디오 코더는 다음의 공간적 이웃 블록들, 즉, 현재 블록의 좌측아래 공간적 이웃 블록, 좌측 공간적 이웃 블록, 우측위 공간적 이웃 블록, 위의 공간적 이웃 블록, 및 좌측위 공간적 이웃 블록을 체크할 수도 있다. NBDV 도출 프로세스의 일부 버전들에서, 디스패리티 벡터 도출 블록들을 위해 사용되는 5 개의 공간적 이웃 블록들은 도 2에 나타낸 바와 같이 로케이션들 (A_0, A_1, B_0, B_1, 및 B_2) 을 각각 커버할 수도 있다. 일부 예들에서, NBDV 도출 프로세스에서 사용되는 공간적 이웃 블록들은 HEVC의 병합 모드들에서 사용되는 것들과 동일하다. 그러므로, 일부 이러한 예들에서, 부가적인 메모리 액세스가 요구되지 않는다.

[0118] 일부 예들에서, 비디오 코더는 공간적 이웃 블록들을 하나씩 체크할 수도 있다. 더욱이, 일부 예들에서, 5 개의 공간적 이웃 블록들의 체킹 순서는 A_1, B_1, B_0, A_0, 및 B_2로서 정의된다.

[0119] 더욱이, 위에서 언급했듯이, 비디오 코더가 현재 블록에 대한 디스패리티 벡터 (예컨대, 현재 PU) 를 결정하는 프로세스의 부분으로서 시간적으로 이웃하는 블록들을 체크할 수도 있다. 비디오 코더가 시간적 이웃 블록들 (예컨대, 시간적 이웃 PU들) 을 체크하는 경우, 비디오 코더는 후보 화상 리스트의 구축 프로세스를 먼저 수행할 수도 있다. 비디오 코더가 후보 화상 리스트의 구축 프로세스를 수행하는 경우, 비디오 코더는 현재 뷰 (즉, 현재 블록과 연관된 뷰) 와 연관된 모든 참조 화상들을 후보 화상들로서 취급할 수도 있다. 더욱이, 비디오 코더가 후보 화상 리스트의 구축 프로세스를 수행하는 경우, 비디오 코더는 후보 화상 리스트에 이른바 "병치된 화상"을 먼저 삽입하며, 뒤이어 후보 화상들의 나머지를 참조 인덱스의 오름 차순으로 삽입할 수도 있다. 다시 말하면, 비디오 코더는 남아있는 후보 화상들이 현재 화상의 참조 화상 리스트들 (예컨대,

RefPicList0 및 RefPicList1) 에서 발생하는 순서에 따라 남아있는 후보 화상들을 후보 화상 리스트에 삽입할 수도 있다. 현재 블록을 포함하는 슬라이스의 슬라이스 헤더에서의 하나 이상의 신택스 엘리먼트들이 병치된 화상을 나타낼 수도 있다. 일부 예들에서, 참조 화상 리스트들 양쪽 모두 (예컨대, RefPicList0 및 RefPicList1) 에서 동일한 참조 인덱스를 갖는 참조 화상들이 NDBV 도출 프로세스에서의 사용을 위해 이용가능한 경우, 병치된 화상과 동일한 참조 화상 리스트에서의 참조 화상은, 후보 화상 리스트에서, 다른 참조 화상에 선행한다.

[0120] 후보 화상 리스트를 생성한 후, 비디오 코더는 후보 화상 리스트에서 후보 화상들 내의 후보 지역들을 결정할 수도 있다. 비디오 코더는 후보 지역들을 사용하여 시간적으로 이웃하는 블록들을 결정할 수도 있다. 위에서 나타낸 바와 같이, 비디오 코더는 시간적으로 이웃하는 블록의 디스패리티 모션 벡터 또는 IDV에 기초하여 현재 블록에 대한 디스패리티 벡터를 도출할 수도 있다. 일부 예들에서, 후보 화상 리스트에서의 각각의 후보 화상에 대해, 비디오 코더는 시간적으로 이웃하는 블록들을 도출하기 위한 3 개의 후보 지역들을 결정할 수도 있다. 3 개의 후보 지역들은 다음과 같이 정의될 수도 있다:

[0121] • CPU: 현재 PU 또는 현재 CU의 병치된 지역.

[0122] • CLCU: 현재 PU의 병치된 지역을 커버하는 최대 코딩 유닛 (LCU).

[0123] • BR: CPU의 우측하단 4x4 블록.

[0124] 16x16 블록에서의 더 작은 블록들이 모션 압축의 결과물과 동일한 모션 정보를 공유할 수도 있기 때문에, 비디오 코더는 디스패리티 벡터에 대해 하나의 샘플 블록만을 체크할 수도 있다. 후보 지역이 하나를 초과하는 16x16 블록을 커버하는 경우, 비디오 코더는 래스터 스캔 순서에 따라 후보 지역에서의 모든 16x16 블록들을 체크할 수도 있다. 예를 들면, 시간적으로 병치된 블록에 대한 모션 벡터가 참조 화상의 16x16 블록에 저장되고, 통상, 비디오 코더는 모션 벡터를 찾아내기 위해 4x4 블록에 액세스한다. 따라서, 일부 예들에서, 비디오 코더가 후보 블록을 16x16 블록에 배치시키면, 모든 4x4 블록들은 공통 모션 벡터를 포함하고 비디오 코더는 상이한 모션 벡터를 찾아내기 위해 모든 4x4 블록들을 체크할 필용가 없다. 한편, 후보 지역이 16x16보다 더 크면, 16x16 블록 외부의 4x4 블록들은 상이한 모션 벡터를 포함할 수도 있다.

[0125] 비디오 코더가 후보 지역 (또는 후보 지역 내의 16x16 블록) 을 체크하는 경우, 비디오 코더는 후보 지역을 커버하는 PU가 디스패리티 모션 벡터를 특정하는지의 여부를 결정할 수도 있다. 후보 지역을 커버하는 PU가 디스패리티 모션 벡터를 특정하면, 비디오 코더는 PU의 디스패리티 모션 벡터에 기초하여 현재 블록에 대한 디스패리티 벡터를 결정할 수도 있다.

[0126] 일부 예들에서, 비디오 코더는 NBDV 도출 프로세스를 수행하는 부분으로서 우선순위 기반 디스패리티 벡터 결정을 수행할 수도 있다. 예를 들어, 일단 비디오 코더가 디스패리티 모션 벡터를 포함하는 이웃 블록을 식별하면, 비디오 코더가 디스패리티 모션 벡터를 현재 블록에 대한 디스패리티 벡터로 변환하도록 비디오 코더는 디스패리티 벡터를 도출할 수도 있다. 비디오 코더는 그 다음에 뷰 간 모션 예측 및/또는 뷰 간 잔차 예측을 위해 디스패리티 벡터를 사용할 수도 있다. 일부 예들에서, 이웃 블록들의 체킹 순서는 이웃 블록들 및 현재 블록 간의 상관에 기초하여 정의된다. 예를 들면, 비디오 코더는 맨 먼저 공간적 이웃 블록들을 하나씩 체크할 수도 있다. 일단 비디오 코더가 디스패리티 모션 벡터를 식별하였다면, 비디오 코더는 디스패리티 모션 벡터를 디스패리티 벡터로서 반환한다. 일부 예들에서, 5 개의 공간적 이웃 블록들의 체킹 순서는 A_1, B_1, B_0, A_0, 및 B_2로서 정의된다.

[0127] 더욱이, 후보 화상 리스트에서의 각각의 후보 화상에 대해, 비디오 코더는 이 후보 화상에서의 3 개의 후보 지역들을 순서대로 체크할 수도 있다. 3 개의 지역들의 체킹 순서는 다음으로 정의된다: 제 1 비-기본 뷰에 대해 CPU, CLCU 및 BR 또는 제 2 비-기본 뷰에 대해 BR, CPU, CLU. 이 예에서, 제 1 비-기본 뷰와 연관된 화상들의 디코딩은 기본 뷰와 연관된 화상들의 디코딩에는 의존할 수도 있지만, 다른 뷰들과 연관된 화상들의 디코딩에는 의존하지 않는다. 더욱이, 이 예에서, 제 2 비-기본 뷰와 연관된 화상들의 디코딩은 기본 뷰 그리고, 어떤 경우들에서는, 제 1 비-기본 뷰와 연관된 화상들의 디코딩에는 의존할 수도 있지만, 만약 존재한다면, 다른 뷰들과 연관된 화상들의 디코딩에는 의존하지 않는다. 단순화를 위해, 공간적 이웃 블록들에서의 디스패리티 모션 벡터들은 공간적 디스패리티 벡터 (spatial disparity vector; SDV) 들로서 표시될 수도 있고 시간적 이웃 블록들에서의 디스패리티 모션 벡터들은 시간적 디스패리티 벡터 (temporal disparity vector; TDV) 들로서 표시될 수도 있다.

[0128] 비디오 코더가 블록 (즉, 공간적 이웃 블록, 후보 화상의 후보 지역, 또는 후보 화상의 후보 지역의 16x16 블록) 의 모션 벡터(들)를 체크하는 경우, 비디오 코더는 블록의 모션 벡터(들)가 디스패리티 모션 벡터인지의 여부를 결정할 수도 있다. 화상의 블록의 디스패리티 모션 벡터가 화상의 디스패리티 참조 화상 내의 로케이션을 가리키는 모션 벡터이다. 주어진 화상의 디스패리티 참조 화상 (또한 본원에서 뷰 간 참조 화상이라고 지칭됨) 이 주어진 화상과는 동일한 액세스 유닛과 연관되지만 주어진 화상과는 상이한 뷰와 연관되는 화상일 수도 있다. 비디오 코더가 디스패리티 모션 벡터를 식별하는 경우, 비디오 코더는 체킹 프로세스를 종료할 수도 있다. 비디오 코더는 반환된 디스패리티 모션 벡터를 디스패리티 벡터로 변환할 수도 있고 그 디스패리티 벡터를 뷰 간 모션 예측 및 뷰 간 잔차 예측을 위해 사용할 수도 있다. 예를 들어, 비디오 코더는 현재 블록에 대한 디스패리티 벡터의 수평 성분을 디스패리티 모션 벡터의 수평 성분과 동일하게 설정할 수도 있고 디스패리티 벡터의 수직 성분을 0으로 설정할 수도 있다. 다른 예에서, 비디오 코더는 디스패리티 벡터를 디스패리티 모션 벡터와 동일하게 설정함으로써 디스패리티 모션 벡터를 디스패리티 벡터로 변환할 수도 있다.

[0129] 비디오 코더가 공간적 이웃 블록 (예컨대, 공간적으로 이웃하는 PU) 을 체크하는 경우, 비디오 코더는 공간적 이웃 블록이 디스패리티 모션 벡터를 갖는지의 여부를 먼저 체크할 수도 있다. 공간적 이웃 블록들 중 어느 것도 디스패리티 모션 벡터를 갖지 않는다면, 비디오 코더는 공간적으로 이웃하는 블록들 중 임의의 것이 IDV를 갖는지의 여부를 결정할 수도 있다. 더욱이, 비디오 코더가 디스패리티 모션 벡터 또는 IDV를 식별하는 경우, 비디오 코더는 식별된 디스패리티 모션 벡터 또는 IDV를 반환할 수도 있다. "암시적 디스패리티 벡터"라는 용어는 뷰 간 모션 예측 또는 뷰 간 잔차 예측을 위해 사용되었던 디스패리티 벡터를 지칭할 수도 있다. 비록 대응 블록이 시간적 모션 예측으로 코딩될 수도 있지만, 비디오 코더는 하나 이상의 다음의 블록들을 코딩할 목적으로 도출된 디스패리티 벡터를 버리지 않는다. 이런 식으로, IDV가 디스패리티 벡터 도출의 목적으로 블록에 저장될 수도 있다.

[0130] 공간적으로 이웃하는 PU들 중 임의의 것이 IDV를 갖는지의 여부를 비디오 코더가 결정하는 경우, 비디오 코더는 공간적으로 이웃하는 PU들을 A_0, A_1, B_0, B_1, 및 B_2의 순서로 체크할 수도 있다. 따라서, 비디오 코더는 디스패리티 모션 벡터들에 대해 공간적으로 이웃하는 PU들을 A_1, B_1, B_0, A_0 및 B_2의 순서로 체크할 수도 있고 IDV들에 대해 공간적으로 이웃하는 PU들을 A_0, A_1, B_0, B_1, 및 B_2의 순서로 체크할 수도 있다. 공간적으로 이웃하는 PU들 중 하나가 IDV를 갖고 IDV가 병합/스킵 모드로서 코딩되면, 비디오 코더는 체킹 프로세스를 종료할 수도 있고 IDV를 최종 현재 블록에 대한 디스패리티 벡터로서 사용할 수도 있다.

[0131] 비디오 코더가 NBDV 도출 프로세스를 수행함으로써 현재 블록에 대한 디스패리티 벡터를 도출할 수 없다면 (즉, 디스패리티 벡터가 찾아지지 않는다면), 비디오 코더는 0의 디스패리티 벡터를 현재 블록에 대한 디스패리티 벡터로서 사용할 수도 있다. 0의 디스패리티 벡터는 0과 동일한 수평 및 수직 성분들을 갖는 디스패리티 벡터이다. 따라서, 심지어 NBDV 도출 프로세스가 이용불가능한 결과를 반환하는 경우라도, 디스패리티 벡터를 필요로 하는 비디오 코더의 다른 코딩 프로세스들은 현재 블록에 대해 0의 디스패리티 벡터를 사용할 수도 있다. 일부 예들에서, 비디오 코더가 NBDV 도출 프로세스를 수행함으로써 현재 블록에 대한 디스패리티 벡터를 도출할 수 없다면, 비디오 코더는 현재 블록에 대한 뷰 간 잔차 예측을 불가능하게 할 수도 있다. 그러나, 비디오 코더가 NBDV 도출 프로세스를 수행함으로써 현재 블록에 대한 디스패리티 벡터를 도출할 수 있는지의 여부에 상관 없이, 비디오 코더는 현재 PU에 대해 뷰 간 모션 예측을 사용할 수도 있다. 다시 말하면, 모든 미리 정의된 이웃 블록들을 체크한 후 디스패리티 벡터가 찾아지지 않으면, 0의 디스패리티 벡터가 뷰 간 모션 예측을 위해 사용될 수도 있는 한편 뷰 간 잔차 예측은 대응하는 CU에 대해 불가능하게 될 수도 있다.

[0132] 비디오 코더가 NBDV 도출 프로세스의 부분으로서 IDV들을 체크하는 일부 예들에서, 비디오 코더는 다음의 단계들을 수행할 수도 있지만, 부가적인 단계들이 다른 예들에서 또한 사용될 수도 있다. 다음의 단계들 중 임의의 것이 디스패리티 벡터를 찾아낸다면, 비디오 코더는 도출 프로세스를 종료할 수도 있다.

[0133] • 단계 1: 디스패리티 모션 벡터를 찾아내기 위해 5 개의 공간적 이웃 블록들을 A_1, B_1, B_0, A_0 및 B_2의 순서로 체크. 일단 비디오 코더가 디스패리티 모션 벡터를 찾아낸다면, 비디오 코더는 디스패리티 모션 벡터를 디스패리티 벡터로 변환한다. 공간적 이웃 블록들이 IDV들을 포함하면, 비디오 코더는 그것들의 IDV 플래그들을 "IDV 사용"으로서 마킹하고 그 IDV 플래그들의 연관된 값들을 저장한다.

[0134] • 단계 2: 시간적 모션 벡터 예측이 가능하게 되는 경우, 다음이 적용된다:

[0135] a) 현재 코딩 모드가 AMVP이면, 타겟 참조 화상 리스트에서 타겟 참조 인덱스를 갖는 참조 화상은 병치된 화상으로서 사용된다. 병치된 화상에서의 2 개의 블록들 (즉, 병치된 PU의 우측하단 블록 (BR) 과 병치된 PU의 중앙 블록 (CB)) 이 정의된다. 이 예에서, 비디오 코더는 병치된 화상의 블록들을 다음의 순서로 체크한다:

[0136] 1) BR이 디스패리티 모션 벡터를 포함하는지의 여부를 알기 위해 BR을 체크한다. 그렇다면, 비디오 코더는 디스패리티 모션 벡터를 디스패리티 벡터로 변환한다. 그렇지 않고, BR이 스킵 모드로서 코딩되고 BR이 IDV를 포함하면 (즉, IDV의 플래그가 1과 동일하면), 비디오 코더는 BR을 "IDV 사용"으로서 마킹하고 연관된 IDV를 저장한다. 비디오 코더는 그 다음에 아래의 단계 3을 수행할 수도 있다.

[0137] 2) CB가 디스패리티 모션 벡터를 포함하는지의 여부를 알기 위해 CB를 체크한다. 그렇다면, 비디오 코더는 디스패리티 모션 벡터를 디스패리티 벡터로 변환한다. 그렇지 않고, CB가 스킵 모드로서 코딩되고 CB가 IDV를 포함하면 (즉, IDV의 플래그가 1과 동일하면), 비디오 코더는 CB를 "IDV 사용"으로서 마킹하고 비디오 코더는 연관된 IDV를 저장한다. 비디오 코더는 그 다음에 단계 3을 수행할 수도 있다.

[0138] b) 현재 코딩 모드가 스킵/병합이면, 비디오 코더는 각각의 참조 화상 리스트에서 2 개의 병치된 참조 화상들을, 적용가능하면, 사용한다. 병치된 참조 화상들을 나타내는 참조 인덱스들은 좌측 이웃 PU의 참조 인덱스 또는 0과 동일할 수도 있다. 참조 화상 리스트 0 및 참조 화상 리스트 1에서의 병치된 화상들의 각각에 대해, 비디오 코더는 단계 2에서의 단계들, 즉, a) 1) 및 a) 2) 를 순서대로 수행한다.

[0139] • 단계 3: 5 개의 공간적 이웃 블록들 중 하나가 스킵 모드로서 코딩되고 공간적 이웃 블록이 IDV를 포함하면 (즉, 공간적 이웃 블록이 "IDV 사용"으로서 마킹된 플래그를 가지면), 비디오 코더는 IDV를 디스패리티 벡터로서 반환한다. 이 예에서, IDV들에 대한 공간적 이웃 블록들의 체킹 순서는 A_0, A_1, B_0, B_1, 및 B_2이다.

[0140] • 단계 4: 시간적 모션 벡터 예측이 가능하게 되고 병치된 화상에는 "IDV 사용"으로서 마킹되어 있는 하나의 블록 (즉, BR 또는 CB) 이 있다면, 비디오 코더는 그 블록과 연관된 IDV를 디스패리티 벡터로 변환한다. 일부 예들에서, 비디오 코더는 IDV의 수평 성분을 디스패리티 벡터의 수평 성분과 동일하게 설정하고 디스패리티 벡터의 수직 성분을 0과 동일하게 설정함으로써 IDV를 디스패리티 벡터로 변환할 수도 있다. 다른 예들에서, 비디오 코더는 디스패리티 벡터를 IDV와 동일하게 설정함으로써 IDV를 디스패리티 벡터로 변환할 수도 있다.

[0141] 디코딩된 화상 버퍼 (DPB) 에서 IDV에 액세스하는 것과 연관된 메모리 대역폭과 복잡도 요건들이 클 수도 있다. 예를 들면, 비디오 코더는 IDV들을 저장하고 DPB로부터 취출하기 위해 여러 메모리 액세스들을 수행할 필요가 있을 수도 있다. 이에 따라, 비디오 코더가 낮은 복잡도 NBDV 도출 프로세스를 수행할 수도 있다. 비디오 코더가 낮은 복잡도 NBDV 도출 프로세스를 수행하는 경우 비디오 코더는 더 적은 블록 후보들을 고려한다. 예를 들어, 비디오 코더는, DPB에, IDV들에 대한 정보를 저장할 수도 있다. 이 예에서, IDV들에 대한 정보는 모든 이전에 코딩된 화상들에 대한 IDV 플래그들 및 벡터들을 포함할 수도 있다. 더욱이, 낮은 복잡도 NBDV 도출 프로세스에서, DPB에서 IDV 후보들을 제거하는 것은 메모리 대역폭을 감소시킬 수 있다. 다르게 말하면, 비디오 코더는 IDV 관련된 정보를 DPB에 저장하지 않는다.

[0142] 일부 낮은 복잡도 NBDV 도출 프로세스들에서, 비디오 코더는 위에서 설명된 NBDV 도출 프로세스에서보다 후보 화상들의 더 작은 후보 지역들을 체크한다. 예를 들어, 도 5는 시간적 후보 화상의 대응하는 PU에서의 시간적 이웃들을 도시하는 개념도이다. 도 5의 예에서, 비디오 코더는 "Pos. A" 및 "Pos. B"에 의해 나타내어진 이 포지션들을 커버하는 후보 지역들을 체크할 수도 있다. 더욱이, 일부 낮은 복잡도 NBDV 도출 프로세스들에서, 비디오 코더는 병치된 화상과 랜덤 액세스 화상만의 후보 지역들을 체크할 수도 있다. 따라서, 일부 예들에서, 병치된 화상과 랜덤 액세스 화상이 시간적 블록 체크들 (즉, 도 5에 도시된 바와 같은 하단아래 블록 및 중앙 블록) 을 위해 고려된다. HEVC 및 다른 비디오 코딩 사양들에서, 랜덤 액세스는 비트스트림에서 첫 번째 코딩된 화상이 아닌 코딩된 화상으로부터 시작하는 비트스트림의 디코딩을 지칭한다. 랜덤 액세스 화상들의 예의 유형들은 IDR 화상들, CRA 화상들, 및 BLA 화상들을 포함한다.

[0143] 더욱이, 일부 낮은 복잡도 NBDV 도출 프로세스들에서, 비디오 코더는 슬라이스 또는 화상 레벨에서 한번 후보 화상 도출을 수행할 수도 있다. 다르게 말하면, 비디오 코더는 NBDV 도출 프로세스에서의 사용을 위한 후보 화상 리스트를 화상 또는 슬라이스 당 한 번 생성할 수도 있다. 결과적으로, 이러한 낮은 복잡도 NBDV 도출 프로세스들에서, 비디오 코더는 PU 또는 CU 레벨에서 후보 화상 도출 프로세스를 더 이상 인보크하지 않는다.

[0144] 각각의 코딩된 화상은 코딩된 화상 또는 코딩된 화상을 뒤따르는 (즉, 미래의) 화상 중 어느 하나에 의한 참조를 위해 사용될 수도 있는 모든 화상들을 포함하는 참조 화상 세트를 가질 수도 있다. 비디오 코더가 미래의 화상의 참조로서만 사용될 수 있는 화상들을 구별할 수도 있다. 참조 화상 리스트들은 현재 화상을 위해 사용될 수 있는 참조 화상 세트 ("RPS") (즉 "현재를 위한 RPS") 에서의 화상들에 기초하여 구축될 수도 있으며 따라서 미래의 화상들의 참조들로서만 사용될 수 있는 화상들에 기초하여 구축될 수 없다. 미래의 RPS에서의 화상은 2 개의 참조 화상 리스트들 중 임의의 것 (RefPicList0 또는 RefPicList1) 에 없을 수도 있다.

[0145] 일부 예들에서, 비디오 인코더 (20) 가 현재 화상을 인코딩하는 것을 시작하는 경우, 비디오 인코더 (20) 는 현재 화상에 대해 참조 화상들의 5 개의 서브 세트들 (즉, 참조 화상 서브 세트들) 을 생성할 수도 있다. 일부 예들에서, 이들 5 개의 참조 화상 서브 세트들은, RefPicSetStCurrBefore, RefPicSetStCurrAfter, RefPicSetStFoll, RefPicSetLtCurr 및 RefPicSetLtFoll이다. 본 개시물은 RefPicSetStCurrBefore, RefPicSetStCurrAfter, RefPicSetStFoll에서의 참조 화상들을 "단기 참조 화상들", "단기 화상들", 또는 "STRP들"이라고 지칭할 수도 있다. 따라서, "단기 참조 화상"이 (예컨대, RefPicSetStCurrBefore, RefPicSetStCurrAfter, 또는 RefPicSetStFoll에 있는 덕분에) 단기 참조를 위해 사용되고 있는 것으로서 마킹되는 화상일 수도 있다. 본 개시물은 RefPicSetLtCurr 및 RefPicSetLtFoll에서의 참조 화상들을 "장기 참조 화상들", "장기 화상들", 또는 "LTRP들"로서 지칭할 수도 있다. 비디오 인코더 (20) 는 각각의 화상에 대해 5 개의 참조 화상 서브 세트들을 재생성할 수도 있다.

[0146] 더욱이, 현재 화상이 P 슬라이스들 (즉, 인트라 예측 및 단방향 인터 예측이 가능하게 된 슬라이스들) 을 포함하는 경우, 비디오 인코더 (20) 는 현재 화상의 RefPicStCurrAfter, RefPicStCurrBefore, 및 RefPicStLtCurr 참조 화상 서브 세트들로부터의 참조 화상들을 사용하여 현재 화상에 대한 단일 참조 화상 리스트 (RefPicList0) 를 생성할 수도 있다. 현재 화상이 B 슬라이스들 (즉, 인트라 예측, 단방향 인터 예측, 및 양방향 인터 예측이 가능하게 된 슬라이스들) 을 포함하는 경우, 비디오 인코더 (20) 는 현재 화상에 대해 2 개의 참조 화상 리스트들 (RefPicList0 및 RefPicList1) 을 생성하기 위해 현재 화상의 RefPicStCurrAfter, RefPicStCurrBefore, 및 RefPicStLtCurr 참조 화상 서브 세트들로부터의 참조 화상들을 사용할 수도 있다. 비디오 인코더 (20) 는, 현재 화상의 제 1 슬라이스에 대한 슬라이스 헤더에, 비디오 디코더 (30) 가 현재 화상의 참조 화상 서브 세트들을 결정하기 위해 사용할 수도 있는 신택스 엘리먼트들을 포함시킬 수도 있다. 비디오 디코더 (30) 가 현재 화상의 현재 슬라이스를 디코딩하는 경우, 비디오 디코더 (30) 는 현재 화상의 참조 화상 서브 세트들을 결정할 수도 있고 RefPicList0 및/또는 RefPicList1을 재생성할 수도 있다.

[0147] 위에서 나타낸 바와 같이, 비디오 코더가 현재 화상을 코딩하는 것을 시작하는 경우, 비디오 코더는 현재 화상에 대해 제 1 참조 화상 리스트 (즉, RefPicList0) 를 초기화할 수도 있다. 더욱이, 현재 화상이 B 슬라이스들을 포함하면, 비디오 코더는 현재 화상에 대해 제 2 참조 화상 리스트 (즉, RefPicList1) 를 초기화할 수도 있다. 따라서, 일부 예들에서, 현재 뷰 성분/계층 표현이 B 슬라이스 (즉, 양-예측 슬라이스) 를 포함하는 경우에만, 비디오 코더가 RefPicList1을 생성할 수도 있다. 일부 예들에서, 참조 화상 리스트 초기화는 참조 화상들의 POC (Picture Order Count) 값들의 순서에 기초하여 참조 화상 메모리 (즉, 디코딩된 화상 버퍼) 에서의 참조 화상들을 리스트에 넣는 명시적 메커니즘 (explicit mechanism) 이다. 동일한 코딩된 비디오 시퀀스에서의 다른 화상들의 출력 순서 포지션들을 기준으로 출력 순서에서의 연관된 화상의 포지션을 나타내는 각각의 화상과 연관되는 변수가 POC 값이다.

[0148] RefPicList0을 생성하기 위해, 비디오 코더 (예컨대, 비디오 인코더 또는 비디오 디코더) 가 RefPicList0의 초기 디폴트 버전을 생성할 수도 있다. 일부 예들에서는, RefPicList0의 초기 버전에서, RefPicSetStCurrBefore에서의 참조 화상들이 먼저 열거되며, 그 뒤에 RefPicSetStCurrAfter에서의 참조 화상들, 그 뒤에 RefPicSetLtCurr에서의 참조 화상들이 열거된다. 마찬가지로, RefPicList1을 생성하기 위해, 비디오 코더는 RefPicList1의 초기 버전을 생성할 수도 있다. 일부 예들에서는, RefPicList1의 초기 버전에서, RefPicSetStCurrAfter에서의 참조 화상들이 먼저 열거되며, 그 뒤에 RefPictSetStCurrBefore에서의 참조 화상들, 그 뒤에 RefPicSetLtCurr에서의 참조 화상들이 열거된다.

[0149] 일부 예들에서, 비디오 코더가 최종 참조 화상 리스트들 (즉, RefPicList0 및 RefPicList1) 을 구축한 후, 비디오 코더는 B 슬라이스에 대해 결합된 리스트 (예컨대, RefPicListC) 를 구축한다. 비디오 코더는 하나 이상의 참조 화상 리스트 수정 신택스 엘리먼트들이 결합된 리스트에 대해 존재하면 결합된 리스트를 추가로 또한 수정할 수도 있다.

[0150] 비디오 코더가 참조 화상 리스트 (예컨대, RefPicList0 또는 RefPicList1) 를 초기화한 후, 비디오 코더는 참조

화상 리스트에서의 참조 화상들의 순서를 수정할 수도 있다. 다르게 말하면, 비디오 코더는 참조 화상 리스트 수정 (reference picture list modification; RPLM) 프로세스를 수행할 수도 있다. 비디오 코더는 하나의 특정 참조 화상이 참조 화상 리스트에서의 하나를 초과하는 포지션에서 보여질 수도 있는 경우를 포함하여, 임의의 순서로 참조 화상들의 순서를 수정할 수도 있다. 따라서, 참조 화상 리스트 재순서화 메커니즘은, 참조 화상 리스트 초기화 동안에 리스트에 넣었던 화상의 포지션을 임의의 새로운 포지션으로 수정하거나, 또는 심지어 임의의 참조 화상이 초기화된 리스트에 속하지 않더라도 임의의 참조 화상을 참조 화상 메모리에서 임의의 포지션에 넣을 수 있다. 그러나, 화상의 포지션이 리스트의 액티브 참조 화상들의 수를 초과하면, 그 화상은 최종 참조 화상 리스트의 엔트리로서 간주되지 않는다. 슬라이스 헤더가 참조 화상 리스트들에서의 액티브 참조 화상들의 수를 나타내는 하나 이상의 신택스 엘리먼트들을 포함할 수도 있다.

[0151] RPLM 프로세스를 구현하기 위해, 슬라이스 헤더가 RPLM 신택스 구조 (예컨대, ref_pic_list_modification ()) 를 포함할 수도 있다. 아래의 표 1은 HEVC 규격 초안 8에서 제시된 RPLM 신택스 구조를 보여준다.

표 1

[0152]

ref_pic_list_modification( ) {	기술자
ref_pic_list_modification_flag_l0	u(1)
if(ref_pic_list_modification_flag_l0 && NumPocTotalCurr > 1)	
for(i = 0; i <= num_ref_idx_l0_active_minus1; i++)	
list_entry_l0[i]	u(v)
if(slice_type == B) {	
ref_pic_list_modification_flag_l1	u(1)
if(ref_pic_list_modification_flag_l1 && NumPocTotalCurr > 1)	
for(i = 0; i <= num_ref_idx_l1_active_minus1; i++)	
list_entry_l1[i]	u(v)
}	
}	

[0153] 위의 표 1과 본 개시물의 다른 신택스 표들의 예에서, n이 음이 아닌 정수인 형태 u(n) 의 기술자들을 갖는 신택스 엘리먼트들은, 길이 n의 부호없는 값들이다. 더욱이, 형태 u(v) 의 기술자들을 갖는 신택스 엘리먼트들은 부호없는 가변 길이 값들이다. 더욱이, 표 1에 관해, 변수 NumPocTotalCurr는 NumPocStCurrBefore + NumPocStCurrAfter + NumPocLtCurr와 동일하게 설정된다. NumPocStCurrBefore는 RefPicSetStBefore에서의 엘리먼트들의 수를 나타낸다. NumPocStCurrAfter는 RefPicSetStAfter에서의 엘리먼트들의 수를 나타낸다. NumPocLtCurr는 RefPicSetLtCurr에서의 엘리먼트들의 수를 나타낸다.

[0154] 표 1에서, 1과 동일한 ref_pic_list_modification_flag_l0 신택스 엘리먼트는 RefPicList0이 list_entry_l0[i] 값들의 리스트로서 명시적으로 특정됨을 나타낸다. 0과 동일한 ref_pic_list_modification_flag_l0 신택스 엘리먼트는 RefPicList0이 암시적으로 결정됨을 나타낸다. 1과 동일한 ref_pic_list_modification_flag_l1 신택스 엘리먼트는 RefPicList1이 list_entry_l1[i] 값들의 리스트로서 명시적으로 특정됨을 나타낸다. 0과 동일한 ref_pic_list_modification_flag_l1 신택스 엘리먼트는 RefPicList1이 암시적으로 결정됨을 나타낸다.

[0155] 더욱이, 표 1의 예에서, list_entry_lX[i] 신택스 엘리먼트 (X는 0 또는 1과 동일함) 는 참조 화상 리스트 LX (X는 0 또는 1과 동일함) 의 현재 포지션에 배치될 RefPicSetCurrTempListX에서의 참조 화상의 인덱스를 특정한다. RefPicSetCurrTempListX (이것은 RefPicListTempX라고 또한 지칭됨) 가 RefPicListX의 초기 버전이다. X의 값은 list_entry_lX, RefPicSetCurrTempListX, 및 LX의 각각에 대해 동일하다. 이 예에서, list_entry_lX[i] 신택스 엘리먼트의 길이는 Ceil(Log2(NumPocTotalCurr)) 비트이다. 더욱이, 이 예에서, list_entry_lX[i]의 값은 0 내지 NumPocTotalCurr - 1의 범위에 있다. list_entry_lX[i] 신택스 엘리먼트가 존재하지 않는다면, list_entry_lX[i] 신택스 엘리먼트는 0과 동일한 것으로 유추될 수도 있다.

[0156] 위에서 나타낸 바와 같이, 비디오 코더가 P 또는 B 슬라이스를 코딩하는 것을 시작하는 경우, 비디오 코더는 RefPicList0의 초기 버전을 생성할 수도 있다. RefPicList0의 초기 버전은 RefPicListTemp0으로 표시될 수도 있다. HEVC 규격 초안 8에서, 비디오 코더는 RefPicList0의 초기 버전을 생성하기 위해 다음의 의사-코드에 의해 설명된 동작을 사용할 수도 있다.

```
rIdx = 0
while( rIdx < NumRpsCurrTempList1 ) {
   for( i = 0; i < NumPocStCurrAfter && rIdx < NumRpsCurrTempList1; rIdx++,
i++ )
      RefPicListTemp1[ rIdx ] = RefPicSetStCurrAfter[ i ]
   for( i = 0;  i < NumPocStCurrBefore && rIdx < NumRpsCurrTempList1;
rIdx++, i++ )
      RefPicListTemp1[ rIdx ] = RefPicSetStCurrBefore[ i ]
   for( i = 0; i < NumPocLtCurr && rIdx < NumRpsCurrTempList1; rIdx++, i++
)
      RefPicListTemp1[ rIdx ] = RefPicSetLtCurr[ i ]
}
```

[0157]

[0158] 위의 의사-코드에서, 변수 NumRpsCurrTempList0은 Max(num_ref_idx_l0_active_minus1 + 1, NumPocTotalCurr)와 동일하게 설정된다. 변수 num_ref_idx_l0_active_minus1은 RefPicList0에서의 액티브 참조 화상들의 수 빼기 1을 나타낸다.

[0159] 더욱이, HEVC 규격 초안 8에서, RefPicList0은 다음과 같이 구축된다:

```
for( rIdx = 0; rIdx ≤ num_ref_idx_l0_active_minus1; rIdx++)
   RefPicList0[ rIdx ] = ref_pic_list_modification_flag_l0 ?
      RefPicListTemp0[ list_entry_l0[ rIdx ] ] :
RefPicListTemp0[ rIdx ]
```

[0160]

[0161] 위의 의사-코드에서, 참조 화상 리스트 수정이 RefPicList0에 대해 가능하게 되면, 비디오 코더는, RefPicList0에서의 각각의 개별 포지션에 대해, RefPicList0에서의 개별 포지션에 대응하는 list_entry_l0 신택스 엘리먼트를 결정할 수도 있다. 비디오 코더는, RefPicList0에서의 개별 포지션에, 결정된 list_entry_l0 신택스 엘리먼트에 의해 나타내어진 RefPicListTemp0에서의 포지션의 참조 화상을 삽입할 수도 있다.

[0162] 더욱이, 비디오 코더가 B 슬라이스를 코딩하는 것을 시작하는 경우 비디오 코더는 RefPicList1의 초기 버전을 생성할 수도 있다. RefPicList1의 초기 버전은 RefPicListTemp1로 표시될 수도 있다. HEVC 규격 초안 8에서, 비디오 코더는 RefPicList1의 초기 버전을 생성하기 위해 다음의 의사-코드에 의해 설명된 동작을 사용할 수도 있다.

```
rIdx = 0
while( rIdx < NumRpsCurrTempList1 ) {
   for( i = 0; i < NumPocStCurrAfter && rIdx < NumRpsCurrTempList1; rIdx++,
i++ )
      RefPicListTemp1[ rIdx ] = RefPicSetStCurrAfter[ i ]
   for( i = 0;  i < NumPocStCurrBefore && rIdx < NumRpsCurrTempList1;
rIdx++, i++ )
      RefPicListTemp1[ rIdx ] = RefPicSetStCurrBefore[ i ]
   for( i = 0; i < NumPocLtCurr && rIdx < NumRpsCurrTempList1; rIdx++, i++
)
      RefPicListTemp1[ rIdx ] = RefPicSetLtCurr[ i ]
}
```

[0163]

[0164] 위의 의사-코드에서, 변수 NumRpsCurrTempList1은 Max(num_ref_idx_l1_active_minus1 + 1, NumPocTotalCurr)와 동일하게 설정된다. 변수 num_ref_idx_l1_active_minus1은 RefPicList1에서의 액티브 참조 화상들의 수 빼기 1을 나타낸다.

[0165] 더욱이, HEVC 규격 초안 8에서, RefPicList1은 다음과 같이 구축된다:

```
for( rIdx = 0; rIdx ≤ num_ref_idx_l1_active_minus1; rIdx++)
      RefPicList1[ rIdx ] = ref_pic_list_modification_flag_l1 ?
            RefPicListTemp1[ list_entry_l1[ rIdx ] ] : RefPicListTemp1[ rIdx ]
```

[0166]

[0167] 위의 의사-코드에서, 참조 화상 리스트 수정이 RefPicList1에 대해 가능하게 되면, 비디오 코더는, RefPicList1에서의 각각의 개별 포지션에 대해, RefPicList1에서의 개별 포지션에 대응하는 list_entry_l1 신택스 엘리먼트를 결정할 수도 있다. 비디오 코더는, RefPicList1에서의 개별 포지션에, 결정된 list_entry_l1 신택스 엘리먼트에 의해 나타내어진 RefPicListTemp1에서의 포지션의 참조 화상을 삽입할 수도 있다.

[0168] 특정 비디오 코딩 표준들 (예컨대, HEVC) 의 일부 코덱 확장본들에서, 참조 화상 리스트가 정상적인 시간적 참조 화상들뿐만 아니라, 현재 뷰/계층과는 다른 뷰들/계층들로부터의 참조 화상들, 또는 그 뷰들/계층들로부터의 화상들로부터 생성된 참조 화상들을 포함할 수도 있다. 다른 뷰들/계층들로부터의 참조 화상들, 또는 그 뷰들/계층들로부터의 화상들로부터 생성된 참조 화상들은 "뷰/계층 간 참조 화상들"이라 명명된다. 다른 뷰들/계층들로부터의 화상들로부터 참조 화상들을 생성하는 프로세스는 뷰 합성 예측이라고 지칭될 수도 있고 이런 식으로 생성된 화상들은 뷰 합성 화상들이라고 지칭될 수도 있다.

[0169] HEVC에서, 참조 화상 리스트들 (즉, RefPicList0 및 RefPicList1) 양쪽 모두는 통상적으로 시간적 참조 화상들을 포함한다. 그러나, HEVC의 일부 확장본들 (예컨대, 3D-HEVC) 에서, 뷰/계층 간 참조 화상들은 RefPicList0에는 통상 한 번만 존재하고 RefPicList1에는 통상 한번도 존재하지 않는다. 더욱이, 3D-HEVC에서, 참조 화상 리스트는 뷰/계층 간 참조 화상들을 포함할 수도 있고 VPS 확장본에서의 ref_layer_id[i][j] 신택스 엘리먼트들은, 아래의 표 2에서 도시된 바와 같이, 비디오 코더가 특정 뷰/계층을 예측하는데 사용할 수 있는 뷰들/계층들을 시그널링할 수도 있다.

표 2

vps_extension( ) {	기술자
while(!byte_aligned( ))	
…	
view_id[i]	u(8)
if (i > 0)	
num_direct_ref_layers[i]	u(6)
for(j = 0; j < num_direct_ref_layers[i]; j++)	
ref_layer_id[i][j]	u(6)
}	
…	
}	

[0170]

[0171] 표 2의 예에서, 비디오 코더는 동일한 액세스 유닛에서 ref_layer_id[i][j]에 의해 식별된 각각의 뷰/계층 간 참조 화상을 이른바 뷰/계층 간 RPS 에 삽입한다. 비디오 코더는 뷰/계층 간 RPS에서의 참조 화상들을 RefPicList0 및/또는 RefPicList1을 초기화하기 위해 사용할 수도 있다. RefPicList0 및/또는 RefPicList1을 초기화한 후, 비디오 코더는 RefPicList0 및/또는 RefPicList1에서 참조 화상들의 순서를 수정할 수도 있다.

다르게 말하면, 뷰/계층 간 RPS는 RefPicList0 및 RefPicList1 양쪽 모두에 대한 참조 화상 리스트 초기화 및 수정을 위해 추가로 고려될 수도 있다.

[0172] 『Ramasubramonian et al., "AHG7: Reference picture list initialization for MV-HEVC", Joint Collaborative Team on 3D Video Coding Extension Development of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11, 3rd meeting, Geneva, CH, 17-23 Jan. 2013』, 문서 JCT3V-C0060 (이후로는, JCT3V-C0060) 이 뷰 간 참조 화상들을 참조 화상 리스트의 주어진 자리 속으로 연속하여 넣는 참조 화상 리스트 초기화 방법을 제안한다. 구체적으로는, JCT3V-C0060은 뷰 간 참조 화상들의 소망의 초기 포지션이 슬라이스 헤더에서 시그널링되는 MV-HEVC에 대한 참조 화상 리스트들의 초기화의 방법을 제안한다. 더욱이, JCT3V-C0060에서, 참조 화상 리스트들의 초기화 프로세스는 뷰 간 참조 화상들이 초기 참조 화상 리스트에서의 시그널링된 시작 포지션에 존재하도록 수정된다.

[0173] 아래의 표 3은 JCT3V-C0060에서 정의된 바와 같은 슬라이스 헤더에 대한 신택스의 일 예의 부분이다. 표 3에서 밑줄 쳐진 텍스트는 MV-HEVC 규격 초안 2에서 정의된 슬라이스 헤더 신택스에 추가된 텍스트를 나타낸다.

표 3

slice_header( ) {	기술자
first_slice_in_pic_flag	u(1)
…	
if(slice_header_extension_present_flag) { // should always be true in MV-HEVC	
slice_header_extension_length	ue(v)
if(slice_type != I_SLICE)	
inter_view_ref_start_position_plus1	ue(v)
…	
}	
byte_alignment( )	
}	

[0174]

[0175] 위의 표 3의 예 및 본 개시물의 다른 신택스 표들에서, 유형 기술자 ue(v) 를 갖는 신택스 엘리먼트들은, 좌측 비트를 첫 째로 하는 0차 지수 골롬 (Exp-Golomb) 코딩을 사용하여 인코딩된 가변 길이 부호없는 정수들일 수도 있다. 표 3에 나타낸 바와 같이, 슬라이스가 I 슬라이스 (즉, 인트라 코딩된 슬라이스) 가 아니면, 그 슬라이스의 슬라이스 헤더가 inter_view_ref_start_position_plus1 신택스 엘리먼트를 포함할 수도 있다.

[0176] 더욱이, JCT3V-C0060은 참조 화상 리스트 초기화 후 inter_view_ref_start_position_plus1 신택스 엘리먼트가 참조 화상 리스트 0에서의 뷰 간 참조 화상들의 시작 포지션을 특정함을 나타낸다. inter_view_ref_start_position_plus1 신택스 엘리먼트는 0 내지 min(num_ref_idx_l0_active_minus1 + 1, NumPocStCurrBefore + NumPocStCurrAfter + NumPocLtCurr) 의 범위에 있다. inter_view_ref_start_position_plus1 신택스 엘리먼트가 0과 동일한 경우, 뷰 간 참조 화상들은 참조 화상 리스트에서의 디폴트 포지션에 존재한다. 다른 0이 아닌 값들에 대해, inter_view_ref_start_position_plus1 신택스 엘리먼트 - 1은 초기 참조 화상 리스트에서의 뷰 간 참조 화상들의 시작 포지션을 나타낸다. 존재하지 않는 경우, inter_view_ref_start_position_plus1 신택스 엘리먼트는 NumPocStCurrBefore + NumPocStCurrAfter + NumPocLtCurr + 1과 동일한 디폴트 값으로 유추될 수도 있다.

[0177] JCT3V-C0060은 변수 IvRefStartPos가 다음과 같이 도출됨을 나타낸다.

```
if( !inter_view_ref_start_position_plus1 )
        IvRefStartPos = NumPocStCurrBefore + NumPocStCurrAfter +
NumPocLtCurr
else
        IvRefStartPos = inter_view_ref_start_position_plus1 - 1
```

[0178]

[0179] 더욱이, JCT3V-C0060은 MV-HEVC 규격 초안 2의 디코딩 프로세스에 대해 다음의 변경들을 제안한다. 다음에서, 밑줄 처진 텍스트는 MV-HEVC 규격 초안 2에 추가되고 2 개로 된 꺽쇠 괄호들 내의 이탤릭체 텍스트는 MV-HEVC 규격 초안 2로부터 제거된다.

[0180] 변수 NumRpsCurrTempList0은 Max(num_ref_idx_l0_active_minus1 + 1, NumPocTotalCurr) 과 동일하게 설정되고 리스트 RefPicListTemp0은 다음과 같이 구축된다:

```
cIdx = 0
while( cIdx < NumRpsCurrTempList0 ) {
  for( i = 0; i < NumPocStCurrBefore && cIdx < NumRpsCurrTempList0;
cIdx++, i++ )
    RefPicListTemp0[ cIdx ] = RefPicSetStCurrBefore[ i ]
  for( i = 0;  i < NumPocStCurrAfter && cIdx < NumRpsCurrTempList0;
cIdx++, i++ )  (F-25)
    RefPicListTemp0[ cIdx ] = RefPicSetStCurrAfter[ i ]
  for( i = 0; i < NumPocLtCurr && cIdx < NumRpsCurrTempList0; cIdx++, i++
)
    RefPicListTemp0[ cIdx ] = RefPicSetLtCurr[ i ]
[[  for( i = 0; i < NumIvCurr && rIdx < NumRpsCurrTempList0; rIdx++, i++
 )
    RefPicListTemp0[ rIdx ] = RefPicSetIvCurr[ i ]]]
 }
```

[0181]

[0182] // 필요하다면, 뷰 간 참조 화상들에 대한 공간을 만들기 위해 리스트에서 화상들을 시프트

```
for( cIdx = NumRpsCurrTempList0 - 1; cIdx >= IvRefStartPos +
NumPocIvCurr; cIdx -- )
    RefPicListTemp0[ cIdx ] = RefPicListTemp0[ cIdx - NumPocIvCurr ]
```

[0183]

[0184] // 뷰 간 참조 화상들을 포함

```
for( i=0, cIdx = IvRefStartPos; i < NumPocIvCurr; cIdx++, i++ )
    RefPicListTemp0[ cIdx ] = RefPicSetIvCurr[ i ]
```

[0185]

[0186] 리스트 RefPicList0은 다음과 같이 구축된다:

```
for( cIdx = 0; cIdX ≤ num_ref_idx_l0_active_minus1; cIdx++)
    (F-26)
    RefPicList0[cIdx] = ref_pic_list_modification_flag_l0 ?
    RefPicListTemp0[list_entry_l0[cIdx] ] : RefPicListTemp0[ cIdx ]
```

[0187]

[0188] JCT3V-C0060은 RefPicList1의 도출이 MV-HEVC 규격 초안 2에서 정의된 바와 동일하다는 것을 나타낸다. 이에 따라, JCT3V-C0060의 방법은 RefPicList1에 대한 복잡도를 줄이는데 도움이 되지 않을 수도 있지만 RefPicList0에 대한 복잡도를 줄이는데 만큼은 도움이 될 수도 있다.

[0189] 일부 HEVC 확장본들, 이를테면 MV-HEVC, 3D-HEVC 및 SHVC에서의 일부 도구들은, 뷰/계층 간 참조 화상들이 RefPicList0 및 RefPicList1 양쪽 모두에 존재한다고 가정한다. 이는 여러 문제들로 이어질 수도 있다. 예를 들어, 뷰 간 예측이 RefPicList1에 존재할 수도 있다는 가정에 관련된 정보에 액세스하는 것을 필요로 하는 모든 낮은 레벨 코딩 도구들은, 중복 (redundant) 체크들을 수행할 필요가 있으며, 이는 더 많은 동작들을 야기할 수도 있고 더 많은 데이터가 메모리에 저장되는 것을 또한 요구할 수도 있다. 예를 들면, 낮은 레벨 코딩 도구들 (예컨대, PU 레벨 또는 CU 레벨에 적용되는 코딩 도구들) 은, RefPicList1이 임의의 뷰/계층 간 참조 화상들을 포함하는지의 여부에 상관 없이, RefPicList1에서의 참조 화상들이 뷰/계층 간 참조 화상들인지의 여부를 계속 체크할 수도 있다.

[0190] 다른 예에서, 참조 화상 리스트 구축 프로세스들은 (참조 화상 리스트 초기화와 참조 화상 리스트 수정을 포함하는) 참조 화상 리스트 구축에 관련된 프로세스들이다. 이 예에서, 참조 화상 리스트 구축 프로세스들은 RefPicList1이 뷰/계층 간 참조 화상들을 포함한다고 가정할 수도 있다. 따라서, 이 예에서, 참조 화상 리스트 구축 프로세스들은 더 많은 프로세스들과 RefPicList0 및 RefPicList1에 존재하는 중복 신택스를 필요로 할 수도 있는 반면, 이러한 프로세스들과 신택스는 RefPicList1에 대해 쓸모 없을 수도 있다.

[0191] 뷰/계층 간 참조 화상들이 RefPicList0 및 RefPicList1 양쪽 모두에 존재한다고 특정한 코딩 도구들이 가정하기 때문에 일어날 수도 있는 다른 예의 문제에서, 참조 화상 리스트 구축에 관련된 신택스 표들에 존재하는 신택스 엘리먼트들 (예컨대, list_entry_l0 신택스 엘리먼트들과 list_entry_l1 신택스 엘리먼트들) 은 u(v) 코딩될 수도 있다. 그 결과, RefPicList1에 대응하는 신택스 엘리먼트들은 불필요하게 길 수도 있다. 다른 예에서, 동일한 뷰/계층 간 참조 화상을 RefPicList0 및 RefPicList1 양쪽 모두에 삽입하는 것은 임의의 코딩 이득을 초래하지 않을 수도 있다.

[0192] 본 개시물의 예들은 높은 레벨 신택스 구조들에서 뷰/계층 간 예측을 불가능하게 하도록 구성된 멀티-뷰/3D 또는 스케일러블 코덱을 포함한다. 높은 레벨 신택스 구조들은 비-VCL NAL 유닛들에 신택스 구조들을 포함시킬 수도 있다. 예를 들면, 높은 레벨 신택스 구조들은 SEI 메시지들, 파라미터 세트들 (예컨대, VPS들, SPS들, PPS들 등) 등을 포함할 수도 있다.

[0193] 본 개시물의 하나의 예에서, 비디오 인코더 (20) 는 VPS 확장본에서 신택스 엘리먼트 (예컨대, inter_view_l1_disable_flag) 를 시그널링한다. 설명의 편의를 위해, 본 개시물은 이 신택스 엘리먼트를 inter_view_l1_disable_flag라고 지칭할 수도 있더라도, 이 신택스 엘리먼트는 적용가능한 코덱에서 상이한 이

름을 가질 수도 있다. inter_view_l1_disable_flag는 VPS를 참조하는 CVS의 임의의 뷰 성분/계층 표현에 대한 RefPicList1에 뷰/계층 간 참조 화상들이 한번이라도 포함되는지의 여부를 나타낸다.

[0194] 다른 예에서, 비디오 인코더 (20) 는 VPS를 참조하는 CVS의 특정 뷰/계층의 임의의 뷰 성분/계층 표현에 대한 RefPicList1에 뷰/계층 간 참조 화상들이 한번이라도 포함되는지의 여부를 나타내기 위하여 각각의 계층에 대한 inter_view_l1_disable_flag를 시그널링할 수도 있다. 이 예에서, 다수의 이러한 신택스 엘리먼트들 (예컨대, 플래그들) 은 VPS에서 시그널링될 수도 있으며, 그 각각은 하나의 특정 계층/뷰에 부속된다. 예를 들면, 이 예에서, 비디오 인코더 (20) 는 상이한 계층들에서의 상이한 CVS들에 대해 별개의 SPS들을 시그널링할 수도 있다. 각각의 SPS는 SPS를 참조하는 화상들에 대한 RefPicList1들이 뷰 간 참조 화상들/계층 표현들을 한번이라도 포함하는지의 여부를 나타내는 inter_view_l1_disable 플래그를 포함할 수도 있다. 따라서, 이 예에서, 비디오 디코더 (30) 는, SPS들로부터, 복수의 계층들에서의 각각의 개별 계층에 대해, 개별 계층에서의 뷰 성분들/계층 표현들의 개별 참조 화상 리스트들에 뷰/계층 간 참조 화상들이 한번이라도 포함되는지의 여부를 나타내는, 개별 계층에 대한 개별 신택스 엘리먼트를 나타내는 신택스 엘리먼트들을 획득할 수도 있다.

[0195] 설명의 편의를 위해, 본 개시물은 inter_view_l1_disable_flag가 적용가능한 RefPicList1들이 뷰/계층 간 참조 화상들을 한번도 포함하지 않음을 1의 값을 갖는 inter_view_l1_disable_flag가 나타낸다고 가정한다. 더욱이, 설명의 편의를 위해, 본 개시물은 inter_view_l1_disable_Flag가 적용가능한 RefPicList1들이 뷰/계층 간 참조 화상들을 포함할 수도 있다는 것을 0의 값을 갖는 inter_view_l1_disable_flag가 나타낸다고 가정한다. 그러나, 다른 예들에서, 0의 값을 갖는 inter_view_l1_disable_flag는 inter_view_l1_disable_flag가 적용가능한 RefPicList1들이 뷰/계층 간 참조 화상들을 한번도 포함하지 않는다는 것을 나타내고 1의 값을 갖는 inter_view_l1_disable_flag는 inter_view_l1_disable_flag가 적용가능한 RefPicList1들이 뷰/계층 간 참조 화상들을 포함할 수도 있다는 것을 나타낸다.

[0196] inter_view_l1_disable_flag가 1인 경우, 비디오 디코더 (30) 에 의해 수행된 비디오 디코딩 프로세스는 다양한 방도들로 단순화될 수도 있다. 이러한 단순화들은 부분적으로는 디코딩 프로세스에서 수행되는 동작들의 수를 감소시키고 및/또는 디코딩 프로세스 동안 수행된 메모리 액세스의 수를 감소시킴으로써 디코딩 프로세스를 가속화할 수도 있다.

[0197] inter_view_l1_disable_flag가 1인 경우 디코딩 프로세스가 단순화될 수도 있는 방법의 하나의 예에서, 비디오 디코더 (30) 가 현재 블록에 대한 디스패리티 벡터를 결정하기 위해 NBDV 도출 프로세스를 수행하고 inter_view_l1_disable_flag가 1일 때, 비디오 디코더 (30) 는 이웃 블록들의 RefPicList1 모션 정보를 체크하지 않는다. 다르게 말하면, 3D-HEVC의 디스패리티 벡터 도출 (NBDV) 에서, RefPicList1에 대응하는 모션 정보 (예컨대, RefPicList1 모션 벡터, RefPicList1 참조 인덱스 등) 는 한번도 체크되지 않고, 따라서 복잡도는 이 구성에서 2배만큼 감소될 수 있다. 더욱이, 이 예 또는 다른 예들에서, 비디오 디코더 (30) 가 NBDV 도출 프로세스를 수행하고 inter_view_l1_disable_flag가 1인 경우, 이웃 블록이 RefPicList0 디스패리티 모션 벡터 또는 RefPicList0 IDV를 갖지 않을 때, 비디오 디코더 (30) 는 이웃 블록에 대한 IDV를 저장하지 않는다. 따라서, 비디오 디코더 (30) 는 각각의 이웃 블록에 대해 많아야 하나의 디스패리티 벡터를 저장할 수도 있다. 다르게 말하면, NBDV에서, IDV 후보들은 각각의 블록에 대해 하나의 디스패리티 모션 벡터만을 저장한다. 이런 식으로, 비디오 디코더 (30) 는 현재 블록에 대한 디스패리티 벡터를 결정하기 위하여 현재 뷰 성분/계층 표현의 현재 블록에 이웃하는 하나 이상의 블록들을 체크하는 디스패리티 벡터 도출 프로세스를 수행할 수도 있다. 디스패리티 벡터 도출 프로세스를 수행하는 부분으로서, 비디오 디코더 (30) 는, 현재 뷰 성분/계층 표현에 대한 참조 화상 리스트 (예컨대, RefPicList1) 에 뷰/계층 간 참조 화상들이 한번도 포함되지 않음을 신택스 엘리먼트 (예컨대, inter_view_l1_disable_flag) 가 나타내는 경우, 현재 뷰 성분/계층 표현에 대한 참조 화상 리스트에 대응하는 모션 정보를 체크하지 않을 수도 있다. 더구나, 일부 예들에서, 현재 뷰 성분/계층 표현에 대한 참조 화상 리스트 (예컨대, RefPicList1) 에 뷰/계층 간 참조 화상들이 한번도 포함되지 않음을 신택스 엘리먼트 (예컨대, inter_view_l1_disable_flag) 가 나타내는 경우, 현재 블록에 이웃하는 하나 이상의 블록들의 각각에 대해 많아야 하나의 암시적 디스패리티 벡터를 저장한다.

[0198] inter_view_l1_disable_flag가 1인 경우 비디오 디코더 (30) 에 의해 수행된 비디오 디코딩 프로세스가 단순화될 수도 있는 방법의 다른 예에서, 비디오 디코더 (30) 는, 병합 후보 리스트 또는 AMVP 후보 리스트에, 뷰/계층 간 참조 화상들에 대응하는 모션 후보들을 포함시키지 않는다. 예를 들면, 3D-HEVC에서, 병합 또는 AMVP 리스트는 잠재적 단순화가 가능하도록 뷰/계층 간 참조 화상에 대응하는 모션 후보를 추가하는 것을 한번도 요구하지 않는다. 뷰/계층 간 참조 화상들에 대응하는 모션 후보들은 뷰/계층 간 참조 화상들에서 참조 로케

이션들을 특정할 수도 있다.

[0199] 이 예에서, 현재 뷰 성분/계층 표현에 대한 참조 화상 리스트 (예컨대, RefPicList1) 에 뷰/계층 간 참조 화상들이 한번도 포함되지 않음을 신택스 엘리먼트 (예컨대, inter_view_l1_disable_flag) 가 나타내는 경우, 비디오 디코더 (30) 는, 후보 리스트에, 뷰/계층 간 참조 화상에 대응하는 후보를 한번도 포함시키지 않을 수도 있다. 더욱이, 현재 뷰 성분/계층 표현에 대한 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번도 포함되지 않음을 신택스 엘리먼트가 나타내는 경우, 비디오 디코더 (30) 는, 후보 리스트에서의 특정 후보에 기초하여, 현재 뷰 성분/계층 표현의 현재 블록에 대한 모션 벡터를 결정할 수도 있다.

[0200] inter_view_l1_disable_flag가 1인 경우 비디오 디코더 (30) 에 의해 수행되는 비디오 디코딩 프로세스가 단순화될 수도 있는 방법의 다른 예에서, 비디오 디코더 (30) 는 RefPicList1에서의 참조 화상이 뷰/계층 간 참조 화상인지의 여부를 체크하는 것을 피할 수도 있다. 현재 3D-HEVC에서, 모션 예측 또는 디스패리티 벡터 도출 동안, 비디오 코더는 RefPicList0에서의 참조 화상이 뷰/계층 간 참조 화상인지의 여부를 체크할 수도 있고 RefPicList1에서의 참조 화상이 뷰/계층 간 참조 화상인지의 여부를 체크할 수도 있다. 다르게 말하면, 참조 화상 리스트로부터의 참조 화상이 뷰/계층 간 참조 화상인지 아닌지의 여부의 체킹은 그 리스트가 RefPicList1임을 알고 있을 경우 회피될 수도 있다. 따라서, 이 예에서, 현재 뷰 성분/계층 표현에 대한 참조 화상 리스트 (RefPicList1) 에 뷰/계층 간 참조 화상들이 한번도 포함되지 않음을 신택스 엘리먼트 (예컨대, inter_view_l1_disable_flag) 가 나타내는 경우, 비디오 디코더 (30) 는 참조 화상 리스트로부터의 참조 화상이 뷰/계층 간 참조 화상인지의 여부의 체킹을 피할 수도 있다.

[0201] inter_view_l1_disable_flag가 1인 경우 비디오 디코더 (30) 에 의해 수행된 비디오 코딩 프로세스가 단순화될 수도 있는 방법의 다른 예에서, CU에 대한 뷰 간 잔차 예측 플래그 (예컨대, res_pred_flag) 를 획득할지의 여부를 결정하는 프로세스는 단순화될 수도 있다. 예를 들면, 3D-HEVC 규격 초안 2에서, 비디오 디코더 (30) 는, resPredEnableFlag 변수가 1과 동일한 경우, 비트스트림으로부터, 현재 CU에 대한 뷰 간 잔차 예측 플래그 (예컨대, res_pred_flag) 를 획득할 수도 있다. 뷰 간 잔차 예측이 가능하게 됨을 VPS에서의 신택스 엘리먼트가 나타내며, 현재 CU에 대한 잔차가 0이 아님을 코딩된 블록 플래그가 나타내고, anyTempRefPicFlag 변수가 1과 동일한 경우, resPredEnableFlag는 1과 동일하다. 현재 CU의 하나 이상의 PU들이 시간적 참조 화상을 이용하는 경우 anyTempRefPicFlag는 1과 동일하다. 3D-HEVC 테스트 모델 설명 초안 2에서, 비디오 디코더 (30) 는 초기에 anyTempRefPicFlag를 0으로 설정할 수도 있다. 현재 CU의 예측 모드가 인트라 예측이 아닌 경우, 다음이 0 및 1에 의해 대체된 X와 1 - X과 동일한 Y에 적용된다.

```
anyTempRefPicFlag = anyTempRefPicFlag ||
(inter_pred_idc[ x0 ][ y0 ] != Pred_LY && refViewIdxLX[ x0 ][ y0 ] == ViewIdx) ||
(inter_pred_idc[ x0 ][ y1 ] != Pred_LY && refViewIdxLX[ x0 ][ y1 ] == ViewIdx) ||
(inter_pred_idc[ x1 ][ y0 ] != Pred_LY && refViewIdxLX[ x1 ][ y0 ] == ViewIdx) ||
(inter_pred_idc[ x1 ][ y1 ] != Pred_LY && refViewIdxLX[ x1 ][ y1 ] == ViewIdx)
```

[0202]

[0203] 위의 수식에서, Pred_LY는 현재 CU의 PU가 RefPicListY로부터 예측이 가능하게 됨을 나타내고, refViewIdxLX는 현재 CU의 PU의 RefPicListX 참조 화상의 뷰 인덱스를 나타내고, ViewIdx는 현재 화상의 뷰 인덱스를 나타낸다.

[0204] inter_view_l1_disable_flag가 1과 동일하고 현재 CU의 적어도 하나의 PU가 RefPicList1 모션 벡터를 갖는 경우, RefPicList1 모션 벡터는 시간적 모션 벡터이다. 따라서, inter_view_l1_disable_flag가 1과 동일하고 현재 CU의 적어도 하나의 PU가 RefPicList1 모션 벡터를 갖는다면, anyTempRefPicFlag의 값을 결정하는 것이 불필요할 수도 있는데, 현재 CU가 인트라 모드에서 코딩되지 않으면 현재 CU의 PU들 중 적어도 하나의 PU가 시간적 참조 화상을 사용하여 코딩되기 때문이다. 따라서, inter_view_l1_disable_flag가 1과 동일하고 현재 CU의 적어도 하나의 PU가 RefPicList1 모션 정보를 사용하여 코딩되는 (즉, Pred_L1 또는 Bi_Pred 모드에서 코딩되는) 경우, 비디오 디코더 (30) 는 anyTempRefPicList의 값을 결정하는 일 없이 resPredEnableFlag의 값을 결정하는 것이 가능할 수도 있다. 따라서, 3D-HEVC에서, 참조 화상 유형들을 체크하는 일 없이 현재 CU 내의 임의의 PU만이 Pred_L1 모드 또는 Bi_Pred를 사용한다면, 뷰 간 잔차 예측 플래그는 시그널링될 수도 있다.

ARP에서, PU가 Pred_L1 또는 Bi_Pred 모드로 코딩되면, 참조 화상 유형이 뷰/계층 간 참조인지 또는 아닌지를 결정하기 위해 참조 화상 유형을 체크하는 일 없이 RefPicList1의 잔차 예측자 생성 프로세스는 항상 가능하게 될 것이다. 따라서, 이 예에서, 현재 뷰 성분/계층 표현에 대한 참조 화상 리스트 (예컨대, RefPicList1) 에 뷰/계층 간 참조 화상들이 한번도 포함되지 않음을 신택스 엘리먼트 (예컨대,

inter_view_l1_disable_flag) 가 나타내는 경우, 특정 참조 화상 내의 로케이션을 나타내는 모션 벡터를 현재 뷰 성분/계층 표현의 현재 CU의 PU가 갖는다면, 비디오 디코더 (30) 는, 참조 화상 리스트에서의 특정 참조 화상의 유형을 체크하는 일 없이, 참조 화상 리스트에 대한 잔차 예측자 생성 프로세스를 가능하게 할 수도 있다.

[0205] inter_view_l1_disable_flag가 1인 경우 비디오 디코더 (30) 에 의해 수행되는 비디오 코딩 프로세스가 단순화될 수도 있는 방법의 다른 예에서, 비디오 디코더 (30) 는 뷰 합성 화상들을 생성하기 위해 RefPicList0에서의 뷰/계층 간 참조 화상들만을 사용할 수도 있다. 더욱이, inter_view_l1_disable_flag가 1인 경우, 비디오 디코더 (30) 는 RefPicList1 내에 뷰 합성 화상들을 포함시키지 않는다. 다시 말하면, 미래의 3D-HEVC로 확장될 수도 있는 뷰 합성 예측은, 뷰 합성을 위해 RefPicList0에 삽입된 뷰 간 참조 화상들만을 사용하거나, 또는 RefPicList1에 추가될 뷰 합성 화상을 한번도 고려하지 않는다. 따라서, 이 예에서, 현재 뷰 성분/계층 표현에 대한 참조 화상 리스트 (예컨대, RefPicList1) 에 뷰/계층 간 참조 화상들이 한번도 포함되지 않음을 신택스 엘리먼트 (예컨대, inter_view_l1_disable_flag) 가 나타내는 경우, 비디오 디코더 (30) 는 상이한 참조 화상 리스트 (예컨대, RefPicList0) 에 삽입된 뷰/계층 간 참조 화상들만을 사용하여 뷰 합성 예측을 수행할 수도 있다.

[0206] 일부 예들에서, inter_view_l1_disable_flag가 1인 경우, 비디오 디코더 (30) 는 단순화된 또는 더욱 효율적으로 설계된 참조 화상 리스트 구축 프로세스를 수행할 수도 있다. 예를 들면, inter_view_l1_disable_flag가 1인 경우, RefPicList1에 대한 초기 참조 화상 리스트를 구축할 때, 비디오 디코더 (30) 는 뷰/계층 간 RPS 또는 뷰/계층 간 참조 화상들을 고려하지 않는다. 더욱이, 본 개시물의 다른 곳에서 설명되는 바와 같이, inter_view_l1_disable_flag가 1인 경우, RPLM 신택스 엘리먼트 list_entry_l0[i]는 RPLM 신택스 엘리먼트 list_entry_l1[i]와는 상이한 방식으로 시그널링될 수도 있다. 따라서, 이 예에서, 현재 뷰 성분/계층 표현에 대한 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번도 포함되지 않음을 신택스 엘리먼트 (예컨대, inter_view_l1_disable_flag) 가 나타내는 경우, 비디오 디코더 (30) 는 참조 화상 리스트의 초기 버전을 구축할 때 뷰/계층 간 참조 화상 세트 또는 뷰/계층 간 참조 화상들을 고려하지 않을 수도 있다.

[0207] 더욱이, 위에서 언급된 본 개시물의 예들 중 적어도 일부는 단일 제어 플래그 (즉, inter_view_l1_disable_flag) 에 의해 관리된다. 그러나, 상이한 곳들 또는 시나리오들에서 RefPicList1에 관한 단순화를 관리할 목적으로, 하나 이상의 별개의 제어 플래그들이 하나 이상의 상이한 곳들 또는 시나리오들을 제어하는데 사용될 수도 있다. 본 개시물의 다른 예들에서, RefPicList1에 대한 참조들은 일부 또는 모든 경우들에서 RefPicList0에 대한 참조로 대체될 수도 있다.

[0208] 도 6은 본 개시물의 기법들을 구현하도록 구성되는 일 예의 비디오 인코더 (20) 를 도시하는 블록도이다. 도 6은 설명의 목적으로 제공되고 본 개시물에서 폭넓게 예시되고 설명된 바와 같은 기법들의 제한으로서 간주되지 않아야 한다. 설명의 목적으로, 본 개시물은 HEVC 코딩의 맥락에서 비디오 인코더 (20) 를 설명한다. 그러나, 본 개시물의 기법들은 다른 코딩 표준들 또는 방법들에 적용가능할 수도 있다.

[0209] 도 6의 예에서, 비디오 인코더 (20) 는 예측 프로세싱 유닛 (100), 잔차 생성 유닛 (102), 변환 프로세싱 유닛 (104), 양자화 유닛 (106), 역 양자화 유닛 (108), 역 변환 프로세싱 유닛 (110), 복원 유닛 (112), 필터 유닛 (114), 디코딩된 화상 버퍼 (116), 및 엔트로피 인코딩 유닛 (118) 을 포함한다. 예측 프로세싱 유닛 (100) 은 인터 예측 프로세싱 유닛 (120) 과 인트라 예측 프로세싱 유닛 (126) 을 구비한다. 인터 예측 프로세싱 유닛 (120) 은 모션 추정 유닛 (122) 과 모션 보상 유닛 (124) 을 구비한다. 다른 예들에서, 비디오 인코더 (20) 는 더 많거나, 더 적거나, 또는 상이한 기능적 컴포넌트들을 포함할 수도 있다.

[0210] 비디오 인코더 (20) 는 비디오 데이터를 수신할 수도 있다. 비디오 인코더 (20) 는 비디오 데이터의 화상의 슬라이스에서의 각각의 CTU를 인코딩할 수도 있다. CTU들의 각각은 화상의 동일 사이즈로 된 루마 코딩 트리 블록 (coding tree block; CTB) 들 및 대응하는 CTB들과 연관될 수도 있다. CTU를 인코딩하는 부분으로서, 예측 프로세싱 유닛 (100) 은 쿼드트리 구획화를 수행하여 CTU의 CTB들을 점차적으로 더 작은 블록들로 분할할 수도 있다. 더 작은 블록들은 CU들의 코딩 블록들일 수도 있다. 예를 들어, 예측 프로세싱 유닛 (100) 은 CTU와 연관된 CTB를 4 개의 동일 사이즈로 된 서브-블록들로 구획화하며, 그 서브-블록들 중 하나 이상을 4 개의 동일 사이즈로 된 서브 서브-블록들로 구획하는 등등을 수행할 수도 있다.

[0211] 비디오 인코더 (20) 는 CTU의 CU들을 인코딩하여 CU들의 인코딩된 표현들 (즉, 코딩된 CU들) 을 생성할 수도 있다. CU를 인코딩하는 부분으로서, 예측 프로세싱 유닛 (100) 은 CU의 하나 이상의 PU들 중에서 CU와 연관된 코딩 블록들을 구획화할 수도 있다. 따라서, 각각의 PU는 루마 예측 블록 및 대응하는 크로마 예측 블록들과 연관될 수도 있다. 비디오 인코더 (20) 와 비디오 디코더 (30) 는 다양한 사이즈들을 갖는 PU들을 지원

할 수도 있다. 위에서 나타낸 바와 같이, CU의 사이즈는 CU의 루마 코딩 블록의 사이즈를 말할 수도 있고 PU의 사이즈는 PU의 루마 예측 블록의 사이즈를 말할 수도 있다. 특정 CU의 사이즈가 2Nx2N이라고 가정하면, 비디오 인코더 (20) 와 비디오 디코더 (30) 는 인트라 예측을 위한 2Nx2N 또는 NxN의 PU 사이즈들과, 인터 예측을 위한 2Nx2N, 2NxN, Nx2N, NxN, 또는 유사한 것의 대칭적 PU 사이즈들을 지원할 수도 있다. 비디오 인코더 (20) 와 비디오 디코더 (30) 는 인터 예측을 위해 2NxnU, 2NxnD, nLx2N, 및 nRx2N의 PU 사이즈들에 대한 비대칭 구획화를 또한 지원할 수도 있다.

[0212] 인터 예측 프로세싱 유닛 (120) 은 CU의 각각의 PU에 대해 인터 예측을 수행함으로써 PU에 대한 예측 데이터를 생성할 수도 있다. PU에 대한 예측 데이터는 PU의 예측 블록들 및 그 PU에 대한 모션 정보를 포함할 수도 있다. 인터 예측 프로세싱 유닛 (120) 은 PU가 I 슬라이스, P 슬라이스, 또는 B 슬라이스 중 어느 것에 있는지에 의존하여 CU의 PU에 대해 상이한 동작들을 수행할 수도 있다. I 슬라이스에서, 인트라 예측이 가능하게 될 수도 있지만, 인터 예측은 가능하지 않게 된다. 따라서, PU가 I 슬라이스에 있으면, 인터 예측 프로세싱 유닛 (120) 은 PU에 대해 인터 예측을 수행하지 않는다.

[0213] PU가 P 슬라이스에 있으면, 모션 추정 유닛 (122) 은 PU에 대한 참조 지역을 참조 화상들의 리스트 (예컨대, "RefPicList0") 에서의 참조 화상들에서 검색할 수도 있다. PU에 대한 참조 지역은, 참조 화상 내의, PU의 예측 블록들에 가장 밀접하게 대응하는 샘플들을 포함하는 지역일 수도 있다. 모션 추정 유닛 (122) 은 PU에 대한 참조 지역을 포함하는 참조 화상의 RefPicList0에서의 포지션을 나타내는 참조 인덱스를 생성할 수도 있다. 덧붙여서, 모션 추정 유닛 (122) 은 PU의 예측 블록 및 참조 지역과 연관된 참조 로케이션 사이의 공간적 변위를 나타내는 모션 벡터를 생성할 수도 있다. 예를 들면, 모션 벡터는 현재 화상에서의 좌표들로부터 참조 화상에서의 좌표들로의 오프셋을 제공하는 2차원 벡터일 수도 있다. 모션 추정 유닛 (122) 은 참조 인덱스 및 모션 벡터를 PU의 모션 정보로서 출력할 수도 있다. 모션 보상 유닛 (124) 은 PU의 모션 벡터에 의해 나타내어진 참조 로케이션에 있는 실제 또는 보간된 샘플들에 기초하여 PU의 예측 블록들을 생성할 수도 있다.

[0214] PU가 B 슬라이스에 있다면, 모션 추정 유닛 (122) 은 PU에 대해 단예측 또는 양예측을 수행할 수도 있다. PU에 대해 단예측을 수행하기 위해, 모션 추정 유닛 (122) 은 그 PU에 대한 단일 모션 벡터 및 단일 참조 인덱스를 결정할 수도 있다. 일부 예들에서, 모션 추정 유닛 (122) 은 모션 벡터와 참조 인덱스를 결정하기 위해 병합 모드 또는 AMVP 모드를 사용할 수도 있다. 일부 예들에서, PU에 대한 참조 지역을 RefPicList0 또는 제 2 참조 화상 리스트 (예컨대, "RefPicList1") 의 참조 화상들에서 검색할 수도 있다. 모션 추정 유닛 (122) 은, PU의 모션 정보로서, 참조 지역을 포함하는 참조 화상의 RefPicList0 또는 RefPicList1에서의 포지션을 나타내는 참조 인덱스, PU의 예측 블록 및 참조 지역과 연관된 참조 로케이션 사이의 공간적 변위를 나타내는 모션 벡터, 및 참조 화상이 RefPicList0에 있는지 또는 RefPicList1에 있는지를 나타내는 하나 이상의 예측 방향 표시자들을 출력할 수도 있다. 모션 보상 유닛 (124) 은 PU의 모션 벡터에 의해 나타내어진 참조 로케이션에 있는 실제 또는 보간된 샘플들에 적어도 부분적으로 기초하여 PU의 예측 블록들을 생성할 수도 있다.

[0215] PU에 대해 양방향 인터 예측을 수행하기 위해, 모션 추정 유닛 (122) 은 그 PU에 대해 2 개의 모션 벡터들 및 2 개의 참조 인덱스들을 결정할 수도 있다. 일부 예들에서, 모션 추정 유닛 (122) 은 모션 벡터들과 참조 인덱스들을 결정하기 위해 병합 모드 또는 AMVP 모드를 수행할 수도 있다. 일부 예들에서, 모션 추정 유닛 (122) 은 그 PU에 대한 참조 지역을 RefPicList0에서의 참조 화상들에서 검색할 수도 있고, 또한 그 PU에 대한 다른 참조 지역을 RefPicList1에서의 참조 화상들에서 검색할 수도 있다. 모션 추정 유닛 (122) 은 참조 지역들을 포함하는 참조 화상들의 RefPicList0 및 RefPicList1에서의 포지션들을 나타내는 참조 인덱스들을 생성할 수도 있다. 덧붙여서, 모션 추정 유닛 (122) 은 그 참조 지역들과 연관된 참조 로케이션들 및 PU의 예측 블록 사이의 공간적 변위들을 나타내는 모션 벡터들을 생성할 수도 있다. PU의 모션 정보는 PU의 참조 인덱스들 및 모션 벡터들을 포함할 수도 있다. 모션 보상 유닛 (124) 은 PU의 모션 벡터들에 의해 나타내어진 참조 로케이션들에 있는 실제 또는 보간된 샘플들에 적어도 부분적으로 기초하여 PU의 예측 블록들을 생성할 수도 있다.

[0216] 인트라 예측 프로세싱 유닛 (126) 은 PU에 대해 인트라 예측을 수행함으로써 그 PU에 대한 예측 데이터를 생성할 수도 있다. PU에 대한 예측 데이터는 PU에 대한 예측 블록들과 다양한 신택스 엘리먼트들을 포함할 수도 있다. 인트라 예측 프로세싱 유닛 (126) 은 I 슬라이스들, P 슬라이스들, 및 B 슬라이스들에서의 PU들에 대해 인트라 예측을 수행할 수도 있다.

[0217] PU에 대해 인트라 예측을 수행하기 위해, 인트라 예측 프로세싱 유닛 (126) 은 PU에 대한 예측 블록들의 다수의

세트들을 생성하기 위해 다수의 인트라 예측 모드들을 사용할 수도 있다. 특정 인트라 예측 모드를 사용하여 인트라 예측을 수행하는 경우, 인트라 예측 프로세싱 유닛 (126) 은 이웃하는 블록들로부터의 샘플들의 특정 세트를 사용하여 PU에 대한 예측 블록들을 생성할 수도 있다. PU들, CU들, 및 CTU들에 대한 좌측에서 우측으로, 상단에서 하단으로의 인코딩 순서를 가정하면, 이웃하는 블록들은 PU의 상측, 우상측, 좌상측, 또는 좌측에 있을 수도 있다. 인트라 예측 프로세싱 유닛 (126) 은 다양한 수들의 인트라 예측 모드들, 예컨대, 33 개의 방향성 인트라 예측 모드들을 사용할 수도 있다. 일부 예들에서, 인트라 예측 모드들의 수는 PU의 예측 블록들의 사이즈에 의존할 수도 있다.

[0218] 예측 프로세싱 유닛 (100) 은 CU의 PU들에 대한 예측 데이터를, 그 PU들에 대해 인터 예측 프로세싱 유닛 (120) 에 의해 생성된 예측 데이터 또는 그 PU들에 대해 인트라 예측 프로세싱 유닛 (126) 에 의해 생성된 예측 데이터 중에서 선택할 수도 있다. 일부 예들에서, 예측 프로세싱 유닛 (100) 은 예측 데이터의 세트들의 레이트/왜곡 메트릭들에 기초하여 CU의 PU들에 대한 예측 데이터를 선택한다. 선택된 예측 데이터의 예측 블록들은 본원에서는 선택된 예측 블록들이라고 지칭될 수도 있다.

[0219] 잔차 생성 유닛 (102) 은, CU의 코딩 블록들 (예컨대, CU의 루마, Cb 및 Cr 코딩 블록) 및 그 CU의 PU들의 선택된 예측 블록들 (예컨대, 그 CU의 PU들의 루마, Cb 및 Cr 블록들) 에 기초하여, CU의 하나 이상의 잔차 블록들 (예컨대, 그 CU의 루마, Cb 및 Cr 잔차 블록들) 을 생성할 수도 있다. 예를 들면, 잔차 생성 유닛 (102) 은 CU의 잔차 블록들에서의 각각의 샘플이 그 CU의 코딩 블록에서의 샘플 및 그 CU의 PU의 대응하는 선택된 예측 블록에서의 대응하는 샘플 사이의 차이와 동일한 값을 가지도록 CU의 잔차 블록들을 생성할 수도 있다. 일부 예들에서, 잔차 생성 유닛 (102) 은 합산기를 포함할 수도 있다.

[0220] 변환 프로세싱 유닛 (104) 은 쿼드트리 구획화를 수행하여 CU의 잔차 블록들을 그 CU의 TU들과 연관된 변환 블록들로 구획화할 수도 있다. 따라서, TU가 루마 변환 블록 및 2 개의 대응하는 크로마 변환 블록들과 연관될 수도 있다. CU의 TU들의 루마 및 크로마 변환 블록들의 사이즈들 및 포지션들은 그 CU의 PU들의 예측 블록들의 사이즈들 및 포지션들에 기초할 수도 있거나 또는 기초하지 않을 수도 있다.

[0221] 변환 프로세싱 유닛 (104) 은 CU의 각각의 TU에 대한 변환 계수 블록들을, 하나 이상의 변환들을 TU의 변환 블록들에 적용함으로써 생성할 수도 있다. 변환 프로세싱 유닛 (104) 은 다양한 변환들을 TU와 연관된 변환 블록에 적용할 수도 있다. 예를 들어, 변환 프로세싱 유닛 (104) 은 이산 코사인 변환 (DCT), 방향성 변환, 또는 개념적으로 유사한 변환을 변환 블록에 적용할 수도 있다. 일부 예들에서, 변환 프로세싱 유닛 (104) 은 변환들을 변환 블록에 적용하지 않는다. 이러한 예들에서, 변환 블록은 변환 계수 블록으로서 취급될 수도 있다.

[0222] 양자화 유닛 (106) 은 계수 블록에서의 변환 계수들을 양자화할 수도 있다. 양자화 프로세스는 변환 계수들의 일부 또는 전부와 연관된 비트 깊이를 감소시킬 수도 있다. 예를 들어, n-비트 변환 계수가 양자화 동안에 m-비트 변환 계수로 버림될 (rounded down) 수도 있으며, 여기서 n은 m보다 크다. 양자화는 정보의 손실을 도입할 수도 있고, 그러므로, 양자화된 변환 계수들은 원래의 것들보다 낮은 정밀도를 가질 수도 있다.

[0223] 역 양자화 유닛 (108) 과 역 변환 프로세싱 유닛 (110) 은 계수 블록으로부터 잔차 블록을 복원하기 위해 역 양자화 및 역 변환들을 계수 블록에 각각 적용할 수도 있다. 복원 유닛 (112) 은 TU와 연관된 복원된 변환 블록을 생성하기 위해 복원된 잔차 블록을 예측 프로세싱 유닛 (100) 에 의해 생성된 하나 이상의 예측 블록들로부터의 대응하는 샘플들에 가산할 수도 있다. CU의 각각의 TU에 대한 변환 블록들을 이런 식으로 복원함으로써, 비디오 인코더 (20) 는 CU의 코딩 블록들을 복원할 수도 있다. 일부 예들에서, 복원 유닛 (112) 은 합산기를 포함할 수도 있다.

[0224] 필터 유닛 (114) 은 하나 이상의 디블록킹 동작들을 수행하여 CU와 연관된 코딩 블록들에서의 블록킹 아티팩트들을 감소시킬 수도 있다. 디코딩된 화상 버퍼 (116) 는, 필터 유닛 (114) 이 복원된 코딩 블록들에 대해 하나 이상의 디블록킹 동작들을 수행한 후에 복원된 코딩 블록들을 저장할 수도 있다. 인터 예측 프로세싱 유닛 (120) 은 다른 화상들의 PU들에 대해 인터 예측을 수행하기 위해 복원된 코딩 블록들을 포함하는 참조 화상을 사용할 수도 있다. 덧붙여서, 인트라 예측 프로세싱 유닛 (126) 은 CU와 동일한 화상에서의 다른 PU들에 대해 인트라 예측을 수행하기 위해 디코딩된 화상 버퍼 (116) 에서의 복원된 코딩 블록들을 사용할 수도 있다. 디코딩된 화상 버퍼 (116) 는 메모리 또는 다른 유형의 컴퓨터 판독가능 데이터 저장 매체를 포함할 수도 있다.

[0225] 엔트로피 인코딩 유닛 (118) 은 비디오 인코더 (20) 의 다른 기능적 컴포넌트들로부터 데이터를 수신할 수도 있

다. 예를 들어, 엔트로피 인코딩 유닛 (118) 은 양자화 유닛 (106) 으로부터 계수 블록들을 수신할 수도 있고 예측 프로세싱 유닛 (100) 으로부터 신택스 엘리먼트들을 수신할 수도 있다. 엔트로피 인코딩 유닛 (118) 은 데이터에 대해 하나 이상의 엔트로피 인코딩 동작들을 수행하여 엔트로피 인코딩된 데이터를 생성할 수도 있다. 예를 들어, 엔트로피 인코딩 유닛 (118) 은 그 데이터에 대해 콘텍스트 적응 가변 길이 코딩 (context-adaptive variable length coding; CAVLC) 동작, CABAC 동작, V2V (variable-to-variable) 길이 코딩 동작, 신택스 기반 콘텍스트 적응 이진 산술 코딩 (syntax-based context-adaptive binary arithmetic coding; SBAC) 동작, 확률 간격 구획화 엔트로피 (Probability Interval Partitioning Entropy; PIPE) 코딩 동작, 지수-골롬 인코딩 동작, 또는 다른 유형의 엔트로피 인코딩 동작을 수행할 수도 있다. 비디오 인코더 (20) 는 엔트로피 인코딩 유닛 (118) 에 의해 생성된 엔트로피 인코딩된 데이터를 포함하는 비트스트림을 출력할 수도 있다. 더욱이, 본 개시물의 예들에 따라, 도 6의 비디오 인코더 (20) 는, 비트스트림에서, 현재 뷰 성분/계층 표현에 대한 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번이라도 포함되는지의 여부를 나타내는 신택스 엘리먼트를 시그널링할 수도 있다.

[0226] 도 7은 본 개시물의 기법들을 구현하도록 구성되는 일 예의 비디오 디코더 (30) 를 도시하는 블록도이다. 도 7은 설명의 목적으로 제공되고 본 개시물에서 폭넓게 예시되고 설명된 바와 같은 기법들에 대해 제한하고 있지는 않다. 설명의 목적으로, 본 개시물은 HEVC 코딩의 맥락에서 비디오 디코더 (30) 를 설명한다. 그러나, 본 개시물의 기법들은 다른 코딩 표준들 또는 방법들에 적용가능할 수도 있다.

[0227] 도 7의 예에서, 비디오 디코더 (30) 는 엔트로피 디코딩 유닛 (150), 예측 프로세싱 유닛 (152), 역 양자화 유닛 (154), 역 변환 프로세싱 유닛 (156), 복원 유닛 (158), 필터 유닛 (160), 및 디코딩된 화상 버퍼 (162) 를 구비한다. 예측 프로세싱 유닛 (152) 은 모션 보상 유닛 (164) 과 인트라 예측 프로세싱 유닛 (166) 을 구비한다. 다른 예들에서, 비디오 디코더 (30) 는 더 많거나, 더 적거나, 또는 상이한 기능적 컴포넌트들을 포함할 수도 있다.

[0228] 코딩된 화상 버퍼 (CPB) (151) 가 비트스트림의 인코딩된 비디오 데이터 (예컨대, NAL 유닛들) 를 수신 및 저장할 수도 있다. CPB (151) 는 메모리 또는 다른 유형의 컴퓨터 판독가능 데이터 저장 매체를 포함할 수도 있다. 엔트로피 디코딩 유닛 (150) 은 CPB (151) 로부터 NAL 유닛들을 수신하고 그 NAL 유닛들을 파싱하여 비트스트림으로부터 신택스 엘리먼트들을 획득할 수도 있다. 엔트로피 디코딩 유닛 (150) 은 NAL 유닛들에서의 엔트로피 인코딩된 신택스 엘리먼트들을 엔트로피 디코딩할 수도 있다. 예측 프로세싱 유닛 (152), 역 양자화 유닛 (154), 역 변환 프로세싱 유닛 (156), 복원 유닛 (158), 및 필터 유닛 (160) 은 비트스트림으로부터 획득된 신택스 엘리먼트들에 기초하여 디코딩된 비디오 데이터를 생성할 수도 있다.

[0229] 비트스트림의 NAL 유닛들은 코딩된 슬라이스 NAL 유닛들을 포함할 수도 있다. 비트스트림을 디코딩하는 부분으로서, 엔트로피 디코딩 유닛 (150) 은 코딩된 슬라이스 NAL 유닛들로부터 신택스 엘리먼트들을 획득 및 엔트로피 디코딩할 수도 있다. 코딩된 슬라이스들의 각각은 슬라이스 헤더 및 슬라이스 데이터를 포함할 수도 있다. 슬라이스 헤더는 슬라이스에 관한 신택스 엘리먼트들을 포함할 수도 있다. 더욱이, 본 개시물의 하나 이상의 예들에 따라, 엔트로피 디코딩 유닛 (150) 은, 비트스트림으로부터, 현재 뷰 성분/계층 표현에 대한 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번이라도 포함되는지의 여부를 나타내는 신택스 엘리먼트를 획득할 수도 있다.

[0230] 비트스트림으로부터 신택스 엘리먼트들을 획득하는 것에 더하여, 비디오 디코더 (30) 는 CU에 대해 디코딩 동작을 수행할 수도 있다. CU에 대해 디코딩 동작을 수행함으로써, 비디오 디코더 (30) 는 그 CU의 코딩 블록들을 복원할 수도 있다.

[0231] CU에 대해 디코딩 동작을 수행하는 부분으로서, 역 양자화 유닛 (154) 은 그 CU의 TU들과 연관된 계수 블록들을 역 양자화, 즉, 양자화해제 (de-quantization) 할 수도 있다. 역 양자화 유닛 (154) 은 TU의 CU와 연관된 QP 값을 사용하여 양자화 정도를 결정하고, 비슷하게, 역 양자화 유닛 (154) 에 대해 적용될 역 양자화 정도를 결정할 수도 있다. 다시 말하면, 압축 비율, 즉, 원래의 시퀀스 및 압축된 시퀀스를 표현하는데 사용된 비트들의 수의 비율은, 변환 계수들을 양자화하는 경우에 사용된 QP의 값을 조정함으로써 제어될 수도 있다. 압축 비율은 채용된 엔트로피 코딩하는 방법에 또한 의존할 수도 있다.

[0232] 역 양자화 유닛 (154) 이 계수 블록을 역 양자화한 후, 역 변환 프로세싱 유닛 (156) 은 TU와 연관된 잔차 블록을 생성하기 위하여 하나 이상의 역 변환들을 계수 블록에 적용할 수도 있다. 예를 들어, 역 변환 프로세싱 유닛 (156) 은 역 DCT, 역 정수 변환, 역 카루넨-루베 변환 (Karhunen-Loeve transform; KLT), 역 회전 변환, 역 방향성 변환, 또는 다른 역 변환을 계수 블록에 적용할 수도 있다.

[0233] PU가 인트라 예측을 사용하여 인코딩되면, 인트라 예측 프로세싱 유닛 (166) 은 PU에 대한 예측 블록들을 생성하기 위해 인트라 예측을 수행할 수도 있다. 인트라 예측 프로세싱 유닛 (166) 은 인트라 예측 모드를 사용하여 공간적으로 이웃하는 PU들의 예측 블록들에 기초하여 PU에 대한 예측 루마, Cb 및 Cr 블록들을 생성할 수도 있다. 인트라 예측 프로세싱 유닛 (166) 은 비트스트림으로부터 디코딩된 하나 이상의 신택스 엘리먼트들에 기초하여 PU에 대한 인트라 예측 모드를 결정할 수도 있다.

[0234] 예측 프로세싱 유닛 (152) 은 비트스트림으로부터 추출된 신택스 엘리먼트들에 기초하여 제 1 참조 화상 리스트 (RefPicList0) 및 제 2 참조 화상 리스트 (RefPicList1) 를 구축할 수도 있다. 더욱이, PU가 인터 예측을 사용하여 인코딩되면, 엔트로피 디코딩 유닛 (150) 은 그 PU에 대한 모션 정보를 획득할 수도 있다. 모션 보상 유닛 (164) 은, PU의 모션 정보에 기초하여, PU에 대한 하나 이상의 참조 지역들을 결정할 수도 있다. 모션 보상 유닛 (164) 은, PU에 대한 하나 이상의 참조 블록들의 샘플들에 기초하여, PU에 대한 예측 루마, Cb 및 Cr 블록들을 생성할 수도 있다.

[0235] 복원 유닛 (158) 은, 적용가능한 바와 같이, CU의 TU들과 연관된 루마, Cb 및 Cr 변환 블록들과 그 CU의 PU들의 예측 루마, Cb 및 Cr 블록들로부터의 잔차 값들, 즉 인트라 예측 데이터 또는 인터 예측 데이터 중 어느 하나를 사용하여 그 CU의 루마, Cb 및 Cr 코딩 블록들을 복원할 수도 있다. 예를 들어, 복원 유닛 (158) 은 루마, Cb 및 Cr 변환 블록들의 샘플들을 예측 루마, Cb 및 Cr 블록들의 대응하는 샘플들에 가산하여 CU의 루마, Cb 및 Cr 코딩 블록들을 복원할 수도 있다. 일부 예들에서, 복원 유닛 (158) 은 합산기를 포함할 수도 있다.

[0236] 필터 유닛 (160) 은 디블록킹 동작을 수행하여 CU의 코딩 블록들 (예컨대, CU의 루마, Cb 및 Cr 코딩 블록들) 과 연관된 블록킹 아티팩트들을 감소시킬 수도 있다. 비디오 디코더 (30) 는 CU의 코딩 블록들 (예컨대, 루마, Cb 및 Cr 코딩 블록들) 을 디코딩된 화상 버퍼 (162) 에 저장할 수도 있다. 디코딩된 화상 버퍼 (162) 는 메모리 또는 다른 유형의 컴퓨터 판독가능 데이터 저장 매체를 포함할 수도 있다. 디코딩된 화상 버퍼 (162) 는 후속하는 모션 보상, 인트라 예측, 및 디스플레이 디바이스, 이를테면 도 1의 디스플레이 디바이스 (32) 상의 프레젠테이션을 위해 참조 화상들을 제공할 수도 있다. 예를 들면, 비디오 디코더 (30) 는, 디코딩된 화상 버퍼 (162) 에서의 블록들 (예컨대, 루마, Cb 및 Cr 블록들) 에 기초하여, 다른 CU들의 PU들에 대해 인트라 예측 또는 인터 예측 동작들을 수행할 수도 있다. 이런 식으로, 비디오 디코더 (30) 는, 비트스트림으로부터, 유효 계수 블록들의 변환 계수 레벨들을 획득하며, 그 변환 계수 레벨들을 역 양자화하며, 그 변환 계수 레벨들에 하나 이상의 변환들을 적용하여 변환 블록들을 생성하며, 그 변환 블록들에 적어도 부분적으로 기초하여, 코딩 블록들을 생성하고, 그 코딩 블록들을 디스플레이를 위해 출력할 수도 있다.

[0237] 위에서 나타낸 바와 같이, 본 개시물의 일부 예들은 RefPicList1 내의 뷰/계층 간 참조 화상들의 포함을 불가능하게 하는 하나 이상의 신택스 엘리먼트들을 제공할 수도 있다. 예를 들어, 하이 레벨 신택스 구조, 이를테면 VPS는 inter_view_ll_disable_flag 신택스 엘리먼트를 포함할 수도 있다. inter_view_ll_disable_flag 신택스 엘리먼트는 적용가능한 RefPicList1들이 뷰/계층 간 참조 화상들을 포함할 수도 있는지의 여부를 나타낼 수도 있다. 이 예에서, 적용가능한 RefPicList1들은 하이 레벨 신택스 구조를 참조하는 코딩된 화상들/계층 표현들의 RefPicList1들일 수도 있다. 더욱이, 이 예에서, 적용가능한 RefPicList1들이 뷰/계층 간 참조 화상들을 포함할 수도 있다는 것을 inter_view_ll_disable_flag 신택스 엘리먼트가 나타내는 경우, 슬라이스의 슬라이스 헤더는 슬라이스에 적용가능한 RefPicList1에서의 뷰/계층 간 참조 화상들의 시작 포지션을 나타내는 신택스 엘리먼트 (예컨대, inter_view_ref_start_position_ll_plus) 를 포함할 수도 있다. 아래의 표 4는, 이 예에 따른 일 예의 슬라이스 헤더 신택스를 보여준다. 표 4에서, 밑줄 쳐진 텍스트는 MV-HEVC 규격 초안 2 및/또는 3D-HEVC 테스트 모델 설명 초안 2에 추가된 텍스트를 나타낸다.

표 4

slice_header( ) {	기술자
first_slice_in_pic_flag	u(1)
…	
if(slice_header_extension_present_flag) { // should always be true in MV-HEVC	
slice_header_extension_length	ue(v)
if(slice_type != I_SLICE) {	
inter_view_ref_start_position_l0_plus1	ue(v)
if (!inter_view_l1_disable_flag)	
inter_view_ref_start_position_l1_plus1	ue(v)
…	
}	
byte_alignment( )	
}	

[0238]

[0239] 표 4에서, inter_view_ref_start_position_l0_plus1 신택스 엘리먼트는, 위에서 설명된 바와 같이, JCT3V-C0060에서의 inter_view_ref_start_position_plus1 신택스 엘리먼트의 시맨틱스와 동일한 시맨틱스를 갖는다.

더욱이, 표 4에서, inter_view_ref_start_position_l1_plus1 신택스 엘리먼트는 JCT3V-C0060에서의 inter_view_ref_start_position_plus1의 시맨틱스와 유사한 시맨틱스를 갖지만, RefPicList1에 적용가능하다.

따라서, 표 4의 예에서, 비디오 인코더 (20) 는 임의의 RPLM 신택스 엘리먼트들 (예컨대, list_entry_l1 신택스 엘리먼트들) 을 반드시 시그널링하는 일 없이, 뷰/계층 간 참조 화상들이 RefPicList1에서의 임의적 포지션에 삽입될 것임을 시그널링할 수도 있다. 따라서, inter_view_ref_start_position_l1_plus1 신택스 엘리먼트의 사용은 비트스트림의 사이즈를 감소시킬 수도 있다. 더욱이, 단일 inter_view_l1_disable_flag 신택스 엘리먼트가 다수의 화상들에 적용가능할 수도 있고 비디오 인코더 (20) 는 화상에 적용가능한 inter_view_l1_disable_flag 신택스 엘리먼트가 0인 경우 화상의 슬라이스에 대한 inter_view_ref_start_position_l1_plus1 신택스 엘리먼트만을 시그널링한다. 따라서, inter_view_l1_disable_flag 신택스 엘리먼트의 사용은 시그널링되는 inter_view_ref_start_position_l1_plus1 신택스 엘리먼트들의 수를 감소시킴으로써 비트스트림의 사이즈를 더욱 감소시킬 수도 있다.

[0240] 따라서, 표 4의 예에서, inter_view_l1_disable_flag는 제 1 신택스 엘리먼트일 수도 있고 비디오 인코더 (20) 는, 비트스트림에서, 제 2 신택스 엘리먼트 (예컨대, inter_view_ref_start_position_l0_plus1) 를 시그널링할 수도 있다. 제 2 신택스 엘리먼트는 현재 뷰 성분/계층 표현에 대한 RefPicList0에서의 뷰/계층 간 참조 화상들의 시작 포지션을 나타낼 수도 있다. 마찬가지로, 비디오 디코더 (30) 는, 비트스트림으로부터, 제 2 신택스 엘리먼트 (예컨대, inter_view_ref_start_position_l0_plus1) 를 획득할 수도 있다. 더욱이, 표 4의 예에서, 현재 뷰 성분/계층 표현에 대한 제 1 참조 화상 리스트 (예컨대, RefPicList1) 에 뷰/계층 간 참조 화상들이 포함됨을 inter_view_l1_disable_flag가 나타내는 경우, 비디오 코더는, 비트스트림으로부터, 제 3 신택스 엘리먼트 (예컨대, inter_view_ref_start_position_l1_plus1) 를 획득할 수도 있다. 제 3 신택스 엘리먼트는 현재 뷰 성분/계층 표현에 대한 제 1 참조 화상 리스트 (예컨대, RefPicList1) 에서의 뷰/계층 간 참조 화상들의 시작 포지션을 나타낼 수도 있다.

[0241] 더욱이, 본 개시물의 기법들은 MV-HEVC 규격 초안 2 및/또는 3D-HEVC 테스트 모델 설명 초안 2에서 정의된 참조 화상 리스트 수정 시맨틱스를 수정할 수도 있다. 본 개시물의 다른 곳에서 설명되는 바와 같이, 비디오 디코더 (30) 는 list_entry_l0 신택스 엘리먼트들 및 list_entry_l1 신택스 엘리먼트들을 근거로 RPLM을 수행할 수도 있다. 더욱이, 본 개시물의 다른 곳에서 설명되는 바와 같이, 비디오 디코더 (30) 는 NumPocTotalCurr 변수에 기초하여 각각의 list_entry_l0 신택스 엘리먼트 및 각각의 list_entry_l1 신택스 엘리먼트의 길이 (예컨대, 비트들의 수) 를 결정할 수도 있다. HEVC 규격 초안 8에서, NumPocTotalCurr는 NumPocStCurrBefore + NumPocStCurrAfter + NumPocLtCurr와 동일하게 설정된다. NumPocStCurrBefore는 RefPicSetStBefore에서의 엘리먼트들의 수를 나타낸다. NumPocStCurrAfter는 RefPicSetStAfter에서의 엘리먼트들의 수를 나타낸다. NumPocLtCurr는 RefPicSetLtCurr에서의 엘리먼트들의 수를 나타낸다.

[0242] 그러나, MV-HEVC 및 3D-HEVC에서, 참조 화상 리스트가 뷰/계층 간 참조 화상들을 포함할 수도 있다. 따라서, NumPocTotalCurr의 정의는 상이할 수도 있다. 아래의 텍스트에서, 밑줄 쳐진 텍스트는 MV-HEVC 규

격 초안 2 및/또는 3D-HEVC 테스트 모델 설명 초안 2에 추가된다. 아래의 텍스트에서 보인 바와 같이, 변수 NumPocTotalCurr는 다음과 같이 도출될 수도 있다.

```
NumPocTotalCurr = 0;
for( i = 0; i < NumNegativePics[ StRpsIdx ]; i++)
    if(UsedByCurrPicS0[ StRpsIdx ][ i ] = = 1)
        NumPocTotalCurr++
for( i = 0; i < NumPositivePics[ StRpsIdx ]; i++)          (7-56)
    if(UsedByCurrPicS1[ StRpsIdx ][ i ] = = 1)
        NumPocTotalCurr++
for( i = 0; i < num_long_term_sps + num_long_term_pics; i++ )
    if( UsedByCurrPicLt[ i ] = = 1)
        NumPocTotalCurr++
```

[0243]

[0244] 변수 NumPocTotalCurr는 NumPocStCurrBefore + NumPocStCurrAfter + NumPocLtCurr + NumIvCurr와 동일하게 설정되는데, NumIvCurr는 뷰/계층 간 RPS에서의 엔트리들의 수이다.

[0245] 더욱이, 이 예에서, list_entry_l0[i] 및 list_entry_l1[i] 신택스 엘리먼트들은 다음의 시맨틱스를 가질 수도 있다:

[0246] **list_entry_l0**[i]는 참조 화상 리스트 0의 현재 포지션에 배치될 RefPicListTemp0에서의 참조 화상의 인덱스를 특정한다. list_entry_l0[i] 신택스 엘리먼트의 길이는 Ceil(Log2(NumPocTotalCurr)) 비트이다. list_entry_l0[i]의 값은 0 내지 NumPocTotalCurr - 1의 범위에 있을 것이다. 신택스 엘리먼트 list_entry_l0[i]가 존재하지 않으면, 그것은 0과 동일한 것으로 유추된다.

[0247] **list_entry_l1**[i]는 참조 화상 리스트 1의 현재 포지션에 배치될 RefPicListTemp1에서의 참조 화상의 인덱스를 특정한다. list_entry_l1[i] 신택스 엘리먼트의 길이는 Ceil(Log2(NumPocTotalCurr)) 비트이다. list_entry_l1[i]의 값은 0 내지 NumPocTotalCurr - 1의 범위에 있을 것이다. 신택스 엘리먼트 list_entry_l1[i]가 존재하지 않으면, 그것은 0과 동일한 것으로 유추된다.

[0248] 이 예에서의 list_entry_l0[i] 및 list_entry_l1[i] 신택스 엘리먼트들에 대한 시맨틱스는 HEVC 규격 초안 8에서 list_entry_l0[i] 및 list_entry_l1[i] 신택스 엘리먼트들에 대해 정의된 시맨틱스들과 유사할 수도 있다. 그러나, 이 예가 상이한 정의의 NumPocTotalCurr를 제공하기 때문에, list_entry_l0[i] 및 list_entry_l1[i] 신택스 엘리먼트들의 길이 및 범위는 HEVC 규격 초안 8에서와는 상이할 수도 있다.

[0249] 더욱이, 본 개시물의 일부 예들은 MV-HEVC 및/또는 3D-HEVC의 RPLM 프로세스를 수정할 수도 있다. 예를 들어, MV-HEVC 및 3D-HEVC의 RPLM 프로세스에서의 변수 NumPocTotalCurr의 정의는, MV-HEVC 규격 초안 2 및 3D-HEVC 테스트 모델 설명 초안 2에서 정의된 바와 같이, NumPocTotalCurr가 변수 NumIvCurr에 의존하도록 수정될 수도 있다. 위에서 나타낸 바와 같이, NumIvCurr는 뷰/계층 간 RPS에서의 엔트리들의 수일 수도 있다. 구체적으로는, 본 개시물의 하나의 예에서, 변수 NumPocTotalCurr는, list_entry_l0 신택스 엘리먼트들의 시맨틱스에 관해, 다음으로 정의될 수도 있다:

[0250] 변수 NumPocTotalCurr는 NumPocStCurrBefore + NumPocStCurrAfter + NumPocLtCurr + NumIvCurr와 동일하게 설정된다.

[0251] 더욱이, 이 예에서, list_entry_l0[i] 신택스 엘리먼트는 다음의 시맨틱스를 가질 수도 있다:

[0252] **list_entry_l0**[i]는 참조 화상 리스트 0의 현재 포지션에 배치될 RefPicListTemp0에서의 참조 화상의 인덱스를 특정한다. list_entry_l0[i] 신택스 엘리먼트의 길이는 Ceil(Log2(NumPocTotalCurr)) 비트이다. list_entry_l0[i]의 값은 0 내지 NumPocTotalCurr - 1의 범위에 있을 것이다. 신택스 엘리먼트 list_entry_l0[i]가 존재하지 않으면, 그것은 0과 동일한 것으로 유추된다.

[0253] 이 예에 따라, inter_view_l1_disable_flag가 1이면, 변수 NumPocTotalCurr는, list_entry_l1[i] 신택스 엘리먼트들의 시맨틱스에 관해, NumPocTotalCurr가 NumPocStCurrBefore + NumPocStCurrAfter + NumPocLtCurr와 동일하도록 추가로 수정된다. 이 예에서, list_entry_l1[i]는 다음의 시맨틱스를 가질 수도 있다:

[0254] **list_entry_l1**[i]는 참조 화상 리스트 1의 현재 포지션에 배치될 RefPicListTemp1에서의 참조 화상의 인덱스를 특정한다. list_entry_l1[i] 신택스 엘리먼트의 길이는 Ceil(Log2(NumPocTotalCurr)) 비트이다. list_entry_l1[i]의 값은 0 내지 NumPocTotalCurr - 1의 범위에 있을 것이다. 신택스 엘리먼트 list_entry_l1[i]가 존재하지 않으면, 그것은 0과 동일한 것으로 유추된다.

[0255] 본 개시물의 다른 예에서, MV-HEVC 규격 초안 2 및/또는 3D-HEVC 테스트 모델 설명 초안 2에서 정의된 list_entry_lX[i] (X는 0 또는 1과 동일함) 의 시맨틱스는 다음과 같이 수정된다. list_entry_lX[i]의 시맨틱스의 다음의 설명에서, 밑줄 쳐진 텍스트는 MV-HEVC 규격 초안 2 및/또는 3D-HEVC 테스트 모델 설명 초안 2에서의 list_entry_l0 및 list_entry_l1의 시맨틱스에 추가된 텍스트이다.

[0256] **list_entry_lX**[i]는 참조 화상 리스트 X의 현재 포지션에 배치될 RefPicListTempX에서의 참조 화상의 인덱스를 특정한다. list_entry_lX[i] 신택스 엘리먼트의 길이는 Ceil(Log2(NumPocTotalCurrLX)) 비트이다. list_entry_lX[i]의 값은 0 내지 NumPocTotalCurrLX - 1의 범위에 있을 것이다. 신택스 엘리먼트 list_entry_lX[i]가 존재하지 않으면, 그것은 0과 동일한 것으로 유추된다. inter_view_l1_disable_flag가 1과 동일한 경우, (NumPocTotalCurrL1- NumPocTotalCurrL0) 은 NumIvCurr와 동일하다. NumPocTotalCurrLX는 다음과 같이 도출된다:

[0257] NumPocTotalCurrLX= NumPocStCurrBefore + NumPocStCurrAfter + NumPocLtCurr + ((inter_view_l1_disable_flag && X) ? 0: NumIvCurr).

[0258] list_entry_l1 (및 list_entry_lX) 에 대한 예의 시맨틱스에서 도시된 바와 같이, 각각의 list_entry_l1 신택스 엘리먼트의 길이는 RefPicList1에서의 참조 화상들의 총 수 (즉, NumPocTotalCurr) 에 의존한다. 따라서, RefPicList1에 더 적은 참조 화상들이 있다면, list_entry_l1 신택스 엘리먼트들의 각각은 더 적은 비트들을 포함한다. inter_view_l1_disable_flag가 1이면, 더 적은 참조 화상들이 RefPicList1에 있는데, RefPicList1이 뷰/계층 간 참조 화상들을 포함하지 않기 때문이다. 따라서, 화상에 적용가능한 inter_view_l1_disable_flag가 1이면, 화상의 RefPicList1에 대한 각각의 list_entry_l1 신택스 엘리먼트는 더 적은 비트들을 포함할 수도 있다.

[0259] 따라서, 비디오 디코더 (30) 는, 비트스트림으로부터, 참조 화상 리스트를 수정하기 위한 참조 화상 리스트 수정 (RPLM) 신택스 엘리먼트들 (예컨대, list_entry_l1 신택스 엘리먼트들) 을 획득할 수도 있고, 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번도 포함되지 않음을 inter_view_l1_disable_flag가 나타내는 경우, 참조 화상 리스트에 뷰/계층 간 참조 화상들이 포함됨을 신택스 엘리먼트가 나타내는 경우보다 더 적은 비트들을 RPLM 신택스 엘리먼트들은 포함한다. 유사하게, 비디오 인코더 (20) 는, 비트스트림에서, 참조 화상 리스트를 수정하기 위한 RPLM 신택스 엘리먼트들 (예컨대, list_entry_l1 신택스 엘리먼트들) 을 시그널링할 수도 있고, 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번도 포함되지 않음을 inter_view_l1_disable_flag가 나타내는 경우, 참조 화상 리스트에 뷰/계층 간 참조 화상들이 포함됨을 신택스 엘리먼트가 나타내는 경우보다 더 적은 비트들을 RPLM 신택스 엘리먼트들의 각각은 포함할 수도 있다. 적어도 이런 식으로, inter_view_l1_disable_flag의 사용은 비트스트림의 사이즈를 감소시킬 수도 있다.

[0260] 도 8a는 본 개시물의 일 예에 따른, 비디오 인코더 (20) 의 동작을 도시하는 흐름도이다. 도 8a의 예에서, 비디오 인코더 (20) 는, 비디오 데이터의 인코딩된 표현을 포함하는 비트스트림에서, 현재 뷰 성분/계층 표현에 대한 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번이라도 포함되는지의 여부를 나타내는 신택스 엘리먼트를 시그널링할 수도 있다 (200). 일부 예들에서, 비디오 인코더 (20) 는 inter_view_l1_disable_flag 신택스 엘리먼트를 VPS에서 시그널링할 수도 있다. 일부 이러한 예들에서, inter_view_l1_disable_flag 신택스 엘리먼트는, CVS의 각각의 개별 뷰 성분/계층 표현에 대해, 개별 뷰 성분/계층 표현에 대한 개별 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번이라도 포함되는지의 여부를 나타낼 수도 있다. 더욱이, 일부 이러한 예들에서, 비디오 인코더 (20) 는, 비트스트림에서, VPS를 참조하는 CVS의 각각의 개별 뷰 성분/계층 표현에 대해, 개별 뷰 성분/계층 표현에 대한 개별 제 2 참조 화상 리스트 (예컨대, RefPicList0) 에서의 뷰/계층 간 참조 화상들의 시작 포지션을 나타내는 개별 제 1 신택스 엘리먼트 (예컨대, inter_view_ref_start_position_l0_plus1) 를 시그널링할 수도 있다.

[0261] 다른 예들에서, 비디오 인코더 (20) 는 복수의 계층들로부터의 각각의 개별 계층에 대한 inter_view_l1_disable_flag 신택스 엘리먼트들을 비트스트림에서 시그널링할 수도 있다. 이러한 예들에서, 개별 계층 신택스 엘리먼트에 대한 inter_view_l1_disable_flag 신택스 엘리먼트는, 개별 계층에서의 뷰 성분들/계층 표현들의 개별 참조 화상 리스트들에 뷰/계층 간 참조 화상들이 한번이라도 포함되는지의 여부를 나타낸다. 더욱이, 일부 예들에서, 비디오 인코더 (20) 는, 비트스트림에서, 복수의 계층들 중 각각의 계층의 각각의 개별 뷰 성분/계층 표현에 대해, 개별 뷰 성분/계층 표현에 대한 개별 부가적인 참조 화상 리스트에서의 뷰/계층 간 참조 화상들의 시작 포지션을 나타내는 개별 부가적인 신택스 엘리먼트 (예컨대, inter_view_ref_start_position_l1_plus1) 를 시그널링할 수도 있다.

[0262] 덧붙여서, 비디오 인코더 (20) 는 현재 뷰 성분/계층 표현을 인코딩할 수도 있다 (202). 본 개시물의 다양한 예들에서 설명되는 바와 같은, inter_view_l1_disable_flag 신택스 엘리먼트가 1인 경우, 비디오 인코더 (20) 는 현재 뷰 성분/계층 표현을 인코딩하기 위해 참조 화상 리스트 (예컨대, RefPicList1) 에서의 뷰/계층 간 참조 화상들을 사용하지 않는다.

[0263] 도 8b는 본 개시물의 일 예에 따른, 비디오 디코더 (30) 의 동작을 도시하는 흐름도이다. 도 8b의 예에서, 비디오 디코더 (30) 는, 비트스트림으로부터, 현재 뷰 성분/계층 표현에 대한 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번이라도 포함되는지의 여부를 나타내는 신택스 엘리먼트 (예컨대, inter_view_l1_disable_flag) 를 획득한다 (210). 일부 예들에서, 현재 뷰 성분/계층 표현은 VPS를 참조하는 CVS의 부분일 수도 있다. 이러한 예들에서, 비디오 디코더 (30) 는 VPS로부터 신택스 엘리먼트를 획득할 수도 있다. 더욱이, 이러한 예들에서, 그 신택스 엘리먼트는, CVS의 각각의 개별 뷰 성분/계층 표현에 대해, 개별 뷰 성분/계층 표현에 대한 개별 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번이라도 포함되는지의 여부를 나타낼 수도 있다. 일부 예들에서, 엔트로피 디코딩 유닛 (150) (도 7) 은 비트스트림으로부터 신택스 엘리먼트를 획득할 수도 있다.

[0264] 다른 예들에서, 현재 뷰 성분/계층 표현은 비트스트림에서 복수의 계층들 중에서의 특정 계층에 있다. 이러한 예들에서, 비디오 디코더 (30) 는, 복수의 계층들로부터의 각각의 개별 계층에 대해, 개별 계층에서의 뷰 성분들/계층 표현들의 개별 참조 화상 리스트들에 뷰/계층 간 참조 화상들이 한번이라도 포함되는지의 여부를 나타내는, 개별 계층에 대한 개별 신택스 엘리먼트를 획득할 수도 있다.

[0265] 비디오 디코더 (30) 는 현재 뷰 성분/계층 표현을 디코딩할 수도 있다 (212). 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번도 포함되지 않음을 신택스 엘리먼트가 나타내는 경우, 비디오 디코더 (30) 는 참조 화상 리스트에서의 뷰/계층 간 참조 화상들의 사용 없이 현재 뷰 성분/계층 표현을 디코딩한다.

[0266] 도 9는 본 개시물의 일 예에 따른, 슬라이스 헤더를 파싱하는 일 예의 동작을 도시하는 흐름도이다. 표 4의 예에 관해 위에서 나타낸 바와 같이, 슬라이스 헤더들은 RefPicList1에서의 뷰/계층 간 참조 화상들의 시작 포지션들을 나타내는 신택스 엘리먼트들 (즉, inter_view_ref_start_position_l1_plus1 신택스 엘리먼트들) 을 적응적으로 포함할 수도 있다.

[0267] 구체적으로는, 도 9의 예에서, 비디오 디코더 (30) 는, 비트스트림으로부터, inter_view_l1_disable_flag를 획득할 수도 있다 (230). 일부 예들에서, 비디오 디코더 (30) 는 VPS로부터 inter_view_l1_disable_flag를 획득할 수도 있다. 다른 예들에서, 비디오 디코더 (30) 는 SPS 또는 다른 신택스 구조로부터 inter_view_l1_disable_flag를 획득할 수도 있다. 예를 들면, 비디오 인코더 (20) 가 각각의 뷰/계층에 대해 inter_view_l1_disable_flag를 시그널링하는 경우, 비디오 인코더 (20) 는 계층들의 각각 내의 CVS들에 적용가능한 SPS들에서 inter_view_l1_disable_flags를 시그널링할 수도 있다. 일부 예들에서, 엔트로피 디코딩 유닛 (150) (도 7) 은 비트스트림으로부터 신택스 엘리먼트를 획득할 수도 있다.

[0268] 더욱이, 도 9의 예에서, 비디오 디코더 (30) 는 슬라이스 헤더의 신택스 엘리먼트들을 획득하기 위해 현재 화상의 슬라이스에 대한 슬라이스 헤더를 파싱할 수도 있다. 일부 예들에서, 엔트로피 디코딩 유닛 (150) 이 슬라이스 헤더를 파싱할 수도 있다. 슬라이스 헤더를 파싱하는 부분으로서, 비디오 디코더 (30) 는 슬라이스가 I 슬라이스인지의 여부를 결정할 수도 있다 (232). 다르게 말하면, 비디오 디코더 (30) 는 인터 예측이 슬라이스에서 허용되는지의 여부를 결정할 수도 있다. 슬라이스가 I 슬라이스가 아니라는 (즉, 블록이 P 또는 B 슬라이스라는) 결정 (232의 "아니오") 에 응답하여, 비디오 디코더 (30) 는, 슬라이스 헤더로부터, 현재 화상의 RefPicList0에서의 뷰/계층 간 참조 화상들의 시작 포지션을 나타내는 신택스 엘리먼트 (예컨대, inter_view_ref_start_position_l0_plus1) 를 획득할 수도 있다 (234).

[0269] 따라서, 현재 뷰 성분/계층 표현이 VPS를 참조하는 CVS의 부분이고 inter_view_l1_disable_flag가 VPS에서 시그널링되는 일부 예들에서, 비디오 디코더 (30) 는, CVS의 각각의 개별 뷰 성분/계층 표현에 대해, 개별 뷰 성분/계층 표현에 대한 개별 제 2 참조 화상 리스트에서의 뷰/계층 간 참조 화상들의 시작 포지션을 나타내는 개별 부가적인 신택스 엘리먼트 (예컨대, inter_view_ref_start_position_l0_plus1) 를 획득할 수도 있다.

[0270] 더욱이, inter_view_l1_disable_flag가 비트스트림의 복수의 계층들에서의 각각의 계층에 대해 시그널링되는 일부 예들에서, 비디오 디코더 (30) 는, 복수의 계층들 중 각각의 계층의 각각의 개별 뷰 성분/계층 표현에 대해, 개별 뷰 성분/계층 표현에 대한 개별 부가적인 참조 화상 리스트에서의 뷰/계층 간 참조 화상들의 시작 포지션을 나타내는 개별 부가적인 신택스 엘리먼트 (예컨대, inter_view_ref_start_position_l0_plus1) 를 획득할 수도 있다.

[0271] 더욱이, 비디오 디코더 (30) 는, inter_view_l1_disable_flag에 기초하여, 현재 화상의 RefPicList1에 뷰/계층 간 참조 화상들이 한번이라도 포함되는지의 여부를 결정할 수도 있다 (236). 현재 화상의 RefPicList1에 뷰/계층 간 참조 화상들이 포함된다는 결정 (236의 "예") 에 응답하여, 비디오 디코더 (30) 는, 슬라이스 헤더로부터, RefPicList1에서의 뷰/계층 간 참조 화상들의 시작 포지션을 나타내는 신택스 엘리먼트 (예컨대, inter_view_ref_start_position_l1_plus1) 를 획득할 수도 있다 (238).

[0272] 그 뒤에, 또는 슬라이스가 I 슬라이스라는 결정 (232의 "예") 에 응답하여, 또는 현재 화상의 RefPicList1에 뷰/계층 간 참조 화상들이 한번도 포함되지 않는다는 결정 (236의 "아니오") 에 응답하여, 비디오 디코더 (30) 는 슬라이스 헤더의 부가적인 신택스 엘리먼트들을, 만약 있다면, 획득할 수도 있다 (240).

[0273] 도 10은 본 개시물의 일 예에 따른, 이웃 블록 기반 디스패리티 벡터 (NBDV) 도출 프로세스를 도시하는 흐름도이다. 도 10의 예에서, 비디오 코더 (예컨대, 비디오 인코더 (20) 또는 비디오 디코더 (30)) 가, 현재 공간적 이웃 블록이 RefPicList0 디스패리티 모션 벡터를 갖는지의 여부를 결정할 수도 있다 (250). 현재 공간적 이웃 블록은 현재 블록의 공간적 이웃 블록들 중 하나일 수도 있다. 현재 공간적 이웃 블록이 RefPicList0 디스패리티 모션 벡터를 갖는다는 결정 (250의 "예") 에 응답하여, 비디오 코더는 현재 공간적 이웃 블록의 RefPicList0 디스패리티 모션 벡터를 현재 블록에 대한 디스패리티 벡터로 변환할 수도 있다 (252). 비디오 코더는 그 다음에 NBDV 도출 프로세스를 종료할 수도 있다.

[0274] 한편, 현재 공간적 이웃 블록이 RefPicList0 디스패리티 모션 벡터를 갖지 않는다는 결정 (250의 "아니오") 에 응답하여, 비디오 코더는 현재 공간적 이웃 블록에 대한 IDV를 저장할 수도 있다 (254). 일부 예들에서, 비디오 코더는 현재 공간적 이웃 블록의 예측 모드가 스킵 모드를 사용하여 코딩되는 경우에만 현재 공간적 이웃 블록에 대한 IDV를 저장할 수도 있다. 일부 예들에서, 비디오 코더는 디코딩된 화상 버퍼 (예컨대, 디코딩된 화상 버퍼 (116) 또는 디코딩된 화상 버퍼 (162)) 에 IDV를 저장할 수도 있다. 더욱이, 비디오 코더는 현재 공간적 이웃 블록의 RefPicList1에 뷰/계층 간 참조 화상들이 한번이라도 포함됨을 inter_view_l1_disable_flag가 나타내는지의 여부를 결정할 수도 있다 (256).

[0275] 현재 공간적 이웃 블록을 포함하는 화상의 RefPicList1 (즉, 현재 공간적 이웃 블록의 RefPicList1) 에 뷰/계층 간 참조 화상들이 포함될 수도 있음을 inter_view_l1_disable_flag가 나타낸다는 결정 (256의 "예") 에 응답하여, 비디오 코더는 현재 공간적 이웃 블록이 RefPicList1 디스패리티 모션 벡터를 갖는지의 여부를 결정할 수도 있다 (258). 현재 공간적 이웃 블록이 RefPicList1 디스패리티 모션 벡터를 갖는다는 결정 (258의 "예") 에 응답하여, 비디오 코더는 현재 공간적 이웃 블록에 대한 RefPicList1 디스패리티 모션 벡터를 현재 블록에 대한 디스패리티 벡터로 변환할 수도 있다 (260). 비디오 코더는 그 다음에 NBDV 도출 프로세스를 종료할 수도 있다.

[0276] 한편, 현재 공간적 이웃 블록이 RefPicList1 디스패리티 모션 벡터를 갖지 않는다는 결정 (258의 "아니오") 에 응답하여, 비디오 코더는 현재 공간적 이웃 블록에 대한 IDV를 저장할 수도 있다 (262). 일부 예들에서, 비디오 코더는 현재 공간적 이웃 블록의 예측 모드가 스킵 모드를 사용하여 코딩되는 경우에만 현재 공간적 이웃 블록에 대한 IDV를 저장할 수도 있다. 따라서, 비디오 코더는 현재 공간적 이웃 블록에 대해 2 개의 IDV들을 잠재적으로 저장할 수도 있다. 일부 예들에서, 비디오 코더는 디코딩된 화상 버퍼 (예컨대, 디코딩된 화상 버퍼 (116) 또는 디코딩된 화상 버퍼 (162)) 에 IDV를 저장할 수도 있다.

[0277] 현재 공간적 이웃 블록의 RefPicList1에 뷰/계층 간 참조 화상들이 한번도 포함되지 않음을 inter_view_l1_disable_flag가 나타낸다는 결정 (256의 "아니오") 에 응답하여, 또는 262에서 현재 공간적 이웃 블록에 대한 IDV를 저장한 후, 비디오 코더는 체크할 임의의 남아 있는 공간적 이웃 블록들이 있는지의 여부

를 결정할 수도 있다 (264). 체크할 하나 이상의 남아 있는 공간적 이웃 블록들이 있다는 결정 (264의 "예") 에 응답하여, 비디오 코더는 남아 있는 공간적 이웃 블록들 중 하나를 현재 공간적 이웃 블록으로 하여 액션들 (250 내지 264) 을 반복할 수도 있다. 따라서, 본 개시물의 이 예에 따라, 현재 공간적 이웃 블록의 RefPicList1에 뷰/계층 간 참조 화상들이 한번도 포함되지 않음을 inter_view_l1_disable_flag가 나타내는 경우, 비디오 코더는 현재 공간적 이웃 블록에 대해 많아야 하나의 IDV를 저장한다.

[0278] 더욱이, 남아 있는 공간적 이웃 블록들이 없다는 결정 (264의 "아니오") 에 응답하여, 비디오 코더는 현재 시간적 이웃 블록이 RefPicList0 디스패리티 모션 벡터를 갖는지의 여부를 결정할 수도 있다 (266). 현재 시간적 이웃 블록은 현재 블록의 시간적 이웃 블록들 중 하나일 수도 있다. 현재 시간적 이웃 블록이 RefPicList0 디스패리티 모션 벡터를 갖는다는 결정 (266의 "예") 에 응답하여, 비디오 코더는 현재 시간적 이웃 블록의 RefPicList0 디스패리티 모션 벡터를 현재 블록에 대한 디스패리티 벡터로 변환할 수도 있다 (268). 비디오 코더는 그 다음에 NBDV 도출 프로세스를 종료할 수도 있다.

[0279] 한편, 현재 시간적 이웃 블록이 RefPicList0 디스패리티 모션 벡터를 갖지 않는다는 결정 (266의 "아니오") 에 응답하여, 비디오 코더는 현재 시간적 이웃 블록에 대한 IDV를 저장할 수도 있다 (270). 일부 예들에서, 비디오 코더는 현재 시간적 이웃 블록의 예측 모드가 스킵 모드를 사용하여 코딩되는 경우에만 현재 시간적 이웃 블록에 대한 IDV를 저장할 수도 있다. 일부 예들에서, 비디오 코더는 디코딩된 화상 버퍼 (예컨대, 디코딩된 화상 버퍼 (116) 또는 디코딩된 화상 버퍼 (162)) 에 IDV를 저장할 수도 있다. 덧붙여서, 비디오 코더는 현재 시간적 이웃 블록의 RefPicList1에 뷰/계층 간 참조 화상들이 한번이라도 포함됨을 inter_view_l1_disable_flag가 나타내는지의 여부를 결정할 수도 있다 (272). 현재 시간적 이웃 블록의 RefPicList1에 뷰/계층 간 참조 화상들이 포함될 수도 있음을 inter_view_l1_disable_flag가 나타낸다는 결정 (272의 "예") 에 응답하여, 비디오 코더는 현재 시간적 이웃 블록이 RefPicList1 디스패리티 모션 벡터를 갖는지의 여부를 결정할 수도 있다 (274). 현재 시간적 이웃 블록이 RefPicList1 디스패리티 모션 벡터를 갖는다는 결정 (274의 "예") 에 응답하여, 비디오 코더는 현재 시간적 이웃 블록에 대한 RefPicList1 디스패리티 모션 벡터를 현재 블록에 대한 디스패리티 벡터로 변환할 수도 있다 (276). 비디오 코더는 그 다음에 NBDV 도출 프로세스를 종료할 수도 있다.

[0280] 한편, 현재 시간적 이웃 블록이 RefPicList1 디스패리티 모션 벡터를 갖지 않는다는 결정 (274의 "아니오") 에 응답하여, 비디오 코더는 현재 시간적 이웃 블록에 대한 IDV를 저장할 수도 있다 (278). 일부 예들에서, 비디오 코더는 현재 시간적 이웃 블록의 예측 모드가 스킵 모드를 사용하여 코딩되는 경우에만 현재 시간적 이웃 블록에 대한 IDV를 저장할 수도 있다. 따라서, 비디오 코더는 현재 시간적 이웃 블록에 대해 2 개의 IDV들을 잠재적으로 저장할 수도 있다. 일부 예들에서, 비디오 코더는 디코딩된 화상 버퍼 (예컨대, 디코딩된 화상 버퍼 (116) 또는 디코딩된 화상 버퍼 (162)) 에 IDV를 저장할 수도 있다.

[0281] 현재 시간적 이웃 블록의 RefPicList1에 뷰/계층 간 참조 화상들이 한번도 포함되지 않음을 inter_view_l1_disable_flag가 나타낸다는 결정 (272의 "아니오") 에 응답하여, 또는 278에서 현재 시간적 이웃 블록에 대한 IDV를 저장한 후, 비디오 코더는 체크할 임의의 남아 있는 시간적 이웃 블록들이 있는지의 여부를 결정할 수도 있다 (280). 체크할 하나 이상의 남아 있는 시간적 이웃 블록들이 있다는 결정 (280의 "예") 에 응답하여, 비디오 코더는 남아 있는 시간적 이웃 블록들 중 하나를 현재 시간적 이웃 블록으로 하여 액션들 (266 내지 280) 을 반복할 수도 있다. 따라서, 본 개시물의 이 예에 따라, 현재 시간적 이웃 블록의 RefPicList1에 뷰/계층 간 참조 화상들이 한번도 포함되지 않음을 inter_view_l1_disable_flag가 나타내는 경우, 비디오 코더는 현재 시간적 이웃 블록에 대해 많아야 하나의 IDV를 저장한다.

[0282] 남아 있는 시간적 이웃 블록들이 없다는 결정 (280의 "아니오") 에 응답하여, 비디오 코더는 현재 이웃 블록이 IDV를 갖는지의 여부를 결정할 수도 있다 (282). 현재 이웃 블록은 현재 블록의 공간적 이웃 블록들 중 하나 또는 공간적 시간적 이웃하는 블록들 중 하나일 수도 있다. 일부 예들에서, 비디오 코더는 IDV들에 대한 시간적 이웃 블록들 중 임의의 것을 체크하기 전에 IDV들에 대한 공간적 이웃 블록들을 체크한다. 현재 공간적 이웃 블록이 IDV를 갖는다는 결정 (282의 "예") 에 응답하여, 비디오 코더는 현재 이웃 블록의 IDV를 현재 블록의 디스패리티 벡터로 변환할 수도 있다 (284). 그러나, 현재 이웃 블록이 IDV를 갖지 않는다는 결정 (282의 "아니오") 에 응답하여, 비디오 코더는 임의의 남아 있는 이웃 블록들이 있는지의 여부를 결정할 수도 있다 (286). 체크할 하나 이상의 남아 있는 이웃 블록들이 있다는 결정 (286의 "예") 에 응답하여, 비디오 코더는 남아 있는 이웃하는 블록들 중 하나를 현재 이웃 블록으로 하여 액션들 (282 내지 286) 을 반복할 수도 있다. 한편, 남아 있는 이웃 블록들이 없다는 결정 (286의 "아니오") 에 응답하여, 비디오 코더는 현재 블

록에 대한 디스패리티 벡터가 이용불가라고 결정할 수도 있다 (288).

[0283] 따라서, 도 10의 예에서 도시된 바와 같이, RefPicList1이 뷰/계층 간 참조 화상들을 한번도 포함하지 않음을 inter_view_l1_disable_flag가 나타내는 경우, 모션 정보는 한번도 체크되지 않는다. 결과적으로, 도 10의 NBDV 도출 프로세스의 복잡도는 2배만큼 감소될 수도 있다. 일부 예들에서, 도 10의 동작은 비디오 인코더 (20) 의 인터 예측 프로세싱 유닛 (120) (도 6) 또는 비디오 디코더 (30) 의 예측 프로세싱 유닛 (152) (도 7) 에 의해 수행될 수도 있다.

[0284] 예에 의존하여, 상이한 시퀀스로 수행될 수도 있는 본원에서 설명된 방법들 중 임의의 것의 특정한 액트들 또는 이벤트들이 부가되거나, 병합되거나, 또는 다 함께 제외될 수도 있다 (예컨대, 모든 설명된 액트들 또는 이벤트들이 방법을 실용화에 필요한 것은 아니다) 는 것이 인정될 것이다. 더구나, 특정한 예들에서, 액트들 또는 이벤트들은 순차적으로라기 보다는, 예컨대, 다중 스레드식 프로세싱, 인터럽트 프로세싱, 또는 다수의 프로세서들을 통하여 동시에 수행될 수도 있다.

[0285] 위의 예들 중 임의의 것의 임의의 세부사항은 다른 예들과 본 개시물에 부합하게 조합될 수도 있다. 하나 이상의 예들에서, 설명된 기능들은 하드웨어, 소프트웨어, 펌웨어, 또는 그것들의 임의의 조합에서 구현될 수도 있다. 소프트웨어에서 구현된다면, 그 기능들은 하나 이상의 명령들 또는 코드로서 컴퓨터 판독가능 매체 상에 저장되거나 또는 그것을 통해 송신될 수도 있고 하드웨어 기반 프로세싱 유닛에 의해 실행될 수도 있다. 컴퓨터 판독가능 매체들은, 데이터 저장 매체들과 같은 유형의 (tangible) 매체에 대응하는 컴퓨터 판독가능 저장 매체들, 또는 예컨대 통신 프로토콜에 따라 한 장소에서 다른 장소로 컴퓨터 프로그램의 전달을 용이하게 하는 임의의 매체를 포함하는 통신 매체들을 포함할 수도 있다. 이런 방식으로, 컴퓨터 판독가능 매체들은 일반적으로 (1) 비일시적 (non-transitory) 인 유형의 컴퓨터 판독가능 저장 매체들 또는 (2) 신호 또는 반송파와 같은 통신 매체에 해당할 수도 있다. 데이터 저장 매체들은 본 개시물에서 설명된 기법들의 구현을 위한 명령들, 코드 및/또는 데이터 구조들을 취출하기 위해 하나 이상의 컴퓨터들 또는 하나 이상의 프로세서들에 의해 액세스될 수 있는 임의의 이용가능 매체들일 수도 있다. 컴퓨터 프로그램 제품은 컴퓨터 판독가능 매체를 포함할 수도 있다.

[0286] 당업자들은 본원에서 개시된 방법들, 시스템들, 및 장치들에 관련하여 설명되는 다양한 구체적인 논리 블록들, 모듈들, 회로들, 및 알고리즘 단계들이 전자 하드웨어, 프로세서에 의해 실행되는 컴퓨터 소프트웨어, 또는 양쪽 모두의 조합들로서 구현될 수도 있다는 것을 인정할 것이다. 하드웨어 및 소프트웨어의 이러한 교환가능성을 명백하게 예증하기 위하여, 다양한 예시적인 컴포넌트들, 블록들, 모듈들, 회로들, 및 단계들이 일반적으로 그것들의 기능성의 관점에서 상기 설명되어 있다. 이러한 기능성이 하드웨어 또는 소프트웨어 중 어느 것으로 구현되는지는 전체 시스템에 부과되는 특정 애플리케이션 및 설계 제약들에 달려있다. 당업자들은 설명된 기능을 각각의 특정한 애플리케이션에 대하여 다양한 방식들로 구현할 수도 있지만, 이러한 구현 결정은 본 발명의 범위를 벗어나게 하는 것으로 해석되지 않아야 한다.

[0287] 비제한적인 예로, 이러한 컴퓨터 판독가능 저장 매체들은 RAM, ROM, EEPROM, CD-ROM 또는 다른 광 디스크 저장, 자기 디스크 저장, 또는 다른 자기 저장 디바이스들, 플래시 메모리, 또는 소망의 프로그램 코드를 컴퓨터에 의해 액세스될 수 있는 명령들 또는 데이터 구조들의 형태로 저장하는데 사용될 수 있는 임의의 다른 매체를 포함할 수 있다. 또한, 임의의 접속이 컴퓨터 판독가능 매체로 적절히 칭해진다. 예를 들어, 명령들이 웹사이트, 서버, 또는 다른 원격 소스로부터 동축 케이블, 광섬유 케이블, 연선 (twisted pair), 디지털 가입자 회선 (DSL), 또는 무선 기술들 이를테면 적외선, 라디오, 및/또는 마이크로파를 이용하여 송신된다면, 동축 케이블, 광섬유 케이블, 연선, DSL, 또는 무선 기술들 이를테면 적외선, 라디오, 및 마이크로파는 매체의 정의에 포함된다. 그러나, 컴퓨터 판독가능 저장 매체들 및 데이터 저장 매체들은 커넥션들, 반송파들, 신호들, 또는 다른 일시적 매체들을 포함하지 않고, 그 대신 비일시적 (non-transient), 유형의 저장 매체들을 지향하고 있음이 이해되어야 한다. 디스크 (disk 및 disc) 는 본원에서 사용되는 바와 같이, 콤팩트 디스크 (compact disc; CD), 레이저 디스크, 광 디스크, 디지털 다용도 디스크 (DVD), 플로피 디스크 (floppy disk) 및 블루레이 디스크를 포함하는데, 디스크 (disk) 들은 보통 데이터를 자기적으로 재생하는 한편, 디스크 (disc) 들은 레이저들로 데이터를 광학적으로 재생한다. 상기한 것들의 조합들은 또한 컴퓨터 판독가능 매체들의 범위 내에 포함되어야 한다.

[0288] 명령들은 하나 이상의 프로세서들, 이를테면 하나 이상의 디지털 신호 프로세서들 (DSP들), 범용 마이크로프로세서들, 주문형 집적회로들 (ASIC들), 필드 프로그램가능 로직 어레이들 (FPGA들), 또는 다른 동등한 집적 또는 개별 로직 회로에 의해 실행될 수도 있다. 이에 따라, 본원에서 사용되는 바와 같은 용어 "프로세서"는 앞

서의 구조 또는 본원에서 설명된 기법들의 구현에 적합한 임의의 다른 구조 중 임의의 것을 말할 수도 있다. 덧붙여서, 일부 양태들에서, 본원에서 설명된 기능성은 인코딩 및 디코딩을 위해 구성되는, 또는 결합형 코덱 (codec) 으로 통합되는 전용 하드웨어 및/또는 소프트웨어 모듈들 내에 제공될 수도 있다. 또한, 본 기법들은 하나 이상의 회로들 또는 로직 엘리먼트들 내에 완전히 구현될 수 있다.

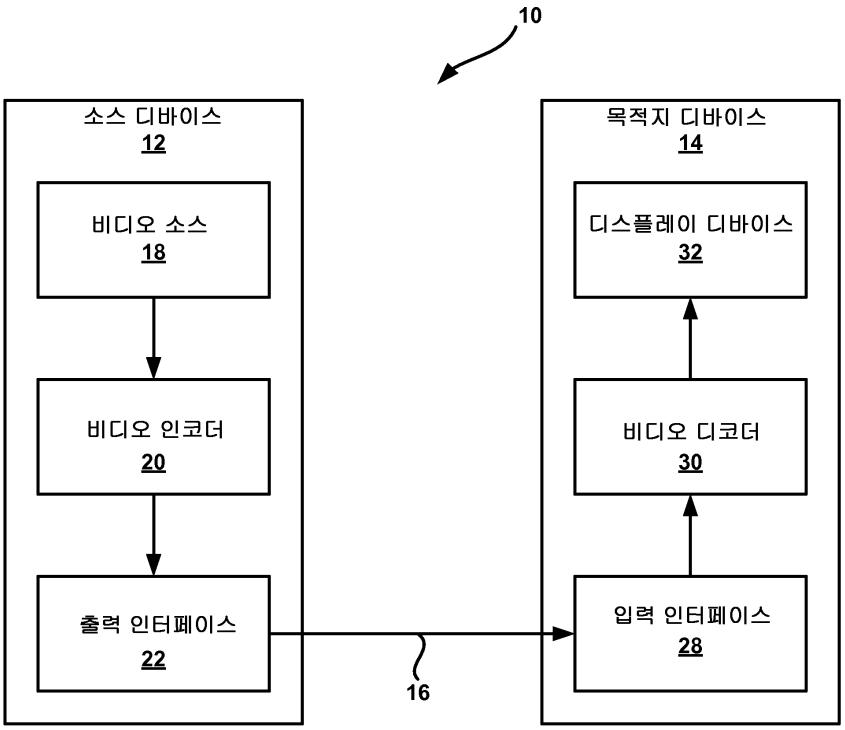
[0289] 본 개시물의 기법들은 무선 핸드셋, 집적회로 (IC) 또는 IC들의 세트 (예컨대, 칩 세트) 를 포함한 매우 다양한 디바이스들 또는 장치들에서 구현될 수도 있다. 다양한 컴포넌트들, 모듈들, 또는 유닛들은 개시된 기법들을 수행하도록 구성된 디바이스들의 기능적 양태들을 강조하기 위해 본 개시물에서 설명되지만, 상이한 하드웨어 유닛들에 의한 실현을 반드시 요구하는 것은 아니다. 오히려, 위에서 설명된 바와 같이, 다양한 유닛들은 코덱 하드웨어 유닛에 결합되거나 또는 적합한 소프트웨어 및/또는 펌웨어와 함께, 위에서 설명된 바와 같은 하나 이상의 프로세서들을 포함하는, 상호운용적 하드웨어 유닛들의 컬렉션에 의해 제공될 수도 있다.

[0290] 본원에 개시된 실시형태들에 관련하여 설명된 방법 또는 알고리즘의 단계들은 직접 하드웨어에서, 프로세서에 의해 실행되는 소프트웨어 모듈에서, 또는 이들 두 가지의 조합에서 실시될 수도 있다. 소프트웨어 모듈은 RAM 메모리, 플래시 메모리, ROM 메모리, EPROM 메모리, EEPROM 메모리, 레지스터들, 하드 디스크, 착탈식 디스크, CD-ROM 또는 당업계에 알려진 임의의 다른 형태의 저장 매체에 상주할 수도 있다. 예시적인 저장 매체는 프로세서와 커플링되어 프로세서는 저장 매체로부터 정보를 읽을 수 있고 그 저장 매체에 정보를 쓸 수 있다. 대체예에서, 저장 매체는 프로세서에 통합될 수도 있다. 프로세서와 저장 매체는 ASIC 에 상주할 수도 있다. ASIC은 사용자 단말에 상주할 수도 있다. 대체예에서, 프로세서와 저장 매체는 사용자 단말에 개별 컴포넌트들로서 존재할 수도 있다.

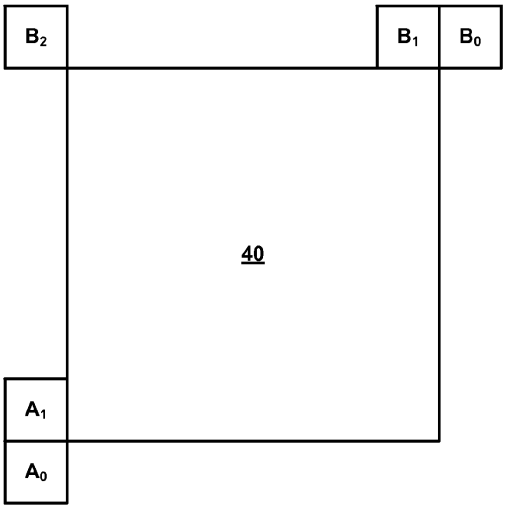
[0291] 다양한 예들이 설명되어 있다. 이들 및 다른 예들은 다음의 청구항들의 범위 내에 있다.

도면

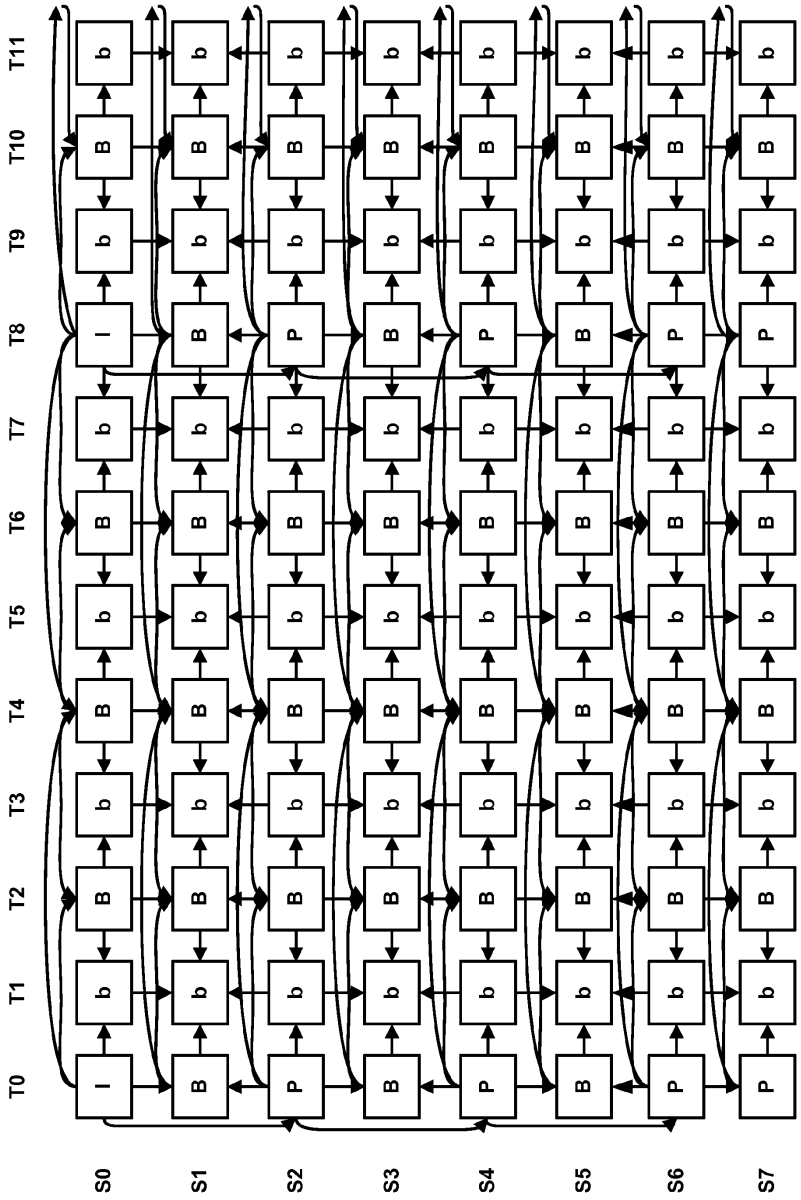
도면1



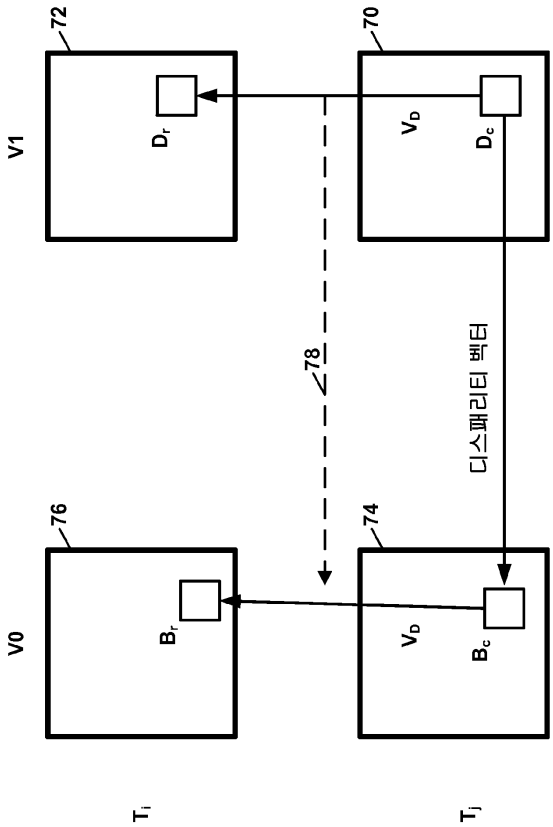
도면2



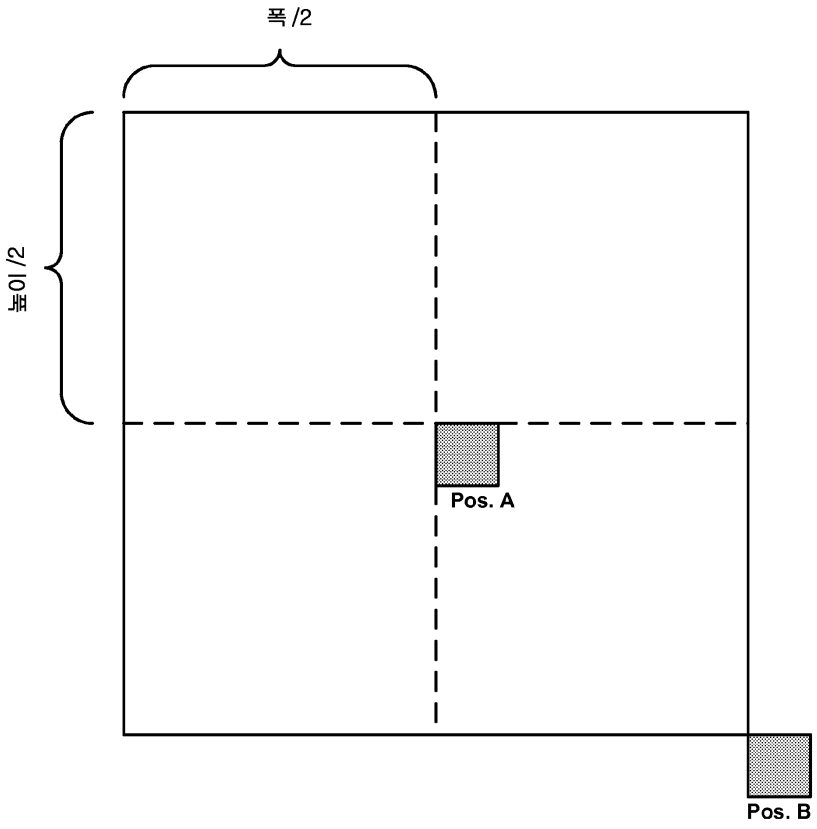
도면3



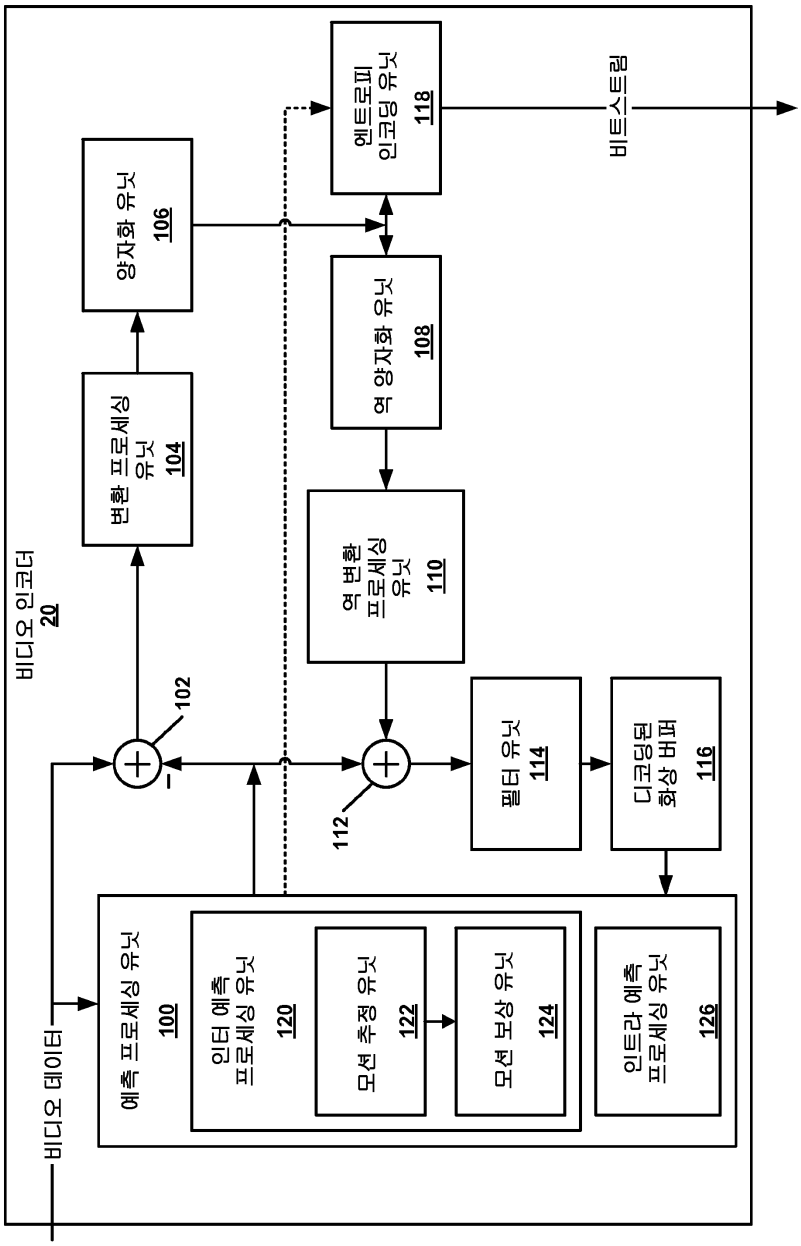
도면4


72
70
V1
D_r
V_D
D_c
78
디스패리티 벡터
76
74
V0
B_r
V_D
B_c
T_i
T_j

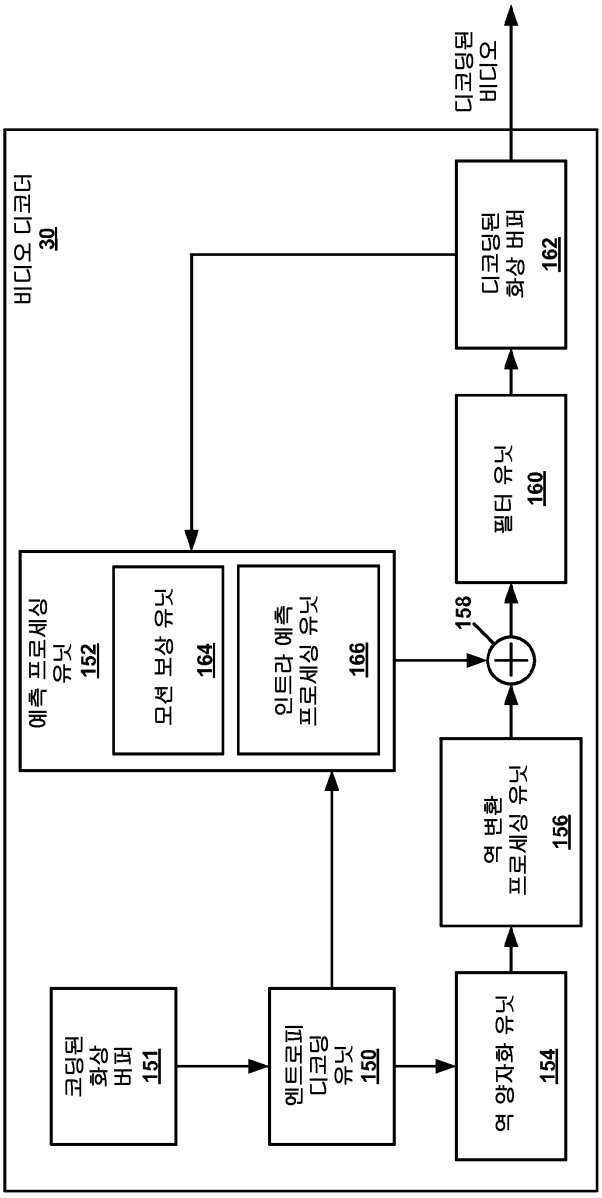
도면5


폭 /2
높이 /2
Pos. A
Pos. B

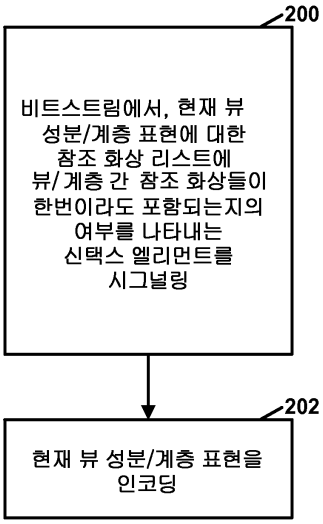
도면6


비디오 인코더
20
비디오 데이터
예측 프로세싱 유닛
100
인터 예측
프로세싱 유닛
120
모션 추정 유닛
122
모션 보상 유닛
124
인트라 예측
프로세싱 유닛
126
102
변환 프로세싱
유닛
104
양자화 유닛
106
역 양자화 유닛
108
역 변환
프로세싱
유닛
110
112
필터 유닛
114
디코딩된
화상 버퍼
116
엔트로피
인코딩 유닛
118
비트스트림

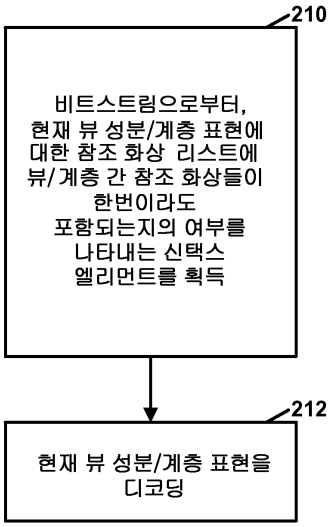
도면7


비디오 디코더
30
코딩된 화상 버퍼
151
엔트로피 디코딩 유닛
150
역 양자화 유닛
154
역 변환 프로세싱 유닛
156
158
예측 프로세싱 유닛
152
모션 보상 유닛
164
인트라 예측 프로세싱 유닛
166
필터 유닛
160
디코딩된 화상 버퍼
162
디코딩된 비디오

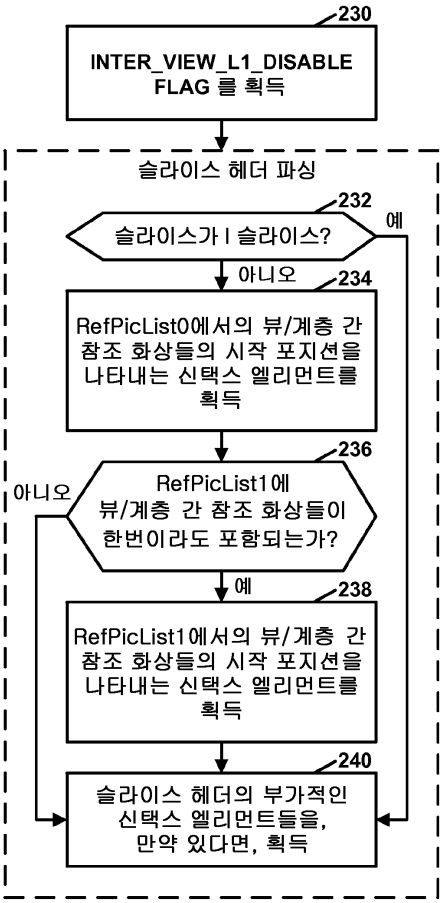
도면8a


200
비트스트림에서, 현재 뷰 성분/계층 표현에 대한 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번이라도 포함되는지의 여부를 나타내는 신택스 엘리먼트를 시그널링
202
현재 뷰 성분/계층 표현을 인코딩

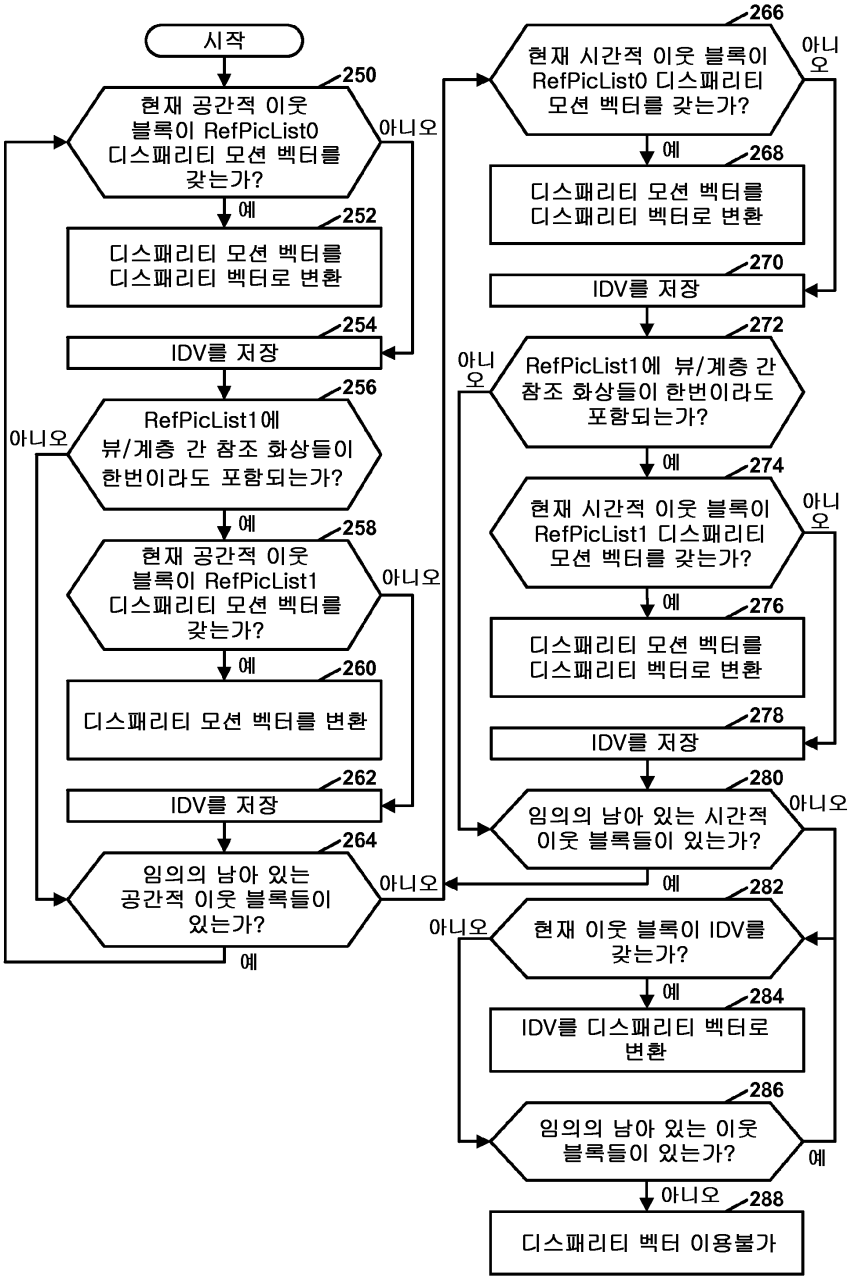
도면8b


210
비트스트림으로부터, 현재 뷰 성분/계층 표현에 대한 참조 화상 리스트에 뷰/계층 간 참조 화상들이 한번이라도 포함되는지의 여부를 나타내는 신택스 엘리먼트를 획득
212
현재 뷰 성분/계층 표현을 디코딩

도면9


230
INTER_VIEW_L1_DISABLE FLAG 를 획득
슬라이스 헤더 파싱
232
슬라이스가 I 슬라이스?
예
아니오
234
RefPicList0에서의 뷰/계층 간 참조 화상들의 시작 포지션을 나타내는 신택스 엘리먼트를 획득
236
RefPicList1에 뷰/계층 간 참조 화상들이 한번이라도 포함되는가?
아니오
예
238
RefPicList1에서의 뷰/계층 간 참조 화상들의 시작 포지션을 나타내는 신택스 엘리먼트를 획득
240
슬라이스 헤더의 부가적인 신택스 엘리먼트들을, 만약 있다면, 획득

도면10


시작
250
현재 공간적 이웃 블록이 RefPicList0 디스패리티 모션 벡터를 갖는가?
아니오
예
252
디스패리티 모션 벡터를 디스패리티 벡터로 변환
254
IDV를 저장
256
RefPicList1에 뷰/계층 간 참조 화상들이 한번이라도 포함되는가?
아니오
예
258
현재 공간적 이웃 블록이 RefPicList1 디스패리티 모션 벡터를 갖는가?
아니오
예
260
디스패리티 모션 벡터를 변환
262
IDV를 저장
264
임의의 남아 있는 공간적 이웃 블록들이 있는가?
아니오
예
266
현재 시간적 이웃 블록이 RefPicList0 디스패리티 모션 벡터를 갖는가?
아니오
예
268
디스패리티 모션 벡터를 디스패리티 벡터로 변환
270
IDV를 저장
272
RefPicList1에 뷰/계층 간 참조 화상들이 한번이라도 포함되는가?
아니오
예
274
현재 시간적 이웃 블록이 RefPicList1 디스패리티 모션 벡터를 갖는가?
아니오
예
276
디스패리티 모션 벡터를 디스패리티 벡터로 변환
278
IDV를 저장
280
임의의 남아 있는 시간적 이웃 블록들이 있는가?
아니오
예
282
현재 이웃 블록이 IDV를 갖는가?
아니오
예
284
IDV를 디스패리티 벡터로 변환
286
임의의 남아 있는 이웃 블록들이 있는가?
예
아니오
288
디스패리티 벡터 이용불가