(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2021년08월24일
(11) 등록번호 10-2292788
(24) 등록일자 2021년08월18일

(51) 국제특허분류(Int. Cl.)
H04N 19/122 (2014.01) *H04N 19/105* (2014.01)
H04N 19/126 (2014.01) *H04N 19/13* (2014.01)
H04N 19/159 (2014.01) *H04N 19/176* (2014.01)
H04N 19/18 (2014.01) *H04N 19/186* (2014.01)
H04N 19/42 (2014.01) *H04N 19/61* (2014.01)
H04N 19/96 (2014.01)
(52) CPC특허분류
H04N 19/122 (2015.01)
H04N 19/105 (2015.01)
(21) 출원번호 10-2019-7019362
(22) 출원일자(국제) 2018년01월05일
심사청구일자 2020년12월23일
(85) 번역문제출일자 2019년07월03일
(65) 공개번호 10-2019-0104032
(43) 공개일자 2019년09월05일
(86) 국제출원번호 PCT/US2018/012589
(87) 국제공개번호 WO 2018/129322
국제공개일자 2018년07월12일
(30) 우선권주장
62/443,569 2017년01월06일 미국(US)
15/862,203 2018년01월04일 미국(US)
(56) 선행기술조사문헌
F. Le Leannec et al., "Asymmetric Coding Units in QTBT", JVET-D0064, 2016 October 15.*
Xiang Li et al., "Multi-Type-Tree", JVET-D0117, 2016 October 15*
*는 심사관에 의하여 인용된 문헌

(73) 특허권자
퀄컴 인코포레이티드
미국 92121-1714 캘리포니아주 샌 디에고 모어하우스 드라이브 5775
(72) 발명자
리 샹
미국 92130 캘리포니아주 샌디에고 게일몬트 레인 10574
자오 신
미국 92127 캘리포니아주 샌디에고 올드 스톤필드 체이스 8526
(뒷면에 계속)
(74) 대리인
특허법인코리아나

전체 청구항 수 : 총 38 항 심사관 : 김영태

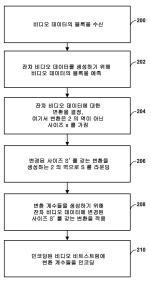
(54) 발명의 명칭 **비디오 코딩을 위한 다중-유형-트리 프레임워크**

(57) 요 약

비디오를 디코딩하는 방법은 비디오 데이터의 인코딩된 블록을 수신하는 단계; 비디오 데이터의 인코딩된 블록에 대한 변환을 결정하는 단계로서, 변환은 2 의 거듭제곱이 아닌 사이즈 (S) 를 갖는, 상기 변환을 결정하는 단계; S 를 2 의 거듭제곱으로 라운딩하여 변경된 사이즈 (S') 를 갖는 변환을 생성하는 단계; 잔차 비디오 데이터를

(뒷면에 계속)

대 표 도 - 도12



생성하기 위해 비디오 데이터의 인코딩된 블록에 변경된 사이즈 (S') 를 갖는 역변환을 적용하는 단계; 및 비디오 데이터의 디코딩된 블록을 생성하기 위해 잔차 비디오 데이터를 디코딩하는 단계를 포함한다.

(52) CPC특허분류
H04N 19/126 (2015.01)
H04N 19/13 (2015.01)
H04N 19/159 (2015.01)
H04N 19/176 (2015.01)
H04N 19/18 (2015.01)
H04N 19/186 (2015.01)
H04N 19/42 (2015.01)
H04N 19/61 (2015.01)
H04N 19/96 (2015.01)

(72) 발명자
장 리
미국 92121-1714 캘리포니아주 샌디에고 모어하우스 드라이브 5775
천 지안레
미국 92130 캘리포니아주 샌디에고 코르테 데 티뷰론 10756
추앙 샤오-창
미국 92121-1714 캘리포니아주 샌디에고 모어하우스 드라이브 5775
카르체비츠 마르타
미국 92121-1714 캘리포니아주 샌디에고 모어하우스 드라이브 5775

명 세 서

청구범위

청구항 1

비디오 데이터를 디코딩하는 방법으로서,

비디오 데이터의 인코딩된 블록을 수신하는 단계로서, 비디오 데이터의 상기 블록은 삼중-트리 파티션 구조로 파티셔닝되고, 상기 삼중-트리 파티션 구조는 상기 블록의 중심을 통해 상기 블록을 분할함이 없이 상기 블록을 3 개의 서브-블록들로 분할하는, 상기 비디오 데이터의 인코딩된 블록을 수신하는 단계;

비디오 데이터의 상기 인코딩된 블록에 대한 변환을 결정하는 단계로서, 상기 변환의 변환 행렬들은 2 의 거듭제곱이 아닌 사이즈 (S) 를 갖는, 상기 변환을 결정하는 단계;

S 를 2 의 거듭제곱으로 라운딩하여 변경된 사이즈 (S') 를 갖는 변환 행렬들을 갖는 역변환을 생성하는 단계;

잔차 비디오 데이터를 생성하기 위해 비디오 데이터의 상기 인코딩된 블록에 상기 변경된 사이즈 (S') 를 갖는 상기 변환 행렬들을 갖는 상기 역변환을 적용하는 단계; 및

비디오 데이터의 디코딩된 블록을 생성하기 위해 상기 잔차 비디오 데이터를 디코딩하는 단계를 포함하는, 비디오 데이터를 디코딩하는 방법.

청구항 2

제 1 항에 있어서,

S 를 2 의 거듭제곱으로 라운딩하는 것은 S 를 가장 가까운 2 의 거듭제곱으로 라운딩하는 것을 포함하는, 비디오 데이터를 디코딩하는 방법.

청구항 3

제 1 항에 있어서,

비디오 데이터의 상기 인코딩된 블록은 역양자화된 변환 계수들을 포함하는, 비디오 데이터를 디코딩하는 방법.

청구항 4

제 1 항에 있어서,

비디오 데이터의 상기 인코딩된 블록은 비정사각형 형상을 갖는, 비디오 데이터를 디코딩하는 방법.

청구항 5

제 1 항에 있어서,

S 는 12 이고, S 를 2 의 거듭제곱으로 라운딩하는 것은 12 를 16 으로 라운딩하는 것을 포함하고, 상기 변경된 사이즈 (S') 는 16 인, 비디오 데이터를 디코딩하는 방법.

청구항 6

제 1 항에 있어서,

S 는 24 이고, S 를 2 의 거듭제곱으로 라운딩하는 것은 24 를 32 로 라운딩하는 것을 포함하고, 상기 변경된 사이즈 (S') 는 32 인, 비디오 데이터를 디코딩하는 방법.

청구항 7

제 1 항에 있어서,

S 는 상기 변환의 너비인, 비디오 데이터를 디코딩하는 방법.

청구항 8

제 1 항에 있어서,

S 는 상기 변환의 높이인, 비디오 데이터를 디코딩하는 방법.

청구항 9

비디오 데이터를 인코딩하는 방법으로서,

비디오 데이터의 블록을 수신하는 단계로서, 비디오 데이터의 상기 블록은 삼중-트리 파티션 구조로 파티셔닝되고, 상기 삼중-트리 파티션 구조는 상기 블록의 중심을 통해 상기 블록을 분할함이 없이 상기 블록을 3 개의 서브-블록들로 분할하는, 상기 비디오 데이터의 블록을 수신하는 단계;

잔차 비디오 데이터를 생성하기 위해 비디오 데이터의 상기 블록을 예측하는 단계;

상기 잔차 비디오 데이터에 대한 변환을 결정하는 단계로서, 상기 변환의 변환 행렬들은 2 의 거듭제곱이 아닌 사이즈 (S) 를 갖는, 상기 변환을 결정하는 단계;

S 를 2 의 거듭제곱으로 라운딩하여 변경된 사이즈 (S') 를 갖는 변환 행렬들을 갖는 변환을 생성하는 단계;

변환 계수들을 생성하기 위해 상기 잔차 비디오 데이터에 상기 변경된 사이즈 (S') 를 갖는 상기 변환 행렬들을 갖는 상기 변환을 적용하는 단계; 및

인코딩된 비디오 비트스트림에 상기 변환 계수들을 인코딩하는 단계를 포함하는, 비디오 데이터를 인코딩하는 방법.

청구항 10

제 9 항에 있어서,

S 를 2 의 거듭제곱으로 라운딩하는 것은 S 를 가장 가까운 2 의 거듭제곱으로 라운딩하는 것을 포함하는, 비디오 데이터를 인코딩하는 방법.

청구항 11

제 9 항에 있어서,

상기 변환 계수들을 양자화하는 단계를 더 포함하는, 비디오 데이터를 인코딩하는 방법.

청구항 12

제 9 항에 있어서,

비디오 데이터의 상기 블록은 비정사각형 형상을 갖는, 비디오 데이터를 인코딩하는 방법.

청구항 13

제 9 항에 있어서,

S 는 12 이고, S 를 2 의 거듭제곱으로 라운딩하는 것은 12 를 16 으로 라운딩하는 것을 포함하고, 상기 변경된 사이즈 (S') 는 16 인, 비디오 데이터를 인코딩하는 방법.

청구항 14

제 9 항에 있어서,

S 는 24 이고, S 를 2 의 거듭제곱으로 라운딩하는 것은 24 를 32 로 라운딩하는 것을 포함하고, 상기 변경된 사이즈 (S') 는 32 인, 비디오 데이터를 인코딩하는 방법.

청구항 15

제 9 항에 있어서,

S 는 상기 변환의 너비인, 비디오 데이터를 인코딩하는 방법.

청구항 16

제 9 항에 있어서,

S 는 상기 변환의 높이인, 비디오 데이터를 인코딩하는 방법.

청구항 17

비디오 데이터를 디코딩하도록 구성된 장치로서,

상기 비디오 데이터를 저장하도록 구성된 메모리; 및

상기 메모리와 통신하는 하나 이상의 프로세서들을 포함하고,

상기 하나 이상의 프로세서들은,

상기 비디오 데이터의 인코딩된 블록을 수신하는 것으로서, 상기 비디오 데이터의 상기 블록은 삼중-트리 파티션 구조로 파티셔닝되고, 상기 삼중-트리 파티션 구조는 상기 블록의 중심을 통해 상기 블록을 분할함이 없이 상기 블록을 3 개의 서브-블록들로 분할하는, 상기 비디오 데이터의 인코딩된 블록을 수신하는 것을 행하고;

비디오 데이터의 상기 인코딩된 블록에 대한 변환을 결정하는 것으로서, 상기 변환의 변환 행렬들은 2 의 거듭제곱이 아닌 사이즈 (S) 를 갖는, 상기 변환을 결정하는 것을 행하며;

S 를 2 의 거듭제곱으로 라운딩하여 변경된 사이즈 (S') 를 갖는 변환 행렬들을 갖는 역변환을 생성하고;

잔차 비디오 데이터를 생성하기 위해 비디오 데이터의 상기 인코딩된 블록에 상기 변경된 사이즈 (S') 를 갖는 상기 변환 행렬들을 갖는 상기 역변환을 적용하며; 그리고

상기 비디오 데이터의 디코딩된 블록을 생성하기 위해 상기 잔차 비디오 데이터를 디코딩하도록

구성되는, 비디오 데이터를 디코딩하도록 구성된 장치.

청구항 18

제 17 항에 있어서,

S 를 2 의 거듭제곱으로 라운딩하기 위해, 상기 하나 이상의 프로세서들은 S 를 가장 가까운 2 의 거듭제곱으로 라운딩하도록 구성되는, 비디오 데이터를 디코딩하도록 구성된 장치.

청구항 19

제 17 항에 있어서,

상기 비디오 데이터의 상기 인코딩된 블록은 역양자화된 변환 계수들을 포함하는, 비디오 데이터를 디코딩하도록 구성된 장치.

청구항 20

제 17 항에 있어서,

상기 비디오 데이터의 상기 인코딩된 블록은 비정사각형 형상을 갖는, 비디오 데이터를 디코딩하도록 구성된 장치.

청구항 21

제 17 항에 있어서,

S 는 12 이고, S 를 2 의 거듭제곱으로 라운딩하기 위해, 상기 하나 이상의 프로세서들은 12 를 16 으로 라운딩하도록 구성되고, 상기 변경된 사이즈 (S') 는 16 인, 비디오 데이터를 디코딩하도록 구성된 장치.

청구항 22

제 17 항에 있어서,

S 는 24 이고, S 를 2 의 거듭제곱으로 라운딩하기 위해, 상기 하나 이상의 프로세서들은 24 를 32 로 라운딩하도록 구성되고, 상기 변경된 사이즈 (S') 는 32 인, 비디오 데이터를 디코딩하도록 구성된 장치.

청구항 23

제 17 항에 있어서,

S 는 상기 변환의 너비인, 비디오 데이터를 디코딩하도록 구성된 장치.

청구항 24

제 17 항에 있어서,

S 는 상기 변환의 높이인, 비디오 데이터를 디코딩하도록 구성된 장치.

청구항 25

제 17 항에 있어서,

상기 비디오 데이터의 상기 디코딩된 블록을 디스플레이하도록 구성된 디스플레이를 더 포함하는, 비디오 데이터를 디코딩하도록 구성된 장치.

청구항 26

비디오 데이터를 인코딩하도록 구성된 장치로서,

상기 비디오 데이터를 저장하도록 구성된 메모리; 및

상기 메모리와 통신하는 하나 이상의 프로세서들을 포함하고,

상기 하나 이상의 프로세서들은,

상기 비디오 데이터의 블록을 수신하는 것으로서, 상기 비디오 데이터의 상기 블록은 삼중-트리 파티션 구조로 파티셔닝되고, 상기 삼중-트리 파티션 구조는 상기 블록의 중심을 통해 상기 블록을 분할함이 없이 상기 블록을 3 개의 서브-블록들로 분할하는, 상기 비디오 데이터의 블록을 수신하는 것을 행하고;

잔차 비디오 데이터를 생성하기 위해 상기 비디오 데이터의 상기 블록을 예측하며;

상기 잔차 비디오 데이터에 대한 변환을 결정하는 것으로서, 상기 변환의 변환 행렬들은 2 의 거듭제곱이 아닌 사이즈 (S) 를 갖는, 상기 변환을 결정하는 것을 행하고;

S 를 2 의 거듭제곱으로 라운딩하여 변경된 사이즈 (S') 를 갖는 변환 행렬들을 갖는 변환을 생성하며;

변환 계수들을 생성하기 위해 상기 잔차 비디오 데이터에 상기 변경된 사이즈 (S') 를 갖는 상기 변환 행렬들을 갖는 상기 변환을 적용하고; 그리고

인코딩된 비디오 비트스트림에 상기 변환 계수들을 인코딩하도록

구성되는, 비디오 데이터를 인코딩하도록 구성된 장치.

청구항 27

제 26 항에 있어서,

S 를 2 의 거듭제곱으로 라운딩하기 위해, 상기 하나 이상의 프로세서들은 S 를 가장 가까운 2 의 거듭제곱으로 라운딩하도록 구성되는, 비디오 데이터를 인코딩하도록 구성된 장치.

청구항 28

제 26 항에 있어서,

상기 하나 이상의 프로세서들은 또한 상기 변환 계수들을 양자화하도록 구성되는, 비디오 데이터를 인코딩하도록 구성된 장치.

청구항 29

제 26 항에 있어서,

비디오 데이터의 상기 블록은 비정사각형 형상을 갖는, 비디오 데이터를 인코딩하도록 구성된 장치.

청구항 30

제 26 항에 있어서,

S 는 12 이고, S 를 2 의 거듭제곱으로 라운딩하기 위해, 상기 하나 이상의 프로세서들은 12 를 16 으로 라운딩하도록 구성되고, 상기 변경된 사이즈 (S') 는 16 인, 비디오 데이터를 인코딩하도록 구성된 장치.

청구항 31

제 26 항에 있어서,

S 는 24 이고, S 를 2 의 거듭제곱으로 라운딩하기 위해, 상기 하나 이상의 프로세서들은 24 를 32 로 라운딩하도록 구성되고, 상기 변경된 사이즈 (S') 는 32 인, 비디오 데이터를 인코딩하도록 구성된 장치.

청구항 32

제 26 항에 있어서,

S 는 상기 변환의 너비인, 비디오 데이터를 인코딩하도록 구성된 장치.

청구항 33

제 26 항에 있어서,

S 는 상기 변환의 높이인, 비디오 데이터를 인코딩하도록 구성된 장치.

청구항 34

제 26 항에 있어서,

상기 비디오 데이터를 캡쳐하도록 구성된 카메라를 더 포함하는, 비디오 데이터를 인코딩하도록 구성된 장치.

청구항 35

비디오 데이터를 디코딩하도록 구성된 장치로서,

비디오 데이터의 인코딩된 블록을 수신하는 수단으로서, 비디오 데이터의 상기 블록은 삼중-트리 파티션 구조로 파티셔닝되고, 상기 삼중-트리 파티션 구조는 상기 블록의 중심을 통해 상기 블록을 분할함이 없이 상기 블록을 3 개의 서브-블록들로 분할하는, 상기 비디오 데이터의 인코딩된 블록을 수신하는 수단;

비디오 데이터의 상기 인코딩된 블록에 대한 변환을 결정하는 수단으로서, 상기 변환의 변환 행렬들은 2 의 거듭제곱이 아닌 사이즈 (S) 를 갖는, 상기 변환을 결정하는 수단;

S 를 2 의 거듭제곱으로 라운딩하여 변경된 사이즈 (S') 를 갖는 변환 행렬들을 갖는 역변환을 생성하는 수단;

잔차 비디오 데이터를 생성하기 위해 비디오 데이터의 상기 인코딩된 블록에 상기 변경된 사이즈 (S') 를 갖는 상기 변환 행렬들을 갖는 상기 역변환을 적용하는 수단; 및

비디오 데이터의 디코딩된 블록을 생성하기 위해 상기 잔차 비디오 데이터를 디코딩하는 수단을 포함하는, 비디오 데이터를 디코딩하도록 구성된 장치.

청구항 36

비디오 데이터를 인코딩하도록 구성된 장치로서,

비디오 데이터의 블록을 수신하는 수단으로서, 비디오 데이터의 상기 블록은 삼중-트리 파티션 구조로 파티셔닝

되고, 상기 삼중-트리 파티션 구조는 상기 블록의 중심을 통해 상기 블록을 분할함이 없이 상기 블록을 3 개의 서브-블록들로 분할하는, 상기 비디오 데이터의 블록을 수신하는 수단;

잔차 비디오 데이터를 생성하기 위해 비디오 데이터의 상기 블록을 예측하는 수단;

상기 잔차 비디오 데이터에 대한 변환을 결정하는 수단으로서, 상기 변환의 변환 행렬들은 2 의 거듭제곱이 아닌 사이즈 (S) 를 갖는, 상기 변환을 결정하는 수단;

S 를 2 의 거듭제곱으로 라운딩하여 변경된 사이즈 (S') 를 갖는 변환 행렬들을 갖는 변환을 생성하는 수단;

변환 계수들을 생성하기 위해 상기 잔차 비디오 데이터에 상기 변경된 사이즈 (S') 를 갖는 상기 변환 행렬들을 갖는 상기 변환을 적용하는 수단; 및

인코딩된 비디오 비트스트림에 상기 변환 계수들을 인코딩하는 수단을 포함하는, 비디오 데이터를 인코딩하도록 구성된 장치.

청구항 37

명령들을 저장하는 비일시적 컴퓨터 판독가능 저장 매체로서,

상기 명령들은, 실행될 때, 비디오 데이터를 디코딩하도록 구성된 디바이스의 하나 이상의 프로세서들로 하여금,

상기 비디오 데이터의 인코딩된 블록을 수신하는 것으로서, 상기 비디오 데이터의 상기 블록은 삼중-트리 파티션 구조로 파티셔닝되고, 상기 삼중-트리 파티션 구조는 상기 블록의 중심을 통해 상기 블록을 분할함이 없이 상기 블록을 3 개의 서브-블록들로 분할하는, 상기 비디오 데이터의 인코딩된 블록을 수신하는 것을 행하게 하고;

비디오 데이터의 상기 인코딩된 블록에 대한 변환을 결정하는 것으로서, 상기 변환의 변환 행렬들은 2 의 거듭제곱이 아닌 사이즈 (S) 를 갖는, 상기 변환을 결정하는 것을 행하게 하며;

S 를 2 의 거듭제곱으로 라운딩하여 변경된 사이즈 (S') 를 갖는 변환 행렬들을 갖는 역변환을 생성하게 하고;

잔차 비디오 데이터를 생성하기 위해 비디오 데이터의 상기 인코딩된 블록에 상기 변경된 사이즈 (S') 를 갖는 상기 변환 행렬들을 갖는 상기 역변환을 적용하게 하며; 그리고

상기 비디오 데이터의 디코딩된 블록을 생성하기 위해 상기 잔차 비디오 데이터를 디코딩하게 하는, 비일시적 컴퓨터 판독가능 저장 매체.

청구항 38

명령들을 저장하는 비일시적 컴퓨터 판독가능 저장 매체로서,

상기 명령들은, 실행될 때, 비디오 데이터를 인코딩하도록 구성된 디바이스의 하나 이상의 프로세서들로 하여금,

상기 비디오 데이터의 블록을 수신하는 것으로서, 상기 비디오 데이터의 상기 블록은 삼중-트리 파티션 구조로 파티셔닝되고, 상기 삼중-트리 파티션 구조는 상기 블록의 중심을 통해 상기 블록을 분할함이 없이 상기 블록을 3 개의 서브-블록들로 분할하는, 상기 비디오 데이터의 블록을 수신하는 것을 행하게 하고;

잔차 비디오 데이터를 생성하기 위해 비디오 데이터의 상기 블록을 예측하게 하며;

상기 잔차 비디오 데이터에 대한 변환을 결정하는 것으로서, 상기 변환의 변환 행렬들은 2 의 거듭제곱이 아닌 사이즈 (S) 를 갖는, 상기 변환을 결정하는 것을 행하게 하고;

S 를 2 의 거듭제곱으로 라운딩하여 변경된 사이즈 (S') 를 갖는 변환 행렬들을 갖는 변환을 생성하게 하며;

변환 계수들을 생성하기 위해 상기 잔차 비디오 데이터에 상기 변경된 사이즈 (S') 를 갖는 상기 변환 행렬들을 갖는 상기 변환을 적용하게 하고; 그리고

인코딩된 비디오 비트스트림에 상기 변환 계수들을 인코딩하게 하는, 비일시적 컴퓨터 판독가능 저장 매체.

청구항 39

삭제

청구항 40

삭제

발명의 설명

기 술 분 야

[0001] 본 개시는 2017년 1월 6일자로 출원된 미국 가출원 제 62/443,569 호의 이익을 주장하며, 그 전체의 내용은 여기에 참조에 의해 포함된다.

[0002] 본 개시는 비디오 인코딩 및 비디오 디코딩에 관한 것이다.

배 경 기 술

[0003] 디지털 비디오 능력들은 디지털 텔레비전, 디지털 직접 브로드캐스트 시스템, 무선 브로드캐스트 시스템, 개인용 휴대정보단말 (PDA), 랩톱 또는 데스크톱 컴퓨터, 태블릿 컴퓨터, e-북 판독기, 디지털 카메라, 디지털 기록 디바이스, 디지털 미디어 재생기, 비디오 게이밍 디바이스, 비디오 게임 콘솔, 셀룰러 또는 위성 무선 전화, 소위 "스마트 폰", 화상 회의 디바이스, 비디오 스트리밍 디바이스 등을 포함하는 광범위한 디바이스들로 포함될 수 있다. 디지털 비디오 디바이스는 MPEG-2, MPEG-4, ITU-T H.263, ITU-T H.264/MPEG-4, Part 10, Advanced Video Coding (AVC), 고효율 비디오 코딩 (HEVC) 표준, 및 그러한 표준들의 확장들에 의해 정의된 표준들에서 기술된 것들과 같은 비디오 코딩 기법들을 구현한다. 비디오 디바이스는 그러한 비디오 코딩 기법들을 구현함으로써 더 효율적으로 디지털 비디오 정보를 송신, 수신, 인코딩, 디코딩, 및/또는 저장할 수도 있다.

[0004] 비디오 코딩 기법들은 비디오 시퀀스들에 고유한 러던던시를 감소시키거나 제거하기 위해 공간 (인트라-픽처) 예측 및/또는 시간 (인터-픽처) 예측을 포함한다. 블록 기반 비디오 코딩의 경우, 비디오 슬라이스 (예를 들어, 비디오 프레임 또는 비디오 프레임의 일부) 는 트리블록, 코딩 유닛 (CU) 및/또는 코딩 노드로서도 지칭될 수도 있는 비디오 블록들로 파티셔닝될 수도 있다. 픽처는 프레임으로서 지칭될 수도 있다. 참조 픽처는 참조 프레임으로서 지칭될 수도 있다.

[0005] 공간 또는 시간 예측은 코딩될 블록에 대한 예측 블록을 야기한다. 잔차 데이터는 코딩될 오리지날 블록과 예측 블록 사이의 픽처 차이들을 나타낸다. 추가의 압축의 경우, 잔차 데이터는 픽셀 도메인에서 변환 도메인으로 변환되어, 양자화될 수도 있는 잔차 변환 계수들을 야기할 수도 있다. 엔트로피 코딩이 훨씬 더 많은 압축을 달성하기 위해 적용될 수도 있다.

발명의 내용

해결하려는 과제

과제의 해결 수단

[0006] 본 개시는 다중-유형-트리 (MTT) 프레임워크를 사용하여 비디오 데이터의 블록들을 파티셔닝하기 위한 기법들을 기술한다. 본 개시의 기법들은 트리 구조의 여러 노드들에서 복수의 파티셔닝 기법들 중 하나를 결정하는 것을 포함한다. 복수의 파티셔닝 기법들의 예들은 블록의 중심을 통해 블록을 대칭적으로 분할하는 파티셔닝 기법들뿐 아니라, 대칭적으로 또는 블록의 중심이 분할되지 않도록 비대칭적으로 블록을 분할하는 파티셔닝 기법들을 포함할 수도 있다. 이러한 방식으로, 비디오 블록들의 파티셔닝은 블록들의 중심에 있는 비디오 데이터에서의 오브젝트들을 더 양호하게 캡처하는 파티셔닝을 포함하는 더 효율적인 코딩을 야기하는 방식으로 수행될 수 있다.

[0007] 본 개시는 또한 MTT 프레임워크에 따라 파티셔닝된 블록들에 변환들을 적용하기 위한 기법들, MTT 프레임워크에 따라 블록들이 분할되는 방법을 나타내는 신택스 엘리먼트들을 생성 및 파싱하기 위한 기법들, MTT 프레임워크에 따라 루마 및 크로마 블록들을 파티셔닝하기 위한 기법들, 및 MTT 프레임워크에 따라 파티셔닝된 블록들을 코딩 (즉, 인코딩 및/또는 디코딩) 하기 위한 기법들을 기술한다. 본 개시에 기술된 기법들은 개별적으로, 또

는 임의의 조합으로 함께 사용될 수도 있다.

[0008] 본 개시의 하나의 예에서, 비디오 데이터를 디코딩하는 방법은 비디오 데이터의 인코딩된 블록을 수신하는 단계; 비디오 데이터의 인코딩된 블록에 대한 변환을 결정하는 단계로서, 그 변환은 2 의 거듭제곱이 아닌 사이즈 (S) 를 갖는, 상기 변환을 결정하는 단계; S 를 2 의 거듭제곱으로 라운딩하여 변경된 사이즈 (S') 를 갖는 역변환을 생성하는 단계; 잔차 비디오 데이터를 생성하기 위해 비디오 데이터의 인코딩된 블록에 변경된 사이즈 (S') 를 갖는 역변환을 적용하는 단계; 및 비디오 데이터의 디코딩된 블록을 생성하기 위해 잔차 비디오 데이터를 디코딩하는 단계를 포함한다.

[0009] 본 개시의 다른 예에서, 비디오 데이터를 인코딩하는 방법은 비디오 데이터의 블록을 수신하는 단계; 잔차 비디오 데이터를 생성하기 위해 비디오 데이터의 블록을 예측하는 단계; 잔차 비디오 데이터에 대한 변환을 결정하는 단계로서, 그 변환은 2 의 거듭제곱이 아닌 사이즈 (S) 를 갖는, 상기 변환을 결정하는 단계; S 를 2 의 거듭제곱으로 라운딩하여 변경된 사이즈 (S') 를 갖는 변환을 생성하는 단계; 변환 계수들을 생성하기 위해 잔차 비디오 데이터에 변경된 사이즈 (S') 를 갖는 변환을 적용하는 단계; 및 인코딩된 비디오 비트스트림에 변환 계수들을 인코딩하는 단계를 포함한다.

[0010] 본 개시의 다른 예에서, 비디오 데이터를 디코딩하도록 구성된 장치는 비디오 데이터를 저장하도록 구성된 메모리, 및 그 메모리와 통신하는 하나 이상의프로세서들을 포함하고, 그 하나 이상의 프로세서들은 비디오 데이터의 인코딩된 블록을 수신하고; 비디오 데이터의 인코딩된 블록에 대한 변환을 결정하는 것으로서, 그 변환은 2 의 거듭제곱이 아닌 사이즈 (S) 를 갖는, 상기 변환을 결정하며; S 를 2 의 거듭제곱으로 라운딩하여 변경된 사이즈 (S') 를 갖는 역변환을 생성하고; 잔차 비디오 데이터를 생성하기 위해 비디오 데이터의 인코딩된 블록에 변경된 사이즈 (S') 를 갖는 역변환을 적용하며; 및 비디오 데이터의 디코딩된 블록을 생성하기 위해 잔차 비디오 데이터를 디코딩하도록 구성된다.

[0011] 본 개시의 다른 예에서, 비디오 데이터를 인코딩하도록 구성된 장치는 비디오 데이터를 저장하도록 구성된 메모리, 및 그 메모리와 통신하는 하나 이상의프로세서들을 포함하고, 그 하나 이상의 프로세서들은 비디오 데이터의 블록을 수신하고; 잔차 비디오 데이터를 생성하기 위해 비디오 데이터의 블록을 예측하며; 잔차 비디오 데이터에 대한 변환을 결정하는 것으로서, 그 변환은 2 의 거듭제곱이 아닌 사이즈 (S) 를 갖는, 상기 변환을 결정하고; S 를 2 의 거듭제곱으로 라운딩하여 변경된 사이즈 (S') 를 갖는 변환을 생성하며; 변환 계수들을 생성하기 위해 잔차 비디오 데이터에 변경된 사이즈 (S') 를 갖는 변환을 적용하고; 및 인코딩된 비디오 비트스트림에 변환 계수들을 인코딩하도록 구성된다.

[0012] 본 개시의 다른 예에서, 비디오 데이터를 디코딩하도록 구성된 장치는 비디오 데이터의 인코딩된 블록을 수신하는 수단; 비디오 데이터의 인코딩된 블록에 대한 변환을 결정하는 수단으로서, 그 변환은 2 의 거듭제곱이 아닌 사이즈 (S) 를 갖는, 상기 변환을 결정하는 수단; S 를 2 의 거듭제곱으로 라운딩하여 변경된 사이즈 (S') 를 갖는 역변환을 생성하는 수단; 잔차 비디오 데이터를 생성하기 위해 비디오 데이터의 인코딩된 블록에 변경된 사이즈 (S') 를 갖는 역변환을 적용하는 수단; 및 비디오 데이터의 디코딩된 블록을 생성하기 위해 잔차 비디오 데이터를 디코딩하는 수단을 포함한다.

[0013] 본 개시의 다른 예에서, 비디오 데이터를 인코딩하도록 구성된 장치는 비디오 데이터의 블록을 수신하는 수단; 잔차 비디오 데이터를 생성하기 위해 비디오 데이터의 블록을 예측하는 수단; 잔차 비디오 데이터에 대한 변환을 결정하는 수단으로서, 그 변환은 2 의 거듭제곱이 아닌 사이즈 (S) 를 갖는, 상기 변환을 결정하는 수단; S 를 2 의 거듭제곱으로 라운딩하여 변경된 사이즈 (S') 를 갖는 변환을 생성하는 수단; 변환 계수들을 생성하기 위해 잔차 비디오 데이터에 변경된 사이즈 (S') 를 갖는 변환을 적용하는 수단; 및 인코딩된 비디오 비트스트림에 변환 계수들을 인코딩하는 수단을 포함한다.

[0014] 다른 예에서, 본 개시는 명령들을 저장하는 컴퓨터 판독가능 저장 매체를 기술하며, 그 명령들은, 실행될 때, 비디오 데이터를 디코딩하도록 구성된 디바이스의 하나 이상의 프로세서들로 하여금, 비디오 데이터의 인코딩된 블록을 수신하게 하고; 비디오 데이터의 인코딩된 블록에 대한 변환을 결정하게 하는 것으로서, 그 변환은 2 의 거듭제곱이 아닌 사이즈 (S) 를 갖는, 상기 변환을 결정하게 하며; S 를 2 의 거듭제곱으로 라운딩하여 변경된 사이즈 (S') 를 갖는 역변환을 생성하게 하고; 잔차 비디오 데이터를 생성하기 위해 비디오 데이터의 인코딩된 블록에 변경된 사이즈 (S') 를 갖는 역변환을 적용하게 하며; 및 비디오 데이터의 디코딩된 블록을 생성하기 위해 잔차 비디오 데이터를 디코딩하게 한다.

[0015] 다른 예에서, 본 개시는 명령들을 저장하는 컴퓨터 판독가능 저장 매체를 기술하며, 그 명령들은, 실행될 때,

비디오 데이터를 인코딩하도록 구성된 디바이스의 하나 이상의 프로세서들로 하여금, 비디오 데이터의 블록을 수신하게 하고; 잔차 비디오 데이터를 생성하기 위해 비디오 데이터의 블록을 예측하게 하며; 잔차 비디오 데이터에 대한 변환을 결정하게 하는 것으로서, 그 변환은 2 의 거듭제곱이 아닌 사이즈 (S) 를 갖는, 상기 변환을 결정하게 하고; S 를 2 의 거듭제곱으로 라운딩하여 변경된 사이즈 (S') 를 갖는 변환을 생성하며; 변환 계수들을 생성하기 위해 잔차 비디오 데이터에 변경된 사이즈 (S') 를 갖는 변환을 적용하게 하고; 및 인코딩된 비디오 비트스트림에 변환 계수들을 인코딩하게 한다.

도면의 간단한 설명

[0016] 도 1 은 본 개시의 기법들을 구현하도록 구성된 예시의 비디오 인코딩 및 디코딩 시스템을 도시하는 블록도이다.

도 2 는 고효율 비디오 코딩 (HEVC) 에서 코딩 유닛 (CU) 구조를 도시하는 개념도이다.

도 3 은 인터 예측 모드에 대한 예시의 파티션 타입들을 도시하는 개념도이다.

도 4a 는 쿼드-트리-이진-트리 (QTBT) 구조를 사용하여 파티셔닝하는 블록의 예를 도시하는 개념도이다.

도 4b 는 도 4a 의 QTBT 구조를 사용하여 파티셔닝하는 블록에 대응하는 예시의 트리 구조를 도시하는 개념도이다.

도 5a 는 쿼드 트리 파니셔닝을 도시하는 개념도이다.

도 5b 는 수직 이진 트리 파니셔닝을 도시하는 개념도이다.

도 5c 는 수평 이진 트리 파니셔닝을 도시하는 개념도이다.

도 5d 는 수직 중심측 트리 파니셔닝을 도시하는 개념도이다.

도 5e 는 수평 중심측 트리 파니셔닝을 도시하는 개념도이다.

도 6 은 본 개시의 기법들에 따른 코딩 트리 유닛 (CTU) 파티셔닝의 예를 도시하는 개념도이다.

도 7 은 QTBT 파티셔닝의 일 예에 따른 예시의 비대칭 파티션들을 도시하는 개념도이다.

도 8 은 데드존 플러스 균일 양자화 방식을 도시하는 개념도이다.

도 9 는 예시의 비대칭 파티션 유형들을 도시한다.

도 10 은 비디오 인코더의 예를 도시하는 블록도이다.

도 11 은 비디오 디코더의 예를 도시하는 블록도이다.

도 12 는 본 개시의 예시의 인코딩 방법을 도시하는 플로우챠트이다.

도 13 은 본 개시의 예시의 디코딩 방법을 도시하는 플로우챠트이다.

발명을 실시하기 위한 구체적인 내용

[0017] 이 개시는 블록 기반 비디오 코딩에서 비디오 데이터의 블록들 (예 : 코딩 유닛들) 의 파티셔닝 및/또는 조직화와 관련된다. 본 개시의 기술들은 비디오 코딩 표준들에서 적용될 수도 있다. 후술되는 다양한 예에서, 본 개시의 기술은 3 개 이상의 상이한 파티셔닝 구조를 사용하여 비디오 데이터의 블록을 파티셔닝하는 것을 포함한다. 일부 예에서, 코딩 트리 구조의 각각의 깊이에서 3 개 이상의 상이한 파티션 구조가 사용될 수도 있다. 이러한 파티셔닝 기술은 다중 유형 트리 (MTT) 파티셔닝으로 지칭될 수도 있다. MTT 파티셔닝을 사용함으로써, 비디오 데이터가 더 유연하게 파티셔닝될 수 있어, 더 큰 코딩 효율을 허용할 수도 있다.

[0018] 이 개시는 MTT 프레임워크에 따라 파티셔닝된 블록에 변환을 적용하는 기술, MTT 프레임워크에 따라 블록이 어떻게 분할되는지를 나타내는 신택스 엘리먼트를 생성 및 파싱하는 기술, MTT 프레임워크에 따라 루마 및 크로마 블록을 파티셔닝하는 기술, MTT 프레임워크에 따라 파티셔닝된 블록을 코딩 (즉, 인코딩 및/또는 디코딩) 하는 기술을 또한 기술한다. 본 개시에 기술된 기술들은 개별적으로 또는 임의의 조합으로 함께 사용될 수도 있다.

[0019] 도 1 은 비디오 데이터의 블록을 파티셔닝하고, 파티션 타입을 시그널링하고 파싱하고, 변환 및 추가의 변한 파티션들을 적용하기 위한 본 개시의 기술을 이용할 수도 있는 예시적인 비디오 인코딩 및 디코딩 시스템 (10) 을

나타내는 블록도이다. 도 1 에 도시된 바와 같이, 시스템 (10) 은, 목적지 디바이스 (14) 에 의해 더 나중 시간에 디코딩될 인코딩된 비디오 데이터를 제공하는 소스 디바이스 (12) 를 포함한다. 특히, 소스 디바이스 (12) 는 비디오 데이터를, 컴퓨터 판독가능 매체 (16) 를 통해 목적지 디바이스 (14) 에 제공한다. 소스 디바이스 (12) 및 목적지 디바이스 (14) 는, 데스크탑 컴퓨터들, 노트북 (즉, 랩톱) 컴퓨터들, 태블릿 컴퓨터들, 셋톱 박스들, 전화기 핸드셋 이를테면 소위 "스마트" 폰들, 태블릿 컴퓨터, 텔레비전들, 카메라들, 디스플레이 디바이스들, 디지털 미디어 재생기들, 비디오 게이밍 콘솔들, 비디오 스트리밍 디바이스 등을 포함한, 광범위한 디바이스들 중 어느 것을 포함할 수도 있다. 일부 경우에, 소스 디바이스 (12) 및 목적지 디바이스 (14) 는 무선 통신을 위해 갖추어질 수도 있다. 따라서, 소스 디바이스 (12) 및 목적지 디바이스 (14) 는 무선 통신 디바이스들일 수도 있다. 소스 디바이스 (12) 는 예시적인 비디오 인코딩 디바이스 (즉, 비디오 데이터를 인코딩하기 위한 디바이스) 이다. 목적지 디바이스 (14) 는 예시적인 비디오 디코딩 디바이스 (예를 들어, 비디오 데이터를 디코딩하기 위한 디바이스 또는 장치) 이다.

[0020] 도 1 의 예에서, 소스 디바이스 (12) 는 비디오 소스 (18), 비디오 데이터를 저장하도록 구성된 저장 매체 (20), 비디오 인코더 (22), 및 출력 인터페이스 (24) 를 포함한다. 목적지 디바이스 (14) 는 입력 인터페이스 (26), 인코딩된 비디오 데이터를 저장하도록 구성된 저장 매체 (28), 비디오 디코더 (30) 및 디스플레이 디바이스 (32) 를 포함한다. 다른 예들에서, 소스 디바이스 (12) 및 목적지 디바이스 (14) 는 다른 컴포넌트들 또는 배열들을 포함한다. 예를 들어, 소스 디바이스 (12) 는 외부 카메라와 같은 외부 비디오 소스로부터 비디오 데이터를 수신할 수도 있다. 유사하게, 목적지 디바이스 (14) 는 통합된 디스플레이 디바이스를 포함하는 것보다는 외부 디스플레이 디바이스와 인터페이싱할 수도 있다.

[0021] 도 1 의 예시된 시스템 (10) 은 단지 일 예일 뿐이다. 비디오 데이터를 프로세싱하기 위한 기법들은 임의의 디지털 비디오 인코딩 및/또는 디코딩 디바이스 또는 장치에 의해 수행될 수도 있다. 비록 일반적으로 본 개시의 기술들이 비디오 인코딩 디바이스 및 비디오 디코딩 디바이스에 의해 수행되지만, 그 기술들은 또한, 통상적으로 "CODEC" 로서 지칭되는 결합된 비디오 인코더/디코더에 의해 수행될 수도 있다. 소스 디바이스 (12) 및 목적지 디바이스 (12) 는, 단지, 소스 디바이스 (14) 가 목적지 디바이스 (14) 로의 송신을 위한 인코딩된 비디오 데이터를 생성하는 그러한 코딩 디바이스들의 예들일 뿐이다. 일부 예들에서, 소스 디바이스 (12) 및 목적지 디바이스 (14) 는, 소스 디바이스 (12) 및 목적지 디바이스 (14) 의 각각이 비디오 인코딩 및 디코딩 컴포넌트들을 포함하도록 실질적으로 대칭 방식으로 동작한다. 그러므로, 시스템 (10) 은 예를 들면, 비디오 스트리밍, 비디오 플레이백, 비디오 브로드캐스팅 또는 화상 통화를 위해, 소스 디바이스 (12) 와 목적지 디바이스 (14) 간의 일방향 또는 양방향 비디오 송신을 지원할 수도 있다.

[0022] 소스 디바이스 (12) 의 비디오 소스 (18) 는 비디오 카메라와 같은 비디오 캡처 디바이스, 이전에 캡처된 비디오를 포함하는 비디오 아카이브 (video archive), 및/또는 비디오 콘텐트 제공자로부터 비디오 데이터를 수신하기 위한 비디오 피드 인터페이스 (video feed interface) 를 포함할 수도 있다. 추가적인 대안으로서, 비디오 소스 (18) 는 컴퓨터 그래픽 기반 데이터를 소스 비디오로서, 또는 라이브 비디오, 아카이브된 비디오, 및 컴퓨터 생성된 비디오의 조합으로서 생성할 수도 있다. 소스 디바이스 (12) 는 비디오 데이터를 저장하도록 구성된 하나 이상의 데이터 저장 매체 (예를 들어, 저장 매체 (20)) 를 포함할 수도 있다. 본 개시에 설명된 기법들은, 일반적으로 비디오 코딩에 적용가능할 수도 있고, 무선 및/또는 유선 어플리케이션들에 적용될 수도 있다. 각각의 경우에 있어서, 캡쳐되거나 사전-캡쳐되거나 또는 컴퓨터 생성된 비디오는 비디오 인코더 (22) 에 의해 인코딩될 수도 있다. 출력 인터페이스 (24) 는 인코딩된 비디오 정보를 컴퓨터 판독 가능 매체 (16) 에 출력할 수도 있다.

[0023] 목적지 디바이스 (14) 는, 컴퓨터 판독 가능 매체 (16) 를 통해 디코딩될 인코딩된 비디오 데이터를 수신할 수도 있다. 컴퓨터 판독 가능 매체 (16) 는, 인코딩된 비디오 데이터를 소스 디바이스 (12) 로부터 목적지 디바이스 (14) 로 이동시킬 수 있는 임의의 유형의 매체 또는 디바이스를 포함할 수도 있다. 일부 예들에서, 컴퓨터 판독가능 매체 (16) 는, 소스 디바이스 (12) 로 하여금 실시간으로 직접 목적지 디바이스 (14) 로, 인코딩된 비디오 데이터를 송신할 수 있게 하기 위한 통신 매체를 포함한다. 인코딩된 비디오 데이터는 무선 통신 프로토콜과 같은 통신 표준에 따라 변조되고, 목적지 디바이스 (14) 로 송신될 수도 있다. 통신 매체는 무선 주파수 (RF) 스펙트럼 또는 하나 이상의 물리적 송신 라인들과 같은 임의의 무선 또는 유선 통신 매체를 포함할 수도 있다. 통신 매체는 로컬 영역 네트워크, 광역 네트워크, 또는 인터넷과 같은 글로벌 네트워크와 같은 패킷 기반 네트워크의 부분을 형성할 수도 있다. 통신 매체는 라우터들, 스위치들, 기지국들, 또는 소스 디바이스 (12) 로부터 목적지 디바이스 (14) 로의 통신을 용이하게 하는데 유용할 수도 있는 임의의 다른 장비를 포함할 수도 있다. 목적지 디바이스 (14) 는 인코딩된 비디오 데이터 및 디코딩된 비디오 데이터를 저장하도록 구성된

하나 이상의 데이터 저장 매체를 포함할 수도 있다.

[0024] 일부 예들에서, 인코딩된 데이터 (예를 들어, 인코딩된 비디오 데이터) 는 출력 인터페이스 (24) 로부터 저장 디바이스로 출력될 수도 있다. 유사하게, 인코딩된 데이터는 입력 인터페이스 (26) 에 의해 저장 디바이스로부터 액세스될 수도 있다. 저장 디바이스는 하드 드라이브, 블루-레이 디스크들, DVD들, CD-ROM들, 플래시 메모리, 휘발성 또는 비휘발성 메모리, 서버, 또는 인코딩된 비디오 데이터를 저장하기 위한 임의의 다른 적합한 디지털 저장 매체들과 같은 다양한 분산된 또는 국부적으로 액세스된 데이터 저장 매체들 중 임의의 데이터 저장 매체를 포함할 수도 있다. 다른 예에서, 저장 디바이스는, 소스 디바이스 (12) 에 의해 생성되는 인코딩된 비디오를 저장할 수도 있는, 파일 서버 또는 또 다른 중간 저장 디바이스에 대응할 수도 있다. 목적지 디바이스 (14) 는, 스트리밍 또는 다운로드를 통해 저장 디바이스로부터 저장된 비디오 데이터에 액세스할 수도 있다. 파일 서버는, 인코딩된 비디오 데이터를 저장하고 그 인코딩된 비디오 데이터를 목적지 디바이스 (14) 로 송신할 수 있는 임의의 타입의 서버일 수도 있다. 예시적인 파일 서버들은, (예를 들어, 웹사이트용) 웹 서버, FTP 서버, NAS (network attached storage) 디바이스, 또는 로컬 디스크 드라이브를 포함한다. 목적지 디바이스 (14) 는, 인터넷 접속을 포함한, 임의의 표준 데이터 접속을 통해 인코딩된 비디오 데이터에 액세스할 수도 있다. 이것은, 파일 서버 상에 저장된 인코딩된 비디오 데이터를 액세스하는데 적합한 무선 채널 (예컨대, Wi-Fi 접속), 유선 접속 (예컨대, DSL, 케이블 모뎀 등), 또는 양자의 조합을 포함할 수도 있다. 저장 디바이스로부터의 인코딩된 비디오 데이터의 송신은 스트리밍 송신, 다운로드 송신, 또는 이들의 조합일 수도 있다.

[0025] 본 개시의 기법들은, 공중 경유 (over-the-air) 텔레비전 브로드캐스트, 케이블 텔레비전 송신, 위성 텔레비전 송신, DASH (dynamic adaptive streaming over HTTP) 와 같은 인터넷 스트리밍 비디오 송신, 데이터 저장 매체 상에 인코딩되는 디지털 비디오, 데이터 저장 매체 상에 저장된 디지털 비디오의 디코딩, 또는 다른 애플리케이션 등의 다양한 멀티미디어 애플리케이션들 중 어느 것을 지원하는 비디오 코딩에 적용될 수도 있다. 일부 예들에서, 시스템 (10) 은, 비디오 스트리밍, 비디오 플레이백, 비디오 브로드캐스팅 및/또는 화상 통화 등의 애플리케이션들을 지원하기 위하여 일방향 또는 양방향 비디오 송신을 지원하도록 구성될 수도 있다.

[0026] 컴퓨터 판독가능 매체 (16) 는 무선 브로드캐스트 또는 유선 네트워크 송신과 같은 일시적인 매체들, 또는 하드 디스크, 플래시 드라이브, 컴팩트 디스크, 디지털 비디오 디스크, 블루-레이 디스크, 서버, 또는 다른 컴퓨터 판독가능 매체들과 같은 저장 매체들 (즉, 비-일시적 저장 매체들) 을 포함할 수도 있다. 일부 예들에서, 네트워크 서버 (미도시) 는 인코딩된 비디오 데이터를 소스 디바이스 (12) 로부터 수신하고, 인코딩된 비디오 데이터를, 예를 들어, 네트워크 송신을 통해 목적지 디바이스 (14) 에 제공할 수도 있다. 유사하게, 디스크 스탬핑 설비와 같은 매체 생성 설비의 컴퓨팅 디바이스는 인코딩된 비디오 데이터를 소스 디바이스 (12) 로부터 수신하고, 인코딩된 비디오 데이터를 포함하는 디스크를 생성할 수도 있다. 따라서, 컴퓨터 판독가능 매체 (16) 는, 다양한 예들에 있어서, 다양한 형태들의 하나 이상의 컴퓨터 판독가능 매체들을 포함하는 것으로 이해될 수도 있다.

[0027] 목적지 디바이스 (14) 의 입력 인터페이스 (26) 는 컴퓨터 판독가능 매체 (16) 로부터 정보를 수신한다. 컴퓨터 판독가능 매체 (16) 의 정보는 비디오 인코더 (20) 의 비디오 인코더 (22) 에 의해 정의되고 또한 비디오 디코더 (30) 에 의해 이용되는 신택스 정보를 포함할 수도 있으며, 이 신택스 정보는 블록들 및 다른 코딩된 유닛들, 예를 들어, 픽처들의 그룹 (GOP들) 의 특징들 및/또는 프로세싱을 기술하는 신택스 엘리먼트들을 포함한다. 저장 매체 (28) 는 입력 인터페이스 (26) 에 의해 수신된 인코딩된 비디오 데이터를 저장할 수도 있다. 디스플레이 디바이스 (32) 는 디코딩된 비디오 데이터를 사용자에게 디스플레이한다. 디스플레이 디바이스 (32) 는 액정 디스플레이 (LCD), 플라즈마 디스플레이, 유기 발광 다이오드 (OLED) 디스플레이, 또는 다른 타입의 디스플레이 디바이스와 같은 다양한 디스플레이 디바이스들 중 임의의 것을 포함할 수도 있다.

[0028] 비디오 인코더 (22) 및 비디오 디코더 유닛 (30) 각각은 하나 이상의 마이크로프로세서들, 디지털 신호 프로세서(DSP)들, 주문형 집적회로(ASIC)들, 필드 프로그래머블 게이트 어레이(FPGA)들, 별도의 로직, 소프트웨어, 하드웨어, 펌웨어 또는 이들의 임의의 조합들과 같은 다양한 적절한 인코더 또는 디코더 회로 중 임의의 회로로서 구현될 수도 있다. 그 기법들이 부분적으로 소프트웨어로 구현될 때, 디바이스는 적합한 비일시적 컴퓨터 판독가능 매체에 그 소프트웨어를 위한 명령들을 저장하고 본 개시의 기법들을 수행하기 위하여 하나 이상의 프로세서들을 이용하여 하드웨어에서 그 명령들을 실행할 수도 있다. 비디오 인코더 (22) 및 비디오 디코더 (30) 의 각각은 하나 이상의 인코더들 또는 디코더들에 포함될 수도 있는데, 이들 중 어느 일방은 각각의 디바이스에서 결합된 인코더/디코더 (CODEC) 의 부분으로서 통합될 수도 있다.

[0029] 일부 예들에 있어서, 비디오 인코더 (22) 및 비디오 디코더 (30) 는 비디오 코딩 표준에 따라 동작할 수도

있다. 예시적인 비디오 코딩 표준들은, 비한정적으로, ITU-T H.261, ISO/IEC MPEG-1 Visual, ITU-T H.262 또는 ISO/IEC MPEG-2 Visual, ITU-T H.263, ISO/IEC MPEG-4 Visual 및 ITU-T H.264 (ISO/IEC MPEG-4 AVC 으로도 알려짐) 을 포함하며, 그의 SVC (Scalable Video Coding) 및 MVC (Multi-View Video Coding) 확장들을 포함한다. 비디오 코딩 표준, 고효율 비디오 코딩 (HEVC) 은 또는 ITU-T H.265ITU-T 은 그 범위 및 스크린 컨텐츠 코딩 확장들, 즉 3D 비디오 코딩 (3D-HEVC) 및 멀티뷰 확장들 (MV-HEVC) 및 스케일러블 확장 (SHVC) 들을 포함하여, ITU-T 비디오 코딩 전문가 그룹 (VCEG) 및 ISO/IEC 모션 픽처 전문가 그룹 (MPEG) 의 비디오 코딩 공동 협력 팀 (JCT-VC) 에 의해 개발되었다. 비디오 인코더 (22) 및 비디오 디코더 (30) 는 또한 JVET (Joint Video Exploration Team) 그룹에 의해 개발되고 있는 비디오 코딩 표준과 같은 장래의 비디오 코딩 표준에 따라 동작하도록 구성될 수도 있다. JEM 소프트웨어는 HEVC 모델 (HM) 소프트웨어에 기초하며 JVET 의 참조 소프트웨어이다.

[0030] HEVC 및 다른 비디오 코딩 규격들에서, 비디오 시퀀스는 통상적으로 일련의 픽쳐 (picture) 들을 포함한다. 픽쳐들은 "프레임들"로서 또한 지칭될 수도 있다. 픽쳐는 S_L, S_{Cb}, 및 S_{Cr} 로서 표기되는 3 개의 샘플 어레이들을 포함할 수도 있다. S_L 은 루마 샘플들의 2차원 어레이 (즉, 블록) 이다. S_{Cb} 는 Cb 크로미넌스 샘플들의 2-차원 어레이이다. S_{Cr} 는 Cr 크로미넌스 샘플들의 2-차원 어레이이다. 크로미넌스 샘플들은 또한, 본 명세서에서 "크로마 (chroma)" 샘플들로서 지칭될 수도 있다. 다른 경우들에서, 픽쳐는 단색 (monochrome) 일 수도 있고, 루마 샘플들의 어레이만을 포함할 수도 있다.

[0031] 또한, HEVC 및 다른 비디오 코딩 사양들에서, 픽쳐의 인코딩된 표현을 생성하기 위해, 비디오 인코더 (22) 는 코딩 트리 유닛 (CTU) 들의 세트를 생성할 수도 있다. CTU 들의 각각은 루마 샘플들의 코딩 트리 블록, 크로마 샘플들의 2개의 대응 코딩 트리 블록들, 및 코딩 트리 블록들의 샘플들을 코딩하는데 사용된 신택스 구조들을 포함할 수도 있다. 단색 화상들 또는 3개의 분리된 색 평면들을 갖는 화상들에서, CTU 는 단일 코딩 트리 블록 및 그 코딩 트리 블록의 샘플들을 코딩하는데 사용된 신택스 구조들을 포함할 수도 있다. 코딩 트리 블록은 샘플들의 NxN 블록일 수도 있다. CTU 는 "트리 블록" 또는 "최대 코딩 유닛" (LCU) 으로 지칭될 수도 있다. HEVC 의 CTU 들은 대체로, H.264/AVC 와 같은 다른 표준들의 매크로블록들에 유사할 수도 있다. 하지만, CTU 가 특정 크기로 반드시 한정되는 것은 아니고 하나 이상의 코딩 유닛들 (CU) 들을 포함할 수도 있다. 슬라이스는 래스터 스캔 순서에서 연속적으로 순서화된 정수 개의 CTU를 포함할 수도 있다.

[0032] HEVC 에 따라 동작하는 경우, 코딩된 CTU 를 생성하기 위하여, 비디오 인코더 (22) 는 CTU 의 코딩 트리 블록들에 대해 쿼드트리 파티셔닝을 재귀적으로 수행하여, 코딩 트리 블록들을 코딩 블록들, 따라서, 일명 "코딩 트리 유닛들"로 분할할 수도 있다. 코딩 블록은 샘플들의 NxN 블록이다. CU 는 루마 샘플들의 코딩 블록과, 루마 샘플 어레이, Cb 샘플 어레이, 및 Cr 샘플 어레이를 가지는 픽쳐의 크로마 샘플들의 2 개의 대응하는 코딩 블록들과, 코딩 블록들의 샘플들을 코딩하기 위하여 이용된 신택스 구조들을 포함할 수도 있다. 단색 픽쳐들 또는 3 개의 별개의 컬러 평면들을 갖는 픽쳐들에 있어서, CU 는 단일의 코딩 블록, 및 그 코딩 블록의 샘플들을 코딩하는데 사용된 신택스 구조들을 포함할 수도 있다.

[0033] 비트 스트림 내의 신택스 데이터는 또한 CTU의 크기를 정의할 수도 있다. 슬라이스는 코딩 순서에서의 다수의 연속적인 CTU 들을 포함한다. 비디오 프레임 또는 픽쳐는 하나 이상의 슬라이스들로 파티셔닝될 수도 있다. 상술된 바와 같이, 각각의 트리 블록은 쿼드트리에 따라 코딩 유닛 (CU) 들로 분할될 수도 있다. 일반적으로, 쿼드트리 데이터 구조는 CU 당 하나의 노드를 포함하며, 루트 노드는 트리블록에 대응한다. CU 가 4개의 서브-CU들로 분할되면, CU 에 대응하는 노드는 4개의 리프 (leaf) 노드들을 포함하며, 이들 각각은 서브-CU들 중 하나에 대응한다.

[0034] 쿼드트리 데이터 구조의 각각의 노드는 대응하는 CU에 대해 신택스 데이터를 제공할 수도 있다. 예를 들어, 쿼드트리에서의 노드는 그 노드에 대응하는 CU 가 서브-CU들로 분할되는지 여부를 표시하는 분할 플래그를 포함할 수도 있다. CU 에 대한 신택스 엘리먼트들은 재귀적으로 정의될 수도 있으며, CU 가 서브-CU들로 분할되는지 여부에 의존할 수도 있다. CU 가 추가로 분할되지 않으면, 리프-CU 로서 지칭된다. CU 의 블록이 더 분할되지 않을 경우, 그것은 일반적으로 비 리프 CU 로서 지칭될 수도 있다. 본 개시의 일부 예들에서, 오리지널 리프-CU 의 명시적인 분할이 존재하지 않더라도, 리프-CU 의 4개의 서브-CU들은 또한 리프-CU 들로서 지칭될 수도 있다. 예를 들어, 16x16 사이즈의 CU 가 더 분할되지 않으면, 16x16 CU 가 결코 분할되지 않았더라도, 4개의 8x8 서브-CU들은 또한 리프-CU들로서 지칭될 수도 있다.

[0035] CU 가 크기 구분 (size distinction) 을 가지지 않는다는 것을 제외하고는, CU 는 H.264 표준의 매크로블록

(macroblock) 과 유사한 목적을 가진다. 예를 들어, 트리 블록은 4 개의 자식 노드 (child node) 들 (또한,서브-CU 들로서 지칭됨) 로 분할될 수도 있고, 각각의 자식 노드는 차례로 부모 노드 (parent node) 일 수도 있고, 또 다른 4 개의 자식 노드들로 분할될 수도 있다. 쿼드트리의 리프 노드로서 지칭되는 최종의 미분할된 자식 노드는, 리프-CU 로서 또한 지칭되는 코딩 노드를 포함한다. 코딩된 비트스트림과 연관된 신택스 데이터는, 최대 CU 깊이로도 지칭되는, 트리 블록이 분할될 수도 있는 최대 횟수를 정의할 수도 있고, 또한 코딩 노드들의 최소 크기를 정의할 수도 있다. 트리 블록 구조의 깊이는 블록이 분할된 횟수를 나타낼 수도 있다. 예를 들어, 깊이 0 은 임의의 분할 이전의 블록과 관련될 수 있고, 깊이 1은 부모 블록의 하나의 분할로부터 생성된 블록과 관련될 수 있고, 깊이 2는 깊이 1 에서의 블록의 하나의 분할로부터 생성된 블록과 관련될 수 있다. 비트스트림은 또한 최소 코딩 유닛 (SCU) 을 정의할 수도 있다. 본 개시는, HEVC 의 컨텍스트에서의 CU, PU, 또는 TU 중 임의의 것을, 또는 다른 표준들의 컨텍스트에서의 유사한 데이터 구조들 (예를 들어, JEM 에서의 코딩 유닛, H.264/AVC 에서의 매크로블록들 및 그 서브-블록들 등) 을 지칭하기 위해 용어 "블록" 을 사용한다.

[0036] CU 는 코딩 노드 뿐아니라 그 코딩 노드와 연관된 예측 유닛들 (PU들) 및 변환 유닛들 (TU들) 을 포함한다. CU 의 사이즈는 코딩 노드의 사이즈에 대응하고, 일부 예들에서, 형상이 정방형일 수도 있다. HEVC 의 예에서, CU 의 사이즈는 8x8 픽셀들로부터, 최대 64x64 픽셀들 이상을 갖는 트리 블록의 사이즈까지 이를 수도 있다. 각각의 CU 는 하나 이상의 PU들 및 하나 이상의 TU들을 포함할 수도 있다. CU에 연관된 신택스 데이터는, 예를 들어, CU의 하나 이상의 PU들로의 파티셔닝을 기술할 수도 있다. 파티셔닝 모드들은, CU 가 스킵 또는 직접 모드 인코딩되는지, 인트라-예측 모드 인코딩되는지, 또는 인터-예측 모드 인코딩되는지 간에 달라질 수도 있다. PU들은 형상이 비-정사각형이 되도록 파티셔닝될 수도 있다. CU 와 연관된 신택스 데이터는 또한, 예를 들어, 쿼드트리에 따라 하나 이상의 TU들로의 CU 의 파티셔닝을 기술할 수도 있다. TU 는 형상이 정방형이거나 비-정방형 (예를 들어, 직방형) 일 수 있다.

[0037] HEVC 표준은 TU에 따른 변환을 허용한다. TU 들은 상이한 CU 들에 대해 상이할 수도 있다. TU들은 통상적으로, 파티셔닝된 LCU 에 대해 정의된 소정의 CU 내에서의 PU들의 사이즈에 기초하여 사이징되지만, 이것이 항상 그 경우인 것은 아닐 수도 있다. TU 들은 통상적으로 동일한 크기이거나 또는 PU 들보다 더 작다. 일부 예들에서, CU 에 대응하는 잔차 샘플들은 때때로 “잔차 쿼드 트리(residual quad tree)” (RQT) 로서 지칭되는 쿼드트리 구조를 사용하여 더 작은 유닛들로 세분될 수도 있다. RQT 의 리프 노드들은 TU들로서 지칭될 수도 있다. TU 들과 연관된 픽셀 차이 값들이 변환되어 변환 계수들을 생성하고, 이들은 양자화될 수도 있다.

[0038] 리프-CU 는 하나 이상의 PU 들을 포함할 수도 있다. 일반적으로, PU 는 대응하는 CU 의 부분 또는 그 모두에 대응하는 공간 영역을 나타내며, PU 에 대한 레퍼런스 샘플을 취출하기 위한 데이터를 포함할 수도 있다. 더욱이, PU 는 예측과 관련된 데이터를 포함한다. 예를 들어, PU 가 인트라-모드 인코딩될 경우, PU 에 대한 데이터는, 그 PU 에 대응하는 TU 에 대한 인트라-예측 모드를 기술하는 데이터를 포함할 수도 있는 RQT 에 포함될 수도 있다. 다른 예로서, PU 가 인터-모드 인코딩될 경우, PU 는 그 PU 에 대한 하나 이상의 모션 벡터들을 정의하는 데이터를 포함할 수도 있다. PU 에 대한 모션 벡터를 정의하는 데이터는 예를 들어, 모션 벡터의 수평 컴포넌트, 모션 벡터의 수직 컴포넌트, 모션 벡터에 대한 분해능 (예를 들어, 1/4 픽셀 정밀도 또는 1/8 픽셀 정밀도), 모션 벡터가 포인팅하는 레퍼런스 픽처, 및/또는 모션 벡터에 대한 레퍼런스 픽처 리스트 (예를 들어, 리스트 0, 리스트 1, 또는 리스트 C) 를 기술할 수도 있다.

[0039] 하나 이상의 PU 들을 갖는 리프 CU 는 또한 하나 이상의 TU들을 포함할 수도 있다. TU 들은, 상기 논의된 바와 같이 RQT (TU 쿼드트리 구조로서도 또한 지칭됨) 를 이용하여 명시될 수도 있다. 예를 들어, 분할된 플래그는 리프-CU 가 4개의 변환 유닛들로 분할되는지 여부를 표시할 수도 있다. 일부 예들에서, 각각의 변환 유닛은 추가의 서브-TU들로 더 분할될 수도 있다. TU 가 더 분할되지 않을 경우, 리프-TU 로서 지칭될 수도 있다. 일반적으로, 인트라 코딩에 대해, 리프-CU 에 속하는 리프-TU들 모두는 동일한 인트라 예측 모드로부터 생성된 잔차 데이터를 포함한다. 즉, 동일한 인트라-예측 모드는 일반적으로, 리프-CU 의 모든 TU 에서 변환될 예측된 값들을 계산하기 위해 적용된다. 인트라 코딩에 대해, 비디오 인코더 (22) 는 인트라 예측 모드를 이용하여 각각의 리프-TU 에 대한 잔차 값을, TU 에 대응하는 CU 의 부분과 오리지널 블록 간의 차이로서 계산할 수도 있다. TU 가 반드시 PU 의 사이즈로 한정될 필요는 없다. 따라서, TU들은 PU 보다 더 크거나 더 작을 수도 있다. 인트라 코딩에 대해, PU 는 동일한 CU 에 대한 대응하는 리프-TU 와 병치될 수도 있다. 일부 예들에 있어서, 리프-TU 의 최대 사이즈는 대응하는 리프-CU 의 사이즈에 대응할 수도 있다.

[0040] 더욱이, 리프-CU들의 TU들은 또한 각각의 RQT 구조들과 연관될 수도 있다. 즉, 리프-CU 는, 리프-CU가 TU 들로 어떻게 파티셔닝되는지를 나타내는 쿼드트리를 포함할 수도 있다. TU 쿼드트리의 루트 노드는 일반적으로 리프

-CU 에 대응하지만, CU 쿼드트리의 루트 노드는 일반적으로 트리블록 (또는 LCU) 에 대응한다.

[0041] 상술된 바와 같이, 비디오 인코더 (22) 는 CU 의 코딩 블록을 하나 이상의 예측 블록들로 파티셔닝할 수도 있다. 예측 블록은, 동일한 예측이 적용되는 샘플들의 직사각형 (즉, 정사각형 또는 비정사각형) 블록이다. CU 의 PU 은 루마 샘플들의 예측 블록, 크로마 샘플들의 2 개의 대응하는 예측 블록들, 및 예측 블록들을 예측하기 위하여 이용된 신택스 구조들을 포함할 수도 있다. 단색 화상들 또는 3개의 분리된 색 평면들을 포함하는 화상들에서, PU 는 단일 예측 블록 및 그 예측 블록을 예측하는데 이용된 신택스 구조들을 포함할 수도 있다. 비디오 인코더 (22) 는 CU의 각 PU 의 예측 블록들 (예를 들어, 루마, Cb 및 Cr 예측 블록들) 에 대한 예측 블록들 (예를 들어, 루마, Cb 및 Cr 예측 블록들) 을 생성할 수도 있다.

[0042] 비디오 인코더 (22) 는 PU 에 대해 예측 블록을 생성하는데 인트라 예측 또는 인터 예측을 사용할 수도 있다. 비디오 인코더 (22) 가 인트라 예측을 이용하여 PU 의 예측 블록들을 생성하면, 비디오 인코더 (22) 는 PU 를 포함하는 픽처의 디코딩된 샘플들에 기초하여 PU 의 예측 블록들을 생성할 수도 있다.

[0043] 비디오 인코더 (22) 가 CU 의 하나 이상의 PU들에 대한 예측 블록들 (예컨대, 루마, Cb, 및 Cr 예측 블록들) 을 생성한 후, 비디오 인코더 (22) 는 CU 에 대한 하나 이상의 잔차 블록들을 생성할 수도 있다. 가령, 비디오 인코더 (22) 는 CU 를 위한 루마 잔차 블록을 생성할 수도 있다. CU 의 루마 잔차 블록에 있는 각각의 샘플은 CU 의 예측 루마 블록들 중 하나에 있는 루마 샘플과 CU 의 원래 루마 코딩 블록에 있는 대응하는 샘플 사이의 차이를 표시한다. 또한, 비디오 인코더 (22) 는 CU 를 위한 Cb 잔차 블록을 생성할 수도 있다. 또한, CU 의 Cb 잔차 블록에서의 각각의 샘플은 CU 의 예측 Cb 블록들 중 하나에 있는 Cb 샘플과 CU 의 원래 Cb 코딩 블록에 있는 대응하는 샘플 사이의 차이를 표시할 수도 있다. 또한, 비디오 인코더 (22) 는 CU 를 위한 Cr 잔차 블록을 생성할 수도 있다. 또한, CU 의 Cr 잔차 블록에 있는 각각의 샘플은 CU 의 예측 Cr 블록들 중 하나에 있는 Cr 샘플과 CU 의 원래 Cr 코딩 블록에 있는 대응하는 샘플 사이의 차이를 표시할 수도 있다.

[0044] 또한, 상술된 바와 같이, 비디오 인코더 (22) 는 CU 의 잔차 블록 (예를 들어, 루마, Cb 및 Cr 잔차 블록) 을 하나 이상의 변환 블록 (예를 들어, 루마, Cb 및 Cr 변환 블록) 으로 분해하기 위하여 쿼드 트리 파티셔닝을 이용할 수도 있다. 변환 블록은, 동일한 변환이 적용되는 샘플들의 직사각형 (예를 들어, 정사각형 또는 비정사각형) 블록이다. CU 의 변환 유닛 (TU) 은 루마 샘플들의 변환 블록, 크로마 샘플들의 2개 대응하는 변환 블록들, 및 변환 블록 샘플들을 변환하는데 사용된 신택스 구조들을 포함할 수도 있다. 따라서, CU 의 각 TU 는 루마 변환 블록, Cb 변환 블록, 및 Cr 변환 블록을 가질 수도 있다. TU 의 루마 변환 블록은 CU 의 루마 잔차 블록의 서브 블록일 수도 있다. Cb 변환 블록은 CU 의 Cb 잔차 블록의 서브-블록일 수도 있다. Cr 변환 블록은 CU 의 Cr 잔차 블록의 서브-블록일 수도 있다. 단색 픽처들 또는 3 개의 별개의 컬러 평면들을 갖는 픽처들에 있어서, TU 는 단일의 변환 블록, 및 그 변환 블록의 샘플들을 변환하는데 사용되는 신택스 구조들을 포함할 수도 있다.

[0045] 비디오 인코더 (22) 는 TU 에 대한 변환 블록에 하나 이상의 변환들을 적용하여 TU 에 대한 계수 블록을 생성할 수도 있다. 예를 들어, 비디오 인코더 (22) 는 TU 의 루마 변환 블록에 하나 이상의 변환들을 적용하여 TU 에 대한 루마 계수 블록을 생성할 수도 있다. 계수 블록은 변환 계수들의 2차원 어레이일 수도 있다. 변환 계수는 스칼라 양일 수도 있다. 비디오 인코더 (22) 는 TU 의 Cb 변환 블록에 하나 이상의 변환들을 적용하여 TU 에 대한 Cb 계수 블록을 생성할 수도 있다. 비디오 인코더 (22) 는 TU 의 Cr 변환 블록에 하나 이상의 변환들을 적용하여 TU 에 대한 Cr 계수 블록을 생성할 수도 있다.

[0046] 일부 예들에서, 비디오 인코더 (22) 는 변환 블록에 대한 변환들의 적용을 생략한다. 이러한 예들에서, 비디오 인코더 (22) 는 변환 계수들과 동일한 방식으로 잔차 샘플 값들을 처리할 수도 있다. 따라서, 비디오 인코더 (22) 가 변환들의 적용을 생략하는 예들에서, 변환 계수들 및 계수 블록들의 다음과 같은 논의가 잔차 샘플들의 변환 블록들에 적용가능할 수도 있다.

[0047] 계수 블록 (예를 들어, 휘도 계수 블록, Cb 계수 블록 또는 Cr 계수 블록)을 생성 한 후, 비디오 인코더 (22)는 계수 블록을 양자화하여 계수 블록을 나타내는 데 사용되는 데이터의 양을 감소시켜 잠재적으로 추가 압축을 제공할 수도 있다. 양자화란 일반적으로 값들의 범위를 단일 값으로 압축하는 프로세스를 말한다. 예를 들어, 양자화는 값을 상수로 나눈 다음 가장 가까운 정수로 라운딩하여 수행될 수도 있다. 계수 블록을 양자화하기 위해, 비디오 인코더 (22) 는 계수 블록의 변환 계수들을 양자화할 수도 있다. 비디오 인코더 (22) 가 계수 블록을 양자화한 후, 비디오 인코더 (22) 는 양자화된 변환 계수들을 나타내는 신택스 엘리먼트들을 엔트로피 인코딩할 수도 있다. 예를 들어, 비디오 인코더 (22) 는 양자화된 변환 계수들을 나타내는 신택스 엘리먼트들에 대해 컨텍스트-적응 바이너리 산술 코딩 (CABAC) 또는 다른 엔트로피 코딩 기법들을 수행할 수도 있다.

[0048] 비디오 인코더 (22) 는 코딩된 픽처들 및 관련 데이터의 표현을 형성하는 비트들의 시퀀스를 포함하는 비트스트림을 출력할 수도 있다. 따라서, 비트스트림은 비디오 데이터의 인코딩된 표현을 포함한다. 비트스트림은 네트워크 추상 계층 (network abstraction layer; NAL) 유닛들의 시퀀스를 포함할 수도 있다. NAL 유닛은 NAL 유닛에서의 데이터의 타입 및 그 데이터를 에뮬레이션 방지 비트와 함께 필요에 따라 산재된 RBSP (raw byte sequence payload) 의 형태로 포함하는 바이트의 표시를 포함하는 신택스 구조이다. NAL 유닛들의 각각은 NAL 유닛 헤더를 포함할 수도 있고 RBSP 를 캡슐화할 수도 있다. NAL 유닛 헤더는, NAL 유닛 타입 코드를 표시하는 신택스 엘리먼트를 포함할 수도 있다. NAL 유닛의 NAL 유닛 헤더에 의해 명시되는 NAL 유닛 타입 코드는 NAL 유닛의 타입을 표시한다. RBSP 는 NAL 유닛 내에 캡슐화되는 정수 개의 바이트들을 포함하는 신택스 구조일 수도 있다. 일부 사례들에서, RBSP 는 제로 비트들을 포함한다.

[0049] 비디오 디코더 (30) 는 비디오 인코더 (22) 에 의해 생성된 비트스트림을 수신할 수도 있다. 비디오 디코더 (30)는 비트 스트림을 디코딩하여 비디오 데이터의 픽처들을 재구성할 수도 있다. 비트스트림을 디코딩하는 것의 부분으로서, 비디오 디코더 (30) 는 비트스트림을 파싱하여, 그 비트스트림으로부터 신택스 엘리먼트들을 획득할 수도 있다. 비디오 디코더 (30) 는 비트스트림으로부터 획득된 신택스 엘리먼트들에 적어도 부분적으로 기초하여 비디오 데이터의 픽처들을 재구성할 수도 있다. 비디오 데이터를 재구성하기 위한 프로세스는 일반적으로, 비디오 인코더 (22) 에 의해 수행된 프로세스에 역일 수도 있다. 실례로, 비디오 디코더 (30) 는 PU들의 모션 벡터들을 이용하여 현재 CU 의 PU 들에 대한 예측 블록들을 결정할 수도 있다. 또한, 비디오 디코더 (30) 는 현재 CU 의 TU들의 계수 블록들을 역양자화할 수도 있다. 비디오 디코더 (30) 는 계수 블록들에 대해 역변환들을 수행하여 현재 CU 의 TU들의 변환 블록들을 재구성할 수도 있다. 비디오 디코더 (30) 는 현재 CU 의 PU 들에 대한 예측성 블록들의 샘플들을, 현재 CU 의 TU 들의 변환 블록들의 대응하는 샘플들에 부가함으로써, 현재 CU 의 코딩 블록들을 재구성할 수도 있다. 픽처의 각각의 CU 에 대한 코딩 블록들을 재구성함으로써, 비디오 디코더 (30) 는 픽처를 재구성할 수도 있다.

[0050] HEVC 의 일반적인 개념과 특정 디자인 양태들은 블록 파티션을 위한 기법들에 초점을 두고 아래에서 설명한다. HEVC 에서 슬라이스의 최대 코딩 유닛은 CTB 으로 지칭된다. CTB 는 쿼드-트리 구조에 따라 분할되며, 그 노드들은 코딩 유닛들이다. 쿼드 트리 구조의 복수의 노드는 리프 노드 및 비 리프 노드를 포함한다. 리프 노드는 트리 구조에 차일드 노드를 갖지 않는다 (즉, 리프 노드는 더 이상 분할되지 않는다). 비 리프 노드는 트리 구조의 루트 노드를 포함한다. 루트 노드는 비디오 데이터의 초기 비디오 블록 (예를 들어, CTB) 에 대응한다. 복수의 노드 중 각각의 별개의 비 루트 노드에 대해, 각각의 비 루트 노드는 각각의 비 루트 노드의 트리 구조에서의 부모 노드에 대응하는 비디오 블록의 서브 블록인 비디오 블록에 대응한다. 복수의 비 리프 노드들의 각각의 별개의 비 리프 노드는 트리 구조에서 하나 이상의 차일드 노드들을 갖는다.

[0051] CTB 의 사이즈는 (비록 기술적으로 8x8 CTB 사이즈들이 지원될 수 있음에도 불구하고) HEVC 메인 프로파일에서 16x16 에서부터 64x64 까지의 범위이다. CTB 는 W. J. Han et al, "Improved Video Compression Efficiency Through Flexible Unit Representation and Corresponding Extension of Coding Tools", IEEE Transaction on Circuits and Systems for Video Technology, vol. 20, no. 12, pp. 1709-1720, Dec. 2010 에 기술되고, 도 2 에 도시된 바와 같은 쿼드 트리 방식으로 CU 들로 재귀적으로 분할될 수도 있다. 도 2 에 도시된 바와 같이, 파티셔닝의 각 레벨은 4 개의 서브 블록으로 분할된 쿼드 트리이다. 검정색 블록은 리프 노드 (즉, 더 이상 분할되지 않은 블록) 의 예이다.

[0052] 몇몇 예들에서, CU는 8x8만큼 작을 수 있지만, CU는 CTB 의 동일한 크기일 수도 있다. 각 CU 는 예를 들어 인트라 코딩 모드 또는 인터 코딩 모드일 수 있는 하나의 코딩 모드로 코딩된다. 스크린 컨텐츠에 대한 코딩 모드들 (예를 들어, 인트라 블록 복사 모드, 팔레트 기반 코딩 모드 등) 을 포함하는 다른 코딩 모드도 가능하다. CU 가 인터 코딩될 때 (즉, 인터 모드가 적용될 때), CU 는 예측 유닛 (PU) 들로 더 파티셔닝될 수도 있다. 예를 들어, CU는 2 또는 4 개의 PU 로 파티셔닝될 수도 있다. 다른 예에서, 추가의 파티셔닝이 적용되지 않을 때 전체 CU 는 단일 PU 로 취급된다. HEVC 예들에서, 2개의 PU들이 하나의 CU 에 존재할 경우, 그 PU들은 하프 사이즈 직사각형들 또는 CU 의 ¼ 또는 ¾ 사이즈인 2개의 직사각형 사이즈일 수 있다.

[0053] HEVC 에서, 도 3 에 도시된 바와 같이, 인터 예측 모드로 코딩된 CU 를 위한 8 개의 파티션 모드들, 즉, PART_2Nx2N, PART_2NxN, PART_Nx2N, PART_NxN, PART_2NxnU, PART_2NxnD, PART_nLx2N 및 PART_nRx2N 이 존재한다. 도 3 에 도시된 바와 같이, 파티션 모드 PART_2Nx2N 으로 코딩된 CU 는 더 이상 분할되지 않는다. 즉, 전체 CU는 단일 PU (PU0)로 취급된다. 파티션 모드 PART_2NxN으로 코딩된 CU는 수평으로 대칭으로 두 개의 PU (PU0 및 PU1)로 분할된다. 파티션 모드 PART_Nx2N으로 코딩된 CU는 수평으로 대칭으로 두 개의 PU 로 분할된다. 파티션 모드 PART_NxN으로 코딩된 CU는 4 개의 동일한 크기의 PU (PU0, PU1, PU2, PU3)로 대칭적으

로 분할된다.

[0054] 파티션 모드 PART_2NxnU로 코딩된 CU는 CU의 크기의 1/4 인 하나의 PU0 (상측 PU) 과 CU 의 크기의 ¾ 인 PU1 (하측 PU)으로 비대칭적으로 수평으로 분할된다. 파티션 모드 PART_2NxnD로 코딩된 CU는 CU의 크기의 3/4 인 하나의 PU0 (상측 PU) 과 CU 의 크기의 1/4 인 하나의 PU1 (하측 PU)으로 비대칭적으로 수평으로 분할된다. 파티션 모드 PART_nLx2N으로 코딩된 CU는 CU의 크기의 1/4 인 하나의 PU0 (좌측 PU)과 CU의 크기의 3/4 인 하나의 PU1 (우측 PU)으로 비대칭적으로 수직으로 분할된다. 파티션 모드 PART_nRx2N으로 코딩된 CU는 CU의 크기의 3/4 인 하나의 PU0 (좌측 PU)과 CU의 크기의 1/4 인 하나의 PU1 (우측 PU)으로 비대칭적으로 수직으로 분할된다.

[0055] CU가 인터 코딩될 때, 각각의 PU에 대해 한 세트의 모션 정보 (예를 들어, 모션 벡터, 예측 방향 및 참조 픽처)가 존재한다. 부가적으로, 각각의 PU 는 모션 정보의 세트를 도출하기 위해 고유한 인터-예측 모드로 코딩된다. 그러나, 비록 2 개의 PU가 고유하게 코딩되더라도, 그들은 어떤 환경에서는 여전히 동일한 모션 정보를 가질 수도 있음을 이해해야한다.

[0056] J. An et al., “Block partitioning structure for next generation video coding”, International Telecommunication Union, COM16-C966, Sep. 2015 (hereinafter, “VCEG proposal COM16-C966”) 에서, 쿼드-트리-이진-트리 (QTBT) 파티셔닝 기법들은 HEVC 이상의 미래 비디오 코딩 표준을 위해 제안되었다. 제안된 QTBT 구조가 사용된 HEVC 에서의 쿼드트리 구조보다 더 효율적이라는 것을 시뮬레이션들이 보여주었다.

[0057] VCEG 제안 COM16-C966 의 제안된 QTBT 구조에서, CTB 는 먼저 쿼드트리 파티셔닝 기술들을 사용하여 파티셔닝되며, 여기서 일 노드의 쿼드트리 분할은 노드가 최소 허용된 쿼드트리 리프 노드 사이즈에 도달할 때까지 반복될 수 있다. 최소 허용된 쿼드트리 리프 노드 사이즈는 신택스 엘리먼트 MinQTSize 의 값으로 비디오 디코더에 표시될 수도 있다. 쿼드트리 리프 노드 사이즈가 최대 허용된 이진 트리 루트 노드 사이즈 (예컨대, 신택스 엘리먼트 MaxBTSize 로 표시됨) 보다 크지 않은 경우, 쿼드트리 리프 노드는 이진 트리 파티셔닝을 사용하여 추가로 파티셔닝될 수 있다. 일 노드의 이진 트리 파티셔닝은 노드가 최소 허용된 이진 트리 리프 노드 사이즈 (예컨대, 신택스 엘리먼트 MinBTSize 로 표시됨) 또는 최대 허용된 이진 트리 깊이 (예컨대, 신택스 엘리먼트 MaxBTDepth 로 표시됨) 에 도달할 때까지 반복될 수 있다. VCEG 제안 COM16-C966 은 이진 트리 리프 노드들을 지칭하기 위해 "CU" 라는 용어를 사용한다. VCEG 제안 COM16-C966 에서, CU 는 예측 (예컨대, 인트라 예측, 인터 예측 등) 에 사용되며, 더 이상의 파티셔닝 없이 변환한다. 일반적으로, QTBT 기술들에 따르면, 이진 트리 분할에 대해 2 개의 분할 타입들, 즉 대칭적 수평 분할과 대칭적 수직 분할이 있다. 각각의 경우에, 블럭은 그 블럭을 수평으로 또는 수직으로 절반으로 나누어 분할하도록 분할된다.

[0058] QTBT 파티셔닝 구조의 일 예에서, CTU 사이즈는 128x128 (예를 들어, 128x128 루마 블록 및 2 개의 대응하는 64x64 크로마 블록들) 로 설정되고, MinQTSize 는 16x16으로 설정되며, MaxBTSize 는 64x64 로 설정되고, (폭 및 높이 양자에 대한) MinBTSize 는 4 로 설정되고, MaxBTDepth 가 4 로 설정된다. 쿼드트리 파티셔닝은 CTU 에 먼저 적용되어 쿼드트리 리프 노드들을 생성한다. 쿼드트리 리프 노드들은 16x16 (즉, MinQTSize 가 16x16) 부터 128x128 (즉, CTU 사이즈) 까지의 사이즈를 가질 수도 있다. QTBT 파티셔닝의 일 예에 따르면, 리프 쿼트트리 노드가 128x128 인 경우, 리프 쿼트트리 노드의 크기가 MaxBTSize (즉, 64x64) 를 초과하기 때문에, 리프 쿼트트리 노드는 이진 트리에 의해 더 이상 분할될 수 없다. 그렇지 않으면, 리프 쿼드트리 노드는 이진 트리에 의해 추가로 파티셔닝된다. 따라서, 쿼드트리 리프 노드는 또한 이진 트리의 루트 노드이며, 이진 트리 깊이를 0 으로 갖는다. MaxBTDepth 에 도달하는 이진 트리 깊이 (예컨대, 4) 는 더 이상의 분할이 없음을 의미한다. MinBTSize (예컨대, 4) 와 동일한 폭을 갖는 이진 트리 노드는 더 이상의수평 분할이 없음을 의미한다. 마찬가지로, 높이가 MinBTSize 와 동일한 이진 트리 노드는 더 이상 수직 분할이 없음을 의미한다. 이진 트리 (CU들) 의 리프 노드들은 어떤 추가의 파티셔닝 없이 (예컨대, 예측 프로세스 및 변환 프로세스를 수행함으로써) 추가로 프로세싱된다.

[0059] 도 4a 는 QTBT 파티셔닝 기술들을 사용하여 파티셔닝된 블록 (50) (예를 들어, CTB) 의 일 예를 도시한다. 도 4a 에 도시된 바와 같이, QTBT 파티션 기술들을 사용하여, 결과 블록들의 각각은 각 블록의 중심을 통해 대칭적으로 분할된다. 도 4b 는 도 4a 의 블록 파티셔닝에 대응하는 트리 구조를 도시한다. 도 4b 의 실선은, 쿼드 트리 분할을 나타내고 점선은 이진 트리 분할을 나타낸다. 일 예에서, 이진 트리의 각 분할 (즉, 비-리프) 노드에서, 수행된 분할의 타입 (예를 들어, 수평 또는 수직) 을 나타내기 위해 신택스 엘리먼트 (예를 들어, 플래그) 가 시그널링되고, 여기서 0 은 수평 분할을 나타내고, 1 은 수직 분할을 나타낸다. 쿼드트리 분할의 경우, 분할 타입을 나타낼 필요가 없는데, 쿼드트리 분할은 항상 블록을 동일한 사이즈의 4 개의 서브블록들로 수

평 및 수직으로 분할하기 때문이다.

[0060] 도 4b 에 도시된 바와 같이, 노드 (70) 에서, 블록 (50) 은 QT 파티셔닝을 사용하여 도 4a 에 도시된 4 개의 블록들 (51, 52, 53 및 54) 로 분할된다. 블록 (54) 은 더 이상 분할되지 않으며, 따라서 리프 노드이다. 노드 (72) 에서, 블록 (51) 은 BT 파티셔닝을 사용하여 2 개의 블록들로 더 분할된다. 도 4b 에 도시된 바와 같이, 노드 (72) 는 수직 분할을 나타내는 1 로 마킹된다. 이와 같이, 노드 (72) 에서의 분할은 블록 (57) 을 초래하고, 블록은 블록들 (55 및 56) 을 포함한다. 블록들 (55 및 56) 은 노드 (74) 에서의 추가의 수직 분할에 의해 생성된다. 노드 (76) 에서, 블록 (52) 은 BT 파티셔닝을 사용하여 2 개의 블록들 (58 및 59) 로 더 분할된다. 도 4b 에 도시된 바와 같이, 노드 (76) 는 수평 분할을 나타내는 1 로 마킹된다.

[0061] 노드 (78) 에서, 블록 (53) 은 QT 파티셔닝을 사용하여 4 개의 동일한 사이즈 블록들로 분할된다. 블록들 (63 및 66) 은 상기 QT 파티셔닝으로부터 생성되고, 추가로 분할되지 않는다. 노드 (80) 에서, 상부 좌측 블록은 먼저 수직 이진 트리 분할을 사용하여 분할되어 블록 (60) 및 우측 수직 블록을 초래한다. 그 다음, 우측 수직 블록은 수평 이진 트리 분할을 사용하여 블록들 (61 및 62) 로 분할된다. 노드 (78) 에서의 쿼드트리 분할로부터 생성된 하부 우측 블록은 블록들 (64 및 65) 로의 수평 이진 트리 분할을 사용하여 노드 (84) 에서 분할된다.

[0062] 전술한 QTBT 구조는 HEVC에서 사용되는 쿼드트리 구조보다 우수한 코딩 성능을 나타내지만, QTBT 구조는 유연성이 부족하다. 예를 들어, 위에서 설명한 QTBT 구조에서, 쿼드 트리 노드는 이진 트리로 더 분할될 수 있지만, 이진 트리 노드는 쿼드 트리로 더 이상 분할될 수 없다. 다른 예에서, 쿼드 트리 및 이진 트리는 모두 균등 분할 (즉, 블록 중심을 분할) 만을 달성할 수 있으며, 이는 분할될 블록의 중심에 오브젝트가 있을 때 효율적이지 못하다. 따라서 QTBT의 코딩 성능은 미래 비디오 코딩 표준에 적합하지 않을 수 있다.

[0063] 위에서 언급 된 문제를 해결하기 위해, 본원에 참고로 포함된 2017 년 1 월 12 일에 출원된 미국 특허 공보 제 20170208336 호 및 2017 년 3 월 20 일자로 출원된 미국 특허 공보 제 20170272782 호는 다중 유형 트리 (MTT) 파티션 구조의 여러 예들을 기술한다. MTT 파티션 구조에 따르면, 트리 노드는 이진 트리, 대칭 중앙측 삼중 트리 및 쿼드 트리와 같은 여러 트리 유형으로 더 분할될 수도 있다. 시뮬레이션들은 다중 트리 구조는 쿼드 트리 이진 트리 구조보다 훨씬 더 효율적이라는 것을 보여주었다.

[0064] CTU 에 대해 보다 유연한 파티셔닝을 보다 잘 달성하기 위해, QT, BT 및/또는 QTBT 기반 CU 구조를 대체하기 위해 MTT 기반 CU 구조가 제안되었다. 본 개시의 MTT 파티셔닝 구조는 여전히 재귀적 트리 구조이다. 그러나 다수의 서로 다른 파티션 구조 (예를 들어, 3 개 이상) 가 사용된다. 예를 들어, 본 개시의 MTT 기술에 따르면, 트리 구조의 각각의 깊이에서 3 개 이상의 상이한 파티션 구조가 사용될 수도 있다. 이러한 문맥에서, 트리 구조의 노드 깊이는 노드에서 트리 구조의 루트까지의 경로 길이 (예를 들어, 분할의 수) 를 지칭할 수도 있다.

[0065] 본 개시의 기술에 따른 일 예시에서, 비디오 인코더 (22) 및/또는 비디오 디코더 (30)는 비디오 데이터의 픽쳐를 수신하고, 3 개 이상의 상이한 파티션 구조을 사용하여 비디오 데이터의 픽쳐를 복수의 블록으로 파티셔닝하고, 비디오 데이터의 픽쳐의 복수의 블록을 재구성/인코딩하도록 구성될 수도 있다. 일 예에서, 비디오 데이터의 픽쳐를 파티셔닝하는 것은 3 개 이상의 상이한 파티션 구조를 사용하여 비디오 데이터의 픽쳐를 복수의 블록으로 파티셔닝하는 것을 포함하며, 3 개 이상의 상이한 파티션 구조 중 적어도 3 개가 비디오 데이터의 픽쳐가 파티셔닝되는 방법을 나타내는 트리 구조의 각각의 깊이에서 사용될 수도 있다. 일 예에서, 3 개 이상의 상이한 파티션 구조는 3 중 트리 파티션 구조를 포함하고, 비디오 인코더 (22) 및/또는 비디오 디코더 (30)는 3 중 트리 파티션 유형의 3 중 트리 파티션 구조를 사용하여 비디오 데이터의 복수의 블록 중 하나를 파티셔닝하도록 구성될 수도 있으며, 여기서 3 중 트리 파티션 구조는 복수의 블록 중 하나의 블록을 중심을 통해 분할하지 않고 3 개의 서브 블록으로 분할한다. 본 개시의 또 다른 예에서, 3 개 이상의 상이한 파티션 구조는 쿼드 트리 (quad-tree) 파티션 구조 및 2 진 트리 파티션 구조를 더 포함한다.

[0066] 따라서, 일 예시에서, 비디오 인코더 (22)는 비디오 데이터의 초기 비디오 블록 (예를 들어, 코딩 트리 블록 또는 CTU) 의 인코딩된 표현을 생성할 수도 있다. 초기 비디오 블록의 인코딩된 표현을 생성하는 것의 부분으로서, 비디오 인코더 (22) 는 복수의 노드를 포함하는 트리 구조를 결정한다. 예를 들어, 비디오 인코더 (22) 는 본 개시의 MTT 파티셔닝 구조를 사용하여 트리 블록을 파티셔닝할 수도 있다.

[0067] MTT 파티셔닝 구조의 복수의 노드는 복수의 리프 노드 및 복수의 비 리프 노드를 포함한다. 리프 노드는 트리 구조에 차일드 노드를 갖지 않는다. 비 리프 노드는 트리 구조의 루트 노드를 포함한다. 루트 노드는 초기 비디오 블록에 대응한다. 복수의 노드 중 각각의 별개의 비 루트 노드에 대해, 각각의 비 루트 노드는 각각의 비

루트 노드의 트리 구조에서의 부모 노드에 대응하는 비디오 블록의 서브 블록인 비디오 블록 (예를 들어, 코딩 블록) 에 대응한다. 복수의 비 리프 노드들의 각각의 별개의 비 리프 노드는 트리 구조에서 하나 이상의 차일드 노드들을 갖는다. 일부 예에서, 픽쳐 경계에서의 비 리프 노드는 강제된 분할로 인해 하나의 차일드 노드만을 가질 수 있고, 차일드 노드들 중 하나는 픽쳐 경계 외부의 블록에 대응한다.

[0068] 본 개시의 기술에 따르면, 트리 구조의 각각의 깊이 레벨에서의 트리 구조의 각각의 별개의 비 리프 노드에 대해, 각각의 비 리프 노드에 대해 복수의 허용된 분할 패턴 (예를 들어, 파티션 구조) 이 존재한다. 예를 들어, 트리 구조의 각 깊이에 허용되는 3 개 이상의 파티션 구조가 존재할 수도 있다. 비디오 인코더 (22)는 복수의 허용 가능한 파티션 구조 중 하나에 따라 각각의 비 리프 노드에 대응하는 비디오 블록을 각각의 비 리프 노드의 차일드 노드에 대응하는 비디오 블록으로 파티셔닝하도록 구성될 수도 있다. 복수의 허용된 파티션 구조들의 각각의 별개의 허용된 파티션 구조는 각각의 비 리프 노드에 대응하는 비디오 블록을 각각의 비 리프 노드의 차일드 노드들에 대응하는 비디오 블록들로 파티셔닝하는 상이한 방식에 대응할 수도 있다. 또한, 이 예에서, 비디오 인코더 (22) 는 비디오 데이터의 인코딩된 표현을 포함하는 비트 스트림에 초기 비디오 블록의 인코딩된 표현을 포함할 수도 있다.

[0069] 유사한 예에서, 비디오 디코더 (30)는 복수의 노드를 포함하는 트리 구조를 결정할 수도 있다. 이전의 예에서와 같이, 복수의 노드는 복수의 리프 노드 및 복수의 비 리프 노드를 포함한다. 리프 노드는 트리 구조에 차일드 노드를 갖지 않는다. 비 리프 노드는 트리 구조의 루트 노드를 포함한다. 루트 노드는 비디오 데이터의 초기 비디오 블록에 대응한다. 복수의 노드 중 각각의 별개의 비 루트 노드에 대해, 각각의 비 루트 노드는 각각의 비 루트 노드의 트리 구조에서의 부모 노드에 대응하는 비디오 블록의 서브 블록인 비디오 블록에 대응한다. 복수의 비 리프 노드들의 각각의 별개의 비 리프 노드는 트리 구조에서 하나 이상의 차일드 노드들을 갖는다. 트리 구조의 각 깊이 레벨에서의 트리 구조의 각각의 개별 비-리프 노드에 대해, 개별 비-리프 노드에 대한 복수의 허용된 분할 패턴들이 존재하고, 개별 비-리프 노드에 대응하는 비디오 블록은 복수의 허용 가능한 분할 패턴들 중 하나에 따라 개별 비-리프 노드의 자식 노드들에 대응하는 비디오 블록들로 파티셔닝된다. 복수의 허용된 분할 패턴들의 각각의 별개의 허용된 분할 패턴은 각각의 비 리프 노드에 대응하는 비디오 블록을 각각의 비 리프 노드의 차일드 노드들에 대응하는 비디오 블록들로 파티셔닝하는 상이한 방식에 대응한다. 또한,이 예에서, 트리 구조의 각각의 (또는 적어도 하나의) 별개의 리프 노드 대해, 비디오 디코더 (30)는 각각의 리프 노드에 대응하는 비디오 블록을 재구성한다.

[0070] 일부 이러한 예에서, 루트 노드 이외의 트리 구조의 각각의 별개의 비 리프 노드에 대해, 각각의 비 리프 노드에 대한 복수의 허용된 분할 패턴 (예를 들어, 파티션 구조) 은 각각의 비 리프 노드의 부모 노드에 대응하는 비디오 블록이 각각의 비 리프 노드의 부모 노드의 차일드 노드에 대응하는 비디오 블록으로 분할되는 파티션 구조에 대해 독립적이다.

[0071] 본 개시의 다른 예에서, 트리 구조의 각각의 깊이에서, 비디오 인코더 (22)는 3 개의 더 많은 파티셔닝 구조들 중 하나의 파티셔닝 구조 중 특정 파티션 유형을 사용하여 서브 트리들을 더 분할하도록 구성될 수도 있다. 예를 들어, 비디오 인코더 (22)는 QT, BT, 삼중 트리 (TT) 및 다른 파티셔닝 구조로부터 특정 파티션 유형을 결정하도록 구성될 수도 있다. 일 예에서, QT 파티셔닝 구조는 정방형 쿼드 트리 및 직사각형 쿼드 트리 파티셔닝 유형을 포함할 수도 있다. 비디오 인코더 (22)는 중심을 수평 및 수직으로 4 개의 동일한 크기의 정사각형 블록으로 분할함으로써 정사각형 쿼드 트리 파티셔닝을 사용하여 정사각형 블록을 파티셔닝할 수도 있다. 마찬가지로, 비디오 인코더 (22) 는 중심을 수평 및 수직으로 4 개의 동일한 크기의 직사각형 블록으로 분할함으로써 직사각형 쿼드 트리 파티션을 사용하여 직사각형 (예를 들어, 비정사각형) 블록을 파티셔닝할 수도 있다.

[0072] BT 파티셔닝 구조는 수평 대칭 2 진 트리, 수직 대칭 2 진 트리, 수평 비대칭 2 진 트리 및 수직 비대칭 2 진 트리 파티션 유형을 포함할 수도 있다. 수평 대칭 2 진 트리 파티션 유형의 경우, 비디오 인코더 (22)는, 블록의 중심을 지나 수평으로, 블록을 동일한 크기의 2 개의 대칭 블록으로 분할하도록 구성될 수도 있다. 수직 대칭 2 진 트리 파티션 유형의 경우, 비디오 인코더 (22)는, 블록의 중심을 지나 수직으로, 블록을 동일한 크기의 2 개의 대칭 블록으로 분할하도록 구성될 수도 있다. 수평 비대칭 2 진 트리 파티션 유형의 경우, 비디오 인코더 (22)는 수평으로 블록을 상이한 크기의 2 개의 블록으로 분할하도록 구성될 수도 있다. 예를 들어, 도 3 의 PART_2NxnU 또는 PART_2NxnD 파티션 유형에서와 같이, 하나의 블록은 부모 블록의 크기의 1/4 일 수 있고, 다른 블록은 부모 블록의 크기의 3/4 일 수도 있다. 수직 비대칭 2 진 트리 파티션 유형의 경우, 비디오 인코더 (22)는 수직으로 블록을 상이한 크기의 2 개의 블록으로 분할하도록 구성될 수도 있다. 예를 들어, 도 3 의 PART_nLx2N 또는 PART_nRx2N 파티션 유형에서와 같이, 하나의 블록은 부모 블록의 크기의 1/4 일 수 있고, 다른

블록은 부모 블록의 크기의 3/4 일 수도 있다.

[0073] 다른 예들에서, 비대칭 2 진 트리 파티션 유형은 부모 블록을 상이한 크기 프랙션들로 분할할 수도 있다. 예를 들어, 하나의 서브 블록은 부모 블록의 3/8 일 수 있고 다른 하나의 서브 블록은 부모 블록의 5/8 일 수도 있다. 물론, 그러한 파티션 유형은 수직 또는 수평일 수도 있다.

[0074] TT 파티션 구조는 TT 파티션 구조가 블록을 중심을 지나 분할하지 않는다는 점에서, QT 또는 BT 구조의 구조와 상이하다. 블록의 중심 영역은 동일한 서브 블록에 함께 유지된다. 4 개의 블록을 생성하는 QT, 또는 2 개의 블록을 생성하는 이진 트리와는 달리, TT 파티션 구조에 따른 분할은 세 개의 블록을 생성한다. TT 파티션 구조에 따른 예시의 파티션 유형들은 수평 대칭 삼중 트리, 수직 대칭 삼중 트리, 수평 비대칭 삼중 트리 및 수직 비대칭 삼중 트리 파티션 유형들을 포함한다. 수평 대칭 삼중 트리 파티션 유형의 경우, 비디오 인코더 (22)는 블록을 중심을 지나 분할하지 않고 수평으로 블록을 3 개의 블록으로 분할하도록 구성될 수도 있다. 수평 대칭 3 중 트리 파티셔닝에 따라 분할될 때, 중심 서브 블록의 위와 아래의 블록들이 미러링되며, 즉, 그들은 동일한 크기이다. 블록이 3 으로 직접 분할 가능한 경우 (예를 들어, 12 개의 샘플 높이), 중심 블록은 상측 블록 및 하측 블록과 같은 크기일 수도 있다. 블록이 3 으로 직접 분할가능하지 않은 경우 (예를 들어, 8 개의 샘플 높이), 중심 블록은 상측 블록 및 하측 블록과 상이한 크기일 수도 있다. 예를 들어, 8 개 샘플 높이의 블록의 경우, 상측 및 하측 블록은 3 개의 샘플 높이일 수도 있으며 중심 블록은 2 개의 샘플 높이일 수 있다. 다른 예에서, 8 개 샘플 높이의 블록의 경우, 상측 및 하측 블록은 2 개의 샘플 높이일 수도 있으며 중심 블록은 4 개의 샘플 높이일 수 있다. 도 5e 는 수평 삼중 트리 파티셔닝의 예를 도시한다.

[0075] 수직 대칭 삼중 트리 파티션 유형의 경우, 비디오 인코더 (22)는 블록을 중심을 지나 분할하지 않고 수직으로 블록을 3 개의 블록으로 분할하도록 구성될 수도 있다. 수직 대칭 3 중 트리 파티셔닝에 따라 분할될 때, 중심 서브 블록의 왼쪽과 오른쪽의 블록들이 미러링되며, 즉, 그들은 동일한 크기이다. 블록이 3 으로 직접 분할 가능한 경우 (예를 들어, 12 개의 샘플 너비), 중심 블록은 좌측 블록 및 우측 블록과 같은 크기일 수도 있다. 블록이 3 으로 직접 분할가능하지 않은 경우 (예를 들어, 8 개의 샘플 너비), 중심 블록은 좌측 블록 및 우측 블록과 상이한 크기일 수도 있다. 예를 들어, 8 개 샘플 너비의 블록의 경우, 좌측 및 우측 블록은 3 개의 샘플 너비일 수도 있으며 중심 블록은 2 개의 샘플 너비일 수 있다. 다른 예에서, 8 개 샘플 너비의 블록의 경우, 좌측 및 우측 블록은 2 개의 샘플 너비일 수도 있으며 중심 블록은 4 개의 샘플 너비일 수 있다. 도 5 d 는 수직 삼중 트리 파티셔닝의 예를 도시한다.

[0076] 수평 비대칭 삼중 트리 파티션 유형의 경우, 비디오 인코더 (22)는 수평으로 블록을, 대칭적으로 미러링되지 않는 3 개의 블록으로 분할하도록 구성될 수도 있다. 일부 예들에서, 수평 비대칭 삼중 트리 분할 유형은 블록을 중심을 지나 분할 할 수도 있고, 다른 예에서는 수평 비대칭 삼중 트리 파티션 유형이 블록을 중심을 지나 분할하지 않을 수도 있다. 수직 비대칭 삼중 트리 파티션 유형의 경우, 비디오 인코더 (22)는 수직으로 블록을, 대칭적으로 미러링되지 않는 3 개의 블록으로 분할하도록 구성될 수도 있다. 일부 예들에서, 수직 비대칭 삼중 트리 분할 유형은 블록을 중심을 지나 분할 할 수도 있고, 다른 예에서는 수직 비대칭 삼중 트리 파티션 유형이 블록을 중심을 지나 분할하지 않을 수도 있다.

[0077] (예를 들어, 서브 트리 노드에서의) 블록이 비대칭 삼중 트리 파티션 유형으로 분할되는 예에서, 비디오 인코더 (22) 및/또는 비디오 디코더 (30)는 3 개의 파티션 중 2 개가 같은 크기를 갖도록 하는 제한을 적용할 수도 있다. 이러한 제한은 비디오 데이터를 인코딩할 때 비디오 인코더 (22)가 준수해야하는 제한에 대응할 수도 있다. 또한, 일부 예들에서, 비디오 인코더 (22) 및 비디오 디코더 (30)는 비대칭 삼중 트리 파티션 유형에 따라 분할할 때 2 개의 파티션들의 면적의 합이 나머지 파티션의 면적과 동일한 제한을 적용할 수도 있다. 예를 들어, 트리 구조의 노드에 대응하는 비디오 블록이 비대칭 삼중 트리 패턴에 따라 파티셔닝될 때, 그 노드는 제 1 차일드 노드, 제 2 차일드 노드 및 제 3 차일드 노드를 가지며, 제 2 차일드 노드는 제 1 및 제 3 차일드 노드들에 대응하는 비디오 블록들 사이의 비디오 블록에 대응하고, 제 1 및 제 3 차일드 노드들에 대응하는 비디오 블록들의 크기의 합은 제 2 차일드 노드에 대응하는 비디오 블록의 크기와 동일하다고 특정하는 제한에 따르는 초기 비디오 블록의 인코딩된 표현을 수신할 수도 있다.

[0078] 본 개시의 일부 예들에서, 비디오 인코더 (22)는 QT, BT 및 TT 파티션 구조 각각에 대해 전술한 모든 파티션 유형 중에서 선택하도록 구성될 수도 있다. 다른 예들에서, 비디오 인코더 (22) 는 전술한 파티션 유형들의 서브 세트 중에서 단지 파티션 유형을 결정하도록 구성될 수도 있다. 예를 들어, 위에서 언급된 파티션 유형 (또는 다른 파티션 유형)의 서브 세트는 특정 블록 크기 또는 쿼드 트리 구조의 특정 깊이에 대해 사용될 수도 있다. 지원되는 파티션 유형들의 서브 세트는 비디오 디코더 (30) 에 의해 사용하기 위해 비트 스트림에서 시그널링될

수도 있거나, 비디오 인코더 (22) 및 비디오 디코더 (30)가 어떠한 시그널링도 없이 서브 세트를 결정할 수 있도록 미리 정의될 수도 있다.

[0079] 다른 예들에서, 지원되는 파티셔닝 유형의 수는 모든 CTU 에서의 모든 깊이에 대해 고정될 수도 있다. 즉, 비디오 인코더 (22) 및 비디오 디코더 (30) 는 CTU 의 임의의 깊이에 대해 동일한 수의 파티셔닝 유형을 사용하도록 사전 구성될 수도 있다. 다른 예들에서, 지원되는 파티셔닝 유형들의 수는 다양할 수 있고, 깊이, 슬라이스 유형, 또는 다른 이전에 코딩된 정보에 의존할 수도 있다. 일 예에서, 트리 구조의 깊이 0 또는 깊이 1 에서, QT 파티션 구조 만이 사용된다. 깊이가 1 보다 크면, QT, BT 및 TT 파티션 구조들 각각이 사용될 수도 있다.

[0080] 일부 예에서, 비디오 인코더 (22) 및/또는 비디오 디코더 (30)는 비디오 프레임의 특정 영역 또는 CTU 의 영역에 대한 중복된 파티셔닝을 피하기 위해 지원되는 파티셔닝 유형에 대해 사전 구성된 제약을 적용할 수도 있다. 일 예시에서, 블록이 비대칭 파티션 타입으로 분할될 때, 비디오 인코더 (22) 및/또는 비디오 디코더 (30) 는 현재 블록으로부터 분할된 최대 서브 블록을 더 이상 분할하지 않도록 구성될 수도 있다. 예를 들어, 정사각형 블록이 비대칭 파티션 타입 (예를 들어, 도 3 의 PART_2NxnU 파티션 유형) 에 따라 분할될 때, 모든 서브 블록들 중 가장 큰 서브 블록 (예를 들어, 도 3 의 PART_2NxnU 파티션 유형의 PU1) 은 주목할만한 리프 노드이며 더 이상 분할될 수 없다. 그러나, 더 작은 서브 블록 (예를 들어, 도 3 의 PART_2NxnU 파티션 유형의 PU0) 은 더 분할될 수 있다.

[0081] 특정 영역에 대해 중복된 파티셔닝을 피하기 위해 지원되는 파티셔닝 유형에 대한 제약들이 적용될 수도 있는 다른 예로서, 블록이 비대칭 파티션 유형으로 분할될 때, 현재 블록에서 분할되는 가장 큰 서브 블록은 더 이상 동일한 방향에서 분할될 수 없다. 예를 들어, 정사각형 블록이 비대칭 파티션 타입 (예를 들어, 도 3 의 PART_2NxnU 파티션 유형) 에 따라 분할될 때, 비디오 인코더 (22) 및/또는 비디오 디코더 (30)는 모든 서브 블록들 중에서 큰 서브 블록 (예를 들어, 도 3 의 PART_2NxnU 파티션 타입의 PU1) 을 수평 방향으로 분할하지 않도록 구성될 수도 있다. 그러나, 이러한 예에서, 비디오 인코더 (22) 및/또는 비디오 디코더 (30)는 수직 방향으로 다시 PU1을 분할할 수도 있다.

[0082] 지원된 파티셔닝 유형에 대한 제약이 추가 분할의 어려움을 피하기 위해 적용될 수도 있는 다른 예로서, 비디오 인코더 (22) 및/또는 비디오 디코더 (30) 는 블록의 폭/높이가 2 의 거듭제곱이 아닐 때 (예를 들어, 폭/높이가 2, 4, 8, 16 등이 아닌 경우) 수평 또는 수직으로 블록을 분할하지 않도록 구성될 수도 있다.

[0083] 상기 예들은 비디오 인코더 (22) 가 본 개시의 기법들에 따라 MTT 파티셔닝을 수행하도록 어떻게 구성될 수 있는지를 설명한다. 또한, 비디오 디코더 (30) 는 그 후 비디오 인코더 (22)에 의해 수행된 것과 동일한 MTT 파티셔닝을 적용할 수도 있다. 일부 예에서, 비디오 데이터의 프레임이 비디오 인코더 (22)에 의해 어떻게 파티셔닝되는지는 비디오 디코더 (30)에서 동일한 미리 정의된 규칙 세트를 적용함으로써 결정될 수도 있다. 그러나, 많은 상황들에서, 비디오 인코더 (22) 는 코딩되는 비디오 데이터의 특정 프레임에 대한 레이트-왜곡 기준에 기초하여 사용할 특정 파티션 구조 및 파티션 유형을 결정할 수도 있다. 이와 같이, 비디오 디코더 (30) 가 특정 프레임에 대한 파티셔닝을 결정하기 위해, 비디오 인코더 (22) 는 프레임 및 프레임의 CTU 들이 어떻게 파티셔닝되어야 하는지를 나타내는 신택스 엘리먼트를 인코딩된 비트 스트림에서 시그널링할 수도 있다. 비디오 디코더 (30) 는 이러한 신택스 엘리먼트들을 파싱하고 이에 따라 프레임 및 CTU 들을 파티셔닝할 수도 있다.

[0084] 본 개시의 일례에서, 비디오 인코더 (22)는 시퀀스 파라미터 세트 (SPS), 픽처 파라미터 세트 (PPS), 슬라이스 헤더, 적응형 파라미터 세트 (APS), 또는 임의의 다른 고레벨 신택스 파라미터 세트에서, 지원되는 파티션 유형들의 특정 서브 세트를 고레벨 신택스 엘리먼트로서 시그널링하도록 구성될 수도 있다. 예를 들어, SPS (Sequence Parameter Set), PPS (Picture Parameter Set) 또는 임의의 다른 고레벨 신택스 파라미터 세트에서, 파티션 유형들의 최대 수 및 어느 유형들이 지원되는지가 고레벨 신택스 엘리먼트로서 비트 스트림에서 시그널링되거나 미리 정의될 수도 있다. 비디오 디코더 (30) 는 그러한 신택스 엘리먼트를 수신 및 파싱하여 사용중인 파티션 유형의 특정 서브 세트 및/또는 지원되는 파티션 구조들 (예를 들어, QT, BT, TT 등) 및 타입들의 최대 수를 결정하도록 구성될 수도 있다.

[0085] 일부 예에서, 각각의 깊이에서, 비디오 인코더 (22) 는 트리 구조의 해당 깊이에서 사용되는 선택된 파티셔닝 유형을 나타내는 인덱스를 시그널링하도록 구성될 수도 있다. 또한, 일부 예들에서, 비디오 인코더 (22) 는 각 CU 에서 이러한 파티션 유형 인덱스를 적응적으로 시그널링할 수도 있으며, 즉, 인덱스는 상이한 CU 에 대해 상이할 수 있다. 예를 들어, 비디오 인코더 (22) 는 하나 이상의 레이트-왜곡 계산에 기초하여 파티셔닝 유형의 인덱스를 설정할 수도 있다. 일 예에서, 특정 조건이 만족되면, 파니셔닝 유형 (예를 들어, 파티셔닝 유형의 인덱스) 의 시그널링은 스킵될 수도 있다. 예를 들어, 비디오 인코더 (22) 는 특정 깊이와 연관된 하나의 지원

되는 파티셔닝 유형만 존재하는 경우 파티션 유형의 시그널링을 스킵할 수도 있다. 이 예에서, 픽쳐 경계에 근접할 때, 코딩될 영역은 CTU 보다 작을 수도 있다. 결과적으로, 이러한 예에서, CTU 는 픽쳐 경계에 맞도록 분할되도록 강제될 수도 있다. 일 예에서, 대칭 2 진 트리 만이 강제된 분할을 위해 사용되고, 파티셔닝 유형이 시그널링되지 않는다. 일부 예에서, 소정의 깊이에서, 파티셔닝 유형은 슬라이스 유형, CTU 깊이, CU 위치와 같은 이전에 코딩된 정보에 기초하여 도출될 수도 있다.

[0086] 본 개시의 다른 예에서, 각각의 CU (리프 노드) 에 대해, 비디오 인코더 (22) 는 또한 변환이 CU 와 동일한 크기로 수행되어야 하는지를 표시하기 위해 신택스 엘리먼트 (예를 들어, 1-비트 transform_split 플래그) 를 시그널링하도록 구성될 수도 있다 (즉, 플래그는 TU 가 CU 와 동일한 크기인지 아니면 더 분할되는지를 표시한다). transform_split 플래그가 참으로 시그널링되는 경우, 비디오 인코더 (22) 는 CU 의 나머지를 다수의 서브 블록들로 더 분할하도록 구성될 수 있고, 변환은 각 서브 블록에 대해 수행된다. 비디오 디코더 (30) 는 상호적 프로세스를 수행할 수도 있다.

[0087] 일례로, transform_split 플래그가 참으로 시그널링될 때, 다음이 수행된다. CU 가 정사각형 블록에 대응하는 경우 (즉, CU 가 정사각형인 경우), 비디오 인코더 (22) 는 쿼드 트리 분할을 사용하여 나머지를 4 개의 정사각형 서브 블록으로 분할하고, 변환은 각각의 정사각형 서브 블록에 대해 수행된다. CU 가 MxN 과 같은 비정사각형 블록에 대응하면, 비디오 인코더 (22) 는 나머지를 2 개의 서브 블록들로 분할하고, 그 서브 블록 크기는 M > N 일 때 0.5MxN 이고, M < N 일 때 Mx0.5N 이다. 다른 예로서, transform_split 플래그가 참으로서 시그널링되고 CU 가 비정사각형 블록, 예를 들어, MxN 에 대응하는 경우 (즉, CU 가 비정사각형인 경우), 비디오 인코더 (22) 는 나머지를 크기 KxK 를 갖는 서브블록들로 분할하도록 구성될 수도 있고, KxK 정사각형 변환이 각 서브 블록에 대해 사용되며, 여기서 K 는 M 및 N 의 최대 팩터와 동일하다. 다른 예로서, transform_split 플래그는 CU 가 정사각형 블록일 때 시그널링되지 않는다.

[0088] 일부 예들에서, 분할 플래그는 시그널링되지 않고 예측 후에 CU 에 잔차가 있을 때 하나의 도출된 크기를 갖는 변환만이 사용된다. 예를 들어, 크기가 MxN 인 CU 는 KxK 정사각형 변환이 사용되며, 여기서 K 는 M 과 N 의 최대 팩터와 동일하다. 따라서, 이 예에서, 크기가 16x8 인 CU 의 경우, 동일한 8x8 변환이 CU 의 잔차 데이터의 2 개의 8x8 서브 블록에 적용될 수도 있다. "분할 플래그 (split flag)" 는 트리 구조의 노드가 트리 구조에서 차일드 노드를 갖는다는 것을 나타내는 신택스 엘리먼트이다.

[0089] 몇몇 예에서, 각 CU 에 대해, CU 가 제곱된 쿼드 트리 또는 대칭 이진 트리로 분할되지 않으면, 비디오 인코더 (22) 는 항상 변환 크기를 파티션의 크기 (예를 들어, CU 의 크기) 와 동일하게 설정하도록 구성된다.

[0090] 비디오 인코더 (22) 를 참조하여 설명된 상기 예들의 각각에 대해, 비디오 디코더 (30) 는 상호적 프로세스를 수행하도록 구성될 수 있음을 이해해야한다. 신택스 엘리먼트를 시그널링하는 것과 관련하여, 비디오 디코더 (30) 는 그러한 신택스 엘리먼트를 수신 및 파싱하고, 이에 따라 연관된 비디오 데이터를 파티션 및 디코딩하도록 구성될 수도 있다.

[0091] 본 개시의 하나의 특정 예에서, 비디오 디코더는 3 개의 상이한 파티션 구조 (QT, BT 및 TT) 에 따라 비디오 블록을 파티셔닝하도록 구성될 수도 있으며, 각각의 깊이에서 5 개의 상이한 파티셔닝 유형들이 허용된다. 파티셔닝 유형들은, 도 5a 내지 도 5e 에 도시된 바와 같이, 쿼드 트리 파티셔닝 (QT 파티션 구조), 수평 이진 트리 파티셔닝 (BT 파티션 구조), 수직 이진 트리 파티셔닝 (BT 파티션 구조), 수평 중심측 삼중 트리 파티셔닝 (TT 파티션 구조), 및 수직 중심측 삼중 트리 파티셔닝 (TT 파티션 구조) 를 포함한다.

[0092] 다섯 가지 예시의 파티셔닝 유형들의 정의는 다음과 같다. 정사각형은 직사각형의 특별한 경우로 간주된다.

[0093] · 쿼드 트리 파티셔닝 : 블록은 네 개의 동일 크기 직사각형 블록으로 더 분할된다. 도 5a 는 쿼드트리 파티셔닝의 예를 도시한다.

[0094] · 수직 이진 트리 파티셔닝: 블록은 2 개의 동일 크기 직사각형 블록으로 수직으로 분할된다. 도 5b 는 수직 이진 트리 파티셔닝의 예이다.

[0095] · 수평 이진 트리 파티셔닝: 블록은 2 개의 동일 크기 직사각형 블록으로 수평으로 분할된다. 도 5c 는 수평 이진 트리 파티셔닝의 예이다.

[0096] · 수직 중심측 삼중 트리 파티셔닝: 블록은 세 개의 직사각형 블록으로 수직으로 분할되어 두 개의 사이드 블록이 같은 크기를 공유하는 한편 센터 블록의 크기가 두 개의 사이드 블록의 합계가 되도록 한다. 도 5d 는 수직 중심측 삼중 트리 파티셔닝의 예이다.

[0097] · 수평 중심측 삼중 트리 파티셔닝: 블록은 세 개의 직사각형 블록으로 수평으로 분할되어 두 개의 사이드 블록이 같은 크기를 공유하는 한편 센터 블록의 크기가 두 개의 사이드 블록의 합계가 되도록 한다. 도 5e 는 수평 중심측 삼중 트리 파티셔닝의 예이다.

[0098] 특정 깊이와 연관된 블록에 대해, 비디오 인코더 (22) 는 (추가 분할을 포함하지 않는) 어떤 파티셔닝 유형이 사용되는지를 결정하고, 그 결정된 파티션 유형을 비디오 디코더 (30) 에 명시적으로 또는 암시적으로 (예를 들어, 파티션 유형이 미리 결정된 규칙으로부터 유도될 수 있음) 시그널링한다. 비디오 인코더 (22) 는 상이한 파티션 유형을 사용하는 블록에 대한 레이트-왜곡 비용을 검사하는 것에 기초하여 사용할 파티션 유형을 결정할 수도 있다. 레이트 왜곡 비용을 얻기 위해, 비디오 인코더 (22) 는 블록에 대한 가능한 파티셔닝 유형을 재귀적으로 검사할 필요가 있을 수도 있다.

[0099] 도 6 은 코딩 트리 유닛 (CTU) 파티셔닝의 일 예를 도시한 개념적 다이어그램이다. 다시 말해서, 도 6 은 CTU 에 대응하는 CTB (80)의 파티셔닝을 도시한다. 도 6 의 예에서,

[0100] · 깊이 0 에서, CTB (80) (즉, 전체 CTB) 은 (단일 점들에 의해 분리된 대시들을 갖는 라인 (82) 에 의해 표시된 바와 같은) 수평 이진 트리 파티셔닝으로 2 개의 블록으로 분할된다.

[0101] · 깊이 1 에서:

[0102] · 상측 블록은 (작은 대시들을 갖는 라인들 (84 와 86) 에 의해 표시된 바와 같은) 수직 중심측 삼중 트리 파티셔닝으로 세 개의 블록으로 분할된다.

[0103] · 하측 블록은 (두 개의 점으로 분리된 대시들을 갖는 라인들 (88 및 90) 에 의해 표시된 바와 같은) 쿼드 트리 파티셔닝으로 4 개의 블록으로 분할된다.

[0104] · 깊이 2 에서:

[0105] · 좌측 블록은 (짧은 대시들에 의해 분리된 긴 대시들을 갖는 라인들 (92 와 94) 에 의해 표시된 바와 같은) 수평 중심측 삼중 트리 파티셔닝으로 세 개의 블록으로 분할된다.

[0106] · 깊이 1 의 상측 블록의 중심 및 우측 블록들에 대해 더 이상의 분할은 없다.

[0107] · 깊이 1 의 하측 블록의 4 개의 블록들에 대해 더 이상의 분할은 없다.

[0108] 도 6 의 예에서 알 수 있듯이, 3 개의 상이한 파티션 구조가 4 가지 상이한 파티션 유형 (수평 2진 트리 파티셔닝, 수직 중심측 트리플 트리 파티셔닝, 쿼드트리 파티셔닝, 및 수평 중심측 트리플 트리 파티셔닝) 과 함께 사용된다 (BT, QT, TT).

[0109] 다른 예에서, 추가의 제약들이 특정 깊이 또는 특정 크기의 블록에 적용될 수도 있다. 예를 들어, 블록의 높이/너비가 16 픽셀보다 작으면, 4 픽셀보다 작은 높이/너비를 갖는 블록을 피하기 위해 블록이 수직/수평 중심측 트리로 분할될 수 없다.

[0110] F. Le Léannec, T. Poirier, F. Urban, "Asymmetric Coding Units in QTBT", JVET-D0064, Chengdu, Oct. 2016 (hereinafter "JVET-D0064") 에서, 비대칭 코딩 유닛들이 QTBT 와 함께 사용되도록 제안되었다. 4 개의 새로운 2 진 트리 분할 모드들 (예를 들어, 파티션 유형들) 이 새로운 분할 구성들을 허용하도록 QTBT 프레임 워크에 도입되었다. 도 7 에 도시 된 바와 같이, QTBT 에서 이미 이용 가능한 분할 모드 이외에 소위 비대칭 분할 모드가 제안되었다. 도 7 에 도시된 바와 같이, HOR_UP, HOR_DOWN, VER_LEFT 및 VER_RIGHT 파티션 유형은 비대칭 분할 모드의 예이다.

[0111] 추가된 비대칭 분할 모드에 따르면, 크기 (S) 를 갖는 코딩 유닛은 크기들 (S/4 및 3S/4) 을 갖는 2 개의 서브 CU 로, 수평 (예를 들어, HOR_UP 또는 HOR_DOWN) 또는 수직 (예를 들어, VER_LEFT 또는 VER_RIGHT) 방향 중 하나로 분할된다. JVET-D0064 에서 새로 추가된 CU 너비 또는 높이는 단지 12 또는 24 일 수 있다.

[0112] 순방향 및 역변환에 대한 기술을 이제 설명합니다. 이미지/비디오 코딩에서 변환은 주로 2-D 입력 데이터 소스에 적용된다. 2-D 입력 데이터에 변환을 적용하는 예시적인 방법은 분리 가능 및 분리 불가능한 2-D 변환들을 포함한다. 분리 가능한 2-D 변환은 분리 불가능한 2-D 변환과 비교할 때 더 적은 연산 (더하기 및 곱하기) 카운트들을 요구하기 때문에 일반적으로 분리 가능한 2-D 변환이 사용된다.

[0113] 일례에서, 변수 X 는 입력 WxH 데이터 배열이며, 여기서 W는 배열의 너비이고 H 는 배열의 높이이다. 예시의

분리가능한 2-D 변환은 순차적으로 X 의 수평 및 수직 벡터들에 대한 1-D 변환들을 적용하며, 이하에 공식화되며:

[0114] $$Y = C \cdot X \cdot R^T$$

[0115] 여기서 Y 는 X 의 변형된 배열이고, C 과 R 는 각각 정수 정밀도 또는 배정밀도의 WxW 및 HxH 변환 행렬을 나타낸다., (아래 수학식 (1) 에서와 같이), 정수값으로 변환 행렬을 나타낸다. 그 공식화로부터, C 는 X 의 열 벡터들에 대한 1-D 수직 (열, 좌측) 변환들을 적용하는 반면, R 은 X 의 행 벡터에 대해 1-D 수평 (행, 우측) 변환들을 적용한다.

[0116] HEVC 에서, W 는 H 와 같다 (그리고 S 와 같다). 이와 같이, WxH 데이터 어레이는 2^{2*K}로 표시될 수 있으며, 여기서 K 는 정수이고 S 는 2^K 와 같다. 변환 행렬 (C 과 R) 은 다음과 같이 T 로서 생성된다.

[0117] $$T = int(\sqrt{S} \cdot M \cdot 2^N) \qquad (1)$$

[0118] 여기서, T 는 정수 값의 변환 행렬을 나타내며, $int()$ 는 행렬의 각 부동 소수점 요소 값에 대해 가장 가까운 정수 값을 얻는 함수이며, S 는 변환의 크기 (예를 들어, 8 포인트 또는 32 포인트 변환) 를 나타내며, M 은 플로우트 (float) 값으로 변환 유형에 대한 유니터리 행렬을 나타내고, 2^N 은 정수 변환 행렬 (T) 의 정확도를 제어하는 스케일링 팩터이다 (예를 들어, N= 6 이 HEVC 변환에서 사용됨). 복소 정사각형 행렬 (U) 은 그것의 공액 전치 (U^*) 가 또한 그의 역인 경우 유니터리이다 (즉, $U^*U = UU^* = I$, 여기서 I 는 단위 행렬이다.) 또한, HEVC 에서, 변환 행렬은 +1 또는 -1 로 약간 조정되는 몇 개의 요소들을 갖는 T 로서 도출될 수도 있다.

[0119] HEVC 에서, 수평 및 수직 변환이 적용된 후, 변환의 출력은 유니터리 변환 (즉, M) 과 비교할 때 대략 $(\sqrt{S} \cdot 2^N)^2$ 에 의해 확대된다. S 가 HEVC 에서 2 의 거듭제곱이기 때문에, $(\sqrt{S} \cdot 2^N)^2$ 의 값 또한 2 의 거듭제곱이 된다. 따라서, 변환 행렬에 도입된 $\sqrt{S}$ 은 변환 동안 비트 시프트로 보상될 수 있다. HEVC 행렬은 직교정규 (orthonormal) DCT 변환과 비교하여, 그리고 순방향 및 역방향 2 차원 변환들을 통해 잔차 블록의 놈 (norm) 을 보존하기 위해, $(\sqrt{S} \cdot 2^N = 2^{N+\frac{K}{2}} = 2^{6+\frac{K}{2}},\ S = 2^K$을 가정) 에 의해 스케일링되기 때문에, 추가적인 스케일 팩터들 (fwd_shift1 및 fwd_shift2) 이 적용된다. 유사하게, 역변환에 대해서, 스케일링 팩터는 또한 $\sqrt{S} \cdot 2^N$ 에 의해 스케일링된 역변환 행렬로 인해 사용된다. 결과적으로, 2 차원 순방향 및 역방향 변환들을 통해 놈을 보존하기 위해, 스케일 팩터들의 곱은 $(1/(2^{N+\frac{K}{2}})^4 = 1/2^{24+2*K})$ 와 동일할 것이다. 예를 들어, HEVC 참조 소프트웨어에서의 순방향 변환에서, 비디오 인코더 (22) 는 수평 (fwd_shift1) 및 수직 (fwd_shift2) 순방향 변환들 후에 시프트를 각각 다음과 같이 적용하여 각각의 변환 후에 D 가 8 과 동일한 경우 출력이 16 비트들로 피팅하는 것을 확실하게 한다.

[0120] $$\text{fwd\_shift1} = \log_2 S + D + 6 - r \qquad (2)$$

[0121] $$\text{fwd\_shift2} = \log_2 S + 6 \qquad (3)$$

[0122] HEVC 참조 소프트웨어의 역변환들에 대해, 비디오 디코더 (30) 는 각각 수직 (inv_shift1) 및 수평 (inv_shift2) 역변환들 후의 시프트를 각각 다음과 같이 적용한다,

[0123] $$\text{Inv\_shift1} = 7 \qquad (4)$$

[0124] $$\text{Inv\_shift2} = 5 - D + r \qquad (5)$$

[0125] 여기서, D 는 비디오를 재구성하는 데 사용되는 비트 깊이를 나타낸다. 4 개의 시프트들의 합은 ($2*\log_2 S + 24 = 2*K + 24$) 이다. 비트 깊이는 SPS 에서 특정될 수도 있다. D= 8 및 D= 10 의 비트 깊이 값들은 각각 8 비트 및 10 비트 픽셀 재구성을 초래한다. 파라미터 r 은 수평 순방향 변환의 정밀도를 제어한다. r 의 더 큰 값은 더 높은 정밀도를 제공한다. 일부 예에서, r 의 값은 SPS에서 특정된 바와 같은 구성에 따라 고정 값 15

또는 15 또는 D+6 중 최대값 (예를 들어, max(15, D+6)) 일 수도 있다.

[0126] HEVC 에서 사각형 변환 (수평 및 수직 변환의 크기가 동일함) 만 사용되기 때문에 S 의 동일한 값 위의 shift1 및 shift2 계산에 사용된다.

[0127] QTBT 파티셔닝을 사용하는 비디오 코딩의 일부 예에서, 몇몇 새로운 변환들, 예를 들어 비정사각형 8x4 변환들. 이 경우, log2(W) + log2(H) 의 출력은 짝수 값이 아니다. 이 경우, 추가적인 팩터 $\sqrt{2}$ 가 도입되지만, 변환 중에 간단한 비트 시프트로 보상될 수 없다. 따라서, 2016 년 11 월 22 일에 출원된 미국 특허 공보 제 20170150176 호 및 2016 년 11 월 22 일자로 출원된 미국 특허 공보 제 20170150183 호에 기술된 바와 같이, 그 값 ($\sqrt{2}$) 을 (예를 들어, 변환 행렬을 변경하는 것과 반대되는) 양자화 프로세스에 흡수하는 것이 제안되었다.

[0128] 비대칭 코딩 유닛들 (예를 들어, 도 7 에 도시된 것들) 에서, 크기 12 및 24 의 변환들과 같이, 2 의 거듭제곱과 동일하지 않은 크기를 갖는 변환들이 사용될 수도 있다. 따라서, 이러한 비대칭 코딩 유닛은 변환 프로세스에서 쉽게 보상될 수 없는 더 많은 팩터들을 도입한다. 이러한 비대칭 코딩 유닛들에 대해 변환 또는 역변환을 수행하기 위해서는 추가적인 프로세싱이 필요할 수도 있다.

[0129] 순방향 및 역방향 양자화에 대한 기술을 이제 설명한다. 잔차 에너지를 보다 낮은 주파수 계수로 압축하는 변환 프로세스 후에, 비디오 인코더 (22) 는 잔차 재구성의 왜곡을 제어하기 위해 양자화를 적용한다. 따라서, 비디오 디코더 (30) 에서, 역양자화 (탈양자화) 프로세스가 역변환 전에 수행된다.

[0130] 순방향 양자화의 경우, HEVC 참조 소프트웨어에서, 비디오 인코더 (22) 는 후술하는 바와 같이 데드존 + 균일 양자화 방식을 적용한다.

[0131] $$y' = \text{sign}(y)(|y|Q + f\ 2^{qbits}) >> qbits \qquad (6)$$

[0132] 여기서, y 는 입력 변환 계수이고, Q 는 양자화 스케일링 팩터이고, f 는 (데드존이 범위 [-(1-f)*Δ, (1-f)*Δ] 내에 있는 도 8 에서 도시된 바와 같은) 데드존 크기의 크기를 제어하는 라운딩 오프셋이고, sign (y) = y>0 ? 1: -1 이고, $qbits$ 는 시프팅 파라미터이며, y' 는 출력에 양자화 된 변환 계수를 제공한다. 데드존 영역에 속하는 모든 값은 0 으로 양자화될 것이다. 도 8 은 데드존 플러스 균일 양자화 방식을 나타내는 개념도이다. 일 예에서, 도 8 에서의 델타 (△) 의 값은 $\Delta = 2^{qbits}$ 일 수도 있다.

[0133] HEVC 에서, 인트라 슬라이스의 경우, f 는 171/512 이고, 그렇지 않으면 f 는 85/512 이다.

[0134] 상기 양자화 스케일링 팩터 Q 및 시프팅 파라미터 $qbits$ 는 다음과 같이 특정되며,

[0135] $$Q = g\_quantScales[QP\%6] \qquad (7)$$

[0136] 여기서, QP 는 사용자가 정의된 양자화 파라미터이고, $g\_quantScales$ 는 아래의 HEVC 참조 소프트웨어에서 특정된 상수 배열이다.

[0137] const Int g_quantScales[SCALING_LIST_REM_NUM] =

[0138] {

[0139] 26214,23302,20560,18396,16384,14564

[0140] };

[0141] 게다가, $qbits$ 는 아래에서 도출되며.

[0142] $$qbits = 14 + \lfloor QP/6 \rfloor + \text{iTransformShift} \qquad (8)$$

[0143] 여기서, iTransformShift = r - D - $\log_2(S)$ 이며, 여기서 S 는 블록 크기이고, D 및 r 은 위의 식 (2) 및 식 (5) 에서 정의된 것과 동일하다.

[0144] HEVC 의 역양자화의 경우, 이하와 같이 역양자화 프로세스가 특정되며,

[0145] $$\hat{y} = sign(y') \cdot \left(|y'| \cdot DQ + 2^{qbits'-1}\right) \gg qbits' \qquad (9)$$

[0146] 여기서, y' 는 입력 양자화된 변환 계수이고, $\hat{y}$ 는 탈양자화된 변환 계수이고, DQ 는 다음과 같이 특정되며,

[0147] Q = g_invQuantScales[QP%6] (10)

[0148] 여기서, g_invQuantScales 는 아래와 같이 특정된 상수 배열이며,

[0149] const Int g_invQuantScales[SCALING_LIST_REM_NUM] =

[0150] {

[0151] 40,45,51,57,64,72

[0152] };

[0153] 또, $qbits'$ 가 아래에서 도출되며,

[0154] $$qbits' = 6 - [QP/6] - \text{iTransformShift} \qquad (11)$$

[0155] 여기서, iTransformShift = $r - D - \log_2(S)$ 이며, 여기서 S 는 블록 크기이고, D 및 r 은 식 (2) 및 식 (5) 에서 정의된 것과 동일하다.

[0156] 식 (8) 의 iTransformShift 의 상기 정의로부터, 역양자화 프로세스는 블록 크기에 의존한다는 것을 알 수있다.

[0157] 인트라 슬라이스에 대한 분리된 트리 구조가 이제 논의될 것이다. VCEG 제안서 COM16-C966 에서, 인트라 슬라이스에 대한 분리된 트리 구조가 제안되었다. 특히 크로마 컴포넌트의 코딩 성능을 향상시키기 위해, 인트라 슬라이스에서 루마 및 크로마 컴포넌트에 대해 서로 다른 트리 구조를 갖는 것이 제안되었다. 즉, 크로마 블록은 루마 블록과 다른 방식으로 파티셔닝될 수도 있다. 인트라 슬라이스는 인트라 코딩된 코딩 유닛을 포함하는 슬라이스이다.

[0158] 다음의 문제점들은 다양한 QTBT 및 MTT 파티셔닝 구조에 따라 비디오 데이터를 코딩하기 위한 현재의 제안에서 관찰된다. 크기가 2 의 거듭제곱이 아닌 변환을 MTT 프레임워크에 통합할 때, 변환 및/또는 역변환 프로세스 동안 시프트 연산으로 보상될 수 없는 팩터들을 보다 효율적으로 처리하려면 특수 프로세싱이 필요할 수도 있다. MTT 프레임워크 하에서 분리된 루마 및 크로마 트리 구조를 사용할 때, 크로마 컴포넌트에 대한 복잡한 트리 구조가 항상 도움이되는 것은 아닐 수도 있다. 분리된 루마 및 크로마 트리를 인터 프레임으로 확장할 때, 모션 정보는 루마 및 크로마 양자 모두에 대해 시그널링되고, 이것은 상당한 시그널링 비용을 발생시킨다. 분리된 루마 및 크로마 트리 구조를 사용할 때, 때로는 루마 및 크로마 트리가 동일한 분할 패턴을 갖는다. 이 경우, 루마 및 크로마 파티션을 별도로 시그널링하는 것이 효율적이지 않을 수 있다.

[0159] 상기 언급된 문제점들을 다루기 위해, 다음의 기법들이 제안된다. 비디오 인코더 (22) 및 비디오 디코더 (30) 는 상호적 방식으로 다음의 기법들을 수행하도록 구성될 수도 있다. 다음의 항목화된 기법들은 개별적으로 적용될 수도 있다. 또한, 다음의 기법들 각각은 임의의 조합으로 함께 사용될 수 있다.

[0160] 본 개시의 제 1 예에서, 크기가 2 의 거듭제곱이 아닌 변환을 통합 할 때, 본 개시는 비디오 디코더 (30) 에서 및/또는 비디오 인코더 (22) 및 비디오 디코더 (30) 양자 모두에서 식 (1) 에서의 진정한 크기 (S) 대신에 라운딩된 변형된 S' 를 사용하여 변환 행렬을 생성할 것을 제안한다. 블록이 정사각형이 아닌 경우 크기가 2 의 거듭제곱이 아닌 변환이 발생할 수도 있다. 이러한 변화로, 비트 시프팅에 의해 보상될 수 없는 변환의 스케일링은 변환 행렬로 흡수된다. 따라서, ($\sqrt{2}$ 문제가 위에서 설명한 방식으로 처리된다고 가정하면) 변환 및 양자화 프로세싱에서 다른 프로세싱 기법들을 변경할 필요가 없을 수도 있다.

[0161] 제 1 예의 일 양태에서, 비디오 인코더 (22) 및 비디오 디코더 (30) 는 진정한 크기 (S) 를 2 의 거듭제곱인 값으로 라운딩함으로써 변경된 값 (S') 을 획득하기 위해 진정한 크기 (S) 를 변환으로 라운딩하도록 구성될 수도 있다. 예를 들어, 12 의 S 의 값은 16 으로 라운딩되고 24 의 S 의 값은 32 로 라운딩된다. 일반적으로, S

' 의 값은 S 를 위로 또는 아래로 또는 가장 가까운 2 의 거듭제곱으로 라운딩함으로써 획득될 수도 있다.

[0162] 일 예에서, 비디오 디코더 (30) 는 비디오 데이터의 인코딩된 블록을 수신하도록 구성될 수도 있다. 일 예시에서, 비디오 데이터의 인코딩된 블록은 역양자화된 변환 계수를 포함한다. 이 예에서, 비디오 데이터의 블록은 비정사각형 형상을 가질 수도 있다. 비디오 디코더 (30) 는 비디오 데이터의 인코딩된 블록에 대한 변환을 결정하도록 추가로 구성될 수 있으며, 변환은 2 의 거듭제곱이 아닌 크기 (S) 를 갖는다. 비디오 디코더 (30) 는 변형된 크기 (S') 를 갖는 변환을 생성하는 2 의 거듭제곱으로 S 를 라운딩하도록 추가로 구성될 수도 있다. 그 다음, 비디오 디코더 (30) 는 변경된 크기 (S') 를 갖는 역변환을 비디오 데이터의 인코딩된 블록에 적용하여 잔차 비디오 데이터를 생성하고, 잔차 비디오 데이터를 디코딩하여 비디오 데이터의 디코딩된 블록을 생성할 수도 있다.

[0163] 유사하게, 비디오 인코더 (22) 는 비디오 데이터의 블록을 수신하도록 구성될 수도 있다. 일부 예에서, 비디오 인코더 (22) 는 비디오 데이터의 블록은 비정사각형 형상으로 파티셔닝했다. 비디오 인코더 (22) 는 비디오 데이터의 블록을 (예를 들어, 인터 예측 및/또는 인트라 예측을 사용하여) 예측하여 잔차 비디오 데이터를 생성할 수도 있다. 비디오 인코더 (22) 는 잔차 비디오 데이터에 대한 변환을 결정할 수도 있고, 여기서 변환은 2 의 거듭제곱이 아닌 크기 (S) 를 갖는다. 비디오 인코더 (22) 는 S 를 변경된 크기 (S') 를 갖는 변환을 생성하는 2 의 거듭제곱으로 라운딩하고, 변환 계수를 생성하기 위해 변경된 크기 (S') 를 갖는 변환을 잔차 비디오 데이터에 적용할 수도 있다. 일부 예에서, 비디오 인코더 (22) 는 또한 변환 계수를 양자화할 수도 있다. 비디오 인코더 (22) 는 그 후 인코딩된 비디오 비트스트림에 변환 계수를 (예를 들어, CABAC 와 같은 엔트로피 코딩을 사용하여) 인코딩할 수도 있다.

[0164] 제 1 예의 다른 양태에서, 식 (2) 및 식 (3) 의 적응적 시프트는 S 대신에 S' 에 기초한다. 예를 들어, 변경된 식 (2)' 및 식 (3)' 은 다음과 같이 변경될 수도 있다.

[0165] $fwd\_shift1 = \log_2 S' + D + 6 - r$ (2)'

[0166] $fwd\_shift2 = \log_2 S' + 6$ (3)'

[0167] 제 1 예의 또 다른 양태에서, 정수 변환 행렬 (T) 를 도출할 때, 식 (1) 에서 도시된 바와 같은 유니터리 변환 뿐아니라 스케일링 팩터 $(\sqrt{S} \cdot 2^N)$ 는 미리 정의된 고정 값, 예를 들어 256, 512 또는 1024 로 대체된다. 일 예에서, 식 (2) 및 식 (3) 에서 설명된 바와 같은 우측 시프트 연산은 fwd_shift1 및/또는 fwd_shift2 의 값이 더 이상 S 에 의존하지 않는, 즉, $\log_2 S$ 가 식 (2) 및/또는 식 (3) 에서 제거되는 방식으로 변경된다. 이 경우, S 는 변환 크기에 독립적이며, 즉 식 (2) 및 식 (3) 의 시프트는 변환 크기에 관계없이 고정된 값으로 설정된다.

[0168] 제 1 예의 다른 양태에서, 역변환에 대해, 원래의 시프트 연산은 변경되지 않고 유지된다. 즉, 비디오 디코더 (30) 는 식 (4) 및 식 (5) 에서 설명된 바와 같이 역시프트를 수행하도록 구성될 수도 있다.

[0169] $Inv\_shift1 = 7$ (4)

[0170] $Inv\_shift2 = 5 - D + r$ (5)

[0171] 제 1 예의 다른 양태에서, 식 (8) 및/또는 식 (11) 에서 설명된 바와 같은 우측 시프트 연산은 qbits 및/또는 qbits' 의 값이 더 이상 S 에 의존하지 않는, 즉, $\log_2 S$ 가 식 (8) 및/또는 식 (11) 에서 제거되는 방식으로 변경된다. 변경된 식 (8)' 및 (11)' 는 아래와 같다.

[0172] 본 개시의 일 예에서, 변환 계수의 양자화를 수행할 때, 비디오 인코더 (22) 는 $qbits$ 의 값을 아래와 같이 결정할 수도 있으며,

[0173] $$qbits = 14 + \lfloor QP/6 \rfloor + \text{iTransformShift} \quad (8)'$$

[0174] 여기서, iTransformShift = $r - D$ 이며, 여기서 D 및 r 은 위의 식 (2) 및 식 (5) 에서 정의된 것과 동일하다.

[0175] 또, 변환 계수의 역양자화를 수행할 때, 비디오 인코더 (30) 는 $qbits'$ 의 값을 아래와 같이 결정할 수도 있으며,

[0176] $$qbits' = 6 - \lfloor QP/6 \rfloor - \text{iTransformShift} \quad (11)'$$

[0177] 여기서, iTransformShift = $r - D$ 이며, 여기서 D 및 r 은 식 (2) 및 식 (5) 에서 정의된 것과 동일하다.

[0178] 제 1 예의 다른 양태에서, 식 (4) 및/또는 식 (5) 에서 설명된 바와 같은 우측 시프트 연산은 inv_shift1 및/또는 inv_shift2 의 값이 S 에 의존하는 방식으로 변경되며, 예를 들어,

[0179] $$\text{Inv\_shift1} = 7 + \log_2 S \quad (4)''$$

[0180] $$\text{Inv\_shift2} = 5 - D + r + \log_2 S \quad (5)''$$

[0181] 본 개시의 제 2 예에서, 이진 트리, 대칭 중심측 삼중 트리, 쿼드 트리 및 비대칭 트리 유형들을 사용하는 구조와 같은 MTT 구조에 따라 블록을 파티셔닝할 때, 비디오 인코더 (22) 및 비디오 디코더 (30) 는 2 레벨 MTT를 사용하여 블록을 파티셔닝하도록 구성될 수도 있다. 예를 들어, 2-레벨 MTT 구조는 2017 년 3 월 20 일자로 출원된 미국 특허 공보 제 US 20170272782 호에 개시되어 있다. 2-레벨 MTT 에서, 제 1 레벨 ("영역 트리 레벨" 이라고 함) 에서, 비디오 데이터의 픽쳐 또는 블록은 (예를 들어, 쿼드트리 또는 16 진수 트리를 사용하여) 큰 블록을 작은 블록들로 빠르게 파티셔닝할 수 있는 하나 또는 다수의 트리 유형을 각각 가진 영역들로 분할된다. 제 2 레벨 ("예측 레벨" 이라고 함) 에서는, 영역이 (추가의 분할을 포함하지 않는) MTT 기법들로 추가로 분할된다. 예측 트리의 리프 노드는 본 명세서에서 코딩 유닛 (CU)으로 지칭된다. 또, 다음 기법들이 사용될 수도 있다.

[0182] 제 2 예의 일 양태에서, 추가 분할로서 시그널링된 예측 트리에 대하여, 비디오 인코더 (22) 는 우선 수직 또는 수평 분할을 나타내는 플래그를 시그널링할 수도 있다. 비디오 인코더 (22) 는 그 후 그러한 분할이 대칭 분할 (예를 들어, 이진 트리 또는 대칭 중심측 삼중 트리) 인지 여부를 나타내는 플래그를 시그널링 할 수도 있다. 분할이 대칭 분할인 경우, 비디오 인코더 (22) 는 이진 트리 또는 대칭 중심측 삼중 트리와 같은 다양한 허용된 대칭 파티션 유형의 유형 인덱스를 시그널링할 수도 있다. 그렇지 않으면 (예를 들어, 분할이 비대칭 분할인 경우), 비디오 인코더 (22) 는 비대칭 분할이 상측 또는 하측 분할 (예를 들어, 분할이 수평 분할인 경우) 인지를 표시하기 위해 또는 분할이 왼쪽 또는 오른쪽 분할 (예를 들어, 분할이 수직 분할인 경우) 인지를 표시하기 위해 플래그를 시그널링할 수도 있다. 비디오 디코더 (30) 는 전술한 플래그들을 수신 및 파싱하고 이에 따라 비디오 블록들을 파티셔닝하도록 구성될 수도 있다.

[0183] 제 2 예의 다른 양태에서, 추가 분할인 것으로서 시그널링된 예측 트리에 대하여, 비디오 인코더 (22) 는 우선 수직 또는 수평 분할을 나타내는 플래그를 시그널링할 수도 있다. 다음으로, 비디오 인코더 (22) 는 분할이 2 진 트리 파티션 유형인지 여부를 나타내는 플래그를 시그널링할 수도 있다. 분할이 이진 트리 파티션 유형이 아닌 경우, 비디오 인코더 (22) 는 대칭 중심측 삼중 트리 또는 비대칭 트리와 같은 다른 트리 유형의 유형 인덱스를 시그널링할 수도 있다. 분할이 비대칭 트리인 경우, 비디오 인코더 (22) 는 분할이 상측 또는 하측 분할 (예를 들어, 분할이 수평 분할인 경우) 인지를 표시하기 위해 또는 분할이 왼쪽 또는 오른쪽 분할 (예를 들어, 분할이 수직 분할인 경우) 인지를 표시하기 위해 플래그를 시그널링할 수도 있다. 비디오 디코더 (30) 는 전술한 플래그들을 수신 및 파싱하고 이에 따라 비디오 블록들을 파티셔닝하도록 구성될 수도 있다.

[0184] 제 2 예의 다른 양태에서, 추가 분할로서 시그널링된 예측 트리에 대하여, 비디오 인코더 (22) 는 수직 또는 수평 분할을 나타내는 플래그를 시그널링할 수도 있다. 비디오 인코더 (22) 는 그 후 분할이 대칭 중심측 삼중 트리인지 여부를 나타내는 플래그를 시그널링할 수도 있다. 분할이 대칭 중심측 삼중 트리 파티션 유형이 아닌 경우, 비디오 인코더 (22) 는 이진 트리 또는 비대칭 트리와 같은 다른 트리 유형의 유형 인덱스를 시그널링할 수도 있다. 분할이 비대칭 트리인 경우, 비디오 인코더 (22) 는 비대칭 트리 분할이 상측 또는 하측 분할 (예를 들어, 분할이 수평 분할인 경우) 인지를 표시하기 위해 또는 비대칭 트리 분할이 왼쪽 또는 오른쪽 분할 (예를 들어, 분할이 수직 분할인 경우) 인지를 표시하기 위해 플래그를 시그널링할 수도 있다. 비디오 디코더 (30) 는 전술한 플래그들을 수신 및 파싱하고 이에 따라 비디오 블록들을 파티셔닝하도록 구성될 수도 있다.

[0185] 본 발명의 다른 양태에서, 비디오 인코더 (22) 는 이웃 블록의 연관된 표시자/파티션 유형, 또는 슬라이스/픽처 유형과 같은 코딩된 정보의 특성에 따라 수직/수평 분할, 및/또는 상측/하측 분할, 및/또는 좌측/우측 분할, 및/또는 트리 유형 파티션 유형을 나타내는 시그널링 신택스 엘리먼트에 대한 순서를 적응적으로 변경하도록 구성될 수도 있다. 일 예에서, 상이한 슬라이스들/픽처들은 파티셔닝 (예를 들어, 블록들이 분할되는 방법) 을 표시하기 위해 시그널링된 신택스 엘리먼트의 상이한 순서를 사용할 수도 있다. 다른 예에서, 비디오 인코더 (22) 는 블록 당 신택스 엘리먼트의 순서를 변경하도록 구성될 수도 있다. 비디오 디코더 (30) 는 비디오 인코

더 (22) 에 의해 결정된 동일한 순서로 전술한 신택스 엘리먼트를 수신하도록 구성될 수도 있다. 비디오 디코더 (30) 는 비디오 인코더 (22) 와 동일한 방식으로 신택스 엘리먼트들의 순서를 결정하도록 구성될 수도 있다.

[0186] 제 2 예의 다른 양태에서, 비대칭 트리 파티션 유형 (예를 들어, 상측/하측 또는 좌측/우측 파티션) 을 나타내는 신택스 엘리먼트를 엔트로피 코딩하기 위해 사용되는 콘텍스트는 다음과 같이 유도될 수도 있다. 도 9 는 본 개시의 이러한 예의 예시의 비대칭 파티션 유형을 도시한다. 도 9 (좌상측) 에 도시된 각 파티션의 중심 위치 바로 위의 위치를 포함하는 블록의 블록 크기를 A, B, C 라 하자. 이 예에서, 비디오 인코더 (22) 및 비디오 디코더 (30) 는 0 으로 초기화되는 카운터를 사용하여 콘텍스트 모델의 인덱스를 결정할 수도 있다. 카운터의 값은 콘텍스트를 결정하는 데 사용된다.

[0187] 다음 조건들을 고려하라.

[0188] 조건 1: A 가 B 와 같지 않고 B 가 C 와 같은 경우.

[0189] 조건 2: 상측 CU 가 좌측 반절에서 경계를 갖는 수직 파티셔닝된 비대칭 트리인 경우.

[0190] 일 예에서, 조건 1 또는 조건 2 가 만족되면, 비디오 인코더 (22) 및 비디오 디코더 (30) 는 카운터를 1만큼 증분시키도록 구성될 수도 있다. 다른 예에서, 조건 1 및 조건 2 가 모두 만족되면, 비디오 인코더 (22) 및 비디오 디코더 (30) 는 카운터를 1만큼 증분시키도록 구성될 수도 있다. 다른 예에서, 조건 1 이 만족되면, 비디오 인코더 (22) 및 비디오 디코더 (30) 는 카운터를 1만큼 증분시키도록 구성될 수도 있다. 유사하게, 조건 2 가 만족되면, 비디오 인코더 (22) 및 비디오 디코더 (30) 는 카운터를 1만큼 증분시키도록 구성될 수도 있다.

[0191] 다른 예에서, A, B, C 는 파티션에 대해 다른 위치에 있을 수도 있다. 하나의 예에서, 도 9 (상부의 중간) 에 도시된 각 파티션의 좌상측 코너 바로 위의 위치를 포함하는 블록의 블록 크기를 A, B, C 라 하자. 다른 예에서, 도 9 (우상측) 에 도시된 각 파티션의 우상측 코너 바로 위의 위치를 포함하는 블록의 블록 크기를 A, B, C 라 하자.

[0192] 다른 예에서, 도 9 (좌하측) 에 도시된 각 파티션의 중심 위치 바로 좌측의 위치를 포함하는 블록의 블록 크기를 D, E, F 라 하자.

[0193] 이 예에서 다음 조건을 고려하라.

[0194] 조건 1: D 가 E 와 같지 않고 E 가 F 와 같은 경우.

[0195] 조건 2: 좌측 CU 가 상측 반절에서 경계를 갖는 수평 파티셔닝된 비대칭 트리인 경우.

[0196] 일 예에서, 조건 1 또는 조건 2 가 만족되면, 비디오 인코더 (22) 및 비디오 디코더 (30) 는 카운터를 1만큼 증분시키도록 구성될 수도 있다. 다른 예에서, 조건 1 및 조건 2 가 모두 만족되면, 비디오 인코더 (22) 및 비디오 디코더 (30) 는 카운터를 1만큼 증분시키도록 구성될 수도 있다. 다른 예에서, 조건 1 이 만족되면, 비디오 인코더 (22) 및 비디오 디코더 (30) 는 카운터를 1만큼 증분시키도록 구성될 수도 있다. 유사하게, 조건 2 가 만족되면, 비디오 인코더 (22) 및 비디오 디코더 (30) 는 카운터를 1만큼 증분시키도록 구성될 수도 있다.

[0197] 다른 예에서, D, E, F는 다른 위치에 있을 수도 있다. 하나의 예에서, 도 9 (하부의 중간) 에 도시된 각 파티션의 좌상측 코너 바로 좌측의 위치를 포함하는 블록의 블록 크기를 D, E, F 라 하자. 다른 예에서, 도 9 (우하측) 에 도시된 각 파티션의 좌하측 코너 바로 좌측의 위치를 포함하는 블록의 블록 크기를 D, E, F 라 하자.

[0198] 다른 예에서, 비디오 인코더 (22) 및 비디오 디코더 (30) 는 미리 정의된 방식으로 A-F 의 위치들을 결정하도록 구성될 수도 있다. 다른 예에서, 비디오 인코더 (22) 는 SPS, PPS 또는 슬라이스 헤더에서 블록 (A-F) 의 위치를 시그널링하도록 구성 될 수있다.

[0199] 제 2 예의 또 다른 양태에서, 비디오 인코더 (22) 및 비디오 디코더 (30) 는 다음과 같이 트리 대칭성을 시그널링하는데 사용되는 플래그의 컨텍스트를 도출하도록 구성될 수도 있다. 하나의 예에서, 단일 콘텍스트 모델이 사용될 수있다. 다른 예에서, 비디오 인코더 (22) 및 비디오 디코더 (30) 는 카운터에 기초하여 다중-레벨 콘텍스트 모델을 사용하도록 구성될 수도 있다. 일례에서, 카운터의 초기 값은 0이다. 상기 CU 가 비대칭 블록인 경우, 비디오 인코더 (22) 및 비디오 디코더 (30) 는 카운터를 1만큼 증분시키도록 구성될 수도 있다. 좌측 CU 가 비대칭 블록인 경우, 비디오 인코더 (22) 및 비디오 디코더 (30) 는 카운터를 1만큼 증분시키도록 구성될 수도 있다. 좌상측 CU 가 비대칭 블록인 경우, 비디오 인코더 (22) 및 비디오 디코더 (30) 는 카운터를 1만큼 증분시키도록 구성될 수도 있다. 우상측 CU 가 비대칭 블록인 경우, 비디오 인코더 (22) 및 비디오 디코더 (30) 는 카운터를 1만큼 증분시키도록 구성될 수도 있다. 상측의 4 개의 블록들이 비대칭 블록에 속하지 않는

경우, 비디오 인코더 (22) 및 비디오 디코더 (30) 는 카운터를 5 가 되도록 설정하도록 구성될 수도 있다.

[0200] 제 2 예의 또 다른 양태에서, 비디오 인코더 (22) 는 예측 트리가 추가로 분할되는지 여부를 나타내기 위해 조건부로 플래그를 생성하고 시그널링하도록 구성될 수도 있다. 추가 분할없이 예측 트리의 크기가 그의 크기가 지원되지 않는 변환을 사용하는 경우, 비디오 인코더 (22) 는 분할 플래그를 시그널링하지 않도록 구성될 수도 있다. 오히려, 더 이상의 분할없이, 예측 트리의 크기가 그의 크기가 지원되지 않는 변환을 사용한다는 조건에 기초하여, 비디오 인코더 (22) 및 비디오 디코더 (30) 모두는 예측이 더 분할될 것으로 추론하도록 구성될 수도 있다.

[0201] 제 2 예의 또 다른 양태에서, 영역 트리 (RT) 에 기초한 가변 예측 트리 (PT) 깊이를 갖는 비디오 시스템 코딩에서, 최소로 허용 가능한 블록 크기는 RT-의존적일 수 있다. 이와 같이, PT 의 분할 플래그, 분할 방향 플래그 (수평/수직), 트리 대칭성 플래그, 또는 다른 전술한 트리 유형 플래그를 시그널링하는 것이 회피될 수도 있다. 또한, CTU 가 특정 유형 또는 모든 유형의 PT로 분할될 수 있는 경우 프레임 경계를 가로 지르는 CTU 에 유사한 제약이 부과될 수도 있다.

[0202] 본 개시의 일 예에서, 비디오 인코더 (22) 는 2-레벨 다중-유형 트리 파티셔닝 구조를 사용하여 비디오 데이터를 파티셔닝하고, 2-레벨 다중-유형 트리 파티셔닝 구조의 예측 트리가 구조화되는 방법을 나타내는 신택스 엘리먼트들을 생성하도록 구성될 수도 있으며, 그 신택스 엘리먼트들은 수직 또는 수평 분할을 나타내는 플래그, 분할이 대칭 분할인지 여부를 나타내는 플래그, 유형 인덱스, 또는 비대칭 분할이 상측 또는 하측 분할 또는 좌측 또는 우측 분할인지 여부를 나타내는 플래그 중 하나 이상을 포함한다. 일 예에서, 비디오 인코더 (22) 는 이웃 블록들의 블록 크기들에 기초하여 신택스 엘리먼트들에 대한 콘텍스트들을 결정하도록 구성될 수도 있다. 본 개시의 다른 예에서, 비디오 인코더 (22) 는 이웃 블록들의 파티션 유형들에 기초하여 신택스 엘리먼트들에 대한 콘텍스트들을 결정하도록 구성될 수도 있다.

[0203] 본 개시의 제 3 예에서, 루마 및 크로마 컴포넌트들에 대한 분리된 트리 구조를 사용할 때 (예를 들어, 루마 블록들 및 크로마 블록들이 별개로 파티셔닝될 때), 비디오 인코더 (22) 는 루마 및 크로마 컴포넌트들에 대한 허용된 트리 유형들을 별개로 나타내는 신택스 엘리먼트들을 생성 및 시그널링하도록 구성될 수도 있다. 즉, 비디오 인코더 (22) 는 루마 블록들 및 크로마 블록들 양자 모두에 대한 허용된 트리 유형들을 나타내는 별개의 신택스 엘리먼트들을 생성할 수도 있다. 신택스 엘리먼트들의 값들은 특정의 루마 또는 크로마 블록에 대해 2 이상의 트리 유형들 중 어느 것이 허용되는지를 나타낼 수도 있다. 예시의 트리 유형들은 대칭 및 비대칭 이진 트리 유형들, 쿼드트리 트리 유형들, 및 대칭 및 비대칭 삼중 트리 유형들을 포함하는, 상술된 트리 유형들 중 임의의 것일 수도 있다. 비디오 디코더 (30) 는 허용된 트리 유형들을 나타내는 신택스 엘리먼트들을 파싱하도록 구성될 수도 있다.

[0204] 비디오 인코더 (22) 는 또한 특정의 블록에 대해 그 허용된 트리 유형들 중 어느 것을 사용할지를 나타내는 추가적인 신택스 엘리먼트들을 시그널링하도록 구성될 수도 있다. 비디오 디코더는 그 추가적인 신택스 엘리먼트들을 파싱하고 허용된 트리 유형들을 나타내는 신택스 엘리먼트들 및 허용된 트리 유형들 중에서 사용할 특정의 트리 유형을 나타내는 추가적인 신택스 엘리먼트들로부터 특정의 블록을 파티셔닝하는 방법을 결정하도록 구성될 수도 있다.

[0205] 일 예에서, 비디오 인코더 (22) 는 비디오 파라미터 세트 (VPS), 시퀀스 파라미터 세트 (SPS), 픽처 파라미터 세트 (PPS), 적응 파라미터 세트 (APS), 또는 임의의 다른 시퀀스/픽처/슬라이스 레벨 신택스 엘리먼트 바디에서 별개로 루마 및 크로마 컴포넌트들에 대한 허용된 트리 유형들을 시그널링할 수도 있다. 일 예에서, 트리 유형은 이진 트리, 대칭 중심측 삼중 트리, 쿼드 트리 또는 비대칭 트리 유형들 중 적어도 2 개를 포함할 수도 있다. 다른 예에서, 이진 및/또는 대칭 중심측 삼중 트리는 항상 온 (on) 일 (예를 들어, 항상 허용될) 수도 있는 반면, 대칭 중심측 삼중 트리 및/또는 비대칭 CU/트리 파티션 유형은 선택적이며 비트스트림에서 시그널링된다..

[0206] 본 발명의 제 4 예에서, 분리된 루마/크로마 트리 구조를 인터 슬라이스로 확장할 때, 비디오 인코더 (22) 는 예를 들어 루마 트리 (일차 트리라고도 함) 에 대해서만, 모션 정보를 한 번 시그널링하도록 구성될 수도 있다. 그 다음, 비디오 디코더 (30) 는 루마 블록에 대해 병치된 위치에 있는 블록에 대한 다른 트리 구조에 대한 (예를 들어, 크로마 블록에 대한 2 차 트리 구조에 대한) 모션 정보를 승계 (예를 들어, 재사용) 하도록 구성될 수도 있다. 병치된 크로마 블록이 단일의 병치된 루마 블록보다 큰 경우, 비디오 디코더 (30) 는 모든 병치된 루마 블록에 대한 모션 정보를 재사용하도록 구성될 수도 있으며, 즉, 하나의 크로마 코딩 블록은 모든 병치된 루마 블록으로부터의 모션 정보의 수개의 세트들을 포함할 수도 있다. 다른 예에서, 비디오 디코더 (30) 는 다른

트리 (예를 들어, 크로마 블록에 대한 트리 구조) 의 모션 정보에 대한 예측자로서 일차 트리의 모션 정보를 사용하도록 구성될 수도 있다.

[0207] 일반적으로, 본 개시의 이러한 예에 따르면, 비디오 디코더 (30) 는 비디오 데이터의 루마 블록을 파티셔닝하고, 비디오 데이터의 루마 블록을 파티셔닝하는 것과는 별도로 비디오 데이터의 하나 이상의 크로마 블록을 파티셔닝하도록 구성될 수도 있다. 비디오 데이터의 루마 블록에 대한 파티션 구조는 다중 유형 트리 파티션 구조일 수도 있고, 비디오 데이터의 하나 이상의 크로마 블록에 대한 파티션 구조는 또한 다중 유형 트리 파티션 구조일 수도 있다. 비디오 디코더 (30) 는 인터 슬라이스에 대해, 비디오 데이터의 루마 블록에 대한 모션 정보를 결정하고, 비디오 데이터의 루마 블록에 대한 결정된 모션 정보로부터 비디오 데이터의 하나 이상의 크로마 블록에 대한 모션 정보를 추론하도록 또한 구성될 수도 있다.

[0208] 본 발명의 제 5 예에서, 분리된 트리 코딩 (예를 들어, 루마 및 크로마 블록이 잠재적으로 상이한 파티셔닝으로 개별적으로 파티셔닝됨) 에서, (2 차 트리라고 지칭될 수도 있는) 크로마 트리의 트리 분할 패턴은 일차 트리의 병치된 블록이 인트라 및 인터 코딩 된 블록을 모두 포함할 때 (일차 트리라고도 지칭되는) 루마 트리로부터 승계된다. 즉, 비디오 인코더 (22) 및 비디오 디코더 (30) 는 루마 블록이 인트라 및 인터 코딩된 블록을 모두 포함하는 경우 연관된 루마 블록과 동일한 분할 패턴을 사용하여 크로마 블록을 파티셔닝하도록 구성될 수도 있다.

[0209] 이 예에서, 비디오 인코더 (22) 는 일차 트리에서의 병치된 블록 (예를 들어, 루마 블록) 이 동일 유형 블록만을 포함할 때만 2 차 트리에 대한 트리 분할 패턴만을 시그널링하도록 구성될 수도 있다. 즉, 병치된 루마 블록은 모든 인터 코딩된 블록 또는 모든 인트라 코딩된 블록을 포함한다. 일 예에서, 트리 분할 패턴은 트리 유형 (비분할은 또한 특별한 트리 유형으로 간주될 수도 있음) 을 포함하지만, 이에 제한되지는 않는다. 트리 유형은 2 진 트리, 3 중 트리 및 쿼드트리와 같은 대칭 및 비대칭 트리 유형 양자 모두를 포함할 수도 있다.

[0210] 일반적으로, 본 개시의 이러한 예에 따르면, 비디오 디코더 (30) 는 비디오 데이터의 루마 블록을 파티셔닝하고, 루마 블록이 인트라 코딩된 블록들 및 인터 코딩된 블록들 양자 모두를 포함하는 경우에 비디오 데이터의 루마 블록에 대한 것과 동일한 비디오 데이터의 하나 이상의 크로마 블록에 대한 파티셔닝을 추론하며, 루마 블록들이 모두 동일한 유형의 코딩된 블록들을 포함할 때, 시그널링된 신택스 엘리먼트들로부터, 비디오 데이터의 하나 이상의 크로마 블록들에 대한 파티셔닝을 결정하도록 구성될 수도 있다.

[0211] 본 발명의 제 6 예에서, PT 분할 플래그의 시그널링 코스트를 감소시키기 위해, 비디오 인코더 (22) 및 비디오 디코더 (30) 는 현재 블록의 그것에 대한 이웃 블록들의 상대 크기를 사용하여 PT 분할 플래그의 콘텍스트를 결정하도록 구성될 수도 있다.

[0212] 일 예시에서, 비디오 인코더 (22) 및 비디오 디코더 (30) 는 그 이웃 블록들에 대한 블록들의 크기를 사용하여 PT 트리 유형 코딩을 위한 콘텍스트를 선택하도록 구성될 수도 있다. 일 예에서, 현재 블록의 폭이 그의 상측의 이웃의 폭보다 클 때, 현재 블록이 더 분할될 가능성이 더 크다. 유사하게, 현재 블록의 높이가 그의 좌측 이웃의 높이보다 클 때, 현재 블록이 더 분할될 가능성이 더 크다. 또한, 현재 블록 크기에 대한 좌상측, 우상측 및 좌하측 이웃의 상대적 크기는 현재 블록이 추가로 분할되어야 하는지 여부를 결정하는 유용한 정보를 또한 제공한다. 현재 블록 크기가 그의 이웃 블록 크기보다 큰 경우, 현재 블록이 추가로 분할되기가 또한 쉽다. 비디오 인코더 (22) 및 비디오 디코더 (30) 는 PT 분할 플래그에 대한 콘텍스트에 대한 인덱스로서 전술한 이벤트들의 집계된 발생 횟수를 사용하도록 구성될 수도 있다. 또한, 개별 이벤트는 PT 분할 플래그에 대한 콘텍스트들의 세트를 또한 형성할 수 있다.

[0213] 일 예에서, 비디오 인코더 (22) 및 비디오 디코더 (30) 는 PT 분할 방향 (예를 들어, 수평 분할 또는 수직 분할) 에 대한 컨텍스트를 결정하기 위해 상측 이웃 블록의 폭 및 좌측 이웃 블록의 높이를 사용하도록 구성될 수도 있다. 상측 이웃 블록의 폭이 현재 블록의 폭보다 작고, 좌측 이웃 블록의 높이가 현재 블록의 높이보다 크거나 같으면, 현재 블록이 수직으로 분할될 가능성이 더 크다. 유사하게, 좌측 이웃 블록의 높이가 현재 블록의 높이보다 작고, 상측 이웃 블록의 폭이 현재 블록의 폭보다 크거나 같으면, 현재 블록이 수평으로 분할될 가능성이 더 크다.

[0214] 다른 예에서, 비디오 인코더 (22) 및 비디오 디코더 (30) 는 PT 분할 모드 (예를 들어, 이진 트리와 중심측 삼중 트리 사이과 같은 분할 모드들 사이의 결정) 에 대한 콘텍스트를 결정하기 위해 상측 이웃 블록의 폭 및 좌측 이웃 블록의 높이를 사용하도록 구성될 수도 있다. 상측 이웃 블록의 폭이 현재 블록의 폭보다 작고 현재

블록이 수직으로 분할되는 경우, 또는 좌측 이웃 블록의 높이가 현재 블록의 높이보다 작고 현재 블록이 수평으로 분할되는 경우, 현재 블록이 삼중 트리로서 분할될 가능성이 더 크다.

[0215] 일 예에서, 이웃 블록이 이용 가능하지 않은 경우, 비디오 인코더 (22) 및 비디오 디코더 (30) 는 현재 콘텍스트의 유도 중에 디폴트 콘텍스트 값을 사용하도록 구성될 수도 있다. 다른 예로서, (Y, Cb, Cr 또는 깊이 컴포넌트와 같은) 상이한 컴포넌트에 대해 상이한 RT 또는 PT 분할이 허용되는 경우, 다른 컴포넌트들에서의 연관된 블록들을 이웃 블록들로서 사용하여 전술 한 모든 방법을 적용할 수 있다.

[0216] 다른 예에서, 비디오 인코더 (22) 및 비디오 디코더 (30) 는 PT 분할 신택스 엘리먼트의 콘텍스트를 결정하기 위해 현재 블록의 깊이에 대한 이웃 블록의 상대적인 깊이를 사용하도록 구성될 수도 있다.

[0217] 다른 예에서, 최소 블록 크기의 3 배와 동일한 폭/높이를 갖는 블록의 한 측면에 대해, 비디오 인코더 (22) 및 비디오 디코더 (30) 는 콘텍스트들을 도출하기 위해 상측 또는 좌측 이웃 블록들의 중심 위치의 블록 크기 또는 PT 깊이를 사용하도록 구성될 수도 있다.

[0218] 다른 예에서, 블록의 폭/높이가 최소 블록 크기의 3 배와 동일한 경우, 비디오 인코더 (22) 및 비디오 디코더 (30) 는 콘텍스트들을 도출하기 위해 3 개의 이웃 블록들의 블록 크기 또는 PT 깊이의 평균값을 사용하도록 구성될 수도 있다.

[0219] 다른 예에서, 블록의 폭/높이가 최소 블록 크기의 3 배와 동일한 경우, 비디오 인코더 (22) 및 비디오 디코더 (30) 는 콘텍스트들을 도출하기 위해 3 개의 이웃 블록들의 블록 크기 또는 PT 깊이 중 최대값을 사용하도록 구성될 수도 있다.

[0220] 다른 예에서, 블록의 폭/높이가 최소 블록 크기의 3 배와 동일한 경우, 비디오 인코더 (22) 및 비디오 디코더 (30) 는 콘텍스트들을 도출하기 위해 3 개의 이웃 블록들의 블록 크기 또는 PT 깊이 중 최소값을 사용하도록 구성될 수도 있다.

[0221] 다른 예에서, 블록의 폭/높이가 최소 블록 크기의 3 배와 동일한 경우, 비디오 인코더 (22) 및 비디오 디코더 (30) 는 콘텍스트들을 도출하기 위해 3 개의 이웃 블록들의 블록 크기 또는 PT 깊이 중 중간값을 사용하도록 구성될 수도 있다.

[0222] 또 다른 예에서, (상술한 바와 같은) 블록 크기 또는 PT 깊이의 집계된 값은 이웃 블록의 크기의 표현이다. 일 예에서, 상측 이웃의 폭이 현재 블록의 폭보다 작은 경우, 콘텍스트 1 이 사용된다. 그렇지 않으면 콘텍스트 0 이 사용된다. 유사하게, 좌측 이웃의 높이가 현재 블록의 높이보다 작은 경우, 콘텍스트 1 이 사용된다. 그렇지 않으면 콘텍스트 0 이 사용된다. 일 예에서, 블럭 크기 또는 PT 깊이의 집계된 값은 사용할 콘텍스트 모델의 인덱스와 동일하거나 이를 제어하는 카운터를 증분시키는데 사용될 수 있다. 또한, 집계된 값은 단일하게 위치된 값의 대체로서 아래 수식의 고안과 연결될 수 있다. 이웃 블록 각각에 대하여, 다음의 콘텍스트 (CTX) 의 설정 프로세스를 순차적으로 수행할 수 있다. 콘텍스트 인덱스를 선택하는 데 CTX 의 합계를 사용할 수도 있다. 다른 예에서, 아래의 첫 번째 두 방정식이 먼저 이웃 블록들 각각에 적용되고, 마지막 방정식이 콘텍스트 인덱스를 선택하는데 이용되며, 여기서 입력 CTX 는 모든 이웃 블록들로부터의 CTX 의 합이다.

[0223] $$CTX = (W > W_{T2}) + (H > H_{L2}) + (W * H > STL) + (W * H > STR)$$

[0224] $$CTX = ((W < W_{T2})\&\&(H < H_{L2})\&\&(W * H < STL)\&\&(W * H < STR))\,?\,0 : CTX$$

[0225] $$CTX = (CTX \geq 3)\,?\,3 : CTX$$

[0226] 일반적으로, 본 개시의 이러한 예에 따르면, 비디오 인코더 (22) 및 비디오 디코더 (30) 는 현재 블록에 대한 이웃 블록들의 상대적 크기에 기초하여 현재 블록의 분할 플래그에 대한 콘텍스트를 결정하고, 그 결정된 콘텍스트에 기초하여 분할 플래그를 콘텍스트 코딩하도록 구성될 수도 있다.

[0227] 본 발명의 제 7 예에서, MTT 와 같은 임의의 트리 구조 프레임 워크에서, 비디오 인코더 (22) 및 / 또는 비디오 디코더 (30) 는 블록 크기에 기초하여 변환을 적용하도록 구성될 수도 있다. 일부 예에서, 특정 블록 크기는 관련된 변환들을 갖지 않을 수도 있다 (즉, 동일 크기 변환이 지원되지 않음). 예를 들어, 비디오 인코더 (22) 및/또는 비디오 디코더 (30) 는 64x48 CU 를 파티셔닝하도록 구성될 수도 있지만 48x48 변환을 사용하도록 구성되지 않을 수도 있다. 다른 예에서, 비디오 인코더 (22) 및/또는 비디오 디코더 (30) 는 256x256 CU 를 파티셔닝하도록 구성될 수도 있지만, 최대 지원 변환은 단지 128x128 뿐이다.

[0228] 이들 예들에서, 비디오 인코더 (22) 및 비디오 디코더 (30) 는 그러한 CU 들에 대한 소정의 코딩 모드들만을 허용하도록 구성될 수도 있다. 일 예에서, 비디오 인코더 (22) 및/또는 비디오 디코더 (30) 는 CU 에 대한 관련된 변환이 없다면 CU 에 대해 스킵 모드만 사용하도록 구성될 수도 있다. 이 예에서, 관련된 변환은 CU 의 적어도 하나의 디멘젼과 동일한 크기를 갖는 변환이다. 이 예에서, 비디오 인코더 (22) 는 스킵 모드 플래그를 시그널링하지 않을 수도 있다. 오히려, 비디오 디코더 (30) 는 관련된 변환이 없는 크기의 CU 에 기초하여 스킵 플래그의 값이 참이라고 추론하도록 구성될 수도 있다.

[0229] 다른 예에서, 비디오 인코더 (22) 및/또는 비디오 디코더 (30) 는 CU 의 크기에 대해 지원되는 변환이 없다면 CU 가 임의의 0 이 아닌 잔차 (잔차 값) 를 가지지 않도록 구성될 수도 있다. 이러한 예에서, 비디오 인코더는 코딩된 블록 플래그 (CBF) 를 시그널링하도록 구성되지 않을 수도 있다. 코딩된 블록 플래그는 블록이 임의의 0 이 아닌 변환 계수를 포함하는지 여부를 나타내는 플래그이다. 이 예에서, CU 가 지원되는 변환을 가지지 않으면 CU 는 임의의 0 이 아닌 잔차 값을 갖지 않을 수도 있기 때문에, 비디오 디코더 (30) 는 CBF 플래그가 0 이라고 (즉, 0 이 아닌 변환 계수가 없다고) 표시할 수도 있다.

[0230] 일 예시에서, 비디오 인코더 (22) 및 비디오 디코더 (30) 는 비디오 데이터의 블록의 크기에 기초하여 비디오 데이터의 블록에 대한 코딩 모드를 결정하도록 구성될 수도 있다. 특히, 비디오 인코더 (22) 및 비디오 디코더 (30) 는 비디오 데이터의 블록의 크기 및 비디오 인코더 (22) 및 비디오 디코더 (30) 에 의해 지원되는 변환들에 기초하여 비디오 데이터의 블록에 대한 코딩 모드를 결정하도록 구성될 수도 있다. 비디오 데이터 블록의 크기와 관련된 지원되는 변환이 없다면, 비디오 인코더 (22) 및 비디오 디코더 (30) 는 비디오 데이터의 블록에 대한 소정의 미리 결정된 코딩 모드를 결정하도록 구성될 수도 있다. 일 예시에서, 코딩 모드는 스킵 모드일 수도 있다. 본 개시의 다른 예에서, 코딩 모드는 다른 코딩 모드 (예를 들어, 병합 모드, AMVP 모드, 인트라 모드) 일 수 있지만, CBF 플래그는 0 으로 추론된다.

[0231] 일반적으로, 비디오 인코더 (22) 및 비디오 디코더 (30) 는 블록이 상기 블록의 크기에 기초하여 연관된 변환을 갖는지 여부를 결정하고, 상기 블록이 연관된 변환을 가지지 않으면 상기 블록에 대한 코딩 모드를 재시작하도록 구성될 수도 있다.

[0232] 도 10 은 본 개시물의 기술들을 구현할 수도 있는 예시적인 비디오 인코더 (22) 를 도시한 블록 다이어그램이다. 도 10 은 설명의 목적들을 위해 제공되고, 본 개시에서 폭넓게 예시되고 설명된 기법들의 한정으로 고려되서는 안된다. 본 개시의 기법들은 다양한 코딩 표준들 또는 방법들에 적용가능할 수도 있다.

[0233] 도 10 의 예에서, 비디오 인코더 (22) 는 예측 프로세싱 유닛 (100), 비디오 데이터 메모리 (101), 잔차 생성 유닛 (102), 변환 프로세싱 유닛 (104), 양자화 유닛 (106), 역양자화 유닛 (108), 역변환 프로세싱 유닛 (110), 재구성 유닛 (112), 필터 유닛 (114), 디코딩된 픽처 버퍼 (116), 및 엔트로피 인코딩 유닛 (118) 을 포함한다. 예측 프로세싱 유닛 (100) 은, 인터 예측 프로세싱 유닛 (120) 및 인트라 예측 프로세싱 유닛 (126) 을 포함한다. 인터 예측 프로세싱 유닛 (120) 은, 모션 추정 유닛 및 모션 보상 유닛 (미도시) 를 포함할 수도 있다.

[0234] 비디오 데이터 메모리 (101) 는, 비디오 인코더 (22) 의 컴포넌트들에 의해 인코딩될 비디오 데이터를 저장할 수도 있다. 비디오 데이터 메모리 (101) 에 저장된 비디오 데이터는, 예를 들어, 비디오 소스 (18) 로부터 획득될 수도 있다. 디코딩된 픽처 버퍼 (116) 는, 예를 들어, 인트라 또는 인터 코딩 모드들에서 비디오 인코더 (22) 에 의해 비디오 데이터를 인코딩함에 있어서 사용하기 위한 레퍼런스 비디오 데이터를 저장하는 레퍼런스 픽처 메모리일 수도 있다. 비디오 데이터 메모리 (101) 및 디코딩된 픽처 버퍼 (116) 는 동기식 DRAM (SDRAM) 을 포함한 동적 랜덤 액세스 메모리 (DRAM), 자기저항성 RAM (MRAM), 저항성 RAM (RRAM), 또는 다른 타입들의 메모리 디바이스들과 같은 다양한 메모리 디바이스들 중 임의의 메모리 디바이스에 의해 형성될 수도 있다. 비디오 데이터 메모리 (101) 및 디코딩된 픽처 버퍼 (116) 는 동일한 메모리 디바이스 또는 별도의 메모리 디바이스들에 의해 제공될 수도 있다. 다양한 예들에 있어서, 비디오 데이터 메모리 (101) 는 비디오 인코더 (22) 의 다른 컴포넌트들과 온-칩형이거나 또는 그 컴포넌트들에 대하여 오프-칩형일 수도 있다. 비디오 데이터 메모리 (101) 는 도 1 의 저장 매체 (20) 와 동일하거나 또는 이의 일부일 수도 있다.

[0235] 비디오 인코더 (22) 는 비디오 데이터를 수신한다. 비디오 인코더 (22) 는 비디오 데이터의 픽처의 슬라이스에서 각각의 CTU 를 인코딩할 수도 있다. CTU 의 각각은 동일한 크기의 루마 코딩 트리 블록들 (CTB) 및 화상의 대응하는 CTB들과 연관될 수도 있다. CTU 를 인코딩하는 부분으로서, 예측 프로세싱 유닛 (100) 은 파티셔닝을 수행하여, CTU 의 CTB들을 점진적으로 더 작은 블록들로 분할할 수도 있다. 더 작은 픽셀 블록들은 CU 들의 코딩 블록들일 수도 있다. 예를 들어, 예측 프로세싱 유닛 (100) 은 CTU 와 연관된 CTB 를 트리 구조에 따라 파

티셔닝할 수도 있다. 본 개시의 하나 이상의 기술들에 따르면, 트리 구조의 각 깊이 레벨에서의 트리 구조의 각각의 개별 비-리프 노드에 대해, 개별 비-리프 노드에 대한 복수의 허용된 분할 패턴들이 존재하고, 개별 비-리프 노드에 대응하는 비디오 블록은 복수의 허용 가능한 분할 패턴들 중 하나에 따라 개별 비-리프 노드의 자식 노드들에 대응하는 비디오 블록들로 파티셔닝된다. 일 예에서, 예측 프로세싱 유닛 (100) 또는 비디오 인코더 (22)의 다른 프로세싱 유닛은 전술한 MTT 파티셔닝 기술들의 임의의 조합을 수행하도록 구성될 수도 있다.

[0236] 비디오 인코더 (22) 는 CTU 의 CU들을 인코딩하여 CU들의 인코딩된 표현들 (즉, 코딩된 CU들) 을 생성할 수도 있다. CU 를 인코딩하는 부분으로서, 예측 프로세싱 유닛 (100) 은 CU 의 하나 이상의 PU들 중에서 CU 와 연관된 코딩 블록들을 파티셔닝할 수도 있다. 본 개시의 기술에 따르면, CU 는 단일 PU 만을 포함할 수도 있다. 즉, 본 개시의 일부 예들에서, CU 는 개별 예측 블록들로 분할되지 않고 오히려 예측 프로세스가 전체 CU에 대해 수행된다. 따라서, 각각의 CU 는 루마 예측 블록 및 대응하는 크로마 예측 블록들과 연관될 수도 있다. 비디오 인코더 (22) 및 비디오 디코더 (30) 는 다양한 사이즈들을 갖는 CU들을 지원할 수도 있다. 상기 나타낸 바와 같이, CU 의 사이즈는 CU 의 루마 코딩 블록의 사이즈를 지칭할 수도 있으며, 또한 루마 예측 블록의 사이즈를 지칭할 수도 있다. 상술한 바와 같이, 비디오 인코더 (22) 및 비디오 디코더 (30) 는 상술한 예시적인 MTT 파티셔닝 유형의 임의의 조합에 의해 정의된 CU 크기를 지원할 수도 있다.

[0237] 인터 예측 프로세싱 유닛 (120) 은 CU 의 각각의 PU에 대한 인터 예측을 수행함으로써 PU 를 위한 예측 데이터를 생성할 수도 있다. 전술한 바와 같이, 본 개시의 일부 MTT 예에서, CU 는 단일 PU만을 포함할 수도 있으며, 즉, CU 및 PU 는 동의어일 수도 있다. PU 를 위한 예측 데이터는 PU 의 예측 블록들 및 PU 를 위한 모션 정보를 포함할 수도 있다. 인터 예측 프로세싱 유닛 (120) 은, PU 가 I 슬라이스인지, P 슬라이스인지 또는 B 슬라이스인지에 의존하여 CU 또는 PU 에 대해 상이한 동작들을 수행할 수도 있다. I 슬라이스에 있어서, 모든 PU들은 인트라 예측된다. 따라서, PU 가 I 슬라이스에 있으면, 인터-예측 프로세싱 유닛 (120) 은 PU 에 대해 인터 예측을 수행하지 않는다. 따라서, I-모드로 인코딩된 블록들에 대해, 예측 블록은 동일 프레임 내의 이전에 인코딩된 이웃 블록들로부터의 공간 예측을 이용하여 형성된다. PU 가 P 슬라이스에 있는 경우에, 인터-예측 프로세싱 유닛 (120) 은 PU 의 예측성 블록을 생성하기 위해 단방향 인터 예측을 이용할 수도 있다. PU 가 B 슬라이스에 있는 경우에, 인터-예측 프로세싱 유닛 (120) 은 PU 의 예측성 블록을 생성하기 위해 단방향 또는 양방향 인터 예측을 수행할 수도 있다.

[0238] 인터 예측 프로세싱 유닛 (126) 은 PU에 대한 인트라 예측을 수행함으로써 PU 를 위한 예측 데이터를 생성할 수도 있다. PU 를 위한 예측 데이터는 PU 의 예측 블록들 및 다양한 신택스 엘리먼트들을 포함할 수도 있다. 인트라-예측 프로세싱 유닛 (126) 은 I 슬라이스들, P 슬라이스들, 및 B 슬라이스들에 있어서 PU들에 대해 인트라 예측을 수행할 수도 있다.

[0239] PU 에 대해 인트라 예측을 수행하기 위해, 인트라-예측 프로세싱 유닛 (126) 은 다중의 인트라 예측 모드들을 이용하여, PU 에 대한 예측 데이터의 다중의 세트들을 생성할 수도 있다. 인트라 예측 프로세싱 유닛 (126) 은 이웃하는 PU들의 샘플 블록들로부터의 샘플들을 이용하여 PU에 대한 예측 블록을 생성할 수도 있다. 이웃 PU들은, PU들, CU들, 및 CTU들에 대한 좌-우로, 상부-저부로의 인코딩 순서를 가정할 때, PU 의 상부, 상부 및 우측으로, 상부 및 좌측으로, 또는 좌측으로일 수도 있다. 인트라 예측 프로세싱 유닛 (126) 은, 다양한 수의 인트라 예측 모드들, 예를 들어, 33개 방향 인트라 예측 모드들을 사용할 수도 있다. 일부 예들에서, 인트라 예측 모드들의 수는 PU 와 연관된 영역의 크기에 의존할 수도 있다.

[0240] 예측 프로세싱 유닛 (100) 은, PU 를 위한 인터 예측 프로세싱 유닛 (120) 에 의해 생성된 예측 데이터 또는 PU 를 위한 인트라 예측 프로세싱 유닛 (126) 에 의해 생성된 예측 데이터 중에서 CU 의 PU 를 위한 예측 데이터를 선택할 수도 있다. 일부 예들에 있어서, 예측 프로세싱 유닛 (100) 은 예측 데이터의 세트들의 레이트/왜곡 메트릭들에 기초하여 CU 의 PU들에 대한 예측 데이터를 선택한다. 선택된 예측성 데이터의 예측성 블록들은 본 명세서에서 선택된 예측성 블록들로서 지칭될 수도 있다.

[0241] 잔차 생성 유닛 (102) 은, CU 에 대한 코딩 블록들 (예를 들어, 루마, Cb 및 Cr 코딩 블록들) 및 CU 의 PU 들에 대해 선택된 예측 블록들 (예를 들어, 예측 루마, Cb 및 Cr 블록들) 에 기초하여, CU 에 대한 잔차 블록들 (예를 들어, 루마, Cb 및 Cr 잔차 블록들) 을 생성할 수도 있다. 가령, 잔차 생성 유닛 (102) 은, 잔차 블록들에 있는 각각의 샘플이 CU 의 코딩 블록에 있는 샘플과 CU 의 PU 의 대응하는 선택된 예측 블록 사이의 차이와 동일한 값을 갖도록 CU 의 잔차 블록들을 생성한다.

[0242] 변환 프로세싱 유닛 (104) 은, CU 의 TU 들과 연관된 변환 블록들로 CU 와 연관된 잔차 블록들을 파티셔닝하기 위하여 쿼드 트리 파티셔닝을 수행할 수도 있다. 따라서, TU 는 루마 변환 블록 및 2개의 크로마 변환 블록들

과 연관될 수도 있다. CU 의 TU 들의 루마 및 크로마 변환 블록들의 크기 및 위치는 CU 의 PU 들의 예측 블록들의 크기 및 위치에 기초하거나 또는 기초하지 않을 수도 있다. "잔차 쿼드 트리" (RQT) 로 알려진 쿼드 트리 구조는 각각의 영역들과 연관된 노드들을 포함할 수도 있다. CU 의 TU 들은 RQT 의 리프 노드들에 대응할 수도 있다. 다른 예에서, 변환 프로세싱 유닛 (104) 은 전술한 MTT 기술에 따라 TU 를 파티셔닝하도록 구성될 수도 있다. 예를 들어, 비디오 인코더 (22) 는 RQT 구조를 사용하여 CU를 TU로 더 이상 분할하지 않을 수도 있다. 이와 같이, 일례에서, CU 는 단일 TU 를 포함한다.

[0243] 변환 프로세싱 유닛 (104) 은, TU 의 변환 블록들에 하나 이상의 변환들을 적용함으로써 CU 의 각각의 TU 에 대해 변환 계수 블록들을 생성할 수도 있다. 변환 프로세싱 유닛 (104) 은 TU 와 연관된 변환 블록에 다양한 변환들을 적용할 수도 있다. 예를 들어, 변환 프로세싱 유닛 (104) 은 이산 코사인 변환 (DCT), 지향성 변환, 또는 개념적으로 유사한 변환을 변환 블록에 적용할 수도 있다. 일부 예들에서, 변환 프로세싱 유닛 (104) 은 변환 블록에 변환들을 적용하지 않는다. 그러한 예들에서, 변환 블록은 변환 계수 블록으로 다루어질 수도 있다.

[0244] 양자화 유닛 (106) 은 계수 블록에 있어서의 변환 계수들을 양자화할 수도 있다. 양자화 프로세스는 변환 계수들의 일부 또는 그 모두와 연관된 비트 심도를 감소시킬 수도 있다. 예를 들어, n-비트 변환 계수는 양자화 동안 m-비트 변환 계수로 라운딩 다운될 수도 있고, 여기서 n은 m보다 더 크다. 양자화 유닛 (106) 은 CU 와 연관된 양자화 파라미터 (QP) 값에 기초하여 CU 의 TU 와 연관된 계수 블록을 양자화할 수도 있다. 비디오 인코더 (22) 는 CU 와 연관된 QP 값을 조정함으로써 CU 와 연관된 계수 블록들에 적용된 양자화도를 조정할 수도 있다. 양자화는 정보의 손실을 가져올 수도 있다. 따라서, 양자화된 변환 계수들은 원래의 것보다 낮은 정확도를 가질 수도 있다.

[0245] 역 양자화 유닛 (108) 및 역 변환 프로세싱 유닛 (110) 은 각각 계수 블록에 역 양자화 및 역 변환들을 적용하여, 계수 블록으로부터 잔차 블록을 재구성할 수도 있다. 재구성 유닛 (112) 은 예측 프로세싱 유닛 (100) 에 의해 생성된 하나 이상의 예측 블록들로부터 대응하는 샘플들에 재구성된 잔차 블록을 가산함으로써, TU 와 연관된 재구성된 변환 블록을 생성할 수도 있다. 이러한 방식으로 CU 의 각각의 TU 에 대한 변환 블록들을 복원함으로써, 비디오 인코더 (22) 는 CU 의 코딩 블록들을 복원할 수도 있다.

[0246] 필터 유닛 (114) 은 CU 와 연관된 코딩 블록들에서 블록킹 아티팩트들을 감소시키기 위하여 하나 이상의 디블록킹 동작들을 수행할 수도 있다. 디코딩된 픽처 버퍼 (116) 는, 필터 유닛 (114) 이 복원된 코딩 블록들에 하나 이상의 디블록킹 동작들을 수행한 이후 복원된 코딩 블록들을 저장할 수도 있다. 인터 예측 프로세싱 유닛 (120) 은 다른 화상들의 PU 들에 대해 인터 예측을 수행하기 위하여 재구성된 코딩 블록들을 포함하는 참조 화상을 사용할 수도 있다. 부가적으로, 인트라-예측 프로세싱 유닛 (126) 은 디코딩된 픽처 버퍼 (116) 에 있어서의 복원된 코딩 블록들을 이용하여, CU 와 동일한 픽처에 있어서의 다른 PU들에 대해 인트라 예측을 수행할 수도 있다.

[0247] 엔트로피 인코딩 유닛 (118) 은 비디오 인코더 (22) 의 다른 기능 컴포넌트들로부터 데이터를 수신할 수도 있다. 예를 들어, 엔트로피 인코딩 유닛 (118) 은 양자화 유닛 (106) 으로부터 계수 블록들을 수신할 수도 있고 예측 프로세싱 유닛 (100) 으로부터 신택스 엘리먼트들을 수신할 수도 있다. 엔트로피 인코딩 유닛 (118) 은 인트로피 인코딩된 데이터를 생성하기 위하여 데이터에 대해 하나 이상의 엔트로피 인코딩 동작들을 수행할 수도 있다. 예를 들어, 엔트로피 인코딩 유닛 (118) 은 CABAC 동작, 컨텍스트 적응형 가변 길이 코딩 (CAVLC) 동작, V2V (variable-to-variable) 길이 코딩 동작, 신택스 기반 컨텍스트 적응형 바이너리 산술 코딩 (SBAC) 동작, 확률 간격 파티셔닝 엔트로피 (PIPE) 코딩 동작, 지수-골롬 인코딩 동작, 또는 다른 타입의 엔트로피 인코딩 동작을 데이터에 대해 수행할 수도 있다. 비디오 인코더 (22) 는 엔트로피 인코딩 유닛 (118) 에 의해 생성된 엔트로피 인코딩된 데이터를 포함하는 비트스트림을 출력할 수도 있다. 예를 들어, 비트 스트림은 본 개시의 기술에 따라 CU에 대한 파티션 구조를 나타내는 데이터를 포함할 수도 있다.

[0248] 도 11 은 본 개시의 기술들을 구현하도록 구성된 예시적인 비디오 디코더 (30) 를 도시한 블록 다이어그램이다. 도 11 은 설명의 목적들을 위해 제공되며, 본 개시에서 넓게 예시화되고 설명된 바와 같은 기법들에 대해 한정하는 것은 아니다. 설명의 목적들로, 본 개시는 HEVC 코딩의 맥락에서 비디오 디코더 (30) 를 설명한다. 하지만, 본 개시의 기법들은 다른 코딩 표준들 또는 방법들에 적용가능할 수도 있다.

[0249] 도 11 의 예에서, 비디오 디코더 (30) 는 엔트로피 디코딩 유닛 (150), 비디오 데이터 메모리 (151), 예측 프로세싱 유닛 (152), 역양자화 유닛 (154), 역변환 프로세싱 유닛 (156), 재구성 유닛 (158), 필터 유닛 (160), 및 디코딩된 픽처 버퍼 (162) 를 포함한다. 예측 프로세싱 유닛 (152) 은, 모션 보상 유닛 (164) 및 인트라 예측 프로세싱 유닛 (166) 을 포함한다. 다른 예들에서, 비디오 디코더 (30) 는, 더 많거나, 더 적거나, 또는 상

이한 기능 컴포넌트들을 포함할 수도 있다.

[0250] 비디오 데이터 메모리 (151) 는 비디오 디코더 (30) 의 컴포넌트들에 의해 디코딩될 인코딩된 비디오 데이터, 이를테면 인코딩된 비디오 비트스트림을 저장할 수도 있다. 비디오 데이터 메모리 (151) 에 저장된 비디오 데이터는, 예를 들어, 컴퓨터 판독가능 매체 (16) 로부터, 예를 들어, 카메라와 같은 로컬 비디오 소스로부터, 비디오 데이터의 유선 또는 무선 네트워크 통신을 통해, 또는 물리적 데이터 저장 매체에 액세스하는 것에 의해, 획득될 수도 있다. 비디오 데이터 메모리 (151) 는 인코딩된 비디오 비트스트림으로부터 인코딩된 비디오 데이터를 저장하는 코딩된 화상 버퍼 (CPB) 를 형성할 수도 있다. 디코딩된 픽처 버퍼 (162) 는, 예를 들어, 인트라- 또는 인터-코딩 모드들에서 또는 출력에 대하여 비디오 디코더 (30) 에 의해 비디오 데이터를 디코딩함에 있어서 사용하기 위한 레퍼런스 비디오 데이터를 저장하는 레퍼런스 픽처 메모리일 수도 있다. 비디오 데이터 메모리 (151) 및 디코딩된 픽처 버퍼 (162) 는 동기식 DRAM (SDRAM) 을 포함한 동적 랜덤 액세스 메모리 (DRAM), 자기저항성 RAM (MRAM), 저항성 RAM (RRAM), 또는 다른 타입들의 메모리 디바이스들과 같은 다양한 메모리 디바이스들 중 임의의 메모리 디바이스에 의해 형성될 수도 있다. 비디오 데이터 메모리 (151) 및 디코딩된 픽처 버퍼 (162) 는 동일한 메모리 디바이스 또는 별도의 메모리 디바이스들에 의해 제공될 수도 있다. 다양한 예들에 있어서, 비디오 데이터 메모리 (151) 는 비디오 디코더 (30) 의 다른 컴포넌트들과 온-칩형이거나 또는 그 컴포넌트들에 대하여 오프-칩형일 수도 있다. 비디오 데이터 메모리 (151) 는 도 1 의 저장 매체 (28) 와 동일하거나 또는 그것의 일부일 수도 있다.

[0251] 비디오 데이터 메모리 (151) 는 비트스트림의 인코딩된 비디오 데이터 (예를 들어, NAL 유닛) 를 수신하여 저장한다. 엔트로피 디코딩 유닛 (150) 은 비디오 데이터 메모리 (151) 로부터 인코딩된 비디오 데이터 (예를 들어, NAL 유닛) 를 수신할 수도 있고 신택스 엘리먼트들을 얻기 위해 NAL 유닛을 파싱할 수도 있다. 엔트로피 디코딩 유닛 (150) 은 NAL 유닛들에서 엔트로피 인코딩된 신택스 엘리먼트들을 엔트로피 디코딩할 수도 있다. 예측 프로세싱 유닛 (152), 역 양자화 유닛 (154), 역 변환 프로세싱 유닛 (156), 재구성 유닛 (158), 및 필터 유닛 (160) 은 비트스트림으로부터 추출된 신택스 엘리먼트들에 기초하여 디코딩된 비디오 데이터를 생성할 수도 있다. 엔트로피 디코딩 유닛 (150) 은 일반적으로 엔트로피 인코딩 유닛 (118) 의 프로세스와 상반되는 프로세스를 수행할 수도 있다.

[0252] 본 개시의 일부 예들에 따르면, 엔트로피 디코딩 유닛 (150), 또는 비디오 디코더 (30) 의 다른 프로세싱 유닛은 비트스트림으로부터 신택스 엘리먼트들을 획득하는 일부로서 트리 구조를 결정할 수도 있다. 트리 구조는 CTB 와 같은 초기 비디오 블록이 코딩 유닛들과 같은 더 작은 비디오 블록들로 파티셔닝되는 방법을 특정할 수도 있다. 본 개시의 하나 이상의 기술들에 따르면, 트리 구조의 각 깊이 레벨에서의 트리 구조의 각각의 개별 비-리프 노드에 대해, 개별 비리프 노드에 대한 복수의 허용된 파티션 유형들이 존재하고, 개별 비리프 노드에 대응하는 비디오 블록은 복수의 허용 가능한 분할 패턴들 중 하나에 따라 개별 비-리프 노드의 자식 노드들에 대응하는 비디오 블록들로 파티셔닝된다.

[0253] 비트스트림으로부터 신택스 엘리먼트들을 얻는 것에 더하여, 비디오 디코더 (30) 는 파티셔닝되지 않은 CU 에 대해 재구성 동작을 수행할 수도 있다. CU 에 대해 재구성 동작을 수행하기 위하여, 비디오 디코더 (30) 는 CU 의 각각의 TU 에 대해 재구성 동작을 수행할 수도 있다. CU 의 각각의 TU 에 대해 재구성 동작을 수행함으로써, 비디오 디코더 (30) 는 CU 의 잔차 블록들을 재구성할 수도 있다. 상술된 바와 같이, 본 개시의 일례에서, CU 는 단일 TU 를 포함한다.

[0254] CU 의 TU 에 대해 재구성 동작을 수행하는 부분으로서, 역 양자화 유닛 (154) 은 TU 와 연관된 계수 블록들을, 역 양자화, 즉, 탈 양자화할 수도 있다. 역양자화 유닛 (154) 이 계수 블록을 역양자화한 이후, 역변환 프로세싱 유닛 (156) 은 TU 와 연관된 잔차 블록을 생성하기 위하여 계수 블록에 하나 이상의 역변환들을 적용할 수도 있다. 예를 들어, 역변환 프로세싱 유닛 (156) 은 역 DCT, 역 정수 변환, 역 KLT (Karhunen-Loeve transform), 역 회전 변환, 역 지향성 변환, 또는 다른 역변환을 계수 블록에 적용할 수도 있다.

[0255] CU 또는 PU 가 인트라 예측을 이용하여 인코딩되면, 인트라 예측 프로세싱 유닛 (166) 은 PU 의 예측 블록들을 생성하기 위하여 인트라 예측을 수행할 수도 있다. 인트라 예측 프로세싱 유닛 (166) 은, 공간적으로 이웃하는 블록들 샘플들에 기초하여 PU 의 예측 블록들을 생성하기 위하여 인트라 예측 모드를 사용할 수도 있다. 인트라 예측 프로세싱 유닛 (166) 은 비트스트림으로부터 획득된 하나 이상의 신택스 엘리먼트들에 기초하여 PU 를 위한 인트라 예측 모드를 결정할 수도 있다.

[0256] PU 가 인터 예측을 이용하여 인코딩되면, 엔트로피 디코딩 유닛 (150) 은 PU 를 위한 모션 정보를 결정할 수도 있다. 모션 보상 유닛 (164) 은 PU 의 모션 정보에 기초하여, 하나 이상의 참조 블록들을 결정할 수도 있다.

모션 보상 유닛 (164) 은, 하나 이상의 참조 블록들에 기초하여, PU 를 위한 예측 블록들 (예를 들어, 예측 루마, Cb 및 Cr 블록들) 을 생성할 수도 있다. 상술된 바와 같이, MTT 파티셔닝을 사용하는 본 개시의 일례에서, CU 는 단일 PU 만을 포함할 수도 있다. 즉, CU 는 다수의 PU 로 분할되지 않을 수 있다.

[0257] 재구성 유닛 (158) 은, CU 에 대한 코딩 블록들 (예를 들어, 루마, Cb 및 Cr 코딩 블록들) 을 재구성하기 위하여, 적용가능한 바에 따라, CU 의 TU 들에 대한 변환 블록들 (예를 들어, 루마, Cb 및 Cr 변환 블록들) 및 CU 의 PU 들의 예측 블록들 (예를 들어, 루마, Cb 및 Cr 블록들), 즉, 인트라 예측 데이터 또는 인터 예측 데이터 중의 어느 하나를 이용할 수도 있다. 예를 들어, 재구성 유닛 (158) 은 CU 의 코딩 블록들 (예를 들어, 루마, Cb 및 Cr 코딩 블록들) 을 재구성하기 위하여 변환 블록들 (예를 들어, 루마, Cb 및 Cr 변환 블록들) 의 샘플들을 예측 블록들 (예를 들어, 루마, Cb 및 Cr 예측 블록들) 의 대응하는 샘플들에 추가할 수도 있다.

[0258] 필터 유닛 (160) 은 CU 의 코딩 블록들과 연관된 블록킹 아티팩트들을 감소시키기 위하여 디블로킹 동작을 수행할 수도 있다. 비디오 디코더 (30) 는 CU 의 코딩 블록을 디코딩된 픽처 버퍼 (162) 에 저장할 수도 있다. 디코딩된 픽처 버퍼 (162) 는 후속 모션 보상, 인트라 예측, 및 도 1 의 디스플레이 디바이스 (32) 와 같은 디스플레이 디바이스 상으로의 프리젠테이션을 위해 참조 픽처들을 제공할 수도 있다. 예를 들어, 비디오 디코더 (30) 는, 디코딩된 픽처 버퍼 (162) 에서의 블록들에 기초하여, 다른 CU들의 PU들에 대해 인트라 예측 또는 인터 예측 동작들을 수행할 수도 있다.

[0259] 도 12 는 본 개시물의 일 예의 인코딩 방법을 도시하는 플로우챠트이다. 도 12 의 기법들은 변환 프로세싱 유닛 (104) 및/또는 양자화 유닛 (106) 을 포함하는 비디오 인코더 (22) 에 의해 수행될 수도 있다.

[0260] 본 개시의 일례에서, 비디오 인코더 (22) 는 비디오 데이터의 블록을 수신하고 (200), 잔차 비디오 데이터를 생성하기 위해 비디오 데이터의 블록을 예측하도록 구성될 수도 있다 (202). 비디오 인코더 (22) 는 잔치 비디오 데이터에 대한 변환을 결정하도록 추가로 구성될 수도 있으며, 상기 변환은 2 의 거듭제곱이 아닌 크기 S를 가지며 (204), 변경된 크기 (S') 를 갖는 변환을 생성하는 2 의 거듭제곱으로 S 를 라운딩하도록 구성될 수도 있다 (206). 비디오 인코더 (22) 는 변경된 크기 (S') 를 갖는 변환을 잔차 비디오 데이터에 적용하여 변환 계수를 생성하고 (208), 인코딩된 비디오 비트 스트림에 변환 계수를 인코딩할 수도 있다 (210).

[0261] 다른 예에서, 비디오 인코더 (22) 는 가장 가까운 2 의 거듭제곱으로 S 를 라운딩하도록 구성될 수도 있다.

[0262] 다른 예에서, 비디오 인코더 (22) 는 변환 계수를 양자화하도록 구성될 수도 있다.

[0263] 다른 예에서, 비디오 데이터의 블록은 비정사각형 형상을 갖는다.

[0264] 다른 예에서, S 는 12이고, 비디오 인코더 (22) 는 12 를 16 으로 라운딩하도록 구성될 수 있으며, 여기서 변경된 크기 (S') 는 16 이다. 다른 예에서, S 는 24이고, 비디오 인코더 (22) 는 24 를 32 으로 라운딩하도록 구성될 수 있으며, 여기서 변경된 크기 (S') 는 32 이다.

[0265] 일 예에서, S는 변환의 폭이다. 다른 예에서, S는 변환의 높이이다.

[0266] 도 13 은 본 개시의 예시의 디코딩 방법을 도시하는 플로우챠트이다. 도 13 의 기법들은 변환 프로세싱 유닛 (156) 및/또는 양자화 유닛 (154) 을 포함하는 비디오 디코더 (30) 에 의해 수행될 수도 있다.

[0267] 본 개시의 일 예에서, 비디오 디코더 (30) 는 비디오 데이터의 인코딩된 블록을 수신하고 (300); 비디오 데이터의 인코딩된 블록에 대한 변환을 결정하는 것으로서, 그 변환은 2 의 거듭제곱이 아닌 크기 (S) 를 가지는, 상기 변환을 결정하며 (302); 변경된 크기 (S') 를 갖는 역변환을 생성하는 2 의 거듭제곱으로 S 를 라운딩하도록 구성될 수도 있다 (304). 비디오 디코더 (30) 는 또한 변경된 크기 (S') 를 갖는 역변환을 비디오 데이터의 인코딩된 블록에 적용하여 잔차 비디오 데이터를 생성하고 (306), 잔차 비디오 데이터를 디코딩하여 비디오 데이터의 디코딩된 블록을 생성하도록 구성될 수도 있다 (308).

[0268] 일 예에서, 비디오 디코더 (30) 는 가장 가까운 2 의 거듭제곱으로 S 를 라운딩하도록 구성될 수도 있다. 다른 예에서, S 는 12이고, 비디오 디코더 (30) 는 12 를 16 으로 라운딩하도록 구성될 수 있으며, 여기서 변경된 크기 (S') 는 16 이다. 다른 예에서, S 는 24이고, 비디오 디코더 (30) 는 24 를 32 으로 라운딩하도록 구성될 수 있으며, 여기서 변경된 크기 (S') 는 32 이다.

[0269] 다른 예시에서, 비디오 데이터의 인코딩된 블록은 역양자화된 변환 계수를 포함한다. 다른 예에서, 비디오 데이터의 인코딩된 블록은 비정사각형 형상을 갖는다. 일 예에서, S는 변환의 폭이다. 다른 예에서, S는 변환의 높이이다.

[0270] 본 발명의 다른 예에서, 비디오 디코더 (30)는 S' 에 기초하여 역 변환에 대한 시프트 값을 결정하도록 구성될 수도 있다.

[0271] 본 개시의 특정 양태들이 예시의 목적을 위해 HEVC 표준의 확장들에 관하여 설명되었다. 하지만, 본 개시에 설명된 기법들은, 아직 개발되지 않은 다른 표준 또는 사유 (proprietary) 비디오 코딩 프로세스들을 포함한, 다른 비디오 코딩 프로세스들에 유용할 수도 있다.

[0272] 본 개시에서 설명된 바와 같은 비디오 코더는 비디오 인코더 또는 비디오 디코더를 지칭할 수도 있다. 유사하게, 비디오 코딩 유닛은 비디오 인코더 또는 비디오 디코더를 지칭할 수도 있다. 마찬가지로, 비디오 코딩은, 적용가능한 바에 따라, 비디오 인코딩 또는 비디오 디코딩을 지칭할 수도 있다. 이 개시물에서, 문구 "~ 에 기초하여" 는 오직 ~ 기초하여, 적어도 부분적으로 기초하여, 또는 어떤 방식으로 기초하여를 나타낼 수도 있다. 본 개시는 하나 이상의 샘플 블록들 및 하나 이상의 샘플 블록들의 샘플들을 코딩하는데 이용된 신택스 구조들을 지칭하기 위하여 용어 "비디오 유닛", "비디오 블록" 또는 "블록" 을 이용할 수도 있다. 비디오 유닛의 예시적인 타입은 CTU, CU, PU, 변환 유닛 (TU), 매크로블록, 매크로블록 파티션 등을 포함할 수도 있다. 일부 맥락에서는, PU에 대한 논의가 매크로블록 또는 매크로블록 파티션에 대한 논의와 상호 교환될 수도 있다. 비디오 블록들의 예시적인 타입들은 코딩 트리 블록들, 코딩 블록들, 및 다른 타입들의 비디오 데이터의 블록들을 포함할 수도 있다.

[0273] 예에 의존하여, 본 명세서에서 설명된 기법들의 임의의 특정 행위들 또는 이벤트들은 상이한 시퀀스로 수행될 수 있고, 전체적으로 부가되거나 병합되거나 또는 제거될 수도 있음 (예를 들어, 설명된 모든 행위들 또는 이벤트들이 그 기법들의 실시를 위해 필수적인 것은 아님) 이 인식되어야 한다. 더욱이, 특정 예들에 있어서, 행위들 또는 이벤트들은 순차적인 것보다는, 예를 들어, 다중-스레딩된 프로세싱, 인터럽트 프로세싱, 또는 다중의 프로세서들을 통해 동시에 수행될 수도 있다.

[0274] 하나 이상의 예들에서, 설명된 기능들은 하드웨어, 소프트웨어, 펌웨어, 또는 그 임의의 조합으로 구현될 수도 있다. 소프트웨어로 구현되면, 그 기능들은 컴퓨터 판독가능 매체 상의 하나 이상의 명령 또는 코드로서 저장되거나 송신될 수도 있고 하드웨어 기반 프로세싱 유닛에 의해 실행될 수도 있다. 컴퓨터 판독가능 매체는, 데이터 저장 매체와 같은 유형의 매체에 대응하는 컴퓨터 판독가능 저장 매체, 또는 예를 들면, 통신 프로토콜에 따라, 일 장소로부터 다른 장소로의 컴퓨터 프로그램의 전송을 가능하게 하는 임의의 매체를 포함하는 통신 매체를 포함할 수도 있다. 이런 방식으로, 컴퓨터 판독가능 매체는 일반적으로, (1) 비일시적인 유형의 컴퓨터 판독가능 저장 매체 또는 (2) 신호 또는 캐리어 파와 같은 통신 매체에 대응할 수도 있다. 데이터 저장 매체들은 본 개시에서 설명된 기법들의 구현을 위한 명령들, 코드 및/또는 데이터 구조들을 취출하기 위해 하나 이상의 컴퓨터들 또는 하나 이상의 프로세서들에 의해 액세스될 수 있는 임의의 이용가능한 매체들일 수도 있다. 컴퓨터 프로그램 제품이 컴퓨터 판독가능 매체를 포함할 수도 있다.

[0275] 제한이 아닌 일 예로, 이러한 컴퓨터 판독가능 저장 매체들은 RAM, ROM, EEPROM, CD-ROM 또는 다른 광학 디스크 스토리지, 자기 디스크 스토리지, 또는 다른 자기 저장 디바이스들, 플래시 메모리, 또는 명령들 또는 데이터 구조들의 형태로 원하는 프로그램 코드를 저장하는데 사용될 수 있고 컴퓨터에 의해 액세스될 수 있는 임의의 다른 매체를 포함할 수 있다. 또한, 임의의 커넥션이 컴퓨터 판독가능 매체로 적절히 명명된다. 예를 들어, 동축 케이블, 광섬유 케이블, 꼬임쌍선, 디지털 가입자 라인 (DSL), 또는 적외선, 무선, 및 마이크로파와 같은 무선 기술들을 이용하여 웹사이트, 서버, 또는 다른 원격 소스로부터 소프트웨어가 송신된다면, 동축 케이블, 광섬유 케이블, 꼬임쌍선, DSL, 또는 적외선, 무선, 및 마이크로파와 같은 무선 기술들은 매체의 정의에 포함된다. 하지만, 컴퓨터 판독가능 저장 매체 및 데이터 저장 매체는 접속, 캐리어 파, 신호 또는 다른 일시적 매체를 포함하는 것이 아니라, 대신에 비일시적, 유형의 저장 매체에 관련된다는 것이 이해되야 한다. 본 명세서에서 사용된 바와 같은 디스크 (disk) 및 디스크 (disc) 는 컴팩트 디스크 (CD), 레이저 디스크, 광학 디스크, 디지털 다기능 디스크 (DVD), 플로피 디스크 및 블루레이 디스크를 포함하며, 여기서, 디스크 (disk) 는 통상적으로 데이터를 자기적으로 재생하지만 디스크 (disc) 는 레이저를 이용하여 데이터를 광학적으로 재생한다. 상기의 조합들이 또한, 컴퓨터 판독가능 매체들의 범위 내에 포함되어야 한다.

[0276] 명령들은 하나 이상의 디지털 신호 프로세서들 (DSP들), 범용 마이크로프로세서들, 주문형 집적회로들 (ASIC들), 필드 프로그래밍가능 로직 어레이들 (FPGA들), 또는 다른 등가의 집적된 또는 별도의 로직 회로부와 같은 하나 이상의 프로세서들에 의해 실행될 수도 있다. 이에 따라, 본 명세서에서 사용된 바와 같은 용어 "프로세서" 는 본 명세서에서 설명된 기법들의 구현에 적합한 전술한 구조 또는 임의의 다른 구조 중 임의의 구조를 지칭할 수도 있다. 추가로, 일부 예들에서, 본 명세서에서 설명된 기능성은 인코딩 및 디코딩을 위해

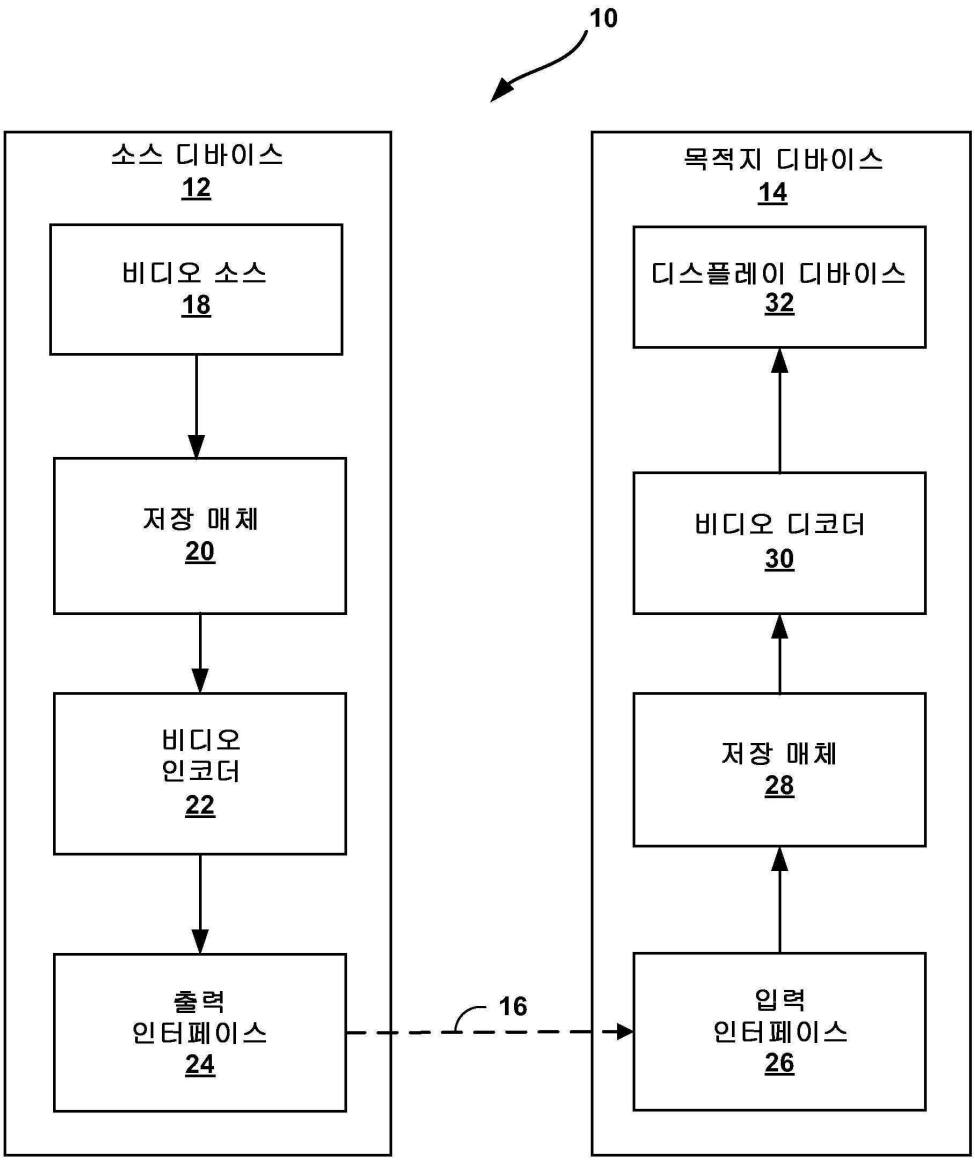
구성되거나, 또는 결합된 코덱에 통합된 전용 하드웨어 및/또는 소프트웨어 모듈들 내에 제공될 수도 있다. 또한, 그 기법들은 하나 이상의 회로들 또는 로직 엘리먼트들에서 완전히 구현될 수 있다.

[0277] 본 개시의 기법들은 무선 핸드셋, 집적 회로 (IC) 또는 IC 들의 세트 (예를 들면, 칩 세트) 를 포함하는, 매우 다양한 디바이스들 또는 장치들에서 구현될 수도 있다. 다양한 컴포넌트들, 모듈들, 또는 유닛들은 개시된 기법들을 수행하도록 구성된 디바이스들의 기능적 양태들을 강조하기 위해 본 개시에 설명되지만, 상이한 하드웨어 유닛들에 의한 실현을 반드시 요구하는 것은 아니다. 오히려, 상기 설명된 바와 같이, 다양한 유닛들은 코덱 하드웨어 유닛에서 결합되거나 또는 적합한 소프트웨어 및/또는 펌웨어와 함께, 상기 설명된 바와 같은 하나 이상의 프로세서들을 포함하는, 상호동작가능한 하드웨어 유닛들의 콜렉션에 의해 제공될 수도 있다.

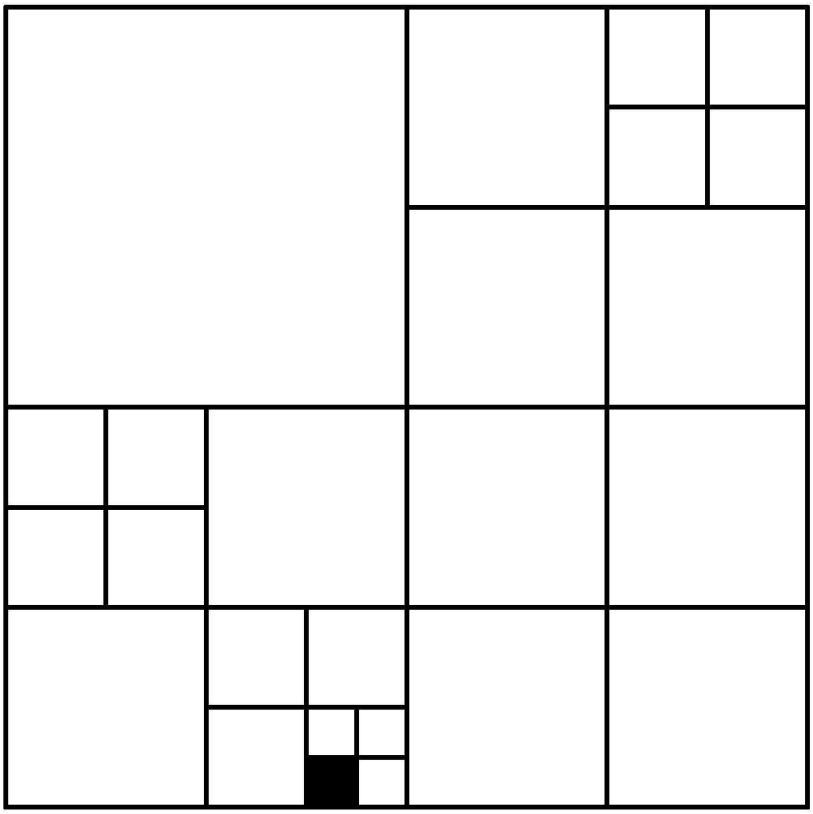
[0278] 다양한 예들이 설명되었다. 이들 및 다른 예들은 다음의 청구항들의 범위 내에 있다.

도면

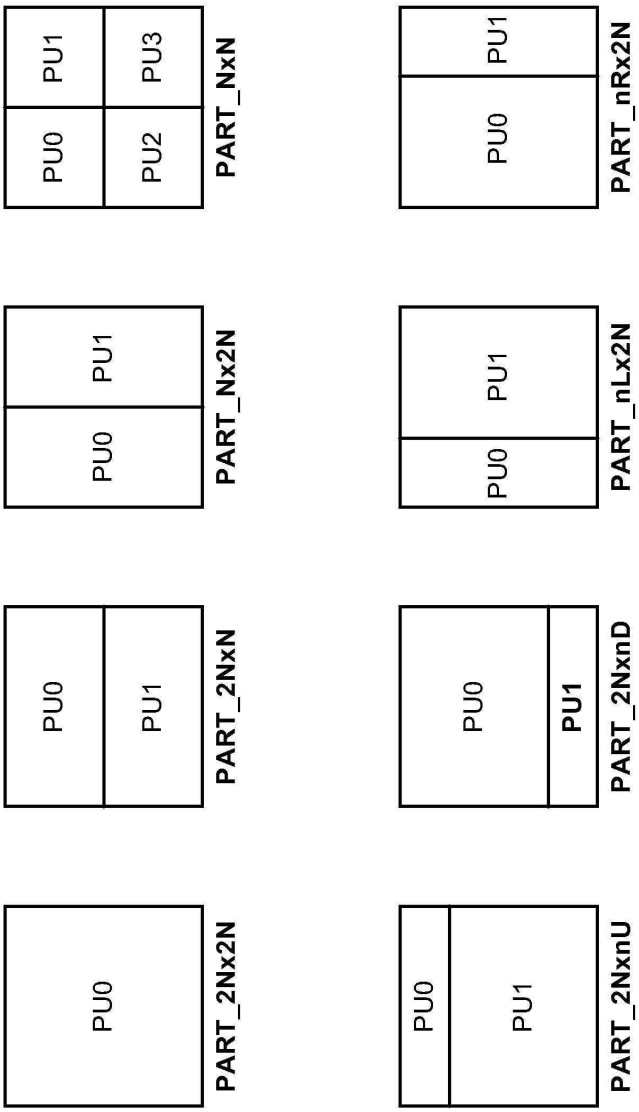
도면1



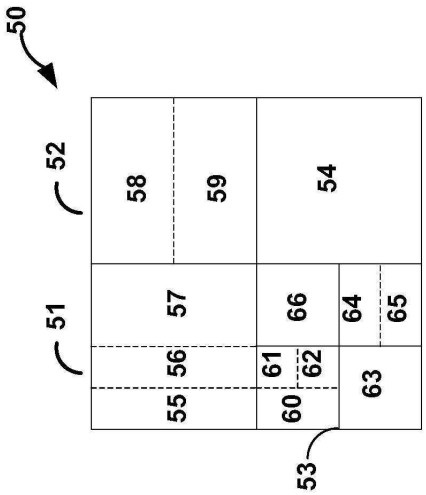
도면2



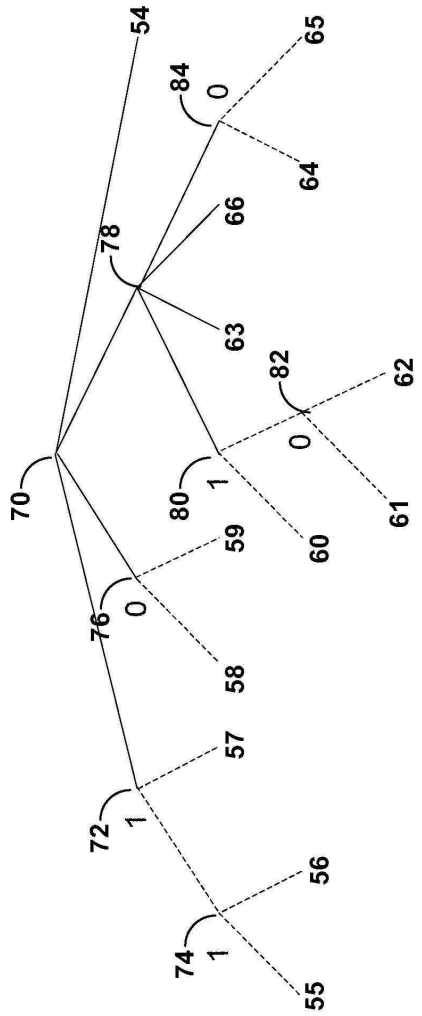
도면3


PU0
PU1
PU2
PU3
PART_NxN
PART_Nx2N
PART_2NxN
PART_2Nx2N
PART_nRx2N
PART_nLx2N
PART_2NxnD
PART_2NxnU

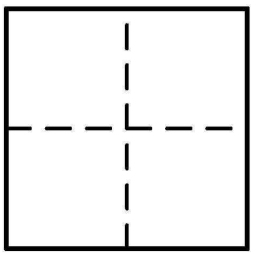
도면4a


50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66

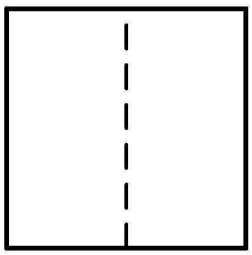
도면4b



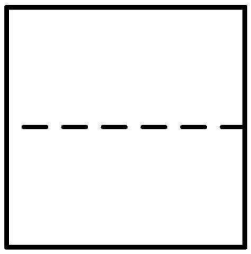
도면5a



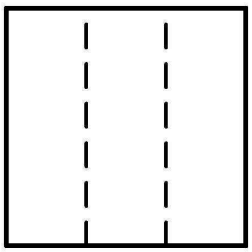
도면5b



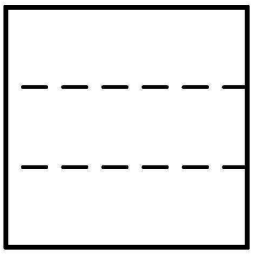
도면5c



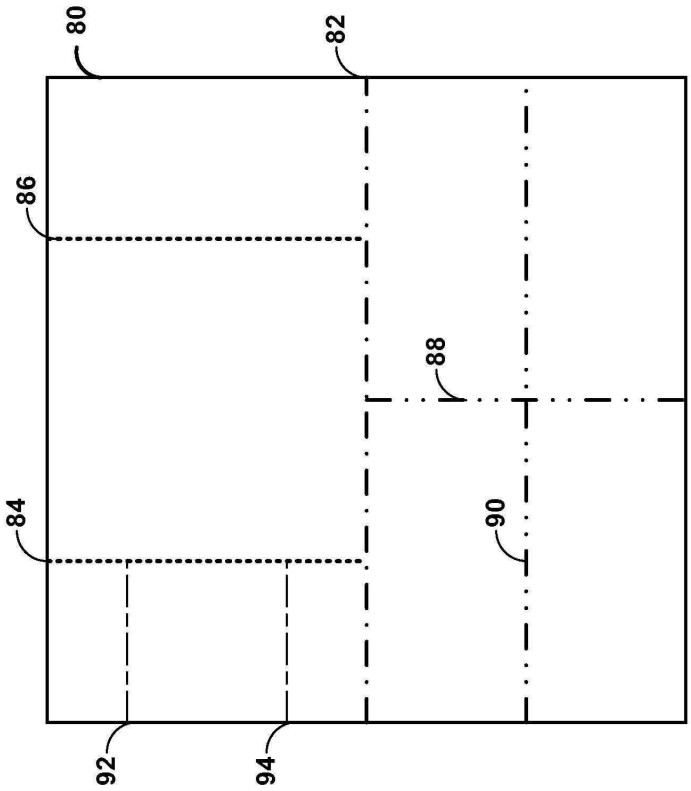
도면5d



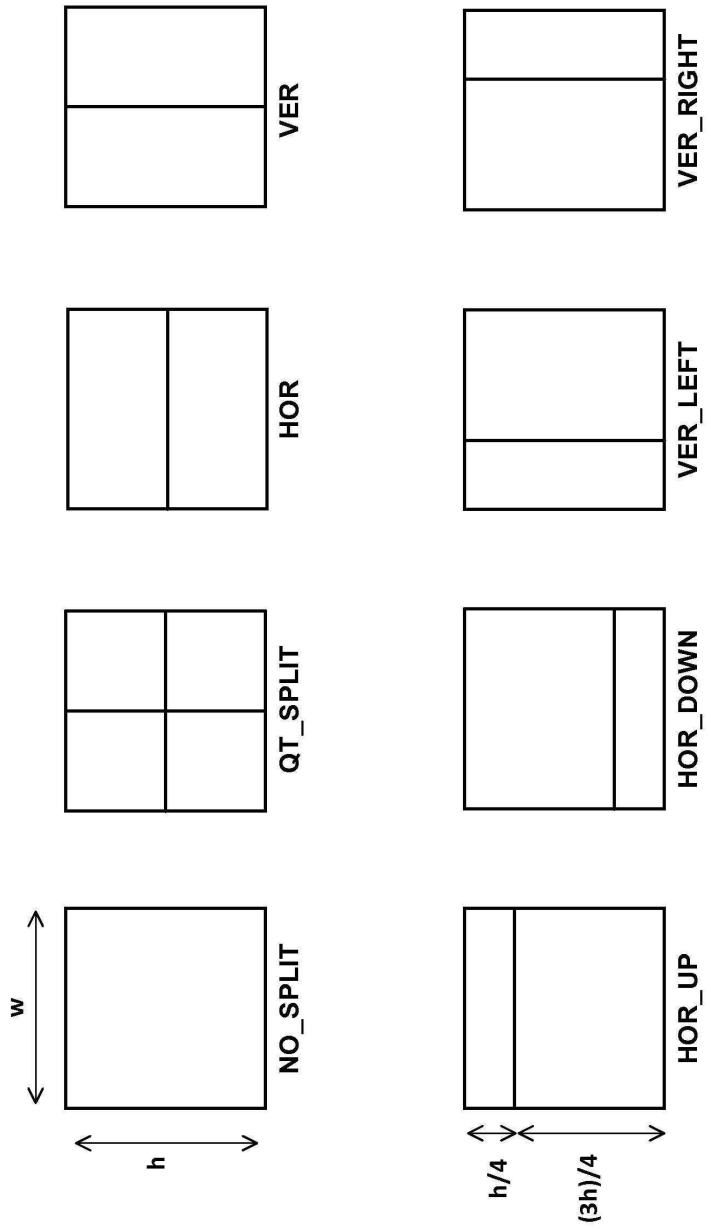
도면5e



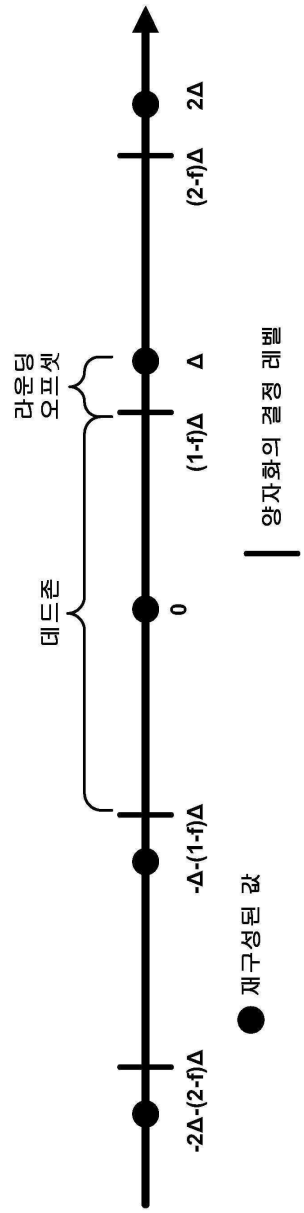
도면6



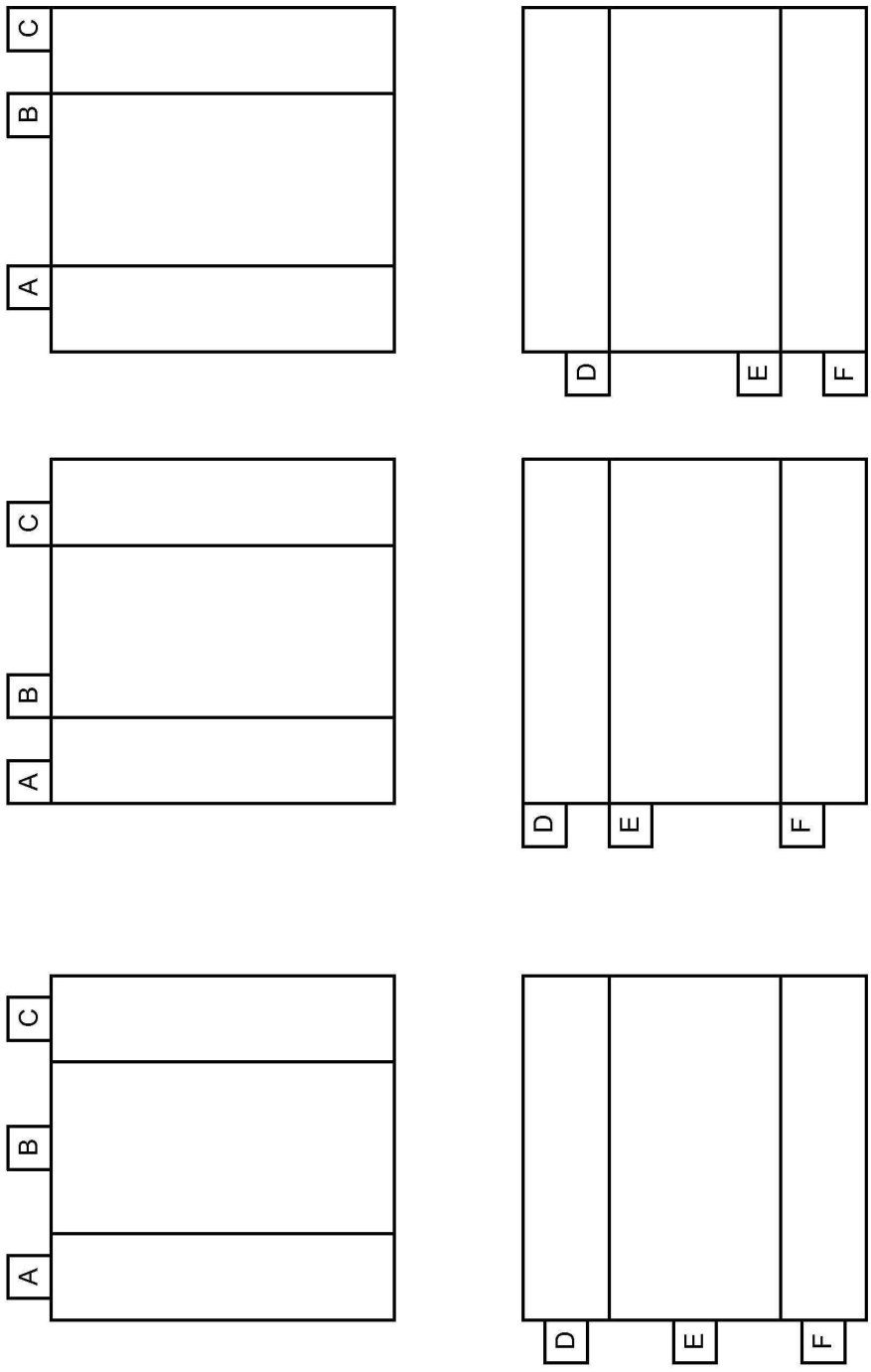
도면7


VER
VER_RIGHT
HOR
VER_LEFT
QT_SPLIT
HOR_DOWN
NO_SPLIT
HOR_UP
w
h
h/4
(3h)/4

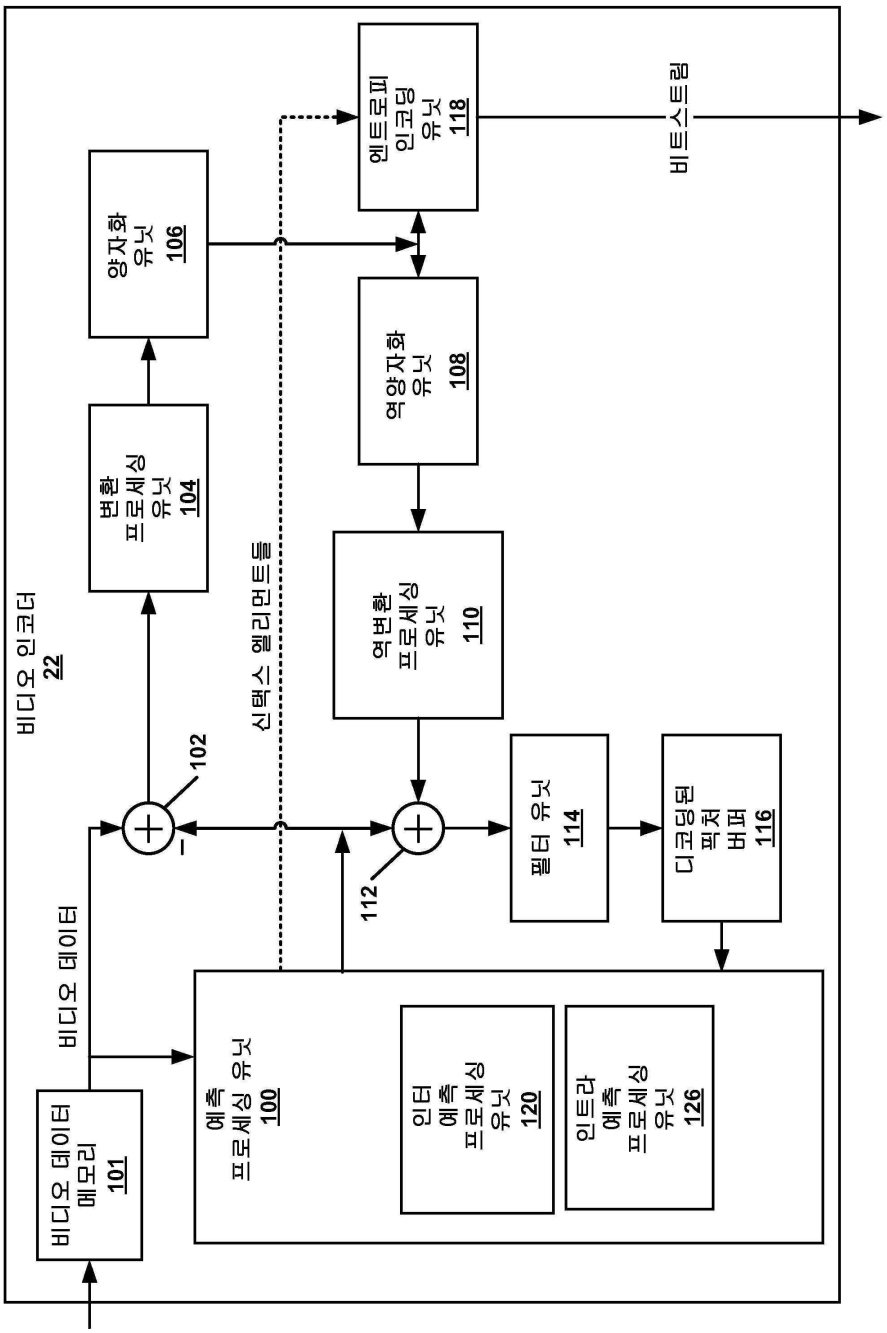
도면8



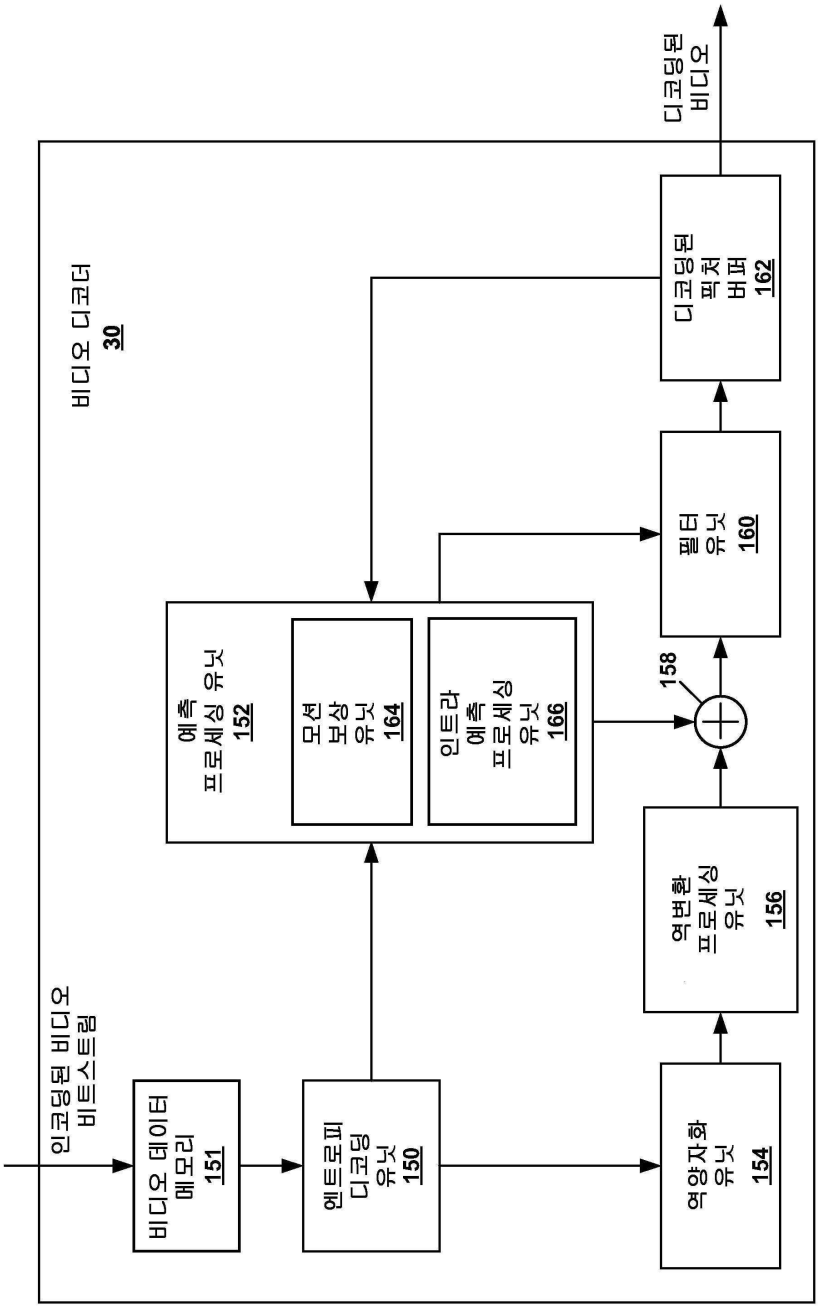
도면9



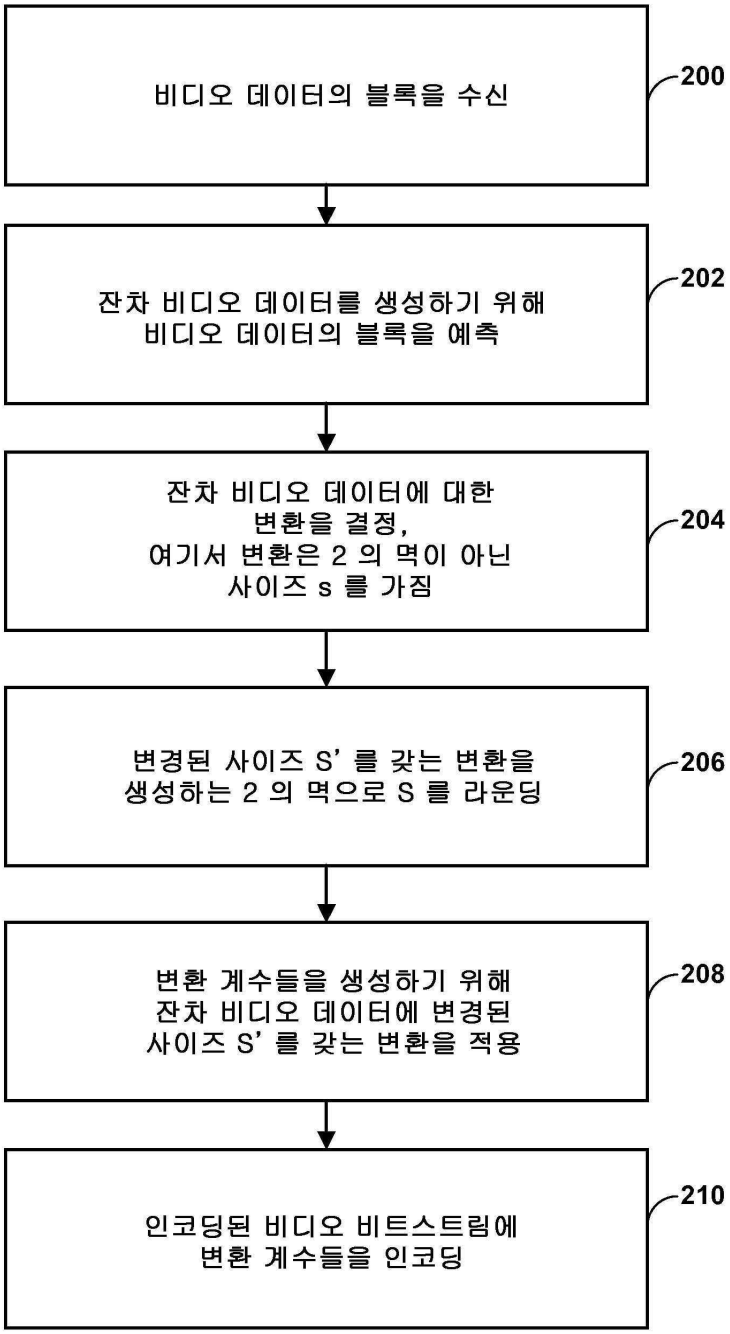
도면10


비디오 인코더
22
비디오 데이터 메모리
101
비디오 데이터
예측 프로세싱 유닛
100
인터 예측 프로세싱 유닛
120
인트라 예측 프로세싱 유닛
126
신택스 엘리먼트들
변환 프로세싱 유닛
104
양자화 유닛
106
엔트로피 인코딩 유닛
118
비트스트림
역양자화 유닛
108
역변환 프로세싱 유닛
110
102
112
필터 유닛
114
디코딩된 픽처 버퍼
116

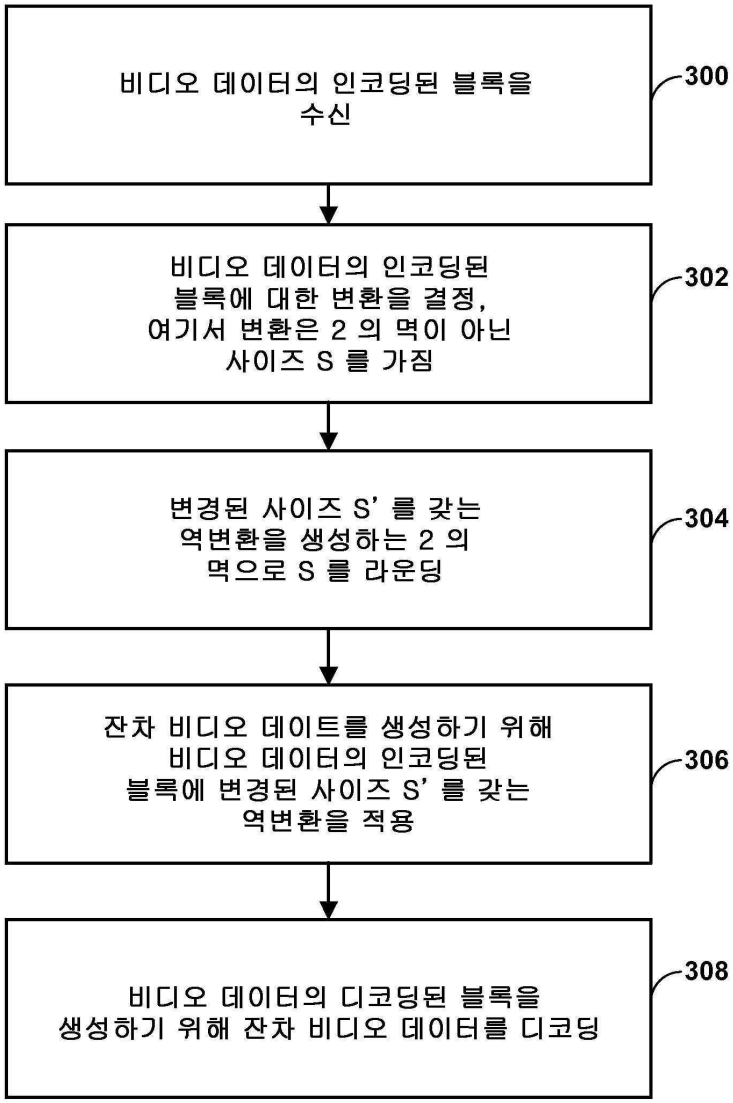
도면11


인코딩된 비디오
비트스트림
비디오 디코더
30
비디오 데이터
메모리
151
엔트로피
디코딩
유닛
150
역양자화
유닛
154
역변환
프로세싱
유닛
156
158
예측
프로세싱 유닛
152
모션
보상
유닛
164
인트라
예측
프로세싱
유닛
166
필터
유닛
160
디코딩된
픽처
버퍼
162
디코딩된
비디오

도면12


비디오 데이터의 블록을 수신
200
잔차 비디오 데이터를 생성하기 위해
비디오 데이터의 블록을 예측
202
잔차 비디오 데이터에 대한
변환을 결정,
여기서 변환은 2 의 멱이 아닌
사이즈 s 를 가짐
204
변경된 사이즈 S' 를 갖는 변환을
생성하는 2 의 멱으로 S 를 라운딩
206
변환 계수들을 생성하기 위해
잔차 비디오 데이터에 변경된
사이즈 S' 를 갖는 변환을 적용
208
인코딩된 비디오 비트스트림에
변환 계수들을 인코딩
210

도면13


비디오 데이터의 인코딩된 블록을 수신
300
비디오 데이터의 인코딩된 블록에 대한 변환을 결정, 여기서 변환은 2 의 멱이 아닌 사이즈 S 를 가짐
302
변경된 사이즈 S' 를 갖는 역변환을 생성하는 2 의 멱으로 S 를 라운딩
304
잔차 비디오 데이트를 생성하기 위해 비디오 데이터의 인코딩된 블록에 변경된 사이즈 S' 를 갖는 역변환을 적용
306
비디오 데이터의 디코딩된 블록을 생성하기 위해 잔차 비디오 데이터를 디코딩
308