(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2021년01월12일
(11) 등록번호 10-2202162
(24) 등록일자 2021년01월07일

(51) 국제특허분류(Int. Cl.)
H04N 19/597 (2014.01) *H04N 13/00* (2018.01)
(21) 출원번호 10-2014-7031679
(22) 출원일자(국제) 2013년04월02일
심사청구일자 2018년03월19일
(85) 번역문제출일자 2014년11월11일
(65) 공개번호 10-2015-0008402
(43) 공개일자 2015년01월22일
(86) 국제출원번호 PCT/US2013/034992
(87) 국제공개번호 WO 2013/154869
국제공개일자 2013년10월17일
(30) 우선권주장
13/839,447 2013년03월15일 미국(US)
(뒷면에 계속)

(56) 선행기술조사문헌
Dong Tian et al. Mitsubishi Response to MPEG Call for Proposal on 3D Video Coding Technology. ISO/IEC JTC1/SC29/WG11 MPEG2011/M22663, 2011.11.24.*
(뒷면에 계속)

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

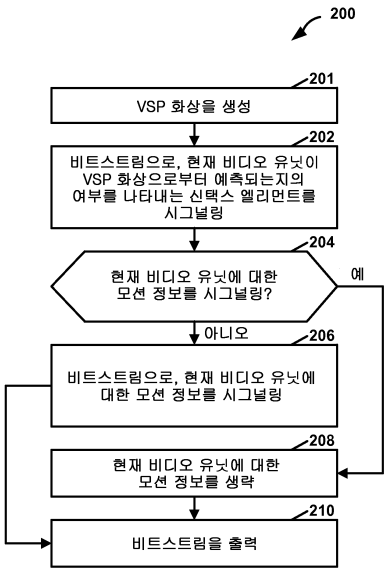
전체 청구항 수 : 총 12 항 심사관 : 이남숙

(54) 발명의 명칭 **3차원 비디오 코딩을 위한 뷰 합성 모드**

(57) 요 약

비디오 인코더는, 비트스트림으로, 현재 비디오 유닛이 VSP 화상으로부터 예측되는지의 여부를 나타내는 신택스 엘리먼트를 시그널링한다. 현재 비디오 유닛은 매크로블록 또는 매크로블록 파티션이다. 비디오 인코더는, 현재 비디오 유닛이 VSP 화상으로부터 예측되는지의 여부에 적어도 부분적으로 기초하여, 비트스트림으로 현재 비디오 유닛에 대한 모션 정보를 시그널링할지의 여부를 결정한다. 비디오 디코더는 그 비트스트림으로부터 신택스 엘리먼트를 디코딩하고, 신택스 엘리먼트에 적어도 부분적으로 기초하여, 그 비트스트림이 모션 정보를 포함하는지의 여부를 결정한다.

대 표 도 - 도4a



(72) 발명자

유 양

미국 92121 캘리포니아주 샌디에고 모어하우스 드라이브 5775

판 더 아우베라 게르트

미국 92121 캘리포니아주 샌디에고 모어하우스 드라이브 5775

리 시앙

미국 92121 캘리포니아주 샌디에고 모어하우스 드라이브 5775

카르체비츠 마르타

미국 92121 캘리포니아주 샌디에고 모어하우스 드라이브 5775

(56) 선행기술조사문헌

Advanced video coding for generic audiovisual services. ITU-T Recommendation H.264. 2009.03.*

Shinya Shimizu, et al. Improved view synthesis prediction using decoder-side motion derivation for multiview video coding. IEEE 3DTV-Conference. 2010.*

US20070030356 A1

US20120062756 A1

*는 심사관에 의하여 인용된 문헌

(30) 우선권주장

61/624,157 2012년04월13일 미국(US)

61/635,761 2012년04월19일 미국(US)

61/639,064 2012년04월26일 미국(US)

61/639,845 2012년04월27일 미국(US)

61/642,379 2012년05월03일 미국(US)

61/646,810 2012년05월14일 미국(US)

61/696,661 2012년09월04일 미국(US)

명 세 서

청구범위

청구항 1

예측적으로 인코딩된 3D 비디오 데이터를 디코딩하는 방법으로서,

현재 비디오 유닛이 제 1 모드를 사용하여 코딩되는지의 여부를 결정하는 단계로서, 상기 현재 비디오 유닛은 현재 액세스 유닛의 현재 뷰의 현재 텍스처 뷰 성분 (texture view component) 의 매크로블록, MB, 또는 MB 파티션인, 상기 현재 비디오 유닛이 제 1 모드를 사용하여 코딩되는지의 여부를 결정하는 단계;

상기 현재 비디오 유닛이 상기 제 1 모드를 사용하여 코딩되는 경우:

뷰 합성 예측을 위한 참조 뷰 화상을 결정하는 단계로서, 상기 참조 뷰 화상은 뷰 합성을 위한 뷰인 것으로서 슬라이스 헤더에 나타낸 뷰에 있는 상기 현재 액세스 유닛에 있는 화상인, 상기 뷰 합성 예측을 위한 참조 뷰 화상을 결정하는 단계;

상기 현재 비디오 유닛에 대한 참조 인덱스를, 상기 참조 인덱스가 상기 참조 뷰 화상을 나타내도록 설정하는 단계; 및

상기 현재 비디오 유닛에 대한 모션 벡터를 도출하는 단계로서, 상기 현재 비디오 유닛에 대한 모션 벡터는 상기 참조 뷰 화상을 참조하고, 상기 모션 벡터를 도출하는 단계는:

깊이 화상의 블록으로부터 대표 깊이 값을 결정하는 단계로서, 상기 깊이 화상의 블록은 대응 텍스처 화상의 현재 비디오 유닛에 대응하고, 상기 대표 깊이 값은 상기 깊이 화상의 블록에 있는 미리 결정된 깊이 화소들의 최대값, 평균값 또는 중간값에 기초하여 결정되고, 상기 미리 결정된 깊이 화소들은 상기 깊이 화상의 블록에 있는 코너 깊이 화소들인, 상기 대표 깊이 값을 결정하는 단계;

디스패리티 벡터를 결정하기 위해 상기 대표 깊이 값을 사용하는 단계; 및

상기 현재 비디오 유닛에 대한 상기 모션 벡터를 상기 디스패리티 벡터와 동일하게 설정하는 단계를 포함하는, 상기 현재 비디오 유닛에 대한 모션 벡터를 도출하는 단계; 또는

상기 현재 비디오 유닛이 제 2 모드를 사용하여 코딩되는 경우:

비트스트림으로부터, 상기 현재 비디오 유닛에 대한 모션 정보를 디코딩하는 단계; 및

상기 현재 비디오 유닛의 모션 정보에 의해 나타낸 참조 블록에 기초하여 상기 현재 비디오 유닛에 대한 예측 블록을 생성하는 단계로서, 상기 현재 비디오 유닛의 모션 정보는 상기 현재 비디오 유닛에 대한 모션 벡터 및 상기 현재 비디오 유닛에 대한 참조 인덱스를 포함하는, 상기 예측 블록을 생성하는 단계; 및

상기 현재 비디오 유닛의 샘플 블록을 구성하기 위해 잔차 블록에 상기 예측 블록을 더하는 단계를 포함하는, 비디오 데이터를 디코딩하는 방법.

청구항 2

제 1 항에 있어서,

상기 제 1 모드는, 3D-AVC 또는 3D-HEVC 에 있는 다음의 신택스 엘리먼트들: sub_mb_vsp_flag, mb_part_vsp_flag, 및 vsp_mb_flag 중 하나에 의해 식별되는, 비디오 데이터를 디코딩하는 방법.

청구항 3

제 1 항에 있어서,

상기 현재 비디오 유닛이 상기 제 1 모드를 사용하여 코딩되는지의 여부를 결정하는 단계는, 상기 현재 비디오 유닛에 대한 시그널링된 참조 인덱스가 특정 값을 갖는 경우, 상기 현재 비디오 유닛이 상기 제 1 모드를 사용하여 코딩된다고 결정하는 단계를 포함하는, 비디오 데이터를 디코딩하는 방법.

청구항 4

제 1 항에 있어서,

상기 현재 비디오 유닛에 대한 참조 인덱스를 설정하는 단계는, 0 에서부터 참조 화상 리스트 RefPicList0 에서의 액티브 참조 화상들의 수까지의 i 각각에 대해, RefPicList0[i] 가 상기 뷰 합성 예측을 위한 참조 뷰 화상과 동일한 경우, 상기 참조 인덱스를 i 로 설정하는 단계를 포함하는, 비디오 데이터를 디코딩하는 방법.

청구항 5

예측적으로 인코딩된 3D 비디오 데이터를 디코딩하는 3D 비디오 디코딩 디바이스 (30) 로서,

현재 비디오 유닛이 제 1 모드를 사용하여 코딩되는지의 여부를 결정하는 수단으로서, 상기 현재 비디오 유닛은 현재 액세스 유닛의 현재 뷰의 현재 텍스처 뷰 성분의 매크로블록, MB, 또는 MB 파티션인, 상기 현재 비디오 유닛이 제 1 모드를 사용하여 코딩되는지의 여부를 결정하는 수단;

상기 현재 비디오 유닛이 상기 제 1 모드를 사용하여 코딩되는 경우, 뷰 합성 예측을 위한 참조 뷰 화상을 결정하는 수단으로서, 상기 참조 뷰 화상은 뷰 합성을 위한 뷰인 것으로서 슬라이스 헤더에 나타낸 뷰에 있는 상기 현재 액세스 유닛에 있는 화상인, 상기 뷰 합성 예측을 위한 참조 뷰 화상을 결정하는 수단;

상기 현재 비디오 유닛이 상기 제 1 모드를 사용하여 코딩되는 경우, 상기 현재 비디오 유닛에 대한 참조 인덱스를, 상기 참조 인덱스가 상기 참조 뷰 화상을 나타내도록 설정하는 수단;

상기 현재 비디오 유닛이 상기 제 1 모드를 사용하여 코딩되는 경우, 상기 현재 비디오 유닛에 대한 모션 벡터를 도출하는 수단으로서, 상기 현재 비디오 유닛에 대한 모션 벡터는 상기 참조 뷰 화상을 참조하고, 상기 모션 벡터를 도출하는 것은:

깊이 화상의 블록으로부터 대표 깊이 값을 결정하는 것으로서, 상기 깊이 화상의 블록은 대응 텍스처 화상의 현재 비디오 유닛에 대응하고, 상기 대표 깊이 값은 상기 깊이 화상의 블록에 있는 미리 결정된 깊이 화소들의 최대값, 평균값 또는 중간값에 기초하여 결정되고, 상기 미리 결정된 깊이 화소들은 상기 깊이 화상의 블록에 있는 코너 깊이 화소들인, 상기 대표 깊이 값을 결정하는 것;

디스패리티 벡터를 결정하기 위해 상기 대표 깊이 값을 사용하는 것; 및

상기 현재 비디오 유닛에 대한 상기 모션 벡터를 상기 디스패리티 벡터와 동일하게 설정하는 것을 포함하는, 상기 현재 비디오 유닛에 대한 모션 벡터를 도출하는 수단;

상기 현재 비디오 유닛이 제 2 모드를 사용하여 코딩되는 경우, 비트스트림으로부터, 상기 현재 비디오 유닛에 대한 모션 정보를 디코딩하는 수단;

상기 현재 비디오 유닛의 모션 정보에 의해 나타낸 참조 블록에 기초하여 상기 현재 비디오 유닛에 대한 예측 블록을 생성하는 수단으로서, 상기 현재 비디오 유닛의 모션 정보는 상기 현재 비디오 유닛에 대한 모션 벡터 및 상기 현재 비디오 유닛에 대한 참조 인덱스를 포함하는, 상기 예측 블록을 생성하는 수단; 및

상기 현재 비디오 유닛의 샘플 블록을 구성하기 위해 잔차 블록에 상기 예측 블록을 더하는 수단을 포함하는, 비디오 디코딩 디바이스.

청구항 6

명령들을 저장하고 있는 컴퓨터 판독가능 저장 매체로서, 상기 명령들은, 비디오 디코딩 디바이스의 하나 이상의 프로세서들에 의해 실행되는 경우, 제 1 항 내지 제 4 항 중 어느 한 항의 방법을 수행하도록 상기 비디오 디코딩 디바이스를 구성하는, 컴퓨터 판독가능 저장 매체.

청구항 7

3D 비디오 데이터를 인코딩하는 방법으로서,

다수의 텍스처 뷰들 및 다수의 깊이 뷰들의 인코딩된 표현을 포함하는 비트스트림으로, 현재 비디오 유닛이 제 1 모드를 사용하여 코딩되는지의 여부를 시그널링하는 단계로서, 상기 현재 비디오 유닛은 현재 액세스 유닛의 현재 뷰의 현재 텍스처 뷰 성분의 매크로블록, MB, 또는 MB 파티션인, 상기 현재 비디오 유닛이 제 1 모드를 사

용하여 코딩되는지의 여부를 시그널링하는 단계;

상기 현재 비디오 유닛이 제 2 모드를 사용하여 코딩되고 상기 제 1 모드를 사용하여 코딩되지 않는 경우, 상기 비트스트림으로, 상기 현재 비디오 유닛에 대한 참조 인덱스 및 상기 현재 비디오 유닛에 대한 모션 정보를 시그널링하는 단계;

상기 현재 비디오 유닛이 상기 제 1 모드를 사용하여 코딩되는 경우:

뷰 합성 예측을 위한 참조 뷰 화상을 결정하는 단계로서, 상기 참조 뷰 화상은 뷰 합성을 위한 뷰인 것으로서 슬라이스 헤더에 나타낸 뷰에 있는 상기 현재 액세스 유닛에 있는 화상인, 상기 뷰 합성 예측을 위한 참조 뷰 화상을 결정하는 단계;

상기 현재 비디오 유닛에 대한 참조 인덱스를, 상기 참조 인덱스가 상기 참조 뷰 화상을 나타내도록 설정하는 단계;

상기 현재 비디오 유닛에 대한 모션 벡터를 도출하는 단계로서, 상기 현재 비디오 유닛에 대한 모션 벡터는 상기 참조 뷰 화상을 참조하고, 상기 모션 벡터를 도출하는 단계는:

깊이 화상의 블록으로부터 대표 깊이 값을 결정하는 단계로서, 상기 깊이 화상의 블록은 대응 텍스처 화상의 현재 비디오 유닛에 대응하고, 상기 대표 깊이 값은 상기 깊이 화상의 블록에 있는 미리 결정된 깊이 화소들의 최대값, 평균값 또는 중간값에 기초하여 결정되고, 상기 미리 결정된 깊이 화소들은 상기 깊이 화상의 블록에 있는 코너 깊이 화소들인, 상기 대표 깊이 값을 결정하는 단계;

디스패리티 벡터를 결정하기 위해 상기 대표 깊이 값을 사용하는 단계; 및

상기 현재 비디오 유닛에 대한 상기 모션 벡터를 상기 디스패리티 벡터와 동일하게 설정하는 단계를 포함하는, 상기 현재 비디오 유닛에 대한 모션 벡터를 도출하는 단계; 및

상기 비트스트림으로부터, 상기 현재 비디오 유닛에 대한 모션 벡터를 생략하는 단계 (208); 및

상기 비트스트림을 출력하는 단계를 포함하는, 비디오 데이터를 인코딩하는 방법.

청구항 8

제 7 항에 있어서,

제 1 VSP 모드는, 3D-AVC 또는 3D-HEVC 에 있는 다음의 신택스 엘리먼트들: sub_mb_vsp_flag, mb_part_vsp_flag, 및 vsp_mb_flag 중 하나에 의해 식별되는, 비디오 데이터를 인코딩하는 방법.

청구항 9

제 7 항에 있어서,

상기 현재 비디오 유닛이 제 1 VSP 모드를 사용하여 코딩되는지의 여부를 시그널링하는 단계는, 상기 비트스트림으로, 상기 현재 비디오 유닛에 대한 참조 인덱스가 특정 값을 갖는다는 것을 시그널링하는 단계를 포함하는, 비디오 데이터를 인코딩하는 방법.

청구항 10

제 7 항에 있어서,

상기 현재 비디오 유닛에 대한 참조 인덱스를 설정하는 단계는, 0 에서부터 참조 화상 리스트 RefPicList0 에서의 액티브 참조 화상들의 수까지의 i 각각에 대해, RefPicList0[i] 가 상기 뷰 합성 예측을 위한 참조 뷰 화상과 동일한 경우, 상기 참조 인덱스를 i 로 설정하는 단계를 포함하는, 비디오 데이터를 인코딩하는 방법.

청구항 11

3D 비디오 인코딩 디바이스 (20) 로서,

다수의 텍스처 뷰들 및 다수의 깊이 뷰들의 인코딩된 표현을 포함하는 비트스트림으로, 현재 비디오 유닛이 제 1 모드를 사용하여 코딩되는지의 여부를 시그널링하는 수단으로서, 상기 현재 비디오 유닛은 현재 액세스 유닛의 현재 뷰의 현재 텍스처 뷰 성분의 매크로블록, MB, 또는 MB 파티션인, 상기 현재 비디오 유닛이 제 1 모드를

사용하여 코딩되는지의 여부를 시그널링하는 수단;

상기 현재 비디오 유닛이 제 2 모드를 사용하여 코딩되고, 상기 제 1 모드를 사용하여 코딩되지 않는 경우, 상기 비트스트림으로, 상기 현재 비디오 유닛에 대한 모션 정보를 시그널링하는 수단;

상기 현재 비디오 유닛이 상기 제 1 모드를 사용하여 코딩되는 경우, 뷰 합성 예측을 위한 참조 뷰 화상을 결정하는 수단으로서, 상기 참조 뷰 화상은 뷰 합성을 위한 뷰인 것으로서 슬라이스 헤더에 나타낸 뷰에 있는 상기 현재 액세스 유닛에 있는 화상인, 상기 뷰 합성 예측을 위한 참조 뷰 화상을 결정하는 수단;

상기 현재 비디오 유닛이 상기 제 1 모드를 사용하여 코딩되는 경우, 상기 현재 비디오 유닛에 대한 참조 인덱스를, 상기 참조 인덱스가 상기 참조 뷰 화상을 나타내도록 설정하는 수단;

상기 현재 비디오 유닛이 상기 제 1 모드를 사용하여 코딩되는 경우, 상기 현재 비디오 유닛에 대한 모션 벡터를 도출하는 수단으로서, 상기 현재 비디오 유닛에 대한 모션 벡터는 상기 참조 뷰 화상을 참조하고, 상기 모션 벡터를 도출하는 것은:

깊이 화상의 블록으로부터 대표 깊이 값을 결정하는 것으로서, 상기 깊이 화상의 블록은 대응 텍스처 화상의 현재 비디오 유닛에 대응하고, 상기 대표 깊이 값은 상기 깊이 화상의 블록에 있는 미리 결정된 깊이 화소들의 최대값, 평균값 또는 중간값에 기초하여 결정되고, 상기 미리 결정된 깊이 화소들은 상기 깊이 화상의 블록에 있는 코너 깊이 화소들인, 상기 대표 깊이 값을 결정하는 것;

디스패리티 벡터를 결정하기 위해 상기 대표 깊이 값을 사용하는 것; 및

상기 현재 비디오 유닛에 대한 상기 모션 벡터의 값을 상기 디스패리티 벡터의 값으로 설정하는 것을 포함하는, 상기 현재 비디오 유닛에 대한 모션 벡터를 도출하는 수단; 및

상기 현재 비디오 유닛이 단일화된 VSP 모드를 사용하여 코딩되는 경우, 상기 비트스트림으로부터, 상기 현재 비디오 유닛에 대한 모션 정보를 생략하는 수단; 및

상기 비트스트림을 출력하는 수단을 포함하는, 비디오 인코딩 디바이스.

청구항 12

명령들을 저장하고 있는 컴퓨터 판독가능 저장 매체로서, 상기 명령들은, 비디오 인코딩 디바이스의 하나 이상의 프로세서들에 의해 실행되는 경우, 제 7 항 내지 제 10 항 중 어느 한 항의 방법을 수행하도록 상기 비디오 인코딩 디바이스를 구성하는, 컴퓨터 판독가능 저장 매체.

청구항 13

삭제

청구항 14

삭제

청구항 15

삭제

청구항 16

삭제

청구항 17

삭제

청구항 18

삭제

청구항 19

삭제

청구항 20

삭제

청구항 21

삭제

청구항 22

삭제

청구항 23

삭제

청구항 24

삭제

청구항 25

삭제

청구항 26

삭제

청구항 27

삭제

청구항 28

삭제

청구항 29

삭제

청구항 30

삭제

청구항 31

삭제

청구항 32

삭제

청구항 33

삭제

청구항 34

삭제

청구항 35

삭제

청구항 36

삭제

청구항 37

삭제

청구항 38

삭제

청구항 39

삭제

청구항 40

삭제

청구항 41

삭제

청구항 42

삭제

청구항 43

삭제

청구항 44

삭제

청구항 45

삭제

청구항 46

삭제

청구항 47

삭제

청구항 48

삭제

청구항 49

삭제

청구항 50

삭제

청구항 51

삭제

청구항 52

삭제

발명의 설명

기 술 분 야

[0001] 본 출원은 2012년 4월 13일자로 출원된 미국 가특허출원 제61/624,157호, 2012년 4월 19일자로 출원된 미국 가특허출원 제61/635,761호, 2012년 4월 26일자로 출원된 미국 가특허출원 제61/639,064호, 2012년 4월 27일자로 출원된 미국 가특허출원 제61/639,845호, 2012년 5월 3일자로 출원된 미국 가특허출원 제61/642,379호, 2012년 5월 14일자로 출원된 미국 가특허출원 제61/646,810호, 및 2012년 9월 4일자로 출원된 미국 가특허출원 제61/696,661호를 우선권 주장하며, 그것들의 각각의 전체 내용은 참조로 본원에 통합된다.

[0002] 기술 분야

[0003] 본 개시물은 비디오 코딩 (즉, 비디오 데이터의 인코딩 및/또는 디코딩) 에 관한 것이다.

배 경 기 술

[0004] 디지털 비디오 능력들은 디지털 텔레비전들, 디지털 직접 브로드캐스트 시스템들, 무선 브로드캐스트 시스템들, 개인휴대 정보단말들 (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 포함하는 넓은 범위의 디바이스들에 통합될 수 있다. 디지털 비디오 디바이스들은 MPEG-2, MPEG-4, ITU-T H.263 또는 ITU-T H.264/MPEG-4, 파트 10, 고급 비디오 코딩 (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 그런 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 (인트라-화상) 예측 및/또는 시간적 (인터-화상) 예측을 수행하여 비디오 시퀀스들에 내재하는 리던던시를 감소시키거나 제거한다. 블록 기반 비디오 코딩의 경우, 비디오 슬라이스 (즉, 비디오 프레임 또는 비디오 프레임의 부분) 는 비디오 블록들로 파티셔닝될 수도 있다. 화상의 인트라 코딩식 (intra-coded; I) 슬라이스에서의 비디오 블록들은 동일한 화상의 이웃 블록들에서의 참조 샘플들에 관한 공간적 예측을 이용하여 인코딩된다. 화상의 인터 코딩식 (inter-coded; P 또는 B) 슬라이스에서의 비디오 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 블록으로 나타나게 된다. 잔차 데이터는 코딩될 원본 블록과 예측 블록 사이의 화소 차이들을 나타낸다. 인터 코딩식 블록은 예측 블록을 형성하는 참조 샘플들의 블록을 가리키는 모션 벡터에 따라 인코딩되고, 잔차 데이터는 코딩된 블록 및 예측 블록 사이의 차이를 나타낸다. 인트라 코딩식 블록은 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 계수들은 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] 멀티-뷰 비트스트림이, 예컨대, 다수의 컬러 카메라들로부터의 뷰들을 인코딩함으로써 생성될 수도 있다. 멀티-뷰 비디오의 유연성을 추가로 확장하기 위해, 3차원 (3D) 비디오 표준들이 개발되어왔다. 3D 비디오 비트스트림은, 다수의 카메라들에 대응하는 뷰들, 즉 텍스처 뷰들뿐만 아니라, 적어도 하나 이상의 텍스처 뷰들에 연관된 깊이 뷰들도 포함할 수도 있다. 예를 들어, 각각의 뷰는 하나의 텍스처 뷰 및 하나의 깊이 뷰로 구성될 수도 있다.

발명의 내용

해결하려는 과제

과제의 해결 수단

[0008] 대체로, 본 개시물은 비디오 유닛의 뷰 합성 예측 (view synthesis prediction; VSP) 모드를 시그널링하는 것을 설명한다. 더 구체적으로는, 비디오 인코더는, 비트스트림으로, 현재 비디오 유닛이 VSP 화상으로부터 예측되는지의 여부를 나타내는 신택스 엘리먼트를 시그널링한다. 현재 비디오 유닛은 매크로블록 또는 매크로블록 파티션 (partition) 일 수도 있다. 더욱이, 비디오 인코더는, 현재 비디오 유닛이 VSP 화상으로부터 예측되는지의 여부에 적어도 부분적으로 기초하여, 비트스트림으로 현재 비디오 유닛에 대한 모션 정보를 시그널링할지의 여부를 결정할 수도 있다.

[0009] 비디오 디코더는, 비트스트림으로부터, 현재 비디오 유닛이 VSP 화상으로부터 예측되는지의 여부를 나타내는 신택스 엘리먼트를 디코딩할 수도 있다. 현재 비디오 유닛이 VSP 화상으로부터 예측되는 경우, 비디오 디코더는, VSP 화상에 적어도 부분적으로 기초하여, 현재 비디오 유닛의 샘플 블록들을 복원할 수도 있다. 현재 비디오 유닛이 VSP 화상으로부터 예측되지 않는 경우, 비디오 디코더는, 비트스트림으로부터, 현재 비디오 유닛에 대한 모션 정보를 디코딩할 수도 있다. 비디오 디코더는 그 모션 정보를 사용하여 현재 비디오 유닛의 샘플 블록들을 복원할 수도 있다.

[0010] 하나의 예에서, 비디오 데이터를 디코딩하는 방법은, 현재 액세스 유닛의 이전에 코딩된 텍스처 뷰 성분 (texture view component) 및 현재 액세스 유닛의 깊이 뷰 성분에 적어도 부분적으로 기초하여, VSP 화상을 생성하는 단계를 포함한다. 덧붙여서, 그 방법은, 다수의 텍스처 뷰들 및 다수의 깊이 뷰들의 코딩된 표현을 포함하는 비트스트림으로부터, 현재 비디오 유닛이 VSP 화상으로부터 예측되는지의 여부를 나타내는 신택스 엘리먼트를 디코딩하는 단계를 포함하며, 현재 비디오 유닛은 현재 액세스 유닛의 현재 뷰의 현재 텍스처 뷰 성분의 매크로블록 (macroblock; MB) 또는 MB 파티션이다. 그 방법은 또한, 현재 비디오 유닛이 VSP 화상으로부터 예측되지 않는 경우, 비트스트림으로부터, 현재 비디오 유닛에 대한 모션 정보를 디코딩하는 단계 및 현재 비디오 유닛에 대한 모션 정보를 사용하여 현재 비디오 유닛의 샘플 블록들을 복원하는 단계를 포함한다. 덧붙여서, 그 방법은 현재 비디오 유닛이 VSP 화상으로부터 예측되는 경우, VSP 화상을 사용하여 현재 비디오 유닛의 샘플 블록들을 복원하는 단계를 포함한다.

[0011] 다른 예에서, 비디오 디코딩 디바이스는, 현재 액세스 유닛의 이전에 코딩된 텍스처 뷰 성분 및 현재 액세스 유닛의 깊이 뷰 성분에 적어도 부분적으로 기초하여, VSP 화상을 생성하도록 구성된 하나 이상의 프로세서들을 포함한다. 더욱이, 하나 이상의 명령들은, 다수의 텍스처 뷰들 및 다수의 깊이 뷰들의 코딩된 표현을 포함하는 비트스트림으로부터, 현재 비디오 유닛이 VSP 화상으로부터 예측되는지의 여부를 나타내는 신택스 엘리먼트를 디코딩하도록 구성되며, 현재 비디오 유닛은 현재 액세스 유닛의 현재 뷰의 현재 텍스처 뷰 성분의 MB 또는 MB 파티션이다. 하나 이상의 프로세서들은 또한, 현재 비디오 유닛이 VSP 화상으로부터 예측되지 않는 경우, 하나 이상의 프로세서들이, 비트스트림으로부터, 현재 비디오 유닛에 대한 모션 정보를 디코딩하고 현재 비디오 유닛에 대한 모션 정보를 사용하여 현재 비디오 유닛의 샘플 블록들을 복원하도록 구성된다. 하나 이상의 프로세서들은 또한, 현재 비디오 유닛이 VSP 화상으로부터 예측되는 경우, 하나 이상의 프로세서들은 VSP 화상를 사용하여 현재 비디오 유닛의 샘플 블록들을 복원하도록 구성된다.

[0012] 다른 예에서, 비디오 디코딩 디바이스는, 현재 액세스 유닛의 이전에 코딩된 텍스처 뷰 성분 및 현재 액세스 유닛의 깊이 뷰 성분에 적어도 부분적으로 기초하여, VSP 화상을 생성하는 수단을 포함한다. 덧붙여서, 비디오 디코딩 디바이스는, 다수의 텍스처 뷰들 및 다수의 깊이 뷰들의 코딩된 표현을 포함하는 비트스트림으로부터, 현재 비디오 유닛이 VSP 화상으로부터 예측되는지의 여부를 나타내는 신택스 엘리먼트를 디코딩하는 수단을 포함하며, 현재 비디오 유닛은 현재 액세스 유닛의 현재 뷰의 현재 텍스처 뷰 성분의 MB 또는 MB 파티션이다. 덧붙여서, 비디오 디코딩 디바이스는, 현재 비디오 유닛이 VSP 화상으로부터 예측되지 않는 경우, 비트스트림으로부터, 현재 비디오 유닛에 대한 모션 정보를 디코딩하는 수단을 포함한다. 비디오 디코딩 디바이스는 또한, 현재 비디오 유닛이 VSP 화상으로부터 예측되지 않는 경우, 현재 비디오 유닛에 대한 모션 정보를 사용하여 현재 비디오 유닛의 샘플 블록들을 복원하는 수단을 포함한다. 덧붙여서, 비디오 디코딩 디바이스는, 현재 비디오 유닛이 VSP 화상으로부터 예측되는 경우, VSP 화상을 사용하여 현재 비디오 유

닛의 샘플 블록들을 복원하는 수단을 포함한다.

[0013] 다른 예에서, 컴퓨터 판독가능 저장 매체는, 비디오 디코딩 디바이스의 하나 이상의 프로세서들에 의해 실행되는 경우, 현재 액세스 유닛의 이전에 코딩된 텍스처 뷰 성분 및 현재 액세스 유닛의 깊이 뷰 성분에 적어도 부분적으로 기초하여, VSP 화상을 생성하도록 비디오 디코딩 디바이스를 구성하는 명령들을 저장하고 있다. 그 명령들은 또한, 다수의 텍스처 뷰들 및 다수의 깊이 뷰들의 코딩된 표현을 포함하는 비트스트림으로부터, 현재 비디오 유닛이 VSP 화상으로부터 예측되는지의 여부를 나타내는 신택스 엘리먼트를 디코딩하도록 비디오 디코딩 디바이스를 구성하며, 현재 비디오 유닛은 현재 액세스 유닛의 현재 뷰의 현재 텍스처 뷰 성분의 MB 또는 MB 파티션이다. 그 명령들은 또한, 현재 비디오 유닛이 VSP 화상으로부터 예측되지 않는 경우, 비디오 디코딩 디바이스가, 비트스트림으로부터, 현재 비디오 유닛에 대한 모션 정보를 디코딩하고 현재 비디오 유닛에 대한 모션 정보를 사용하여 현재 비디오 유닛의 샘플 블록들을 복원하도록 비디오 디코딩 디바이스를 구성한다. 현재 비디오 유닛이 VSP 화상으로부터 예측되는 경우, 그 명령들은, VSP 화상을 사용하여 현재 비디오 유닛의 샘플 블록들을 복원하도록 비디오 디코딩 디바이스를 구성한다.

[0014] 다른 예에서, 비디오 데이터를 인코딩하는 방법은, 현재 액세스 유닛의 이전에 코딩된 텍스처 뷰 성분 및 현재 액세스 유닛의 깊이 뷰 성분에 적어도 부분적으로 기초하여, VSP 화상을 생성하는 단계를 포함한다. 더욱이, 그 방법은, 다수의 텍스처 뷰들 및 다수의 깊이 뷰들의 인코딩된 표현을 포함하는 비트스트림으로, 현재 비디오 유닛이 VSP 화상으로부터 예측되는지의 여부를 나타내는 신택스 엘리먼트를 시그널링하는 단계를 포함하며, 현재 비디오 유닛은 현재 액세스 유닛의 현재 뷰의 현재 텍스처 뷰 성분의 MB 또는 MB 파티션이다. 덧붙여서, 그 방법은, 현재 비디오 유닛이 VSP 화상으로부터 예측되지 않는 경우, 비트스트림으로, 현재 비디오 유닛에 대한 모션 정보를 시그널링하는 단계를 포함한다. 그 방법은 또한, 현재 비디오 유닛이 VSP 화상으로부터 예측되는 경우, 비트스트림으로부터, 현재 비디오 유닛에 대한 모션 정보를 생략하는 단계를 포함한다. 더욱이, 그 방법은, 그 비트스트림을 출력하는 단계를 포함한다.

[0015] 다른 예에서, 비디오 인코딩 디바이스는, 현재 액세스 유닛의 이전에 코딩된 텍스처 뷰 성분 및 현재 액세스 유닛의 깊이 뷰 성분에 적어도 부분적으로 기초하여, VSP 화상을 생성하도록 구성된 하나 이상의 프로세서들을 포함한다. 덧붙여서, 하나 이상의 프로세서들은, 다수의 텍스처 뷰들 및 다수의 깊이 뷰들의 인코딩된 표현을 포함하는 비트스트림으로, 현재 비디오 유닛이 VSP 화상으로부터 예측되는지의 여부를 나타내는 신택스 엘리먼트를 시그널링하도록 구성되며, 현재 비디오 유닛은 현재 액세스 유닛의 현재 뷰의 현재 텍스처 뷰 성분의 MB 또는 MB 파티션이다. 하나 이상의 프로세서들은, 현재 비디오 유닛이 VSP 화상으로부터 예측되지 않는 경우, 하나 이상의 프로세서들이, 비트스트림으로, 현재 비디오 유닛에 대한 모션 정보를 시그널링하도록 구성된다. 더욱이, 하나 이상의 프로세서들은 현재 비디오 유닛이 VSP 화상으로부터 예측되는 경우, 하나 이상의 프로세서들이, 비트스트림으로부터, 현재 비디오 유닛에 대한 모션 정보를 생략하도록 구성된다. 하나 이상의 프로세서들은 또한 그 비트스트림을 출력하도록 구성된다.

[0016] 다른 예에서, 비디오 인코딩 디바이스는, 현재 액세스 유닛의 이전에 코딩된 텍스처 뷰 성분 및 현재 액세스 유닛의 깊이 뷰 성분에 적어도 부분적으로 기초하여, VSP 화상을 생성하는 수단을 포함한다. 덧붙여서, 비디오 인코딩 디바이스는, 다수의 텍스처 뷰들 및 다수의 깊이 뷰들의 인코딩된 표현을 포함하는 비트스트림으로, 현재 비디오 유닛이 VSP 화상으로부터 예측되는지의 여부를 나타내는 신택스 엘리먼트를 시그널링하는 수단을 포함하며, 현재 비디오 유닛은 현재 액세스 유닛의 현재 뷰의 현재 텍스처 뷰 성분의 MB 또는 MB 파티션이다. 비디오 인코딩 디바이스는 또한, 현재 비디오 유닛이 VSP 화상으로부터 예측되지 않는 경우, 비트스트림으로, 현재 비디오 유닛에 대한 모션 정보를 시그널링하는 수단을 포함한다. 더욱이, 비디오 인코딩 디바이스는, 현재 비디오 유닛이 VSP 화상으로부터 예측되는 경우, 비트스트림으로부터, 현재 비디오 유닛에 대한 모션 정보를 생략하는 수단을 포함한다. 덧붙여서, 비디오 인코딩 디바이스는 그 비트스트림을 출력하는 수단을 포함한다.

[0017] 다른 예에서, 컴퓨터 판독가능 저장 매체는, 비디오 인코딩 디바이스의 하나 이상의 프로세서들에 의해 실행되는 경우, 현재 액세스 유닛의 이전에 코딩된 텍스처 뷰 성분 및 현재 액세스 유닛의 깊이 뷰 성분에 적어도 부분적으로 기초하여, VSP 화상을 생성하도록 비디오 인코딩 디바이스를 구성하는 명령들을 저장하고 있다. 그 명령들은 또한, 비디오 인코딩 디바이스로 하여금, 다수의 텍스처 뷰들 및 다수의 깊이 뷰들의 인코딩된 표현을 포함하는 비트스트림으로, 현재 비디오 유닛이 VSP 화상으로부터 예측되는지의 여부를 나타내는 신택스 엘리먼트를 시그널링하게 하며, 현재 비디오 유닛은 현재 액세스 유닛의 현재 뷰의 현재 텍스처 뷰 성분의 MB 또는 MB 파티션이다. 현재 비디오 유닛이 VSP 화상으로부터 예측되지 않는 경우, 그 명령들은, 비트스트림으로, 현재 비디오 유닛에 대한 모션 정보를 시그널링하도록 비디오 인코딩 디바이스를 구성한다. 더욱이, 현

재 비디오 유닛이 VSP 화상으로부터 예측되는 경우, 그 명령들은, 비트스트림으로부터, 현재 비디오 유닛에 대한 모션 정보를 생략하도록 비디오 인코딩 디바이스를 구성한다. 덧붙여서, 그 명령들은 그 비트스트림을 출력하도록 비디오 인코딩 디바이스를 구성한다.

[0018] 본 개시물의 하나 이상의 예들의 상세는 첨부 도면들 및 다음의 설명에서 언급된다. 다른 특징들, 목적들, 및 장점들은 상세한 설명 및 도면들로부터, 그리고 청구항들로부터 명확하게 될 것이다.

도면의 간단한 설명

[0019] 도 1은 본 개시물에서 설명되는 기법들을 이용할 수도 있는 일 예의 비디오 코딩 시스템을 도시하는 블록도이다.

도 2는 본 개시물에서 설명되는 기법들을 구현할 수도 있는 일 예의 비디오 인코더를 도시하는 블록도이다.

도 3은 본 개시물에서 설명되는 기법들을 구현할 수도 있는 일 예의 비디오 디코더를 도시하는 블록도이다.

도 4a는 본 개시물의 하나 이상의 기법들에 따른, 비디오 인코더의 일 예의 동작을 도시하는 흐름도이다.

도 4b는 본 개시물의 하나 이상의 기법들에 따른, 비디오 디코더의 일 예의 동작을 도시하는 흐름도이다.

도 5a는 본 개시물의 기법들에 따른, 비디오 인코더의 다른 예의 동작을 도시하는 흐름도이다.

도 5b는 본 개시물의 기법들에 따른, 비디오 디코더의 다른 예의 동작을 도시하는 흐름도이다.

도 6a는 본 개시물의 기법들에 따른, 비디오 인코더의 다른 예의 동작을 도시하는 흐름도이다.

도 6b는 본 개시물의 기법들에 따른, 비디오 디코더의 다른 예의 동작을 도시하는 흐름도이다.

도 7a는 본 개시물의 기법들에 따른, 비디오 인코더의 다른 예의 동작을 도시하는 흐름도이다.

도 7b는 본 개시물의 기법들에 따른, 비디오 디코더의 다른 예의 동작을 도시하는 흐름도이다.

도 8a는 본 개시물의 기법들에 따른, 비디오 인코더의 다른 예의 동작을 도시하는 흐름도이다.

도 8b는 본 개시물의 기법들에 따른, 비디오 디코더의 다른 예의 동작을 도시하는 흐름도이다.

도 9a는 본 개시물의 기법들에 따른, 비디오 인코더의 다른 예의 동작을 도시하는 흐름도이다.

도 9b는 본 개시물의 기법들에 따른, 비디오 디코더의 다른 예의 동작을 도시하는 흐름도이다.

도 10은 일 예의 3차원 비디오 (3-dimensional video; 3DV) 디코딩 순서를 도시하는 개념도이다.

도 11은 일 예의 시간적 및 뷰간 예측 구조를 도시하는 개념도이다.

발명을 실시하기 위한 구체적인 내용

[0020] 3차원 비디오 (3-dimensional video; 3DV) 코딩에서, 동일한 장면의 이미지들이 상이한 관점들로부터 캡처된다. 상이한 관점들로부터 동일한 장면의 화상들을 보여주는 것은 관람자에게 입체 3차원 효과를 제공할 수도 있다. 상이한 관점들로부터 동시에 캡처된 동일한 장면의 화상들이 상당히 유사할 수도 있기 때문에, 비디오 인코더는, 화상들의 블록들을 상이한 관점들로부터의 다른 화상들에서의 블록들에 기반을 두고 예측함으로써, 전송되는 데이터의 양을 뷰간 예측을 사용하여 감소시킬 수도 있다. 용어 "액세스 유닛"은 동일한 시간 인스턴스에 대응하는 화상들의 세트를 지칭하는데 사용된다. "뷰 성분 (view component)"은 단일 액세스 유닛에서의 뷰의 코딩된 표현일 수도 있다.

[0021] 전송된 데이터의 양을 더욱 감소시키기 위해, 비디오 인코더는 현재 코딩되고 있는 화상과는 동일한 액세스 유닛에서의 이전에 코딩된 뷰 성분들에 기초하여 뷰 합성 예측 (view synthesis prediction; VSP) 화상을 생성할 수도 있다. 비디오 인코더는 그 VSP 화상을 참조 화상 리스트에 포함시킬 수도 있다. 비디오 인코더가 현재 비디오 유닛 (예컨대, 매크로블록 (macroblock; MB), MB 파티션 (partition), 서브 MB 파티션, 예측 유닛 (PU) 등) 을 인코딩하는 경우, 비디오 인코더는 현재 비디오 유닛에 대한 예측 블록을 생성함에 있어서 VSP 화상을 참조 화상으로서 사용할 수도 있다. 더욱이, 비디오 인코더는 참조 인덱스 및 모션 벡터를 시그널링할 수도 있다. 참조 인덱스는 참조 화상 리스트 내의 VSP 화상의 포지션을 나타낼 수도 있다. 모션 벡터는 VSP 화상 내의 참조 블록 및 현재 비디오 유닛의 샘플 블록들 사이의 공간적 변위를 나타낸다.

[0022] 비디오 인코더는 모션 벡터 차이 (MVD) 를 사용하여 모션 벡터를 시그널링할 수도 있다. MVD는 모션 벡터 예측변수 (predictor) 와 현재 비디오 유닛의 모션 벡터 사이의 차이를 나타낼 수도 있다. 모션 벡터 예측변수는 이웃 블록의 모션 벡터일 수도 있다.

[0023] 비디오 디코더는 비디오 인코더와 동일한 VSP 화상을 생성할 수도 있고 비디오 인코더와 동일한 참조 화상 리스트를 생성할 수도 있다. 더욱이, 비디오 디코더는, 참조 인덱스에 기초하여, 현재 비디오 유닛에 대한 예측 블록이 VSP 화상에 기초하여 생성된다고 결정할 수도 있다. 덧붙여서, 비디오 디코더는, 시그널링된 MVD에 적어도 부분적으로 기초하여, 현재 비디오 유닛에 대한 모션 벡터를 결정할 수도 있다. 비디오 디코더는 그 다음에, 모션 벡터에 적어도 부분적으로 기초하여, VSP 화상 내의 참조 블록을 결정할 수도 있다. 다음으로, 비디오 디코더는, 참조 블록에 적어도 부분적으로 기초하여, 현재 비디오 유닛에 대한 예측 블록을 결정할 수도 있다. 비디오 디코더는, 현재 비디오 유닛에 대한 예측 블록에 적어도 부분적으로 기초하여, 현재 비디오 유닛의 샘플 블록들을 복원할 수도 있다.

[0024] 위에서 언급했듯이, 비디오 인코더가 VSP 화상을 사용하여 현재 비디오 유닛에 대한 예측 블록을 생성하는 경우, 비디오 인코더는 비디오 디코더가 현재 비디오 유닛의 모션 벡터를 도출하는 MVD를 시그널링한다. 비디오 인코더가 VSP 화상을 사용하여 현재 비디오 유닛에 대한 예측 블록을 생성하는 경우, 모션 벡터는 거의 항상 0에 매우 가깝다. 다시 말하면, VSP 화상에서의 참조 블록은 거의 항상 현재 비디오 유닛의 샘플 블록들과 병치된다.

[0025] VSP 화상에서의 참조 블록이 거의 항상 현재 비디오 유닛의 샘플 블록들과 병치되기 때문에, 비디오 디코더는, 현재 비디오 유닛이 VSP 화상에 기초하여 인코딩되면, 비트스트림으로부터 현재 비디오 유닛에 대한 MVD를 디코딩하는 일 없이, 현재 비디오 유닛의 모션 벡터가 0과 동일하다고 결정할 수도 있다. 그런고로, 현재 비디오 유닛이 VSP 화상에 기초하여 인코딩되는 경우 현재 비디오 유닛에 대한 MVD를 시그널링하는 것은 비트들을 낭비하는 것일 수도 있다. 더구나, 현재 비디오 유닛이 VSP 화상에 기초하여 인코딩되는 경우 현재 비디오 유닛에 대한 MVD를 시그널링하는 것은 불필요하기 때문에, 현재 비디오 유닛이 VSP 화상에 기초하여 인코딩된다는 것을 나타내는 신택스 엘리먼트의 시그널링은, VSP 화상을 참조 화상 리스트 내에 포함시키는 것을 불필요하게 만들 수도 있다.

[0026] 본 개시물의 기법들에 따라, 비디오 인코더는, 다수의 텍스처 뷰들 및 다수의 깊이 뷰들의 코딩된 표현을 포함하는 비트스트림으로, 현재 비디오 유닛이 현재 텍스처 뷰 성분 (texture view component) 의 VSP 화상으로부터 예측되는지의 여부를 나타내는 신택스 엘리먼트를 시그널링할 수도 있다. 일부 예들에서, 현재 비디오 유닛은 MB, MB 파티션, 또는 서브 MB 파티션일 수도 있다. 다른 예들에서, 현재 비디오 유닛은 예측 유닛 (PU) 일 수도 있다. 일부 예들에서, 현재 비디오 유닛이 VSP 화상으로부터 예측된다는 것을 신택스 엘리먼트가 나타내는 경우, 비디오 인코더는 현재 비디오 유닛에 대한 모션 정보를 시그널링하지 않는다. 다르게 말하면, 비디오 인코더는, 비트스트림으로부터, 현재 비디오 유닛에 대한 모션 정보를 생략한다. 예를 들어, 비디오 인코더는 현재 비디오 유닛이 VSP 화상으로부터 예측되는 경우 현재 비디오 유닛에 대한 참조 인덱스들 또는 MVD들을 시그널링하지 않는다. 반대로, 현재 비디오 유닛이 VSP 화상으로부터 예측되지 않는 경우, 비디오 인코더는, 비트스트림으로, 현재 비디오 유닛에 대한 모션 정보를 시그널링할 수도 있다. 따라서, 비디오 인코더는, 현재 비디오 유닛이 VSP 화상으로부터 예측되는지의 여부에 적어도 부분적으로 기초하여, 비트스트림으로 현재 비디오 유닛에 대한 모션 정보를 시그널링할지의 여부를 결정할 수도 있다. 현재 비디오 유닛이 VSP 화상으로부터 예측되는 경우 현재 비디오 유닛의 모션 정보를 시그널링하지 않는 것에 의해, 비디오 인코더는 비트스트림에서의 비트들의 수를 줄일 수도 있다.

[0027] 마찬가지로, 본 개시물의 기법들에 따라, 비디오 디코더는, 현재 액세스 유닛의 이전에 코딩된 텍스처 뷰 성분 및 현재 액세스 유닛의 깊이 뷰 성분에 적어도 부분적으로 기초하여, VSP 화상을 생성할 수도 있다. 더욱이, 비디오 디코더는, 다수의 텍스처 뷰들 및 다수의 깊이 뷰들의 코딩된 표현을 포함하는 비트스트림으로부터, 현재 비디오 유닛이 VSP 화상으로부터 예측되는지의 여부를 나타내는 신택스 엘리먼트를 디코딩할 수도 있다. 현재 비디오 유닛이 VSP 화상으로부터 예측되지 않는 경우, 비디오 디코더는, 비트스트림으로부터, 현재 비디오 유닛에 대한 모션 정보를 디코딩할 수도 있고 현재 비디오 유닛에 대한 모션 정보를 사용하여 현재 비디오 유닛의 샘플 블록들을 복원할 수도 있다. 현재 비디오 유닛이 VSP 화상으로부터 예측되는 경우, 비디오 디코더는 VSP 화상을 사용하여 현재 비디오 유닛의 샘플 블록들을 복원할 수도 있다.

[0028] 도 1은 본 개시물의 기법들을 활용할 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 일반적

으로 지칭한다. 본 개시물에서, 용어들 "비디오 코딩" 또는 "코딩"은 비디오 인코딩 또는 비디오 디코딩을 일반적으로 지칭할 수도 있다.

[0029] 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. 따라서, 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. 따라서, 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.

[0030] 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 컴퓨터 등을 포함한 다양한 범위의 디바이스들을 포함할 수도 있다.

[0031] 목적지 디바이스 (14) 는 소스 디바이스 (12) 로부터 채널 (16) 을 통해 인코딩된 비디오 데이터를 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 부분을 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.

[0032] 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 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.

[0033] 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) 에 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.

[0034] 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, 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.

[0035] 본 개시물의 기법들은 무선 애플리케이션들 또는 설정 (setting) 들로 제한되지 않는다. 그 기법들은, 다양한 멀티미디어 애플리케이션들, 이를테면 OTA (over-the-air) 텔레비전 브로드캐스트들, 케이블 텔레비전 송신들, 위성 텔레비전 송신들, 예컨대, 인터넷을 통한 스트리밍 비디오 송신들 중 임의의 것의 지원 하의 비디오 코딩, 데이터 저장 매체 상의 저장을 위한 비디오 데이터의 인코딩, 데이터 저장 매체 상에 저장된 비디오 데이터의 디코딩, 또는 다른 애플리케이션들에 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.

[0036] 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.

[0037] 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 송신할 수도 있다. 다른 예들에서, 인코딩된 비디오 데이터는 또한 디코딩 및/또는 플레이백을 위한 목적지 디바이스 (14) 에 의한 나중의 액세스를 위해 저장 매체 또는 파일 서버 상에 저장될 수도 있다.

[0038] 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. 디스플레이 디바이스 (32) 는 목적지 디바이스 (14) 와 통합되거나 또는 그것 외부에 있을 수도 있다. 대체로, 디스플레이 디바이스 (32) 는 디코딩된 비디오 데이터를 디스플레이한다. 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.

[0039] 일부 예들에서, 비디오 인코더 (20) 및 비디오 디코더 (30) 는, ISO/IEC MPEG-4 비주얼 그리고 SVC (Scalable Video Coding) 및 MVC (Multiview Video Coding) 확장본들을 포함한 ITU-T H.264 (또한 ISO/IEC MPEG-4 AVC로 알려짐) 와 같은 비디오 압축 표준에 따라 동작한다. MVC의 초안은 『"Advanced video coding for generic audiovisual services", ITU-T Recommendation H.264, March 2010』에 기재되어 있으며, 그것은 2013년 3월 13일 현재 http://www.itu.int/rec/T-REC-H.264-201003-S/en으로부터 입수할 수 있으며, 그 전체 내용은 참조로 본원에 통합된다. H.264/AVC의 MVC 확장본의 다른 최근의 초안은, 2013년 3월 13일 현재, http://wftp3.itu.int/av-arch/jvt-site/2009_01_Geneva/JVT-AD007.zip에서 다운로드 가능하며, 그 전체 내용은 참조로 본원에 통합된다.

[0040] 덧붙여서, 『"WD of MVC extension for inclusion of depth maps", MPEG document w12351』에 기재된 MVC 표준의 확장본, 즉, "MVC 기반 3DV" (즉, MVC-호환 3DV) 이 있으며, 그 전체 내용은 참조로 본원에 통합된다. 일부 경우들에서, MVC 기반 3DV에 부합하는 임의의 법적 비트스트림은 MVC 프로파일, 예컨대, 스테레오 하이 프로파일을 준수하는 서브-비트스트림을 항상 포함할 수도 있다.

[0041] 더욱이, H.264/AVC에 대한 3차원 비디오 (3DV) 코딩 확장본, 즉 AVC 기반 3DV를 생성하려는 지속적인 노력이 있다. 이후로는 "3DV-AVC 규격 초안 1"이라고 지칭되는 AVC 기반 3DV의 규격 초안 (WD) 이 참조로 본원에 통합된다. AVC 기반 3DV본의 다른 초안은 『Mannuksela et al., "3D-AVC Draft Text 4", 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 China, October 2012』에 기재되어 있으며, 그것은, 2013년 3월 13일 현재, http://phenix.it-sudparis.eu/jct2/doc_end_user/documents/2_Shanghai/wg11/JCT3V-B1002-v1.zip에서 입수가능하며, 그 전체 내용은 참조로 본원에 통합된다. AVC 기반 3DV에 대한 참조 소프트웨어의 설명이 『Miska M. Hannuksela, "Test Model for AVC based 3D video coding", ISO/IEC JTC1/SC29/WG11 MPEG2011/N12558, San Jose, USA, Feb. 2012』로서 입수가능하다. 참조 소프트웨어는, 2013년 3월 13일 현재, http://mpeg3dv.research.nokia.com/svn/mpeg3dv/trunk/로부터 입수가능하며, 그 전체 내용은 참조로 본원에 통합된다.

[0042] 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.264, ISO/IEC 비주얼에 따라 동작할 수도 있다. 비디오 인코더 (20) 와 비디오 디코더 (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 현재 개발 중인 고 효율 비디오 코딩 (HEVC) 표준을 포함하는 다른 비디오 압축 표준들에 따라 동작할 수도 있다.

[0043] "HEVC 규격 초안 4"라고 지칭되는 근간의 HEVC 표준의 초안은 『Bross et al., "WD4: Working Draft 4 of High Efficiency Video Coding", Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG16 WP3 and

ISO/IEC JTC1/SC29/WG11, 6th Meeting, Torino, IT, July, 2011』에 기재되어 있으며, 그것은 2013년 3월 13일 현재 http://phenix.int-evry.fr/jct/doc_end_user/documents/6_Torino/wg11/JCTVC-F803-v8.zip로부터 입수가능하며, 그 전체 내용은 참조로 본원에 통합된다. "HEVC 규격 초안 6"이라고 지칭되는 근간의 HEVC 표준의 다른 초안은, 『Bross et al., "High Efficiency Video Coding (HEVC) text specification draft 6", Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, 8th Meeting, San Jose, CA, February, 2012』에 기재되어 있으며, 그것은 2013년 3월 13일 현재, http://phenix.int-evry.fr/jct/doc_end_user/documents/8_San%20Jose/wg11/JCTVC-H1003-v22.zip에서 다운로드가능하고, 그 전체 내용은 참조로 본원에 통합된다. "HEVC 규격 초안 9"이라고 지칭되는 근간의 HEVC 표준의 다른 초안은, 『Bross et al., "High Efficiency Video Coding (HEVC) text specification draft 9", Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, 11th Meeting: Shanghai, China, October, 2012』에 기재되어 있으며, 이는 2013년 3월 13일 현재, http://phenix.int-evry.fr/jct/doc_end_user/documents/11_Shanghai/wg11/JCTVC-K1003-v8.zip에서 다운로드가능하고, 그 전체가 참조로 본원에 통합된다.

[0044] 더욱이, HEVC에 대한 3DV 확장본을 생성하려는 노력이 지속되고 있다. HEVC의 3DV 확장본은 HEVC 기반 3DV 또는 HEVC-3DV라고 지칭될 수도 있다. MPEG에서의 HEVC 기반 3DV 코덱은, 전체 내용이 본원에 참조로 통합되는『Schwarz et al., "Description of 3D Video Technology Proposal by Fraunhofer HHI (HEVC compatible; configuration A)", ISO/IEC JTC1/SC29/WG11 MPEG2011/m22570, Geneva, Switzerland, Nov. 2011』(이후로는 "문서 m22570") 및 전체 내용이 본원에 참조로 통합되는 『Wegner et al., "Poznan University of Technology tools for 3DV coding integrated into 3D-HTM", ISO/IEC JTC1/SC29/WG11 MPEG2011/m23783, San Jose, USA, Feb. 2012』(이후로는 "문서 m23783")에서 제안된 솔루션들에 기초한다. 참조 소프트웨어 설명은 『Schwarz et al., "Test Model under Consideration for HEVC based 3D video coding", ISO/IEC JTC1/SC29/WG11 MPEG2011/N12559, San Jose, USA, Feb. 2012』로서 입수가능하며, 그 전체 내용은 참조로 본원에 통합된다. 참조 소프트웨어는 2013년 3월 13일 현재 https://hevc.hhi.fraunhofer.de/svn/svn_3DVCSoftware/trunk로부터 입수가능하다. 비록 본 개시물의 기법들이 H.264/AVC 및 HEVC에 관해 설명되고 있지만, 이러한 기법들은 다른 코딩 표준들에 적용될 수도 있고 임의의 특정 코딩 표준 또는 기법으로 제한되지 않는다.

[0045] 도 1은 단지 일 예이고 본 개시물의 기법들은 인코딩 및 디코딩 디바이스들 사이에 임의의 데이터 통신을 반드시 포함하지는 않는 비디오 코딩 설정들 (예컨대, 비디오 인코딩 또는 비디오 디코딩) 에 적용될 수도 있다. 다른 예들에서, 데이터는 로컬 메모리로부터 취출되며, 네트워크를 통해 스트리밍되는 등등이 된다. 비디오 인코딩 디바이스가 데이터를 인코딩하고 메모리에 저장할 수도 있으며, 그리고/또는 비디오 디코딩 디바이스는 메모리로부터 데이터를 취출하고 디코딩할 수도 있다. 많은 예들에서, 인코딩 및 디코딩은, 서로 통신하지 않지만 단순히 데이터를 메모리에 인코딩하고 및/또는 메모리로부터 데이터를 취출하고 디코딩하는 디바이스들에 의해 수행된다.

[0046] 비디오 인코더 (20) 와 비디오 디코더 (30) 각각은 다양한 적합한 회로, 이를테면 하나 이상의 마이크로프로세서들, 디지털 신호 프로세서들 (DSP들), 주문형 집적회로들 (ASIC들), 필드 프로그램가능 게이트 어레이들 (FPGA들), 개별 로직, 하드웨어, 또는 그것들의 임의의 조합 중 임의의 것으로서 구현될 수도 있다. 그 기법들이 부분적으로 소프트웨어로 구현되면, 디바이스는 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) 의 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.

[0047] 본 개시물은 다른 디바이스, 이를테면 비디오 디코더 (30) 에 특정한 정보를 "시그널링하는" 비디오 인코더 (20) 를 일반적으로 지칭할 수도 있다. 용어 "시그널링"은 일반적으로 압축된 비디오 데이터를 디코딩하는 데 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 비트스트림으로 컴퓨터 판독가능 저장 매체에 저장하고 그 신택스 엘리먼트들이 이 매체에 저장된 후의 임의의 시간에 디코딩 디바이스에 의해 취출될 수도 있는 경우에 일어날 바와 같이 시간 (span of time) 에 걸쳐 일어

날 수도 있다.

[0048] 비디오 시퀀스는 통상 일련의 화상들을 포함한다. 화상들은 또한 "프레임들"이라고 지칭될 수도 있다. 화상은, 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 색차 샘플들의 2차원 어레이이다. S_{Cr}은 Cr 색차 샘플들의 2차원 어레이이다. 색차 샘플들은 또한 본원에서 "크로마" 샘플들이라고 지칭될 수도 있다. 다른 경우들에서, 화상은 모노크롬일 수도 있고 루마 샘플들의 어레이만을 포함할 수도 있다.

[0049] 화상의 인코딩된 표현을 생성하기 위해, 비디오 인코더 (20) 는 화상의 샘플 어레이들을 동일 사이즈로 된 블록들로 분할할 수도 있다. 예를 들어, H.264/AVC에서, 비디오 인코더 (20) 는 화상을 매크로블록들 (MB들) 로 분할할 수도 있다. MB는 3 개의 샘플 어레이들을 갖는 화상의 루마 샘플들의 16x16 블록 및 크로마 샘플들의 2 개의 대응하는 블록들, 또는 모노크롬 화상 또는 3 개의 별개의 컬러 평면들을 사용하여 코딩된 화상의 샘플들의 16x16 블록이다. H.264/AVC에서, 슬라이스는 특정 슬라이스 그룹 내에서 래스터 스캔으로 연속적으로 정렬된 정수 수의 MB들 또는 MB 쌍들을 포함할 수도 있다.

[0050] 비디오 인코더 (20) 는 MB를 하나 이상의 MB 파티션들의 세트로 파티셔닝할 수도 있다. MB 파티션은, 3 개의 샘플 어레이들을 갖는 화상에 대한 인터 예측을 위한 MB의 파티셔닝의 결과로 생긴 루마 샘플들의 블록 및 크로마 샘플들의 2 개의 대응하는 블록들, 또는 모노크롬 화상 또는 3 개의 별개의 컬러 평면들을 사용하여 코딩된 화상의 인터 예측을 위한 MB의 파티셔닝의 결과로 생긴 루마 샘플들의 블록이다. 일부 경우들에서, 비디오 인코더 (20) 는 MB를 복수의 서브 MB들로 파티셔닝할 수도 있다. 각각의 서브 MB는 MB의 샘플들의 1/4, 즉, 하나의 코너가 MB의 코너에 위치되는, 3 개의 샘플 어레이들을 가지는 화상에 대한 8x8 루마 블록 및 2 개의 대응 크로마 블록들 또는 하나의 코너가 MB의 코너에 위치되는, 모노크롬 화상 또는 3 개의 별개의 컬러 평면들을 사용하여 인코딩된 화상에 대한 8x8 루마 블록이다. 서브 MB 파티션은, 3 개의 샘플 어레이들을 갖는 화상에 대한 인터 예측을 위한 서브 MB의 파티셔닝의 결과로 생긴 루마 샘플들의 블록 및 크로마 샘플들의 2 개의 대응하는 블록들, 또는 모노크롬 화상 또는 3 개의 별개의 컬러 평면들을 사용하여 코딩된 화상에 대한 인터 예측을 위한 서브 MB의 파티셔닝의 결과로 생긴 루마 샘플들의 블록이다. MB 또는 MB 파티션의 루마 및 크로마 블록들은 일반적으로 MB 또는 MB 파티션의 샘플 블록들이라고 지칭될 수도 있다.

[0051] HEVC에서, 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 구조들일 수도 있다. 코딩 트리 블록은 샘플들의 NxN 블록일 수도 있다. CTU은 또한 "최대 코딩 유닛 (LCU)"이라고 지칭될 수도 있다. HEVC에서, 슬라이스가 정수 수의 CTU들을 포함할 수도 있다. HEVC의 CTU들은 다른 표준들, 이를테면 H.264/AVC의 MB들과 대체로 유사할 수도 있다. 그러나, CTU는 특정 사이즈로 반드시 제한되는 것은 아니고 하나 이상의 코딩 유닛들 (CU들) 을 포함할 수도 있다. CU는, 루마 샘플 어레이, Cb 샘플 어레이 및 Cr 샘플 어레이를 가지는 화상의 루마 샘플들의 코딩 블록 및 크로마 샘플들의 2 개의 대응 코딩 블록들과, 그리고 코딩 블록들의 샘플들을 코딩하는데 사용된 신택스 구조들일 수도 있다. 코딩 블록은 샘플들의 NxN 블록이다.

[0052] 더욱이, HEVC에서, CU는 하나 이상의 예측 유닛들 (PU들) 을 가질 수도 있다. PU는 화상의 루마 샘플들의 예측 블록, 크로마 샘플들의 2 개의 대응하는 예측 블록들, 및 예측 블록 샘플들을 예측하는데 사용된 신택스 구조들일 수도 있다. 예측 블록은 동일한 예측이 적용되는 샘플들의 직사각형 (예컨대, MxN) 블록일 수도 있다. CU의 PU의 예측 블록이 CU의코딩 블록의 파티션일 수도 있다. PU의 루마 및 크로마 블록들은 일반적으로 PU의 샘플 블록들이라고 지칭될 수도 있다.

[0053] 비디오 인코더 (20) 가 현재 비디오 유닛 (이를테면 MB, MB 파티션, PU 등) 을 인코딩하는 경우, 비디오 인코더 (20) 는 현재 비디오 유닛을 위한 예측 루마 및 크로마 블록들을 생성할 수도 있다. 비디오 인코더 (20) 는 인트라 예측 또는 인터 예측을 수행하여 예측 블록들을 생성할 수도 있다. 비디오 인코더 (20) 가 인트라 예측을 수행하는 경우, 비디오 인코더 (20) 는, 현재 비디오 유닛과 동일한 화상 내의 샘플들에 적어도 부분적으로 기초하여, 현재 비디오 유닛에 대한 예측 루마 및 크로마 블록들을 생성할 수도 있다.

[0054] 비디오 인코더 (20) 가 인터 예측을 수행하여 현재 비디오 유닛에 대한 예측 루마 및 크로마 블록들을 생성하는 경우, 비디오 인코더 (20) 는 하나 이상의 참조 화상들 내의 참조 블록들에 기초하여 예측 블록들을 생성할 수도 있다. 참조 화상들은 현재 비디오 유닛을 포함하는 화상과는 다른 화상들일 수도 있다. 더 구체적으로는, 비디오 인코더 (20) 는 제 1 참조 화상 리스트 (RefPicList0) 및 제 2 참조 화상 리스트 (RefPicList1)

를 생성할 수도 있다. RefPicList0 및 RefPicList1은 참조 화상들의 리스트들이다. 비디오 인코더 (20) 가 현재 비디오 유닛을 인코딩하기 위해 단방향 인터 예측을 사용하면, 비디오 인코더 (20) 는 참조 블록을 포함하는 참조 화상의 RefPicList0 또는 RefPicList1 중 어느 하나 내의 포지션을 나타내는 참조 인덱스를 시그널링할 수도 있다. 더욱이, 비디오 인코더 (20) 는 현재 비디오 유닛의 루마 블록 및 참조 블록 사이의 공간적 변위를 나타내는 모션 벡터를 시그널링할 수도 있다. 비디오 인코더 (20) 가 양방향 인터 예측을 사용하면, 비디오 인코더 (20) 는 참조 블록들을 포함하는 참조 화상들의 RefPicList0 및 RefPicList1 내의 포지션들을 나타내는 2 개의 참조 인덱스들을 시그널링할 수도 있다. 비디오 인코더 (20) 는 현재 비디오 유닛의 루마 블록들 및 참조 블록들 사이의 공간적 변위들을 나타내는 모션 벡터들을 시그널링할 수도 있다.

[0055] H.264/AVC에서, 각각의 인터 MB (즉, 인터 예측을 사용하여 인코딩되는 각각의 MB) 는 4 개의 상이한 방법들 중 하나, 즉, 하나의 16x16 MB 파티션, 2 개의 16x8 MB 파티션들, 2 개의 8x16 MB 파티션들, 또는 4 개의 8x8 MB 파티션들로 파티셔닝될 수도 있다. 하나의 블록에서의 상이한 MB 파티션들은 각각의 방향 (RefPicList0 또는 RefPicList1) 에 대해 상이한 참조 인덱스들을 가질 수도 있다. MB가 4 개의 8x8 MB 파티션들로 파티셔닝되지 않은 경우, MB는 각각의 방향에서 전체 MB 파티션에 대해 단지 하나의 모션 벡터를 가진다. 이 경우, MB 파티션은 16x16, 8x16 또는 16x8의 사이즈를 가질 수 있다. MB가 4 개의 8x8 MB 파티션들로 파티셔닝되는 경우, 각각의 8x8 MB 파티션은 서브블록들로 추가로 파티셔닝될 수 있으며, 그 서브블록들의 각각은 각각의 방향에서 상이한 모션 벡터를 가질 수 있다. 8x8 MB 파티션을 서브블록들, 즉, 하나의 8x8 서브블록, 2 개의 8x4 서브블록들, 2 개의 4x8 서브블록들, 및 4 개의 4x4 서브블록들로 파티셔닝하는 4 개의 상이한 방법들이 있다. 서브블록들의 각각은 각각의 방향에서 상이한 모션 벡터를 가질 수도 있다.

[0056] H.264/AVC에서, 비디오 인코더 (20) 는 모션 벡터 차이 (MVD) 를 시그널링함으로써 현재 비디오 유닛 (예컨대, MB 또는 MB 파티션) 에 대한 모션 벡터를 시그널링할 수도 있다. MVD는 모션 벡터 예측변수와 현재 비디오 유닛에 대한 모션 벡터 사이의 차이를 나타낸다. 모션 벡터 예측변수는 이웃 블록의 모션 벡터일 수도 있다. 이웃 블록은 현재 비디오 유닛의 상측 또는 좌측의 샘플 블록들일 수도 있다. 이웃 블록이 모션 벡터 예측변수를 생성함에 있어서 사용할 수 없으면, 모션 벡터 예측변수의 수평 및 수직 성분들은 0과 동일할 수도 있다. 다른 이유들보다도, 이웃 블록과 현재 블록이 상이한 슬라이스들에 있거나, 이웃 블록이 현재 화상의 경계 내에 있지 않는 등의 이유로, 이웃 블록은 이용가능하지 않을 수도 있다.

[0057] 더욱이, H.264/AVC에서, 비디오 인코더 (20) 는 MB에 대한 신택스 엘리먼트들을 포함하는 MB 계층 신택스 구조를 생성할 수도 있다. MB 계층 신택스 구조는, MB의 파티셔닝 모드에 의존하여, MB 예측 신택스 구조 또는 서브 MB 예측 신택스 구조를 포함할 수도 있다. MB 예측 신택스 구조 또는 서브 MB 예측 신택스 구조는 MB에 대한 모션 정보 또는 MB의 MB 파티션들에 대한 모션 정보를 나타내는 신택스 엘리먼트들을 포함할 수도 있다. 예를 들면, MB 예측 신택스 구조들 및 서브 MB 예측 신택스 구조들은 참조 인덱스들 및 MVD들을 특정하는 신택스 엘리먼트들을 포함할 수도 있다.

[0058] 더욱이, HEVC에서, 비디오 인코더 (20) 가 현재 PU에 대한 예측 블록들을 생성하기 위해 인터 예측을 사용하는 경우, 비디오 인코더 (20) 는 병합 모드 또는 적응적 모션 벡터 예측 (AMVP) 모드를 사용하여 현재 PU에 대한 모션 정보를 시그널링할 수도 있다. 병합 모드 또는 AMVP 모드 중 어느 하나에서, 비디오 인코더 (20) 는 예측변수 후보들의 리스트 (즉, 후보 리스트) 를 생성할 수도 있다. 예측변수 후보들은 현재 PU와는 다른 PU들의 모션 정보를 특정할 수도 있다. 병합 모드에서, 비디오 인코더 (20) 는 선택된 예측변수 후보의 후보 리스트 내의 포지션을 시그널링할 수도 있다. PU의 모션 정보는 선택된 예측변수 후보에 의해 특정된 모션 정보와 동일할 수도 있다. AMVP 모드에서, 비디오 인코더 (20) 는 선택된 예측변수 후보의 후보 리스트 내의 포지션, 참조 인덱스, 및 현재 PU에 대한 MVD를 시그널링할 수도 있다. 현재 PU에 대한 MVD는 선택된 예측변수 후보의 모션 벡터와 현재 PU의 모션 벡터 사이의 차이에 기초할 수도 있다.

[0059] 비디오 인코더 (20) 가 현재 비디오 유닛 (이를테면 MB, MB 파티션, PU 등) 에 대응하는 예측 블록을 생성한 후, 비디오 인코더 (20) 는 잔차 블록을 생성할 수도 있다. 잔차 블록에서의 각각의 샘플은 현재 비디오 유닛의 샘플 블록 및 예측 블록에서의 대응하는 샘플들 사이의 차이에 기초할 수도 있다. 비디오 인코더 (20) 는 하나 이상의 변환 계수 블록들을 생성하기 위해 잔차 블록에 변환을 적용할 수도 있다. 비디오 인코더 (20) 는 변환 계수 블록들을 양자화하여 현재 비디오 유닛을 표현하는데 사용되는 비트들의 수를 더 감소시킬 수도 있다. 변환 계수 블록을 양자화한 후, 비디오 인코더 (20) 는 변환 계수 블록에서의 변환 계수들을 나타내는 신택스 엘리먼트들과 다른 신택스 엘리먼트들을 엔트로피 인코딩할 수도 있다. 예를 들어, 비디오 인코더 (20) 는 신택스 엘리먼트들에 대해 콘텍스트 적응 이진 산술 코딩 (CABAC), 콘텍스트 적응 가변 길이 코딩 (CAVLC), 지수-골롬 코딩 (exponential-Golomb coding) 또는 다른 유형의 엔트로피 인코딩을 수행할 수도

있다. 비디오 인코더 (20) 는 엔트로피 인코딩된 신택스 엘리먼트들을 포함하는 비트스트림을 출력할 수도 있다.

[0060] CABAC 인코딩을 신택스 엘리먼트에 적용하기 위해, 비디오 인코더 (20) 는 신택스 엘리먼트를 이진화하여 "빈 (bin) 들"이라고 지칭되는 일련의 하나 이상의 비트들을 형성할 수도 있다. 덧붙여서, 비디오 인코더 (20) 는 코딩 콘텍스트를 식별할 수도 있다. 코딩 콘텍스트는 특정 값들을 갖는 빈들을 코딩하는 확률들을 식별할 수도 있다. 예를 들면, 코딩 콘텍스트는 0 값의 빈을 코딩하는 0.7 확률과 1 값의 빈을 코딩하는 0.3 확률을 나타낼 수도 있다. 코딩 콘텍스트를 식별한 후, 비디오 인코더 (20) 는 간격을 하부 서브간격 및 상부 서브간격으로 나눌 수도 있다. 서브간격들 중 하나는 값 0과 연관될 수도 있고 나머지 서브간격은 값 1과 연관될 수도 있다. 서브간격들의 폭들은 식별된 코딩 콘텍스트에 의해 연관된 값들에 대해 나타내어진 확률들에 비례할 수도 있다. 신택스 엘리먼트의 빈이 하부 서브간격에 연관된 값을 가지면, 인코딩된 값은 하부 서브간격의 하부 경계와 동일할 수도 있다. 신택스 엘리먼트의 동일한 빈이 상부 서브간격에 연관된 값을 가지면, 인코딩된 값은 상부 서브간격의 하부 경계와 동일할 수도 있다. 신택스 엘리먼트의 다음의 빈을 인코딩하기 위해, 비디오 인코더 (20) 는 인코딩된 비트의 값에 연관된 서브간격인 간격으로 이들 단계들을 반복할 수도 있다. 비디오 인코더 (20) 가 다음 빈에 대해 이들 단계들을 반복하는 경우, 비디오 인코더 (20) 는 식별된 코딩 콘텍스트 및 인코딩된 빈들의 실제 값들에 의해 나타내어진 확률들에 기초한 수정된 확률들을 사용할 수도 있다.

[0061] 비디오 인코더 (20) 는 엔트로피 인코딩된 신택스 엘리먼트들을 포함하는 비트스트림을 출력할 수도 있다. 그 비트스트림은 코딩된 화상들의 표현 및 연관된 데이터를 형성하는 비트들의 시퀀스를 포함할 수도 있다. 그 비트스트림은 네트워크 추상화 계층 (NAL) 유닛들의 시퀀스를 포함할 수도 있다. NAL 유닛들의 각각은 NAL 유닛 헤더를 포함하고 원시 바이트 시퀀스 페이로드 (raw byte sequence payload; RBSP) 를 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. RBSP는 NAL 유닛 내에 캡슐화되는 정수 수의 바이트들을 포함하는 신택스 구조일 수도 있다. 일부 경우들에서, RBSP는 0 비트들을 포함한다.

[0062] 상이한 유형들의 NAL 유닛들이 상이한 유형들의 RBSP들을 캡슐화할 수도 있다. 예를 들어, 제 1 유형의 NAL 유닛은 화상 파라미터 세트 (PPS) 에 대한 RBSP를 캡슐화할 수도 있으며, 제 2 유형의 NAL 유닛은 코딩된 슬라이스에 대한 RBSP를 캡슐화할 수도 있으며, 제 3 유형의 NAL 유닛은 추가 향상 정보 (supplemental enhancement information; SEI) 에 대한 RBSP를 캡슐화할 수도 있다는 등등이다. 비디오 코딩 데이터에 대한 RBSP들 (파라미터 세트들 및 SEI 메시지들에 대한 RBSP과는 대조적임) 을 캡슐화하는 NAL 유닛들은, 비디오 코딩 계층 (VCL) NAL 유닛들이라고 지칭될 수도 있다.

[0063] 비디오 디코더 (30) 는 비디오 데이터의 인코딩된 표현을 포함하는 비트스트림을 수신할 수도 있다. 비디오 디코더 (30) 는 비트스트림의 신택스 엘리먼트들을 디코딩하기 위해 그 비트스트림을 파싱할 수도 있다. 비트스트림의 신택스 엘리먼트들을 디코딩하는 부분으로서, 비디오 디코더 (30) 는 비트스트림의 신택스 엘리먼트들을 엔트로피 디코딩할 수도 있다. 예를 들어, 비디오 디코더 (30) 는 적어도 일부 신택스 엘리먼트들에 대해 CABAC 디코딩을 수행할 수도 있다. 비디오 디코더 (30) 는, 현재 비디오 유닛 (예컨대, MB 또는 MB 파티션, PU 등) 에 연관된 신택스 엘리먼트들에 적어도 부분적으로 기초하여, 인터 또는 인트라 예측을 수행하여 그 현재 비디오 유닛에 대한 예측 블록들을 생성할 수도 있다. 덧붙여서, 비디오 디코더 (30) 는 현재 비디오 유닛에 연관된 변환 계수 블록들의 변환 계수들을 역 양자화할 수도 있고, 하나 이상의 역 변환들을 변환 계수 블록들에 적용하여 잔차 블록들을 생성할 수도 있다. 비디오 디코더 (30) 는 잔차 블록 및 예측 블록에 적어도 부분적으로 기초하여 현재 비디오 유닛에 대한 샘플 블록을 복원할 수도 있다. 이런 식으로, 화상의 블록들을 복원함으로써, 비디오 디코더 (30) 는 그 화상을 복원할 수도 있다.

[0064] 비디오 디코더 (30) 가 신택스 엘리먼트에 대해 CABAC 디코딩을 수행하는 경우, 비디오 디코더 (30) 는 코딩 콘텍스트를 식별할 수도 있다. 비디오 디코더 (30) 는 그 다음에 간격을 하부 서브간격 및 상부 서브간격으로 분할할 수도 있다. 서브간격들 중 하나는 값 0과 연관될 수도 있고 나머지 서브간격은 값 1과 연관될 수도 있다. 서브간격들의 폭들은 식별된 코딩 콘텍스트에 의해 연관된 값들에 대해 나타내어진 확률들에 비례할 수도 있다. 인코딩된 값이 하부 서브간격 내에 있으면, 비디오 디코더 (30) 는 하부 서브간격에 연관된 값을 갖는 빈을 디코딩할 수도 있다. 인코딩된 값이 상부 서브간격 내에 있으면, 비디오 디코더 (30) 는 상부 서브간격에 연관된 값을 갖는 빈을 디코딩할 수도 있다. 신택스 엘리먼트의 다음의 빈을 디코딩하기 위해, 비디오 디코더 (30) 는 인코딩된 값을 포함하는 서브간격인 간격을 가지고서 이들 단계들을 반복할 수도 있다.

비디오 디코더 (30) 가 이들 단계들을 다음의 빈에 대해 반복하는 경우, 비디오 디코더 (30) 는 식별된 코딩 콘텍스트 및 디코딩된 빈들에 의해 나타내어진 확률들에 기초한 수정된 확률들을 사용할 수도 있다. 비디오 디코더 (30) 는 그 다음에 빈들을 이진화제거하여 신택스 엘리먼트를 복원할 수도 있다.

[0065] 위에서 언급했듯이, 멀티뷰 비디오 코딩 (MVC) 은 H.264/AVC 표준의 확장본이다. H.264/AVC에 대한 MVC 확장본에서, 상이한 관점들로부터의 동일한 장면의 다수의 뷰들이 있을 수도 있다. 용어 "액세스 유닛"은 동일한 시간 인스턴스에 대응하는 화상들의 세트를 지칭하는데 사용된다. 따라서, 비디오 데이터는 시간이 지남에 따라 발생하는 일련의 액세스 유닛들로서 개념화될 수도 있다. "뷰 성분"은 단일 액세스 유닛에서의 뷰의 코딩된 표현일 수도 있다. 본 개시물에서, "뷰"는 동일한 뷰 식별자에 연관된 뷰 성분들의 시퀀스를 지칭할 수도 있다.

[0066] H.264/AVC의 MVC 확장본은 뷰간 예측을 지원한다. 뷰간 예측은 H.264/AVC에서 사용된 인터 예측과 유사하고, 동일한 신택스 엘리먼트들을 사용할 수도 있다. 그러나, 비디오 코더가 현재 비디오 유닛 (이를테면, MB 또는 MB 파티션) 에 대해 뷰간 예측을 수행하는 경우, 비디오 인코더 (20) 는, 참조 화상으로서, 현재 비디오 유닛과 동일한 액세스 유닛에 있지만 상이한 뷰에 있는 화상을 사용할 수도 있다. 그 반면, 기존의 인터 예측은 상이한 액세스 유닛들에서의 화상들만을 참조 화상들로서 사용한다.

[0067] MVC에서, 비디오 디코더 (예컨대, 비디오 디코더 (30)) 가 뷰에서의 화상들을 임의의 다른 뷰에서의 화상들을 참조하지 않고 디코딩할 수 있다면, 그 뷰는 "기본 뷰 (base view)"라고 지칭될 수도 있다. 비기본 뷰들 (non-base views) 중 하나의 비기본 뷰에서의 화상을 코딩하는 경우, 화상이 상이한 뷰에 있지만 비디오 코더가 현재 코딩하고 있는 화상과는 동일한 시간 인스턴스 (즉, 액세스 유닛) 내에 있다면, 비디오 코더 (예컨대, 비디오 인코더 (20) 또는 비디오 디코더 (30)) 는 그 화상을 참조 화상 리스트에 추가할 수도 있다. 다른 인터 예측 참조 화상들처럼, 비디오 코더는 뷰간 예측 참조 화상을 참조 화상 리스트의 임의의 포지션에 삽입할 수도 있다.

[0068] MVC에서, 뷰간 예측은 디스패리티 (disparity) 모션 보상에 의해 지원될 수도 있다. 디스패리티 모션 보상은 H.264/AVC 모션 보상의 신택스를 사용하지만, 상이한 뷰에서의 화상이 참조 화상으로서 사용되는 것을 허용할 수도 있다. 둘 이상의 뷰들의 코딩은 MVC에 의해 지원될 수도 있다. MVC의 장점들 중 하나는, MVC 인코더가 2 개를 초과하는 뷰들을 3D 비디오 입력으로서 사용할 수도 있고 MVC 디코더는 이러한 멀티뷰 표현을 디코딩할 수도 있다는 것일 수도 있다. 그 결과, MVC를 지원하는 비디오 디코더들은 2 개를 초과하는 뷰들로 3D 비디오 콘텐츠를 프로세싱할 수도 있다.

[0069] 더욱이, H.264/AVC에 대한 신흥 MVC 기반 3DV 확장본이 있다. MVC 기반 3DV는 MVC 호환성을 유지하면서 3D 향상을 가능하게 하도록 설계된다. MVC 기반 3DV는 깊이 맵들을 제공한다. 따라서, MVC 기반 3DV는 "MVC 플러스 깊이", "MVC+D"로서, 또는 "깊이를 포함하는 MVC-호환 확장본"으로서 지칭될 수도 있다. 전체 내용이 본원에 참조로 통합되는 『Suzuki et al., "WD of MVC extensions for inclusion of depth maps", ISO/IEC/JTC1/SC29/WG11/N12351, December 2011』은 MVC 호환 3DV의 초안이다. 전체 내용이 본원에 참조로 통합되는 『Suzuki et al., "WD od MVC extensions for inclusion of depth maps", ISO/IEC/JTC1/SC29/WG11/N12544, February 2012』은 MVC 호환 3DV의 나중의 초안이다.

[0070] 깊이 맵들은, 화소 (예컨대, 샘플) 값들이 대응하는 "텍스처" 화상들에 보여진 오브젝트들의 3차원 깊이들을 나타내는 화상들이다. 일부 예들에서, 깊이 맵에서의 더 밝은 화소 값들일수록 카메라에 더 가까이 있는 오브젝트들에 대응할 수도 있고, 깊이 맵에서의 더 어두운 화소 값들일수록 카메라로부터 더 멀리 있는 오브젝트들에 대응할 수도 있다. "텍스처" 화상들은 일반 H.264/AVC 화상들일 수도 있다.

[0071] 본 개시물에서, 뷰의 텍스처 부분은 "텍스처 뷰"라고 지칭될 수도 있고 뷰의 깊이 부분은 "깊이 뷰"라고 지칭될 수도 있다. 하나의 액세스 유닛에서의 뷰의 텍스처 부분, 즉, 액세스 유닛에서의 텍스처 뷰는, "텍스처 뷰 성분"이라 지칭될 수도 있다. 하나의 액세스 유닛에서의 뷰의 깊이 부분, 즉, 액세스 유닛에서의 깊이 뷰는, "깊이 뷰 성분"이라고 지칭될 수도 있다. 그래서, 용어 "뷰 성분"은 하나의 액세스 유닛에서의 뷰를 지칭할 수 있고 동일한 액세스 유닛에서의 텍스처 뷰 성분 및 깊이 뷰 성분 양쪽 모두를 총칭하여 지칭할 수도 있다.

[0072] 위에서 언급했듯이, H.264/AVC에 대한 3DV 확장본, 즉 AVC 기반 3DV를 생성하려는 지속적인 노력이 있다. MVC 기반 3DV처럼, AVC 기반 3DV는 깊이 맵들을 제공한다. AVC 기반 3DV에서, 비디오 인코더 (20) 는 액세스 유닛의 다른 뷰들과 동일한 방식으로 깊이 맵을 인코딩할 수도 있다. MVC 기반 3DV와는 대조적으로, AVC

기반 3DV는 깊이 뷰 성분이 텍스처 뷰 성분에 기초하여 인코딩되는 것을 허용할 수도 있다. 이는 코딩 효율을 증가시킬 수도 있지만, 복잡도를 증가시킬 수도 있다. AVC 기반 3DV는 MVC와는 호환하지 않을 수도 있다.

[0073] AVC 기반 3DV에서, 비디오 인코더 (20) 는, 이용가능한 텍스처 및 깊이 뷰 성분들에 기초하여, 합성 텍스처 뷰 성분을 생성할 수도 있다. 다시 말하면, 인-루프 뷰 합성 예측 (VSP) 이 AVC 기반 3DV (및 다른 비디오 코딩 표준들) 에서 향상된 텍스처 코딩을 위해 지원된다. 합성 텍스처 뷰 성분은, 깊이 맵 및 하나 이상의 텍스처 뷰 성분들에 기초하여 합성된 텍스처 뷰 성분일 수도 있다. 다시 말하면, 현재 뷰의 코딩을 위한 VSP를 가능하게 하기 위해, 동일한 액세스 유닛의 이전에 코딩된 텍스처 및 깊이 뷰 성분들은 뷰 합성을 위해 사용될 수도 있다.

[0074] 예를 들어, 특정 텍스처 뷰 성분은 왼쪽 눈 텍스처 뷰 성분일 수도 있고 비디오 인코더 (20) 는 3DV 플레이백을 위해 오른쪽 눈 텍스처 뷰 성분을 생성할 수도 있다. 어떤 경우들에서는, 합성 텍스처 뷰 성분이 액세스 유닛간 예측 또는 뷰간 예측을 위한 참조 화상으로서 사용될 수도 있다. 따라서, VSP의 결과인 합성된 화상은 시간적 및 뷰간 참조 프레임들을 뒤따르는 초기 참조 화상 리스트들 (즉, RefPicList0 및/또는 RefPicList1) 에 포함될 수도 있다. 참조 화상들로서 사용되는 합성 텍스처 뷰 성분들은, 뷰 합성 참조 화상 (view synthesis reference picture; VSRP) 들, 뷰 합성 예측 (view synthesis prediction; VSP) 참조 화상들, 또는 간단히 VSP 화상들이라고 지칭될 수도 있다.

[0075] AVC 기반 3DV를 위한 일부 테스트 모델들에서, VSP는 합성된 화상을 참조 화상 리스트, 이를테면 RefPicList0 또는 RefPicList1에 추가함으로써 실현된다. 이 접근법에는 여러 잠재적인 문제들이 있다. 예를 들어, VSP 참조 화상들에 대한 모션 벡터들은 보통 0에 매우 가깝다. 다시 말하면, VSP 화상들 내의 참조 블록들에 대한 모션 벡터들은 거의 항상 0의 크기를 가진다. 그러나, 이러한 테스트 모델들의 접근법은 모션 벡터 차이들에 대한 콘텍스트를 덜 효율적이게 만들 수도 있다. 예를 들어, VSP 블록에 대한 모션 벡터는 통상 0이지만, 이웃 블록들이 시간적 화상들로 예측되면, 모션 벡터 예측은 0에 가깝지 않게 된다는 것이 도출될 수도 있으며, 그러므로, 불필요한 모션 벡터 차이가 시그널링될 필요가 있을 수도 있거나, 또는 그렇지 않으면, 모션 벡터 예측변수는 충분하지 않다.

[0076] 본 개시물의 일부 기법들에 따르면, 비디오 인코더 (20) 는, 비트스트림으로, 현재 비디오 유닛이 VSP 화상으로부터 예측되는지의 여부를 나타내는 신택스 엘리먼트를 시그널링할 수도 있다. 일부 예들에서, 현재 비디오 유닛은 MB, MB 파티션, 또는 다른 유형의 유닛일 수도 있다. H.264/AVC가 사용되는 예들에서, VSP 모드의 시그널링은, MB 또는 MB 파티션이 VSP 화상으로부터 예측되는지의 여부를 나타내기 위해 MB 또는 MB 파티션 레벨에서 도입된다. 비디오 유닛 (예컨대, MB 또는 MB 파티션) 의 VSP 모드는 비디오 유닛이 VSP 화상으로부터 예측되는지의 여부를 나타낸다.

[0077] 일부 예들에서, 현재 비디오 유닛이 VSP 화상으로부터 예측되는 경우, 비디오 인코더 (20) 는 현재 비디오 유닛에 대한 모션 정보를 시그널링하지 않는다. H.264/AVC가 사용되는 예들에서, 현재 비디오 유닛에 대한 모션 정보는 하나 이상의 참조 인덱스들 및 하나 이상의 MVD들을 포함할 수도 있다. HEVC가 사용되는 예들에서, 현재 비디오 유닛에 대한 모션 정보는 하나 이상의 참조 인덱스들, 하나 이상의 모션 벡터 후보 인덱스들, 하나 이상의 MVD들, 및 예측 방향 표시자를 포함할 수도 있다.

[0078] 더욱이, 현재 비디오 유닛이 VSP 화상으로부터 예측되는 경우, 비디오 인코더 (20) 는, 현재 비디오 유닛의 샘플 블록들과 병치되는 VSP 화상의 블록에 적어도 부분적으로 기초하여, 현재 비디오 유닛에 대한 예측 블록을 생성할 수도 있다. 비디오 인코더 (20) 는 현재 비디오 유닛에 대한 잔차 블록을 생성할 수도 있다. 잔차 블록은 현재 비디오 유닛의 샘플 블록 및 현재 비디오 유닛에 대한 예측 블록 사이의 차이들을 나타낼 수도 있다. 비디오 인코더 (20) 는 잔차 블록의 샘플들을 변환, 양자화, 및 엔트로피 인코딩할 수도 있다.

[0079] 비디오 디코더 (30) 는, 비트스트림으로부터 신택스 엘리먼트를 디코딩하고, 그 신택스 엘리먼트에 적어도 부분적으로 기초하여, 현재 비디오 유닛이 VSP 화상으로부터 예측되는지의 여부를 결정할 수도 있다. 현재 비디오 유닛이 VSP 화상으로부터 예측되는 경우, 비디오 디코더 (30) 는, 현재 비디오 유닛과 병치되는 VSP 화상의 블록에 적어도 부분적으로 기초하여, 현재 비디오 유닛에 대한 예측 블록을 생성할 수도 있다. 비디오 디코더 (30) 는 비트스트림으로부터 현재 비디오 유닛에 대한 모션 정보를 디코딩하는 일 없이, 현재 비디오 유닛에 대한 예측 블록을 생성할 수도 있다.

[0080] 따라서, 본 개시물의 기법들에 따라, 비디오 코더는 예측 블록이 VSP 화상 내의 병치된 블록과 일치하도록 현재

비디오 유닛 (예컨대, MB 또는 MB 파티션) 에 대한 예측 블록을 생성할 수도 있다. 다르게 말하면, 현재 비디오 유닛이 VSP 화상으로부터 예측되는 경우, 비디오 코더는 VSP 화상으로부터 현재 비디오 유닛의 병치된 블록을 복사한다.

[0081] 현재 비디오 유닛이 VSP 화상으로부터 예측되는 경우 비디오 디코더 (30) 가 현재 비디오 유닛에 대한 모션 정보를 디코딩하는 일 없이 현재 비디오 유닛에 대한 예측 블록을 생성할 수 있기 때문에, VSP 화상을 참조 화상 리스트 내에 포함시키는 것은 불필요할 수도 있다. 그런고로, 본 개시물의 기법들에 따라, VSP 화상을 참조 화상 리스트 속에 추가하는 대신, 비디오 코더 (이를테면 비디오 인코더 (20) 또는 비디오 디코더 (30)) 는 VSP 화상을 참조 화상 리스트 속에 추가하지 않는다.

[0082] 더욱이, 본 개시물의 기법들에 따라, 비디오 코더는, 이웃 블록 (예컨대 이웃 MB 또는 MB 파티션) 에 연관된 정보에 적어도 부분적으로 기초하여, 현재 비디오 유닛 (예컨대, 현재 MB 또는 MB 파티션) 의 모션 정보를 엔트로피 코딩하기 위한 코딩 콘텍스트를 선택할 수도 있다. 이웃 블록 (예컨대, 이웃 MB 또는 MB 파티션) 이 VSP 화상으로부터 예측되는 경우, 비디오 코더는 현재 비디오 유닛의 모션 정보를 엔트로피 인코딩하기 위한 코딩 콘텍스트를 선택함에 있어서 이웃 블록에 연관된 정보를 사용할 수 없게 된다고 결정할 수도 있다. 예를 들면, 특정 MB 또는 특정 MB 파티션의 모션 정보를 엔트로피 인코딩하기 위한 엔트로피 코딩 콘텍스트를 구성하는 경우, VSP 화상을 사용하는 MB 또는 MB 파티션은 엔트로피 코딩 콘텍스트의 선택에 사용할 수 없게 된다고 간주된다. 이웃 블록에 연관된 정보가 코딩 콘텍스트를 선택함에 있어서 사용할 수 없다고 비디오 코더가 결정하는 경우, 비디오 코더는 코딩 콘텍스트를 선택하기 위해 이웃 블록에 연관된 정보를 사용하지 않는다. 그런고로, VSP 화상을 사용하는 블록 (예컨대, MB 또는 MB 파티션) 은 현재 비디오 유닛에 대한 모션에 관련된 신택스 엘리먼트들을 위한 엔트로피 코딩 콘텍스트에 영향을 주지 않을 수도 있다.

[0083] 위에서 나타낸 바와 같이, 비디오 인코더 (20) 는 비디오 유닛들, 이를테면 MB들, MB 파티션들, 서브 MB 파티션들 등이 VSP 화상으로부터 예측되는지의 여부를 나타내는 신택스 엘리먼트들을 시그널링할 수도 있다. 예를 들어, 비디오 인코더 (20) 는 MB가 VSP 화상으로부터 예측되는지의 여부를 나타내는 신택스 엘리먼트를 생성할 수도 있으며, 비디오 인코더 (20) 는 MB 파티션이 VSP 화상으로부터 예측되는지의 여부를 나타내는 신택스 엘리먼트를 생성할 수도 있고, 서브 MB 파티션이 VSP 화상으로부터 예측되는지의 여부를 나타내는 신택스 엘리먼트를 생성할 수도 있다. 본 개시물의 기법들에 따라, 비디오 코더는, MB들, MB 파티션들, 및 서브 MB 파티션들이 VSP 화상들로부터 예측되는지의 여부를 나타내는 신택스 엘리먼트들을 엔트로피 인코딩하는 경우, 동일한 또는 상이한 코딩 콘텍스트들을 사용할 수도 있다. 예를 들면, VSP 모드를 나타내는 도입된 MB 또는 MB 파티션 레벨 신택스 엘리먼트들은 엔트로피 코딩을 위해 동일한 또는 상이한 콘텍스트들을 공유할 수도 있다.

[0084] 비디오 디코더 (30) 는, 모션 벡터 예측변수 및 MVD에 기초하여, 현재 비디오 유닛에 대한 모션 벡터를 예측 (즉, 결정) 할 수도 있다. H.264/AVC에서, 모션 벡터 예측변수는 이웃 블록이 이용가능한 경우 이웃 블록 (예컨대, 이웃 MB, MB 파티션, 또는 서브 MB 파티션) 의 모션 벡터로부터 도출될 수도 있다. 본 개시물의 기법들에 따라, 비디오 코더가 현재 비디오 유닛에 대한 모션 벡터를 예측하는 경우, VSP 화상으로부터 예측되는 블록 (예컨대, MB 또는 MB 파티션) 은 이용가능하지 않은 것으로서 간주될 수도 있다.

[0085] H.264/AVC 및 그것의 확장본들에서, 비디오 인코더 (20) 는 슬라이스에 대한 슬라이스 헤더 신택스 구조 및 그 슬라이스에 대한 슬라이스 데이터 신택스 구조를 생성할 수도 있다. 위에서 나타낸 바와 같이, 슬라이스는 정수 수의 MB들을 포함할 수도 있다. 슬라이스에 대한 슬라이스 데이터 신택스 구조는 슬라이스의 MB들에 대한 MB 계층 신택스 구조들을 포함할 수도 있다. MB에 대한 MB 계층 신택스 구조는 MB에 대한 신택스 엘리먼트들을 포함할 수도 있다. AVC 기반 3DV 테스트 모델 (3D-ATM) 의 일부 버전들에서, 슬라이스에 대한 슬라이스 데이터 신택스 구조는 슬라이스의 MB들에 대한 mb_skip_flag 신택스 엘리먼트들을 포함할 수도 있다. 슬라이스가 P 슬라이스 또는 SP 슬라이스인 경우, MB에 대한 mb_skip_flag 신택스 엘리먼트는 MB가 P_Skip 모드에서 인코딩되는지의 여부를 나타낸다. 슬라이스가 B 슬라이스인 경우, MB에 대한 mb_skip_flag 신택스 엘리먼트는 MB가 B_Skip 모드에서 인코딩되는지의 여부를 나타낸다. 예를 들어, MB에 대한 mb_skip_flag 신택스 엘리먼트가 1과 동일하고 슬라이스가 P 또는 SP 슬라이스이면, 비디오 디코더 (30) 는 MB에 대한 mb_type이 P_Skip이라고 (그리고 MB 유형이 총칭하여 P MB 유형이라고 지칭된다) 유추할 수도 있다. mb_skip_flag 신택스 엘리먼트가 1과 동일하고 슬라이스가 B 슬라이스이면, 비디오 디코더 (30) 는 MB에 대한 mb_type이 B_Skip이라고 (그리고 MB 유형이 총칭하여 B MB 유형이라고 지칭된다) 유추할 수도 있다. 이 예에서, MB에 대한 mb_skip_flag가 0과 동일하면, MB는 스킵되지 않는다.

[0086] MB가 P_Skip 모드에서 인코딩되는 경우, 비디오 디코더 (30) 는 MB의 예측 루마 및 크로마 블록들이 참조 화상

에서의 병치된 MB의 루마 및 크로마 블록들과 일치하도록 MB에 대한 예측 루마 및 크로마 블록들을 도출할 수도 있다. 그런고로, MB가 P_Skip 모드에서 인코딩되는 경우, MB에 대한 MB 계층 신택스 구조는 참조 화상의 RefPicList0 또는 RefPicList1 내의 로케이션을 식별하는 참조 인덱스를 포함할 수도 있다. 마찬가지로, MB가 B_Skip 모드에서 인코딩되는 경우, 비디오 디코더 (30) 는 2 개의 참조 화상들의 병치된 MB들로부터 MB의 예측 루마 및 크로마 블록들을 도출할 수도 있다. MB가 B_Skip 모드에서 인코딩되는 경우, MB에 대한 MB 계층 신택스 구조는 참조 화상들의 RefPicList0 및 RefPicList1 내의 로케이션들을 식별하는 참조 인덱스들을 포함할 수도 있다. MB가 P_Skip 모드 또는 B_Skip 모드에서 인코딩되는 경우, MB에 대한 MB 계층 신택스 구조는 다른 신택스 엘리먼트들, 이를테면 모션 정보, 변환 계수 레벨들 등을 특정하는 신택스 엘리먼트들을 포함할 필요가 없다.

[0087] 더욱이, 3D-ATM의 일부 버전들에서, 슬라이스에 대한 슬라이스 데이터 신택스 구조는 현재 MB가 VSP 화상으로부터 스킵되는지의 여부를 나타내는 VSP 스킵 신택스 엘리먼트를 포함할 수도 있다. 다르게 말하면, VSP 스킵 신택스 엘리먼트는 현재 MB에 대한 예측 루마 및 크로마 블록들이 VSP 화상의 병치된 루마 및 크로마 블록들과 일치함을 나타낼 수도 있다. 현재 MB가 VSP 화상으로부터 스킵됨을 VSP 스킵 신택스 엘리먼트가 나타내는 경우, 현재 MB는 VSP 화상으로부터 항상 단방향으로 예측된다. VSP 스킵 신택스 엘리먼트 및 mb_skip_flag 신택스 엘리먼트는 함께 시그널링될 수도 있고 현재 MB의 상측 및 좌측의 MB들에 기초하는 콘텍스트에 기초하여 엔트로피 인코딩될 수도 있다.

[0088] mb_skip_flag 신택스 엘리먼트 및 skip_from_vsp_flag 신택스 엘리먼트는 비교적 복잡한 방법으로 시그널링될 수도 있다. 본 개시물은 이 문제를 스킵 모드 시그널링 복잡도 문제라고 지칭할 수도 있다. 더욱이, 위에서 설명된 기법들의 일부에서, 단지 하나의 VSP 화상만이 전체 뷰 성분 또는 슬라이스에 대해 이용가능하다. 일부 이러한 기법들은 VSP 화상으로부터의 단방향 예측만을 지원할 수도 있고 다수의 VSP 화상들로부터의 예측은 지원되지 않는다. 본 개시물은 이 문제를 단방향 VSP 스킵 모드 문제라고 지칭할 수도 있다. 본 개시물의 부가적인 기법들은 이들 문제들을 해결할 수도 있다. 이들 부가적인 기법들은 또는 완전한 해법을 위해 함께 작동하거나 또는 함께 작동하지 않을 수도 있다.

[0089] 스킵 모드 시그널링 복잡도 문제를 해결하는 일 예의 기법에서, 하나의 VSP 화상만이 이용가능한 경우, mb_skip 플래그 신택스 엘리먼트 및 VSP 화상으로부터 스킵함을 나타내는 플래그 (예컨대, VSP 스킵 신택스 엘리먼트) 는 단일 신택스 엘리먼트로 결합된다. 이 결합된 단일 신택스 엘리먼트는 본원에서 mb_skip_idc 신택스 엘리먼트라고 지칭될 수도 있다. 더욱이, 이웃 블록들에 대한 mb_skip_idc 신택스 엘리먼트의 값들의 콘텍스트는 현재 MB에 대한 mb_skip_idc 신택스 엘리먼트를 예측하는데 사용될 수 있다.

[0090] 단방향 VSP 스킵 모드 문제를 해결하기 위한 제 1 예의 기법은, MB에 대한 적어도 하나의 예측변수 (즉, 참조 화상) 가 VSP 화상이고 각각의 예측 방향에 이용가능한 VSP 화상이 하나만 있는 경우 적용가능하다. 이 예에서, mb_part_vsp_flag 신택스 엘리먼트 및 sub_mb_vsp_flag 신택스 엘리먼트는 MB 파티션의 주어진 예측 방향이 VSP로부터 예측되는지의 여부를 나타내기 위하여 양쪽 모두의 방향들로 확장된다. 위에서 나타낸 바와 같이, MB 예측 신택스 구조의 mb_part_vsp_flag 신택스 엘리먼트는 현재 MB 파티션이 VSP 화상으로부터 예측되는지의 여부를 나타낸다. 서브 MB 예측 신택스 구조의 sub_mb_vsp_flag 신택스 엘리먼트는 현재 서브 MB 파티션이 VSP 화상으로부터 예측되는지의 여부를 나타낸다.

[0091] 단방향 VSP 스킵 모드 문제를 해결하기 위한 제 2 예의 기법은, MB에 대한 적어도 하나의 예측변수 (즉, 참조 화상) 가 VSP 화상이고 각각의 예측 방향에 이용가능한 VSP 화상이 하나만 있는 경우 적용가능하다. 이 예에서, VSP 화상은 참조 화상 리스트 (예컨대, RefPicList0 또는 RefPicList1) 내에 남아 있다. MB 또는 서브 MB 의 참조 인덱스 (예컨대, ref_idx 신택스 엘리먼트) 가 VSP 화상에 대응하는 경우, VSP 화상으로부터 자동으로 양방향 예측이 된다. 그러나, 본 개시물의 다른 기법들과 유사하게, 이러한 ref_idx (ref_idx가 RefPicListX에 속한다고 가정함) 에 대응하는 모션 벡터들은 시그널링되지 않고, MB 파티션의 RefPicListX에 관련된 모션 정보는 이용가능하지 않은 것으로서 간주된다. 이는 단방향 예측에도 적용될 수도 있다.

[0092] 단방향 VSP 스킵 모드 문제를 해결하는 다른 예에서, 다수의 VSP 화상들이 지원된다. 이 예에서, VSP가 사용된다는 것을 임의의 플래그가 나타내는 경우, 추가의 인덱스가 시그널링될 수 있다. 대안으로, 모든 가능한 VSP 화상들 및 정상 스킵 화상을 고려하고 그것들을 하나의 신택스 엘리먼트로 공동으로 시그널링하는 직접 표시가 주어진다.

[0093] 비록 본 개시물의 기법들이 주로 H.264/AVC를 참조하여 위에서 설명되었지만, 본 개시물의 기법들은 또한 HEVC, 그리고 구체적으로는 HEVC의 3DV 확장본에 적용가능할 수도 있다. HEVC의 3DV 확장본에서는, 문서 m23783에

서 제안된 바와 같이, 합성된 이미지 (예컨대, VSP 화상) 의 일부 지역들은 이용가능하지 않은데, 그 지역들이 다른 뷰들 (즉, VSP 화상이 합성되었던 뷰들) 로 가려졌기 때문이다. VSP 화상의 이러한 지역들은 가려지지 않은 지역들이라고 지칭될 수도 있는데, 그 지역들이 다른 뷰들로 숨겨졌기 (즉, 가려졌기) 때문이다. 가려지지 않은 지역들은 이진 맵, 즉 가용성 맵 상에서 식별되고 마킹되는데, 이는 코딩 및 디코딩 프로세스들을 제어한다. 비디오 코더 및 비디오 디코더 양쪽 모두는 가용성 맵을 사용하여 주어진 CU가 코딩되는지의 여부를 결정할 수도 있다. 그러나, 이러한 기법의 코딩 성능이 최적 미만이라는 것을 관찰이 보여주었다. 그런고로, HEVC 기반 3DV에서는, 주로 다음의 문제들로 인해 효율적인 VSP 메커니즘이 부족하다. 첫째, 모드로서의 뷰 합성은 일부 지역들에 대해서만 도움이 될 수 있다. 둘째, 뷰 합성 모드는 전체 HEVC 설계 속에 잘 통합되지 않는다.

[0094] 본 개시물의 기법들은 HEVC 기반 3DV에서의 VPS 지원을 위한 해결책을 제공할 수도 있다. 본 개시물의 하나 이상의 기법들에 따르면, 비디오 인코더 (20) 는 현재 CU가 VSP로 코딩되는지 (VSP 화상으로부터 예측되는지) 의 여부를 나타내기 위해 CU 레벨에서의 플래그를 시그널링할 수도 있다. 현재 CU가 VSP로 코딩되는 (즉, 현재 CU가 VSP CU인) 경우, VSP CU의 잔차는 다른 모드들과 동일한 방식으로 시그널링될 수 있다.

[0095] 더욱이, 일부 예들에서, 비디오 인코더 (20) 는, 각각의 PU에 대해, PU가 VSP로부터 예측되는지의 여부를 나타내는 신택스 엘리먼트 (예컨대, 플래그) 를 시그널링할 수도 있다. 그런 예들에서, 비디오 코더는 VSP로 CU에서의 하나의 PU를 예측 (즉, 그 PU에 대한 예측 블록을 생성) 할 수도 있는 한편 비디오 코더는 정상 인터 또는 인트라와 같은 다른 모드들로 CU의 다른 PU를 예측할 수도 있다. 더욱이, 그런 예들에서, 비디오 코더가 PU의 모션 정보를 엔트로피 코딩하기 위한 코딩 콘텍스트를 구성하는 경우, 비디오 코더는 CU 레벨 플래그 (즉, CU의 모든 PU들이 VSP 화상으로부터 예측되는지의 여부를 나타내는 신택스 엘리먼트) 및 PU 레벨 플래그 (즉, 단일 PU가 VSP 화상으로부터 예측되는지의 여부를 나타내는 신택스 엘리먼트) 에 대해 동일한 또는 상이한 콘텍스트 모델들을 사용할 수도 있다.

[0096] 비디오 코더가 CU 또는 PU에 대한 예측 블록을 VSP 화상으로부터 생성하는 경우, 비디오 코더는 VSP 화상으로부터 CU 또는 PU의 병치된 블록을 복사할 수도 있다. 다르게 말하면, CU 또는 PU에 대한 예측 블록은 VSP 화상의 병치된 블록에 일치할 수도 있다.

[0097] 위에서 설명된 바와 같이, 비디오 인코더 (20) 는 병합 모드 또는 AMVP 모드를 사용하여 현재 PU의 모션 정보를 시그널링할 수도 있다. 병합 모드 또는 AMVP 모드 중 어느 하나에서, 비디오 인코더 (20) 는 예측변수 후보들의 리스트 (즉, 후보 리스트) 를 생성할 수도 있다. 예측변수 후보들은 현재 PU와는 다른 PU들의 모션 정보를 특정할 수도 있다. 다른 PU들 중 하나가 이용가능하지 않은 경우, 비디오 인코더 (20) 는 다른 PU의 모션 정보를 특정하는 예측변수 후보를 포함하지 않는다. 본 개시물의 기법들에 따라, 비디오 코더가 모션 벡터 예측 동안 이웃 PU/CU를 가용성에 대해 체크하는 경우, 비디오 코더는 VSP 화상으로부터 예측되는 PU/CU (즉, VSP PU/CU) 를 이용가능하지 않은 것으로서 간주할 수도 있다.

[0098] 도 2는 본 개시물의 기법들을 구현할 수도 있는 일 예의 비디오 인코더 (20) 를 도시하는 블록도이다. 도 2는 설명의 목적으로 제공되고 본 개시물에서 폭넓게 예시되고 설명된 바와 같은 기법들의 제한으로서 고려되지 않아야 한다. 설명의 목적으로, 본 개시물은 H.264/AVC 코딩의 맥락에서 비디오 인코더 (20) 를 주로 설명한다. 그러나, 본 개시물의 기법들은 다른 코딩 표준들 또는 방법들, 이를테면 HEVC에 적용가능할 수도 있다.

[0099] 도 2의 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 기능성 컴포넌트들을 포함할 수도 있다.

[0100] 비디오 인코더 (20) 는 비디오 데이터를 수신한다. 비디오 데이터를 인코딩하기 위해, 비디오 인코더 (20) 는 비디오 데이터의 각각의 화상의 각각의 MB를 인코딩할 수도 있다. MB를 인코딩하기 위해, 예측 프로세싱 유닛 (100) 은 MB에 대한 파티셔닝 모드를 선택할 수도 있다. 비디오 인코더 (20) 는 MB에 대한 MB 계층 신택스 구조에서 mb_type 신택스 엘리먼트를 사용하여 MB에 대한 파티셔닝 모드를 시그널링할 수도 있다. MB에 대한 파티셔닝 모드는 MB의 루마 및 크로마 블록들이 MB의 MB 파티션들의 루마 및 크로마 블록들로 파티셔닝되는 방법을 나타낼 수도 있다.

[0101] 슬라이스는 정수 수의 MB들을 포함할 수도 있다. 더욱이, 슬라이스들은 I 슬라이스들, P 슬라이스들, SP 슬라이스들, SI 슬라이스들, 또는 B 슬라이스들일 수도 있다. MB가 I 슬라이스이면, MB의 모든 MB 파티션들은 인트라 예측된다. 그런고로, MB가 I 슬라이스에 있으면, 모션 추정 유닛 (122) 과 모션 보상 유닛 (124) 은 MB에 대해 인터 예측을 수행하지 않는다. SP 슬라이스는, 각각의 블록의 샘플 값들을 예측하기 위해 기껏해야 하나의 모션 벡터 및 참조 인덱스를 사용하여 예측 샘플들의 양자화와 함께 인트라 예측 또는 인터 예측을 사용하여 코딩될 수도 있는 슬라이스이다. SP 슬라이스는 그것의 디코딩된 샘플들이 다른 SP 슬라이스 또는 SI 슬라이스와 동일하게 구성될 수 있도록 코딩될 수 있다. SI 슬라이스는 인트라 예측만을 사용하여 그리고 예측 샘플들의 양자화를 사용하여 코딩된 슬라이스이다. SI 슬라이스는 그것의 디코딩된 샘플들이 SP 슬라이스와 동일하게 구성될 수 있도록 코딩될 수 있다.

[0102] 인터 예측 프로세싱 유닛 (120) 은 각각의 P, SP, 또는 B 슬라이스를 코딩하는 초반에 참조 화상 리스트 구성 프로세스를 수행할 수도 있다. 인터 예측 프로세싱 유닛 (120) 이 P 또는 SP 슬라이스를 코딩하고 있으면, 인터 예측 프로세싱 유닛 (120) 은 제 1 참조 화상 리스트 (예컨대, RefPicList0) 를 생성할 수도 있다. 인터 예측 프로세싱 유닛 (120) 이 B 슬라이스를 코딩하고 있으면, 인터 예측 프로세싱 유닛 (120) 은 제 1 참조 화상 리스트 (예컨대, RefPicList0) 를 생성하고 또한 제 2 참조 화상 리스트 (예컨대, RefPicList1) 를 생성할 수도 있다.

[0103] 비디오 인코더 (20) 가 P 슬라이스에서의 현재 비디오 유닛 (예컨대, MB 또는 MB 파티션) 을 코딩하고 있으면, 모션 추정 유닛 (122) 은 현재 비디오 유닛에 대한 참조 블록을 참조 화상 리스트 (예컨대, RefPicList0) 에서의 참조 화상들에서 검색할 수도 있다. 비디오 인코더 (20) 가 MVC-호환 3DV 또는 AVC-호환 3DV를 사용하는 예들에서, 참조 화상 리스트는 뷰간 참조 화상들을 포함할 수도 있다. 비디오 인코더 (20) 가 AVC-호환 3DV를 사용하는 예들에서, 참조 화상 리스트에서의 뷰간 참조 화상들은 깊이 맵에 기초하여 합성된 참조 화상들을 포함할 수도 있다. 현재 비디오 유닛의 참조 블록은 현재 비디오 유닛의 루마 및 크로마 블록들에 가장 밀접하게 대응하는 루마 샘플들의 블록 및 크로마 샘플들의 2 개의 대응하는 블록들을 포함할 수도 있다.

[0104] 모션 추정 유닛 (122) 은 P 슬라이스에서의 현재 비디오 유닛의 참조 블록을 포함하는 리스트 0에서의 참조 화상을 포함하는 RefPicList0에서의 참조 화상을 나타내는 참조 인덱스와 현재 비디오 유닛의 루마 샘플 블록 및 참조 블록 사이의 공간적 변위를 나타내는 모션 벡터를 생성할 수도 있다. 현재 비디오 유닛에 대한 모션 정보는 현재 비디오 유닛의 참조 인덱스 및 현재 비디오 유닛의 모션 벡터를 포함할 수도 있다. 모션 보상 유닛 (124) 은 현재 비디오 유닛의 모션 정보에 의해 나타내어진 참조 블록에 기초하여 현재 비디오 유닛에 대한 예측 블록들을 생성할 수도 있다.

[0105] 현재 비디오 유닛이 B 슬라이스에 있으면, 모션 추정 유닛 (122) 은 현재 비디오 유닛에 대해 단방향 인터 예측 또는 양방향 인터 예측을 수행할 수도 있다. 현재 비디오 유닛에 대해 단방향 인터 예측을 수행하기 위해, 모션 추정 유닛 (122) 은 현재 비디오 유닛에 대한 참조 블록을 RefPicList0 또는 제 2 참조 화상 리스트 (예컨대, RefPicList1) 의 참조 화상들에서 검색할 수도 있다. 비디오 인코더 (20) 가 MVC 또는 3DV를 사용하는 예들에서, RefPicList0 및/또는 RefPicList1은 뷰간 참조 화상들을 포함할 수도 있다. 모션 추정 유닛 (122) 은 참조 블록을 포함하는 참조 화상의 RefPicList0 또는 RefPicList1에서의 포지션을 나타내는 참조 인덱스와, 현재 비디오 유닛의 샘플 블록들 및 참조 블록 사이의 공간적 변위를 나타내는 모션 벡터를 생성할 수도 있다. 모션 추정 유닛 (122) 은 참조 화상이 RefPicList0에 있는지 또는 RefPicList1에 있는지를 나타내는 예측 방향 표시자를 또한 생성할 수도 있다.

[0106] 현재 비디오 유닛에 대한 양방향 인터 예측을 수행하기 위해, 모션 추정 유닛 (122) 은 현재 비디오 유닛에 대한 참조 블록을 RefPicList0에서의 참조 화상들에서 검색할 수도 있고 또한 현재 비디오 유닛에 대한 다른 참조 블록을 RefPicList1에서의 참조 화상들에서 검색할 수도 있다. 모션 추정 유닛 (122) 은 참조 블록들을 포함하는 참조 화상들의 RefPicList0 및 RefPicList1에서의 포지션들을 나타내는 참조 인덱스들을 생성할 수도 있다. 덧붙여서, 모션 추정 유닛 (122) 은 참조 블록들 및 현재 비디오 유닛의 루마 블록 사이의 공간적 변위들을 나타내는 모션 벡터들을 결정할 수도 있다. 현재 비디오 유닛에 대한 모션 정보는 참조 인덱스들 및 현재 비디오 유닛의 모션 벡터들을 포함할 수도 있다. 모션 보상 유닛 (124) 은 현재 비디오 유닛의 모션 정보에 의해 나타내어진 참조 블록들에 기초하여 현재 비디오 유닛에 대한 예측 블록들을 생성할 수도 있다.

[0107] 인트라 예측 프로세싱 유닛 (126) 은 현재 비디오 유닛에 대해 인트라 예측을 수행함으로써 그 현재 비디오 유닛에 대한 예측 데이터를 생성할 수도 있다. 현재 비디오 유닛에 대한 예측 데이터는 현재 비디오 유닛에 대한 예측 블록들과 다양한 신택스 엘리먼트들을 포함할 수도 있다. 인트라 예측 프로세싱 유닛 (126) 은 I

슬라이스들, P 슬라이스들, 및 B 슬라이스들에서의 비디오 유닛들에 대해 인트라 예측을 수행할 수도 있다.

[0108] 예측 프로세싱 유닛 (100) 은 현재 비디오 유닛에 대해 인터 예측 프로세싱 유닛 (120) 에 의해 생성된 예측 데이터 또는 그 현재 비디오 유닛에 대해 인트라 예측 프로세싱 유닛 (126) 에 의해 생성된 예측 데이터 중에서 현재 비디오 유닛에 대한 예측 데이터를 선택할 수도 있다. 일부 예들에서, 예측 프로세싱 유닛 (100) 은 예측 데이터의 세트들의 레이트/왜곡 메트릭들에 기초하여 현재 비디오 유닛에 대한 예측 데이터를 선택한다.

[0109] 잔차 생성 유닛 (102) 은 현재 비디오 유닛에 대한 예측 블록들에서의 샘플들을 현재 비디오 유닛의 샘플 블록들의 대응하는 샘플들로부터 감산함으로써 잔차 블록들을 생성할 수도 있다. 변환 프로세싱 유닛 (104) 은 하나 이상의 변환들을 잔차 블록에 적용함으로써 각각의 잔차 블록에 대한 변환 계수 블록들을 생성할 수도 있다. 변환 프로세싱 유닛 (104) 은 다양한 변환들을 잔차 블록에 적용할 수도 있다. 예를 들어, 변환 프로세싱 유닛 (104) 은 이산 코사인 변환 (DCT), 방향성 변환, 정수 변환, 웨이브릿 변환, 또는 개념적으로 유사한 변환을 잔차 블록에 적용할 수도 있다.

[0110] 양자화 유닛 (106) 은 변환 계수 블록에서의 변환 계수들을 양자화할 수도 있다. 양자화 프로세스는 변환 계수들의 일부 또는 전부에 연관된 비트 깊이를 감소시킬 수도 있다. 예를 들어, *n*-비트 변환 계수는 양자화 동안에 *m*-비트 변환 계수로 버림될 (rounded down) 수도 있으며, 여기서 *n*은 *m*보다 크다. 양자화 유닛 (106) 은 양자화 파라미터 (QP) 값에 기초하여 변환 계수 블록을 양자화할 수도 있다. 비디오 인코더 (20) 는 QP 값을 조정함으로써 변환 계수 블록들에 적용되는 양자화 정도를 조정할 수도 있다.

[0111] 역 양자화 유닛 (108) 과 역 변환 프로세싱 유닛 (110) 은 역 양자화 및 역 변환들을 변환 계수 블록에 각각 적용하여, 변환 계수 블록으로부터 잔차 블록을 복원할 수도 있다. 복원 유닛 (112) 은 복원된 잔차 블록들에서의 샘플들을 예측 프로세싱 유닛 (100) 에 의해 생성된 하나 이상의 예측 블록들로부터의 대응하는 샘플들에 가산하여 복원된 블록들을 생성할 수도 있다. 필터 유닛 (114) 은 블록화해제 동작을 수행하여 복원된 블록들에서의 블록화 아티팩트들을 감소시킬 수도 있다. 디코딩된 화상 버퍼 (116) 는, 필터 유닛 (114) 이 복원된 블록들에 대해 하나 이상의 블록화해제 동작들을 수행한 후에 복원된 블록들을 저장할 수도 있다. 모션 추정 유닛 (122) 과 모션 보상 유닛 (124) 은 후속 화상들의 비디오 유닛들에 대해 인터 예측을 수행하기 위해 복원된 블록들을 포함하는 참조 화상을 사용할 수도 있다. 덧붙여서, 인트라 예측 프로세싱 유닛 (126) 은 디코딩된 화상 버퍼 (116) 에서의 복원된 블록들을 사용하여 인트라 예측을 수행할 수도 있다.

[0112] 엔트로피 인코딩 유닛 (118) 은 비디오 인코더 (20) 의 기능성 컴포넌트들로부터 데이터를 수신할 수도 있다. 예를 들어, 엔트로피 인코딩 유닛 (118) 은 양자화 유닛 (106) 으로부터 변환 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, 비디오 인코더 (20) 는 그 데이터에 대해 CAVLC 동작, CABAC 동작, 가변 대 가변 (variable-to-variable; V2V) 길이 코딩 동작, 신택스 기반 콘텍스트 적응 이진 산술 코딩 (SBAC) 동작, 확률 간격 파티셔닝 엔트로피 (PIPE) 코딩 동작, 지수-골롬 코딩 동작, 또는 다른 유형의 엔트로피 인코딩 동작을 수행할 수도 있다.

[0113] 도 3은 본 개시물의 기법들을 구현할 수도 있는 일 예의 비디오 디코더 (30) 를 도시하는 블록도이다. 도 3은 설명의 목적으로 제공되고 본 개시물에서 폭넓게 예시되고 설명된 바와 같은 기법들로 제한하고 있지는 않다. 설명의 목적으로, 본 개시물은 H.264/AVC 코딩의 맥락에서 비디오 디코더 (30) 를 설명한다. 그러나, 본 개시물의 기법들은 다른 코딩 표준들 또는 방법들에 적용가능할 수도 있다.

[0114] 도 3의 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 기능성 컴포넌트들을 포함할 수도 있다.

[0115] 비디오 디코더 (30) 는 비디오 데이터를 수신할 수도 있다. 엔트로피 디코딩 유닛 (150) 은 비트스트림을 파싱하여 비트스트림으로부터 신택스 엘리먼트들을 디코딩할 수도 있다. 비트스트림을 파싱하는 부분으로서, 엔트로피 디코딩 유닛 (150) 은 비트스트림에서의 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 디코딩된 신택스 엘리먼트들에 기초하여 디코딩된

비디오 데이터를 생성 (즉, 그 비디오 데이터를 복원) 할 수도 있다. 비트스트림으로부터 디코딩된 신택스 엘리먼트들은 변환 계수 블록들을 나타내는 신택스 엘리먼트들을 포함할 수도 있다.

[0116] 역 양자화 유닛 (154) 은 변환 계수 블록들을 역 약자화, 즉 탈 양자화 (de-quantization) 할 수도 있다. 역 양자화 유닛 (154) 은 QP 값을 사용하여 양자화 정도를 결정하고, 비슷하게, 역 양자화 유닛 (154) 에 대해 적용할 역 양자화 정도를 결정할 수도 있다. 역 양자화 유닛 (154) 이 변환 계수 블록을 역 양자화한 후, 역 변환 프로세싱 유닛 (156) 은 잔차 블록을 생성하기 위하여 하나 이상의 역 변환들을 변환 계수 블록에 적용할 수도 있다. 예를 들어, 역 변환 프로세싱 유닛 (156) 은 역 DCT, 역 정수 변환, 역 카루넨-뢰베 변환 (Karhunen-Loeve transform; KLT), 역 회전 변환, 역 방향성 변환, 또는 다른 역 변환을 변환 계수 블록에 적용할 수도 있다.

[0117] 현재 비디오 유닛이 인트라 예측을 사용하여 인코딩되면, 인트라 예측 프로세싱 유닛 (166) 은 인트라 예측을 수행하여 현재 비디오 유닛에 대해 예측 블록들을 생성할 수도 있다. 예를 들어, 인트라 예측 프로세싱 유닛 (166) 은 비트스트림에서의 신택스 엘리먼트들에 기초하여 현재 비디오 유닛에 대한 인트라 예측 모드를 결정할 수도 있다. 인트라 예측 프로세싱 유닛 (166) 은 인트라 예측 모드를 사용하여 공간적으로 이웃하는 블록들에 기초하여 현재 비디오 유닛에 대한 예측 블록들을 생성할 수도 있다.

[0118] 모션 보상 유닛 (164) 은 비트스트림으로부터 디코딩된 신택스 엘리먼트들에 기초하여 제 1 참조 화상 리스트 (RefPicList0) 및 제 2 참조 화상 리스트 (RefPicList1) 를 구성할 수도 있다. 비트스트림이 MVC 호환 3DV 또는 AVC 기반 3DV를 사용하여 인코딩되는 예들에서, RefPicList0 및/또는 RefPicList1은 뷰간 참조 화상들을 포함할 수도 있다. 비트스트림이 AVC 기반 3DV를 사용하여 인코딩되는 예들에서, RefPicList0 및/또는 RefPicList1에서의 뷰간 참조 화상들은 깊이 맵들에 기초하여 합성된 참조 화상들을 포함할 수도 있다. 더욱이, 현재 비디오 유닛이 인터 예측을 사용하여 인코딩되면, 엔트로피 디코딩 유닛 (150) 은 현재 비디오 유닛에 대한 모션 정보를 디코딩할 수도 있다. 모션 보상 유닛 (164) 은, 현재 비디오 유닛의 모션 정보에 기초하여, 현재 비디오 유닛에 대한 하나 이상의 참조 블록들을 결정할 수도 있다. 모션 보상 유닛 (164) 은, 현재 비디오 유닛에 대한 하나 이상의 참조 블록들에 기초하여, 현재 비디오 유닛에 대한 예측 블록들을 생성할 수도 있다.

[0119] 복원 유닛 (158) 은 현재 비디오 유닛에 대한 잔차 블록들 및 현재 비디오 유닛에 대한 예측 블록들에 기초하여 현재 비디오 유닛에 대한 샘플 블록들을 복원할 수도 있다. 특히, 복원 유닛 (158) 은 잔차 블록들의 샘플들 (예컨대, 루마 또는 크로마 성분들) 을 예측 블록들의 대응하는 샘플들에 가산하여 현재 비디오 유닛의 샘플 블록들을 복원할 수도 있다.

[0120] 필터 유닛 (160) 은 블록화해제 동작을 수행하여 현재 비디오 유닛의 샘플 블록들에 연관된 블록화 아티팩트들을 감소시킬 수도 있다. 비디오 인코더 (20) 의 필터 유닛 (114) 은 필터 유닛 (160) 의 블록화해제 동작과 유사한 블록화해제 동작을 수행할 수도 있으며, 그래서 간결함을 위해 본 개시물은 필터 유닛 (160) 에 관한 블록화해제 동작만을 설명한다. 필터 유닛 (160) 이 H.264/AVC에서 블록화해제 동작 수행하는 경우, 필터 유닛 (160) 은 블록 에지들에 대해 필터링 프로세스를 수행할 수도 있다. 필터 유닛 (160) 은 필터링 프로세스를 수평 또는 수직 에지의 4x4 블록에 걸쳐 8 개 샘플들의 세트에 적용할 수도 있다. 이들 샘플들은 "입력 샘플들"이라고 지칭될 수도 있고, i=0..3이고 에지가 p_0 및 q_0 사이에 놓인 p_i 및 q_i로서 표시될 수도 있다.

필터 유닛 (160) 이 필터링 프로세스를 샘플들의 세트에 적용하는 경우, 필터 유닛 (160) 은 경계 필터링 강도 값 (bS) 을 결정할 수도 있다. 덧붙여서, 필터 유닛 (160) 은 그 블록들에 대한 양자화 파라미터들 (qP_p, qP_q) 을 결정할 수도 있다. 필터 유닛 (160) 은 그 다음에 샘플 값들, bS, 필터 오프셋들, qP_p 및 qP_q에 적어도 부분적으로 기초하여 임계치 도출 프로세스를 수행할 수도 있다. 임계치 도출 프로세스는 입력 샘플들이 필터링되는지의 여부를 나타내는 값을 반환할 수도 있다. 임계치 도출 프로세스는 또한 값 (indexA) 과 임계 변수들 (α 및 β)의 값들을 반환할 수도 있다. 필터 유닛 (160) 은 그 다음에 bS, α, β 및 인덱스 A에 적어도 부분적으로 기초하여, 입력 샘플들에 대한 필터 동작을 수행할 수도 있다.

[0121] 위에서 언급했듯이, 필터 유닛 (160) 은 경계 필터링 강도 값 (bS) 을 결정할 수도 있다. 필터 유닛 (160) 은 다양한 상이한 유형들의 정보에 기초하여 bS를 결정할 수도 있다. 예를 들어, 필터 유닛 (160) 은, 블록들의 예측 (예컨대, 인터 또는 인트라) 모드들, 블록들의 참조 인덱스들, 블록들이 단방향으로 또는 양방향으로 인터 예측되는지의 여부, 블록들의 모션 벡터들 등에 적어도 부분적으로 기초하여 bS를 결정할 수도 있다.

[0122] 비디오 디코더 (30) 는 복원된 블록들을 디코딩된 화상 버퍼 (162) 에 저장할 수도 있다. 디코딩된 화상 버

퍼 (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 복원된 블록들에 기초하여, 다른 CU들의 PU들에 대해 인트라 예측 또는 인터 예측 동작들을 수행할 수도 있다.

[0123] 위에서 논의된 바와 같이, 비트스트림은 일련의 NAL 유닛들을 포함할 수도 있다. NAL 유닛들은 비디오 데이터의 화상들의 코딩된 슬라이스들을 캡슐화하는 코딩된 슬라이스 NAL 유닛들을 포함할 수도 있다. 각각의 코딩된 슬라이스는 슬라이스 헤더 신택스 구조 및 슬라이스 데이터 신택스 구조를 포함한다. 본 개시물의 기법들의 제 1 구현예에 따라, 슬라이스 헤더 신택스 구조는 아래의 표 1의 예의 신택스에 부합할 수도 있다.

[0124] **표 1 - 슬라이스 헤더 신택스**

slice_header( ) {	C	기술어
first_mb_in_slice	2	ue(v)
slice_type	2	ue(v)
pic_parameter_set_id	2	ue(v)
frame_num	2	u(v)
if(!frame_mbs_only_flag) {		
field_pic_flag	2	u(1)
if(field_pic_flag)		
bottom_field_flag	2	u(1)
}		
if(nal_unit_type == 5)		
idr_pic_id	2	ue(v)
if(pic_order_cnt_type == 0) {		
pic_order_cnt_lsb	2	u(v)
if(pic_order_present_flag && !field_pic_flag)		
delta_pic_order_cnt_bottom	2	se(v)
}		
if(pic_order_cnt_type = = 1 && !delta_pic_order_always_zero_flag) {		
delta_pic_order_cnt[0 **]**	2	se(v)
if(pic_order_present_flag && !field_pic_flag)		
delta_pic_order_cnt[1 **]**	2	se(v)
}		
if(redundant_pic_cnt_present_flag)		
redundant_pic_cnt	2	ue(v)
if(slice_type == B)		
direct_spatial_mv_pred_flag	2	u(1)
if (seq_vsp_enabled_flag)		
slice_vsp_flag	2	u(1)
if(slice_type = = P | | slice_type = = SP | | slice_type = = B) {		
num_ref_idx_active_override_flag	2	u(1)
if(num_ref_idx_active_override_flag) {		
num_ref_idx_l0_active_minus1	2	ue(v)
if(slice_type == B)		
num_ref_idx_l1_active_minus1	2	ue(v)
}		
}		
ref_pic_list_reordering( )	2	
if((weighted_pred_flag && (slice_type = = P || slice_type = = SP)) || (weighted_bipred_idc == 1 && slice_type == B))		
pred_weight_table( )	2	
if(nal_ref_idc != 0)		
dec_ref_pic_marking( )	2	

[0125]

if(entropy_coding_mode_flag && slice_type != I && slice_type != SI)		
cabac_init_idc	2	ue(v)
slice_qp_delta	2	se(v)
if(slice_type == SP || slice_type == SI) {		
if(slice_type == SP)		
sp_for_switch_flag	2	u(1)
slice_qs_delta	2	se(v)
}		
if(deblocking_filter_control_present_flag) {		
disable_deblocking_filter_idc	2	ue(v)
if(disable_deblocking_filter_idc != 1) {		
slice_alpha_c0_offset_div2	2	se(v)
slice_beta_offset_div2	2	se(v)
}		
}		
if(num_slice_groups_minus1 > 0 && slice_group_map_type >= 3 && slice_group_map_type <= 5)		
slice_group_change_cycle	2	u(v)
}		

[0126]

[0127] 위의 표 1 및 본 개시물의 다른 신택스 표들에서, 유형 기술자 ue(v) 를 갖는 신택스 엘리먼트들은, 좌측 비트를 첫째로 하는 0차 지수 골롬 (Exp-Golomb) 코딩을 사용하여 인코딩된 가변 길이 부호없는 정수들일 수도 있다. 표 1 및 다음의 표들의 예에서, *n*이 음이 아닌 정수인 형태 u(*n*) 의 기술자를 갖는 신택스 엘리먼트는, 길이 *n*의 부호없는 값들이다.

[0128] 표 1의 seq_vsp_enabled_flag 변수는 VSP가 코딩된 비디오 시퀀스에 허용되는지의 여부를 나타낸다. VSP가 슬라이스를 포함하는 코딩된 비디오 시퀀스에 허용됨을 seq_vsp_enabled_flag가 나타내면, 그 슬라이스에 대한 슬라이스 헤더 신택스 구조는 slice_vsp_flag 신택스 엘리먼트를 포함할 수도 있다. slice_vsp_flag 신택스 엘리먼트는 VSP가 슬라이스에 허용되는지의 여부를 나타낸다. slice_vsp_flag가 존재하지 않는 경우, 비디오 디코더 (30) 는 slice_vsp_flag가 0과 동일함을 유추 (즉, 자동으로 결정) 수도 있는데, 이는 VSP가 슬라이스에 허용되지 않을 의미한다.

[0129] 슬라이스 헤더 신택스 구조를 포함하는 NAL 유닛은 또한 슬라이스 데이터 신택스 구조를 포함할 수도 있다. 슬라이스 헤더 신택스 구조 및 슬라이스 데이터 신택스 구조가 동일한 NAL 유닛에 의해 캡슐화되면, 슬라이스 헤더 신택스 구조는 슬라이스 데이터 신택스 구조에 대응하고, 반대의 경우도 마찬가지이다. 슬라이스 데이터 신택스 구조는 슬라이스의 MB들에 대한 MB 계층 신택스 구조들을 포함할 수도 있다. 아래의 표 2는 본 개시물의 기법들의 제 1 구현예에 따른 슬라이스 데이터 신택스 구조를 위한 일 예의 신택스이다.

[0130] **표 2 - 슬라이스 데이터 신택스**

slice_data( ) {	C	기술어
if(entropy_coding_mode_flag)		
while(!byte_aligned( ))		
cabac_alignment_one_bit	2	f(1)
CurrMbAddr = first_mb_in_slice * (1 + MbaffFrameFlag)		
moreDataFlag = 1		
prevMbSkipped = 0		
do {		
if(slice_type != I && slice_type != SI)		
if(!entropy_coding_mode_flag) {		
mb_skip_run	2	ue(v)
prevMbSkipped = (mb_skip_run > 0)		
for(i=0; i<mb_skip_run; i++)		
CurrMbAddr = NextMbAddress(CurrMbAddr)		
moreDataFlag = more_rbsp_data( )		
} else {		
mb_skip_flag	2	ae(v)
if (slice_vsp_flag && mb_skip_flag && VspRefExist)		
skip_from_vsp_flag		ae(v)
moreDataFlag = !mb_skip_flag		
}		
if(moreDataFlag) {		
if(MbaffFrameFlag && (CurrMbAddr % 2 == 0 || (CurrMbAddr % 2 == 1 && prevMbSkipped)))		
mb_field_decoding_flag	2	u(1) | ae(v)
macroblock_layer( )	2 | 3 | 4	
}		
if(!entropy_coding_mode_flag)		
moreDataFlag = more_rbsp_data( )		
else {		
if(slice_type != I && slice_type != SI)		
prevMbSkipped = mb_skip_flag		
if(MbaffFrameFlag && CurrMbAddr % 2 == 0)		
moreDataFlag = 1		
else {		
end_of_slice_flag	2	ae(v)
moreDataFlag = !end_of_slice_flag		
}		
}		
CurrMbAddr = NextMbAddress(CurrMbAddr)		
} while(moreDataFlag)		
}		

[0131]

[0132] 표 2의 예의 신택스에서, 슬라이스 데이터 신택스 구조는 mb_skip_flag 신택스 엘리먼트를 포함할 수도 있다. mb_skip_flag 신택스 엘리먼트는 MB가 스킵 모드를 사용하여 인코딩되는지의 여부를 나타낸다. 전체 MB는 VSP 화상으로부터 예측될 수도 있거나 또는 정상 스킵 모드가 사용될 수도 있다. 다르게 말하면, MB에 대한 예측 블록은 VSP 화상 또는 다른 화상으로부터 생성될 수도 있다. 예를 들어, 1과 동일한 mb_skip_flag는 스킵 모드가 MB를 인코딩하는데 사용된다는 것을 나타낼 수도 있다. 이 예에서, 0과 동일한 mb_skip_flag는 스킵 모드가 MB를 인코딩하는데 사용되지 않는다는 것을 나타낼 수도 있다.

[0133] 더욱이, 표 2의 예의 신택스에서, 슬라이스 데이터 신택스 구조는, VSP가 슬라이스에 허용됨을 대응하는 슬라이스 헤더 신택스 구조의 slice_vsp_flag가 나타내며, MB가 스킵 모드를 사용하여 인코딩되고, RefPicList0 또는 RefPicList1에 VSP 화상이 있는 경우, MB에 대한 skip_from_vsp_flag를 포함할 수도 있다. 표 2에서, 변수 VspRefExist는 RefPicList0 또는 RefPicList1에 VSP 화상이 있는지를 나타낸다. 일부 예들에서, VspRefExist 조건이 생략된다. MB에 대한 skip_from_vsp_flag는 전체 MB가 VSP 화상으로부터 예측되는지의

여부를 나타낸다. 예를 들어, 1과 동일한 skip_from_vsp_flag는, 스킵 모드가 사용되는 경우, 전체 MB가 VSP 화상으로부터 예측된다는 것을 나타낼 수도 있다. 이 예에서, 0과 동일한 skip_from_vsp_flag는 정상 스킵 모드를 나타낼 수도 있다. skip_from_vsp_flag가 슬라이스 데이터 신택스 구조에 존재하지 않는 경우, 비디오 디코더 (30) 는, MB가 정상 스킵 모드를 사용하여 인코딩된다고 유추할 수도 있다 (예컨대, 비디오 디코더 (30) 는 skip_from_vsp_flag가 0과 동일하다고 유추할 수도 있다).

[0134] 위의 표 2에 도시된 바와 같이, 슬라이스 데이터는 하나 이상의 MB 계층 신택스 구조들을 포함할 수도 있다. MB 계층 신택스 구조는 인코딩된 MB를 포함할 수도 있다. 아래의 표 3은 본 개시물의 기법들의 제 1 구현예에 따른 MB 계층 신택스 구조에 대한 일 예의 신택스이다.

[0135] **표 3 - 매크로블록 계층 신택스**

macroblock_layer( ) {	C	기술어
if (slice_vsp_flag && VspRefExist)		
vsp_mb_flag	2	ae(v)
if (! vsp_mb_flag)		
mb_type	2	ue(v) | ae(v)
if(mb_type == I_PCM) {		
while(!byte_aligned( ))		
pcm_alignment_zero_bit	3	f(1)
for(i = 0; i < 256; i++)		
pcm_sample_luma[i **]**	3	u(v)
for(i = 0; i < 2 * MbWidthC * MbHeightC; i++)		
pcm_sample_chroma[i **]**	3	u(v)
} else {		
noSubMbPartSizeLessThan8x8Flag = 1		
if(mb_type != I_NxN && MbPartPredMode(mb_type, 0) != Intra_16x16 && NumMbPart(mb_type) == 4) {		
if (!vsp_mb_flag)		
sub_mb_pred(mb_type)	2	
for(mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++)		
if(sub_mb_type[mbPartIdx] != B_Direct_8x8) {		
if(NumSubMbPart(sub_mb_type[mbPartIdx]) > 1)		
noSubMbPartSizeLessThan8x8Flag = 0		
} else if(!direct_8x8_inference_flag)		
noSubMbPartSizeLessThan8x8Flag = 0		
} else {		
if(transform_8x8_mode_flag && mb_type == I_NxN)		
transform_size_8x8_flag	2	u(1) | ae(v)
if (!vsp_mb_flag)		
mb_pred(mb_type)	2	
}		
if(MbPartPredMode(mb_type, 0) != Intra_16x16) {		
coded_block_pattern	2	me(v) | ae(v)
if(CodedBlockPatternLuma > 0 && transform_8x8_mode_flag && mb_type != I_NxN && noSubMbPartSizeLessThan8x8Flag && (vsp_mb_flag || (!vsp_mb_flag && (mb_type != B_Direct_16x16 || direct_8x8_inference_flag))))		
transform_size_8x8_flag	2	u(1) | ae(v)
}		
if(CodedBlockPatternLuma > 0 || CodedBlockPatternChroma > 0 || MbPartPredMode(mb_type, 0) == Intra_16x16) {		
mb_qp_delta	2	se(v) | ae(v)
residual(0, 15)	3 | 4	
}		
}		

[0136]

}		

[0137]

[0138] 표 3의 예의 신택스에서, VSP가 MB를 포함하는 슬라이스에서 허용되고 RefPicList0 또는 RefPicList1에 VSP 화

상이 있는 경우, MB에 대한 MB 계층 신택스 구조는 vsp_mb_flag 신택스 엘리먼트를 포함할 수도 있다. 다르게 말하면, MB 계층 신택스 구조는, slice_vsp_flag가 1과 동일하고 VspRefExist가 1과 동일한 경우 vsp_mb_flag를 포함할 수도 있다. 일부 예들에서, VspRefExist 조건이 생략된다. vsp_mb_flag 신택스 엘리먼트는 전체 MB가 VSP 화상으로부터 예측되는지의 여부를 나타낼 수도 있다. 예를 들어, 1과 동일한 vsp_mb_flag는 전체 MB가 VSP 화상으로부터 예측된다는 것을 나타낼 수도 있다. 이 예에서, 0과 동일한 vsp_mb_flag는 전체 MB가 다른 모드들에 의해 예측될 수도 있다는 것을 나타낸다.

[0139] 전체 MB가 VSP 화상으로부터 예측되는 경우 (예컨대, vsp_mb_flag가 1과 동일한 경우), mb_type 신택스 엘리먼트는 MB 계층 신택스 구조로 시그널링되지 않는다. mb_type 신택스 엘리먼트는 MB의 유형을 나타낸다. 더욱이, vsp_mb_flag가 존재하지 않는 경우, 비디오 디코더 (30) 는 전체 MB가 다른 모드들에 의해 예측될 수도 있다는 것을 나타낸다고 vsp_mb_flag를 유추할 수도 있다 (예컨대, 비디오 디코더 (30) 는 vsp_mb_flag가 0과 동일하다고 유추할 수도 있다).

[0140] 더욱이, 표 3의 예는 noSubMbPartLessThan8x8Flag 변수를 포함한다. 현재 MB의 MB 파티션들의 수가 4 미만이면, noSubMbPartLessThan8x8Flag는, H.264/AVC에서처럼, 참이다. 그러나, 현재 MB의 MB 파티션들의 수가 4와 동일하면 (즉, NumMbPart(mb_type) = 4이면), noSubMbPartLessThan8x8Flag는 0과 동일한 mb_part_vsp_flag를 갖는 8x8 MB 파티션들만을 체크함으로써 도출될 수도 있다. 예를 들면, 임의의 MB 파티션이 8x8보다 작은 서브 MB 파티션을 가지면, noSubMbPartLessThan8x8Flag는 거짓일 수도 있다. 그렇지 않으면, noSubMbPartLessThan8x8Flag는 참일 수도 있다 (즉, noSubMbPartLessThan8x8Flag = 1).

[0141] 더욱이, 표 3의 예의 신택스에서 도시된 바와 같이, 비디오 인코더 (20) 는, 전체 MB가 VSP 화상으로부터 예측되는지의 여부에 적어도 부분적으로 기초하여, 신택스 엘리먼트 (transform_size_8x8_flag) 를 MB 계층 신택스 구조에 포함시킬지의 여부를 결정할 수도 있다. 비슷하게, 비디오 디코더 (30) 는, 전체 MB가 VSP 화상으로부터 예측되는지의 여부에 기초하여, MB 계층 신택스 구조로부터 신택스 엘리먼트를 디코딩할지의 여부를 결정할 수도 있다. transform_size_8x8_flag는, 현재 MB에 대해, 잔차 8x8 또는 4x4 블록들에 대한 블록화해제 필터 프로세스 전에 변환 계수 디코딩 프로세스 및 화상 구성 프로세스가 루마 샘플들을 위해, 그리고, 어떤 경우들에서는, Cb 및 Cr 샘플들을 위해 호출되는지의 여부를 나타낼 수도 있다.

[0142] 표 3의 예의 신택스에서 도시된 바와 같이, MB 계층 신택스 구조는, 전체 MB가 VSP 화상으로부터 예측되지 않으면 (예컨대, vsp_mb_flag가 1과 동일하지 않으면), MB 예측 신택스 구조 (mb_pred(mb_type)) 를 포함할 수도 있다. 반대로, MB 계층 신택스 구조는, 전체 MB가 VSP 화상으로부터 예측되면 (예컨대, vsp_mb_flag가 1과 동일하면), MB 예측 신택스 구조를 포함하지 않는다. 다른 예들에서, MB 계층 신택스 구조는, MB가 VSP 화상으로부터 예측되는지의 여부에 상관없이, MB 예측 신택스 구조를 포함할 수도 있다.

[0143] 위의 표 3의 예에 도시된 바와 같이, MB 계층 신택스 구조는 MB 예측 신택스 구조를 포함할 수도 있다. MB 예측 신택스 구조는 MB에 대한 모션 정보를 포함할 수도 있다. 아래의 표 4는, 본 개시물의 기법들의 제 1 구현예에 따른 MB 예측 신택스 구조를 위한 일 예의 신택스이다.

[0144] **표 4 - 매크로블록 예측 신택스**

mb_pred(mb_type) {	C	기술어
if(MbPartPredMode(mb_type, 0) == Intra_4x4 || MbPartPredMode(mb_type, 0) == Intra_16x16) {		
if(MbPartPredMode(mb_type, 0) == Intra_4x4)		
for(luma4x4BlkIdx=0; luma4x4BlkIdx<16; luma4x4BlkIdx++) {		
prev_intra4x4_pred_mode_flag[luma4x4BlkIdx **]**	2	u(1) | ae(v)
if(!prev_intra4x4_pred_mode_flag**[** luma4x4BlkIdx **]**)		
rem_intra4x4_pred_mode[luma4x4BlkIdx **]**	2	u(3) | ae(v)
}		
intra_chroma_pred_mode	2	ue(v) | ae(v)
} else if(MbPartPredMode(mb_type, 0) != Direct) {		
for(mbPartIdx = 0; mbPartIdx < NumMbPart(mb_type); mbPartIdx++)		
if (slice_vsp_flag && NumMbPart(mb_type) != 1 && VspRefExist) //vsp_mb_flag is not 1		
mb_part_vsp_flag[mbPartIdx **]**	2	ae(v)
if(!**mb_part_vsp_flag[** mbPartIdx **]** &&(num_ref_idx_l0_active_minus1 > 0 || mb_field_decoding_flag) && MbPartPredMode(mb_type, mbPartIdx) != Pred_L1)		
ref_idx_l0[mbPartIdx **]**	2	te(v) | ae(v)
for (mbPartIdx = 0; mbPartIdx < NumMbPart(mb_type) && !mb_part_vsp_flag[mbPartIdx]; mbPartIdx++)		
if((num_ref_idx_l1_active_minus1 > 0 || mb_field_decoding_flag) && MbPartPredMode(mb_type, mbPartIdx) != Pred_L0)		
ref_idx_l1[mbPartIdx **]**	2	te(v) | ae(v)
for(mbPartIdx = 0; mbPartIdx < NumMbPart(mb_type); mbPartIdx++)		
if(!**mb_part_vsp_flag[** mbPartIdx **]** && MbPartPredMode (mb_type, mbPartIdx) != Pred_L1)		
for(compIdx = 0; compIdx < 2; compIdx++)		
mvd_l0[mbPartIdx **][** 0 **][** compIdx **]**	2	se(v) | ae(v)
for(mbPartIdx = 0; mbPartIdx < NumMbPart(mb_type); mbPartIdx++)		
if(!**mb_part_vsp_flag[** mbPartIdx **]** && MbPartPredMode(mb_type, mbPartIdx) != Pred_L0)		
for(compIdx = 0; compIdx < 2; compIdx++)		
mvd_l1[mbPartIdx **][** 0 **][** compIdx **]**	2	se(v) | ae(v)
}		
}		

[0145]

[0146] 표 4의 예의 신택스에서, MB 예측 신택스 구조는 신택스 엘리먼트들의 어레이 (mb_part_vsp_flag[ ]) 를 포함할 수도 있다. mb_part_vsp_flag[ ]에서의 각각의 엔트리는 MB의 상이한 MB 파티션이 VSP 화상으로부터 예측되는지의 여부를 나타낸다. MB 예측 신택스 구조는, VSP가 특정 MB 파티션을 포함하는 슬라이스에 대해 가능하게 되며 (예컨대, slice_vsp_flag = 1), MB의 MB 파티션들의 수가 1과 동일하지 않고, RefPicList0 또는 RefPicList1에 VSP 화상이 없으면 (예컨대, VspRefExist = 1이면), 특정 MB 파티션에 대한 mb_part_vsp_flag를 포함할 수도 있다. 다른 예들에서, MB 예측 신택스 구조는, VSP가 특정 MB 파티션을 포함하는 슬라이스에 대해 가능하게 되면 (예컨대, slice_vsp_flag = 1이면) mb_part_vsp_flag를 포함할 수도 있고, MB의 MB 파티션들의 수가 1과 동일하지 않은지 또는 VspRefExist = 1인지의 여부를 평가하지 않는다.

[0147] 일부 예들에서, 1과 동일한 mb_part_vsp_flag[mbPartIdx]는 특정 MB 파티션이 VSP 화상으로부터 예측된다는 것을 나타낼 수도 있으며, 여기서 mbPartIdx는 특정 MB 파티션의 인덱스이다. 이 예에서, 0과 동일한 mb_part_vsp_flag[mbPartIdx]는 전체 MB 파티션이 VSP 화상으로부터 예측되지 않는다는 것을 나타낼 수도 있으며, 여기서 mbPartIdx는 MB 파티션의 인덱스이다. 일부 예들에서, mb_part_vsp_flag[mbPartIdx]가 존재하지 않는 경우, 비디오 디코더 (30) 는, 전체 MB 파티션이 VSP 화상으로부터 예측되지 않는다는 것을 나타낸다고 mb_part_vsp_flag[mbPartIdx]를 유추할 수도 있다 (예컨대, 비디오 디코더 (30) 는 mb_part_vsp_flag[mbPartIdx]가 0과 동일하다고 유추할 수도 있다).

[0148] 표 4에 도시된 바와 같이, mbPartIdx가 특정 MB 파티션의 인덱스이면, 특정 MB 파티션이 VSP 화상으로부터 예측된다는 것을 mb_part_vsp_flag[mbPartIdx]가 나타내는 경우, MB 예측 신택스 구조는, 특정 MB 파티션에 대한

참조 인덱스들 (ref_idx_l0 및 ref_idx_l1) 또는 모션 벡터 차이들 (mvd_l0 및 mvd_l1) 을 포함하지 않는다. 다른 예들에서, MB 예측 신택스 구조는, 특정 MB 파티션이 VSP 화상으로부터 예측되는지의 여부에 상관없이, ref_idx_l0 또는 ref_idx_l1을 포함할 수도 있다.

[0149] 더욱이, MB가 VSP 화상으로부터 전적으로 예측되지 않으면 (예컨대, MB의 vsp_mb_flag가 1과 동일하지 않으면), MB에 대한 MB 계층 신택스 구조는 서브 MB 예측 신택스 구조를 포함할 수도 있다. 서브 MB 예측 신택스 구조들은 MB 파티션들에 대한 모션 정보 (예컨대, 모션 벡터들, 참조 인덱스들 등) 를 포함할 수도 있다. 아래의 표 5는, 본 개시물의 기법들의 제 1 구현예에 따른 서브 MB 예측 신택스 구조를 위한 일 예의 신택스를 도시한다.

[0150] **표 5 - 서브 매크로블록 예측 신택스**

sub_mb_pred(mb_type) {	C	기술어
for(mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++) {		
if (slice_vsp_flag && VspRefExist) // vsp_mb_flag is not 1		
sub_mb_vsp_flag[mbPartIdx **]**		
if (!sub_mb_vsp_flag[mbPartIdx])		
sub_mb_type[mbPartIdx **]**	2	ue(v) | ae(v)
}		
for(mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++)		
if(!sub_mb_vsp_flag[mbPartIdx] &&(num_ref_idx_l0_active_minus1 > 0 || mb_field_decoding_flag) && mb_type != P_8x8ref0 && sub_mb_type[mbPartIdx] != B_Direct_8x8 && SubMbPredMode(sub_mb_type[mbPartIdx]) != Pred_L1)		
ref_idx_l0[mbPartIdx **]**	2	te(v) | ae(v)
for(mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++)		
if(!sub_mb_vsp_flag[mbPartIdx] && (num_ref_idx_l1_active_minus1 > 0 || mb_field_decoding_flag) && sub_mb_type[mbPartIdx] != B_Direct_8x8 && SubMbPredMode(sub_mb_type[mbPartIdx]) != Pred_L0)		
ref_idx_l1[mbPartIdx **]**	2	te(v) | ae(v)
for(mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++)		
if(!sub_mb_vsp_flag[mbPartIdx] &&(sub_mb_type[mbPartIdx] != B_Direct_8x8 && SubMbPredMode(sub_mb_type[mbPartIdx]) != Pred_L1)		
for(subMbPartIdx = 0; subMbPartIdx < NumSubMbPart(sub_mb_type[mbPartIdx]); subMbPartIdx++)		
for(compIdx = 0; compIdx < 2; compIdx++)		
mvd_l0[mbPartIdx **][** subMbPartIdx **][** compIdx **]**	2	se(v) | ae(v)
for(mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++)		
if(!sub_mb_vsp_flag[mbPartIdx] &&sub_mb_type[mbPartIdx] != B_Direct_8x8 && SubMbPredMode(sub_mb_type[mbPartIdx]) != Pred_L0)		
for(subMbPartIdx = 0; subMbPartIdx < NumSubMbPart(sub_mb_type[mbPartIdx]); subMbPartIdx++)		
for(compIdx = 0; compIdx < 2; compIdx++)		
mvd_l1[mbPartIdx **][** subMbPartIdx **][** compIdx **]**	2	se(v) | ae(v)
}		

[0151]

[0152] 표 5의 예의 신택스에서, 서브 MB 예측 신택스 구조는 sub_mb_vsp_flag[ ] 신택스 엘리먼트들을 포함할 수도 있다. 각각의 sub_mb_vsp_flag[ ]는 상이한 MB 파티션 (예컨대, 8x8) 이 VSP 화상으로부터 예측되는지의 여부를 나타낸다. 서브 MB 예측 신택스 구조는, VSP가 특정 MB 파티션을 포함하는 슬라이스에 대해 가능하게 되고 (예컨대, slice_vsp_flag = 1) VSP 화상이 RefPicList0 또는 RefPicList1 내에 있는 경우 (예컨대, VspRefExist = 1인 경우), 특정 MB 파티션에 대한 sub_mb_vsp_flag를 포함할 수도 있다. 다른 예들에서, VspRefExist = 1의 조건은 결정되지 않는다.

[0153] 하나의 예에서, 1과 동일한 sub_mb_vsp_flag[mbPartIdx]는 특정 MB 파티션이 VSP 화상으로부터 예측된다는 것을 나타낼 수도 있으며, 여기서 mbPartIdx는 특정 MB 파티션의 인덱스이다. 이 예에서, 0과 동일한 sub_mb_vsp_flag[mbPartIdx]는 특정 MB 파티션의 전체가 VSP 화상으로부터 예측되지 않는다는 것을 나타낼 수도 있다. sub_mb_vsp_flag[mbPartIdx]가 존재하지 않는 경우, 특정 MB 파티션의 전체는 VSP 화상으로부터 예측되지 않는다 (예컨대, 비디오 디코더 (30) 는 sub_mb_vsp_flag[mbPartIdx]가 0과 동일하다고 유추할 수도 있다).

[0154] 더욱이, 표 5의 예의 신택스에서 도시된 바와 같이, mbPartIdx가 특정 MB 파티션의 인덱스이고 mb_part_vsp_flag[mbPartIdx]는 특정 MB 파티션이 VSP 화상으로부터 예측된다는 것을 나타내는 경우, 서브 MB 예측 신택스 구조는 특정 MB 파티션의 서브 MB 유형을 특정하는 신택스 엘리먼트 (sub_mb_type[mbPartIdx]) 를 포함하지 않는다. 더구나, 특정 MB 파티션이 VSP 화상으로부터 예측된다는 것을 mb_part_vsp_flag[mbPartIdx]가 나타내는 경우, 서브 MB 예측 구조는 특정 MB 파티션에 대한 참조 인덱스들 (ref_idx_l0[mbPartIdx] 및 ref_idx_l1[mbPartIdx]) 을 포함하지 않는다. 더욱이, 특정 MB 파티션이 VSP 화상으로부터 예측된다는 것을 mb_part_vsp_flag[mbPartIdx]가 나타내는 경우, 서브 MB 예측 신택스 구조는 특정 MB 파티션에 대한 모션 벡터 차이들 (mvd_l0[mbPartIdx] 및 mvd_l1[mbPartIdx]) 을 포함하지 않는다.

[0155] 위의 표 1 내지 표 5는, vsp_mb_flag, mb_part_vsp_flag, sub_mb_vsp_flag, 및 skip_from_vsp_flag 신택스 엘리먼트들을 포함한다. 비디오 디코더 (30) 는 vsp_mb_flag, mb_part_vsp_flag, sub_mb_vsp_flag, 및 skip_from_vsp_flag 신택스 엘리먼트들을 사용하여, 임의의 사이즈의 MB 파티션을 위한 플래그, 즉 VSPFlag를 도출할 수도 있다. VSPFlag는 본 개시물의 다른 곳에서 설명되는 바와 같이 MB 파티션이 VSP 화상으로부터 예측되는지의 여부를 나타낼 수도 있다. 예를 들어, MB 파티션에 대한 VSPFlag가 1과 동일한 경우, 그 MB 파티션은 VSP 화상으로부터 예측될 수도 있다.

[0156] 본 개시물의 기법들의 제 1 구현예에 따라, MB 파티션이 VSP 화상으로부터 예측된다는 것을 VSPFlag가 나타내는 경우 (예컨대, VSPFlag가 1과 동일한 경우), 모션 보상 유닛 (164) 은 MB 파티션의 예측 블록들의 샘플들을 VSP 화상의 샘플들에 설정할 수도 있다. 예를 들면, 현재 MB 파티션이 루마 성분에서의 좌표들 (x, y) 및 크로마 성분들에서의 (x', y') 를 갖는 좌측 상단 화소를 가진다고 가정한다. 더욱이, 현재 MB 파티션의 사이즈가 NxM이며, 여기서 N, M은 8 또는 16과 동일하다고 가정한다. 이 경우, 모션 보상 유닛 (164) 은, NxM의 동일한 사이즈를 갖는 좌표들 (x, y) 로부터 시작하여, VSP 화상의 루마 성분의 화소들의 값들을 현재 MB 파티션의 화소들로 설정할 수도 있다. 예를 들어, 모션 보상 유닛 (164) 은 예측 루마 블록에서의 각각의 루마 샘플이 VSP 화상의 대응하는 로케이션에서의 루마 샘플에 일치하도록 현재 MB 파티션에 대한 예측 루마 블록을 생성할 수도 있다.

[0157] 더욱이, 4:2:0 비디오 포맷의 경우의 N/2xM/2, 4:4:4 비디오 포맷의 경우의 NxM, 또는 4:2:2 비디오 포맷의 경우의 NxM/2의 사이즈를 갖는 좌표들 (x', y') 로부터 시작하여, 모션 보상 유닛 (164) 은 VSP 화상의 크로마 성분의 각각의 블록에서의 화소들의 값들을, 크로마 성분들의 동일한 좌표들 (x', y') 에서 시작하는 현재 MB 파티션의 화소들에 설정할 수도 있다. 예를 들어, 모션 보상 유닛 (164) 은 각각의 Cb 또는 Cb 샘플이 VSP 화상의 대응하는 로케이션에서의 Cb 또는 Cr 샘플과 일치하도록 현재 MB 파티션에 대한 예측 Cb 또는 Cr 블록을 생성할 수도 있다. 이 예에서, 모션 보상 유닛 (164) 은 VSP 화상의 Cb 또는 Cr 블록의 좌표들 (x', y') 로부터 예측 Cb 또는 Cr 블록을 생성하는 것을 시작할 수도 있다. 더욱이, 이 예에서, 예측 Cb 또는 Cr 블록은,VSP 화상의 크로마 블록들이 4:2:0 비디오 포맷, 4:4:4 비디오 포맷, 또는 4:2:2 비디오 포맷 각각에 따라 다운샘플링되면, 사이즈 N/2xM/2, NxM, 또는 nxM/2로 될 수도 있다.

[0158] vsp_mb_flag 또는 skip_from_vsp_flag가 1과 동일한 경우, MB의 각각의 블록은 1과 동일한 sub_mb_vsp_flag 및 mb_part_vsp_flag을 가진다고 유추된다. 전체 MB가 VSP 화상으로부터 예측되지 않는 경우 (예컨대, vsp_mb_flag가 0과 동일한 경우), MB의 MB 파티션들은 mb_part_vsp_flag의 상이한 값들을 가질 수도 있다.

[0159] 하나의 예의 대안적 해결책에서, vsp_mb_flag가 0과 동일하면 모든 MB 파티션들이 1과 동일한 mb_part_vsp_flag를 가질 수 있는 것이 아니라는 제약이 있다. 다르게 말하면, 전체 MB가 VSP 화상으로부터 예측되지 않는다는 것을 비디오 인코더 (20) 가 시그널링하였다면, 비디오 인코더 (20) 는 MB의 각각의 MB 파티션이 VSP 화상으로부터 예측된다는 것을 시그널링하지 않는다. MB의 모든 MB 파티션들이 VSP 화상으로부터 예측된다는 것을 시그널링하는 것은, 전체 MB가 VSP 화상으로부터 예측된다는 것을 시그널링하는 것과 동일한 결과들을 산출할 것이다. 그런고로, MB의 모든 MB 파티션들이 VSP 화상으로부터 예측되었으면, MB 파티션들을 생성하는데 비트들의 낭비가 있을 수도 있다.

[0160] 위에서 간략히 설명된 바와 같이, 엔트로피 디코딩 유닛 (150) 은 엔트로피 디코딩 동작들을 수행하여 신택스 엘리먼트들을 엔트로피 디코딩할 수도 있다. 더구나, 비디오 인코더 (20) 의 엔트로피 인코딩 유닛 (118) 은 엔트로피 인코딩 동작들을 수행하여 신택스 엘리먼트들을 엔트로피 인코딩할 수도 있다. 엔트로피 코더 (이를테면 엔트로피 인코딩 유닛 (118) 및 엔트로피 디코딩 유닛 (150)) 가 신택스 엘리먼트에 대해 엔트로피 인코딩 또는 엔트로피 디코딩 (즉, 엔트로피 코딩) 을 수행하는 경우, 엔트로피 코더는 코딩 모델을 결정할 수도 있다. 엔트로피 코더는 콘텍스트 모델을 콘텍스트적 (contextual) 정보 (즉, 콘텍스트) 에 기초하여 결

정할 수도 있다. 콘텍스트 모델은 특정 값들을 갖는 빈들의 확률들을 나타낼 수도 있다.

[0161] 본 개시물의 기법들의 제 1 구현예에 따라, 엔트로피 코더는 VSPFlag에 대한 콘텍스트를 유지할 수도 있다. 위에서 나타낸 바와 같이, VSPFlag는 MB 파티션이 VSP 화상으로부터 예측되는지의 여부를 나타낼 수도 있다. 엔트로피 코더는 vsp_mb_flag, mb_part_vsp_flag, sub_mb_vsp_flag, 및 skip_from_vsp_flag의 엔트로피 코딩에 대한 콘텍스트를 사용할 수도 있다. 엔트로피 디코딩 유닛 (150) 은 각각의 vsp_mb_flag, mb_part_vsp_flag, sub_mb_vsp_flag, 및 skip_from_vsp_flag에 기초하여 VSPFlag에 대한 콘텍스트를 업데이트할 수도 있다.

[0162] 슬라이스의 제 1 신택스 엘리먼트를 엔트로피 코딩하기 전에, 엔트로피 코더는 슬라이스의 신택스 엘리먼트들을 엔트로피 코딩함에 있어서 사용되는 콘텍스트 변수들을 초기화하기 위해 콘텍스트 초기화 프로세스를 수행한다. 본 개시물의 기법들에 따라, P 슬라이스들의 VSPFlag의 콘텍스트에 대한 초기 확률은, 심지어 VSP 모드가 P 슬라이스들 및 B 슬라이스들 양쪽 모두에 적용되더라도, B 슬라이스들의 VSPFlag의 콘텍스트에 대한 초기 확률과는 상이할 수도 있다.

[0163] 더욱이, 엔트로피 디코더가 신택스 엘리먼트를 엔트로피 디코딩하는 것을 시작하는 경우, 엔트로피 디코더는 신택스 엘리먼트의 모든 가능한 이진화들의 세트를 취출할 수도 있고 빈 인덱스 (binIdx) 를 -1로 설정할 수도 있다. 엔트로피 디코더는 그 다음에 비트스트림으로부터의 비트들을 순차적으로 프로세싱할 수도 있다. 엔트로피 디코더가 비트를 프로세싱하는 경우, 엔트로피 디코더는 binIdx를 점증시킬 수도 있고 binIdx에 적어도 부분적으로 기초하여 콘텍스트 인덱스 (ctxIdx) 를 결정할 수도 있다. 엔트로피 디코더는 ctxIdx에 연관된 콘텍스트 모델을 사용하여 현재 빈을 엔트로피 디코딩할 수도 있다. 다음으로, 엔트로피 디코더는 신택스 엘리먼트를 위해 지금까지 디코딩되었던 빈들이 신택스 엘리먼트의 가능한 이진화들 중 임의의 것과 일치하는지의 여부를 결정할 수도 있다. 아니라면, 엔트로피 디코더는 다른 비트를 이 단락에 기재된 방식으로 프로세싱할 수도 있다. 디코딩된 빈들이 신택스 엘리먼트의 가능한 이진화와 일치하는 경우, 엔트로피 디코더는 신택스 엘리먼트의 이진화를 신택스 엘리먼트의 원래의 값으로 다시 컨버팅할 수도 있다.

[0164] 엔트로피 인코더는 유사한 프로세스를 수행할 수도 있다. 이 프로세스에서, 엔트로피 인코더는 신택스 엘리먼트를 이진화할 수도 있고 빈 인덱스 (binIdx) 를 -1로 설정할 수도 있다. 엔트로피 인코더는 그 다음에 이진화된 신택스 엘리먼트의 빈들을 순차적으로 프로세싱할 수도 있다. 엔트로피 인코더가 빈을 프로세싱하는 경우, 엔트로피 인코더는, 빈에 대한 binIdx에 적어도 부분적으로 기초하여, 빈에 대한 콘텍스트 인덱스 (ctxIdx) 를 결정할 수도 있다. 엔트로피 인코더는 그 다음에 빈에 대한 ctxIdx에 연관된 콘텍스트 모델을 사용하여 빈을 엔트로피 인코딩할 수도 있다. 엔트로피 인코더가 이진화된 신택스 엘리먼트의 각각의 빈을 처리한 경우, 엔트로피 인코더는 신택스 엘리먼트를 엔트로피 인코딩하는 것을 완료했다.

[0165] 위에서 언급했듯이, 엔트로피 코더 (예컨대, 엔트로피 디코더 또는 엔트로피 인코더) 가 신택스 엘리먼트의 빈을 프로세싱하는 경우, 엔트로피 코더는, 빈의 빈 인덱스 (binIdx) 에 적어도 부분적으로 기초하여, 빈에 대한 콘텍스트 인덱스 (ctxIdx) 를 결정한다. 빈에 대한 ctxIdx를 결정하기 위해, 엔트로피 코더는 빈 인덱스 및 콘텍스트 인덱스 오프셋 (ctxIdxOffset) 을 사용하여, 미리 결정된 표에서, 빈에 대한 콘텍스트 인덱스 증분 (ctxIdxInc) 을 조회할 수도 있다. 엔트로피 코더는 ctxIdxOffset을 신택스 엘리먼트를 이진화하는 부분으로서 결정할 수도 있다. 빈에 대한 ctxIdx는 빈에 대한 ctxIdxInc 더하기 ctxIdxOffset과 동일할 수도 있다.

[0166] 본 개시물의 기법들의 제 1 구현예에 따라, 엔트로피 코더가 VSPFlag (디코딩될 신택스 엘리먼트에 의존하여 skip_from_vsp_flag 신택스 엘리먼트, vsp_mb_flag_syntax 엘리먼트, 또는 sub_mb_vsp_flag 신택스 엘리먼트일 수도 있음) 를 엔트로피 인코딩하는 경우, 엔트로피 코더는 VSPFlag에 연관된 빈에 대한 ctxIdxInc를 결정할 수도 있다. 일부 예들에서, 빈에 대한 ctxIdxInc는 condTermFlagA 및 condTermFlagB의 합일 수도 있다. 다시 말하면, ctxIdxInc는 ctxIdxInc = condTermFlagA + condTermFlagB로서 도출될 수도 있다. 다른 예에서, 빈에 대한 ctxIdxInc는 condTermFlagA과 동일할 수도 있다. 다시 말하면, ctxIdxInc는 ctxIdxInc = condTermFlagA로서 도출될 수도 있다. 다른 예에서, 빈에 대한 ctxIdxInc는 condTermFlagB와 동일하다. 다시 말하면, ctxIdxInc는 ctxIdxInc = condTermFlagB로서 도출될 수도 있다.

[0167] 위의 예들에서, 엔트로피 코더는, 현재 MB 또는 MB 파티션에 이웃하는 특정 블록들에 연관된 데이터가 현재 MB 또는 MB 파티션에 연관된 신택스 엘리먼트들에 대한 엔트로피 코딩 프로세스에서 사용할 수 있는지의 여부에 적어도 부분적으로 기초하여 condTermFlagA 및 condTermFlagB를 결정할 수도 있다. 예를 들어, mbPAddrA는 현재 MB 또는 MB 파티션의 좌측의 이웃 8x8 블록을 나타낼 수도 있고 mbPAddrB는 현재 MB 또는 MB 파티션 상측

의 이웃 8x8 블록을 나타낼 수도 있다. mbPAddrA가 이용가능하지 않거나 또는 블록 mbPAddrA에 대한 VSPFlag가 0과 동일하면, condTermFlagA는 0과 동일하다. 그렇지 않고, mbPAddrA가 이용가능하고 블록 mbPAddrA에 대한 VSPFlag가 1과 동일하면, condTermFlagA는 1과 동일하다. 마찬가지로, mbPAddrB가 이용가능하지 않거나 또는 블록 mbPAddrB에 대한 VSPFlag가 0과 동일하면, condTermFlagB는 0과 동일하다. 그렇지 않고, mbPAddrB가 이용가능하고 블록 mbPAddrB에 대한 VSPFlag가 1과 동일하면, condTermFlagB는 1과 동일하다.

[0168] 다른 예에서, mbPAddrA가 이용가능하고 블록 mbPAddrA에 대한 VSPFlag가 0과 동일하면, condTermFlagA는 1과 동일하다. 그렇지 않고, mbPAddrA가 이용불가능하거나 또는 블록 mbPAddrA에 대한 VSPFlag가 1과 동일하면, condTermFlagA는 0과 동일하다. 마찬가지로, mbPAddrB가 이용가능하고 블록 mbPAddrB에 대한 VSPFlag가 0과 동일하면, condTermFlagB는 1과 동일하다. 그렇지 않고, mbPAddrB가 이용불가능하거나 또는 블록 mbPAddrB에 대한 VSPFlag가 1과 동일하면, condTermFlagB는 0과 동일하다.

[0169] 위에서 간략히 언급된 바와 같이, 필터 유닛 (160) 은 블록화해제 동작을 수행하여 현재 비디오 유닛의 샘플 블록들에 연관된 블록화 아티팩트들을 감소시킬 수도 있다. 본 개시물의 기법들의 제 1 구현예에 따라, 필터 유닛 (160) 은 VSP 모드에 관련된 특정 블록화해제 동작들을 수행할 수도 있다. 제 1 예의 블록화해제 필터 프로세스에서, 1과 동일한 VSPFlag를 갖는 MB 파티션은, 블록화해제 필터 프로세스 동안 0과 동일한 모션 벡터 및 참조 화상 리스트 0으로부터 단방향으로 인터 코딩된, -1과 동일한 참조 인덱스에 의한, 인터 코딩으로서 간주된다. 필터 유닛 (160) 은 MB 파티션 및 다른 블록 사이의 경계에 대한 경계 필터링 강도 값 (bS) 을 결정할 수도 있다. 필터 유닛 (160) 은 bS를 사용하여 필터를 경계에서의 입력 샘플들의 어레이에 적용할 수도 있다. 이 제 1 예의 블록화해제 필터 프로세스에서, 필터 유닛 (160) 은, 블록들의 예측 (예컨대, 인터 또는 인트라) 모드들, 블록들의 참조 인덱스들, 블록들이 단방향으로 또는 양방향으로 인터 예측되는지의 여부, 및 블록들의 모션 벡터들에 적어도 부분적으로 기초하여 bS를 결정할 수도 있다.

[0170] 제 2 예의 블록화해제 필터 프로세스에서, 필터 유닛 (160) 은 블록화해제 필터 프로세스의 부분으로서 루마 콘텐츠 의존 경계 필터링 강도에 대한 도출 프로세스를 수행할 수도 있다. 이 예에서, 도출 프로세스에 대한 입력들은 필터링될 에지를 가로지르는 샘플들의 단일 세트의 입력 샘플 값들 (p_0 및 q_0) 이다. 도출 프로세스에 대한 입력들은 또한 verticalEdgeFlag를 포함한다. verticalEdgeFlag는 에지가 수평인지 또는 수직인지를 나타낼 수도 있다. 도출 프로세스의 출력은 경계 강도 변수 (bS) 이다.

[0171] 제 2 예의 블록화해제 필터 프로세스에서, SPS에서의 mb_adaptive_frame_field_flag 신택스 엘리먼트가 1과 동일하고 슬라이스 헤더에서의 field_pic_flag가 1과 동일하지 않으면, 변수 MbaffFrameFlag는 1과 동일할 수도 있다. 다시 말하면, MbaffFrameFlag = mb_adaptive_frame_field_flag && !field_pic_flag 이다. mb_adaptive_frame_field_flag 신택스 엘리먼트는 화상 내의 필드 및 프레임 MB들 간을 스위칭하는 것이 가능한지의 여부를 특정한다. 필드는 프레임의 교번하는 행들의 어셈블리이다. 0과 동일한 Mb_adaptive_frame_field는 화상 내의 프레임 및 필드 MB들 간을 스위칭하지 않는 것을 특정한다. 1과 동일한 Mb_adaptive_frame_field는 프레임들 내의 프레임 및 필드 MB들 간의 스위칭의 가능한 사용을 특정한다. field_pic_flag 신택스 엘리먼트는 슬라이스가 코딩된 필드의 슬라이스인지 또는 코딩된 프레임의 슬라이스인지를 특정한다. 코딩된 필드가 필드의 코딩된 표현이다. 1과 동일한 Field_pic_flag는 슬라이스가 코딩된 필드의 슬라이스임을 특정할 수도 있다. 0과 동일한 Field_pic_flag는 슬라이스가 코딩된 프레임의 슬라이스임을 특정할 수도 있다. 따라서, 1과 동일한 MbaffFrameFlag는, 프레임들 내의 프레임 및 필드 MB들 간의 스위칭의 가능한 사용과 현재 슬라이스가 코딩된 프레임의 슬라이스임을 나타낸다.

[0172] 더욱이, 제 2 예의 블록화해제 필터 프로세스에서, 필터 유닛 (160) 은 변수 mixedModeEdgeFlag를 다음과 같이 도출할 수도 있다. 먼저, MbaffFrameFlag가 1과 동일하고 샘플들 (p_0 및 q_0) 이 상이한 MB 쌍들에 있고, 그것들 중 하나가 필드 MB 쌍이고 나머지 하나가 프레임 MB 쌍이면, 필터 유닛 (160) 은 mixedModeEdgeFlag를 1과 동일하게 설정한다. 그렇지 않고, MbaffFrameFlag가 1과 동일하지 않거나 또는 샘플들 (p_0 및 q_0) 이 상이한 MB 쌍들에 있지 않으면, 필터 유닛 (160) 은 mixedModeEdgeFlag를 0과 동일하게 설정한다.

[0173] 다음으로, 제 2 예의 블록화해제 필터 프로세스에서, 필터 유닛 (160) 은 필터링되고 있는 블록 에지에 대해 변수 bS를 도출한다. 블록 에지가 MB 에지이고 다음의 조건들 중 임의의 것이 참이면, 4와 동일한 bS의 값이 출력된다. 제 1 조건에 따라, 필터 유닛 (160) 은, 샘플들 (p_0 및 q_0) 양쪽 모두가 프레임 MB들 내에 있고 샘플들 (p_0 및 q_0) 중 어느 하나 또는 양쪽 모두가 인트라 MB 예측 모드를 사용하여 코딩된 MB 내에 있으면, bS

의 값을 4로 설정한다. 제 2 조건에 따라, 필터 유닛 (160) 은, 샘플들 (p_0 및 q_0) 양쪽 모두가 프레임 MB 들 내에 있고 샘플들 (p_0 또는 q_0) 중 어느 하나 또는 양쪽 모두가 SP 또는 SI와 동일한 slice_type을 갖는 슬라이스 내에 있는 MB 내에 있으면, bS의 값을 4로 설정한다. 제 3 조건에 따라, 필터 유닛 (160) 은, MbaffFrameFlag가 1과 동일하거나 또는 field_pic_flag가 1과 동일하며, verticalEdgeFlag가 1과 동일하고 샘플들 (p_0 또는 q_0) 중 어느 하나 또는 양쪽 모두가 인트란 MB 예측 모드를 사용하여 코딩된 MB 내에 있으면, bS의 값을 4로 설정한다. 제 4 조건에 따라, 필터 유닛 (160) 은, MbaffFrameFlag가 1과 동일하거나 또는 field_pic_flag가 1과 동일하며, verticalEdgeFlag가 1과 동일하고 샘플들 (p_0 또는 q_0) 중 어느 하나 또는 양쪽 모두가 SP 또는 SI와 동일한 slice_type을 갖는 슬라이스 내에 있는 MB 내에 있으면, bS의 값을 4로 설정할 수도 있다.

[0174] 그렇지 않고, 다음의 조건들 중 임의의 것이 참이면, 필터 유닛 (160) 은 bS의 값을 3으로 설정한다. 제 1 조건에 따르면, 필터 유닛 (160) 은, mixedModeEdgeFlag가 0과 동일하고 샘플들 (p_0 또는 q_0) 중 어느 하나 또는 양쪽 모두가 인트라 MB 예측 모드를 사용하여 코딩된 MB 내에 있으면, bS의 값을 3으로 설정한다. 제 2 조건에 따르면, 필터 유닛 (160) 은, mixedModeEdgeFlag가 0과 동일하고 샘플들 (p_0 또는 q_0) 중 어느 하나 또는 양쪽 모두가 SP 또는 SI와 동일한 slice_type을 갖는 슬라이스 내에 있는 MB 내에 있으면, bS의 값을 3으로 설정한다. 제 3 조건에 따르면, 필터 유닛 (160) 은, mixedModeEdgeFlag가 0과 동일하며, verticalEdgeFlag가 0과 동일하고 샘플들 (p_0 또는 q_0) 중 어느 하나 또는 양쪽 모두가 인트라 MB 예측 모드를 사용하여 코딩된 MB 내에 있으면, bS의 값을 3으로 설정한다. 제 4 조건에 따르면, 필터 유닛 (160) 은, mixedModeEdgeFlag가 0과 동일하며, verticalEdgeFlag가 0과 동일하고 샘플들 (p_0 또는 q_0) 중 어느 하나 또는 양쪽 모두가 SP 또는 SI와 동일한 slice_type을 갖는 슬라이스 내에 있는 MB 내에 있으면, bS의 값을 3으로 설정한다.

[0175] 그렇지 않고, 다음의 조건들 중 임의의 것이 참이면, 필터 유닛 (160) 은 bS의 값을 2로 설정한다. 제 1 조건에 따르면, 필터 유닛 (160) 은, transform_size_8x8_flag가 샘플 p_0를 포함하는 MB에 대해 1과 동일하고 샘플 p_0를 포함하는 8x8 루마 블록에 연관된 8x8 루마 변환 블록이 영이 아닌 변환 계수 레벨들을 포함하면, bS의 값을 2로 설정한다. 제 2 조건에 따르면, 필터 유닛 (160) 은, transform_size_8x8_flag가 샘플 p_0를 포함하는 MB에 대해 0과 동일하고 샘플 p_0를 포함하는 4x4 루마 블록에 연관된 4x4 루마 변환 블록이 영이 아닌 변환 계수 레벨들을 포함하면, bS의 값을 2로 설정한다. 제 3 조건에 따르면, 필터 유닛 (160) 은, transform_size_8x8_flag가 샘플 p_0를 포함하는 MB에 대해 1과 동일하고 샘플 q_0를 포함하는 8x8 루마 블록에 연관된 8x8 루마 변환 블록이 영이 아닌 변환 계수 레벨들을 포함하면, bS의 값을 2로 설정한다. 제 4 조건에 따르면, 필터 유닛 (160) 은, transform_size_8x8_flag가 샘플 q_0를 포함하는 MB에 대해 0과 동일하고 샘플 q_0를 포함하는 4x4 루마 블록에 연관된 4x4 루마 변환 블록이 영이 아닌 변환 계수 레벨들을 포함하면, bS의 값을 2로 설정한다.

[0176] 그렇지 않고, 다음의 조건들 중 임의의 것이 참이면, 필터 유닛 (160) 은 bS의 값을 1과 동일하게 설정한다. 제 1 조건에 따르면, 필터 유닛 (160) 은, mixedModeEdgeFlag가 1과 동일하면, bS의 값을 1로 설정한다. 제 2 조건에 따르면 그리고 본 개시물의 기법들에 따라, 필터 유닛 (160) 은, mixedModeEdgeFlag가 0과 동일하고 샘플 p_0을 포함하는 MB/서브 MB 파티션에 대한 VSPFlag의 값이 샘플 q_0를 포함하는 MB/서브 MB 파티션에 대한 VSPFlag과 상이하면, bS의 값을 1로 설정한다. 제 3 조건에 따르면, 필터 유닛 (160) 은, mixedModeEdgeFlag가 0과 동일하면 그리고 샘플 p_0를 포함하는 MB/서브 MB 파티션의 예측 상이한 참조 화상들 또는 상이한 수의 모션 벡터들이 샘플 q_0를 포함하는 MB/서브 MB 파티션의 예측을 위해 대신 사용되면 bS의 값을 1로 설정한다. 2 개의 MB/서브 MB 파티션들을 위해 사용되는 참조 화상들이 동일한지 또는 상이한지의 결정은, 예측참조 화상 리스트 0으로의 인덱스를 사용하여 형성되는지 또는 참조 화상 리스트 1로의 인덱스를 사용하여 형성되는지에 무관하게, 그리고 또한 참조 화상 리스트 내의 인덱스 포지션이 상이한지의 여부에 무관하게, 어떤 화상들이 참조되는지에만 기초한다. 제 4 조건에 따르면, 필터 유닛 (160) 은, mixedModeEdgeFlag가 0과 동일하고 하나의 모션 벡터가 샘플 p_0를 포함하는 MB/서브 MB 파티션을 예측하는데 사용되고 하나의 모션 벡터가 샘플 q_0를 포함하는 MB/서브 MB 파티션을 예측하는데 사용되고 사용되는 모션 벡터들

의 수평 또는 수직 성분 사이의 절대차가 1/4 루마 프레임 샘플들의 유닛의 4 이상이면, bS의 값을 1로 설정한다. 제 5 조건에 따르면, 필터 유닛 (160) 은, mixedModeEdgeFlag가 0과 동일하고 2 개의 모션 벡터들 및 2 개의 상이한 참조 화상들이 샘플 p_0를 포함하는 MB/서브 MB 파티션을 예측하는데 사용되고 동일한 2 개의 참조 화상들에 대한 2 개의 모션 벡터들이 샘플 q_0를 포함하는 MB/서브 MB 파티션을 예측하는데 사용되고 동일한 참조 화상에 대한 2 개의 MB/서브 MB 파티션들의 예측에 사용되는 2 개의 모션 벡터들의 수평 또는 수직 성분 사이의 절대차가 1/4 루마 프레임 샘플들의 유닛의 4 이상이면, bS의 값을 1로 설정한다.

[0177] 제 6 조건에 따르면, 필터 유닛 (160) 은, mixedModeEdgeFlag가 0과 동일하고 동일한 참조 화상에 대한 2 개의 모션 벡터들이 샘플 p_0를 포함하는 MB/서브 MB 파티션을 예측하는데 사용되고 동일한 참조 화상에 대한 2 개의 모션 벡터들이 샘플 q_0를 포함하는 MB/서브 MB 파티션을 예측하는데 사용되고 양쪽 모두 다음의 조건들이 참이면, bS의 값을 1로 설정한다. 첫째, 2 개의 MB/서브 MB 파티션들의 예측에 사용되는 RefPicList0 모션 벡터들의 수평 또는 수직 성분 사이의 절대차가 1/4 루마 프레임 샘플들의 4 이상이거나 또는 2 개의 MB/서브 MB 파티션들의 예측에 사용되는 RefPicList1 모션 벡터들의 수평 또는 수직 성분 사이의 절대차가 1/4 루마 프레임 샘플들의 유닛의 4 이상이다. 둘째, 샘플 p_0를 포함하는 MB/서브 MB 파티션의 예측에 사용되는 RefPicList0 모션 벡터 및 샘플 q_0를 포함하는 MB/서브 MB 파티션의 예측에 사용되는 RefPicList1 모션 벡터의 수평 또는 수직 성분 사이의 절대차가 1/4 루마 프레임 샘플들의 유닛의 4 이상이거나 또는 샘플 p_0를 포함하는 MB/서브 MB 파티션의 예측에 사용되는 RefPicList1 모션 벡터 및 샘플 q_0를 포함하는 MB/서브 MB 파티션의 예측에 사용되는 RefPicList0 모션 벡터의 수평 또는 수직 성분 사이의 절대차가 1/4 루마 프레임 샘플들의 유닛의 4 이상이다. 1/4 루마 프레임 샘플들의 유닛의 4의 수직 차이는 1/4 루마 필드 샘플들의 유닛의 2의 차이이다. 그렇지 않으면, 필터 유닛 (160) 은 bS의 값을 0과 동일하게 설정한다.

[0178] 제 3 예의 블록화해제 필터 프로세스에서, 필터 유닛 (160) 은 블록화해제 동작을 수행하는 부분으로서 루마 콘텐츠 의존 경계 필터링 강도에 대한 도출 프로세스를 수행할 수도 있다. 이 예에서, 도출 프로세스에 대한 입력들은 필터링될 에지를 가로지르는 샘플의 단일 세트의 입력 샘플 값들 (p_0 및 q_0) 이다. 도출 프로세스에 대한 입력들은 또한 verticalEdgeFlag를 포함한다. 도출 프로세스의 출력은 경계 강도 변수 (bS) 일 수도 있다. 이 예에서, 필터 유닛 (160) 은 변수 mixedModeEdgeFlag를 다음과 같이 도출할 수도 있다. 먼저, MbaffFrameFlag가 1과 동일하고 샘플들 (p_0 및 q_0) 이 상이한 쌍들에 있고, 그것들 중 하나가 필드 MB 쌍이고 나머지 하나가 프레임 MB 쌍이면, 필터 유닛 (160) 은 mixedModeEdgeFlag를 1과 동일하게 설정한다. 그렇지 않고, MbaffFrameFlag가 1과 동일하지 않거나 또는 샘플들 (p_0 및 q_0) 이 상이한 MB 쌍들에 있지 않으면, 필터 유닛 (160) 은 mixedModeEdgeFlag를 0과 동일하게 설정한다.

[0179] 더욱이, 제 3 예의 블록화해제 필터 프로세스에서, 필터 유닛 (160) 은 MB들 사이의 블록 에지에 대한 변수 bS를 도출할 수도 있다. 블록 에지가 MB 에지이고 다음의 조건들 중 임의의 것이 참이면, 4와 동일한 bS의 값이 출력된다. 제 1 조건에 따라, 필터 유닛 (160) 은, 샘플들 (p_0 및 q_0) 양쪽 모두가 프레임 MB들 내에 있고 샘플들 (p_0 및 q_0) 중 어느 하나 또는 양쪽 모두가 인트라 MB 예측 모드를 사용하여 코딩된 MB 내에 있으면, bS의 값을 4로 설정한다. 제 2 조건에 따라 그리고 본 개시물의 기법들에 따라, 필터 유닛 (160) 은, 샘플들 (p_0 및 q_0) 양쪽 모두가 프레임 MB들 내에 있고 샘플 (p_0 또는 q_0) 중 어느 하나 또는 양쪽 모두가 1과 동일한 VSPFlag로 코딩된 MB 내에 있으면, bS의 값을 4로 설정한다. 제 3 조건에 따라, 필터 유닛 (160) 은, 샘플들 (p_0 및 q_0) 양쪽 모두가 프레임 MB들 내에 있고 샘플들 (p_0 또는 q_0) 중 어느 하나 또는 양쪽 모두가 SP 또는 SI와 동일한 slice_type을 갖는 슬라이스 내에 있는 MB 내에 있으면, bS의 값을 4로 설정한다. 제 4 조건에 따라, 필터 유닛 (160) 은, MbaffFrameFlag가 1과 동일하거나 또는 field_pic_flag가 1과 동일하며, verticalEdgeFlag가 1과 동일하고 샘플들 (p_0 또는 q_0) 중 어느 하나 또는 양쪽 모두가 인트라 MB 예측 모드를 사용하여 코딩된 MB 내에 있으면, bS의 값을 4로 설정한다. 제 5 조건에 따라 그리고 본 개시물의 기법들에 따라, 필터 유닛 (160) 은, MbaffFrameFlag가 1과 동일하거나 또는 field_pic_flag가 1과 동일하고 verticalEdgeFlag가 1과 동일하고 샘플들 (p_0 또는 q_0) 중 어느 하나 또는 양쪽 모두가 1과 동일한 VSPFlag로 코딩된 MB 내에 있으면, bS의 값을 4로 설정한다. 제 6 조건에 따라, 필터 유닛 (160) 은, MbaffFrameFlag가 1과 동일하거나 또는 field_pic_flag가 1과 동일하며, verticalEdgeFlag가 1과 동일하고 샘플들 (p_0 또는

q_0) 중 어느 하나 또는 양쪽 모두가 SP 또는 SI와 동일한 slice_type을 갖는 슬라이스 내에 있는 MB 내에 있으면, bS의 값을 4로 설정할 수도 있다.

[0180] 그렇지 않으면, 제 3 예의 블록화해제 필터 프로세스에서, 다음의 조건들 중 임의의 것이 참이면, 필터 유닛 (160) 은 bS의 값을 3으로 설정한다. 제 1 조건에 따르면, 필터 유닛 (160) 은, mixedModeEdgeFlag가 0과 동일하고 샘플들 (p_0 또는 q_0) 중 어느 하나 또는 양쪽 모두가 인트라 MB 예측 모드를 사용하여 코딩된 MB 내에 있으면, bS의 값을 3으로 설정한다. 제 2 조건에 따르면 그리고 본 개시물의 기법들에 따라, 필터 유닛 (160) 은, mixedModeEdgeFlag가 0과 동일하고 샘플들 (p_0 또는 q_0) 중 어느 하나 또는 양쪽 모두가 1과 동일한 VSPFlag로 코딩된 MB 내에 있으면, bS의 값을 3으로 설정한다. 제 3 조건에 따르면, 필터 유닛 (160) 은, mixedModeEdgeFlag가 0과 동일하고 샘플들 (p_0 또는 q_0) 중 어느 하나 또는 양쪽 모두가 SP 또는 SI와 동일한 slice_type을 갖는 슬라이스 내에 있는 MB 내에 있으면, bS의 값을 3으로 설정한다. 제 4 조건에 따르면, 필터 유닛 (160) 은, mixedModeEdgeFlag가 1과 동일하며, verticalEdgeFlag가 0과 동일하고 샘플들 (p_0 또는 q_0) 중 어느 하나 또는 양쪽 모두가 인트라 MB 예측 모드를 사용하여 코딩된 MB 내에 있으면, bS의 값을 3으로 설정한다. 제 5 조건에 따르면 그리고 본 개시물의 기법들에 따라, 필터 유닛 (160) 은, mixedModeEdgeFlag가 0과 동일하며, verticalEdgeFlag가 0과 동일하고 샘플들 (p_0 또는 q_0) 중 어느 하나 또는 양쪽 모두가 1과 동일한 VSPFlag로 코딩된 MB 내에 있으면, bS의 값을 3으로 설정한다. 제 6 조건에 따르면, 필터 유닛 (160) 은, mixedModeEdgeFlag가 1과 동일하며, verticalEdgeFlag가 0과 동일하고 샘플들 (p_0 또는 q_0) 중 어느 하나 또는 양쪽 모두가 SP 또는 SI와 동일한 slice_type을 갖는 슬라이스 내에 있는 MB 내에 있으면, bS의 값을 3으로 설정한다.

[0181] 그렇지 않으면, 제 3 예의 블록화해제 필터 프로세스에서, 필터 유닛 (160) 은, 제 2 예의 블록화해제 필터 프로세스에서 위에서 설명된 조건들 중 임의의 것이 충족되면, bS의 값을 2로 설정할 수도 있다. 그렇지 않으면, 필터 유닛 (160) 은, 제 2 예의 블록화해제 필터 프로세스에서, 제 2 조건을 제외한, 위에서 설명된 조건들 중 임의의 것이 충족되면, bS의 값을 1로 설정할 수도 있다. 그렇지 않으면, 필터 유닛 (160) 은 bS의 값을 0으로 설정할 수도 있다.

[0182] 본 개시물의 기법들의 제 2 구현예는 위에서 설명된 제 1 예와는 상이한 신택스들 및 시맨틱스를 사용한다. 제 2 구현예에서, 신택스 및 시맨틱스는, mb_skip_flag 및 skip_from_vsp_flag가 조건부적으로 그리고 유연하게 시그널링되도록 변화된다. 이 제 2 구현예에서, MB 계층, MB 예측, 및 서브 MB 예측에 대한 신택스는 위에서 설명된 제 1 구현예와 동일할 수도 있다. 그러나, 제 2 구현예에서, 슬라이스 데이터 및 대응하는 의미구조의 신택스 및 시맨틱스는 제 1 구현예에서와 상이할 수도 있다.

[0183] 아래의 표 6은 제 2 구현예에 따른 일 슬라이스 데이터 신택스 구조에 대한 일 예의 신택스이다.

[0184] 표 6 - 슬라이스 데이터 신택스

slice_data( ) {	C	기술어
if(entropy_coding_mode_flag)		
while(!byte_aligned( ))		
cabac_alignment_one_bit	2	f(1)
CurrMbAddr = first_mb_in_slice * (1 + MbaffFrameFlag)		
moreDataFlag = 1		
prevMbSkipped = 0		
do {		
if(slice_type != I && slice_type != SI)		
if(!entropy_coding_mode_flag) {		
mb_skip_run	2	ue(v)
prevMbSkipped = (mb_skip_run > 0)		
for(i=0; i<mb_skip_run; i++)		
CurrMbAddr = NextMbAddress(CurrMbAddr)		
moreDataFlag = more_rbsp_data( )		
} else {		
if(slice_vsp_flag && MbSkipFromVSPFlagLeft && MbSkipFromVSPFlagUp) {		
if(VspRefExist){		
skip_from_vsp_flag	2	ae(v)
moreDataFlag = !skip_from_vsp_flag		
}		
if(!skip_from_vsp_flag) {		
mb_skip_flag	2	ae(v)
moreDataFlag = !mb_skip_flag		
}		
} else {		
mb_skip_flag	2	ae(v)
if (slice_vsp_flag && !mb_skip_flag && VspRefExist)		
skip_from_vsp_flag	2	ae(v)
moreDataFlag = !skip_from_vsp_flag && !mb_skip_flag		
}		
}		
if(moreDataFlag) {		
if(MbaffFrameFlag && (CurrMbAddr % 2 == 0 || (CurrMbAddr % 2 == 1 && prevMbSkipped)))		
mb_field_decoding_flag	2	u(1) | ae(v)
macroblock_layer( )	2 | 3 | 4	
}		
if(!entropy_coding_mode_flag)		
moreDataFlag = more_rbsp_data( )		
else {		
if(slice_type != I && slice_type != SI)		
prevMbSkipped = mb_skip_flag		

[0185]

if(MbaffFrameFlag && CurrMbAddr % 2 == 0)		
moreDataFlag = 1		
else {		
end_of_slice_flag	2	ae(v)
moreDataFlag = !end_of_slice_flag		
}		
}		
CurrMbAddr = NextMbAddress(CurrMbAddr)		
} while(moreDataFlag)		
}		

[0186]

[0187] 표 6의 예의 신택스에서, 그리고 본 개시물의 기법들에 따라, 슬라이스에 대한 슬라이스 데이터 신택스 구조는, VSP가 슬라이스에 대해 가능하게 되며, 변수 MbSkipFromVSPFlagLeft = 1이고 변수 MbSkipFromVSPFlagUp = 1이

고 RefPicList0 또는 RefPicList1가 VSP 화상을 포함하는 경우, 현재 MB에 대해 skip_from_vsp_flag 신택스 엘리먼트를 포함할 수도 있다. 다른 예들에서, 슬라이스에 대한 슬라이스 데이터 신택스 구조는, 변수 MbSkipFromVSPFlagLeft = 1이고 변수 MbSkipFromVSPFlagUp = 1이고 RefPicList0 또는 RefPicList1이 VSP 화상을 포함하는 경우, 현재 MB에 대해 신택스 엘리먼트 skip_from_vsp_flag를 포함할 수도 있다. 예를 들면, 슬라이스 데이터 신택스 구조는, slice_vsp_flag = 1, MbSkipFromVSPFlagLeft = 1, MbSkipFromVSPFlagUp = 1 및 VspRefExist = 1인 경우, skip_from_vsp_flag를 포함할 수도 있다. 일부 예들에서, MbSkipFromVSPFlagLeft 및 MbSkipFromVSPFlagUp은 각각 좌측 및 상측 이용가능 MB들의 skip_from_vsp_flag의 값들로 설정될 수도 있다. 대안적 예에서, 좌측 MB에서의 이용가능한 블록들 중 임의의 것이 1과 동일한 VSPFlag를 가지면, MbSkipFromVSPFlagLeft는 1로 설정된다. 이 예에서, 상측 MB에서의 이용가능한 블록들 중 임의의 것이 1과 동일한 VSPFlag를 가지면, MbSkipFromVSPFlagUp는 1로 설정된다. 다른 대안적 예에서, 양쪽 모두의 좌측 8x8 블록들이 이용가능하고 1과 동일한 VSPFlag를 가지면, MbSkipFromVSPFlagLeft은 1로 설정된다. 이 예에서, 양쪽 모두의 상측 8x8 블록들이 이용가능하고 1과 동일한 VSPFlag를 가지면, MbSkipFromVSPFlagUp은 1로 설정된다.

[0188] 현재 MB에 대한 skip_from_vsp_flag는, 전체 현재 MB가 VSP 화상으로부터 예측되는지, 현재 MB가 영이 아닌 변환 계수 레벨들을 포함하는 루마 및/또는 크로마 변환 계수 블록들과 연관되는지, 및 현재 MB에 대한 mb_type 신택스 엘리먼트가 존재하는지를 나타낸다. 예를 들어, 1과 동일한 skip_from_vsp_flag는, 전체 현재 MB가 VSP 화상으로부터 예측되고 CodedBlockPatternLuma 및 CodedBlockPatternChroma가 0으로 설정되고, mb_type는 존재하지 않는다는 것을 나타낸다. skip_from_vsp_flag가 0과 동일한 경우, 현재 MB는 VSP 화상으로부터 예측되지 않거나 또는 CodedBlockPatternLuma 및 CodedBlockPatternChroma 중 적어도 하나는 0과 동일하지 않다.

대안으로, skip_from_vsp_flag가 0과 동일한 경우, 현재 MB는 정상 스킵 모드를 사용하여 예측된다. skip_from_vsp_flag가 존재하지 않는 경우, skip_from_vsp_flag는 0과 동일한 것으로 유추될 수도 있다. 일부 대체 예들에서, skip_from_vsp_flag가 1과 동일한 경우, mb_skip_flag는, mb_skip_flag가 존재하는지의 여부에 상관없이, 1과 동일한 것으로 유추될 수도 있다.

[0189] 표 6의 예의 신택스에서, 1과 동일한 mb_skip_flag는, 현재 MB에 대해, P 또는 SP 슬라이스를 디코딩하는 경우, mb_type은 P_Skip이라고 유추될 것이고 MB 유형은 총칭하여 P MB 유형이라고 지칭될 것이거나, 또는 그것에 대해, B 슬라이스를 디코딩하는 경우, mb_type은 B_Skip이라고 유추될 것이고 MB 유형은 총칭하여 B MB 유형이라고 지칭될 것임을 특정한다. 0과 동일한 mb_skip_flag는 현재 MB가 스킵되지 않는다는 것을 특정한다.

[0190] 더욱이, 표 6의 예의 신택스에서, skip_from_vsp_flag가 0과 동일하고 mb_skip_flag가 0과 동일하면 (즉, moreDataFlag = !skip_from_vsp_flag && !mb_skip_flag이면), 변수 moreDataFlag는 1과 동일하다. moreDataFlag가 1과 동일하면, 슬라이스 데이터 신택스 구조는 현재 MB에 대한 매크로블록 계층 신택스 구조를 포함할 수도 있다. 다른 예들에서, mb_skip_flag가 0과 동일하면 (즉, moreDataFlag = !mb_skip_flag이면), moreDataFlag는 1과 동일할 수도 있다.

[0191] 제 2 구현예에서, 엔트로피 코더, 이를테면 엔트로피 인코딩 유닛 (118) 또는 엔트로피 디코딩 유닛 (150) 은, 제 1 구현예에 관해 설명된 엔트로피 코딩 프로세스와 유사한 엔트로피 코딩 프로세스를 수행할 수도 있다. 대안으로, 제 2 구현예에서, 엔트로피 코더는 제 1 구현예에 관해 위에서 설명된 것들과는 상이한 콘텍스트 초기화 및 상이한 콘텍스트 선택 동작들을 수행할 수도 있다.

[0192] 예를 들면, 제 2 구현예에서, 엔트로피 코더는, 하나는 skip_from_vsp_flag를 위한 것이고 나머지 하나는 vsp_mb_flag, mb_part_vsp_flag 및 sub_mb_vsp_flag를 위한 것인 콘텍스트들의 2 개의 세트를 사용할 수도 있다. 콘텍스트 개시 동작 동안, 엔트로피 코더는 P 슬라이스들 및 B 슬라이스들에 대한 상이한 값들로 vsp_mb_flag, mb_part_vsp_flag, 또는 sub_mb_vsp_flag의 콘텍스트들의 초기 확률들을 초기화할 수도 있다. VSP 모드는 P 슬라이스들 및 B 슬라이스들 모두에 적용할 수도 있다. 더욱이, 엔트로피 인코더는 P 슬라이스들의 skip_from_vsp_flag의 콘텍스트에 대한 초기 확률들을 B 슬라이스들의 그것과는 상이하게 초기화할 수도 있다.

[0193] 제 2 구현예에서, 엔트로피 코더가 mb_part_vsp_flag, sub_mb_vsp_flag, vsp_mb_flag, 또는 skip_from_vsp_flag를 엔트로피 코딩하고 있는 경우, 엔트로피 코더는 이들 신택스 엘리먼트들 중 하나에 연관된 빈에 대한 ctxIdxInc를 결정할 수도 있다. 엔트로피 코더는 condTermFlagA 및 condTermFlagB에 적어도 부분적으로 기초하여 빈에 대한 ctxIdxInc를 결정할 수도 있다. 다양한 예들에서, 엔트로피 코더는, ctxIdxInc = condTermFlagA + condTermFlagB가 되도록, ctxIdxInc = condTermFlagA가 되도록, 또는 ctxIdxInc

= condTermFlagB가 되도록 빈에 대한 ctxIdxInc를 결정할 수도 있다. 제 2 구현예에서, 엔트로피 코더는, 현재 MB 또는 MB 파티션에 이웃하는 특정 블록들에 연관된 데이터가 현재 MB 또는 MB 파티션에 연관된 신택스 엘리먼트들에 대한 엔트로피 코딩 프로세스에서 사용할 수 있는지의 여부에 적어도 부분적으로 기초하여 condTermFlagA 및 condTermFlagB를 결정할 수도 있다.

[0194] 예를 들어, 제 2 구현예에서, 엔트로피 코더가 mb_part_vsp_flag, sub_mb_vsp_flag, vsp_mb_flag 신택스 엘리먼트를 코딩하고 있으면, mbPAddrA는 현재 MB 또는 MB 파티션 좌측의 이웃 8x8 블록을 나타낼 수도 있고 mbPAddrB는 현재 MB 또는 MB 파티션 상측의 이웃 8x8 블록을 나타낼 수도 있다. 이 예에서, mbPAddrA가 이용가능하지 않거나 또는 mbPAddrA에 대한 NonSkipVSPFlag가 0과 동일하면, condTermFlagA는 0과 동일하다. 이 예에서, NonSkipVSPFlag는 코딩될 신택스 엘리먼트에 의존하여 vsp_mb_flag, mb_part_vsp_flag, 또는 sub_mb_vsp_flag 중 어느 하나일 수도 있다. 그렇지 않고, mbPAddrA가 이용가능하고 mbPAddrA에 대한 NonSkipVSPFlag가 1과 동일하면, condTermFlagA는 1과 동일하다. 마찬가지로, mbPAddrB가 이용가능하지 않거나 또는 mbPAddrB에 대한 NonSkipVSPFlag가 0과 동일하면, condTermFlagB는 0과 동일하다. mbPAddrB가 이용가능하고 mbPAddrB에 대한 NonSkipVSPFlag가 1과 동일하면, condTermFlagB는 1과 동일하다.

[0195] 더욱이, 제 2 구현예에서, 엔트로피 코더가 skip_from_vsp_flag 신택스 엘리먼트를 코딩하고 있으면, mbPAddrA는 현재 MB 또는 MB 파티션 좌측의 이웃 8x8 블록을 나타낼 수도 있고 mbPAddrB는 현재 MB 또는 MB 파티션 상측의 이웃 8x8 블록을 나타낼 수도 있다. 이 예에서, mbPAddrA가 이용가능하고 mbPAddrA에 대한 skip_from_vsp_flag가 0과 동일하면, condTermFlagA는 1과 동일할 수도 있다. 그렇지 않고, mbPAddrA가 이용가능하지 않고 mbPAddrA에 대한 skip_from_vsp_flag가 1과 동일하면, condTermFlagA는 0과 동일할 수도 있다. 마찬가지로, mbPAddrB가 이용가능하고 mbPAddrB에 대한 skip_from_vsp_flag가 0과 동일하면, condTermFlagB는 1과 동일할 수도 있다. 그렇지 않고, mbPAddrB가 이용가능하지 않거나 또는 mbPAddrB에 대한 skip_from_vsp_flag가 1과 동일하면, condTermFlagB는 0과 동일할 수도 있다.

[0196] 엔트로피 코더가 skip_from_vsp_flag를 코딩하고 있는 다른 예에서, mbPAddrA가 이용가능하고 mbPAddrA에 대한 skip_from_vsp_flag가 1과 동일하면, condTermFlagA는 1과 동일할 수도 있다. 그렇지 않고, mbPAddrA가 이용가능하지 않고 mbPAddrA에 대한 skip_from_vsp_flag가 0과 동일하면, condTermFlagA는 0과 동일할 수도 있다. 마찬가지로, mbPAddrB가 이용가능하고 mbPAddrB에 대한 skip_from_vsp_flag가 1과 동일하면, condTermFlagB는 1과 동일할 수도 있다. 그렇지 않고, mbPAddrB가 이용가능하지 않거나 또는 mbPAddrB에 대한 skip_from_vsp_flag가 0과 동일하면, condTermFlagB는 0과 동일할 수도 있다.

[0197] 엔트로피 코더가 skip_from_vsp_flag를 코딩하고 있는 다른 예에서, mbPAddrA가 이용가능하고 mbPAddrA에 대한 vsp_mb_flag가 0과 동일하면, condTermFlagA는 1과 동일할 수도 있다. 그렇지 않고, mbPAddrA가 이용가능하지 않고 mbPAddrA에 대한 vsp_mb_flag가 1과 동일하면, condTermFlagA는 0과 동일할 수도 있다. 마찬가지로, mbPAddrB가 이용가능하고 mbPAddrB에 대한 vsp_mb_flag가 0과 동일하면, condTermFlagB는 1과 동일할 수도 있다. 그렇지 않고, mbPAddrB가 이용가능하지 않거나 또는 mbPAddrB에 대한 vsp_mb_flag가 1과 동일하면, condTermFlagB는 0과 동일할 수도 있다.

[0198] 다른 예들에서, 엔트로피 코더는 skip_from_vsp_flag 및 vsp_mb_flag 양쪽 모두로부터 condTermFlagA 및 condTermFlagB를 결정할 수도 있다. 다른 예에서, 엔트로피 코더는 vsp_mb_flag로부터 condTermFlagA 및 condTermFlagB를 결정할 수도 있다.

[0199] 본 개시물의 기법들의 제 3 구현예에서, MB 계층, 서브 MB 예측, 및 슬라이스 데이터에 대한 신택스는 위에서 설명된 제 1 구현예에서와 동일할 수도 있다. 그러나, 제 3 구현예에서, 신택스는 다양한 상황들에 기초하여 더 최적화될 수도 있다. 예를 들어, MB 예측 신택스 구조 및 대응하는 의미구조의 신택스 및 시맨틱스는 제 1 구현예에서와 상이할 수도 있다. 아래의 표 7은 제 3 구현예에 따른 슬라이스 데이터에 대한 일 예의 신택스이다.

[0200] **표 7 - 매크로블록 예측 신택스**

mb_pred(mb_type) {	C	기술어
if(MbPartPredMode(mb_type, 0) == Intra_4x4 || MbPartPredMode(mb_type, 0) == Intra_16x16) {		
if(MbPartPredMode(mb_type, 0) == Intra_4x4)		
for(luma4x4BlkIdx=0; luma4x4BlkIdx<16; luma4x4BlkIdx++) {		
prev_intra4x4_pred_mode_flag[luma4x4BlkIdx **]**	2	u(1) | ae(v)
if(!prev_intra4x4_pred_mode_flag**[** luma4x4BlkIdx **]**)		
rem_intra4x4_pred_mode[luma4x4BlkIdx **]**	2	u(3) | ae(v)
}		
intra_chroma_pred_mode	2	ue(v) | ae(v)
} else if(MbPartPredMode(mb_type, 0) != Direct) {		
for(mbPartIdx = 0; mbPartIdx < NumMbPart(mb_type); mbPartIdx++)		
if (slice_vsp_flag && NumMbPart(mb_type) != 1 && VspRefExist) //vsp_mb_flag is not 1		
mb_part_vsp_flag[mbPartIdx **]**	2	ae(v)
if(!mb_part_vsp_flag[mbPartIdx] &&(num_ref_idx_l0_active_minus1 > 0 || mb_field_decoding_flag) && MbPartPredMode(mb_type, mbPartIdx) != Pred_L1)		
ref_idx_l0[mbPartIdx **]**	2	te(v) | ae(v)
for(!mb_part_vsp_flag[mbPartIdx] &&(mbPartIdx = 0; mbPartIdx < NumMbPart(mb_type); mbPartIdx++)		
if((num_ref_idx_l1_active_minus1 > 0 || mb_field_decoding_flag) && MbPartPredMode(mb_type, mbPartIdx) != Pred_L0)		
ref_idx_l1[mbPartIdx **]**	2	te(v) | ae(v)
for(mbPartIdx = 0; mbPartIdx < NumMbPart(mb_type); mbPartIdx++)		
if(!mb_part_vsp_flag[mbPartIdx] &&(MbPartPredMode (mb_type, mbPartIdx) != Pred_L1)		
for(compIdx = 0; compIdx < 2; compIdx++)		
mvd_l0[mbPartIdx **][** 0 **][** compIdx **]**	2	se(v) | ae(v)
for(mbPartIdx = 0; mbPartIdx < NumMbPart(mb_type); mbPartIdx++)		
if(!mb_part_vsp_flag[mbPartIdx] &&(MbPartPredMode(mb_type, mbPartIdx) != Pred_L0)		
for(compIdx = 0; compIdx < 2; compIdx++)		
mvd_l1[mbPartIdx **][** 0 **][** compIdx **]**	2	se(v) | ae(v)
}		
}		

[0201]

[0202] 표 7은 위의 표 4와 유사하다. 그러나, 표 7의 예 신택스는 ref_idx_l1 신택스 엘리먼트들을 생성/디코딩하기 위한 루프에서 "for (mbPartIdx = 0; mbPartIdx < NumMbPart(mb_type) && !mb_part_vsp_flag[mbPartIdx]; mbPartIdx++)" 대신에 "for (!mb_part_vsp_flag[mbPartIdx] && (mbPartIdx = 0; mbPartIdx < NumMbPart(mb_type); mbPartIdx++)" 를 나타낸다. 다른 예들에서, MB 예측 신택스 구조는, slice_vsp_flag && NumMbPart(mb_type) != 1 && MbPartPredMode(mb_type, mbPartIdx) == Pred_L0이면, mb_part_vsp_flag[mbPartIdx]를 포함할 수도 있다.

[0203] 본 개시물의 기법들의 제 4 구현예는 위에서 설명된 제 2 구현예와 유사할 수도 있다. 그러나, 제 4 구현예에서, MB 계층, MB 예측, 및 서브 MB 예측의 신택스 설계는 제 2 구현예의 그것에 상대적으로 변화될 수도 있다. 아래의 표 8은 제 4 구현예에 따른 MB 계층 신택스 구조의 일 예의 신택스이다.

[0204] **표 8 - 매크로블록 계층 신택스**

macroblock_layer( ) {	C	기술어
if(slice_vsp_flag && MbVSPFlagLeft && bMbVSPFlagUp){		
if(VspRefExist)		
vsp_mb_flag	2	ae(v)
if (! vsp_mb_flag)		
mb_type	2	ue(v) | ae(v)
}		
else {		
mb_type	2	ue(v) | ae(v)
if (slice_vsp_flag && VspRefExist && (mb_type = = B_L0_16x16 || mb_type == P_L0_16x16))		
vsp_mb_flag	2	ae(v)
}		
if(mb_type = = I_PCM) {		
while(!byte_aligned( ))		
pcm_alignment_zero_bit	3	f(1)
for(i = 0; i < 256; i++)		
pcm_sample_luma[i **]**	3	u(v)
for(i = 0; i < 2 * MbWidthC * MbHeightC; i++)		
pcm_sample_chroma[i **]**	3	u(v)
} else {		
noSubMbPartSizeLessThan8x8Flag = 1		
if(mb_type != I_NxN && MbPartPredMode(mb_type, 0) != Intra_16x16 && NumMbPart(mb_type) = = 4) {		
if (!vsp_mb_flag)		
sub_mb_pred(mb_type)	2	
for(mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++)		
if(sub_mb_type[mbPartIdx] != B_Direct_8x8) {		
if(NumSubMbPart(sub_mb_type[mbPartIdx]) > 1)		
noSubMbPartSizeLessThan8x8Flag = 0		
} else if(!direct_8x8_inference_flag)		
noSubMbPartSizeLessThan8x8Flag = 0		
} else {		
if(transform_8x8_mode_flag && mb_type = = I_NxN)		
transform_size_8x8_flag	2	u(1) | ae(v)
if (!vsp_mb_flag)		
mb_pred(mb_type)	2	
}		
if(MbPartPredMode(mb_type, 0) != Intra_16x16) {		
coded_block_pattern	2	me(v) | ae(v)
if(CodedBlockPatternLuma > 0 && transform_8x8_mode_flag && mb_type != I_NxN && noSubMbPartSizeLessThan8x8Flag && (vsp_mb_flag || (!vsp_mb_flag && (mb_type != B_Direct_16x16 || direct_8x8_inference_flag))))		

[0205]

transform_size_8x8_flag	2	u(1) | ae(v)
}		
if(CodedBlockPatternLuma > 0 || CodedBlockPatternChroma > 0 || MbPartPredMode(mb_type, 0) = = Intra_16x16) {		
mb_qp_delta	2	se(v) | ae(v)
residual(0, 15)	3 | 4	
}		
}		
}		

[0206]

[0207] 표 8의 예에서, MB 계층 신택스 구조의 콘텐츠는 변수들인 MbVSPFlagLeft 및 MbVSPFlagUp에 의존한다. MbVSPFlagLeft 및 MbVSPFlagUp은 각각 현재 MB의 좌측 및 상측인 MB들의 vsp_mb_flag의 값으로 설정될 수도 있다. 대안으로, MbVSPFlagLeft 및 MbVSPFlagUp은 각각 현재 MB의 좌측 및 상측인 MB들에 대한 skip_from_vsp_flag 및 vsp_mb_flag 중 더 큰 것으로 설정될 수도 있다. 다시 말하면, MbVSPFlagLeft는 좌

측 MB의 max(skip_from_vsp_flag, vsp_mb_flag) 로 설정될 수도 있고 MbVSPFlagUp는 상측 MB의 max(skip_from_vsp_flag, vsp_mb_flag) 로 설정될 수도 있다. 다른 대안적 예에서, 현재 MB 좌측의 임의의 이용가능한 블록들이 1과 동일한 VSPFlag를 가지면, MbVSPFlagLeft는 1로 설정된다. 이 예에서, 현재 MB 상측의 임의의 이용가능한 블록들이 0과 동일한 VSPFlag를 가지면 MbVSPFlagUp는 1로 설정될 수도 있다. 또 다른 예에서, 현재 MB 좌측의 8x8 블록들 양쪽 모두가 현재 MB의 코딩에서 사용할 수 있고 1과 동일한 VSPFlags를 가지면, MbVSPFlagLeft는 1로 설정된다. 이 예에서, 현재 MB 상측의 양쪽 모두의 8x8 블록이 현재 MB의 코딩에서 사용할 수 있고 1과 동일한 VSPFlags를 가지면, MbVSPFlagUp은 1로 설정된다.

[0208] 표 8의 예에서, MB 계층 신택스 구조는, slice_vsp_flag가 1과 동일하며, MbVSPFlagLeft가 1과 동일하며, bMbVSPFlagUp이 1과 동일하고 VspRefExist가 1과 동일하면, vsp_mb_flag를 포함할 수도 있다. slice_vsp_flag가 1과 동일하며, MbVSPFlagLeft가 1과 동일하며, bMbVSPFlagUp이 1과 동일하고 VspRefExist가 1과 동일하지 않으면, MB 계층 신택스 구조는 mb_type을 포함할 수도 있다. 다른 예들에서, MbVSPFlagLeft가 1과 동일하고, bMbVSPFlagUp이 1과 동일하고 VspRefExist가 1과 동일하면, MB 계층 신택스 구조는 vsp_mb_flag를 포함할 수도 있다. MbVSPFlagLeft가 1과 동일하며, bMbVSPFlagUp이 1과 동일하고 VspRefExist가 1과 동일하지 않으면, MB 계층 신택스 구조는 mb_type을 포함할 수도 있다.

[0209] 더구나, 표 8의 예에서, slice_vsp_flag && VspRefExist && (mb_type == B_L0_16x16 || mb_type == P_L0_16x16) 이면, MB 계층 신택스 구조는 vsp_mb_flag를 포함할 수도 있다. 다른 예들에서, mb_type == (B_L0_16x16 || mb_type == P_L0_16x16) 이면, MB 계층 신택스 구조는 vsp_mb_flag를 포함할 수도 있다.

[0210] 더욱이, 제 4 구현예에서, MB 예측 신택스 구조의 신택스는 위의 표 7에 도시된 바와 동일할 수도 있다. 그러나, 일부 예들에서, MB 예측 신택스 구조는, slice_vsp_flag && NumMbPart(mb_type) != 1이고 MbPartPredMode(mb_type, mbPartIdx) == Pred_L0이면, mb_part_vsp_flag[mbPartIdx]를 포함할 수도 있다. 아래의 표 9는 제 4 구현예에서 사용된 일 예의 서브 MB 예측 신택스를 도시한다.

[0211] **표 9 - 서브 매크로블록 예측 신택스**

sub_mb_pred(mb_type) {	C	기술어
for(mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++) {		
if (slice_vsp_flag){ // vsp_mb_flag is not 1		
if(SubMbVSPFlagLeft && SubMbVSPFlagUp){		
if(VspRefExist)		
sub_mb_vsp_flag[mbPartIdx **]**	2	ae(v)
if (!sub_mb_vsp_flag[mbPartIdx])		
sub_mb_type[mbPartIdx **]**	2	ue(v) | ae(v)
} else {		
sub_mb_type[mbPartIdx **]**	2	ae(v)
if (slice_vsp_flag && VspRefExist && (NumSubMbPart (sub_mb_type[mbPartIdx]) = = 1 || sub_mb_type[mbPartIdx] = = B_Direct_8x8))		
sub_mb_vsp_flag[mbPartIdx **]**	2	ae(v)
}		
}		
for(mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++)		
if(!sub_mb_vsp_flag[mbPartIdx]) &&(num_ref_idx_l0_active_minus1 > 0 || mb_field_decoding_flag) && mb_type != P_8x8ref0 && sub_mb_type[mbPartIdx] != B_Direct_8x8 && SubMbPredMode(sub_mb_type[mbPartIdx]) != Pred_L1)		
ref_idx_l0[mbPartIdx **]**	2	te(v) | ae(v)
for(mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++)		
if(!sub_mb_vsp_flag[mbPartIdx]) && (num_ref_idx_l1_active_minus1 > 0 || mb_field_decoding_flag) && sub_mb_type[mbPartIdx] != B_Direct_8x8 && SubMbPredMode(sub_mb_type[mbPartIdx]) != Pred_L0)		
ref_idx_l1[mbPartIdx **]**	2	te(v) | ae(v)
for(mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++)		
if(!sub_mb_vsp_flag[mbPartIdx]) &&(sub_mb_type[mbPartIdx] != B_Direct_8x8 && SubMbPredMode(sub_mb_type[mbPartIdx]) != Pred_L1)		
for(subMbPartIdx = 0; subMbPartIdx < NumSubMbPart(sub_mb_type[mbPartIdx]); subMbPartIdx++)		
for(compIdx = 0; compIdx < 2; compIdx++)		
mvd_l0[mbPartIdx **][** subMbPartIdx **][** compIdx **]**	2	se(v) | ae(v)
for(mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++)		
if(!sub_mb_vsp_flag[mbPartIdx]) &&sub_mb_type[mbPartIdx] != B_Direct_8x8 && SubMbPredMode(sub_mb_type[mbPartIdx]) != Pred_L0)		
for(subMbPartIdx = 0; subMbPartIdx < NumSubMbPart(sub_mb_type[mbPartIdx]); subMbPartIdx++)		
for(compIdx = 0; compIdx < 2; compIdx++)		
mvd_l1[mbPartIdx **][** subMbPartIdx **][** compIdx **]**	2	se(v) | ae(v)
}		

[0212]

[0213] 표 9의 예에서, 변수들인 SubMbVSPFlagLeft 및 SubMbVSPFlagUp은 서브 MB 예측의 콘텐츠를 제어한다. SubMbVSPFlagLeft 및 SubMbVSPFlagUp은 각각 좌측 8x8 및 상측 8x8 블록들의 sub_mb_vsp_flag의 값으로 설정된다. 좌측 8x8 블록은 현재 서브 MB 파티션의 좌측이다. 상측 8x8 블록은 현재 서브 MB 파티션의 상측이다. 특히, 표 9의 예의 신택스에서, 서브 MB 예측 신택스 구조는, SubMBVspFlagLeft 및 SubMBVspFlagUp이 1과 동일하고 RefPicList0 또는 RefPicList1이 VSP 화상을 포함하는 (예컨대, VspRefExist = 1인) 경우, 서브 MB 파티션에 대한 sub_mb_vsp_flag를 포함할 수도 있다. 서브 MB 파티션에 대한 sub_mb_vsp_flag는 서브 MB 파티션들이 VSP로부터 예측되는지의 여부를 나타낸다. 더욱이, 표 9의 예의 신택스에서, 서브 MB 예측 신택스 구조는, 서브 MB 파티션이 VSP로부터 예측되지 않음을 서브 MB 파티션에 대한 sub_mb_vsp_flag가 나타내는 경우, 서브 MB 파티션에 대한 sub_mb_type을 포함할 수도 있다.

[0214] 더욱이, 표 9의 예의 신택스에서, SubMbVSPFlagLeft 또는 SubMbVSPFlagUp 중 어느 하나가 1과 동일하지 않으면, 서브 MB 예측 신택스 구조는, VSP가 슬라이스에 대해 가능하게 되며, RefPicList0 또는 RefPicList1이 VSP 화상을 포함하고, 서브 MB 파티션의 서브 MB 파티션들의 수가 1이거나 또는 서브 MB 파티션의 서브 MB 유형이 B_direct_8x8인 경우, 서브 MB 파티션에 대한 sub_mb_vsp_flag를 포함할 수도 있다. 서브 MB 파티션의 서브 MB 유형이 B_direct_8x8이면, 서브 MB 파티션은 양방향으로 인터 예측되고 8x8 블록들로 파티셔닝된다.

[0215] 다른 예에서, subMbVSPFlagLeft 및 SubMbVSPFlagUp은, 각각 현재 서브 MB 파티션을 코딩함에 있어서 사용할 수 있으면, 좌측 8x8 및 상측 8x8 블록들의 VSPFlag의 값으로 설정될 수도 있다.

[0216] 더욱이, 표 9의 대체 버전에서, 서브 MB 예측 신택스 구조는, sub_mb_type[mbPartIdx] == P_L0_8x8 또는 sub_mb_type[mbPartIdx] == B_L0_8x8이면, sub_mb_vsp_flag[mbPartIdx]를 포함할 수도 있다.

[0217] 본 개시물의 기법들의 제 5 구현예는 위에서 설명된 제 1 구현예와 유사할 수도 있다. 이 제 5 구현예에서, mb_skip_flag 및 skip_from_vsp_flag는 하나의 신택스 엘리먼트로 결합될 수도 있다. 위에서 나타낸 바와 같이, mb_skip_flag는 스킵 모드가 사용되는지의 여부를 나타낸다. skip_from_vsp_flag는, 스킵 모드가 사용되는 경우, 전체 MB가 VSP 화상으로부터 예측된다는 것을 나타낸다. 따라서, 제 5 구현예에서, 결합된 플래그는, 스킵 모드가 사용되는지의 여부와, 스킵 모드가 사용되면, 전체 MB가 VSP 화상으로부터 예측된다는 것을 나타낸다.

[0218] 본 개시물의 기법들의 제 6 구현예는 위에서 설명된 제 2 구현예와 유사할 수도 있다. 그러나, 제 6 구현예에서, MB 계층 신택스 구조들 및 MB 예측 신택스 구조들은 아래의 표 10 및 표 11의 신택스들에 따라 정의될 수도 있다.

[0219] 표 10 - 매크로블록 계층 신택스 구조

macroblock_layer( ) {	C	기술어
mb_type	2	ue(v) | ae(v)
if(mb_type = = B_Direct_16x16 && slice_vsp_flag && VspRefExist)		
vsp_direct_flag		
if(mb_type = = I_PCM) {		
while(!byte_aligned( ))		
pcm_alignment_zero_bit	3	f(1)
for(i = 0; i < 256; i++)		
pcm_sample_luma[i **]**	3	u(v)
for(i = 0; i < 2 * MbWidthC * MbHeightC; i++)		
pcm_sample_chroma[i **]**	3	u(v)
} else {		
noSubMbPartSizeLessThan8x8Flag = 1		
if(mb_type != I_NxN && MbPartPredMode(mb_type, 0) != Intra_16x16 && NumMbPart(mb_type) = = 4) {		
sub_mb_pred(mb_type)	2	
for(mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++)		
if(sub_mb_type[mbPartIdx] != B_Direct_8x8) {		
if(NumSubMbPart(sub_mb_type[mbPartIdx]) > 1)		
noSubMbPartSizeLessThan8x8Flag = 0		
} else if(!direct_8x8_inference_flag)		
noSubMbPartSizeLessThan8x8Flag = 0		
} else {		
if(transform_8x8_mode_flag && mb_type = = I_NxN)		
transform_size_8x8_flag	2	u(1) | ae(v)
mb_pred(mb_type)	2	
}		
if(MbPartPredMode(mb_type, 0) != Intra_16x16) {		
coded_block_pattern	2	me(v) | ae(v)
if(CodedBlockPatternLuma > 0 && transform_8x8_mode_flag && mb_type != I_NxN && noSubMbPartSizeLessThan8x8Flag && (**vsp_direct_flag**||(!**vsp_direct_flag** && (mb_type != B_Direct_16x16 || direct_8x8_inference_flag))))		
transform_size_8x8_flag	2	u(1) | ae(v)
}		
if(CodedBlockPatternLuma > 0 || CodedBlockPatternChroma > 0 || MbPartPredMode(mb_type, 0) = = Intra_16x16) {		
mb_qp_delta	2	se(v) | ae(v)
residual(0, 15)	3 | 4	
}		
}		
}		

[0220]

[0221] **표 11 - 매크로블록 예측 신택스**

mb_pred(mb_type) {	C	기술어
if(MbPartPredMode(mb_type, 0) == Intra_4x4 || MbPartPredMode(mb_type, 0) == Intra_16x16) {		
if(MbPartPredMode(mb_type, 0) == Intra_4x4)		
for(luma4x4BlkIdx=0; luma4x4BlkIdx<16; luma4x4BlkIdx++) {		
prev_intra4x4_pred_mode_flag[luma4x4BlkIdx **]**	2	u(1) | ae(v)
if(!prev_intra4x4_pred_mode_flag**[** luma4x4BlkIdx **]**)		
rem_intra4x4_pred_mode[luma4x4BlkIdx **]**	2	u(3) | ae(v)
}		
intra_chroma_pred_mode	2	ue(v) | ae(v)
} else if(MbPartPredMode(mb_type, 0) != Direct) {		
for(mbPartIdx = 0; mbPartIdx < NumMbPart(mb_type); mbPartIdx++)		
if (slice_vsp_flag && VspRefExist)		
mb_part_vsp_flag[mbPartIdx **]**	2	ae(v)
if(**!mb_part_vsp_flag[** mbPartIdx **]** &&(num_ref_idx_l0_active_minus1 > 0 || mb_field_decoding_flag) && MbPartPredMode(mb_type, mbPartIdx) != Pred_L1)		
ref_idx_l0[mbPartIdx **]**	2	te(v) | ae(v)
for(**!mb_part_vsp_flag[** mbPartIdx **]** &&(mbPartIdx = 0; mbPartIdx < NumMbPart(mb_type); mbPartIdx++)		
if((num_ref_idx_l1_active_minus1 > 0 || mb_field_decoding_flag) && MbPartPredMode(mb_type, mbPartIdx) != Pred_L0)		
ref_idx_l1[mbPartIdx **]**	2	te(v) | ae(v)
for(mbPartIdx = 0; mbPartIdx < NumMbPart(mb_type); mbPartIdx++)		
if(**!mb_part_vsp_flag[** mbPartIdx **]** &&(MbPartPredMode (mb_type, mbPartIdx) != Pred_L1)		
for(compIdx = 0; compIdx < 2; compIdx++)		
mvd_l0[mbPartIdx **][** 0 **][** compIdx **]**	2	se(v) | ae(v)
for(mbPartIdx = 0; mbPartIdx < NumMbPart(mb_type); mbPartIdx++)		
if(**!mb_part_vsp_flag[** mbPartIdx **]** &&(MbPartPredMode(mb_type, mbPartIdx) != Pred_L0)		
for(compIdx = 0; compIdx < 2; compIdx++)		
mvd_l1[mbPartIdx **][** 0 **][** compIdx **]**	2	se(v) | ae(v)
}		
}		

[0222]

[0223] 표 10의 예의 신택스에서, 슬라이스의 MB에 대한 MB 계층 신택스 구조는, MB의 유형이 B_Direct_16x16이고 VSP가 슬라이스에 대해 가능하게 되고 RefPicList0 또는 RefPicList1이 VSP 화상을 포함하면, vsp_direct_flag 신택스 엘리먼트를 포함할 수도 있다. MB의 유형이 B_Direct_16x16이면, B 슬라이스 내에 있는 MB는 직접 모드에서 코딩되고 (즉, 모션 정보는 시그널링되지 않지만 잔차 정보는 시그널링되고), MB는 더 작은 MB 파티션들로 파티셔닝되지 않는다. vsp_direct_flag는 MB가 VSP 화상으로부터 예측되는지의 여부를 나타낼 수도 있다. 예를 들어, 1과 동일한 vsp_direct_flag는 B_direct_16x16 모드를 갖는 MB가 VSP 프레임으로부터 예측됨을 나타낼 수도 있고 0과 동일한 vsp_direct_flag는 원래의 B_direct_16x16 모드가 사용됨을 나타낼 수도 있다. 더욱이, 표 10에 도시된 바와 같이, vsp_direct_flag는 MB 계층 신택스 구조가 transform_size_8x8_flag 신택스 엘리먼트를 포함하는지의 여부를 적어도 부분적으로 제어할 수도 있다. vsp_direct_flag에 대한 CABAC 코딩 프로세스는 3D-AVC 규격 초안 1에서의 mb_direct_type_flag에 대한 CABAC 코딩 프로세스와 동일할 수도 있다.

[0224] 표 11의 예에서, MB 예측 신택스 구조는, slice_vsp_flag가 1과 동일하고 VspRefExist가 1과 동일하면, mb_part_vsp_flag[mbPartIdx]를 포함할 수도 있다. 다른 예들에서, MB 예측 신택스 구조는 slice_vsp_flag가 1과 동일하면 mb_part_vsp_flag[mbPartIdx]를 포함할 수도 있다.

[0225] 본 개시물의 기법들의 제 7 구현예는, MB가 스킵 모드에서 인코딩되는지의 여부와, 만약 그렇다면, 그 MB가 VSP 화상에 기초하여 디코딩될 지의 여부를 나타내는 결합된 신택스 엘리먼트를 제공한다. 제 7 구현예에서, 슬라이스 데이터 신택스 구조들은 아래의 표 12의 신택스에 부합할 수도 있다.

[0226] **표 12 - 슬라이스 데이터 신택스**

slice_data( ) {	C	기술어
if(entropy_coding_mode_flag)		
while(!byte_aligned( ))		
cabac_alignment_one_bit	2	f(1)
CurrMbAddr = first_mb_in_slice * (1 + MbaffFrameFlag)		
moreDataFlag = 1		
prevMbSkipped = 0		
do {		
if(slice_type != I && slice_type != SI)		
if(!entropy_coding_mode_flag) {		
mb_skip_run	2	ue(v)
prevMbSkipped = (mb_skip_run > 0)		
for(i=0; i<mb_skip_run; i++)		
CurrMbAddr = NextMbAddress(CurrMbAddr)		
moreDataFlag = more_rbsp_data( )		
} else {		
if(slice_vsp_enable && VspRefExist) {		
mb_skip_idc	2	ae(v)
moreDataFlag = (mb_skip_idc>1)		
}else{		
mb_skip_flag	2	ae(v)
moreDataFlag = !mb_skip_flag		
}		
}		
if(moreDataFlag) {		
if(MbaffFrameFlag && (CurrMbAddr % 2 == 0 || (CurrMbAddr % 2 == 1 && prevMbSkipped)))		
mb_field_decoding_flag	2	u(1) | ae(v)
macroblock_layer( )	2 | 3 | 4	
}		
if(!entropy_coding_mode_flag)		
moreDataFlag = more_rbsp_data( )		
else {		
if(slice_type != I && slice_type != SI)		
prevMbSkipped = mb_skip_flag		
if(MbaffFrameFlag && CurrMbAddr % 2 == 0)		
moreDataFlag = 1		
else {		
end_of_slice_flag	2	ae(v)
moreDataFlag = !end_of_slice_flag		
}		
}		
CurrMbAddr = NextMbAddress(CurrMbAddr)		
} while(moreDataFlag)		
}		

[0227]

[0228]

[0229] 표 12의 예의 신택스에서, 슬라이스에 대한 슬라이스 데이터 신택스 구조는, VSP가 슬라이스에 대해 가능하게 되고 VSP 참조 화상이 슬라이스에 대해 존재하면, mb_skip_idc 신택스 엘리먼트를 포함할 수도 있다. 하나의 예에서, mb_skip_idc 신택스 엘리먼트가 0과 동일하면, 현재 MB는 VSP 화상으로부터 스킵된다. 이는 모션 또는 잔차가 현재 MB에 대해 시그널링되지 않는다는 것을 의미한다. 현재 MB는 mb_skip_idc 신택스 엘리먼트가 적용되는 MB이다. 이 예에서, mb_skip_idc 신택스 엘리먼트가 1과 동일하면, 현재 MB는 정상 스킵 모드를 사용하여 인코딩된다. 다르게 말하면, 현재 MB는 VSP 화상으로부터 스킵되지 않는다. 더욱이, 이 예에서, mb_skip_idc 신택스 엘리먼트가 2와 동일하면, 현재 MB는 스킵 모드를 사용하여 코딩되지 않는다.

[0230] 일부 예들에서, mb_skip_idc 신택스 엘리먼트는 부가적인 값들을 가질 수도 있다. 예를 들면, 하나의 예에

서, mb_skip_idc 신택스 엘리먼트가 2와 동일하면, 현재 MB는 전적으로 VSP 화상으로부터 예측되고 이에 따라 현재 MB에 대한 모션 정보는 시그널링되지 않지만 현재 MB에 대한 잔차 정보는 시그널링될 수도 있다. 다른 예에서, mb_skip_idc 신택스 엘리먼트가 3과 동일하면, 현재 MB는 전적으로 정상 직접 예측 모드로부터 예측되는 반면, MB의 잔차는 시그널링된다. 직접 예측 모드는 모션 벡터가 시그널링되지 않거나 또는 디코딩되지 않는 인터 예측의 유형이다. 직접 예측 모드에서, 참조 화상의 참조 인덱스는 시그널링되고 MB (또는 MB 파티션) 의 잔차는 참조 화상의 병치된 MB (또는 MB 파티션) 에 대하여 시그널링된다. 다른 예에서, MB에 연관된 신택스 엘리먼트는 전체 MB가 VSP 화상으로부터 예측되는 것과 그 MB에 대한 잔차가 시그널링된다는 것을 나타낼 수도 있다. 다시 말하면, 본 개시물의 다른 기법들과 유사하게, 전체 MV가 VSP로부터 예측되지만 잔차는 여전히 송신된다는 것을 나타내는 부가적인 모드가 MB에서 시그널링될 수 있다.

[0231] 다른 예에서, mb_skip_idc 신택스 엘리먼트가 1과 동일하면, 현재 MB는 VSP 화상으로부터 스킵된다. 이 예에서, 모션 또는 잔차 정보는 현재 MB에 대해 시그널링되지 않는다. 더욱이, 이 예에서, mb_skip_idc 신택스 엘리먼트가 0과 동일하면, 현재 MB는 정상 스킵 모드에서 코딩된다.

[0232] 덧붙여서, MB들에 대한 MB 계층 신택스 구조들은 아래의 표 13의 신택스에 부합할 수도 있다.

[0233] 표 13 - 매크로블록 계층 신택스

macroblock_layer( ) {	C	기술어
if (slice_vsp_flag && VspRefExist)		
vsp_mb_flag	2	ae(v)
if (!vsp_mb_flag)		
mb_type	2	ue(v) | ae(v)
if(mb_type == I_PCM) {		
while(!byte_aligned( ))		
pcm_alignment_zero_bit	3	f(1)
for(i = 0; i < 256; i++)		
pcm_sample_luma[i **]**	3	u(v)
for(i = 0; i < 2 * MbWidthC * MbHeightC; i++)		
pcm_sample_chroma[i **]**	3	u(v)
} else {		
noSubMbPartSizeLessThan8x8Flag = 1		
if(mb_type != I_NxN && MbPartPredMode(mb_type, 0) != Intra_16x16 && NumMbPart(mb_type) == 4) {		
if (!vsp_mb_flag)		
sub_mb_pred(mb_type)	2	
for(mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++)		
if(sub_mb_type[mbPartIdx] != B_Direct_8x8) {		
if(NumSubMbPart(sub_mb_type[mbPartIdx]) > 1)		
noSubMbPartSizeLessThan8x8Flag = 0		
} else if(!direct_8x8_inference_flag)		
noSubMbPartSizeLessThan8x8Flag = 0		
} else {		
if(transform_8x8_mode_flag && mb_type == I_NxN)		
transform_size_8x8_flag	2	u(1) | ae(v)
if (!vsp_mb_flag)		
mb_pred(mb_type)	2	
}		
if(MbPartPredMode(mb_type, 0) != Intra_16x16) {		
coded_block_pattern	2	me(v) | ae(v)
if(CodedBlockPatternLuma > 0 && transform_8x8_mode_flag && mb_type != I_NxN && noSubMbPartSizeLessThan8x8Flag && (vsp_mb_flag || (!vsp_mb_flag && (mb_type != B_Direct_16x16 || direct_8x8_inference_flag))))		
transform_size_8x8_flag	2	u(1) | ae(v)
}		
if(CodedBlockPatternLuma > 0 || CodedBlockPatternChroma > 0 || MbPartPredMode(mb_type, 0) == Intra_16x16) {		
mb_qp_delta	2	se(v) | ae(v)
residual(0, 15)	3 | 4	
}		
}		

[0234]

[0235]

}		

[0236] 표 13의 예의 MB 계층 신택스 구조는, VSP가 슬라이스에 대해 가능하게 되고 VSP 참조 화상이 그 슬라이스에 대해 존재하면, vsp_mb_flag 신택스 엘리먼트를 포함할 수도 있다. vsp_mb_flag 신택스 엘리먼트는, MB 계층 신택스 구조에 대응하는 전체 MB가 VSP 화상으로부터 예측되는지의 여부 및 그 MB에 대한 잔차 데이터가 시그널링되는지의 여부를 나타낸다. 예를 들어, vsp_mb_flag 신택스 엘리먼트가 1과 동일하면, 전체 MB는 VSP 화상으로부터 예측될 수도 있고 그 MB에 대한 잔차 데이터는 시그널링된다. 이 예에서, vsp_mb_flag 신택스 엘리먼트가 0과 동일하면, 전체 MB는 다른 코딩 모드들에 의해 예측될 수도 있다. 더욱이, 이 예에서, vsp_mb_flag 신택스 엘리먼트가 1과 동일한 경우, mb_type 신택스 엘리먼트는 MB 계층 신택스 구조에서 시그널링되지 않는다. vsp_mb_flag 신택스 엘리먼트가 MB 계층 신택스 구조에 존재하지 않는 경우, 비디오 디코더 (30) 는 vsp_mb_flag 신택스 엘리먼트가 0과 동일하다고 유추할 수도 있다.

[0237] 표 12 및 표 13에서 설명된 예의 신택스 구조들에서, mb_skip_idc 신택스 엘리먼트와는 다른 신택스 엘리먼트들

에 대한 콘텍스트 모델링은 본 개시물의 다른 곳에서 설명되는 바와 동일할 수도 있다. mb_skip_idc 신택스 엘리먼트의 콘텍스트들에 대한 초기 확률들은 P 슬라이스들 및 B 슬라이스들에 대해 상이한 값들로 초기화될 수도 있다. 다양한 예들에서, mb_skip_idc 신택스 엘리먼트는 다양한 방법들로 이진화될 수도 있고 콘텍스트는 mb_skip_idc 신택스 엘리먼트에 대해 다양한 방법들로 선택될 수도 있다.

[0238] 하나의 예에서, mb_skip_idc 신택스 엘리먼트는 MB 표시된 mbPAddrA 및 MB 표시된 mbPAddrB로부터의 정보를 사용하여 이진화될 수도 있다. mbPAddrA는 현재 MB 또는 현재 MB 파티션 좌측의 이웃 MB (또는 8x8 블록) 일 수도 있다. mbPAddrB는 현재 MB 또는 현재 MB 파티션 상측의 이웃 MB (또는 8x8 블록) 일 수도 있다. mbPAddrA 및 mbPAddrB 양쪽 모두가 이용가능하고 mbPAddrA 및 mbPAddrB 양쪽 모두가 VSP 스킵 모드를 사용하여 코딩되면, mb_skip_idc 신택스 엘리먼트는 아래의 표 14에 따라 이진화될 수도 있다:

[0239] **표 14 - mb_skip_idx의 이진화**

[0240]

mb_skip_idc의 값 (이름)	빈 스트링						
0 (VSP 스킵 모드)	1						
1 (정상 스킵 모드)	0	0					
2 (비 스킵 모드들)	0	1					
빈 인덱스 (binIdx)	*0*	*1*	*2*	*3*	*4*	*5*	*6*

[0241] 대안으로, mb_skip_idc 신택스 엘리먼트는 아래의 표 15에 따라 이진화될 수도 있다:

[0242] **표 15 - mb_skip_idx의 이진화**

[0243]

mb_skip_idc의 값 (이름)	빈 스트링						
0 (VSP 스킵 모드)	1						
1 (정상 스킵 모드)	0	1					
2 (비 스킵 모드들)	0	0					
빈 인덱스 (binIdx)	*0*	*1*	*2*	*3*	*4*	*5*	*6*

[0244] 엔트로피 코더가 표 14 또는 표 15에 따라 이진화된 mb_skip_idx 신택스 엘리먼트를 엔트로피 코딩하고 있는 경우, 엔트로피 코더는 이진화된 mb_skip_idx 신택스 엘리먼트의 각각의 빈에 대한 ctxIdxInc를 결정할 수도 있다. 이 예에서, 이진화된 mb_skip_idx 신택스 엘리먼트의 빈의 binIdx가 0과 동일하면 (즉, 엔트로피 코더가 mb_skip_idx 신택스 엘리먼트의 제 1 빈을 엔트로피 코딩하고 있으면), 빈에 대한 ctxIdxInc는 condTermFlagA 및 condTermFlagB의 합과 동일할 수도 있다. 다시 말하면, ctxIdxInc = condTermFlagA + condTermFlagB이다. 이 예에서, mbPAddrN이 이용가능하지 않거나 또는 mbPAddrN의 모드가 VSP 스킵 모드이면, condTermFlagN (여기서 N은 A 또는 B 중 어느 하나임) 은 0과 동일할 수도 있다. 그렇지 않으면, condTermFlagN은 1과 동일할 수도 있다.

[0245] 더욱이, 이 예에서, 빈의 binIdx가 1과 동일하면, 빈에 대한 ctxIdxInc는 condTermFlagA 더하기 condTermFlagB와 동일할 수도 있다. 빈의 binIdx가 1과 동일한 경우, mbPAddrN이 이용가능하지 않거나 또는 mbPAddrN의 모드가 정상 스킵 모드이면, condTermFlagN (여기서 N은 A 또는 B 중 어느 하나) 은 0과 동일할 수도 있다. 그렇지 않으면, condTermFlagN은 1과 동일할 수도 있다.

[0246] 그렇지 않고, mbPAddrA 또는 mbPAddrB 중 어느 하나가 이용가능하지 않거나 또는 mbPAddrA 또는 mbPAddrB가 VSP 스킵 모드에서 인코딩되지 않으면, mb_skip_idc 신택스 엘리먼트는 아래의 표 16에 따라 이진화될 수도 있다:

[0247] **표 16 - mb_skip_idc의 이진화**

[0248]

mb_skip_idc의 값 (이름)	빈 스트링						
0 (정상 스킵 모드)	1						
1 (VSP 스킵 모드)	0	0					
2 (비 스킵 모드들)	0	1					
빈 인덱스 (binIdx)	*0*	*1*	*2*	*3*	*4*	*5*	*6*

[0249] 대안으로, mbPAddrA 또는 mbPAddrB 중 어느 하나가 이용가능하지 않거나 또는 mbPAddrA 또는 mbPAddrB가 VSP 스킵 모드에서 인코딩되지 않으면, mb_skip_idc 신택스 엘리먼트는 아래의 표 17에 따라 이진화될 수도 있다:

[0250] **표 17 - mb_skip_idc의 이진화**

[0251]

mb_skip_idc의 값 (이름)	빈 스트링						
0 (정상 스킵 모드)	1						
1 (VSP 스킵 모드)	0	1					
2 (비 스킵 모드들)	0	0					
빈 인덱스 (binIdx)	*0*	*1*	*2*	*3*	*4*	*5*	*6*

[0252] mbPAddrA 또는 mbPAddrB 중 어느 하나가 이용가능하지 않거나 또는 mbPAddrA 또는 mbPAddrB가 VSP 스킵 모드에서 인코딩되지 않으면, 엔트로피 코더는 이진화된 mb_skip_idx 신택스 엘리먼트의 각각의 빈에 대한 ctxIdxInc를 결정할 수도 있다. 이 예에서, 이진화된 mb_skip_idx 신택스 엘리먼트의 빈의 binIdx가 0과 동일하면 (즉, 엔트로피 코더가 mb_skip_idx 신택스 엘리먼트의 제 1 빈을 엔트로피 코딩하고 있으면), 빈에 대한 ctxIdxInc는 condTermFlagA 및 condTermFlagB의 합과 동일할 수도 있다. 다시 말하면, ctxIdxInc = condTermFlagA + condTermFlagB이다. 이 예에서, mbPAddrN이 이용가능하지 않거나 또는 mbPAddrN의 모드가 정상 스킵 모드이면, condTermFlagN (여기서 N은 A 또는 B 중 어느 하나임) 은 0과 동일할 수도 있다. 그렇지 않으면, condTermFlagN은 1과 동일할 수도 있다.

[0253] 더욱이, 이 예에서, 빈의 binIdx가 1과 동일하면, 빈에 대한 ctxIdxInc는 condTermFlagA 더하기 condTermFlagB와 동일할 수도 있다. 빈의 binIdx가 1과 동일한 경우, mbPAddrN이 이용가능하지 않거나 또는 mbPAddrN의 모드가 VSP 스킵 모드이면, condTermFlagN (여기서 N은 A 또는 B 중 어느 하나임) 은 0과 동일할 수도 있다. 그렇지 않으면, condTermFlagN은 1과 동일할 수도 있다.

[0254] 다른 예들에서, 빈의 ctxIdxInc는 단일 공간적 이웃에만 의존할 수도 있다. 예를 들어, ctxIdxInc는 condTermFlagA와 동일할 수도 있다. 다른 예에서, ctxIdxInc는 condTermFlagB와 동일할 수도 있다. 이 예에서, condTermFlagA 및 condTermFlagB는 위에서처럼 정의될 수도 있다.

[0255] 본 개시물의 기법들의 제 8 구현예는 제 7 구현예와 유사하지만, 이진화 및 콘텍스트 선택 방법들은 상이하다. 제 8 구현예에서, P 슬라이스들에서의 P MB들에 대한 및 B 슬라이스들에서의 B MB들에 대한 이진화 체계들은 아래의 표 18에 따라 특정된다:

[0256] **표 18 - mb_skip_idc의 이진화**

[0257]

mb_skip_idc의 값 (이름)	빈 스트링						
0 (정상 스킵 모드/VSP 스킵 모드)	0	0					
1 (VSP 스킵 모드/정상 스킵 모드)	0	1					
2 (비 스킵 모드들)	1						
빈 인덱스 (binIdx)	*0*	*1*	*2*	*3*	*4*	*5*	*6*

[0258] 대안으로, 제 8 구현예에서, P 슬라이스들에서의 P MB들에 대한 및 B 슬라이스들에서의 B MB들에 대한 이진화 프로세스는 아래의 표 19에 따라 특정된다:

[0259] **표 19 - mb_skip_idc의 이진화**

[0260]

mb_skip_idc의 값 (이름)	빈 스트링						
0 (정상 스킵 모드/VSP 스킵 모드)	1	0					
1 (VSP 스킵 모드/정상 스킵 모드)	1	1					
2 (비 스킵 모드들)	0						
빈 인덱스 (binIdx)	*0*	*1*	*2*	*3*	*4*	*5*	*6*

[0261] 제 8 구현예에서, 엔트로피 코더가 MB_skip_idc 신택스 엘리먼트를 코딩하고 있는 경우, 엔트로피 코더는 이진화된 mb_skip_idc 신택스 엘리먼트의 각각의 빈에 대한 ctxIdxInc를 결정할 수도 있다. 예를 들면, mb_skip_idc를 디코딩하는 경우, mb_skip_idc에 대한 ctxIdxInc는 binIdx에 의존하여 특정될 수도 있다. 빈에 대한 ctxIdxInc를 결정함에 있어서, mbPAddrA는 현재 MB 좌측의 이웃 MB (또는 8x8 블록) 를 나타낼 수도 있고 mbPAddrB는 현재 MB 또는 MB 파티션 상측의 이웃 MB (또는 8x8 블록) 를 나타낼 수도 있다.

[0262] 이진화된 mb_skip_idc 신택스 엘리먼트의 빈의 binIdx이 0과 동일하면, 엔트로피 인코더는 빈에 대한 ctxIdxInc를 다양한 방법들로 결정할 수도 있다. 예를 들어, 이진화된 mb_skip_idc 신택스 엘리먼트의 빈의 binIdx가 0과 동일하면, 빈에 대한 ctxIdxInc는 condTermFlagA 및 condTermFlagB의 합과 동일할 수도 있다. 이 예에서, mbPAddrN이 이용가능하지 않거나 또는 mbPAddrN의 모드가 스킵 모드 (정상 스킵 모드 또는 VSP 스킵 모드 중 어느 하나) 이면, condTermFlagN (여기서 N은 A 또는 B 중 어느 하나임) 은 0과 동일할 수도 있다. 그렇

지 않으면,이 예에서, condTermFlagN은 1과 동일하다. 다른 예에서, 이진화된 mb_skip_idc 신택스 엘리먼트의 빈의 binIdx가 0과 동일하면, 빈에 대한 ctxIdxInc는 condTermFlagA 및 condTermFlagB의 합과 동일할 수도 있다. 이 예에서, mbPAddrN이 이용가능하지 않거나 또는 mbPAddrN의 모드가 스킵 모드가 아니면 (정상 스킵 모드도 또는 VSP 스킵 모드도 아니면), condTermFlagN (여기서 N은 A 또는 B 중 어느 하나임) 은 0과 동일할 수도 있다. 그렇지 않으면,이 예에서, condTermFlagN은 1과 동일하다. 다른 예에서, 이진화된 mb_skip_idc 신택스 엘리먼트의 빈의 binIdx가 0과 동일하면, 빈에 대한 ctxIdxInc는 condTermFlagN과 동일할 수도 있는데, 여기서 condTermFlagN (여기서 N은 A 또는 B 중 어느 하나임) 은 이 단락의 다른 예들에 의해 설명된 방식으로 정의된다.

[0263] 이진화된 mb_skip_idc 신택스 엘리먼트의 빈의 binIdx이 0보다 크면, 엔트로피 코더는 빈에 대한 ctxIdxInc를 다양한 방법들로 결정할 수도 있다. 예를 들어, 이진화된 mb_skip_idc 신택스 엘리먼트의 빈의 binIdx가 0보다 크면, 빈에 대한 ctxIdxInc는 condTermA, condTermB 및 ctxOffset의 합과 동일할 수도 있다. 이 예에서, condTermFlagN (여기서 N은 A 또는 B 중 어느 하나일 수 있음) 및 ctxOffset은 양쪽 모두가 mbPAddrN의 모드에 의존할 수도 있다. 더욱이, 이 예에서, mbPAddrA 및 mbPAddrB 양쪽 모두가 이용가능하고 VSP 스킵 모드로서 코딩되면, ctxOffset은 0과 동일할 수도 있다. 그렇지 않으면, ctxOffset은 3과 동일할 수도 있다. 하나의 대안에서, ctxOffset은 0으로 설정된다. 다른 대안에서, mbPAddrA 및 mbPAddrB 양쪽 모두가 이용가능하고 VSP 스킵 모드로서 코딩되면, ctxOffset은 3과 동일하고 그렇지 않으면 0과 동일하다. 이 예에서, mbPAddrA 및 mbPAddrB 양쪽 모두가 이용가능하고 VSP 스킵 모드로서 코딩되면, condTermFlagA 및 condTermFlagB는 0과 동일하게 설정될 수도 있다. 대안으로, mbPAddrN (여기서 N은 A 또는 B 중 어느 하나일 수 있음) 이 이용가능하지 않거나 또는 mbPAddrN의 모드가 VSP 스킵 모드이면, condTermFlagN은 0으로 설정되고 그렇지 않으면 1로 설정된다.

[0264] 이전 단락의 예를 계속하여, mbPAddrA 또는 mbPAddrB 중 어느 하나가 이용가능하지 않거나 또는 VSP 스킵 모드로서 코딩되지 않으면, condTermFlagN (여기서 N은 A 또는 B 중 어느 하나임) 은 0과 동일하게 설정될 수도 있다. 그렇지 않으면, condTermFlagN은 1과 동일하게 설정될 수도 있다. 하나의 대안적 예에서, mbPAddrN이 이용가능하지 않거나 또는 mbPAddrN의 모드가 VSP 스킵 모드이면, condTermFlagN은 0으로 설정되고 그렇지 않으면 1로 설정된다. 다른 대안적 예에서, mbPAddrN이 이용가능하지 않거나 또는 mbPAddrN의 모드가 VSP 스킵 모드가 아니면, condTermFlagN은 0과 동일하게 설정될 수도 있고 그렇지 않으면 1과 동일하게 설정될 수도 있다. 다른 대체예에서, mbPAddrN이 이용가능하지 않거나 또는 mbPAddrN의 모드가 정상 스킵 모드이면, condTermFlagN은 0으로 설정될 수도 있다. 그렇지 않으면, condTermFlagN은 0과 동일하게 설정될 수도 있다. 다른 대체예에서, mbPAddrN이 이용가능하지 않거나 또는 mbPAddrN의 모드가 정상 스킵 모드가 아니면, condTermFlagN은 0과 동일하게 설정될 수도 있다. 그렇지 않으면, condTermFlagN은 1로 설정될 수도 있다.

[0265] 제 8 구현예에 따른 다른 예에서, 이진화된 mb_skip_idc 신택스 엘리먼트의 빈의 binIdx가 0보다 크면, 엔트로피 코더는 condTermFlagN 및 ctxOffset의 합으로서 빈에 대한 ctxIdxInc를 결정할 수도 있다. 이 예에서, condTermFlagN (여기서 N은 A 또는 B 중 어느 하나일 수도 있음) 및 ctxOffset은 양쪽 모두가 mbPAddrN의 모드에 의존할 수도 있다. 더욱이, 이 예에서, mbPAddrN이 이용가능하고 VSP 스킵 모드로서 코딩되면, ctxOffset은 0으로 설정될 수도 있다. 그렇지 않으면, ctxOffset은 2로 설정될 수도 있다. 하나의 대안으로, ctxOffset은 0으로 설정된다. 다른 대체예에서, mbPAddrN가 이용가능하고 VSP 스킵 모드로서 코딩되면, ctxOffset은 2로 설정된다. 그렇지 않으면, 이 대체예에서, ctxOffset은 0으로 설정된다.

[0266] 이전 단락의 예를 계속하여, mbPAddrN가 이용가능하고 VSP 스킵 모드로서 코딩되면, condTermFlagN은 다음과 같이 정의될 수도 있다: mbPAddrN이 이용가능하지 않거나 또는 mbPAddrN의 모드가 정상 스킵 모드이면, condTermFlagN은 0과 동일하게 설정되고, 그렇지 않으면 condTermFlagN은 1로 설정된다. 그렇지 않고, mbPAddrN이 이용가능하지 않거나 또는 VSP 스킵 모드로서 코딩되지 않으면, condTermFlagN은 다음처럼 정의될 수도 있다: mbPAddrN이 이용가능하지 않거나 또는 mbPAddrN의 모드가 정상 스킵 모드이면, condTermFlagN은 0과 동일하게 설정되고, 그렇지 않으면 1과 동일하게 설정된다. 하나의 대체예에서, mbPAddrN이 이용가능하지 않거나 또는 mbPAddrN의 모드가 VSP 스킵 모드이면, condTermFlagN은 0과 동일하게 설정된다. 그렇지 않으면, condTermFlagN은 1과 동일하게 설정된다. 다른 대체예에서, mbPAddrN이 이용가능하지 않거나 또는 mbPAddrN의 모드가 VSP 스킵 모드가 아니면, condTermSkipFlagN은 0과 동일하게 설정된다. 그렇지 않으면, 그것은 1로 설정된다. 다른 대체예에서, mbPAddrN이 이용가능하지 않거나 또는 mbPAddrN의 모드가 정상 스킵 모드이면, condTermFlagN은 0과 동일하게 설정된다. 그렇지 않으면, condTermFlagN은 0과 동일하게 설

정된다. 다른 대체예에서, mbPAddrN이 이용가능하지 않거나 또는 mbPAddrN의 모드가 정상 스킵 모드가 아니면, condTermSkipFlagN은 0과 동일하게 설정된다. 그렇지 않으면, 그것은 1로 설정된다.

[0267] 위에서 언급했듯이, 본 개시물의 일부 기법들은 다수의 VSP 화상들을 지원한다. 이러한 기법들은 본 개시물에서 설명되는 제 7 또는 제 8 구현예들 (또는 다른 구현예들) 에 추가될 수도 있다. 이러한 기법들에서, 이용가능한 VSP 화상들의 총 수는 비트스트림의 다양한 부분들로 시그널링될 수도 있다. 예를 들어, 이용가능한 VSP 화상들의 총 수는 시퀀스 파라미터 세트, 화상 파라미터 세트, 슬라이스 헤더에서, 또는 비트스트림의 다른 부분에서 시그널링될 수도 있다. 이용가능한 다수의 VSP 화상들이 있는 경우, 신택스 엘리먼트는 적용가능한 VSP 화상의 인덱스를 나타내기 위해 시그널링될 수도 있다. 다음의 논의는 VSP 화상들의 수가 1보다 큰 경우 적용할 수도 있다. 아래의 표 20은, 슬라이스 데이터에 대한 일 예의 신택스 구조를 나타낸다. 표 20의 신택스 구조는 MB에 적용가능한 VSP 화상의 인덱스를 특정하는 신택스 엘리먼트 vsp_ref_idx를 포함한다.

[0268] **표 20 - 슬라이스 데이터 신택스**

slice_data( ) {	C	기술어
if(entropy_coding_mode_flag)		
while(!byte_aligned( ))		
cabac_alignment_one_bit	2	f(1)
CurrMbAddr = first_mb_in_slice * (1 + MbaffFrameFlag)		
moreDataFlag = 1		
prevMbSkipped = 0		
do {		
if(slice_type != I && slice_type != SI)		
if(!entropy_coding_mode_flag) {		
mb_skip_run	2	ue(v)
prevMbSkipped = (mb_skip_run > 0)		
for(i=0; i<mb_skip_run; i++)		
CurrMbAddr = NextMbAddress(CurrMbAddr)		
moreDataFlag = more_rbsp_data( )		
} else {		
if(slice_vsp_enable && VspRefExist) {		
mb_skip_idc	2	ae(v)
if(MbVspSkip)		
vsp_ref_idx	2	ae(v)
moreDataFlag = (mb_skip_idc>1)		
}else{		
mb_skip_flag	2	ae(v)
moreDataFlag = !mb_skip_flag		
}		
}		
if(moreDataFlag) {		
if(MbaffFrameFlag && (CurrMbAddr % 2 == 0 || (CurrMbAddr % 2 == 1 && prevMbSkipped)))		
mb_field_decoding_flag	2	u(1) | ae(v)
macroblock_layer( )	2 | 3 | 4	
}		
if(!entropy_coding_mode_flag)		
moreDataFlag = more_rbsp_data( )		
else {		
if(slice_type != I && slice_type != SI)		
prevMbSkipped = mb_skip_flag		
if(MbaffFrameFlag && CurrMbAddr % 2 == 0)		
moreDataFlag = 1		
else {		
end_of_slice_flag	2	ae(v)
moreDataFlag = !end_of_slice_flag		
}		
}		

[0269]

CurrMbAddr = NextMbAddress(CurrMbAddr)		
} while(moreDataFlag)		
}		

[0270]

[0271] 표 20의 예의 신택스 구조에서, vsp_ref_idx 신택스 엘리먼트는 예측을 위해 사용될 VSP 화상의 인덱스를 특정할 수도 있다. 슬라이스 데이터 신택스 구조는, 현재 MB가 mb_skip_idc 신택스 엘리먼트에 의해 나타내어진 VSP 화상으로부터 도출될 수 있는 VSP 화상으로부터 스킵되면, vsp_ref_idx 신택스 엘리먼트를 포함할 수도 있다. 더욱이, 슬라이스 데이터 신택스 구조는, vsp_mb_flag 신택스 엘리먼트가 1과 동일하면, vsp_ref_idx 신택스 엘리먼트를 포함할 수도 있다. 덧붙여서, 슬라이스 데이터 신택스 구조는, mb_part_vsp_flag[mbPartIdx]가 1과 동일하면, vsp_mb_flag syntax 엘리먼트를 포함할 수도 있다. 이전에 논의했듯이, mb_part_vsp_flag[mbPartIdx]는 mbPartIdx로 식별된 현재 MB 파티션이 VSP 화상으로부터 예측되는지의 여부를 나타낸다. 슬라이스 데이터 신택스 구조는 또한, sub_mb_vsp_flag[mbPartIdx]가 1과 동일하면, vsp_mb_flag 신택스 엘리먼트를 포함할 수도 있다. 이전에 논의했듯이, sub_mb_vsp_flag[mbPartIdx]는 mbPartIdx로 식별된 현재 MB 파티션 (8x8) 이 VSP 화상으로부터 예측되는지의 여부를 나타낸다.

[0272] 아래의 표 21은, MV 계층 신택스 엘리먼트에 대한 일 예의 신택스를 도시한다. 표 21에 도시된 바와 같이, MB 계층 신택스 구조는 복수의 이용가능한 VSP 화상들 중에서 VSP 화상을 식별하는 vsp_ref_idx 신택스 엘리먼트를 포함할 수도 있다.

[0273] 표 21 - 매크로블록 계층 신택스

macroblock_layer( ) {	C	기술어
if (slice_vsp_flag && VspRefExist) {		
vsp_mb_flag	2	ae(v)
if(vsp_mb_flag)		
vsp_ref_idx	2	ae(v)
}		
if (! vsp_mb_flag)		
mb_type	2	ue(v) | ae(v)
if(mb_type == I_PCM) {		
while(!byte_aligned( ))		
pcm_alignment_zero_bit	3	f(1)
for(i = 0; i < 256; i++)		
pcm_sample_luma[i **]**	3	u(v)
for(i = 0; i < 2 * MbWidthC * MbHeightC; i++)		
pcm_sample_chroma[i **]**	3	u(v)
} else {		
noSubMbPartSizeLessThan8x8Flag = 1		
if(mb_type != I_NxN && MbPartPredMode(mb_type, 0) != Intra_16x16 && NumMbPart(mb_type) == 4) {		
if (!vsp_mb_flag)		
sub_mb_pred(mb_type)	2	
for(mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++)		
if(sub_mb_type[mbPartIdx] != B_Direct_8x8) {		
if(NumSubMbPart(sub_mb_type[mbPartIdx]) > 1)		
noSubMbPartSizeLessThan8x8Flag = 0		
} else if(!direct_8x8_inference_flag)		
noSubMbPartSizeLessThan8x8Flag = 0		
} else {		
if(transform_8x8_mode_flag && mb_type == I_NxN)		
transform_size_8x8_flag	2	u(1) | ae(v)
if (!vsp_mb_flag)		
mb_pred(mb_type)	2	
}		
if(MbPartPredMode(mb_type, 0) != Intra_16x16) {		
coded_block_pattern	2	me(v) | ae(v)
if(CodedBlockPatternLuma > 0 && transform_8x8_mode_flag && mb_type != I_NxN && noSubMbPartSizeLessThan8x8Flag && (vsp_mb_flag || (!vsp_mb_flag && (mb_type != B_Direct_16x16 || direct_8x8_inference_flag))))		
transform_size_8x8_flag	2	u(1) | ae(v)
}		
if(CodedBlockPatternLuma > 0 || CodedBlockPatternChroma > 0 || MbPartPredMode(mb_type, 0) == Intra_16x16) {		
mb_qp_delta	2	se(v) | ae(v)

[0274]

residual(0, 15)	3 | 4	
}		
}		
}		

[0275]

[0276] 표 21에서, 동일한 조건들이 발생하는 경우 vsp_ref_idx 신택스 엘리먼트의 시맨틱스는 표 20을 참조하여 위에서 설명된 것들과 동일할 수도 있고 MB 계층 신택스 구조는 vsp_ref_idx 신택스 엘리먼트를 포함할 수도 있다.

[0277] 더욱이, 아래의 표 22는 MB 예측 신택스 구조에 대한 일 예의 신택스를 도시한다. 표 22에 도시된 바와 같이, MB 예측 신택스 구조는 mbPartIdx에 의해 식별된 MB 파티션에 대해, 복수의 이용가능한 VSP 화상들 중에서 VSP 화상을 식별하는 vsp_ref_idx[mbPartIdx] 신택스 엘리먼트를 포함할 수도 있다.

[0278] 표 22 - 매크로블록 예측 신택스

mb_pred(mb_type) {	C	기술어
if(MbPartPredMode(mb_type, 0) == Intra_4x4 || MbPartPredMode(mb_type, 0) == Intra_16x16) {		
if(MbPartPredMode(mb_type, 0) == Intra_4x4)		
for(luma4x4BlkIdx=0; luma4x4BlkIdx<16; luma4x4BlkIdx++) {		
prev_intra4x4_pred_mode_flag[luma4x4BlkIdx **]**	2	u(1) | ae(v)
if(!prev_intra4x4_pred_mode_flag[luma4x4BlkIdx])		
rem_intra4x4_pred_mode[luma4x4BlkIdx **]**	2	u(3) | ae(v)
}		
intra_chroma_pred_mode	2	ue(v) | ae(v)
} else if(MbPartPredMode(mb_type, 0) != Direct) {		
for(mbPartIdx = 0; mbPartIdx < NumMbPart(mb_type); mbPartIdx++)		
if (slice_vsp_flag && VspRefExist) //vsp_mb_flag is not 1		
mb_part_vsp_flag[mbPartIdx **]**	2	ae(v)
if(mb_part_vsp_flag[mbPartIdx])		
vsp_ref_idx[mbPartIdx]	2	ae(v)
if(!mb_part_vsp_flag[mbPartIdx] &&(num_ref_idx_l0_active_minus1 > 0 || mb_field_decoding_flag) && MbPartPredMode(mb_type, mbPartIdx) != Pred_L1)		
ref_idx_l0[mbPartIdx **]**	2	te(v) | ae(v)
for(!mb_part_vsp_flag[mbPartIdx] &&(mbPartIdx = 0; mbPartIdx < NumMbPart(mb_type); mbPartIdx++)		
if((num_ref_idx_l1_active_minus1 > 0 || mb_field_decoding_flag) && MbPartPredMode(mb_type, mbPartIdx) != Pred_L0)		
ref_idx_l1[mbPartIdx **]**	2	te(v) | ae(v)
for(mbPartIdx = 0; mbPartIdx < NumMbPart(mb_type); mbPartIdx++)		
if(!mb_part_vsp_flag[mbPartIdx] &&(MbPartPredMode (mb_type, mbPartIdx) != Pred_L1)		
for(compIdx = 0; compIdx < 2; compIdx++)		
mvd_l0[mbPartIdx **][** 0 **][** compIdx **]**	2	se(v) | ae(v)
for(mbPartIdx = 0; mbPartIdx < NumMbPart(mb_type); mbPartIdx++)		
if(!mb_part_vsp_flag[mbPartIdx] &&(MbPartPredMode(mb_type, mbPartIdx) != Pred_L0)		
for(compIdx = 0; compIdx < 2; compIdx++)		
mvd_l1[mbPartIdx **][** 0 **][** compIdx **]**	2	se(v) | ae(v)
}		
}		

[0279]

[0280] 표 22에서, 동일한 조건들이 발생하는 경우 vsp_ref_idx 신택스 엘리먼트의 시맨틱스는 표 20을 참조하여 위에서 설명된 것들과 동일할 수도 있고 MB 예측 신택스 구조는 vsp_ref_idx 신택스 엘리먼트를 포함할 수도 있다.

[0281] 아래의 표 23은, 서브 MB 예측 신택스 구조에 대한 일 예의 신택스를 도시한다. 표 23에 도시된 바와 같이, 서브 MB 예측 신택스 구조는, mbPartIdx에 의해 식별된 서브 MB 파티션에 대해, 복수의 이용가능한 VSP 화상들 중에서 VSP 화상을 식별하는 vsp_ref_idx[mbPartIdx] 신택스 엘리먼트를 포함할 수도 있다.

[0282] **표 23 - 서브 매크로블록 예측 신택스**

sub_mb_pred(mb_type) {	C	기술어
for(mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++) {		
if (slice_vsp_flag && VspRefExist) // vsp_mb_flag is not 1		
sub_mb_vsp_flag[mbPartIdx **]**		
if(sub_mb_vsp_flag[mbPartIdx])		
vsp_ref_idx[mbPartIdx]	2	ae(v)
else		
sub_mb_type[mbPartIdx **]**	2	ue(v) | ae(v)
}		
for(mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++)		
if(!sub_mb_vsp_flag[mbPartIdx]) &&(num_ref_idx_l0_active_minus1 > 0 || mb_field_decoding_flag) && mb_type != P_8x8ref0 && sub_mb_type[mbPartIdx] != B_Direct_8x8 && SubMbPredMode(sub_mb_type[mbPartIdx]) != Pred_L1)		
ref_idx_l0[mbPartIdx **]**	2	te(v) | ae(v)
for(mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++)		
if(!sub_mb_vsp_flag[mbPartIdx]) && (num_ref_idx_l1_active_minus1 > 0 || mb_field_decoding_flag) && sub_mb_type[mbPartIdx] != B_Direct_8x8 && SubMbPredMode(sub_mb_type[mbPartIdx]) != Pred_L0)		
ref_idx_l1[mbPartIdx **]**	2	te(v) | ae(v)
for(mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++)		
if(!sub_mb_vsp_flag[mbPartIdx]) &&(sub_mb_type[mbPartIdx] != B_Direct_8x8 && SubMbPredMode(sub_mb_type[mbPartIdx]) != Pred_L1)		
for(subMbPartIdx = 0; subMbPartIdx < NumSubMbPart(sub_mb_type[mbPartIdx]); subMbPartIdx++)		
for(compIdx = 0; compIdx < 2; compIdx++)		
mvd_l0[mbPartIdx **][** subMbPartIdx **][** compIdx **]**	2	se(v) | ae(v)
for(mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++)		
if(!sub_mb_vsp_flag[mbPartIdx]) &&sub_mb_type[mbPartIdx] != B_Direct_8x8 && SubMbPredMode(sub_mb_type[mbPartIdx]) != Pred_L0)		
for(subMbPartIdx = 0; subMbPartIdx < NumSubMbPart(sub_mb_type[mbPartIdx]); subMbPartIdx++)		
for(compIdx = 0; compIdx < 2; compIdx++)		
mvd_l1[mbPartIdx **][** subMbPartIdx **][** compIdx **]**	2	se(v) | ae(v)
}		

[0283]

[0284] 표 23에서, 동일한 조건들이 발생하는 경우 vsp_ref_idx 신택스 엘리먼트의 시맨틱스는 표 20을 참조하여 위에서 설명된 것들과 동일할 수도 있고 MB 예측 신택스 구조는 vsp_ref_idx 신택스 엘리먼트를 포함할 수도 있다.

[0285] vsp_ref_idx에 대한 이진화 프로세스는 ref_idx_l0 및 ref_idx_l1과 동일할 수도 있다. vsp_ref_idx의 콘텍스트 선택는 ref_idx_l0 및 ref_idx_l1과 유사한 방법으로 수행될 수 있다. H.264/AVC 표준은 ref_idx_l0 및 ref_idx_l1에 대한 이진화 프로세스 및 콘텍스트 선택을 기술한다. vsp_ref_idx에 대한 콘텍스트 모델링은 이웃 MB들 또는 MB 파티션들의 vsp_ref_idx의 값들에 기초할 수도 있지만, ref_idx_l0 또는 ref_idx_l1의 값들에는 기초하지 않을 수도 있다.

[0286] 일부 예들에서, MB 또는 MB 파티션은 2 개의 VSP 화상들에 기초하여 예측될 수도 있다. 일부 이러한 예들에서, MB 예측 신택스 구조들은 아래의 표 24에 의해 기술된 신택스를 가질 수도 있다.

[0287] **표 24 - 매크로블록 예측 신택스**

mb_pred(mb_type) {	C	기술어
if(MbPartPredMode(mb_type, 0) == Intra_4x4 || MbPartPredMode(mb_type, 0) == Intra_16x16) {		
if(MbPartPredMode(mb_type, 0) == Intra_4x4)		
for(luma4x4BlkIdx=0; luma4x4BlkIdx<16; luma4x4BlkIdx++) {		
prev_intra4x4_pred_mode_flag[luma4x4BlkIdx **]**	2	u(1) | ae(v)
if(!prev_intra4x4_pred_mode_flag**[** luma4x4BlkIdx **]**)		
rem_intra4x4_pred_mode[luma4x4BlkIdx **]**	2	u(3) | ae(v)
}		
intra_chroma_pred_mode	2	ue(v) | ae(v)
} else if(MbPartPredMode(mb_type, 0) != Direct) {		
for(mbPartIdx = 0; mbPartIdx < NumMbPart(mb_type); mbPartIdx++)		
if (slice_vsp_flag && NumMbPart(mb_type) != 1 && VspRefExist) {		
if (MbPartPredMode(mb_type, mbPartIdx) != Pred_L1)		
mb_part_vsp_flag_L0[mbPartIdx **]**	2	ae(v)
if (MbPartPredMode(mb_type, mbPartIdx) != Pred_L0)		
mb_part_vsp_flag_L1[mbPartIdx **]**	2	ae(v)
}		
if(!mb_part_vsp_flag_L0[mbPartIdx] &&(num_ref_idx_l0_active_minus1 > 0 || mb_field_decoding_flag) && MbPartPredMode(mb_type, mbPartIdx) != Pred_L1)		
ref_idx_l0[mbPartIdx **]**	2	te(v) | ae(v)
for (mbPartIdx = 0; mbPartIdx < NumMbPart(mb_type) && !mb_part_vsp_flag_L1[mbPartIdx]; mbPartIdx++)		
if((num_ref_idx_l1_active_minus1 > 0 || mb_field_decoding_flag) && MbPartPredMode(mb_type, mbPartIdx) != Pred_L0)		
ref_idx_l1[mbPartIdx **]**	2	te(v) | ae(v)
for(mbPartIdx = 0; mbPartIdx < NumMbPart(mb_type); mbPartIdx++)		
if(!mb_part_vsp_flag_L0**[** mbPartIdx **]** && MbPartPredMode (mb_type, mbPartIdx) != Pred_L1)		
for(compIdx = 0; compIdx < 2; compIdx++)		
mvd_l0[mbPartIdx **][** 0 **][** compIdx **]**	2	se(v) | ae(v)
for(mbPartIdx = 0; mbPartIdx < NumMbPart(mb_type); mbPartIdx++)		
if(!mb_part_vsp_flag_L1**[** mbPartIdx **]** && MbPartPredMode(mb_type, mbPartIdx) != Pred_L0)		
for(compIdx = 0; compIdx < 2; compIdx++)		
mvd_l1[mbPartIdx **][** 0 **][** compIdx **]**	2	se(v) | ae(v)
}		
}		

[0288]

[0289] 표 24의 예의 신택스에서, MB 예측 신택스 구조는 mb_part_vsp_flag_l0[mbPartIdx] 신택스 엘리먼트 및 mb_part_vsp_flagl1[mbPartIdx] 신택스 엘리먼트를 포함할 수도 있다. mb_part_vsp_flag_l0[mbPartIdx] 신택스 엘리먼트는 현재 MB 파티션이 RefPicList0에서의 정상 참조 화상으로부터 예측되는지 또는 VSP 화상으로부터 예측되는지를 나타낸다. 예를 들어, mb_part_vsp_flag_l0[mbPartIdx]가 1과 동일하면, 현재 MB 파티션은 예측을 위한 RefPicList0에서의 정상 화상으로부터 예측되지 않고, VSP 화상으로부터 예측된다. 이 예에서, mb_part_vsp_flag_l0[mbPartIdx]가 0과 동일하면, MB 파티션은 예측을 위한 RefPicList0에서의 정상 화상으로부터 예측된다. 마찬가지로, mb_part_vsp_flag_l1[mbPartIdx] 신택스 엘리먼트는, 현재 MB 파티션이 RefPicList1에서의 정상 참조 화상으로부터 예측되는지 또는 VSP 화상으로부터 예측되는지를 나타낸다. 예를 들어, mb_part_vsp_flag_l1[mbPartIdx]가 1과 동일하면, 현재 MB 파티션은 예측을 위한 RefPicList에서의 정상 화상으로부터 예측되지 않고 VSP 화상으로부터 예측된다. 이 예에서, mb_part_vsp_flag_l1[mbPartIdx] 플래그가 0과 동일하면, MB 파티션은 예측을 위한 RefPicList1에서의 정상 화상으로부터 예측된다.

[0290] 더욱이, MB 또는 MB 파티션이 2 개의 VSP 화상들, 서브 MB 예측 신택스 구조들에 기초하여 예측될 수도 있는 일부 예들은, 아래의 표 25에 의해 기술되는 신택스를 가질 수도 있다.

[0291] **표 25 - 서브 매크로블록 예측 신택스**

sub_mb_pred(mb_type) {	C	기술어
for(mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++) {		
if (slice_vsp_flag && VspRefExist) // vsp_mb_flag is not 1		
sub_mb_vsp_flag[mbPartIdx **]**		
if (!sub_mb_vsp_flag[mbPartIdx])		
sub_mb_type[mbPartIdx **]**	2	ue(v) | ae(v)
if (slice_vsp_flag && VspRefExist && (NumSubMbPart (sub_mb_type[mbPartIdx]) = = 1 || sub_mb_type[mbPartIdx] = = B_Direct_8x8)) {		
if(SubMbPredMode(sub_mb_type[mbPartIdx]) != Pred_L1)		
sub_mb_vsp_flag_l0[mbPartIdx **]**	2	ae(v)
if(SubMbPredMode(sub_mb_type[mbPartIdx]) != Pred_L0)		
sub_mb_vsp_flag_l1[mbPartIdx **]**	2	ae(v)
}		
}		
for(mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++)		
if(!sub_mb_vsp_flag_l0[mbPartIdx] &&(num_ref_idx_l0_active_minus1 > 0 || mb_field_decoding_flag) && mb_type != P_8x8ref0 && sub_mb_type[mbPartIdx] != B_Direct_8x8 && SubMbPredMode(sub_mb_type[mbPartIdx]) != Pred_L1)		
ref_idx_l0[mbPartIdx **]**	2	te(v) | ae(v)
for(mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++)		
if(!sub_mb_vsp_flag_l1[mbPartIdx] && (num_ref_idx_l1_active_minus1 > 0 || mb_field_decoding_flag) && sub_mb_type[mbPartIdx] != B_Direct_8x8 && SubMbPredMode(sub_mb_type[mbPartIdx]) != Pred_L0)		
ref_idx_l1[mbPartIdx **]**	2	te(v) | ae(v)
for(mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++)		
if(!sub_mb_vsp_flag_l0[mbPartIdx] &&(sub_mb_type[mbPartIdx] != B_Direct_8x8 && SubMbPredMode(sub_mb_type[mbPartIdx]) != Pred_L1)		
for(subMbPartIdx = 0; subMbPartIdx < NumSubMbPart(sub_mb_type[mbPartIdx]); subMbPartIdx++)		
for(compIdx = 0; compIdx < 2; compIdx++)		
mvd_l0[mbPartIdx **][** subMbPartIdx **][** compIdx **]**	2	se(v) | ae(v)
for(mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++)		
if(!sub_mb_vsp_flag_l1[mbPartIdx] &&sub_mb_type[mbPartIdx] != B_Direct_8x8 && SubMbPredMode(sub_mb_type[mbPartIdx]) != Pred_L0)		
for(subMbPartIdx = 0; subMbPartIdx < NumSubMbPart(sub_mb_type[mbPartIdx]); subMbPartIdx++)		
for(compIdx = 0; compIdx < 2; compIdx++)		
mvd_l1[mbPartIdx **][** subMbPartIdx **][** compIdx **]**	2	se(v) | ae(v)
}		

[0292]

[0293] 표 25의 예의 신택스에서, 서브 MB 예측 신택스 구조는 sub_mb_vsp_flag[mbPartIdx] 신택스 엘리먼트, sub_mb_vsp_flag_l0[mbPartIdx] 신택스 엘리먼트, 및 sub_mb_vsp_flag_l1[mbPartIdx] 신택스 엘리먼트를 포함할 수도 있다. sub_mb_vsp_flag[mbPartIdx]는 현재 MB 파티션 (8x8) 이 VSP 화상으로부터 예측되는지의 여부를 나타낸다. 예를 들어, sub_mb_vsp_flag[mbPartIdx]가 1과 동일하면, 현재 MB 파티션 (8x8) 은 VSP 화상으로부터 예측된다. 이 예에서, sub_mb_vsp_flag[mbPartIdx]가 0과 동일하면, 현재 MB 파티션 (8x8) 은 VSP 화상으로부터 예측되지 않는다. sub_mb_vsp_flag[mbPartIdx]가 존재하지 않는 경우, sub_mb_vsp_flag[mbPartIdx]는 0과 동일하다고 유추될 수도 있다. sub_mb_vsp_flag[mbPartIdx]가 1과 동일한 경우, mb_part_vsp_flag_l0[mbPartIdx] 및 mb_part_vsp_flag_l1[mbPartIdx] 양쪽 모두는 1과 동일한 것으로 도출된다.

[0294] sub_mb_vsp_flag_l0[mbPartIdx]는 현재 MB 파티션이 RefPicList0에서의 정상 참조 화상으로부터 예측되는지 또는 VSP 화상으로부터 예측되는지를 나타낸다. 예를 들어, sub_mb_vsp_flag_l0[mbPartIdx]가 1과 동일하면, 현재 MB 파티션은 예측을 위한 RefPicList0에서의 정상 화상으로부터 예측되지 않지만, VSP 화상으로부터 예측된다. 이 예에서, sub_mb_vsp_flag_l0[mbPartIdx]가 0과 동일하면, MB 파티션은 예측을 위한 RefPicList0에서의 정상 화상으로부터 예측된다. 마찬가지로, sub_mb_vsp_flag_l1[mbPartIdx] 신택스 엘리먼트는, 현재 MB 파티션이 RefPicList1에서의 정상 참조 화상으로부터 예측되는지 또는 VSP 화상으로부터 예측되는지를 나타

낸다. 예를 들어, sub_mb_vsp_flag_l1[mbPartIdx]가 1과 동일하면, 현재 MB 파티션은 예측을 위한 RefPicList1에서의 정상 화상으로부터 예측되지 않지만, VSP 화상으로부터 예측된다. 이 예에서, sub_mb_vsp_flag_l1[mbPartIdx] 플래그가 0과 동일하면, MB 파티션은 예측을 위한 RefPicList1에서의 정상 화상으로부터 예측된다.

[0295] 일부 예들에서, VSP 화상은 참조 화상 리스트에 포함되고 ref_idx는 sub_mb_vsp_flagl0, sub_mb_vsp_flagl1, mb_part_vsp_flag_l0, 및 mb_part_vsp_flag_l1의 값들을 도출하는데 사용된다. 하나의 이러한 예에서, VSP 화상은 RefPicList0의 ref_idx_l0_vsp 엔트리로서 포함될 수도 있다. 이 예에서, ref_idx_l0_vsp는 VSP 화상의 RefPicList0에서의 포지션을 나타낸다. 더욱이, 이 예에서, VSP 화상은 RefPicList1의 ref_idx_l1_vsp 엔트리로서 포함될 수도 있다. 다르게 말하면, ref_idx_l1_vsp는 VSP 화상의 RefPicList1에서의 포지션을 나타낸다. VSP 화상이 RefPicList0 내에 있지 않으면, ref_idx_l0_vsp는 -1과 동일하다. VSP 화상이 RefPicList1 내에 있지 않으면, ref_idx_l1_vsp는 -1과 동일하다. 이는 아래의 표 26 및 표 27에서 도시되어 있다.

[0296] **표 26 - 매크로블록 예측 신택스**

mb_pred(mb_type) {	C	기술어
if(MbPartPredMode(mb_type, 0) == Intra_4x4 || MbPartPredMode(mb_type, 0) == Intra_16x16) {		
if(MbPartPredMode(mb_type, 0) == Intra_4x4)		
for(luma4x4BlkIdx=0; luma4x4BlkIdx<16; luma4x4BlkIdx++) {		
prev_intra4x4_pred_mode_flag[luma4x4BlkIdx **]**	2	u(1) | ae(v)
if(!prev_intra4x4_pred_mode_flag**[** luma4x4BlkIdx **]**)		
rem_intra4x4_pred_mode[luma4x4BlkIdx **]**	2	u(3) | ae(v)
}		
intra_chroma_pred_mode	2	ue(v) | ae(v)
} else if(MbPartPredMode(mb_type, 0) != Direct) {		
for(mbPartIdx = 0; mbPartIdx < NumMbPart(mb_type); mbPartIdx++)		
if(num_ref_idx_l0_active_minus1 > 0 || mb_field_decoding_flag) && MbPartPredMode(mb_type, mbPartIdx) != Pred_L1) {		
ref_idx_l0[mbPartIdx **]**	2	te(v) | ae(v)
mb_part_vsp_flag_L0[mbPartIdx]= (ref_idx_l0[mbPartIdx] == ref_idx_l0_vsp)		
}		
for (mbPartIdx = 0; mbPartIdx < NumMbPart(mb_type); mbPartIdx++)		
if((num_ref_idx_l1_active_minus1 > 0 || mb_field_decoding_flag) && MbPartPredMode(mb_type, mbPartIdx) != Pred_L0) {		
ref_idx_l1[mbPartIdx **]**	2	te(v) | ae(v)
mb_part_vsp_flag_L1[mbPartIdx]= (ref_idx_l1[mbPartIdx] == ref_idx_l1_vsp)		
}		
for(mbPartIdx = 0; mbPartIdx < NumMbPart(mb_type); mbPartIdx++)		
if(!mb_part_vsp_flag_L0**[** mbPartIdx **]** && MbPartPredMode (mb_type, mbPartIdx) != Pred_L1)		
for(compIdx = 0; compIdx < 2; compIdx++)		
mvd_l0[mbPartIdx **][** 0 **][** compIdx **]**	2	se(v) | ae(v)
for(mbPartIdx = 0; mbPartIdx < NumMbPart(mb_type); mbPartIdx++)		
if(!mb_part_vsp_flag_L1**[** mbPartIdx **]** && MbPartPredMode(mb_type, mbPartIdx) != Pred_L0)		
for(compIdx = 0; compIdx < 2; compIdx++)		
mvd_l1[mbPartIdx **][** 0 **][** compIdx **]**	2	se(v) | ae(v)
}		
}		

[0297]

[0298] 표 26에 도시된 바와 같이, ref_idx_l0[mbPartIdx]가 ref_idx_l0_vsp와 동일하면, mb_part_vsp_flag_L0[mbPartIdx]는 1과 동일할 수도 있다. 위에서 나타낸 바와 같이, 1과 동일한 mb_part_vsp_flag_L0[mbPartIdx]는, 현재 MB 파티션이 RefPicList0에서의 정상 화상으로부터 예측되지 않지만, 대신에 VSP 화상으로부터 예측된다는 것을 나타낼 수도 있다. 마찬가지로, 비디오 디코더 (30) 는,

ref_idx_l1[mbPartIdx]가 ref_idx_l1_vsp와 동일하면, mb_part_vsp_flag_L1[mbPartIdx]가 1과 동일하다고 결정할 수도 있다. 1과 동일한 mb_part_vsp_flag_L0[mbPartIdx]는, 현재 MB 파티션이 RefPicList1에서의 정상 화상으로부터 예측되지 않지만, 대신에 VSP 화상으로부터 예측된다는 것을 나타낼 수도 있다. 따라서, 표 26의 예의 신택스에서, 현재 MB 파티션이 정상 화상으로부터 예측되는지 또는 VSP 화상으로부터 예측되는지를 나타내기 위해 별개의 신택스 엘리먼트를 시그널링하는 것은 불필요할 수도 있다. 대신, ref_idx_l0 신택스 엘리먼트는 현재 MB 파티션이 정상 화상으로부터 예측되는지 또는 VSP 화상으로부터 예측되는지를 결정할 목적으로 재사용될 수 있다.

[0299] **표 27 - 서브 매크로블록 예측 신택스**

sub_mb_pred(mb_type) {	C	기술어
for(mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++)		
if (slice_vsp_flag && VspRefExist) // vsp_mb_flag is not 1		
sub_mb_vsp_flag[mbPartIdx **]**		
if (!sub_mb_vsp_flag[mbPartIdx])		
sub_mb_type[mbPartIdx **]**	2	ue(v) | ae(v)
for(mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++)		
if((num_ref_idx_l0_active_minus1 > 0 || mb_field_decoding_flag) && mb_type != P_8x8ref0 && sub_mb_type[mbPartIdx] != B_Direct_8x8 && SubMbPredMode(sub_mb_type[mbPartIdx]) != Pred_L1) {		
ref_idx_l0[mbPartIdx **]**	2	te(v) | ae(v)
sub_mb_vsp_flag_l0[mbPartIdx] = (ref_idx_l0[mbPartIdx] = = ref_idx_l0_vsp)		
}		
for(mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++)		
if(!sub_mb_vsp_flag_l1[mbPartIdx] && (num_ref_idx_l1_active_minus1 > 0 || mb_field_decoding_flag) && sub_mb_type[mbPartIdx] != B_Direct_8x8 && SubMbPredMode(sub_mb_type[mbPartIdx]) != Pred_L0) {		
ref_idx_l1[mbPartIdx **]**	2	te(v) | ae(v)
sub_mb_vsp_flag_l1[mbPartIdx] = (ref_idx_l1[mbPartIdx] = = ref_idx_l1_vsp)		
}		
for(mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++)		
if(!sub_mb_vsp_flag_l0[mbPartIdx] &&(sub_mb_type[mbPartIdx] != B_Direct_8x8 && SubMbPredMode(sub_mb_type[mbPartIdx]) != Pred_L1)		
for(subMbPartIdx = 0; subMbPartIdx < NumSubMbPart(sub_mb_type[mbPartIdx]); subMbPartIdx++)		
for(compIdx = 0; compIdx < 2; compIdx++)		
mvd_l0[mbPartIdx **][** subMbPartIdx **][** compIdx **]**	2	se(v) | ae(v)
for(mbPartIdx = 0; mbPartIdx < 4; mbPartIdx++)		
if(!sub_mb_vsp_flag_l1[mbPartIdx] &&sub_mb_type[mbPartIdx] != B_Direct_8x8 && SubMbPredMode(sub_mb_type[mbPartIdx]) != Pred_L0)		
for(subMbPartIdx = 0; subMbPartIdx < NumSubMbPart(sub_mb_type[mbPartIdx]); subMbPartIdx++)		
for(compIdx = 0; compIdx < 2; compIdx++)		
mvd_l1[mbPartIdx **][** subMbPartIdx **][** compIdx **]**	2	se(v) | ae(v)
}		

[0300]

[0301] 표 27에 도시된 바와 같이, 비디오 디코더 (30) 는, ref_idx_l0[mbPartIdx]가 ref_idx_l0_vsp와 동일하면, sub_mb_vsp_flag_L0[mbPartIdx]가 1과 동일하다고 결정할 수도 있다. 위에서 나타낸 바와 같이, 1과 동일한 sub_mb_vsp_flag_L0[mbPartIdx]는, 현재 MB 파티션 (8x8) 이 RefPicList0에서의 정상 화상으로부터 예측되지 않지만, 대신에 VSP 화상으로부터 예측된다는 것을 나타낼 수도 있다. 마찬가지로, 비디오 디코더 (30) 는, ref_idx_l1[mbPartIdx]가 ref_idx_l1_vsp와 동일하면, sub_mb_vsp_flag_L1[mbPartIdx]가 1과 동일하다고 결정할 수도 있다. 1과 동일한 sub_mb_vsp_flag_L0[mbPartIdx]는, 현재 MB 파티션 (8x8) 이 RefPicList1에서의 정상 화상으로부터 예측되지 않지만, 대신에 VSP 화상으로부터 예측된다는 것을 나타낼 수도 있다. 따라서, 표 27의 예의 신택스에서, 현재 MB 파티션 (8x8) 이 정상 화상으로부터 예측되는지 또는 VSP 화상으로부터 예측되는지를 나타내기 위해 별개의 신택스 엘리먼트를 시그널링하는 것은 불필요할 수도 있다. 대신, ref_idx_l0 신택스 엘리먼트는 현재 MB 파티션 (8x8) 이 정상 화상으로부터 예측되는지 또는 VSP 화상으로부터 예측되는지를 결정할 목적으로 재사용될 수 있다.

[0302] 모션 벡터 예측은 현재 비디오 유닛, 이를테면 MB 또는 MB 파티션에 대한 모션 벡터들을 결정하는 프로세스이다. 대체로, 비디오 코더, 이를테면 비디오 인코더 (20) 또는 비디오 디코더 (30) 가 현재 비디오 유닛의 모션 벡터를 결정하는 경우, 비디오 코더는 깊이 뷰 성분에서의 대표 깊이 값을 결정할 수도 있다.

일부 예들에서, 대표 깊이 값은 현재 비디오 유닛의 샘플 블록들과는 병치된 깊이 값들의 블록에서의 깊이 값들의 평균 깊이 값일 수도 있다. 비디오 디코더는 그 다음에, 대표 깊이 값에 적어도 부분적으로 기초하여, 현재 비디오 유닛에 대한 디스패리티 벡터를 결정할 수도 있다. 디스패리티 벡터의 원점은, 현재 비디오 유닛과 동일한 액세스 유닛의 참조 텍스처 뷰 성분 내의, 현재 비디오 유닛의 샘플 블록 내의 로케이션과는 병치된 로케이션일 수도 있다. 참조 텍스처 뷰 성분의 참조 비디오 유닛의 샘플 블록은 디스패리티 벡터의 종점을 포함할 수도 있다. 참조 비디오 유닛이 다른 액세스 유닛의 텍스처 뷰로부터 예측되면, 현재 비디오 유닛의 모션 벡터는 참조 비디오 유닛의 모션 벡터와 동일할 수도 있다. 다른 경우들에서, 현재 비디오 유닛의 모션 벡터는 디스패리티 벡터와 동일할 수도 있다.

[0303] 깊이 기반 모션 파라미터 예측 (depth-based motion parameter prediction; DBMP) 에서, 인코딩된 비디오 유닛 (예컨대, MB, MB 파티션, 등) 의 각각의 화소에 대해, 모션 정보, 이를테면 모션 벡터들 및 참조 인덱스들은, 동일한 시간적 인스턴스 (즉, 액세스 유닛) 에서 이웃 뷰에서의 인코딩된 비디오 유닛들로부터의 이미 코딩된 디스패리티 맵들의 사용으로 직접 유추될 수도 있다. 다르게 말하면, 비디오 코더는 각각의 화소에 대한 모션 벡터들 및 참조 인덱스들을 디스패리티 맵들에 기초하여 결정할 수도 있다. 이 프로시저는 인코딩된 비디오 유닛의 각각의 화소에 대해 독립적으로 반복될 수도 있다. 결과적으로, DBMP에서, 모션 벡터 및 참조 인덱스들은 비트스트림으로 송신되지 않지만, 대신에 비디오 디코더 (30) 에서의 참조 뷰로부터 획득된다.

[0304] 일부 예들에서, VSP는 모션 벡터 예측으로 단일화 (unification) 될 수도 있다. 예를 들면, VSP는 홀 충전 (hole-filling) 없이 후방 워핑 (backward warping) 으로 단순화될 수 있다. 후방 워핑은 워핑의 역이다.

워핑은 참조 뷰 성분들에서의 샘플들이, 디스패리티에 기초하여, 타겟 뷰 성분에서의 로케이션들에 매핑되는 프로세스이다. 홀 충전은 타겟 뷰 성분에서의 백그라운드 오브젝트들의 가려지지 않은 부분들을 나타내는 화소들 내를 충전하는 프로세스이다. 더욱이, VSP가 모션 벡터 예측과 단일화되는 경우, 디스패리티 값 또는 깊이 값은 전체 4x4 블록에 적용될 수도 있다. 이런 식으로, 뷰 합성 예측은, 모션 벡터가 깊이 또는 디스패리티로부터 도출되고 MB 또는 MB 파티션의 각각의 4x4 블록과 연관되는 전통적인 모션 보상으로 단순화될 수 있는 것이 가능하다. 이 경우, VSP 모드는 이른바 단일화된 VSP 모드로 재설계된다.

[0305] 위의 표 3, 표 4 및 표 5에 기재된 신택스는, VSP 및 모션 벡터 예측의 단일화를 지원하기 위해 재사용될 수도 있다. 그러나, 표 3, 표 4 및 표 5에 기재된 신택스가 뷰 합성 예측 및 모션 벡터 예측의 단일화를 지원하기 위해 재사용되는 경우 여러 문제들이 발생할 수도 있다. 예를 들어, 위의 표 3, 표 4 및 표 5에 기재된 바와 같은 정상 VSP에서, 각각의 블록에 대응하는 모션 벡터는 0인 반면 디스패리티 벡터들은 VSP를 사용하여 블록에 대한 모션 벡터들을 도출하는데 사용될 수도 있다. 따라서, 모션 벡터는 항상 0과 동일해야 하는 것은 아니다. 다른 예에서는, 정상 VSP에서, 모션 벡터들이 0이기 때문에 VSP 모드들로 코딩된 이웃 MB들에 대한 도출된 모션 벡터들의 예측은 없다. 그러나, 단일화된 VSP 모드에서, VSP를 사용하여 블록들에 대해 도출된 모션 벡터들은 항상 0이 아니다.

[0306] 본 개시물의 하나 이상의 기법들에 따라, VSP 화상은 참조 화상 리스트에 추가되지 않는다. 따라서, 비디오 인코더 (20) 는 VSP 화상을 사용하여 코딩된 블록에 대해 모션 정보 (예컨대, 참조 인덱스들, 모션 벡터들 등) 를 시그널링할 필요가 없을 수도 있다. 더욱이, 뷰 합성이 단일화된 VSP 모드에 의해 실현되기 때문에, 비디오 코더가 VSP 화상을 생성하는 것이 불필요할 수도 있다.

[0307] 예를 들면, VSP가 깊이 기반 모션 벡터 예측 또는 디스패리티 모션 벡터 기반 모션 벡터 예측으로 단일화되는 경우, 위의 표 3, 표 4 및 표 5에 기재된 바와 동일한 신택스 설계를 사용함으로써, 단일화된 VSP 모드의 시그널링은 MB 또는 MB 파티션이 VSP 화상으로부터 예측되는지의 여부를 나타내기 위해 MB 또는 MB 파티션 레벨에 도입된다. 더욱이, VSP가 깊이 기반 모션 벡터 예측으로 단일화되는 경우, 단일화된 VSP 모드에 속하는 블록들의 각각은, 깊이 맵에 의해 도출된 그것의 모션 벡터들 또는 다른 공간적/시간적 이웃 블록들로부터의 디스패리티 모션 벡터들을 가진다.

[0308] 일부 예들에서, 단일화된 VSP 모드는 1과 동일한 sub_mb_vsp_flag, mb_part_vsp_flag, 또는 vsp_mb_flag 중 하나에 의해 식별된다. 다른 예들에서, sub_mb_vsp_flag, mb_part_vsp_flag, 또는 vsp_mb_flag 중 어느 것도 시그널링되지 않는 경우, 단일화된 VSP 모드가 특정 값으로 시그널링되고 있는 참조 인덱스에 의해 식별될 수도 있다. 이들 예들에서, sub_mb_vsp_flag, mb_part_vsp_flag, 및 vsp_mb_flag의 엔트로피 코딩 (예컨대, 엔

트로피 인코딩 또는 엔트로피 디코딩) 은 본 개시물의 다른 곳에서 설명되는 바와 동일할 수도 있다.

[0309] 비디오 코더가 참조 화상에 기초하여 VSP 화상을 생성한다. 슬라이스 헤더가 VSP 화상을 생성하게 하는 참조 화상을 포함하는 뷰를 나타낼 수도 있다. 3D-AVC 사양은 이미, 슬라이스 헤더에서, 뷰 합성을 위해 사용될 뷰를 나타낸다. 본 개시물은 슬라이스 헤더에서 나타내어진 그리고 VSP에 대한 참조 뷰 화상과는 동일한 액세스 유닛 내에 있는 뷰의 디코딩된 화상을 나타내기 위해 "refViewPicVSP"를 사용할 수도 있다. 비디오 코더는 refViewPicVSP를 참조 화상 리스트에 이미 추가하였을 수도 있는데, 뷰 합성을 위해 사용된 뷰가 통상은 정상 MVC-형태 뷰간 예측을 위해 사용되는 뷰이기 때문이다. 비디오 코더는 참조 인덱스 (RefPicList0에 대응함) 를 다음과 같이 결정할 수도 있다. 0 내지 num_ref_idx_l0_active_minus1의 각각의 *i*에 대해, RefPicList0[*i*]가 refViewPicVSP와 동일하면, 비디오 코더는 참조 인덱스를 *i*로 설정할 수도 있다. num_ref_idx_l0_active_minus1은 RefPicList0에서의 액티브 참조 화상들의 수 빼기 1을 나타낼 수도 있다. *i*가 num_ref_idx_l0_active_minus1 더하기 1과 동일하면, 비디오 코더는 참조 인덱스가 *i*와 동일하다고 그리고 refViewPicVSP가 RefPicList0 속에 마지막 엔트리로서 추가된다고 결정할 수도 있다.

[0310] 대안으로, 플래그들 (예컨대, vsp_mb_flag, mb_part_vsp_flag, 등) 이 단일화된 VSP 모드를 나타내는데 사용되지 않고 참조 인덱스 (ref_idx_lx) 가 명시적으로 시그널링되는 경우, 덧붙여서, 참조 인덱스 (ref_idx_lx) 는 다음과 같이 재설정될 수도 있다. 0 내지 num_ref_idx_lX_active_minus1의 각각의 *i*에 대해, RefPicListX[*i*]가 RefViewPicVSP와 동일하면, ref_idx_lx는 *i*로 설정된다.

[0311] 참조 인덱스에 대한 이 도출 프로세스는 비디오 디코더 (30) 에서 참조 화상 리스트들의 사이즈를 감소시키는데 사용될 수도 있다. 실제적으로, 참조 화상 리스트에 추가되는 참조 인덱스 ref_idx_lx에 대응하는 화상은 없다. 대신, ref_idx_lx가 재설정된 후, ref_idx_lx는 단일화된 VSP 모드를 시그널링하기 위해 전적으로 사용될 수도 있고 단일화된 VSP 모드로 코딩된 블록은 RefPicList[ref_idx_lx]로부터 모션 보상된다.

[0312] 현재 비디오 유닛 (예컨대, MB, MB 파티션, 또는 서브 MB 파티션) 이 뷰 합성 모드로 코딩되는 예들에서, 모션 벡터들의 도출은 모션 벡터의 현재 비디오 유닛의 각각의 4x4 블록에 대한 도출을 포함할 수도 있다. 모션 벡터들은 모두가 refViewPicVSP를 참조할 수도 있고 통상 RefPicList0에 대해서만 예측되는데, 이는 현재 비디오 유닛이 단방향으로만 예측된다는 것을 의미한다.

[0313] 현재 텍스처 뷰 성분의 현재 비디오 유닛에 대한 모션 벡터를 도출하는 다수의 방법들이 있을 수도 있다. 예를 들어, 비디오 코더는 모션 벡터를 대응하는 디코딩된 깊이 뷰 블록으로부터 도출할 수도 있다. 이 예에서, 비디오 코더는 대표 깊이 값을 얻을 수도 있다. 깊이 화상이 현재 텍스처 화상의 1/4 해상도인 경우 (즉, 현재 텍스처 뷰 성분에서의 하나의 4x4 블록은 깊이 뷰 성분에서의 하나의 2x2 지역에 대응함), 비디오 코더는 2x2 지역에서의 4 개의 값들을 다양한 방법들로 프로세싱할 수도 있다. 예를 들어, 4 개의 값들의 최대 값은 대표 깊이 값으로서 반환될 수도 있다. 다른 예에서, 4 개의 값들의 평균 값은 대표 깊이 값으로서 반환될 수도 있다. 다른 예에서, 4 개의 값들의 중간 값은 대표 깊이 값으로서 반환된다. 깊이 화상이 현재 텍스처 화상의 1/4 해상도가 아닌 경우 (즉, 현재 텍스처 뷰 성분에서의 하나의 4x4 블록은 깊이 뷰 성분에서의 하나의 NxM 지역에 대응함), 비디오 코더는 대표 깊이 값을 다양한 방법들로 결정할 수도 있다. 예를 들어, 비디오 디코더는 대표 깊이 값을 대응하는 NxM 지역의 모든 깊이 화소들의 평균이 되게 결정할 수도 있다. 다른 예에서, 비디오 디코더는 대표 깊이 값을 대응하는 NxM 지역의 4 개의 코너 깊이 화소들의 평균이 되게 결정할 수도 있다. 다른 예에서, 비디오 디코더는 대표 깊이 값을 대응하는 NxM 지역의 4 개의 코너 깊이 화소들의 최대가 되게 결정할 수도 있다. 다른 예에서, 비디오 디코더는 대표 깊이 값을 대응하는 NxM 지역의 4 개의 코너 깊이 값들의 중간 값이 되게 결정할 수도 있다. 대표 깊이 값을 결정한 후, 비디오 디코더는 대표 깊이 값을 디스패리티 벡터로 컨버팅하고 모션 벡터를 디스패리티 벡터로 설정할 수도 있다. 디스패리티 벡터 (및 모션 벡터) 의 y-축 성분은 0과 동일할 수도 있다.

[0314] 더욱이, 깊이 뷰 (즉, 샘플, view_id를 갖는 뷰) 가 디코딩되지 않은 경우 대응하는 참조 뷰의 깊이 맵으로부터 모션 벡터를 도출하는 다수의 방법들이 있을 수도 있다. 더구나, 현재 블록의 공간적/시간적 이웃 블록으로부터 모션 벡터를 도출하는 다수의 방법들이 있을 수도 있다. 대안으로, 도출된 모션 벡터의 수직 성분은 0으로 추가로 재설정될 수도 있다.

[0315] 위에서 설명된 일부 예들에서, VSP 모드로 또는 그 VSP 모드 없이 코딩된 2 개의 이웃 블록들은 2 개의 참조 화상들로부터 인터 예측된 것으로 간주될 수도 있다. 따라서, 2 개의 이웃 블록들 사이의 경계의 경계 강도는, 2 개의 이웃 블록들 양쪽 모두가 2 개의 상이한 참조 화상들로부터 인터 예측되는 경우와 동일할 수도 있다. 블록화해제 필터의 이 설계는, 단일화된 VSP 모드가 이웃 블록들 중 하나에서 사용되는 경우 블록화

해제 필터를 위해 사용될 수 있다.

[0316] 예를 들어, 이웃 블록들의 양쪽 모두가 VSP 모드들에 의해 예측되는 경우, 필터 유닛 (이를테면 필터 유닛 (114 또는 160)) 은, 양쪽 모두의 블록들이 동일한 참조 화상으로부터 예측되고 이에 따라 모션 벡터들이 비교되는 블록화해제 필터에서의 경우와 유사하게, 도출된 모션 벡터들을 비교할 수도 있다. 한편, 하나의 블록이 단일화된 VSP 모드를 사용하여 코딩되고 다른 블록이 뷰간 예측으로 코딩되고 refViewPicVSP가 뷰간 예측 참조 화상과 동일한 경우, 2 개의 블록들은 동일한 참조 화상을 가진다고 간주된다. 더욱이, 하나의 블록이 단일화된 VSP 모드를 사용하여 코딩되고 다른 블록이 시간적 참조 화상으로 코딩되는 경우, 단일화된 VSP 모드로 코딩된 블록은 인트라 코딩된 블록으로서 간주될 수도 있다.

[0317] 위에서 설명된 바와 같이, 본 개시물의 기법들은 효율적인 VSP 메커니즘을 HEVC 기반 3DV에 제공할 수도 있다. 이들 기법들은 다양한 방법들로 구현될 수도 있다. HEVC 기반 3DV에 대한 제 1 구현예에 따라, 시퀀스 파라미터 세트는 HEVC 기반 3DV에서 VSP를 지원하는 플래그들을 포함할 수도 있다. 아래의 표 28은, HEVC 기반 3DV에서의 시퀀스 파라미터 세트에 대한 일 예의 신택스의 부분을 도시한다.

[0318] **표 28 - 시퀀스 파라미터 세트 RBSP 신택스**

seq_parameter_set_rbsp( ) {	기술어
profile_idc	u(8)
reserved_zero_8bits /* equal to 0 */	u(8)
level_idc	u(8)
…	
if(sps_extension_flag) {	
…	
seq_vsp_flag	u(1)
if(seq_vsp_flag)	
vsp_in_cu_level_only_flag	u(1)
…	
}	
rbsp_trailing_bits( )	
}	

[0319]

[0320] 표 28의 예에서, SPS는 seq_vsp_flag 신택스 엘리먼트를 포함한다. seq_vsp_flag 신택스 엘리먼트가 특정 값 (예컨대, 1) 을 가지면, SPS는 vsp_in_cu_level_only_flag 신택스 엘리먼트를 포함한다. seq_vsp_flag 신택스 엘리먼트는 VSP 모드가 SPS를 참조하는 모든 화상들에 대해 가능하게 되는지의 여부를 나타낸다. 예를 들어, 1과 동일한 seq_vsp_flag는 VSP 모드가 SPS를 참조하는 모든 화상들에 대해 가능하게 된다는 것을 나타낼 수도 있다. 이 예에서, 0과 동일한 seq_vsp_flag는 그 VSP 모드가 가능하지 않게 됨을 나타낼 수도 있다. 더욱이, 이 예에서, seq_vsp_flag가 존재하지 않는 경우, 비디오 디코더 (30) 는 seq_vsp_flag를 0과 동일한 것으로 유추할 수도 있다.

[0321] vsp_in_cu_level_only_flag 신택스 엘리먼트는 VSP 모드가 CU 레벨에만 적용되는지의 여부를 나타낼 수도 있다. 예를 들어, 1과 동일한 vsp_in_cu_level_only_flag는 VSP 모드가 CU 레벨에만 적용됨을 나타낼 수도 있다. 따라서, vsp_in_cu_level_only_flag가 1과 동일하면, 하나의 PU는 VSP 화상으로부터 예측될 수도 있지만 다른 PU는 다른 모드들에 의해 예측될 수도 있다. 이 예에서, 0과 동일한 vsp_in_cu_level_only_flag는 VSP 모드가 PU 레벨에 적용됨을 나타낼 수도 있다. 더욱이, 이 예에서, vsp_in_cu_level_only_flag가 존재하지 않는 경우, 비디오 디코더 (30) 는 vsp_in_cu_level_only_flag를 0과 동일한 것으로 유추할 수도 있다. 다른 예들에서, vsp_in_cu_level_only_flag는 존재하지 않고 VSP 모드는 항상 PU 레벨에 적용될 수도 있다.

[0322] HEVC 기반 3DV에 대한 제 1 구현예에서, 슬라이스 헤더들은 HEVC 기반 3DV에서 VSP를 지원하는 플래그들을 포함할 수도 있다. 아래의 표 29는 HEVC 기반 3DV에서의 슬라이스 헤더에 대한 일 예의 신택스를 도시한다.

[0323] 표 29 - 슬라이스 헤더 신택스

slice_header( ) {	기술어
first_slice_in_pic_flag	u(1)
if(first_slice_in_pic_flag = = 0)	
slice_address	u(v)
slice_type	ue(v)
entropy_slice_flag	u(1)
if(!entropy_slice_flag) {	
pic_parameter_set_id	ue(v)
if(output_flag_present_flag)	
pic_output_flag	u(1)
if(**separate_colour_plane_flag** = = 1)	
colour_plane_id	u(2)
if(IdrPicFlag) {	
idr_pic_id	ue(v)
no_output_of_prior_pics_flag	u(1)
} else {	
pic_order_cnt_lsb	u(v)
short_term_ref_pic_set_sps_flag	u(1)
if(!short_term_ref_pic_set_sps_flag)	
short_term_ref_pic_set(num_short_term_ref_pic_sets)	
else	
short_term_ref_pic_set_idx	u(v)
if(long_term_ref_pics_present_flag) {	
num_long_term_pics	ue(v)
for(i = 0; i < num_long_term_pics; i++) {	
delta_poc_lsb_lt[i]	ue(v)
delta_poc_msb_present_flag[i]	u(1)
if(delta_poc_msb_present_flag[i])	
delta_poc_msb_cycle_lt_minus1[i]	ue(v)
used_by_curr_pic_lt_flag[i]	u(1)
}	
}	
}	
if(sample_adaptive_offset_enabled_flag) {	
slice_sao_interleaving_flag	u(1)
slice_sample_adaptive_offset_flag	u(1)
if(slice_sao_interleaving_flag && slice_sample_adaptive_offset_flag) {	
sao_cb_enable_flag	u(1)
sao_cr_enable_flag	u(1)
}	
}	

[0324]

if(scaling_list_enable_flag || deblocking_filter_in_aps_enabled_flag || (sample_adaptive_offset_enabled_flag && !slice_sao_interleaving_flag) || adaptive_loop_filter_enabled_flag)	
aps_id	ue(v)
if(slice_type == P || slice_type == B) {	
num_ref_idx_active_override_flag	u(1)
if(num_ref_idx_active_override_flag) {	
num_ref_idx_l0_active_minus1	ue(v)
if(slice_type == B)	
num_ref_idx_l1_active_minus1	ue(v)
}	
}	
if(lists_modification_present_flag) {	
ref_pic_list_modification( )	
ref_pic_list_combination( )	
}	
if(slice_type = = B)	
mvd_l1_zero_flag	u(1)
}	
if(cabac_init_present_flag && slice_type != I)	
cabac_init_flag	u(1)
if(!entropy_slice_flag) {	
slice_qp_delta	se(v)
if(deblocking_filter_control_present_flag) {	
if(deblocking_filter_in_aps_enabled_flag)	
inherit_dbl_params_from_aps_flag	u(1)
if(!inherit_dbl_params_from_aps_flag) {	
disable_deblocking_filter_flag	u(1)
if(!disable_deblocking_filter_flag) {	
beta_offset_div2	se(v)
tc_offset_div2	se(v)
}	
}	
}	
if (seq_vsp_flag && slice_type !=I)	
slice_vsp_enable_flag	u(1)
if(slice_type = = B)	
collocated_from_l0_flag	u(1)
if(slice_type != I && ((collocated_from_l0_flag && num_ref_idx_l0_active_minus1 > 0) || (!collocated_from_l0_flag && num_ref_idx_l1_active_minus1 > 0))	
collocated_ref_idx	ue(v)
if((weighted_pred_flag && slice_type = = P) || (weighted_bipred_idc == 1 && slice_type == B))	
pred_weight_table( )	
}	

[0325]

if(slice_type = = P | | slice_type = = B)	
five_minus_max_num_merge_cand	ue(v)
if(adaptive_loop_filter_enabled_flag) {	
slice_adaptive_loop_filter_flag	u(1)
if(slice_adaptive_loop_filter_flag && alf_coef_in_slice_flag)	
alf_param( )	
if(slice_adaptive_loop_filter_flag && !alf_coef_in_slice_flag)	
alf_cu_control_param( )	
}	
if(seq_loop_filter_across_slices_enabled_flag && (slice_adaptive_loop_filter_flag | | slice_sample_adaptive_offset_flag | | !disable_deblocking_filter_flag))	
slice_loop_filter_across_slices_enabled_flag	u(1)
if(tiles_or_entropy_coding_sync_idc > 0) {	
num_entry_point_offsets	ue(v)
if(num_entry_point_offsets > 0) {	
offset_len_minus1	ue(v)
for(i = 0; i < num_entry_point_offsets; i++)	
entry_point_offset[i]	u(v)
}	
}	
}	

[0326]

[0327] 표 29의 예의 신택스에서, 슬라이스의 슬라이스 헤더는, 만약 VSP 모드가 가능하게 되고 슬라이스가 I 슬라이스가 아님을 적용가능한 SPS의 seq_vsp_flag 신택스 엘리먼트가 나타내면, slice_vsp_enable_flag 신택스 엘리먼트를 포함한다. 슬라이스 헤더의 slice_vsp_enable_flag 신택스 엘리먼트는 VSP가 슬라이스에 대해 가능하게 되는지의 여부를 나타낼 수도 있다. 예를 들어, 1과 동일한 slice_vsp_enable_flag는 VSP가 현재 슬라이스에 대해 가능하게 됨을 나타낼 수도 있다. 이 예에서, 0과 동일한 slice_vsp_enable_flag는 VSP가 현재 슬라이스에 대해 가능하지 않게 됨을 나타낼 수도 있다. 더욱이, 이 예에서, slice_vsp_enable_flag가 존재하지 않는 경우, 비디오 디코더 (30) 는 slice_vsp_enable_flag를 0과 동일한 것으로 유추할 수도 있다.

[0328] HEVC 기반 3DV에 대한 제 1 구현예에서, CU들은 HEVC 기반 3DV에서 VSP를 지원하는 플래그들을 포함할 수도 있다. 아래의 표 30은 HEVC 기반 3DV에서의 CU에 대한 일 예의 신택스를 도시한다.

[0329] **표 30 - 코딩 유닛 신택스**

coding_unit(x0, y0, log2CbSize) {	기술어
CurrCbAddrTS = MinCbAddrZS[x0 >> Log2MinCbSize][y0 >> Log2MinCbSize]	
if(slice_type != I) {	
if (slice_vsp_enable_flag)	
vsp_cu_flag[x0 **][** y0 **]**	ae(v)
if(!vsp_cu_flag[x0][y0])	
skip_flag[x0 **][** y0 **]**	ae(v)
}	
if(!vsp_cu_flag[x0][y0] && skip_flag[x0][y0])	
prediction_unit(x0, y0 , log2CbSize)	
else if(!vsp_cu_flag[x0][y0] && (slice_type != I || log2CbSize = = Log2MinCbSize)){	
if(slice_type != I)	
pred_mode_flag	ae(v)
if(PredMode != MODE_INTRA || log2CbSize = = Log2MinCbSize)	
part_mode	ae(v)
x1 = x0 + ((1 << log2CbSize) >> 1)	
y1 = y0 + ((1 << log2CbSize) >> 1)	
x2 = x1 − ((1 << log2CbSize) >> 2)	
y2 = y1 − ((1 << log2CbSize) >> 2)	
x3 = x1 + ((1 << log2CbSize) >> 2)	
y3 = y1 + ((1 << log2CbSize) >> 2)	
if(PartMode = = PART_2Nx2N)	
prediction_unit(x0, y0 , log2CbSize)	
else if(PartMode = = PART_2NxN) {	
prediction_unit(x0, y0 , log2CbSize)	
prediction_unit(x0, y1 , log2CbSize)	
} else if(PartMode = = PART_Nx2N) {	
prediction_unit(x0, y0 , log2CbSize)	
prediction_unit(x1, y0 , log2CbSize)	
} else if(PartMode = = PART_2NxnU) {	
prediction_unit(x0, y0 , log2CbSize)	
prediction_unit(x0, y2 , log2CbSize)	
} else if(PartMode = = PART_2NxnD) {	
prediction_unit(x0, y0 , log2CbSize)	
prediction_unit(x0, y3 , log2CbSize)	
} else if(PartMode = = PART_nLx2N) {	
prediction_unit(x0, y0 , log2CbSize)	
prediction_unit(x2, y0 , log2CbSize)	
} else if(PartMode = = PART_nRx2N) {	
prediction_unit(x0, y0 , log2CbSize)	
prediction_unit(x3, y0 , log2CbSize)	
} else { /* PART_NxN */	
prediction_unit(x0, y0 , log2CbSize)	
prediction_unit(x1, y0 , log2CbSize)	
prediction_unit(x0, y1 , log2CbSize)	
prediction_unit(x1, y1 , log2CbSize)	
}	
}	

[0330]

if(!skip_flag[x0][y0]&&!pcm_flag)	
transform_tree(x0, y0, x0, y0, log2CbSize, log2CbSize, log2CbSize, 0, 0)	
}	
}	

[0331]

[0332] 표 30의 예에서, VSP가 현재 슬라이스에 대해 가능하게 된다는 것을 현재 슬라이스 (즉, CU를 포함하는 슬라이스) 의 슬라이스 헤더의 slice_vsp_enable_flag 신택스 엘리먼트가 나타내면, 그 CU는 vsp_cu_flag 신택스 엘리먼트를 포함한다. vsp_cu_flag[x0][y0] 신택스 엘리먼트는, 현재 화상의 좌표들 (x0, y0) 에서의 화소를 포함하는 CU가 VSP 화상으로부터 예측된다는 것을 나타낼 수도 있다. 다시 말하면, 어레이 인덱스들인 x0, y0는 현재 화상의 좌측상단 루마 샘플을 기준으로 현재 CU (즉, 고려되는 코딩 블록) 의 지역의 좌측상단 루마 샘플의 로케이션 (x0, y0) 를 특정할 수도 있다. 예를 들어, 현재 CU가 P 또는 B 슬라이스 내에 있는 경우, 1과 동일한 vsp_cu_flag[x0][y0]는, 현재 CU의 pred_mode_flag 및 part_mode 신택스 엘리먼트들이 존재하지 않는다는 것, 현재 CU가 어떠한 PU들도 포함하지 않는다는 것, 및 전체 현재 CU가 VSP 화상으로부터 예측된다는

것을 특정할 수도 있다. 이 예에서, 0과 동일한 vsp_cu_flag[x0][y0]는 전체 현재 CU가 VSP 화상으로부터 완전히 예측되지 않는다는 것을 특정할 수도 있다. 따라서, vsp_cu_flag[x0][y0]가 0과 동일하면, 현재 CU는 하나 이상의 PU들을 포함할 수도 있다. 더욱이, 이 예에서, vsp_cu_flag[x0][y0]가 존재하지 않으면, 비디오 디코더 (30) 는 vsp_cu_flag[x0][y0]를 0과 동일한 것으로 유추할 수도 있다.

[0333] 더욱이, HEVC 기반 3DV에 대한 제 1 구현예에서, PU들은 VSP를 지원하는 플래그들을 포함할 수도 있다. 아래의 표 31은 HEVC 기반 3DV에서의 CU에 대한 일 예의 신택스를 도시한다.

[0334] **표 31 - 예측 유닛 신택스**

prediction_unit(x0, y0, log2CbSize) {	기술어
if (!vsp_cu_flag)	
if(PartMode ! = PART_2Nx2N && !vsp_in_cu_level_only_flag)	
vsp_pu_flag[x0][y0]	ae(v)
}	
if (!vsp_pu_flag[x0][y0]) {	
if(skip_flag[x0][y0]) {	
if(MaxNumMergeCand > 1)	
merge_idx[x0 **][** y0 **]**	ae(v)
} else if(PredMode = = MODE_INTRA) {	
if(PartMode = = PART_2Nx2N && pcm_enabled_flag && log2CbSize >= Log2MinIPCMCUSize && log2CbSize <= Log2MaxIPCMCUSize)	
pcm_flag	ae(v)
if(pcm_flag) {	
num_subsequent_pcm	tu(3)
NumPCMBlock = num_subsequent_pcm + 1	
while(!byte_aligned( ))	
pcm_alignment_zero_bit	u(v)
pcm_sample(x0, y0, log2CbSize)	
} else {	
prev_intra_luma_pred_flag[x0 **][** y0 **]**	ae(v)
if(prev_intra_luma_pred_flag[x0][y0])	
mpm_idx[x0 **][** y0 **]**	ae(v)
Else	
rem_intra_luma_pred_mode[x0 **][** y0 **]**	ae(v)
intra_chroma_pred_mode[x0][y0]	ae(v)
SignalledAsChromaDC = (chroma_pred_from_luma_enabled_flag ? intra_chroma_pred_mode[x0][y0] = = 3 : intra_chroma_pred_mode[x0][y0] = = 2)	
}	
} else { /* MODE_INTER */	
merge_flag[x0 **][** y0 **]**	ae(v)
if(merge_flag[x0][y0]) {	
if(MaxNumMergeCand > 1)	
merge_idx[x0 **][** y0 **]**	ae(v)
} else {	
if(slice_type = = B)	
inter_pred_flag[x0 **][** y0 **]**	ae(v)
if(inter_pred_flag[x0][y0] = = Pred_LC) {	
if(num_ref_idx_lc_active_minus1 > 0)	
ref_idx_lc[x0 **][** y0 **]**	ae(v)
mvd_coding(mvd_lc[x0][y0][0], mvd_lc[x0][y0][1])	
mvp_lc_flag[x0][y0]	ae(v)

[0335]

} else { /* Pred_L0 or Pred_BI */	
if(num_ref_idx_l0_active_minus1 > 0)	
ref_idx_l0[x0][y0]	ae(v)
mvd_coding(mvd_l0[x0][y0][0], mvd_l0[x0][y0][1])	
mvp_l0_flag[x0][y0]	ae(v)
}	
if(inter_pred_flag[x0][y0] == Pred_BI) {	
if(num_ref_idx_l1_active_minus1 > 0)	
ref_idx_l1[x0 **][** y0 **]**	ae(v)
if(mvd_l1_zero_flag) {	
mvd_l1[x0][y0][0] = 0	
mvd_l1[x0][y0][1] = 0	
} else	
mvd_coding(mvd_l1[x0][y0][0], mvd_l1[x0][y0][1])	
mvp_l1_flag[x0][y0]	ae(v)
}	
}	
}	
}	
}	

[0336]

[0337] 표 31의 신택스에서, 전체 CU가 VSP로부터 예측되지 않는다는 것을 CU에 대한 vsp_cu_flag가 나타내면, CU가 2Nx2N 파티셔닝 모드에 따라 파티셔닝되지 않으면, 그리고 VSP 모드만이 CU 레벨에 적용된다는 것을 적용가능한 SPS의 vsp_in_cu_level_only_flag가 나타내지 않는다면, PU는 vsp_pu_flag[x0][y0] 신택스 엘리먼트를 포함할 수도 있다. vsp_pu_flag[x0][y0] 신택스 엘리먼트는 PU가 VSP 예측으로 코딩되는지의 여부를 나타낼 수도 있다. 다르게 말하면, vsp_pu_flag[x0][y0]는 PU가 VSP 화상으로부터 예측되는지의 여부를 나타낼 수도 있다. 예를 들어, 1과 동일한 vsp_pu_flag[x0][y0]는 현재 PU가 VSP 예측으로 코딩된다는 것과 추가의 PU 관련 엘리먼트들이 PU에 존재하지 않는다는 것을 특정할 수도 있다. 이 예에서, 0과 동일한 vsp_pu_flag[x0][y0]는 현재 PU가 VSP 예측로 코딩되지 않는다는 것을 특정한다. vsp_pu_flag[x0][y0]가 존재하지 않는 경우, 비디오 디코더 (30) 는 vsp_pu_flag[x0][y0]를 PU를 포함하는 CU의 vsp_cu_flag[x0][y0]와 동일하다고 유추할 수도 있다. 다른 예에서, 비트스트림은, vsp_cu_flag 및 vsp_in_cu_level_only_flag 양쪽 모두가 0과 동일하면, 하나의 CU 내의 모든 PU들이 1과 동일한 vsp_pu_flag를 가질 수 없도록 제약된다.

[0338] 위의 표 31의 예에서, vsp_cu_flag는 이 PU를 포함하는 CU의 vsp_cu_flag의 값일 수도 있다. 대안으로, 표 31에서, 조건 "if(vsp_cu_flag)"은 다음과 같이 CU의 좌표들 및 사이즈에 기초하여 vsp_cu_flag를 명시적으로 특정하도록 변화될 수 있다:

[0339] if(vsp_cu_flag[x0>>log2CbSize << log2CbSize][y0 >>log2CbSize << log2CbSize])

[0340] 더욱이, HEVC 기반 3DV에 대한 제 1 구현예에서, 비디오 코더는 CU 또는 PU가 VSP 모드를 사용하는 경우 CU 또는 PU에 대한 예측 블록을 생성하는 프로세스를 수행한다. 일부 예들에서, PU에 적용가능한 vsp_cu_flag 신택스 엘리먼트 또는 vsp_pu_flag 신택스 엘리먼트가 1과 동일하면, CU 또는 PU는 VSP 모드를 사용할 수도 있다. 이 프로세스에서, 현재 CU 또는 PU는 루마 성분에서의 좌표들 (x, y) 및 크로마 성분들에서의 (x', y') 을 갖는 좌측 상단 화소를 가질 수도 있다. 더욱이, 이 프로세스에서, 현재 CU 또는 PU의 사이즈는 NxM이라고 가정된다. 4:2:0 비디오 포맷의 경우의 N/2xM/2, 4:4:4 비디오 포맷의 경우의 NxM, 또는 4:2:2 비디오 포맷의 경우의 NxM/2의 사이즈를 갖는 좌표들 (x', y') 로부터 시작하여, 모션 보상 유닛 (164) 은 VSP 화상의 크로마 성분의 각각의 블록에서의 화소들의 값들을, 크로마 성분들의 동일한 좌표들 (x', y') 로 시작하는 현재 CU 또는 PU의 화소들에 설정할 수도 있다. 예를 들어, 모션 보상 유닛 (164) 은 각각의 Cb 또는 Cb 샘플이 VSP 화상의 대응하는 로케이션에서의 Cb 또는 Cr 샘플과 일치하도록 현재 CU 또는 PU에 대한 예측 Cb 또는 Cr 블록을 생성할 수도 있다. 이 예에서, 모션 보상 유닛 (164) 은 VSP 화상의 Cb 또는 Cr 블록의 좌표들 (x', y') 로부터 예측 Cb 또는 Cr 블록을 생성하는 것을 시작할 수도 있다. 더욱이, 이 예에서, 예측 Cb 또는 Cr 블록은,VSP 화상의 크로마 블록들이 4:2:0 비디오 포맷, 4:4:4 비디오 포맷, 또는 4:2:2 비디오 포맷 각각에 따라 다운샘플링되면, 사이즈 N/2xM/2, NxM, 또는 nxM/2로 될 수도 있다.

[0341] 더욱이, HEVC 기반 3DV에 대한 제 1 구현예에서, 비디오 코더는 vsp_cu_flag에 대한 콘텍스트를 유지할 수도 있다. vsp_cu_flag에 대한 콘텍스트는 vsp_pu_flag 및 vsp_cu_flag 양쪽 모두를 엔트로피 인코딩하기 위한 콘텍스트를 제공하는데 사용될 수도 있다. 비디오 코더가 vsp_pu_flags 및 vsp_cu_flags 양쪽 모두를 엔트로피 코딩하는 경우 비디오 코더는 vsp_cu_flag에 대한 콘텍스트를 업데이트할 수도 있다. vsp_cu_flag가 1과 동일한 경우, CU의 각각의 PU는 1과 동일한 VSPFlag를 가진다고 유추된다. 더욱이, vsp_cu_flag가 1과

동일한 경우, CU에서의 각각의 4x4 블록은 1과 동일한 VSPFlag를 가진다. vsp_cu_flag가 0과 동일한 경우, CU에서의 PU들은 vsp_pu_flag의 상이한 값들을 가질 수도 있다. 하나의 PU가 1과 동일한 vsp_pu_flag를 가지면, PU에서의 모든 4x4 블록들에 대한 VspFlags는 1로 설정된다.

[0342] HEVC 기반 3DV에 대한 제 1 구현예에서, 비디오 인코더는 VspFlag에 대한 콘텍스트를 초기화할 수도 있다. P 슬라이스들의 VspFlag의 콘텍스트에 대한 초기 확률은 B 슬라이스들의 그것과는 상이할 수도 있다. VspFlag는 P 및 B 슬라이스들 양쪽 모두에 적용될 수도 있다. 더욱이, HEVC 기반 3DV에 대한 제 1 구현예에서, 엔트로피 코더가 VspFlag를 엔트로피 코딩하고 있는 경우, 엔트로피 코더는 VspFlag에 연관된 빈에 대한 ctxIdxInc를 결정할 수도 있다. 일부 예들에서, 빈에 대한 ctxIdxInc는 condTermFlagA 및 condTermFlagB의 합일 수도 있다. 다시 말하면, ctxIdxInc는 ctxIdxInc = condTermFlagA + condTermFlagB로서 도출될 수도 있다. 다른 예에서, 빈에 대한 ctxIdxInc는 condTermFlagA과 동일할 수도 있다. 다시 말하면, ctxIdxInc는 ctxIdxInc = condTermFlagA로서 도출될 수도 있다. 다른 예에서, 빈에 대한 ctxIdxIdc는 condTermFlagB와 동일하다. 다시 말하면, ctxIdxInc는 ctxIdxInc = condTermFlagB로서 도출될 수도 있다. 다른 예에서, ctxIdxInc는 0과 동일하게 고정된다.

[0343] 위의 예들에서, 엔트로피 코더는, 현재 CU 또는 PU에 이웃하는 특정 블록들에 연관된 데이터가 현재 CU 또는 PU에 연관된 신택스 엘리먼트들에 대한 엔트로피 코딩 프로세스에서 사용할 수 있는지의 여부에 적어도 부분적으로 기초하여 condTermFlagA 및 condTermFlagB를 결정할 수도 있다. 예를 들어, uiPAddrA는 현재 CU 또는 PU 좌측의 이웃 4x4 블록을 나타낼 수도 있고 uiPAddrB는 현재 CU 또는 PU 상측의 이웃 4x4 블록을 나타낼 수도 있다. 일부 예들에서, uiPAddrN (여기서 N은 A 또는 B 중 어느 하나임) 이 이용가능하지 않거나 또는 블록 uiPAddrN에 대한 VspFlag가 0과 동일하면, condTermFlagN은 0과 동일하다. 그렇지 않고, 이 예에서, uiPAddrN이 이용가능하고 블록 uiPAddrN에 대한 VspFlag가 1과 동일하면, condTermFlagN은 1과 동일하다.

[0344] HEVC 기반 3DV에 대한 제 1 구현예에서, 필터 유닛 (이를테면 비디오 인코더 (20) 의 필터 유닛 (114) 또는 비디오 디코더 (30)) 의 필터 유닛 (160) 은 블록화해제 필터 프로세스를 수행할 수도 있다. 이 프로세스에서, 필터 유닛은 2 개의 이웃하는 화소들 (p0 및 q0) 사이의 경계에 대한 경계 강도 (bS) 를 계산한다. 이 예에서, 필터 유닛은 VSP 지역에 관련된 경계 강도를 결정할 수도 있다. VSP 지역은 VSP 모드로 각각 코딩되는 이웃 PU들 또는 CU들의 지역일 수도 있다.

[0345] VSP 지역에 관련된 경계 강도를 결정하기 위해, 필터 유닛은 p0 및 q0의 경계가 VSP 지역의 경계를 가로지르는지의 여부를 결정할 수도 있다. p0 및 q0의 경계가 VSP 지역의 경계를 가로지르지 않는다면, 필터 유닛은 그 경계 강도를 p0 및 q0의 경계가 인터 예측된 지역 내에 있는 경우를 위한 경계 강도와 동일한 경계 강도로 설정할 수도 있다. 인터 예측된 지역은, 동일한 참조 화상/화상 쌍들 및 동일한 또는 유사한 (즉, 특정한 정확도로 반올림된 후 동일한) 모션 벡터들로 모두 코딩되는 이웃 PU들 또는 CU들의 지역이다.

[0346] 한편, p0 및 q0의 경계가 VSP 지역 및 인터 예측된 지역 사이의 경계를 가로지르면, 필터 유닛은 경계 강도를, 경계가 2 개의 인터 예측된 지역들을 가로지르는 경우에 대한 경계 강도와 동일한 경계 강도로 설정할 수도 있다. p0 또는 q0 중 어느 하나가 인트라 코딩된 CU에 속하는 경우, 필터 유닛은 p0가 인트라 예측된 CU에 속하고 q0가 인터 예측된 지역에 속하는 경우와 동일한 경계 강도로 경계 강도를 설정할 수도 있다. p0가 인트라 코딩된 CU에 속하는 경우, q0는 VSP 지역에 속한다. 비슷하게, q0가 인트라 코딩된 CU에 속하는 경우, p0는 VSP 지역에 속한다. 인트라 예측된 지역은 특정한 인트라 예측 모드로 코딩된 CU일 수도 있다.

[0347] HEVC 기반 3DV에 대한 제 2 구현예는 HEVC 기반 3DV에 대한 제 1 구현예와 유사할 수도 있다. 그러나, HEVC 기반 3DV에 대한 제 2 구현예에서, VSP는 CU 레벨에만 적용되고, PU 레벨에는 적용되지 않는다. HEVC 기반 3DV에 대한 이 제 2 구현예에서, PU들에 대한 신택스 및 시맨틱스는 HEVC 규격 초안 6에 기재된 메인 프로파일에 대하여 수정되지 않는다. 더욱이, HEVC 기반 3DV에 대한 제 1 구현예에서와는 달리, vsp_cu_flag의 엔트로피 코딩은 vsp_cu_flag에만 의존할 수도 있는데, HEVC 기반 3DV에 대한 이 제 2 구현예에서, vsp_pu_flag가 없기 때문이다. HEVC 기반 3DV에 대한 이 제 2 구현예에서, 비디오 인코더 (20) 는 vsp_in_pu_level_flag를 시그널링하지 않고 비디오 디코더 (30) 는 vsp_in_pu_level_flag를 0과 동일한 것으로 항상 유추 (즉, 자동으로 결정) 할 수도 있다.

[0348] HEVC 기반 3DV에 대한 제 3 구현예는 HEVC 기반 3DV에 대한 제 1 구현예와 유사할 수도 있다. 그러나, HEVC 기반 3DV에 대한 제 3 구현예는 HEVC 기반 3DV에 대한 제 1 구현예와는 CU들에 대해 상이한 신택스 구조를 가질 수도 있다. HEVC 기반 3DV에 대한 제 3 구현예에서, CU에 대한 신택스 구조는 아래의 표 32에서 설명된다.

[0349] 표 32 - 코딩 유닛 신택스

coding_unit(x0, y0, log2CbSize) {	기술어
CurrCbAddrTS = MinCbAddrZS[x0 >> Log2MinCbSize][y0 >> Log2MinCbSize]	
if(slice_type != I) {	
skip_flag[x0 **][** y0 **]**	ae(v)
if (slice_vsp_enable_flag)	
vsp_cu_skip_flag[x0 **][** y0 **]**	ae(v)
}	
if(skip_flag[x0][y0] &&!vsp_cu_skip_flag[x0][y0])	
prediction_unit(x0, y0 , log2CbSize)	
else if(slice_type != I || log2CbSize == Log2MinCbSize){	
if (slice_vsp_enable_flag && slice_type != I)	
vsp_cu_flag[x0 **][** y0 **]**	ae(v)
if(!vsp_cu_flag[x0][y0]) {	
if(slice_type != I)	
pred_mode_flag	ae(v)
if(PredMode != MODE_INTRA || log2CbSize = = Log2MinCbSize)	
part_mode	ae(v)
x1 = x0 + ((1 << log2CbSize) >> 1)	
y1 = y0 + ((1 << log2CbSize) >> 1)	
x2 = x1 − ((1 << log2CbSize) >> 2)	
y2 = y1 − ((1 << log2CbSize) >> 2)	
x3 = x1 + ((1 << log2CbSize) >> 2)	
y3 = y1 + ((1 << log2CbSize) >> 2)	
if(PartMode = = PART_2Nx2N)	
prediction_unit(x0, y0 , log2CbSize)	
else if(PartMode = = PART_2NxN) {	
prediction_unit(x0, y0 , log2CbSize)	
prediction_unit(x0, y1 , log2CbSize)	
} else if(PartMode = = PART_Nx2N) {	
prediction_unit(x0, y0 , log2CbSize)	
prediction_unit(x1, y0 , log2CbSize)	
} else if(PartMode = = PART_2NxnU) {	
prediction_unit(x0, y0 , log2CbSize)	
prediction_unit(x0, y2 , log2CbSize)	
} else if(PartMode = = PART_2NxnD) {	
prediction_unit(x0, y0 , log2CbSize)	
prediction_unit(x0, y3 , log2CbSize)	
} else if(PartMode = = PART_nLx2N) {	
prediction_unit(x0, y0 , log2CbSize)	
prediction_unit(x2, y0 , log2CbSize)	
} else if(PartMode = = PART_nRx2N) {	
prediction_unit(x0, y0 , log2CbSize)	
prediction_unit(x3, y0 , log2CbSize)	
} else { /* PART_NxN */	
prediction_unit(x0, y0 , log2CbSize)	
prediction_unit(x1, y0 , log2CbSize)	
prediction_unit(x0, y1 , log2CbSize)	
prediction_unit(x1, y1 , log2CbSize)	

[0350]

}	
}	
if(pcm_flag)	
transform_tree(x0, y0, x0, y0, log2CbSize, log2CbSize, log2CbSize, 0, 0)	
}	
}	

[0351]

[0352] 표 32의 예의 신택스 구조는 skip_flag[x0][y0]를 포함한다. 1과 동일한 skip_flag[x0][y0]는, 현재 CU에 대해, P 또는 B 슬라이스를 디코딩하는 경우, 모션 벡터 예측변수 인덱스들을 제외하면 더 이상 신택스 엘리먼트들이 skip_flag[x0][y0] 뒤에서 디코딩되지 않을 것임을 특정한다. 0과 동일한 skip_flag[x0][y0]는 코딩 유닛이 스킵되지 않는다는 것을 특정한다. 어레이 인덱스들인 x0, y0는 화상의 좌측상단 루마 샘플을 기준으로 고려되는 코딩 블록의 좌측상단 루마 샘플의 로케이션 (x0, y0) 을 특정한다. skip_flag[x0][y0]가 존재하지 않는 경우, 비디오 디코더 (30) 는 skip_flag[x0][y0]를 0과 동일한 것으로 유추할 수도 있다. 1과 동일한 skip_flag[x0][y0]를 갖는 CU는 HEVC에서의 기존의 스킵 CU 또는 VSP로 코딩된 CU 중 어느 하나일

수도 있고, 잔차 시그널링되지 않고, 따라서 transform_tree 신택스 구조는 CU 내에 존재하지 않는다.

[0353] 본 개시물의 하나 이상의 기법들에 따라, CU는, slice_vsp_enable_flag가 1과 동일하면, vsp_cu_skip_flag[x0][y0] 신택스 엘리먼트를 포함할 수도 있다. vsp_cu_skip_flag[x0][y0] 신택스 엘리먼트는 현재 CU가 VSP 모드에 의해 예측되는지의 여부를 나타낸다. 예를 들어, 1과 동일한 vsp_cu_skip_flag[x0][y0]는 현재 CU가 스킵 모드를 사용하여 인코딩되고 VSP 화상으로부터 예측된다는 것을 특정한다. 이 예에서, 0과 동일한 vsp_cu_skip_flag[x0][y0]는 현재 CU가 기존의 HEVC 스킵 모드로 예측된다는 것을 특정한다. CU가 P 또는 B 슬라이스 내에 있고 CU가 기존의 HEVC 스킵 모드에서 인코딩되는 경우, 모션 벡터 예측변수를 제외하면 더 이상 신택스 엘리먼트들이 skip_flag[x0][y0] 뒤에서 디코딩되지 않는다. 그런고로, 본 개시물의 하나 이상의 기법들에 따라, skip_flag[x0][y0]가 1과 동일하고 vsp_cu_skip_flag[x0][y0]가 1과 동일하지 않으면, CU는 PU를 포함할 수도 있다.

[0354] 더욱이, skip_flag[x0][y0]가 0과 동일하지 않거나 또는 vsp_cu_skip_flag[x0][y0]가 1과 동일하고, 현재 슬라이스가 I 슬라이스가 아니고 slice_vsp_enable_flag가 1과 동일하면, CU는, 본 개시물의 하나 이상의 기법들에 따라, vsp_cu_flag[x0][y0] 신택스 엘리먼트를 포함할 수도 있다. vsp_cu_flag[x0][y0]는, P 또는 B 슬라이스를 디코딩하는 경우, 예측 유닛 신택스 표에서의 pred_mode_flag, part_mode, 또는 신택스 엘리먼트가 CU 내에 존재하는지와 전체 CU가 VSP 화상으로부터 예측되는지의 여부를 나타낼 수도 있다. 예를 들어, 1과 동일한 vsp_cu_flag[x0][y0]는 현재 CU에 대해, P 또는 B 슬라이스를 디코딩하는 경우, 예측 유닛 신택스 표에서의 pred_mode_flag, part_mode 또는 신택스 엘리먼트들이 존재하지 않고 전체 CU가 VSP 화상으로부터 예측된다는 것을 특정할 수도 있다. 이 예에서, 0과 동일한 vsp_cu_flag[x0][y0]는 전체 CU가 VSP 화상으로부터 완전히 예측되지 않는다는 것을 특정한다. 어레이 인덱스들인 x0, y0는 화상의 좌측상단 루마 샘플을 기준으로 고려되는 코딩 블록의 좌측상단 루마 샘플의 로케이션 (x0, y0) 를 특정한다. 존재하지 않는 경우, vsp_cu_flag[x0][y0]는 max(0, vsp_cu_skip_flag[x0][y0]) 와 동일한 것으로 유추된다.

[0355] HEVC 기반 3DV에 대한 제 3 구현예에서, 엔트로피 코딩 유닛, 이를테면 도 2의 엔트로피 인코딩 유닛 (118) 또는 도 3의 엔트로피 디코딩 유닛 (150) 은, vsp_cu_flag에 대한 엔트로피 코딩 콘텍스트를 유지할 수도 있다. 엔트로피 코딩 유닛은 vsp_pu_flag, vsp_cu_skip_flag 및 vsp_cu_flag의 코딩을 위해 vsp_cu_flag에 대한 엔트로피 코딩 콘텍스트를 사용할 수도 있다. 엔트로피 코딩 유닛은, 엔트로피 코딩 유닛이 vsp_cu_skip_flags, vsp_cu_flags, 및 vsp_cu_flags를 코딩하는 경우, vsp_cu_flag에 대한 엔트로피 코딩 콘텍스트를 업데이트할 수도 있다. vsp_cu_flag 또는 vsp_cu_skip_flag가 1과 동일하면, CU의 각각의 PU는 1과 동일한 VSPFlag를 가진다고 유추될 수도 있다. 덧붙여서, CU에서의 각각의 4x4 블록은 1과 동일한 VSPFlag를 가진다. 더욱이, vsp_cu_flag가 0인 경우, CU에서의 PU들은 vsp_pu_flag의 상이한 값들을 가질 수도 있다. 하나의 PU가 1과 동일한 vsp_pu_flag를 가지면, PU에서의 모든 4x4 블록들에 대한 VspFlag는 1로 설정된다.

[0356] HEVC 기반 3DV에 대한 제 3 구현예에서, VspFlag에 대한 콘텍스트 모델링은 콘텍스트 초기화 및 콘텍스트 선택을 포함한다. 콘텍스트 초기화에서, VSP 모드는 P 슬라이스들 및 B 슬라이스들 모두에 적용된다. P 슬라이스들의 VspFlag의 콘텍스트에 대한 초기 확률은 B 슬라이스들과는 상이할 수도 있다.

[0357] 콘텍스트 선택에서, uiPAddrA는 현재 CU 또는 PU 좌측의 이웃하는 4x4 블록을 나타내고 uiPAddrB는 현재 CU 또는 PU 상측의 이웃하는 4x4 블록을 나타낸다. 더욱이, 엔트로피 코딩 유닛은 변수 condTermFlagN (A 또는 B 중 어느 하나인 N을 가짐) 을 결정할 수도 있다. uiPAddrA가 이용가능하지 않거나 또는 블록 uiPAddrA에 대한 VspFlag가 0과 동일하면, 엔트로피 코딩 유닛은 condTermFlagN을 0과 동일하게 설정할 수도 있다. 그렇지 않고, uiPAddrN가 이용가능하고 블록 uiPAddrN에 대한 VspFlag가 1과 동일하면, 엔트로피 코딩 유닛은 condTermFlagN을 1과 동일하게 설정할 수도 있다.

[0358] 더욱이, HEVC 기반 3DV에 대한 제 3 구현예에서, 엔트로피 코딩 유닛은 콘텍스트 인덱스 ctxIdx를 사용하여 VspFlag를 코딩할 수도 있다. 엔트로피 코딩 유닛은 ctxIdx에 대한 증분 (즉, ctxIdxInc) 을 ctxIdxInc = condTermFlagA + condTermFlagB로서 결정할 수도 있다. 대안적 예에서, 엔트로피 코딩 유닛은 ctxIdxInc = condTermFlagA가 되도록 ctxIdxInc를 도출할 수도 있다. 다른 대안적 예에서, 엔트로피 코딩 유닛은 ctxIdxInc = condTermFlagB가 되도록 ctxIdxInc를 도출할 수도 있다. 다른 대안적 예에서, ctxIdxInc는 0으로 고정될 수도 있다.

[0359] HEVC 기반 3DV에 대한 제 4 구현예는 HEVC 기반 3DV에 대한 제 1 구현예와 유사할 수도 있다. 그러나, HEVC 기반 3DV에 대한 제 4 구현예에서, CU들에 대한 신택스는 아래의 표 33에 의해 정의될 수도 있다.

[0360] **표 33 - 코딩 유닛 신택스**

coding_unit(x0, y0, log2CbSize) {	기술어
CurrCbAddrTS = MinCbAddrZS[x0 >> Log2MinCbSize][y0 >> Log2MinCbSize]	
if(slice_type != I) {	
if (slice_vsp_enable_flag)	
vsp_cu_flag[x0 **][** y0 **]**	ae(v)
skip_flag[x0 **][** y0 **]**	ae(v)
}	
if(!vsp_cu_flag[x0][y0] && skip_flag[x0][y0])	
prediction_unit(x0, y0 , log2CbSize)	
else if(!vsp_cu_flag[x0][y0] && (slice_type != I || log2CbSize == Log2MinCbSize)){	
if(slice_type != I)	
pred_mode_flag	ae(v)
if(PredMode != MODE_INTRA || log2CbSize = = Log2MinCbSize)	
part_mode	ae(v)
x1 = x0 + ((1 << log2CbSize) >> 1)	
y1 = y0 + ((1 << log2CbSize) >> 1)	
x2 = x1 − ((1 << log2CbSize) >> 2)	
y2 = y1 − ((1 << log2CbSize) >> 2)	
x3 = x1 + ((1 << log2CbSize) >> 2)	
y3 = y1 + ((1 << log2CbSize) >> 2)	
if(PartMode = = PART_2Nx2N)	
prediction_unit(x0, y0 , log2CbSize)	
else if(PartMode = = PART_2NxN) {	
prediction_unit(x0, y0 , log2CbSize)	
prediction_unit(x0, y1 , log2CbSize)	
} else if(PartMode = = PART_Nx2N) {	
prediction_unit(x0, y0 , log2CbSize)	
prediction_unit(x1, y0 , log2CbSize)	
} else if(PartMode = = PART_2NxnU) {	
prediction_unit(x0, y0 , log2CbSize)	
prediction_unit(x0, y2 , log2CbSize)	
} else if(PartMode = = PART_2NxnD) {	
prediction_unit(x0, y0 , log2CbSize)	
prediction_unit(x0, y3 , log2CbSize)	
} else if(PartMode = = PART_nLx2N) {	
prediction_unit(x0, y0 , log2CbSize)	
prediction_unit(x2, y0 , log2CbSize)	
} else if(PartMode = = PART_nRx2N) {	
prediction_unit(x0, y0 , log2CbSize)	
prediction_unit(x3, y0 , log2CbSize)	
} else { /* PART_NxN */	
prediction_unit(x0, y0 , log2CbSize)	
prediction_unit(x1, y0 , log2CbSize)	
prediction_unit(x0, y1 , log2CbSize)	
prediction_unit(x1, y1 , log2CbSize)	
}	
}	
if(!skip_flag[x0][y0]&&!pcm_flag)	

[0361]

transform_tree(x0, y0, x0, y0, log2CbSize, log2CbSize, log2CbSize, 0, 0)	
}	
}	

[0362]

[0363] 표 33의 예의 신택스에서, CU는 vsp_cu_flag[x0][y0]를 포함한다. CU에 대한 vsp_cu_flag[x0][y0]는, P 또는 B 슬라이스를 디코딩하는 경우, 예측 유닛 신택스 표에서의 pred_mode_flag, part_mode, 또는 신택스 엘리먼트가 존재하지 않는 지와 전체 CU가 VSP 화상으로부터 예측되는지의 여부를 나타낼 수도 있다. 예를 들어, 1과 동일한 vsp_cu_flag[x0][y0]는 현재 CU에 대해, P 또는 B 슬라이스를 디코딩하는 경우, PU 신택스 표에서의 pred_mode_flag, part_mode 또는 신택스 엘리먼트들이 존재하지 않고 전체 CU가 VSP 화상으로부터 예측된다는 것을 특정할 수도 있다. 이 예에서, 0과 동일한 vsp_cu_flag[x0][y0]는 전체 CU가 VSP 화상으로부터 완전히 예측되지 않는다는 것을 특정한다. 어레이 인덱스들인 x0, y0는 화상의 좌측상단 루마 샘플을 기준으로 고려되는 코딩 블록의 좌측상단 루마 샘플의 로케이션 (x0, y0) 를 특정한다. vsp_cu_flag[x0][y0]가 존재하지 않는 경우, vsp_cu_flag[x0][y0]는 0과 동일한 것으로 유추될 수도 있다.

[0364] 덧붙여서, CU는, P 또는 B 슬라이스를 디코딩하는 경우, 현재 CU가, 모션 벡터 예측변수 인덱스들 (예컨대, merge_idx, mvp_lc_flag, mvp_l0_flag, mvp_l1_flag 등) 과는 다른 신택스 엘리먼트들을 포함하는지의 여부를 나타내는 skip_flag[x0][y0]를 포함할 수도 있다. 예를 들어, 1과 동일한 skip_flag[x0][y0]는, 현재 CU에 대해, P 또는 B 슬라이스를 디코딩하는 경우, 모션 벡터 예측변수 인덱스들을 제외하면 더 이상 신택스 엘리먼트들이 skip_flag[x0][y0] 뒤에서 디코딩되지 않을 것임을 특정한다. 이 예에서, 0과 동일한 skip_flag[x0][y0]는 CU가 스킵되지 않는다는 것을 특정한다. 어레이 인덱스들인 x0, y0는 화상의 좌측상단 루마 샘플을 기준으로 고려되는 코딩 블록의 좌측상단 루마 샘플의 로케이션 (x0, y0) 를 특정한다. skip_flag[x0][y0]가 존재하지 않는 경우, skip_flag[x0][y0]는 0과 동일한 것으로 유추될 수도 있다. 1과 동일한 skip_flag를 갖는 CU는 HEVC에서의 기존의 스킵 CU 또는 VSP로 코딩된 CU 중 어느 하나일 수도 있고, 잔차 시그널링되지 않고, 따라서 변환 트리 신택스 구조는 존재하지 않는다. 변환 트리 신택스 구조는 CU에 대한 잔차 정보를 포함할 수도 있다.

[0365] 도 4a는 본 개시물의 하나 이상의 기법들에 따른, 비디오 인코더 (20) 의 일 예의 동작 (200) 을 도시하는 흐름도이다. 도 4a의 흐름도 및 다음의 도면들의 흐름도들은 예들로서 제공된다. 다른 예들에서, 본 개시물의 기법들은 도 4a의 예 및 다음의 도면들에서 도시된 것들에 비해 많거나, 적거나, 또는 상이한 단계들을 사용하여 구현될 수도 있다.

[0366] 도 4a의 예에서, 비디오 인코더 (20) 는, 현재 액세스 유닛의 이전에 코딩된 텍스처 뷰 성분 및 현재 액세스 유닛의 깊이 뷰 성분에 적어도 부분적으로 기초하여, VSP 화상을 생성할 수도 있다 (201). 더욱이, 비디오 인코더 (20) 는, 다수의 텍스처 뷰들 및 깊이 뷰들의 인코딩된 표현을 포함하는 비트스트림으로, 현재 비디오 유닛이 VSP 화상으로부터 예측되는지의 여부를 나타내는 신택스 엘리먼트를 시그널링할 수도 있다 (202). 일부 예들에서, 현재 비디오 유닛은 현재 액세스 유닛의 현재 텍스처 뷰 성분의 MB 또는 MB 파티션일 수도 있다.

현재 비디오 유닛이 MB인 적어도 일부 예들에서, 비디오 인코더 (20) 는, 비트스트림으로, 신택스 엘리먼트를 포함하는 MB 계층 신택스 구조를 시그널링할 수도 있다. 그런 예들에서, 신택스 엘리먼트는 vsp_mb_flag 일 수도 있다. 현재 비디오 유닛이 MB 파티션인 적어도 일부 예들에서, 비디오 인코더 (20) 는, 비트스트림으로, 신택스 엘리먼트를 포함하는 신택스 구조를 시그널링할 수도 있다. 그런 예들에서, 신택스 구조는 MB 예측 신택스 구조 또는 서브 MB 예측 신택스 구조일 수도 있고 신택스 엘리먼트는 mb_part_vsp_flag 또는 sub_mb_vsp_flag일 수도 있다.

[0367] 더욱이, 비디오 인코더 (20) 는, 현재 비디오 유닛이 VSP 화상으로부터 예측되는지의 여부에 적어도 부분적으로 기초하여, 비트스트림으로 현재 비디오 유닛에 대한 모션 정보를 시그널링할지의 여부를 결정할 수도 있다 (204). 현재 비디오 유닛이 VSP 화상으로부터 예측되지 않는다는 결정에 응답하여 (204의 "아니오"), 비디오 인코더 (20) 는, 비트스트림으로, 현재 비디오 유닛에 대한 모션 정보를 시그널링할 수도 있다 (206). 그렇지 않으면 (204의 "예"), 비디오 인코더 (20) 는, 비트스트림으로부터, 현재 비디오 유닛에 대한 모션 정보를 생략할 수도 있다 (208). 어느 경우에나, 비디오 인코더 (20) 는 그 비트스트림을 출력할 수도 있다 (210). 현재 비디오 유닛에 대한 모션 정보는 참조 인덱스 및 모션 벡터 차이를 포함할 수도 있다. 참조 인덱스는 참조 화상 리스트 (예컨대 RefPicList0 또는 RefPicList1) 내의 포지션을 나타낼 수도 있고 모션 벡터 차이는 모션 벡터 예측변수 및 현재 비디오 유닛의 모션 벡터 사이의 차이를 나타낼 수도 있다.

[0368] 도 4b는 본 개시물의 하나 이상의 기법들에 따른, 비디오 디코더 (30) 의 일 예의 동작 (250) 을 도시하는 흐름도이다. 도 4b의 예에서, 비디오 디코더 (30) 는, 현재 액세스 유닛의 이전에 코딩된 텍스처 뷰 성분 및 현재 액세스 유닛의 깊이 뷰 성분에 적어도 부분적으로 기초하여, VSP 화상을 생성할 수도 있다 (251). 더욱이, 비디오 디코더 (30) 는, 다수의 텍스처 뷰들 및 다수의 깊이 뷰들의 코딩된 표현을 포함하는 비트스트림으로부터, 현재 비디오 유닛이 VSP 화상으로부터 예측되는지의 여부를 나타내는 신택스 엘리먼트를 디코딩할 수도 있다 (252). 일부 예들에서, 현재 비디오 유닛은 현재 액세스 유닛의 현재 뷰의 현재 텍스처 뷰 성분의 MB 또는 MB 파티션일 수도 있다. 현재 비디오 유닛이 MB인 적어도 일부 예들에서, 비디오 디코더 (30) 는, 비트스트림으로부터, 신택스 엘리먼트를 포함하는 MB 계층 신택스 구조를 디코딩할 수도 있다. 일부 이러한 예들에서, 신택스 엘리먼트는 vsp_mb_flag일 수도 있다. 현재 비디오 유닛이 MB 파티션인 적어도 일부 예들에서, 비디오 디코더는, 비트스트림으로부터, 신택스 엘리먼트를 포함하는 신택스 구조를 디코딩할 수도 있다.

그런 예들에서, 신택스 구조는 MB 예측 신택스 구조 또는 서브 MB 예측 신택스 구조일 수도 있고 신택스 엘리먼트는 mb_part_vsp_flag 또는 sub_mb_vsp_flag일 수도 있다.

[0369] 더욱이, 비디오 디코더 (30) 는 현재 비디오 유닛이 VSP 화상으로부터 예측되는지의 여부를 결정할 수도 있다 (254). 현재 비디오 유닛이 VSP 화상으로부터 예측되지 않는 경우 (254의 "아니오"), 비디오 디코더 (30)

는, 비트스트림으로부터, 현재 비디오 유닛에 대한 모션 정보를 디코딩할 수도 있다 (256). 덧붙여서, 비디오 디코더 (30) 는 현재 비디오 유닛에 대한 모션 정보를 사용하여 현재 비디오 유닛의 샘플 블록들을 복원할 수도 있다 (258). 현재 비디오 유닛이 VSP 화상으로부터 예측되는 경우 (254의 "예"), 비디오 디코더 (30) 는 VSP 화상을 사용하여 현재 비디오 유닛의 샘플 블록들을 복원할 수도 있다 (260).

[0370] 도 5a는 본 개시물의 기법들에 따른, 비디오 인코더 (20) 의 다른 예의 동작 (300) 을 도시하는 흐름도이다. 도 5a의 예에서 도시된 바와 같이, 비디오 인코더 (20) 는 깊이 맵 및 하나 이상의 텍스처 뷰 성분들에 적어도 부분적으로 기초하여 VSP 화상을 합성할 수도 있다 (302). 더욱이, 비디오 인코더 (20) 는 현재 비디오 유닛의 VSP 모드를 나타내는 신택스 엘리먼트를 생성할 수도 있다 (304). VSP 모드는 현재 비디오 유닛이 VSP 화상으로부터 예측되는지의 여부를 나타낼 수도 있다. 덧붙여서, 비디오 인코더 (20) 는 현재 비디오 유닛이 VSP 화상으로부터 예측되는지의 여부를 결정할 수도 있다 (306).

[0371] 현재 비디오 유닛이 VSP 화상으로부터 예측되지 않는 경우 (306의 "아니오"), 비디오 인코더 (20) 는, 이웃 비디오 유닛이 VSP 화상으로부터 예측되는지의 여부에 적어도 부분적으로 기초하여, 코딩 콘텍스트를 선택할 수도 있다 (308). 이웃 비디오 유닛의 샘플 블록은 현재 비디오 유닛의 샘플 블록과 이웃한다. 덧붙여서, 비디오 인코더 (20) 는 선택된 코딩 콘텍스트를 사용하여 현재 비디오 유닛의 적어도 일부 모션 정보를 엔트로피 인코딩할 수도 있다 (310).

[0372] 더욱이, 현재 비디오 유닛이 VSP 화상으로부터 예측되는지의 여부에 상관없이, 비디오 인코더 (20) 는, 현재 비디오 유닛의 VSP 모드에 적어도 부분적으로 기초하여, 경계 강도 값을 결정할 수도 있다 (312). 덧붙여서, 비디오 인코더 (20) 는, 경계 강도 값에 적어도 부분적으로 기초하여, 현재 비디오 유닛의 샘플 블록에 대해 블록화해제 동작을 수행할 수도 있다 (314). 비디오 인코더 (20) 는 현재 비디오 유닛의 샘플 블록을 디코딩된 화상 버퍼 (116) 내에 저장할 수도 있다 (316).

[0373] 도 5b는 본 개시물의 기법들에 따른, 비디오 디코더 (30) 의 다른 예의 동작 (350) 을 도시하는 흐름도이다. 도 5b의 예에서 도시된 바와 같이, 비디오 디코더 (30) 는 깊이 맵 및 하나 이상의 텍스처 뷰 성분들에 적어도 부분적으로 기초하여 VSP 화상을 합성할 수도 있다 (352). 더욱이, 비디오 디코더 (30) 는, 비트스트림으로부터, 현재 비디오 유닛의 VSP 모드를 나타내는 신택스 엘리먼트를 디코딩할 수도 있다 (354). VSP 모드는 현재 비디오 유닛이 VSP 화상으로부터 예측되는지의 여부를 나타낼 수도 있다. 덧붙여서, 비디오 디코더 (30) 는, 신택스 엘리먼트에 적어도 부분적으로 기초하여, 현재 비디오 유닛이 VSP 화상으로부터 예측되는지의 여부를 결정할 수도 있다 (356).

[0374] 현재 비디오 유닛이 VSP 화상으로부터 예측되지 않는 경우 (356의 "아니오"), 비디오 디코더 (30) 는, 이웃 비디오 유닛이 VSP 화상으로부터 예측되는지의 여부에 적어도 부분적으로 기초하여, 코딩 콘텍스트를 선택할 수도 있다 (358). 이웃 비디오 유닛의 샘플 블록은 현재 비디오 유닛의 샘플 블록과 이웃한다. 덧붙여서, 비디오 디코더 (30) 는 선택된 코딩 콘텍스트를 사용하여 현재 비디오 유닛의 적어도 일부 모션 정보를 엔트로피 디코딩할 수도 있다 (360).

[0375] 현재 비디오 유닛이 VSP 화상으로부터 예측되는 경우 (356의 "예"), 비디오 디코더 (30) 는, 현재 비디오 유닛의 VSP 모드에 적어도 부분적으로 기초하여, 경계 강도 값을 결정할 수도 있다 (362). 덧붙여서, 비디오 디코더 (30) 는, 경계 강도 값에 적어도 부분적으로 기초하여, 현재 비디오 유닛의 샘플 블록에 대해 블록화해제 동작을 수행할 수도 있다 (364). 비디오 디코더 (30) 는 현재 비디오 유닛의 샘플 블록을 디코딩된 화상 버퍼 (162) 내에 저장할 수도 있다 (366).

[0376] 도 6a는 본 개시물의 기법들에 따른, 비디오 인코더 (20) 의 다른 예의 동작 (400) 을 도시하는 흐름도이다. 도 6a의 예에서, 비디오 인코더 (20) 는, 비트스트림으로, 현재 MB를 포함하는 슬라이스에 대한 슬라이스 헤더 신택스 구조를 시그널링할 수도 있다 (402). 슬라이스 헤더 신택스 구조는 VSP가 슬라이스에 허용되는지의 여부를 나타내는 신택스 엘리먼트 (예컨대, slice_vsp_flag) 를 포함할 수도 있다.

[0377] 더욱이, 비디오 인코더 (20) 는 VSP가 슬라이스에 허용되는지의 여부 및 현재 MB가 스킵 모드를 사용하여 인코딩되는지의 여부를 결정할 수도 있다 (404). VSP가 슬라이스에 허용되고 현재 MB가 스킵 모드를 사용하여 인코딩되는 경우 (404의 "예"), 비디오 인코더 (20) 는, 그 슬라이스에 대한 슬라이스 데이터 신택스 구조 내에, 현재 MB가 VSP 화상으로부터 예측되는지의 여부 또는 현재 MB가 다른 참조 화상으로부터 예측되는지의 여부를 나타내는 신택스 엘리먼트 (예컨대, skip_from_vsp_flag) 를 포함시킬 수도 있다 (406). 그렇지 않으면, VSP가 슬라이스에 허용되지 않거나 또는 현재 MB가 스킵 모드를 사용하여 인코딩되지 않는 경우 (404의 "아

니오"), 비디오 인코더 (20) 는 슬라이스 데이터 신택스 구조로부터 신택스 엘리먼트 (예컨대, skip_from_vsp_flag) 를 생략할 수도 있다 (408). 어느 경우에나, 비디오 인코더 (20) 는, 비트스트림으로, 슬라이스 헤더 신택스 구조 및 슬라이스 데이터 신택스 구조를 시그널링할 수도 있다 (410).

이런 식으로, 비디오 인코더 (20), 비트스트림으로, MB를 포함하는 슬라이스에 대한 슬라이스 데이터 신택스 구조 신택스 엘리먼트를 시그널링하며, 여기서 슬라이스 데이터 신택스 구조는 MB가 스킵 모드를 사용하여 인코딩되는 경우 제 2 신택스 엘리먼트를 포함하며, 제 2 신택스 엘리먼트는 MB의 모두가 VSP 화상으로부터 예측되는지의 여부 또는 MB가 다른 참조 화상으로부터 예측되는지의 여부를 나타낸다.

[0378] 도 6b는 본 개시물의 기법들에 따른, 비디오 디코더 (30) 의 다른 예의 동작 (450) 을 도시하는 흐름도이다.

도 6b의 예에서, 비디오 디코더 (30) 는, 비트스트림으로부터, 현재 MB를 포함하는 슬라이스에 대한 슬라이스 헤더 신택스 구조를 디코딩할 수도 있다 (452). 슬라이스 헤더 신택스 구조는 VSP가 슬라이스에 허용되는지의 여부를 나타내는 신택스 엘리먼트 (예컨대, slice_vsp_flag) 를 포함할 수도 있다.

[0379] 더욱이, 비디오 디코더 (30) 는 VSP가 슬라이스에 허용되는지의 여부 및 현재 MB가 스킵 모드를 사용하여 인코딩되는지의 여부를 결정할 수도 있다 (454). VSP가 슬라이스에 허용되고 현재 MB가 스킵 모드를 사용하여 인코딩되는 경우 (454의 "예"), 비디오 디코더 (30) 는 현재 MB가 VSP 화상으로부터 예측되는지의 여부 또는 현재 MB가 다른 참조 화상으로부터 예측되는지의 여부를 나타내는 신택스 엘리먼트 (예컨대, skip_from_vsp_flag) 를 슬라이스에 대한 슬라이스 데이터 신택스 구조가 포함한다고 결정할 수도 있다 (456). 그렇지 않으면, VSP가 슬라이스에 허용되지 않거나 또는 현재 MB가 스킵 모드를 사용하여 인코딩되지 않는 경우 (454의 "아니오"), 비디오 디코더 (30) 는 신택스 엘리먼트 (예컨대, skip_from_vsp_flag) 가 슬라이스 데이터 신택스 구조로부터 생략된다고 결정할 수도 있다 (458). 어느 경우에나, 비디오 디코더 (30) 는 현재 MB의 샘플 블록들을 복원할 수도 있다 (460). 이런 식으로, 비디오 디코더 (30), 비트스트림으로부터, MB를 포함하는 슬라이스에 대한 슬라이스 데이터 신택스 구조 신택스 엘리먼트를 디코딩하며, 여기서 슬라이스 데이터 신택스 구조는 MB가 스킵 모드를 사용하여 인코딩되는 경우 제 2 신택스 엘리먼트를 포함하며, 제 2 신택스 엘리먼트는 MB의 모두가 VSP 화상으로부터 예측되는지의 여부 또는 MB가 다른 참조 화상으로부터 예측되는지의 여부를 나타낸다.

[0380] 도 7a는 본 개시물의 기법들에 따른, 비디오 인코더 (20) 의 다른 예의 동작 (500) 을 도시하는 흐름도이다.

도 7a의 예에서, 비디오 인코더 (20) 는, 상측 이웃 MB 및 좌측 이웃 MB가 현재 MB를 코딩함에 있어서 사용할 수 있고 상측 이웃 MB 및 좌측 이웃 MB가 VSP 화상으로부터 예측되고 VSP 화상이 존재하는지의 여부를 결정할 수도 있다 (502).

[0381] 상측 이웃 MB 및 좌측 이웃 MB가 현재 MB를 코딩함에 있어서 사용할 수 있으며, 상측 이웃 MB 및 좌측 이웃 MB가 VSP 화상으로부터 예측되고 VSP 화상이 존재하는 경우 (502의 "예"), 비디오 인코더 (20) 는, 슬라이스 데이터 신택스 구조 내에, 현재 MB가 VSP 화상으로부터 예측된다는 것과 현재 MB에 대한 잔차가 시그널링되지 않는다는 것을 나타내는 신택스 엘리먼트를 포함시킬 수도 있다 (504). 그렇지 않으면 (502의 "아니오"), 비디오 인코더 (20) 는 슬라이스 데이터 신택스 구조부터 신택스 엘리먼트를 생략할 시킬 수도 있다 (506). 어느 경우에나, 비디오 인코더 (20) 는, 비트스트림으로, 슬라이스 데이터 신택스 구조를 시그널링할 수도 있다 (508). 이런 식으로, 비디오 인코더 (20) 는, 비트스트림으로, 상측 이웃 MB 및 좌측 이웃 MB가 현재 비디오 유닛의 코딩에 사용가능하며, 상측 이웃 MB 및 좌측 이웃 MB가 VSP 화상으로부터 예측되고, VSP 화상이 존재하는 경우, 제 2 신택스 엘리먼트를 포함하는 슬라이스 데이터 신택스 구조를 시그널링할 수도 있으며, 제 2 신택스 엘리먼트는 MB가 VSP 화상으로부터 예측된다는 것과 MB에 대한 잔차가 시그널링되지 않는다는 것을 나타낸다.

[0382] 도 7b는 본 개시물의 기법들에 따른, 비디오 디코더 (30) 의 다른 예의 동작 (550) 을 도시하는 흐름도이다.

도 7b의 예에서, 비디오 디코더 (30) 는, 상측 이웃 MB 및 좌측 이웃 MB가 현재 MB를 코딩함에 있어서 사용할 수 있고 상측 이웃 MB 및 좌측 이웃 MB가 VSP 화상으로부터 예측되고 VSP 화상이 존재하는지의 여부를 결정할 수도 있다 (552).

[0383] 상측 이웃 MB 및 좌측 이웃 MB가 현재 MB를 코딩함에 있어서 사용할 수 있으며, 상측 이웃 MB 및 좌측 이웃 MB가 VSP 화상으로부터 예측되고 VSP 화상이 존재하는 경우 (552의 "예"), 비디오 디코더 (30) 는, 현재 MB가 VSP 화상으로부터 예측된다는 것과 현재 MB에 대한 잔차가 시그널링되지 않는다는 것을 나타내는 신택스 엘리먼트를 슬라이스 데이터 신택스 구조가 포함한다고 결정할 수도 있다 (554). 그렇지 않으면 (552의 "아니오"), 비디오 디코더 (30) 는 신택스 엘리먼트가 슬라이스 데이터 신택스 구조로부터 생략된다고 결정할 수도 있다

(556). 어느 경우에나, 비디오 디코더 (30) 는 현재 MB의 샘플 블록들을 복원할 수도 있다 (558). 이런 식으로, 비디오 디코더 (30) 는, 상측 이웃 MB 및 좌측 이웃 MB가 현재 비디오 유닛의 코딩에 사용가능하며, 상측 이웃 MB 및 좌측 이웃 MB가 VSP 화상으로부터 예측되고, VSP 화상이 존재하는 경우, 비트스트림으로부터 제 2 신택스 엘리먼트를 포함하는 슬라이스 데이터 신택스 구조를 디코딩할 수도 있으며, 제 2 신택스 엘리먼트는 MB가 VSP 화상으로부터 예측된다는 것과 MB에 대한 잔차가 시그널링되지 않는다는 것을 나타낸다.

[0384] 도 8a는 본 개시물의 기법들에 따른, 비디오 인코더 (20) 의 다른 예의 동작 (600) 을 도시하는 흐름도이다. 도 8a의 예에서, 비디오 인코더 (20) 는, 상측 이웃 MB 및 좌측 이웃 MB가 현재 MB를 코딩함에 있어서 사용할 수 있으며, 상측 이웃 MB 및 좌측 이웃 MB가 VSP 화상으로부터 예측되고, 현재 MB가 VSP 화상으로부터 예측되지 않는지의 여부를 결정할 수도 있다 (602). 상측 이웃 MB 및 좌측 이웃 MB가 현재 MB를 코딩함에 있어서 사용할 수 있으며, 상측 이웃 MB 및 좌측 이웃 MB가 VSP 화상으로부터 예측되고, 현재 MB가 VSP 화상으로부터 예측되지 않는 경우 (602의 "예"), 비디오 인코더 (20) 는, 현재 MB에 대한 MB 계층 신택스 구조 내에, 현재 MB의 유형을 나타내는 신택스 엘리먼트를 포함시킬 수도 있다 (604). 그렇지 않으면 (602의 "아니오"), 비디오 인코더 (20) 는 MB 계층 신택스 구조로부터 신택스 엘리먼트를 생략할 수도 있다 (606).

[0385] 더욱이, 비디오 인코더 (20) 는 현재 MB가 VSP 화상으로부터 예측되는지의 여부를 결정할 수도 있다 (608). 현재 MB가 VSP 화상으로부터 예측되지 않는 경우 (608의 "아니오"), 비디오 인코더 (20) 는, MB 계층 신택스 구조 내에, 서브 MB 예측 신택스 구조 또는 MB 예측 신택스 구조를 포함시킬 수도 있다 (610). 그렇지 않으면 (608의 "예"), 비디오 인코더 (20) 는 임의의 서브 MB 예측 신택스 구조들 및 임의의 MB 예측 신택스 구조들을 MB 계층 신택스 구조로부터 생략할 수도 있다 (612). 어느 경우에나, 비디오 인코더 (20) 는, 비트스트림으로, MB 계층 신택스 구조를 시그널링할 수도 있다 (614).

[0386] 이런 식으로, 비디오 인코더 (20) 는, 비트스트림으로, MB에 대한 MB 계층 신택스 구조를 시그널링할 수도 있다. MB 계층 신택스 구조는 상측 이웃 MB 및 좌측 이웃 MB가 현재 비디오 유닛의 코딩에 이용가능하며, 상측 이웃 MB 및 좌측 이웃 MB가 VSP 화상으로부터 예측되고, MB는 VSP 화상으로부터 예측되지 않는 경우 제 2 신택스 엘리먼트를 포함할 수도 있으며, 제 2 신택스 엘리먼트는 MB의 유형을 나타낸다. MB 계층 신택스 구조는, MB가 VSP 화상으로부터 예측되지 않는 경우 서브 MB 예측 신택스 구조 또는 MB 예측 신택스 구조를 포함할 수도 있다.

[0387] 도 8b는 본 개시물의 기법들에 따른, 비디오 디코더 (30) 의 다른 예의 동작 (650) 을 도시하는 흐름도이다. 도 8b의 예에서, 비디오 디코더 (30) 는, 상측 이웃 MB 및 좌측 이웃 MB가 현재 MB를 코딩함에 있어서 사용할 수 있으며, 상측 이웃 MB 및 좌측 이웃 MB가 VSP 화상으로부터 예측되고, 현재 MB가 VSP 화상으로부터 예측되지 않는지의 여부를 결정할 수도 있다 (652). 상측 이웃 MB 및 좌측 이웃 MB가 현재 MB를 코딩함에 있어서 사용할 수 있으며, 상측 이웃 MB 및 좌측 이웃 MB가 VSP 화상으로부터 예측되고, 현재 MB가 VSP 화상으로부터 예측되지 않는 경우 (652의 "예"), 비디오 디코더 (30) 는, 현재 MB에 대한 MB 계층 신택스 구조가 현재 MB의 유형을 나타내는 신택스 엘리먼트를 포함한다고 결정할 수도 있다 (654). 그렇지 않으면 (552의 "아니오"), 비디오 디코더 (30) 는 신택스 엘리먼트가 MB 계층 신택스 구조로부터 생략된다고 결정할 수도 있다 (656).

[0388] 더욱이, 비디오 디코더 (30) 는 현재 MB가 VSP 화상으로부터 예측되는지의 여부를 결정할 수도 있다 (658). 현재 MB가 VSP 화상으로부터 예측되지 않는 경우 (658의 "아니오"), 비디오 디코더 (30) 는 MB 계층 신택스 구조가 서브 MB 예측 신택스 구조 또는 MB 예측 신택스 구조를 포함한다고 결정할 수도 있다 (660). 그렇지 않으면 (658의 "예"), 비디오 디코더 (30) 는 임의의 서브 MB 예측 신택스 구조들 및 임의의 MB 예측 신택스 구조들이 MB 계층 신택스 구조로부터 생략된다고 결정할 수도 있다 (662). 어느 경우에나, 비디오 디코더 (30) 는 그 MB의 샘플 블록들을 복원할 수도 있다 (664).

[0389] 이런 식으로, 비디오 디코더 (30) 는, 비트스트림으로부터, 상측 이웃 MB 및 좌측 이웃 MB가 현재 비디오 유닛의 코딩에 이용가능하며, 상측 이웃 MB 및 좌측 이웃 MB가 VSP 화상으로부터 예측되고, MB는 VSP 화상으로부터 예측되지 않는 경우, 제 2 신택스 엘리먼트를 포함하는, MB에 대한 MB 계층 신택스 구조를 디코딩할 수도 있으며, 제 2 신택스 엘리먼트는 MB의 유형을 나타내고, MB 계층 신택스 구조는 MB가 VSP 화상으로부터 예측되지 않는 경우 서브 MB 예측 신택스 구조 또는 MB 예측 신택스 구조를 나타낸다.

[0390] 도 9a는 본 개시물의 기법들에 따른, 비디오 인코더 (20) 의 다른 예의 동작 (700) 을 도시하는 흐름도이다. 도 9a의 예에서, 비디오 인코더 (20) 는, 상측 이웃 MB 및 좌측 이웃 MB가 현재 서브 MB 파티션을 코딩함에 있어서 사용할 수 있으며, 상측 이웃 MB 및 좌측 이웃 MB가 VSP 화상으로부터 예측되고, 현재 서브 MB 파티션이 VSP 화상으로부터 예측되지 않는지의 여부를 결정할 수도 있다 (702).

[0391] 상측 이웃 MB 및 좌측 이웃 MB가 현재 서브 MB 파티션을 코딩함에 있어서 사용할 수 있으며, 상측 이웃 MB 및 좌측 이웃 MB가 VSP 화상으로부터 예측되고, 현재 서브 MB 파티션이 VSP 화상으로부터 예측되지 않는 경우 (702의 "예"), 비디오 인코더 (20) 는, 현재 서브 MB 파티션에 대한 서브 MB 예측 신택스 구조 내에, 서브 MB 파티션의 유형을 나타내는 신택스 엘리먼트를 포함시킬 수도 있다 (704). 그렇지 않으면 (702의 "아니오"), 비디오 인코더 (20) 는 서브 MB 예측 신택스 구조로부터 신택스 엘리먼트를 생략할 수도 있다 (706). 어느 경우에나, 비디오 인코더 (20) 는, 비트스트림으로, 서브 MB 예측 신택스 구조를 시그널링할 수도 있다 (708).

이런 식으로, 비디오 인코더 (20) 는, 비트스트림으로, 서브 MB 파티션에 대한 서브 MB 예측 신택스 구조를 시그널링할 수도 있으며, 상측 이웃 MB 및 좌측 이웃 MB가 현재 비디오 유닛의 코딩에 이용가능하며, 상측 이웃 MB 및 좌측 이웃 MB가 VSP 화상으로부터 예측되고, 서브 MB 파티션이 VSP 화상으로부터 예측되지 않는 경우, 서브 MB 예측 신택스 구조는 제 2 신택스 엘리먼트를 포함한다.

[0392] 도 9b는 본 개시물의 기법들에 따른, 비디오 디코더 (30) 의 다른 예의 동작 (750) 을 도시하는 흐름도이다.

도 9b의 예에서, 비디오 디코더 (30) 는, 상측 이웃 MB 및 좌측 이웃 MB가 현재 서브 MB 파티션을 코딩함에 있어서 사용할 수 있으며, 상측 이웃 MB 및 좌측 이웃 MB가 VSP 화상으로부터 예측되고, 현재 서브 MB 파티션이 VSP 화상으로부터 예측되지 않는지의 여부를 결정할 수도 있다 (752).

[0393] 상측 이웃 MB 및 좌측 이웃 MB가 현재 서브 MB 파티션을 코딩함에 있어서 사용할 수 있으며, 상측 이웃 MB 및 좌측 이웃 MB가 VSP 화상으로부터 예측되고, 현재 서브 MB 파티션이 VSP 화상으로부터 예측되지 않는 경우 (752의 "예"), 비디오 디코더 (30) 는, 현재 서브 MB 파티션에 대한 서브 MB 예측 신택스 구조가 서브 MB 파티션의 유형을 나타내는 신택스 엘리먼트를 포함한다고 결정할 수도 있다 (754). 그렇지 않으면 (752의 "아니오"), 비디오 디코더 (30) 는 신택스 엘리먼트가 서브 MB 예측 신택스 구조로부터 생략된다고 결정할 수도 있다 (756). 어느 경우에나, 비디오 디코더 (30) 는 서브 MB 파티션의 샘플 블록들을 복원할 수도 있다 (758).

이런 식으로, 비디오 디코더 (30) 는, 비트스트림으로부터, 상측 이웃 MB 및 좌측 이웃 MB가 현재 비디오 유닛의 코딩에 이용가능하며, 상측 이웃 MB 및 좌측 이웃 MB가 VSP 화상으로부터 예측되고, 서브 MB 파티션이 VSP 화상으로부터 예측되지 않는 경우, 서브 MB 파티션에 대한 서브 MB 예측 신택스 구조를 복원할 수도 있으며, 서브 MB 예측 신택스 구조는 제 2 신택스 엘리먼트를 포함하며, 제 2 신택스 엘리먼트는 서브 MB 파티션의 유형을 나타낸다.

[0394] 도 10은 일 예의 3DV 디코딩 순서를 도시하는 개념도이다. MVC 디코딩 순서는 비트스트림 순서일 수도 있다. 도 10의 예에서, 각각의 정사각형은 뷰 성분에 대응한다. 정사각형들의 열들은 액세스 유닛들에 대응한다. 각각의 액세스 유닛은 시간 인스턴스의 모든 뷰들의 코딩된 화상들을 포함하도록 정의될 수도 있다. 정사각형들의 행들은 뷰들에 대응한다. 도 10의 예에서, 액세스 유닛들은 T0,..., T7로 라벨 표시되고 뷰들은 S0,..., S7로 라벨 표시된다. 액세스 유닛의 각각의 뷰 성분이 다음 액세스 유닛의 임의의 뷰 성분 전에 디코딩되기 때문에, 도 10의 디코딩 순서는 시간 우선 코딩 (time-first coding) 이라고 지칭될 수도 있다. 도 10의 예에 도시된 바와 같이, 액세스 유닛들의 디코딩 순서는 뷰들의 출력 또는 디스플레이 순서와 동일하지 않을 수도 있다.

[0395] 도 11은 일 예의 시간적 및 뷰간 예측 구조를 도시하는 개념도이다. 도 11의 예에서, 각각의 정사각형은 뷰 성분에 대응한다. "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 11에서 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 뷰간 예측으로서 지칭될 수도 있다.

[0396] 하나 이상의 예들에서, 설명된 기능들은 하드웨어, 소프트웨어, 펌웨어, 또는 그것들의 임의의 조합으로 구현될 수도 있다. 소프트웨어로 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 또

는 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.

[0397] 비제한적인 예로, 그러한 컴퓨터 판독가능 저장 매체는 RAM, ROM, EEPROM, CD-ROM 또는 다른 광 디스크 스토리지, 자기 디스크 스토리지, 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 자원으로부터 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 적외선, 라디오, 및 마이크로파와 같은 무선 기술은 매체의 정의에 포함된다. 그러나, 컴퓨터 판독가능 저장 매체들 및 데이터 저장 매체들은 커넥션들, 반송파들, 신호들, 또는 다른 일시적인 매체들을 포함하지 않지만, 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, disk들은 보통 데이터를 자기적으로 재생하지만, disc들은 레이저들로써 광적으로 데이터를 재생한다. 상기한 것들의 조합들은 또한 컴퓨터 판독가능 매체들의 범위 내에 포함되어야 한다.

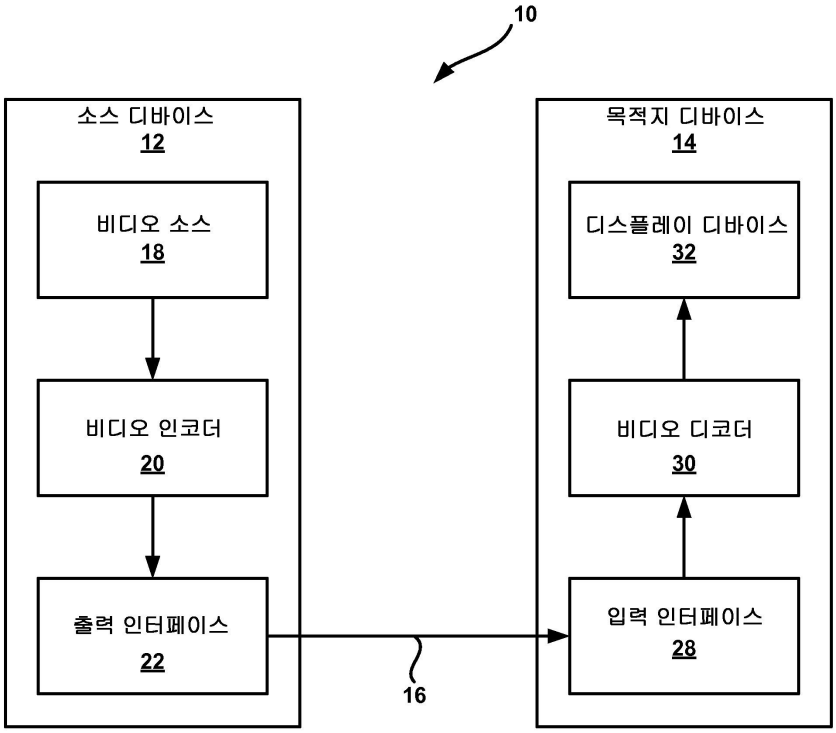
[0398] 명령들은 하나 이상의 프로세서들, 이를테면 하나 이상의 디지털 신호 프로세서들 (DSP들), 범용 마이크로프로세서들, 주문형 집적회로들 (ASIC들), 필드 프로그램가능 게이트 어레이들 (FPGA들), 또는 다른 동등한 집적 또는 개별 로직 회로에 의해 실행될 수도 있다. 따라서, 본원에서 사용되는 바와 같은 용어 "프로세서"는 앞서의 구조 또는 본원에서 설명된 기법들의 구현에 적합한 임의의 다른 구조 중 임의의 것을 말할 수도 있다. 덧붙여서, 일부 양태들에서, 본원에서 설명된 기능성은 인코딩 및 디코딩을 위해 구성되는, 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.

[0399] 본 개시물의 기법들은 무선 핸드셋, 집적회로 (IC) 또는 IC들의 세트 (예컨대, 칩 셋) 를 포함하는, 매우 다양한 디바이스들 또는 장치들로 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 요구하지는 않는다. 대신에, 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.

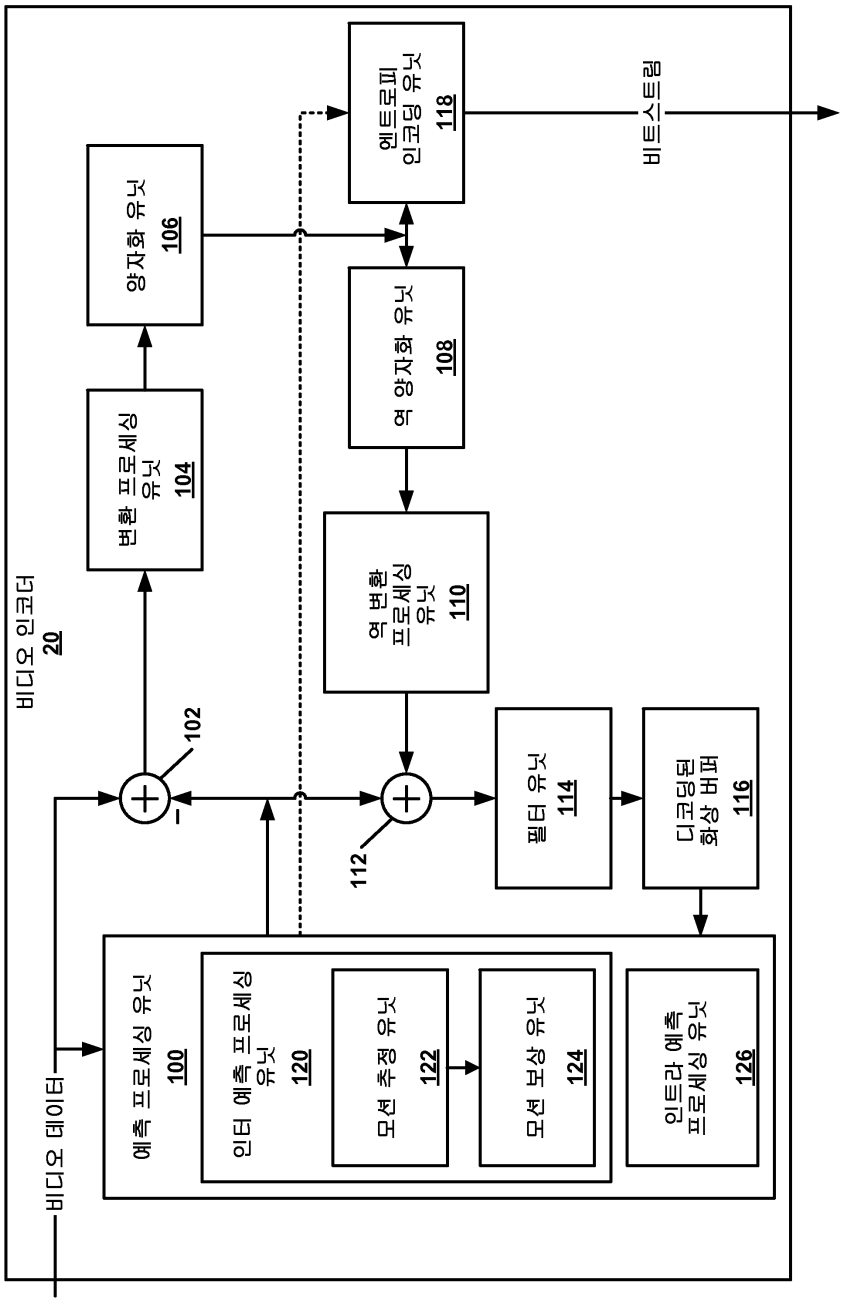
[0400] 다양한 예들이 설명되어 있다. 이들 및 다른 예들은 다음의 청구항들의 범위 내에 있다.

도면

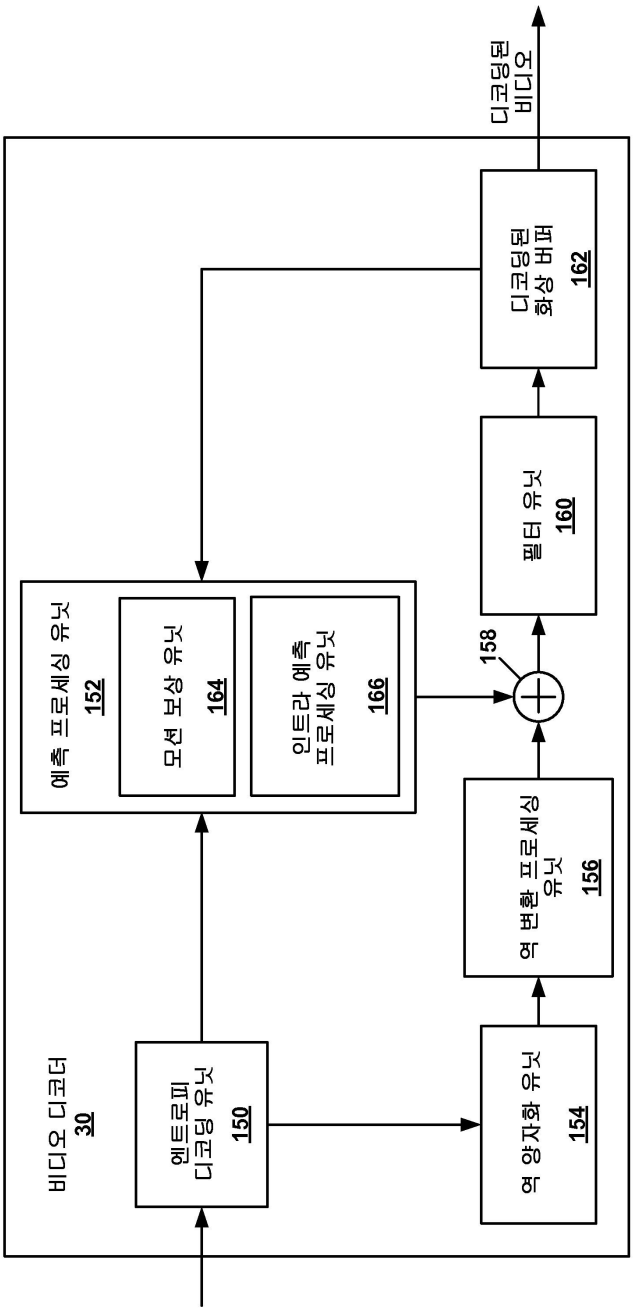
도면1


10
소스 디바이스
12
비디오 소스
18
비디오 인코더
20
출력 인터페이스
22
16
목적지 디바이스
14
디스플레이 디바이스
32
비디오 디코더
30
입력 인터페이스
28

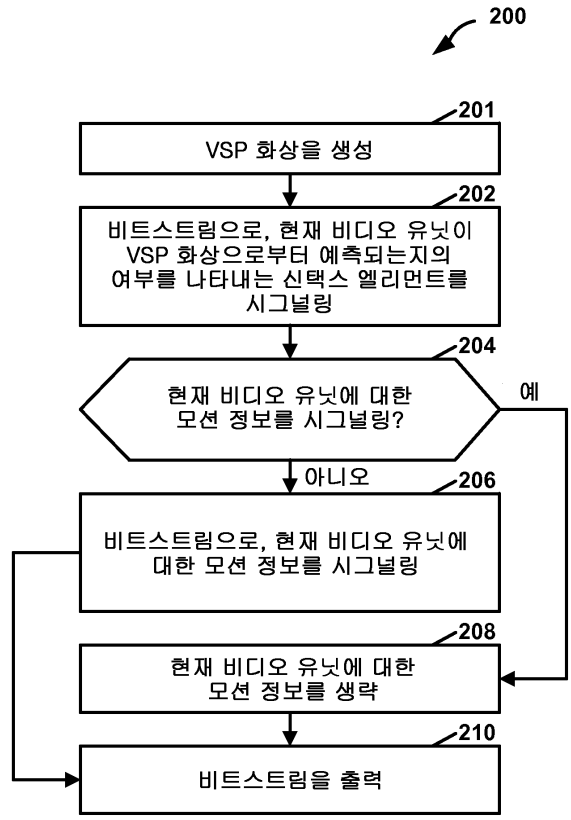
도면2


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
112
변환 프로세싱 유닛
104
양자화 유닛
106
역 양자화 유닛
108
역 변환 프로세싱 유닛
110
필터 유닛
114
디코딩된 화상 버퍼
116
엔트로피 인코딩 유닛
118
비트스트림

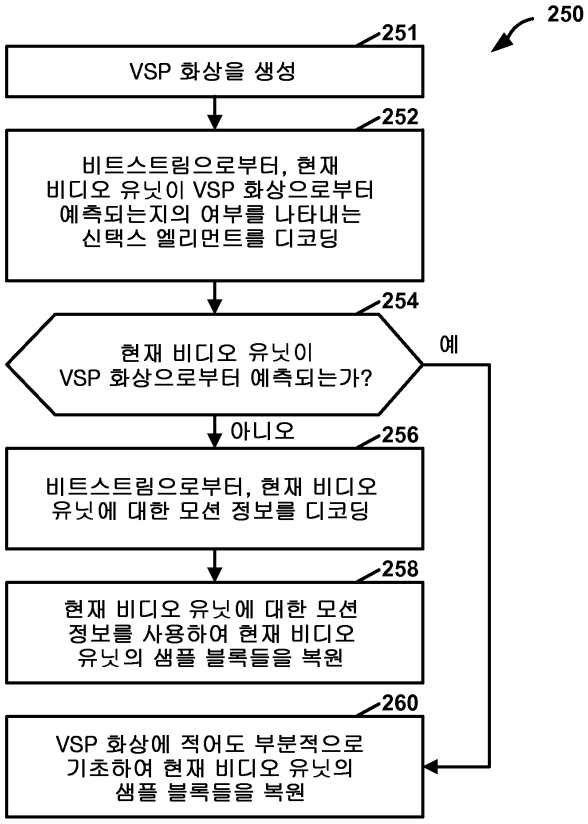
도면3


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

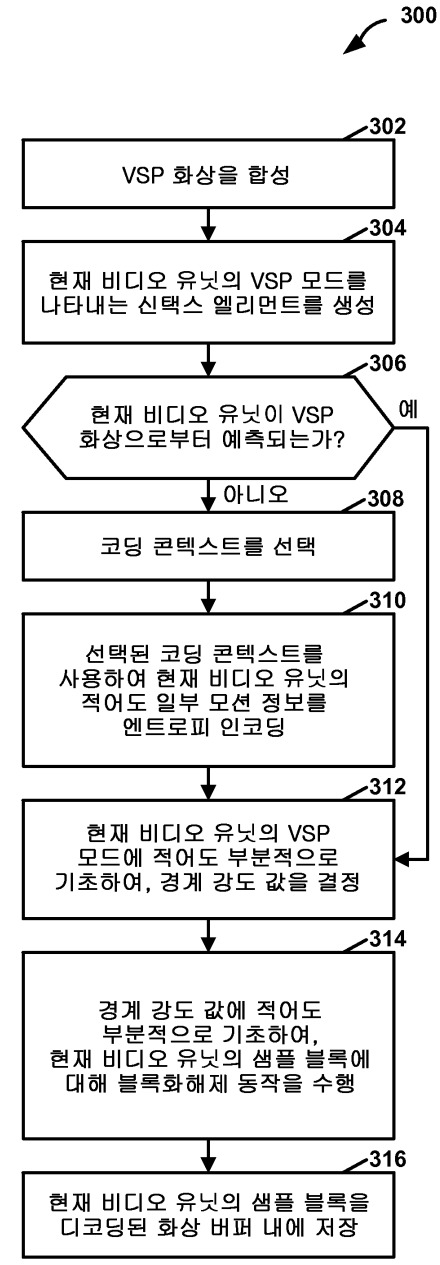
도면4a


200
201
VSP 화상을 생성
202
비트스트림으로, 현재 비디오 유닛이 VSP 화상으로부터 예측되는지의 여부를 나타내는 신택스 엘리먼트를 시그널링
204
현재 비디오 유닛에 대한 모션 정보를 시그널링?
예
아니오
206
비트스트림으로, 현재 비디오 유닛에 대한 모션 정보를 시그널링
208
현재 비디오 유닛에 대한 모션 정보를 생략
210
비트스트림을 출력

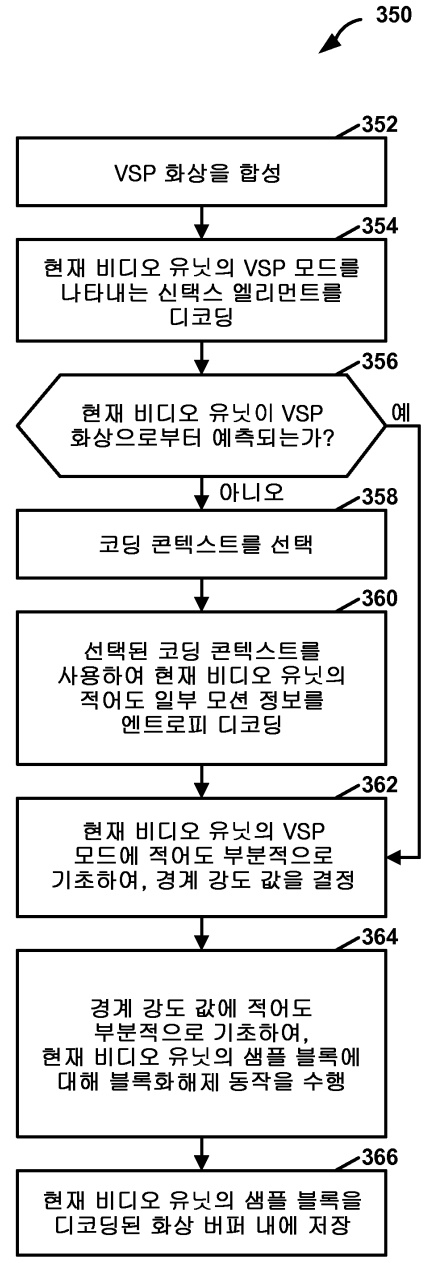
도면4b


250
251
VSP 화상을 생성
252
비트스트림으로부터, 현재 비디오 유닛이 VSP 화상으로부터 예측되는지의 여부를 나타내는 신택스 엘리먼트를 디코딩
254
현재 비디오 유닛이 VSP 화상으로부터 예측되는가?
예
아니오
256
비트스트림으로부터, 현재 비디오 유닛에 대한 모션 정보를 디코딩
258
현재 비디오 유닛에 대한 모션 정보를 사용하여 현재 비디오 유닛의 샘플 블록들을 복원
260
VSP 화상에 적어도 부분적으로 기초하여 현재 비디오 유닛의 샘플 블록들을 복원

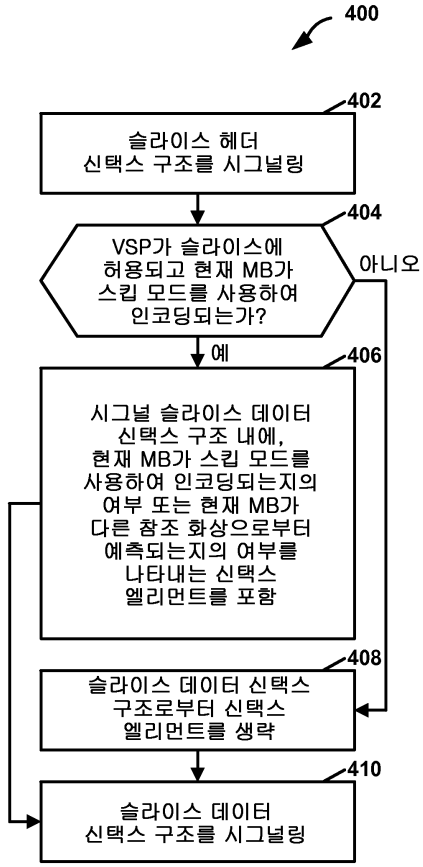
도면5a


300
302
VSP 화상을 합성
304
현재 비디오 유닛의 VSP 모드를 나타내는 신택스 엘리먼트를 생성
306
현재 비디오 유닛이 VSP 화상으로부터 예측되는가?
예
아니오
308
코딩 콘텍스트를 선택
310
선택된 코딩 콘텍스트를 사용하여 현재 비디오 유닛의 적어도 일부 모션 정보를 엔트로피 인코딩
312
현재 비디오 유닛의 VSP 모드에 적어도 부분적으로 기초하여, 경계 강도 값을 결정
314
경계 강도 값에 적어도 부분적으로 기초하여, 현재 비디오 유닛의 샘플 블록에 대해 블록화해제 동작을 수행
316
현재 비디오 유닛의 샘플 블록을 디코딩된 화상 버퍼 내에 저장

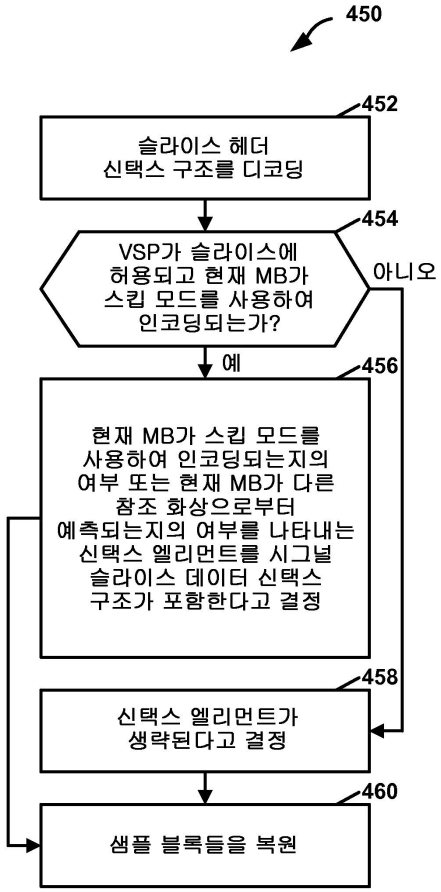
도면5b


350
352
VSP 화상을 합성
354
현재 비디오 유닛의 VSP 모드를 나타내는 신택스 엘리먼트를 디코딩
356
현재 비디오 유닛이 VSP 화상으로부터 예측되는가?
예
아니오
358
코딩 콘텍스트를 선택
360
선택된 코딩 콘텍스트를 사용하여 현재 비디오 유닛의 적어도 일부 모션 정보를 엔트로피 디코딩
362
현재 비디오 유닛의 VSP 모드에 적어도 부분적으로 기초하여, 경계 강도 값을 결정
364
경계 강도 값에 적어도 부분적으로 기초하여, 현재 비디오 유닛의 샘플 블록에 대해 블록화해제 동작을 수행
366
현재 비디오 유닛의 샘플 블록을 디코딩된 화상 버퍼 내에 저장

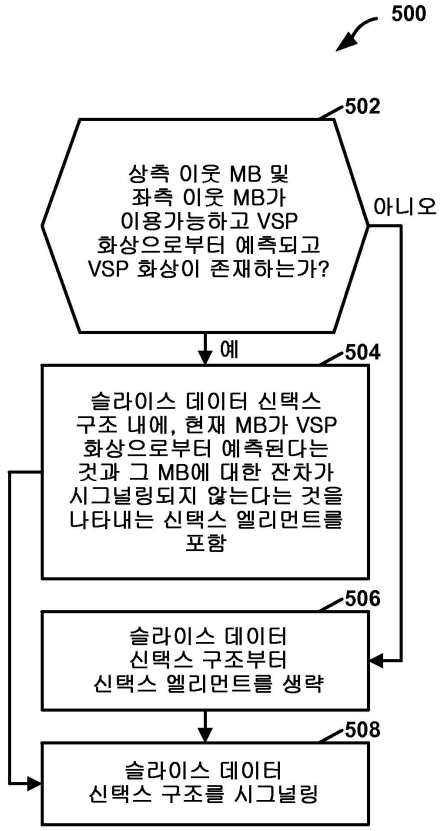
도면6a


400
402
슬라이스 헤더
신택스 구조를 시그널링
404
VSP가 슬라이스에
허용되고 현재 MB가
스킵 모드를 사용하여
인코딩되는가?
아니오
예
406
시그널 슬라이스 데이터
신택스 구조 내에,
현재 MB가 스킵 모드를
사용하여 인코딩되는지의
여부 또는 현재 MB가
다른 참조 화상으로부터
예측되는지의 여부를
나타내는 신택스
엘리먼트를 포함
408
슬라이스 데이터 신택스
구조로부터 신택스
엘리먼트를 생략
410
슬라이스 데이터
신택스 구조를 시그널링

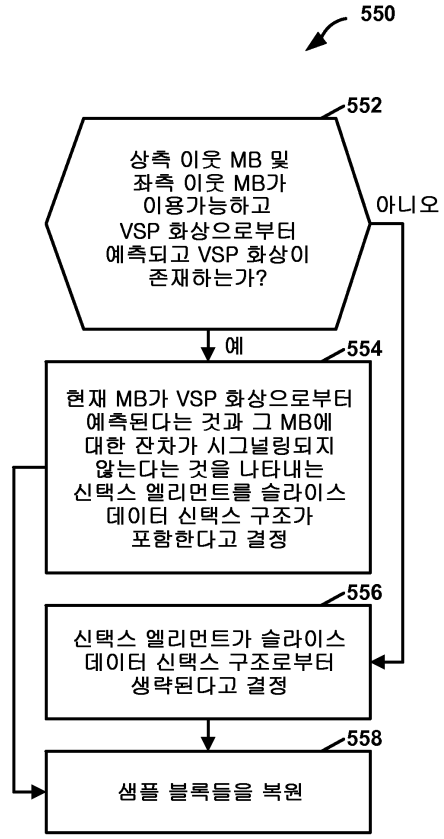
도면6b


450
452
슬라이스 헤더
신택스 구조를 디코딩
454
VSP가 슬라이스에
허용되고 현재 MB가
스킵 모드를 사용하여
인코딩되는가?
아니오
예
456
현재 MB가 스킵 모드를
사용하여 인코딩되는지의
여부 또는 현재 MB가 다른
참조 화상으로부터
예측되는지의 여부를 나타내는
신택스 엘리먼트를 시그널
슬라이스 데이터 신택스
구조가 포함한다고 결정
458
신택스 엘리먼트가
생략된다고 결정
460
샘플 블록들을 복원

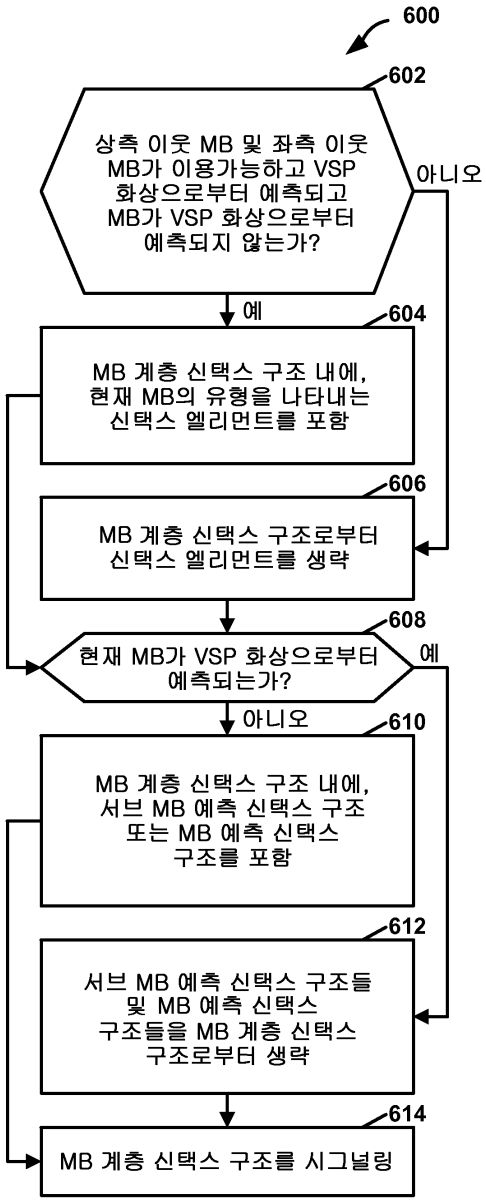
도면7a


500
502
상측 이웃 MB 및 좌측 이웃 MB가 이용가능하고 VSP 화상으로부터 예측되고 VSP 화상이 존재하는가?
아니오
예
504
슬라이스 데이터 신택스 구조 내에, 현재 MB가 VSP 화상으로부터 예측된다는 것과 그 MB에 대한 잔차가 시그널링되지 않는다는 것을 나타내는 신택스 엘리먼트를 포함
506
슬라이스 데이터 신택스 구조부터 신택스 엘리먼트를 생략
508
슬라이스 데이터 신택스 구조를 시그널링

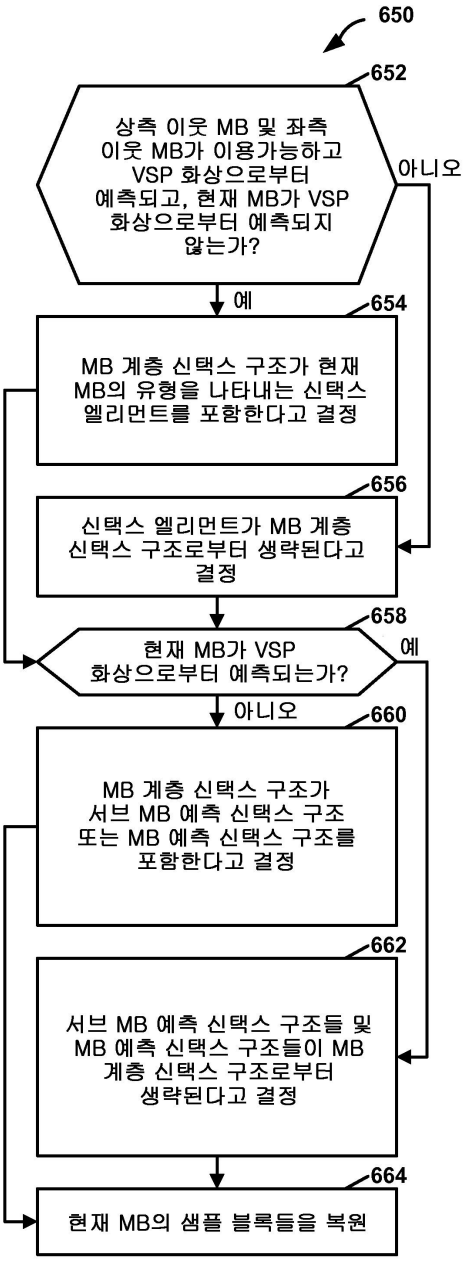
도면7b


550
552
상측 이웃 MB 및 좌측 이웃 MB가 이용가능하고 VSP 화상으로부터 예측되고 VSP 화상이 존재하는가?
아니오
예
554
현재 MB가 VSP 화상으로부터 예측된다는 것과 그 MB에 대한 잔차가 시그널링되지 않는다는 것을 나타내는 신택스 엘리먼트를 슬라이스 데이터 신택스 구조가 포함한다고 결정
556
신택스 엘리먼트가 슬라이스 데이터 신택스 구조로부터 생략된다고 결정
558
샘플 블록들을 복원

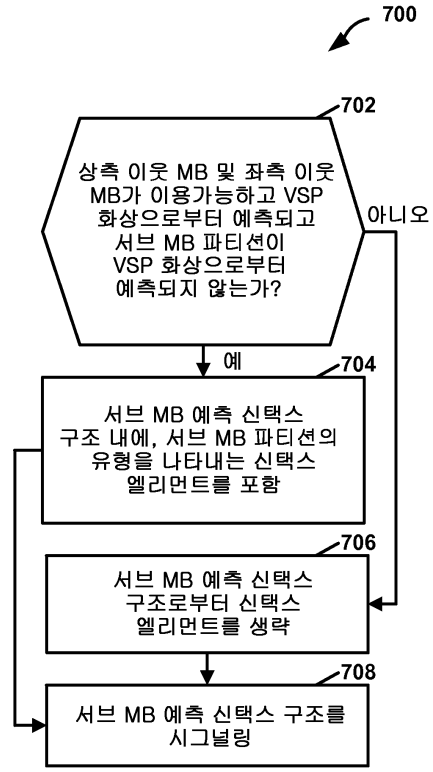
도면8a


600
602
상측 이웃 MB 및 좌측 이웃 MB가 이용가능하고 VSP 화상으로부터 예측되고 MB가 VSP 화상으로부터 예측되지 않는가?
아니오
예
604
MB 계층 신택스 구조 내에, 현재 MB의 유형을 나타내는 신택스 엘리먼트를 포함
606
MB 계층 신택스 구조로부터 신택스 엘리먼트를 생략
608
현재 MB가 VSP 화상으로부터 예측되는가?
예
아니오
610
MB 계층 신택스 구조 내에, 서브 MB 예측 신택스 구조 또는 MB 예측 신택스 구조를 포함
612
서브 MB 예측 신택스 구조들 및 MB 예측 신택스 구조들을 MB 계층 신택스 구조로부터 생략
614
MB 계층 신택스 구조를 시그널링

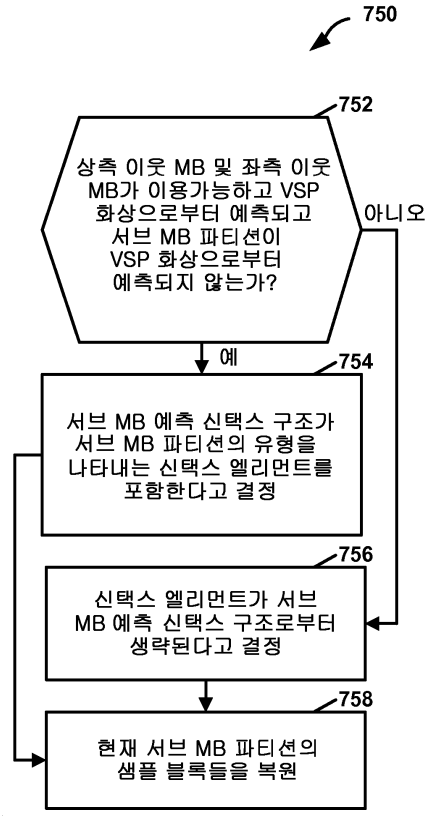
도면8b


650
652
상측 이웃 MB 및 좌측 이웃 MB가 이용가능하고 VSP 화상으로부터 예측되고, 현재 MB가 VSP 화상으로부터 예측되지 않는가?
아니오
예
654
MB 계층 신택스 구조가 현재 MB의 유형을 나타내는 신택스 엘리먼트를 포함한다고 결정
656
신택스 엘리먼트가 MB 계층 신택스 구조로부터 생략된다고 결정
658
현재 MB가 VSP 화상으로부터 예측되는가?
예
아니오
660
MB 계층 신택스 구조가 서브 MB 예측 신택스 구조 또는 MB 예측 신택스 구조를 포함한다고 결정
662
서브 MB 예측 신택스 구조들 및 MB 예측 신택스 구조들이 MB 계층 신택스 구조로부터 생략된다고 결정
664
현재 MB의 샘플 블록들을 복원

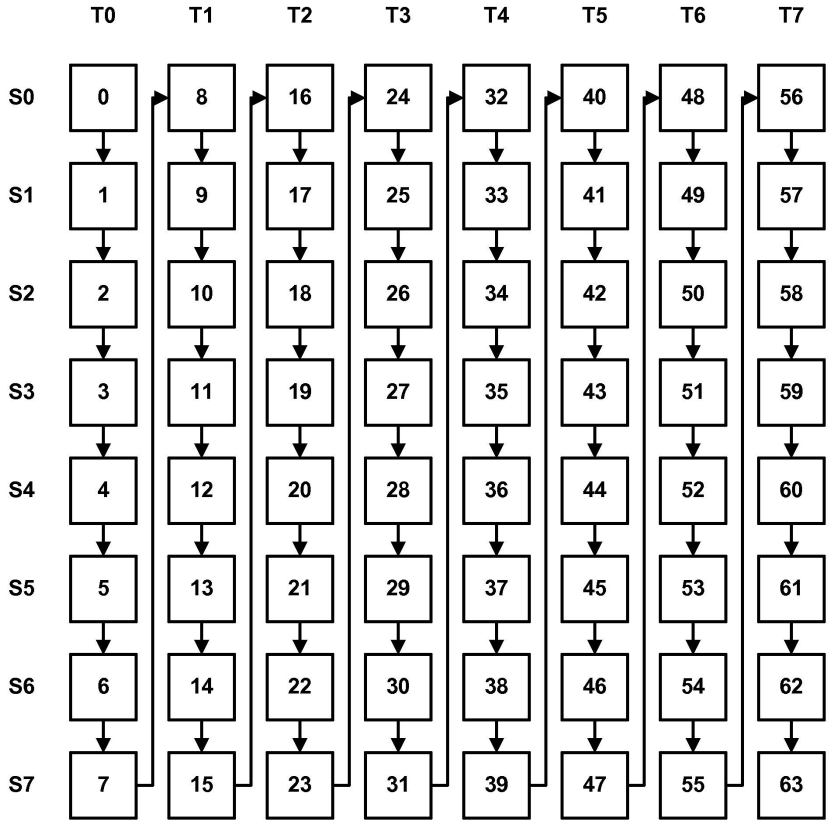
도면9a


700
702
상측 이웃 MB 및 좌측 이웃 MB가 이용가능하고 VSP 화상으로부터 예측되고 서브 MB 파티션이 VSP 화상으로부터 예측되지 않는가?
아니오
예
704
서브 MB 예측 신택스 구조 내에, 서브 MB 파티션의 유형을 나타내는 신택스 엘리먼트를 포함
706
서브 MB 예측 신택스 구조로부터 신택스 엘리먼트를 생략
708
서브 MB 예측 신택스 구조를 시그널링

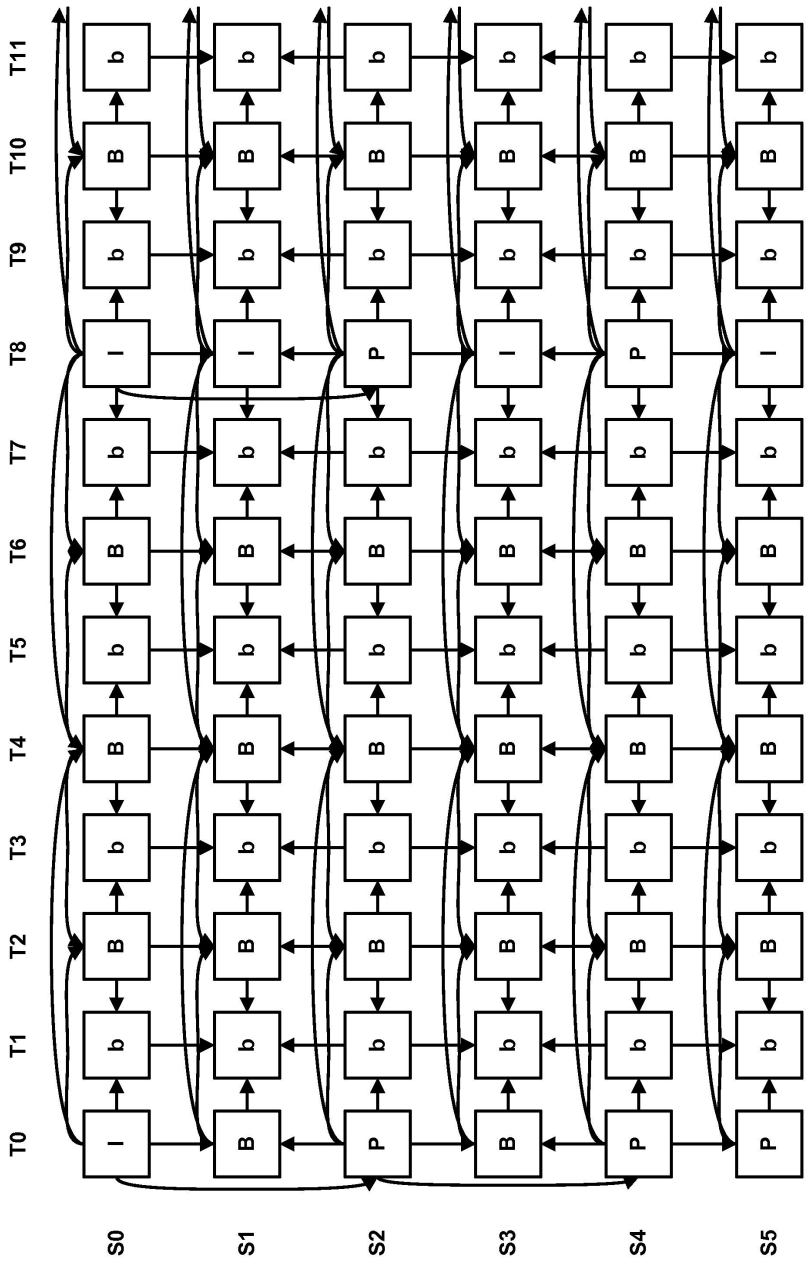
도면9b


750
752
상측 이웃 MB 및 좌측 이웃 MB가 이용가능하고 VSP 화상으로부터 예측되고 서브 MB 파티션이 VSP 화상으로부터 예측되지 않는가?
아니오
예
754
서브 MB 예측 신택스 구조가 서브 MB 파티션의 유형을 나타내는 신택스 엘리먼트를 포함한다고 결정
756
신택스 엘리먼트가 서브 MB 예측 신택스 구조로부터 생략된다고 결정
758
현재 서브 MB 파티션의 샘플 블록들을 복원

도면10


T0
T1
T2
T3
T4
T5
T6
T7
S0
S1
S2
S3
S4
S5
S6
S7
0
8
16
24
32
40
48
56
1
9
17
25
33
41
49
57
2
10
18
26
34
42
50
58
3
11
19
27
35
43
51
59
4
12
20
28
36
44
52
60
5
13
21
29
37
45
53
61
6
14
22
30
38
46
54
62
7
15
23
31
39
47
55
63

도면11



【심사관 직권보정사항】

【직권보정 1】

【보정항목】 청구범위

【보정세부항목】 청구항 11

【변경전】

3D 비디오 인코딩 디바이스 (20) 로서,

다수의 텍스처 뷰들 및 다수의 깊이 뷰들의 인코딩된 표현을 포함하는 비트스트림으로, 현재 비디오 유닛이 제 1 모드를 사용하여 코딩되는지의 여부를 시그널링하는 수단으로서, 상기 현재 비디오 유닛은 현재 액세스 유닛의 현재 뷰의 현재 텍스처 뷰 성분의 매크로블록, MB, 또는 MB 파티션인, 상기 현재 비디오 유닛이 제 1 모드를 사용하여 코딩되는지의 여부를 시그널링하는 수단;

상기 현재 비디오 유닛이 제 2 모드를 사용하여 코딩되고, 상기 제 1 모드를 사용하여 코딩되지 않는 경우, 상기 비트스트림으로, 상기 현재 비디오 유닛에 대한 모션 정보를 시그널링하는 수단;

상기 현재 비디오 유닛이 상기 제 1 모드를 사용하여 코딩되는 경우, 뷰 합성 예측을 위한 참조 뷰 화상을 결정하는 수단으로서, 상기 참조 뷰 화상은 뷰 합성을 위한 뷰인 것으로서 슬라이스 헤더에 나타낸 뷰에 있는 상기 현재 액세스 유닛에 있는 화상인, 상기 뷰 합성 예측을 위한 참조 뷰 화상을 결정하는 수단;

상기 현재 비디오 유닛이 상기 제 1 모드를 사용하여 코딩되는 경우, 상기 현재 비디오 유닛에 대한 참조 인덱스를, 상기 참조 인덱스가 상기 참조 뷰 화상을 나타내도록 설정하는 수단;

상기 현재 비디오 유닛이 상기 제 1 모드를 사용하여 코딩되는 경우, 상기 현재 비디오 유닛에 대한 모션 벡터를 도출하는 수단으로서, 상기 현재 비디오 유닛에 대한 모션 벡터는 상기 참조 뷰 화상을 참조하고, 상기 모션 벡터를 도출하는 것은:

깊이 화상의 블록으로부터 대표 깊이 값을 결정하는 것으로서, 상기 깊이 화상의 블록은 대응 텍스처 화상의 현재 비디오 유닛에 대응하고, 상기 대표 깊이 값은 상기 깊이 화상의 블록에 있는 미리 결정된 깊이 화소들의 최대값, 평균값 또는 중간값에 기초하여 결정되고, 상기 미리 결정된 깊이 화소들은 상기 깊이 화상의 블록에 있는 코너 깊이 화소들인, 상기 대표 깊이 값을 결정하는 것;

디스패리티 벡터를 결정하기 위해 상기 대표 깊이 값을 사용하는 것; 및

상기 현재 비디오 유닛에 대한 상기 모션 벡터의 값을 상기 디스패리티 벡터의 값으로 설정하는 것을 포함하는, 상기 현재 비디오 유닛에 대한 모션 벡터를 도출하는 수단; 및

상기 현재 비디오 유닛이 상기 단일화된 VSP 모드를 사용하여 코딩되는 경우, 상기 비트스트림으로부터, 상기 현재 비디오 유닛에 대한 모션 정보를 생략하는 수단; 및

상기 비트스트림을 출력하는 수단을 포함하는, 비디오 인코딩 디바이스.

【변경후】

3D 비디오 인코딩 디바이스 (20) 로서,

다수의 텍스처 뷰들 및 다수의 깊이 뷰들의 인코딩된 표현을 포함하는 비트스트림으로, 현재 비디오 유닛이 제 1 모드를 사용하여 코딩되는지의 여부를 시그널링하는 수단으로서, 상기 현재 비디오 유닛은 현재 액세스 유닛의 현재 뷰의 현재 텍스처 뷰 성분의 매크로블록, MB, 또는 MB 파티션인, 상기 현재 비디오 유닛이 제 1 모드를 사용하여 코딩되는지의 여부를 시그널링하는 수단;

상기 현재 비디오 유닛이 제 2 모드를 사용하여 코딩되고, 상기 제 1 모드를 사용하여 코딩되지 않는 경우, 상기 비트스트림으로, 상기 현재 비디오 유닛에 대한 모션 정보를 시그널링하는 수단;

상기 현재 비디오 유닛이 상기 제 1 모드를 사용하여 코딩되는 경우, 뷰 합성 예측을 위한 참조 뷰 화상을 결정하는 수단으로서, 상기 참조 뷰 화상은 뷰 합성을 위한 뷰인 것으로서 슬라이스 헤더에 나타낸 뷰에 있는 상기 현재 액세스 유닛에 있는 화상인, 상기 뷰 합성 예측을 위한 참조 뷰 화상을 결정하는 수단;

상기 현재 비디오 유닛이 상기 제 1 모드를 사용하여 코딩되는 경우, 상기 현재 비디오 유닛에 대한 참조 인덱스를, 상기 참조 인덱스가 상기 참조 뷰 화상을 나타내도록 설정하는 수단;

상기 현재 비디오 유닛이 상기 제 1 모드를 사용하여 코딩되는 경우, 상기 현재 비디오 유닛에 대한 모션 벡터를 도출하는 수단으로서, 상기 현재 비디오 유닛에 대한 모션 벡터는 상기 참조 뷰 화상을 참조하고, 상기 모션 벡터를 도출하는 것은:

깊이 화상의 블록으로부터 대표 깊이 값을 결정하는 것으로서, 상기 깊이 화상의 블록은 대응 텍스처 화상의 현재 비디오 유닛에 대응하고, 상기 대표 깊이 값은 상기 깊이 화상의 블록에 있는 미리 결정된 깊이 화소들의 최대값, 평균값 또는 중간값에 기초하여 결정되고, 상기 미리 결정된 깊이 화소들은 상기 깊이 화상의 블록에 있는 코너 깊이 화소들인, 상기 대표 깊이 값을 결정하는 것;

디스패리티 벡터를 결정하기 위해 상기 대표 깊이 값을 사용하는 것; 및

상기 현재 비디오 유닛에 대한 상기 모션 벡터의 값을 상기 디스패리티 벡터의 값으로 설정하는 것을 포함하는, 상기 현재 비디오 유닛에 대한 모션 벡터를 도출하는 수단; 및

상기 현재 비디오 유닛이 단일화된 VSP 모드를 사용하여 코딩되는 경우, 상기 비트스트림으로부터, 상기 현재 비디오 유닛에 대한 모션 정보를 생략하는 수단; 및

상기 비트스트림을 출력하는 수단을 포함하는, 비디오 인코딩 디바이스.