(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2018-0084816
(43) 공개일자 2018년07월25일

(51) 국제특허분류(Int. Cl.)
H04N 19/13 (2014.01) *H04N 19/105* (2014.01)
H04N 19/119 (2014.01) *H04N 19/132* (2014.01)
H04N 19/176 (2014.01)
(52) CPC특허분류
H04N 19/13 (2015.01)
H04N 19/105 (2015.01)
(21) 출원번호 10-2018-7014617
(22) 출원일자(국제) 2016년11월23일
심사청구일자 없음
(85) 번역문제출일자 2018년05월23일
(86) 국제출원번호 PCT/US2016/063680
(87) 국제공개번호 WO 2017/091776
국제공개일자 2017년06월01일
(30) 우선권주장
62/260,103 2015년11월25일 미국(US)
(뒷면에 계속)

(71) 출원인
퀄컴 인코포레이티드
미국 92121-1714 캘리포니아주 샌 디에고 모어하우스 드라이브 5775
(72) 발명자
장 리
미국 92121-1714 캘리포니아주 샌디에고 모어하우스 드라이브 5775
차오 팡
미국 90292 캘리포니아주 마리나 델 레이 비아 마리나 4169 넘버307
(뒷면에 계속)
(74) 대리인
특허법인코리아나

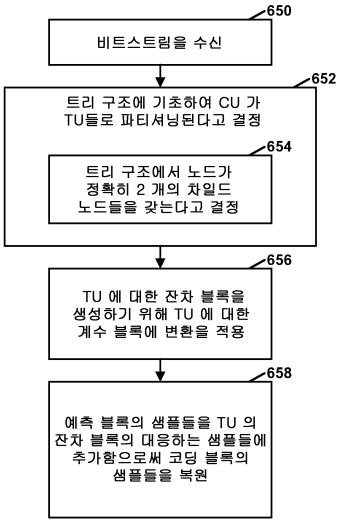
전체 청구항 수 : 총 32 항

(54) 발명의 명칭 **비디오 코딩에서의 플렉서블 변환 트리 구조**

(57) 요 약

비디오 코더는, 코딩 유닛 (CU) 이 트리 구조에 기초하여 CU 의 변환 유닛들 (TU들) 로 파티셔닝된다고 결정한다. CU 가 트리 구조에 기초하여 CU 의 TU들로 파티셔닝된다고 결정하는 것의 부분으로서, 비디오 코더는 트리 구조에서 노드가 트리 구조에서 정확히 2 개의 차일드 노드들을 갖는다고 결정한다. 트리 구조의 루트 노드는 CU 의 코딩 블록에 대응하고, 트리 구조의 각각의 개별의 비-루트 노드는 개별의 비-루트 노드의 페어런트 노드에 대응하는 블록의 파티션인 개별의 블록에 대응하고, 트리 구조의 리프 노드들은 CU 의 TU들에 대응한다.

대 표 도 - 도35



(52) CPC특허분류
H04N 19/119 (2015.01)
H04N 19/132 (2015.01)
H04N 19/176 (2015.01)

(72) 발명자
진 궈신
미국 92121-1714 캘리포니아주 샌디에고 모어하우스 드라이브 5775
천 지안레
미국 92121-1714 캘리포니아주 샌디에고 모어하우스 드라이브 5775
치엔 웨이-정
미국 92121-1714 캘리포니아주 샌디에고 모어하우스 드라이브 5775
자오 신
미국 92121-1714 캘리포니아주 샌디에고 모어하우스 드라이브 5775
리 샹
미국 92121-1714 캘리포니아주 샌디에고 모어하우스 드라이브 5775
카르체비츠 마르타
미국 92121-1714 캘리포니아주 샌디에고 모어하우스 드라이브 5775

(30) 우선권주장
62/310,271 2016년03월18일 미국(US)
15/359,577 2016년11월22일 미국(US)

명 세 서

청구범위

청구항 1

비디오 데이터를 디코딩하는 방법으로서,

비디오 디코더에 의해, 상기 비디오 데이터의 인코딩된 표현을 포함하는 비트스트림을 수신하는 단계;

상기 비디오 디코더에 의해, 상기 비디오 데이터의 코딩 유닛 (CU) 이 트리 구조에 기초하여 상기 CU 의 변환 유닛들 (TU들) 로 파티셔닝된다고 결정하는 단계로서,

상기 CU 가 트리 구조에 기초하여 상기 CU 의 TU들로 파티셔닝된다고 결정하는 단계는 상기 비디오 디코더에 의해, 상기 트리 구조에서 노드가 상기 트리 구조에서 정확히 2 개의 차일드 노드들을 갖는다고 결정하는 단계를 포함하고,

상기 트리 구조의 루트 노드는 상기 CU 의 코딩 블록에 대응하고, 상기 트리 구조의 각각의 개별의 비-루트 노드는 상기 개별의 비-루트 노드의 페어런트 노드에 대응하는 블록의 파티션인 개별의 블록에 대응하고, 상기 트리 구조의 리프 노드들은 상기 CU 의 상기 TU들에 대응하는, 상기 CU 가 트리 구조에 기초하여 상기 CU 의 TU들로 파티셔닝된다고 결정하는 단계; 및

상기 CU 의 상기 TU들 중 적어도 하나에 대해,

상기 비디오 디코더에 의해, 상기 TU 에 대한 잔차 블록을 생성하기 위해 상기 TU 에 대한 계수 블록에 변환을 적용하는 단계; 및

상기 비디오 디코더에 의해, 예측 블록의 샘플들을 상기 CU 의 상기 TU 에 대한 상기 잔차 블록의 대응하는 샘플들에 추가함으로써 코딩 블록의 샘플들을 복원하는 단계를 포함하는, 비디오 데이터를 디코딩하는 방법.

청구항 2

제 1 항에 있어서,

상기 노드는 제 1 노드이고,

상기 CU 가 트리 구조에 기초하여 상기 CU 의 TU들로 파티셔닝된다고 결정하는 단계는,

상기 비디오 디코더에 의해, 상기 트리 구조에서 제 2 노드가 상기 트리 구조에서 정확히 4 개의 차일드 노드들을 갖는다고 결정하는 단계를 포함하는, 비디오 데이터를 디코딩하는 방법.

청구항 3

제 1 항에 있어서,

상기 노드가 정확히 2 개의 차일드 노드들을 갖는다고 결정하는 단계는, 상기 비디오 디코더에 의해, 상기 CU 의 예측 유닛들 (PU들) 의 총 수에 기초하여, 상기 트리 구조가 바이너리 트리인지 또는 쿼터 트리인지를 결정하는 단계를 포함하고,

상기 CU 가 2 개의 PU들을 갖는 것에 기초하여, 상기 노드는 상기 트리 구조에서 정확히 2 개의 차일드 노드들을 갖는, 비디오 데이터를 디코딩하는 방법.

청구항 4

제 1 항에 있어서,

상기 트리 구조의 특정 심도에서의 노드들에 대응하는 블록들은 상기 CU 의 예측 유닛들 (PU들) 의 예측 블록들과 동일한 사이즈를 갖는, 비디오 데이터를 디코딩하는 방법.

청구항 5

제 4 항에 있어서,

상기 CU 의 상기 PU들 중 적어도 하나의 예측 블록은 직사각형인, 비디오 데이터를 디코딩하는 방법.

청구항 6

제 1 항에 있어서,

상기 CU 가 트리 구조에 기초하여 상기 CU 의 TU들로 파티셔닝된다고 결정하는 단계는 상기 비디오 디코더에 의해, 상기 트리 구조의 적어도 하나의 노드의 차일드 노드들이 상이한 사이즈들의 블록들에 대응한다고 결정하는 단계를 포함하는, 비디오 데이터를 디코딩하는 방법.

청구항 7

제 1 항에 있어서,

상기 노드는 제 1 노드이고,

상기 CU 가 트리 구조에 기초하여 상기 CU 의 TU들로 파티셔닝된다고 결정하는 단계는,

상기 비디오 디코더에 의해, 상기 제 1 노드의 차일드 노드들에 대응하는 블록들 간의 경계가 수직적이라고 결정하는 단계; 및

상기 비디오 디코더에 의해, 상기 트리 구조에서 제 2 노드가 정확히 2 개의 차일드 노드들을 갖고 그리고 상기 제 2 노드의 차일드 노드들에 대응하는 블록들 간의 경계는 수평적이라고 결정하는 단계를 더 포함하는, 비디오 데이터를 디코딩하는 방법.

청구항 8

비디오 데이터를 인코딩하는 방법으로서,

비디오 인코더에 의해, 상기 비디오 데이터를 수신하는 단계;

트리 구조에 기초하여 상기 비디오 데이터의 코딩 유닛 (CU) 을 상기 CU 의 변환 유닛들 (TU들) 로 파티셔닝하는 단계로서,

상기 트리 구조에 기초하여 CU 를 상기 CU 의 TU들로 파티셔닝하는 단계는 상기 비디오 인코더에 의해, 상기 트리 구조에서 노드가 상기 트리 구조에서 정확히 2 개의 차일드 노드들을 갖는다고 결정하는 단계를 포함하고,

상기 트리 구조의 루트 노드는 상기 CU 의 코딩 블록에 대응하고, 상기 트리 구조의 각각의 개별의 비-루트 노드는 상기 개별의 비-루트 노드의 페어런트 노드에 대응하는 블록의 파티션인 개별의 블록에 대응하고, 상기 트리 구조의 리프 노드들은 상기 CU 의 상기 TU들에 대응하는, 상기 트리 구조에 기초하여 CU 를 상기 CU 의 TU들로 파티셔닝하는 단계; 및

상기 CU 의 상기 TU들 중 적어도 하나에 대해,

상기 비디오 인코더에 의해, 상기 TU 에 대한 변환 계수들의 블록을 생성하기 위해 상기 TU 에 대한 잔차 블록에 변환을 적용하는 단계; 및

상기 비디오 인코더에 의해, 상기 TU 에 대한 변환 계수들을 나타내는 신택스 엘리먼트들을 엔트로피 인코딩하는 단계를 포함하는, 비디오 데이터를 인코딩하는 방법.

청구항 9

제 8 항에 있어서,

상기 노드는 제 1 노드이고,

상기 트리 구조에 기초하여 CU 를 상기 CU 의 TU들로 파티셔닝하는 단계는, 상기 비디오 인코더에 의해, 상기 트리 구조에서 제 2 노드가 상기 트리 구조에서 정확히 4 개의 차일드 노드들을 갖는다고 결정하는 단계를 더 포함하는, 비디오 데이터를 인코딩하는 방법.

청구항 10

제 8 항에 있어서,

상기 CU 를 TU들로 파티셔닝하는 단계는, 상기 비디오 인코더에 의해, 상기 CU 의 예측 유닛들 (PU들) 의 총 수에 기초하여, 상기 트리 구조가 바이너리 트리인지 또는 쿼터 트리인지를 결정하는 단계를 포함하고,

상기 CU 가 2 개의 PU들을 갖는 것에 기초하여, 상기 노드는 상기 트리 구조에서 정확히 2 개의 차일드 노드들을 갖는, 비디오 데이터를 인코딩하는 방법.

청구항 11

제 8 항에 있어서,

상기 트리 구조의 특정 심도에서의 노드들에 대응하는 블록들은 상기 CU 의 예측 유닛들 (PU들) 의 예측 블록들과 동일한 사이즈를 갖는, 비디오 데이터를 인코딩하는 방법.

청구항 12

제 10 항에 있어서,

상기 CU 의 상기 PU들 중 적어도 하나의 예측 블록은 직사각형인, 비디오 데이터를 인코딩하는 방법.

청구항 13

제 8 항에 있어서,

상기 트리 구조에 기초하여 CU 를 상기 CU 의 TU들로 파티셔닝하는 단계는 상기 비디오 인코더에 의해, 상기 트리 구조의 적어도 하나의 노드의 차일드 노드들이 상이한 사이즈들의 블록들에 대응하도록 상기 CU 를 TU들로 파티셔닝하는 단계를 포함하는, 비디오 데이터를 인코딩하는 방법.

청구항 14

제 8 항에 있어서,

상기 노드는 제 1 노드이고,

상기 트리 구조에 기초하여 CU 를 상기 CU 의 TU들로 파티셔닝하는 단계는

비디오 디코더에 의해, 상기 노드의 차일드 노드들에 대응하는 블록들 간의 경계가 수직적이라고 결정하는 단계; 및

상기 비디오 디코더에 의해, 상기 트리 구조에서 제 2 노드가 정확히 2 개의 차일드 노드들을 갖고 그리고 상기 제 2 노드의 차일드 노드들에 대응하는 블록들 간의 경계는 수평적이라고 결정하는 단계를 더 포함하는, 비디오 데이터를 인코딩하는 방법.

청구항 15

비디오 데이터를 디코딩하기 위한 장치로서,

상기 비디오 데이터를 저장하도록 구성된 하나 이상의 저장 매체들; 및

비디오 디코더를 포함하고,

상기 비디오 디코더는,

상기 비디오 데이터의 인코딩된 표현을 포함하는 비트스트림을 수신하고;

상기 비디오 데이터의 코딩 유닛 (CU) 이 트리 구조에 기초하여 상기 CU 의 변환 유닛들 (TU들) 로 파티셔닝된다고 결정하는 것으로서,

상기 비디오 디코더는, 상기 CU 가 트리 구조에 기초하여 상기 CU 의 TU들로 파티셔닝된다고 결정하는 것의 부분으로서, 상기 비디오 디코더가 상기 트리 구조에서 노드가 상기 트리 구조에서 정확히 2 개의 차일드 노드들을 갖는다고 결정하도록 구성되고,

상기 트리 구조의 루트 노드는 상기 CU 의 코딩 블록에 대응하고, 상기 트리 구조의 각각의 개별의 비-루트 노

드는 상기 개별의 비-루트 노드의 페어런트 노드에 대응하는 블록의 파티션인 개별의 블록에 대응하고, 상기 트리 구조의 리프 노드들은 상기 CU 의 상기 TU들에 대응하는, 상기 CU 가 트리 구조에 기초하여 상기 CU 의 TU들로 파티셔닝된다고 결정하고; 그리고

상기 CU 의 상기 TU들 중 적어도 하나에 대해,

상기 TU 에 대한 잔차 블록을 생성하기 위해 상기 TU 에 대한 계수 블록에 변환을 적용하고; 그리고

예측 블록의 샘플들을 상기 CU 의 상기 TU 에 대한 상기 잔차 블록의 대응하는 샘플들에 추가함으로써 코딩 블록의 샘플들을 복원하도록

구성되는, 비디오 데이터를 디코딩하기 위한 장치.

청구항 16

제 15 항에 있어서,

상기 노드는 제 1 노드이고,

상기 비디오 디코더는, 상기 CU 가 트리 구조에 기초하여 상기 CU 의 TU들로 파티셔닝된다고 결정하는 것의 부분으로서, 상기 비디오 디코더가 상기 트리 구조에서 제 2 노드가 상기 트리 구조에서 정확히 4 개의 차일드 노드들을 갖는다고 결정하도록 구성되는, 비디오 데이터를 디코딩하기 위한 장치.

청구항 17

제 15 항에 있어서,

상기 비디오 디코더는, 상기 노드가 정확히 2 개의 차일드 노드들을 갖는다고 결정하는 것의 부분으로서, 상기 비디오 디코더가 상기 CU 의 예측 유닛들 (PU들) 의 총 수에 기초하여, 상기 트리 구조가 바이너리 트리인지 또는 쿼터 트리인지를 결정하도록 구성되고,

상기 CU 가 2 개의 PU들을 갖는 것에 기초하여, 상기 노드는 상기 트리 구조에서 정확히 2 개의 차일드 노드들을 갖는, 비디오 데이터를 디코딩하기 위한 장치.

청구항 18

제 15 항에 있어서,

상기 트리 구조의 특정 심도에서의 노드들에 대응하는 블록들은 상기 CU 의 예측 유닛들 (PU들) 의 예측 블록들과 동일한 사이즈를 갖는, 비디오 데이터를 디코딩하기 위한 장치.

청구항 19

제 18 항에 있어서,

상기 CU 의 상기 PU들 중 적어도 하나의 예측 블록은 직사각형인, 비디오 데이터를 디코딩하기 위한 장치.

청구항 20

제 15 항에 있어서,

상기 비디오 디코더는, 상기 CU 가 트리 구조에 기초하여 상기 CU 의 TU들로 파티셔닝된다고 결정하는 것의 부분으로서, 상기 비디오 디코더가 상기 트리 구조의 적어도 하나의 노드의 차일드 노드들이 상이한 사이즈들의 블록들에 대응한다고 결정하도록 구성되는, 비디오 데이터를 디코딩하기 위한 장치.

청구항 21

제 15 항에 있어서,

상기 노드는 제 1 노드이고,

상기 비디오 디코더는, 상기 CU 가 트리 구조에 기초하여 상기 CU 의 TU들로 파티셔닝된다고 결정하는 것의 부분으로서, 상기 비디오 디코더가,

상기 제 1 노드의 차일드 노드들에 대응하는 블록들 간의 경계가 수직적이라고 결정하고; 그리고

상기 트리 구조에서 제 2 노드가 정확히 2 개의 차일드 노드들을 갖고 그리고 상기 제 2 노드의 차일드 노드들에 대응하는 블록들 간의 경계는 수평적이라고 결정하도록

구성되는, 비디오 데이터를 디코딩하기 위한 장치.

청구항 22

비디오 데이터를 인코딩하기 위한 장치로서,

상기 비디오 데이터를 저장하도록 구성된 하나 이상의 저장 매체들; 및

비디오 인코더를 포함하고,

상기 비디오 인코더는,

상기 비디오 데이터를 수신하고;

트리 구조에 기초하여 상기 비디오 데이터의 코딩 유닛 (CU) 을 상기 CU 의 변환 유닛들 (TU들) 로 파티셔닝하는 것으로서,

상기 비디오 인코더는, 상기 트리 구조에 기초하여 CU 를 상기 CU 의 TU들로 파티셔닝하는 것의 부분으로서, 상기 비디오 인코더가 상기 트리 구조에서 노드가 상기 트리 구조에서 정확히 2 개의 차일드 노드들을 갖는다고 결정하도록 구성되고,

상기 트리 구조의 루트 노드는 상기 CU 의 코딩 블록에 대응하고, 상기 트리 구조의 각각의 개별의 비-루트 노드는 상기 개별의 비-루트 노드의 페어런트 노드에 대응하는 블록의 파티션인 개별의 블록에 대응하고, 상기 트리 구조의 리프 노드들은 상기 CU 의 상기 TU들에 대응하는, 상기 트리 구조에 기초하여 CU 를 상기 CU 의 TU들로 파티셔닝하고; 그리고

상기 CU 의 상기 TU들 중 적어도 하나에 대해,

상기 TU 에 대한 변환 계수들의 블록을 생성하기 위해 상기 TU 에 대한 잔차 블록에 변환을 적용하고; 그리고

상기 TU 에 대한 변환 계수들을 나타내는 신택스 엘리먼트들을 엔트로피 인코딩하도록

구성되는, 비디오 데이터를 인코딩하기 위한 장치.

청구항 23

제 22 항에 있어서,

상기 노드는 제 1 노드이고,

상기 비디오 인코더는, 상기 트리 구조에 기초하여 CU 를 상기 CU 의 TU들로 파티셔닝하는 것의 부분으로서, 상기 비디오 인코더가 상기 트리 구조에서 제 2 노드가 상기 트리 구조에서 정확히 4 개의 차일드 노드들을 갖는다고 결정하도록 구성되는, 비디오 데이터를 인코딩하기 위한 장치.

청구항 24

제 22 항에 있어서,

상기 비디오 인코더는, 상기 CU 를 TU들로 파티셔닝하는 것의 부분으로서, 상기 비디오 인코더가 상기 CU 의 예측 유닛들 (PU들) 의 총 수에 기초하여, 상기 트리 구조가 바이너리 트리인지 또는 쿼터 트리인지를 결정하도록 구성되고,

상기 CU 가 2 개의 PU들을 갖는 것에 기초하여, 상기 노드는 상기 트리 구조에서 정확히 2 개의 차일드 노드들을 갖는, 비디오 데이터를 인코딩하기 위한 장치.

청구항 25

제 22 항에 있어서,

상기 트리 구조의 특정 심도에서의 노드들에 대응하는 블록들은 상기 CU 의 예측 유닛들 (PU들) 의 예측 블록들

과 동일한 사이즈를 갖는, 비디오 데이터를 인코딩하기 위한 장치.

청구항 26

제 25 항에 있어서,

상기 CU 의 상기 PU들 중 적어도 하나의 예측 블록은 직사각형인, 비디오 데이터를 인코딩하기 위한 장치.

청구항 27

제 22 항에 있어서,

상기 비디오 인코더는, 상기 트리 구조에 기초하여 CU 를 상기 CU 의 TU들로 파티셔닝하는 것의 부분으로서, 상기 비디오 인코더가 상기 트리 구조의 적어도 하나의 노드의 차일드 노드들이 상이한 사이즈들의 블록들에 대응하도록 상기 CU 를 TU들로 파티셔닝하도록 구성되는, 비디오 데이터를 인코딩하기 위한 장치.

청구항 28

제 22 항에 있어서,

상기 노드는 제 1 노드이고,

상기 비디오 인코더는, 상기 트리 구조에 기초하여 CU 를 상기 CU 의 TU들로 파티셔닝하는 것의 부분으로서, 상기 비디오 인코더가,

상기 노드의 차일드 노드들에 대응하는 블록들 간의 경계가 수직적이라고 결정하고; 그리고

상기 트리 구조에서 제 2 노드가 정확히 2 개의 차일드 노드들을 갖고 그리고 상기 제 2 노드의 차일드 노드들에 대응하는 블록들 간의 경계는 수평적이라고 결정하도록

구성되는, 비디오 데이터를 인코딩하기 위한 장치.

청구항 29

비디오 데이터를 디코딩하기 위한 장치로서,

상기 비디오 데이터의 인코딩된 표현을 포함하는 비트스트림을 수신하기 위한 수단;

상기 비디오 데이터의 코딩 유닛 (CU) 이 트리 구조에 기초하여 상기 CU 의 변환 유닛들 (TU들) 로 파티셔닝된다고 결정하기 위한 수단으로서,

상기 CU 가 트리 구조에 기초하여 상기 CU 의 TU들로 파티셔닝된다고 결정하기 위한 수단은 상기 트리 구조에서 노드가 상기 트리 구조에서 정확히 2 개의 차일드 노드들을 갖는다고 결정하기 위한 수단을 포함하고,

상기 트리 구조의 루트 노드는 상기 CU 의 코딩 블록에 대응하고, 상기 트리 구조의 각각의 개별의 비-루트 노드는 상기 개별의 비-루트 노드의 페어런트 노드에 대응하는 블록의 파티션인 개별의 블록에 대응하고, 상기 트리 구조의 리프 노드들은 상기 CU 의 상기 TU들에 대응하는, 상기 CU 가 트리 구조에 기초하여 상기 CU 의 TU들로 파티셔닝된다고 결정하기 위한 수단; 및

상기 CU 의 상기 TU들 중 적어도 하나에 대해,

상기 TU 에 대한 잔차 블록을 생성하기 위해 상기 TU 에 대한 계수 블록에 변환을 적용하기 위한 수단; 및

예측 블록의 샘플들을 상기 CU 의 상기 TU 에 대한 상기 잔차 블록의 대응하는 샘플들에 추가함으로써 코딩 블록의 샘플들을 복원하기 위한 수단을 포함하는, 비디오 데이터를 디코딩하기 위한 장치.

청구항 30

비디오 데이터를 인코딩하기 위한 장치로서,

상기 비디오 데이터를 수신하기 위한 수단;

트리 구조에 기초하여 상기 비디오 데이터의 코딩 유닛 (CU) 을 상기 CU 의 변환 유닛들 (TU들) 로 파티셔닝하기 위한 수단으로서,

상기 트리 구조에 기초하여 CU 를 상기 CU 의 TU들로 파티셔닝하기 위한 수단은 상기 트리 구조에서 노드가 상기 트리 구조에서 정확히 2 개의 차일드 노드들을 갖는다고 결정하기 위한 수단을 포함하고,

상기 트리 구조의 루트 노드는 상기 CU 의 코딩 블록에 대응하고, 상기 트리 구조의 각각의 개별의 비-루트 노드는 상기 개별의 비-루트 노드의 페어런트 노드에 대응하는 블록의 파티션인 개별의 블록에 대응하고, 상기 트리 구조의 리프 노드들은 상기 CU 의 상기 TU들에 대응하는, 상기 트리 구조에 기초하여 CU 를 상기 CU 의 TU들로 파티셔닝하기 위한 수단; 및

상기 CU 의 상기 TU들 중 적어도 하나에 대해,

상기 TU 에 대한 변환 계수들의 블록을 생성하기 위해 상기 TU 에 대한 잔차 블록에 변환을 적용하기 위한 수단; 및

상기 TU 에 대한 변환 계수들을 나타내는 신택스 엘리먼트들을 엔트로피 인코딩하기 위한 수단을 포함하는, 비디오 데이터를 인코딩하기 위한 장치.

청구항 31

명령들이 저장되어 있는 컴퓨터 판독가능 데이터 저장 매체로서,

상기 명령들은, 실행되는 경우, 비디오 데이터를 디코딩하기 위한 장치를 구성하여:

상기 비디오 데이터의 인코딩된 표현을 포함하는 비트스트림을 수신하고;

상기 비디오 데이터의 코딩 유닛 (CU) 이 트리 구조에 기초하여 상기 CU 의 변환 유닛들 (TU들) 로 파티셔닝된다고 결정하는 것으로서,

명령들은, 실행되는 경우, 상기 CU 가 트리 구조에 기초하여 상기 CU 의 TU들로 파티셔닝된다고 결정하는 것의 부분으로서, 상기 장치가 상기 트리 구조에서 노드가 상기 트리 구조에서 정확히 2 개의 차일드 노드들을 갖는다고 결정하도록 상기 장치를 구성하고,

상기 트리 구조의 루트 노드는 상기 CU 의 코딩 블록에 대응하고, 상기 트리 구조의 각각의 개별의 비-루트 노드는 상기 개별의 비-루트 노드의 페어런트 노드에 대응하는 블록의 파티션인 개별의 블록에 대응하고, 상기 트리 구조의 리프 노드들은 상기 CU 의 상기 TU들에 대응하는, 상기 CU 가 트리 구조에 기초하여 상기 CU 의 TU들로 파티셔닝된다고 결정하고; 그리고

상기 CU 의 상기 TU들 중 적어도 하나에 대해,

상기 TU 에 대한 잔차 블록을 생성하기 위해 상기 TU 에 대한 계수 블록에 변환을 적용하고; 그리고

예측 블록의 샘플들을 상기 CU 의 상기 TU 에 대한 상기 잔차 블록의 대응하는 샘플들에 추가함으로써 코딩 블록의 샘플들을 복원하는, 컴퓨터 판독가능 데이터 저장 매체.

청구항 32

명령들이 저장되어 있는 컴퓨터 판독가능 저장 매체로서,

상기 명령들은, 실행되는 경우, 비디오 데이터를 인코딩하기 위한 장치를 구성하여:

상기 비디오 데이터를 수신하고;

트리 구조에 기초하여 상기 비디오 데이터의 코딩 유닛 (CU) 을 상기 CU 의 변환 유닛들 (TU들) 로 파티셔닝하는 것으로서,

상기 명령들은, 실행되는 경우, 상기 트리 구조에 기초하여 CU 를 상기 CU 의 TU들로 파티셔닝하는 것의 부분으로서, 상기 장치가 상기 트리 구조에서 노드가 상기 트리 구조에서 정확히 2 개의 차일드 노드들을 갖는다고 결정하도록 상기 장치를 구성하고,

상기 트리 구조의 루트 노드는 상기 CU 의 코딩 블록에 대응하고, 상기 트리 구조의 각각의 개별의 비-루트 노드는 상기 개별의 비-루트 노드의 페어런트 노드에 대응하는 블록의 파티션인 개별의 블록에 대응하고, 상기 트리 구조의 리프 노드들은 상기 CU 의 상기 TU들에 대응하는, 상기 트리 구조에 기초하여 CU 를 상기 CU 의 TU들로 파티셔닝하고; 그리고

상기 CU 의 상기 TU들 중 적어도 하나에 대해,

상기 TU 에 대한 변환 계수들의 블록을 생성하기 위해 상기 TU 에 대한 잔차 블록에 변환을 적용하고; 그리고

상기 TU 에 대한 변환 계수들을 나타내는 신택스 엘리먼트들을 엔트로피 인코딩하는, 컴퓨터 판독가능 저장 매체.

발명의 설명

기 술 분 야

[0001] 본 출원은 2015년 11월 25일자로 출원된 미국 가특허출원 제 62/260,103 호, 및 2016년 3월 18일자로 출원된 미국 가특허출원 제 62/310,271 호의 이익을 주장하며, 이들 각각의 전체 내용은 참조로서 본원에 포함된다.

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

배 경 기 술

[0003] 디지털 비디오 능력들은 디지털 텔레비전들, 디지털 직접 방송 시스템들, 무선 방송 시스템들, PDA (personal digital assistant)들, 랩톱 또는 데스크톱 컴퓨터들, 태블릿 컴퓨터들, 전자책 리더들, 디지털 카메라들, 디지털 녹음 디바이스들, 디지털 미디어 재생기들, 비디오 게이밍 디바이스들, 비디오 게임 콘솔들, 셀룰러 또는 위성 무선 전화들, 소위 "스마트 폰들", 비디오 원격화상회의 디바이스들, 비디오 스트리밍 디바이스들 등을 포함하는 광범위한 디바이스들에 통합될 수 있다. 디지털 비디오 디바이스들은 MPEG-2, MPEG-4, ITU-T H.263, ITU-T H.264/MPEG-4, 파트 10, 진보된 비디오 코딩 (AVC), 고효율 비디오 코딩 (HEVC) 에 의해 정의된 표준들, 및 이러한 표준들의 확장들에서 설명된 바와 같은 비디오 코딩 기법들을 구현한다. 비디오 디바이스들은 이러한 비디오 코딩 기법들을 구현함으로써 디지털 비디오 정보를 보다 효율적으로 송신, 수신, 인코딩, 디코딩, 및/또는 저장할 수도 있다.

[0004] 비디오 코딩 기법들은 비디오 시퀀스들에 내재하는 리던던시를 감소시키기나 제거하기 위한 공간적 (인트라-픽처) 예측 및/또는 시간적 (인터-픽처) 예측을 포함한다. 블록-기반 비디오 코딩에 있어서, 비디오 슬라이스 (예를 들어, 비디오 프레임 또는 비디오 프레임의 일부) 는 비디오 블록들로 파티셔닝될 수도 있으며, 이 비디오 블록들은 또한 트리블록들, 코딩 유닛 (CU)들 및/또는 코딩 노드들로 지칭될 수도 있다. 픽처들은 프레임들로서 지칭될 수도 있고, 레퍼런스 픽처들은 레퍼런스 프레임들로 지칭될 수도 있다.

[0005] 공간 예측 또는 시간 예측은 코딩될 블록에 대한 예측 블록을 초래한다. 잔차 데이터는 코딩될 원래의 블록과 예측 블록 간의 픽셀 차이들을 나타낸다. 추가의 압축을 위해, 잔차 데이터는 픽셀 도메인에서 변환 도메인으로 변환되어, 잔차 변환 계수들을 초래하고 이 변환 계수들은, 그 후 양자화될 수도 있다. 더욱 더 많은 압축을 달성하기 위해 엔트로피 코딩이 적용될 수도 있다.

발명의 내용

해결하려는 과제

과제의 해결 수단

[0006] 본 개시물은 인트라 및 인터 예측 파티션들, 비-스퀘어 (non-square) 변환들, 비-스퀘어 블록들에 대한 인트라 및 인터 코딩 모드들, 및 연관된 엔트로피 코딩에 관련된다. 본 개시물의 기법들은, HEVC 의 확장들 또는 차세대 비디오 코딩 표준들과 같은 진보된 비디오 코덱들의 맥락에서 사용될 수도 있다. 일 예에 있어서, 노드가 정확히 2 개의 차일드 노드들을 가질 수도 있는 플렉서블 변환 트리가 사용된다.

[0007] 일 예에 있어서, 본 개시물은 비디오 데이터를 디코딩하는 방법을 설명하고, 그 방법은 비디오 디코더에 의해, 비디오 데이터의 인코딩된 표현을 포함하는 비트스트림을 수신하는 단계; 비디오 디코더에 의해, 비디오 데이터의 코딩 유닛 (CU) 이 트리 구조에 기초하여 CU 의 변환 유닛들 (TU들) 로 파티셔닝된다고 결정하는 단계로서, CU 가 트리 구조에 기초하여 CU 의 TU들로 파티셔닝된다고 결정하는 단계는 비디오 디코더에 의해, 트리 구조에서 노드가 트리 구조에서 정확히 2 개의 차일드 노드들을 갖는다고 결정하는 단계를 포함하고, 트리 구조의 루트 노드는 CU 의 코딩 블록에 대응하고, 트리 구조의 각각의 개별의 비-루트 노드는 개별의 비-루트 노드의 페

어런트 노드에 대응하는 블록의 파티션인 개별의 블록에 대응하고, 트리 구조의 리프 노드들은 CU 의 TU들에 대응하는, 상기 CU 가 트리 구조에 기초하여 CU 의 TU들로 파티셔닝된다고 결정하는 단계; CU 의 TU들 중 적어도 하나에 대해, 비디오 디코더에 의해, TU 에 대한 잔차 블록을 생성하기 위해 TU 에 대한 계수 블록에 변환을 적용하는 단계; 및 비디오 디코더에 의해, 예측 블록의 샘플들을 CU 의 TU 에 대한 잔차 블록의 대응하는 샘플들에 추가함으로써 코딩 블록의 샘플들을 복원하는 단계를 포함한다.

[0008] 다른 예에 있어서, 본 개시물은 비디오 데이터를 인코딩하는 방법을 설명하고, 그 방법은 비디오 인코더에 의해, 비디오 데이터를 수신하는 단계; 트리 구조에 기초하여 비디오 데이터의 코딩 유닛 (CU) 을 CU 의 변환 유닛들 (TU들) 로 파티셔닝하는 단계로서, 트리 구조에 기초하여 CU 를 CU 의 TU들로 파티셔닝하는 단계는 비디오 인코더에 의해, 트리 구조에서 노드가 트리 구조에서 정확히 2 개의 차일드 노드들을 갖는다고 결정하는 단계를 포함하고, 트리 구조의 루트 노드는 CU 의 코딩 블록에 대응하고, 트리 구조의 각각의 개별의 비-루트 노드는 개별의 비-루트 노드의 페어런트 노드에 대응하는 블록의 파티션인 개별의 블록에 대응하고, 트리 구조의 리프 노드들은 CU들의 TU들에 대응하는, 상기 트리 구조에 기초하여 CU 를 CU 의 TU들로 파티셔닝하는 단계; CU 의 TU들 중 적어도 하나에 대해, 비디오 인코더에 의해, TU 에 대한 변환 계수들의 블록을 생성하기 위해 TU 에 대한 잔차 블록에 변환을 적용하는 단계; 및 비디오 인코더에 의해, TU 에 대한 변환 계수들을 나타내는 신택스 엘리먼트들을 엔트로피 인코딩하는 단계를 포함한다.

[0009] 다른 예에 있어서, 본 개시물은 비디오 데이터를 디코딩하기 위한 장치를 설명하고, 그 장치는 비디오 데이터를 저장하도록 구성된 하나 이상의 저장 매체들; 및 비디오 디코더를 포함하고, 비디오 디코더는, 비디오 데이터의 인코딩된 표현을 포함하는 비트스트림을 수신하고; 비디오 데이터의 코딩 유닛 (CU) 이 트리 구조에 기초하여 CU 의 변환 유닛들 (TU들) 로 파티셔닝된다고 결정하는 것으로서, 비디오 디코더는, CU 가 트리 구조에 기초하여 CU 의 TU들로 파티셔닝된다고 결정하는 것의 부분으로서, 비디오 디코더가 트리 구조에서 노드가 트리 구조에서 정확히 2 개의 차일드 노드들을 갖는다고 결정하도록 구성되고, 트리 구조의 루트 노드는 CU 의 코딩 블록에 대응하고, 트리 구조의 각각의 개별의 비-루트 노드는 개별의 비-루트 노드의 페어런트 노드에 대응하는 블록의 파티션인 개별의 블록에 대응하고, 트리 구조의 리프 노드들은 CU 의 TU들에 대응하는, 상기 CU 가 트리 구조에 기초하여 CU 의 TU들로 파티셔닝된다고 결정하고; CU 의 TU들 중 적어도 하나에 대해, TU 에 대한 잔차 블록을 생성하기 위해 TU 에 대한 계수 블록에 변환을 적용하고; 그리고 예측 블록의 샘플들을 CU 의 TU 에 대한 잔차 블록의 대응하는 샘플들에 추가함으로써 코딩 블록의 샘플들을 복원하도록 구성된다.

[0010] 다른 예에 있어서, 본 개시물은 비디오 데이터를 인코딩하기 위한 장치를 설명하고, 그 장치는 비디오 데이터를 저장하도록 구성된 하나 이상의 저장 매체들; 및 비디오 인코더를 포함하고, 비디오 인코더는, 비디오 데이터를 수신하고; 트리 구조에 기초하여 비디오 데이터의 코딩 유닛 (CU) 을 CU 의 변환 유닛들 (TU들) 로 파티셔닝하는 것으로서, 비디오 인코더는, 트리 구조에 기초하여 CU 를 CU 의 TU들로 파티셔닝하는 것의 부분으로서, 비디오 인코더가 트리 구조에서 노드가 트리 구조에서 정확히 2 개의 차일드 노드들을 갖는다고 결정하도록 구성되고, 트리 구조의 루트 노드는 CU 의 코딩 블록에 대응하고, 트리 구조의 각각의 개별의 비-루트 노드는 개별의 비-루트 노드의 페어런트 노드에 대응하는 블록의 파티션인 개별의 블록에 대응하고, 트리 구조의 리프 노드들은 CU들의 TU들에 대응하는, 상기 트리 구조에 기초하여 CU 를 CU 의 TU들로 파티셔닝하고; CU 의 TU들 중 적어도 하나에 대해, TU 에 대한 변환 계수들의 블록을 생성하기 위해 TU 에 대한 잔차 블록에 변환을 적용하고; 그리고 TU 에 대한 변환 계수들을 나타내는 신택스 엘리먼트들을 엔트로피 인코딩하도록 구성된다.

[0011] 다른 예에 있어서, 본 개시물은 비디오 데이터를 디코딩하기 위한 장치를 설명하고, 그 장치는 비디오 데이터의 인코딩된 표현을 포함하는 비트스트림을 수신하기 위한 수단; 비디오 데이터의 코딩 유닛 (CU) 이 트리 구조에 기초하여 CU 의 변환 유닛들 (TU들) 로 파티셔닝된다고 결정하기 위한 수단으로서, CU 가 트리 구조에 기초하여 CU 의 TU들로 파티셔닝된다고 결정하기 위한 수단은 트리 구조에서 노드가 트리 구조에서 정확히 2 개의 차일드 노드들을 갖는다고 결정하기 위한 수단을 포함하고, 트리 구조의 루트 노드는 CU 의 코딩 블록에 대응하고, 트리 구조의 각각의 개별의 비-루트 노드는 개별의 비-루트 노드의 페어런트 노드에 대응하는 블록의 파티션인 개별의 블록에 대응하고, 트리 구조의 리프 노드들은 CU 의 TU들에 대응하는, 상기 CU 가 트리 구조에 기초하여 CU 의 TU들로 파티셔닝된다고 결정하기 위한 수단; CU 의 TU들 중 적어도 하나에 대해, TU 에 대한 잔차 블록을 생성하기 위해 TU 에 대한 계수 블록에 변환을 적용하기 위한 수단; 및 예측 블록의 샘플들을 CU 의 TU 에 대한 잔차 블록의 대응하는 샘플들에 추가함으로써 코딩 블록의 샘플들을 복원하기 위한 수단을 포함한다.

[0012] 다른 예에 있어서, 본 개시물은 비디오 데이터를 인코딩하기 위한 장치를 설명하고, 그 장치는 비디오 데이터를 수신하기 위한 수단; 트리 구조에 기초하여 비디오 데이터의 코딩 유닛 (CU) 을 CU 의 변환 유닛들 (TU들) 로 파티셔닝하기 위한 수단으로서, 트리 구조에 기초하여 CU 를 CU 의 TU들로 파티셔닝하기 위한 수단은 트리 구조

에서 노드가 트리 구조에서 정확히 2 개의 차일드 노드들을 갖는다고 결정하기 위한 수단을 포함하고, 트리 구조의 루트 노드는 CU 의 코딩 블록에 대응하고, 트리 구조의 각각의 개별의 비-루트 노드는 개별의 비-루트 노드의 페어런트 노드에 대응하는 블록의 파티션인 개별의 블록에 대응하고, 트리 구조의 리프 노드들은 CU들의 TU들에 대응하는, 상기 트리 구조에 기초하여 CU 를 CU 의 TU들로 파티셔닝하기 위한 수단; CU 의 TU들 중 적어도 하나에 대해, TU 에 대한 변환 계수들의 블록을 생성하기 위해 TU 에 대한 잔차 블록에 변환을 적용하기 위한 수단; 및 TU 에 대한 변환 계수들을 나타내는 신택스 엘리먼트들을 엔트로피 인코딩하기 위한 수단을 포함한다.

[0013] 다른 예에 있어서, 본 개시물은 명령들이 저장되어 있는 컴퓨터 판독가능 데이터 저장 매체를 설명하고, 명령들은, 실행되는 경우, 비디오 데이터를 디코딩하기 위한 장치를 구성하여: 비디오 데이터의 인코딩된 표현을 포함하는 비트스트림을 수신하고; 비디오 데이터의 코딩 유닛 (CU) 이 트리 구조에 기초하여 CU 의 변환 유닛들 (TU들) 로 파티셔닝된다고 결정하는 것으로서, 명령들은, 실행되는 경우, CU 가 트리 구조에 기초하여 CU 의 TU들로 파티셔닝된다고 결정하는 것의 부분으로서, 그 장치가 트리 구조에서 노드가 트리 구조에서 정확히 2 개의 차일드 노드들을 갖는다고 결정하도록 그 장치를 구성하고, 트리 구조의 루트 노드는 CU 의 코딩 블록에 대응하고, 트리 구조의 각각의 개별의 비-루트 노드는 개별의 비-루트 노드의 페어런트 노드에 대응하는 블록의 파티션인 개별의 블록에 대응하고, 트리 구조의 리프 노드들은 CU 의 TU들에 대응하는, 상기 CU 가 트리 구조에 기초하여 CU 의 TU들로 파티셔닝된다고 결정하고; CU 의 TU들 중 적어도 하나에 대해, TU 에 대한 잔차 블록을 생성하기 위해 TU 에 대한 계수 블록에 변환을 적용하고; 그리고 예측 블록의 샘플들을 CU 의 TU 에 대한 잔차 블록의 대응하는 샘플들에 추가함으로써 코딩 블록의 샘플들을 복원한다.

[0014] 다른 예에 있어서, 본 개시물은 명령들이 저장되어 있는 컴퓨터 판독가능 저장 매체를 설명하고, 명령들은, 실행되는 경우, 비디오 데이터를 인코딩하기 위한 장치를 구성하여: 비디오 데이터를 수신하고; 트리 구조에 기초하여 비디오 데이터의 코딩 유닛 (CU) 을 CU 의 변환 유닛들 (TU들) 로 파티셔닝하는 것으로서, 명령들은, 실행되는 경우, 트리 구조에 기초하여 CU 를 CU 의 TU들로 파티셔닝하는 것의 부분으로서, 그 장치가 트리 구조에서 노드가 트리 구조에서 정확히 2 개의 차일드 노드들을 갖는다고 결정하도록 그 장치를 구성하고, 트리 구조의 루트 노드는 CU 의 코딩 블록에 대응하고, 트리 구조의 각각의 개별의 비-루트 노드는 개별의 비-루트 노드의 페어런트 노드에 대응하는 블록의 파티션인 개별의 블록에 대응하고, 트리 구조의 리프 노드들은 CU들의 TU들에 대응하는, 상기 트리 구조에 기초하여 CU 를 CU 의 TU들로 파티셔닝하고; CU 의 TU들 중 적어도 하나에 대해, TU 에 대한 변환 계수들의 블록을 생성하기 위해 TU 에 대한 잔차 블록에 변환을 적용하고; 그리고 TU 에 대한 변환 계수들을 나타내는 신택스 엘리먼트들을 엔트로피 인코딩한다.

[0015] 본 개시물의 하나 이상의 양태들의 세부사항들은 첨부되는 도면들 및 하기의 설명들에서 기술된다. 본 개시물에 설명된 기법들의 다른 특성들, 목적들 및 이점들은 상세한 설명, 도면들 및 청구항들로부터 명백해질 것이다.

도면의 간단한 설명

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

도 2a 는 고효율 비디오 코딩 (HEVC) 에서의 잔차 쿼드트리에 기초하여 예시의 변환 스킴을 예시하는 개념도이다.

도 2b 는 도 2a 의 코딩 유닛에 대한 잔차 쿼드트리를 예시하는 개념도이다.

도 3 은 HEVC 에서의 계수 그룹들에 기초한 예시의 계수 스캔을 예시하는 개념도이다.

도 4 는 16×16 블록에 대한 인트라 예측의 일 예를 예시하는 개념도이다.

도 5 는 HEVC 에서 정의된 35 개의 인트라 예측 모드들의 예를 예시하는 개념도이다.

도 6 은 HEVC 에서 정의된 평면 모드를 예시하는 개념도이다.

도 7 은 HEVC 에서 정의된 예시의 각 모드 (angular mode) 의 개념도이다.

도 8 은 HEVC 에서 인트라 예측을 위해 코딩 유닛을 스플리팅 (splitting) 하는 파티션 모드들의 개념도이다.

도 9 는 단거리 인트라 예측 (SDIP) 유닛 파티션들의 개념도이다.

도 10 은 8×8 매트릭스로 스캐닝 및 재구성된 16×4 계수 매트릭스의 개념도이다.

도 11 은 64 개의 인트라 예측 모드들의 개념도이다.

도 12a 는 인트라 모드 (34) 에 대한 경계 필터들의 개념도이다.

도 12b 는 인트라 모드 (30-33) 에 대한 경계 필터들의 개념도이다.

도 13 은 선형 모델 (LM) 파라미터들 α 및 β 의 도출에 사용된 샘플들의 예시의 로케이션들을 예시하는 개념도이다.

도 14 는 현재 예측 유닛 (PU) 의 복원된 루마 블록의 샘플들을 다운-샘플링하기 위한 루마 포지션들 및 크로마 포지션들의 예를 예시하는 개념도이다.

도 15 는 예측 블록을 생성하기 위한 루마 블록의 샘플들을 다운-샘플링하기 위한 루마 포지션들 및 크로마 포지션들의 예를 예시하는 개념도이다.

도 16 은 N×N 변환을 갖는 nR×2N 예측 모드를 예시하는 개념도이다.

도 17 은 2N×N, 2N×nD, 및 2N×nU 예측 모드들에 대한 비-스퀘어 쿼드트리 (NSQT) 를 예시하는 개념도이다.

도 18 은 N×2N, nR×2N, 및 nL×2N 예측 모드들에 대한 NSQT 를 예시하는 개념도이다.

도 19 는 일루미네이션 보상 (Illumination Compensation; IC) 모델에서 파라미터들을 추정하기 위해 사용된 이웃하는 픽셀들을 예시한다.

도 20 은 IC 모델에서 파라미터들을 추정하는데 사용된 예시의 이웃하는 픽셀들을 예시하는 개념도이고, 여기서 현재 코딩 유닛의 레퍼런스 블록은 현재 예측 유닛의 디스패리티 또는 모션 벡터를 사용함으로써 발견된다.

도 21 은 2N×N 과 동일한 파티션 사이즈에 대한 예시의 변환 구조를 예시하는 개념도이다.

도 22 는 본 개시물의 기법에 따라, N×N/4(U) 와 동일한 파티션 사이즈에 대한 변환 구조를 예시하는 개념도이다.

도 23 은 본 개시물의 기법에 따라, N×N/4(U) 와 동일한 파티션 사이즈에 대한 변환 구조를 예시하는 개념도이다.

도 24 는 본 개시물에서 기법들을 구현할 수도 있는 예시의 비디오 인코더를 예시하는 블록도이다.

도 25 는 본 개시물의 기법들을 구현하도록 구성되는 예시의 비디오 디코더를 예시하는 블록도이다.

도 26 은 본 개시물의 기법에 따라 LM-기반의 인코딩을 지원하는 예시의 비디오 인코더를 예시하는 블록도이다.

도 27 은 본 개시물의 기법에 따라 LM-기반의 디코딩을 지원하는 예시의 비디오 디코더를 예시하는 블록도이다.

도 28 은 본 개시물의 LM-기반의 코딩 기법에 따른 비디오 인코더의 예시의 동작을 예시하는 플로우차트이다.

도 29 는 본 개시물의 LM-기반의 코딩 기법에 따른 비디오 디코더의 예시의 동작을 예시하는 플로우차트이다.

도 30 은 본 개시물의 양자화 기법에 따른, 비디오 인코더의 예시의 동작을 예시하는 플로우차트이다.

도 31 은 본 개시물의 양자화 기법에 따른, 비디오 디코더의 예시의 동작을 예시하는 플로우차트이다.

도 32 는 IC 를 사용하는 본 개시물의 기법에 따른, 비디오 인코더의 예시의 동작을 예시하는 플로우차트이다.

도 33 은 IC 를 사용하는 본 개시물의 기법에 따른, 비디오 디코더의 예시의 동작을 예시하는 플로우차트이다.

도 34 는 플렉서블 잔차 트리를 사용하는 본 개시물의 기법에 따른, 비디오 인코더의 예시의 동작을 예시하는 플로우차트이다.

도 35 는 플렉서블 잔차 트리를 사용하는 본 개시물의 기법에 따른, 비디오 디코더의 예시의 동작을 예시하는 플로우차트이다.

발명을 실시하기 위한 구체적인 내용

[0017] 일반적으로, 본 개시물은 인트라 및 인터 예측 파티션들, 비-스퀘어 변환들, 비-스퀘어 블록들에 대한 인트라 및 인터 코딩 모드들, 및 연관된 엔트로피 코딩에 관련된다. 본 개시물의 기법들은 고효율 비디오 코딩 (HEVC) 의 확장들 또는 차세대의 비디오 코딩 표준들과 같은, 진보된 비디오 코덱들의 맥락에서 사용될 수도 있다.

[0018] HEVC 에서, 비디오 코더 (즉, 비디오 인코더 또는 비디오 디코더) 는 픽처의 코딩 유닛 (CU) 을 하나 이상의 예측 유닛 (PU)들로 파티셔닝한다. 비디오 코더는 인트라 예측 또는 인터 예측을 사용하여 CU 의 각각의 PU 에 대한 예측 블록들을 생성한다. CU 의 잔차 데이터는 CU 의 원래의 코딩 블록과 CU 의 PU들에 대한 예측 블록들 간의 차이들을 나타낸다. CU 가 인트라 예측되는 경우들 (즉, CU 의 PU들에 대한 예측 블록들이 인트라 예측을 사용하여 생성됨) 에서, CU 의 잔차 데이터는 하나 이상의 스퀘어-형상 변환 유닛 (TU)들로 파티셔닝될 수도 있다. 그러나, CU 가 인터 예측되는 경우들 (즉, CU 의 PU들에 대한 예측 블록들이 인터 예측을 사용하여 생성됨) 에서, CU 의 잔차 데이터는 하나 이상의 스퀘어 또는 비-스퀘어 TU들로 파티셔닝될 수도 있다. 본 개시물에서, 유닛들 (예를 들어, CU들, PU들, TU들) 의 형상들에 대한 참조들은 대응하는 블록들의 형상들을 지칭할 수도 있다. 따라서, 비-스퀘어 PU 는 비-스퀘어 예측 블록을 지칭하는 것으로서 해석될 수도 있고, 비-스퀘어 TU 는 비-스퀘어 변환 블록을 지칭하는 것으로서 해석될 수도 있으며, 그 반대일 수도 있다. 또한, 예측 블록은 PU 가 HEVC 에서 정의될 때 PU 의 개념에 묶일 필요는 없고, 차라리 예측 (예를 들어, 인터 예측, 인트라 예측) 이 수행되는 샘플들의 블록의 의미를 갖는다는 것이 주목된다. 유사하게, 변환 블록은 TU 가 HEVC 에서 정의될 때 TU 의 개념에 묶일 필요는 없고, 차라리 변환이 적용되는 샘플들의 블록의 의미를 갖는다.

[0019] 이하에서 설명되는 바와 같이, 비-스퀘어 TU들의 도입은, 특정 코딩 툴들과 사용되는 경우 소정의 문제들을 도입할 수도 있다.

[0020] 예를 들어, 선형 모델링 (LM) 예측 모드는 HEVC 의 개발 동안 연구되었던 크로스-컴포넌트 상관을 감소시키기 위한 기법이다. 비디오 코더가 LM 예측 모드를 사용하는 경우, 비디오 코더는 CU 의 PU 의 복원된 루마 샘플들에 기초하여 PU 의 크로마 샘플들을 예측한다. PU 의 크로마 샘플들은 PU 의 크로마 예측 블록의 크로마 샘플들이다. 크로마 샘플들의 예시의 유형들은 Cb 샘플들 및 Cr 샘플들을 포함한다. 비디오 코더는 PU 의 루마 예측 블록의 샘플들을 PU 의 대응하는 루마 잔차 샘플들과 합산함으로써 PU 의 복원된 루마 샘플들을 생성할 수도 있다.

[0021] 특히, 비디오 코더가 LM 예측 모드를 사용하는 경우, 비디오 코더는 포지션 (i,j) 에서 PU 의 예측된 크로마 샘플을 $\alpha \cdot rec_L(i,j) + \beta$ 로서 결정할 수도 있고, 여기서 $rec_L(i,j)$ 는 포지션 (i,j) 에서 PU 의 복원된 루마 샘플이고, α 및 β 는 파라미터들이다. 4:2:0 컬러 포맷에서와 같은 일부 경우들에서, 하나의 M×K 크로마 블록은 2M×2K 루마 블록에 대응하고, 이 경우에서 $rec_L(i,j)$ 는 2M×2K 루마 블록의 (M×K 인) 다운-샘플링된 버전의 (i,j) 에 위치된 값을 나타낸다. 비디오 코더는, 복원된 루마 레퍼런스 샘플들 및 복원된 크로마 레퍼런스 샘플들의 값들에 기초하여 α 및 β 의 값을 결정한다. 복원된 루마 레퍼런스 샘플들 및 복원된 크로마 레퍼런스 샘플들은 PU 의 상단 및 좌측 사이드들을 따른 샘플들이다. β 를 결정하기 위한 수식들은 레퍼런스 샘플들의 총 수 (M 및 K 의 합과 동일한, I 로 표기됨) 에 의한 나누기 연산을 수반한다. 통상적인 경우들에서, M 및 K 는 동일하고, HEVC 에서 2^l 로 표현될 수 있고, l 은 양의 정수 값이다. 예측 블록이 스퀘어인 한, I 는 2^m 과 동일하고, 여기서 m 은 상이한 예측 블록 사이즈들에 대해 가변적이다. 따라서, I 로 나누기 위해 나누기 연산을 수행하는 대신에, 비디오 코더는 β 의 값을 계산하는 경우 우측 시프트 연산 (right shift operation) 을 수행할 수도 있다. 우측 시프트 연산들은 나누기 연산들보다 구현하기가 상당히 더 빠르고 덜 복잡하다. 본 개시물에서, CU들, TU들, 및 PU들과 같은 블록들의 다양한 유형들의 사이즈들에 대한 레퍼런스들은 CU들, TU들, 및 PU들의 코딩 블록들, 변환 블록들, 및 예측 블록들의 사이즈들을 각각 지칭한다. 또한, 본 개시물에서, CU들, TU들, 및 PU들과 같은 비디오 코딩 유닛들의 다양한 유형들의 사이드들에 대한 레퍼런스들은 블록들의 다양한 유형들에 대응하는 블록들 (예를 들어, 코딩 블록들, 변환 블록들, 예측/예측적 블록들) 의 사이드들을 참조한다.

[0022] 그러나, 루마 블록 (예를 들어, PU 의 루마 예측 블록) 이 스퀘어가 아니면 (예를 들어, M 인 12 와 동일하고 K 가 16 과 동일함), I 는 2^m 과 항상 동일하지는 않다. 따라서, 루마 블록이 스퀘어가 아니면, β 의 값을 계산하는 경우 나누기 연산 대신에 우측 시프트 연산을 사용하는 것이 가능하지 않을 수도 있다. 따라서, 비디오 코더는 β 의 값을 계산하기 위해 비용이 드는 나누기 연산을 구현할 필요가 있을 수도 있다.

[0023] 본 개시물은 비-스퀘어 블록들에 대해 LM 예측 모드를 사용하는 경우 β 의 값을 계산할 때 나누기 연산을 구현할 필요성을 제거할 수도 있는 기법을 설명한다. 일부 경우들에서, M 이 K 와 동일하지만 M 이 2 의 거듭제곱이 아닌 스퀘어 PU 에 대해서도, 여기에 설명된 기법이 또한, 적용 가능할 수도 있다. 본 기법의 예에 따르면, 비디오 코더는 루마 레퍼런스 샘플들의 세트 및 크로마 레퍼런스 샘플들의 세트를 복원할 수도 있다. 루마 레퍼런스 샘플들의 세트는 비디오 데이터의 현재 픽처의 비-스퀘어 루마 블록의 상단 사이드를 이웃하는 루마 샘플들 및 비-스퀘어 루마 블록의 좌측 사이드를 이웃하는 루마 샘플들을 포함할 수도 있다. 비-스퀘어 루마 블록은 PU 의 루마 예측 블록일 수도 있다. 따라서, PU 는 비-스퀘어 PU 로 간주될 수도 있다. 크로마 레퍼런스 샘플들의 세트는 비-스퀘어 크로마 블록의 상단 사이드를 이웃하는 크로마 샘플들 및 비-스퀘어 크로마 블록의 좌측 사이드를 이웃하는 크로마 샘플들을 포함할 수도 있다. 비-스퀘어 크로마 블록은 PU 의 크로마 예측 블록일 수도 있다. 부가적으로, 비디오 코더는 비-스퀘어 예측 블록의 루마 샘플들을 복원할 수도 있다. 또한, 비디오 코더는, 비-스퀘어 루마 블록의 더 긴 사이드를 이웃하는 루마 레퍼런스 샘플들의 세트에서 루마 레퍼런스 샘플들의 총 수가 비-스퀘어 루마 블록의 더 짧은 사이드를 이웃하는 루마 레퍼런스 샘플들의 세트의 루마 레퍼런스 샘플들의 총 수와 동일하도록 루마 레퍼런스 샘플들의 세트를 서브-샘플링할 수도 있다. 비디오 코더는 다음과 동일한 제 1 파라미터를 결정할 수도 있다:

[0024] $$\alpha = \frac{I\sum x_i \cdot y_i - \sum x_i \cdot \sum y_i}{I\sum x_i \cdot x_i - \sum x_i \cdot \sum x_i}$$

[0025] 여기서, I 는 루마 레퍼런스 샘플들의 세트에서 레퍼런스 샘플들의 총 수이고, x_i 는 루마 레퍼런스 샘플들의 세트에서 루마 레퍼런스 샘플이며, y_i 는 크로마 레퍼런스 샘플들의 세트에서 크로마 레퍼런스 샘플이다. 부가적으로, 비디오 코더는 다음과 동일한 제 2 파라미터를 결정할 수도 있다:

[0026] $$\beta = \left(\sum y_i - \alpha \cdot \sum x_i\right) / I$$

[0027] 예측 크로마 블록의 각각의 개별의 크로마 샘플에 대해, 비디오 코더는, 개별의 크로마 샘플이 α 곱하기 개별의 크로마 샘플에 대응하는 개별의 복원된 루마 샘플, 플러스 β 와 동일하도록 개별의 크로마 샘플의 값을 결정할 수도 있다. 예측 크로마 블록은 비-스퀘어 PU 에 대한 예측 크로마 블록일 수도 있다. 비디오 코더는, 예측 크로마 블록에 적어도 부분적으로 기초하여 코딩 블록을 복원할 수도 있다.

[0028] 상기 예에서, 비-스퀘어 루마 블록의 더 긴 사이드를 이웃하는 루마 레퍼런스 샘플들의 총 수가 비-스퀘어 루마 블록의 더 짧은 사이드를 이웃하는 루마 레퍼런스 샘플들의 총 수와 동일하도록 루마 레퍼런스 샘플들의 세트를 서브-샘플링함으로써, 비디오 코더는, 루마 레퍼런스 샘플들에서 레퍼런스 샘플들의 총 수가 2 의 거듭제곱이라는 것을 보장할 수도 있다. 따라서, 비디오 코더는, β 의 값을 계산할 때 나누기 연산 대신에 우측 시프트 연산을 사용할 수도 있다. 따라서, 상기의 예를 구현하는 비디오 코더는, β 의 값을 계산할 때 나누기 연산을 사용하도록 강요된 비디오 디코더보다 덜 복잡하고/하거나 더 빠를 수도 있다. 그러나, 비디오 코더는 시프트 연산 대신에 나누기 연산을 사용하는 상기의 예에서 설명된 액션들을 수행할 수도 있지만, 이러한 비디오 코더가 나누기 연산 대신에 시프트 연산을 사용하는 이점들을 갖지 않을 수도 있다는 것이 주목된다. 일부 예들에서, 비-스퀘어 예측 블록의 더 짧은 또는 더 긴 사이드를 이웃하는 레퍼런스 샘플들은 이용 가능하지 않을 수도 있고, 이 경우에서 다른 사이드에 위치된 이용 가능한 레퍼런스 샘플들에 서브-샘플링 프로세스를 수행할 필요가 없을 수도 있다.

[0029] HEVC 에서, 비디오 인코더는 잔차 데이터의 블록들 (즉, 변환 블록들) 에 변환을 적용하여 잔차 데이터의 블록들을 변환 계수들의 블록들로 컨버팅한다. 고 레벨에서, 비디오 인코더는, 변환 블록의 컬럼들에 N-포인트 1-차원 DCT 변환을 적용함으로써 중간 값들의 블록을 먼저 생성함으로써, 변환 계수들의 블록 (즉, 변환 계수 블록) 을 생성할 수도 있다. N 은 변환 블록의 높이 및 폭과 동일하다. 비디오 인코더는 그 후, 동일한 N-포인트 1-차원 DCT 변환을 중간 값들의 블록의 로우들에 적용함으로써 변환 계수들의 블록을 생성할 수도 있다. 비디오 디코더는 변환 블록을 복구하기 위한 유사한 방식으로 변환을 반전시킨다.

[0030] 상기의 논의로부터 알 수 있는 바와 같이, HEVC 에서 변환을 적용하는 프로세스는 스퀘어인 변환 블록들에 의존한다. 그러나, 비-스퀘어 변환 블록들을 갖는 것이 바람직할 수도 있다. 예를 들어, 압축 성능은, 변환 블록들의 경계들이 인터 또는 인트라 예측 블록들의 경계들을 가로지르는 경우 감소될 수도 있다. 비-스퀘어 예측 블록들의 사용은 스퀘어 영역들 안에 있지 않은 오브젝트들을 캡처하는데 가치가 있을 수도 있다. 따라서, 비-스퀘어 예측 블록들 및/또는 비-스퀘어 변환들은 코딩 성능 개선의 관점에서 유용할 수도 있다.

변환 매트릭스 계수는, 변환 매트릭스 계수가 N-포인트 1-차원 DCT 변환인 경우 $\sqrt{N}$ 과 동일한 분모로 정의된다. 본 개시물 전에, 분모 $\sqrt{N}$ 은 정규화 팩터로서 간주되었고 양자화 프로세스에서 우측 시프트에 의해 구현되었다. 2-차원 DCT 변환, 예를 들어 K×L 변환을 고려하면, 정규화 팩터는 ($\sqrt{K}*\sqrt{L}$) 일 것이다. N 이 등식 log2(N*N) = ((log2(K) + log2(L)) >> 1) << 1) 을 충족시키는 것에 의해 정의되면, 이용된 정규화 팩터 (sqrt(N)*sqrt(N)) 및 실제 정규화 팩터 ($\sqrt{K}*\sqrt{L}$) 의 비는 $1/\sqrt{2}$ 일 것이다. 비-스퀘어 변환 블록에 스퀘어 변환 (예를 들어, 컬럼들 및 로우들 양자 모두에 적용된 N-포인트 변환) 을 직접적으로 적용하는 것은, 감소된 압축 성능을 초래하는, 증가된 정규화 팩터로 인해 결과의 변환 계수 블록에서 총 에너지 (예를 들어, 양자화 후의 모든 변환된 계수들의 제곱들의 합) 를 변화시킬 수도 있다.

[0031] 본 개시물의 다른 곳에서 상세히 설명되는 바와 같이, 사이즈 K×L 의 변환 블록에 대해, (log2(K) + log2(L)) 이 홀수인 경우 비디오 인코더는 변환 계수들에 $\sqrt{2}$ 를 곱하고, (log2(K) + log2(L)) 가 홀수인 경우 비디오 디코더는 변환 계수들에 $\sqrt{2}$ 를 곱하여 이 문제를 다룰 수도 있다.

[0032] 3D-HEVC 는 3-차원 (3D) 비디오 데이터에 대한 HEVC 의 확장이다. 3D-HEVC 는 상이한 뷰포인트들로부터 동일한 장면의 다수의 뷰들을 제공한다. 3D-HEVC 에 대한 표준화 노력들은 HEVC 에 기초한 멀티-뷰 비디오 코덱의 표준화를 포함한다. 3D-HEVC 에서, 상이한 뷰들로부터 복원된 뷰 컴포넌트들 (즉, 복원된 픽처들) 에 기초한 인터-뷰 예측이 인에이블된다. 또한, 3D-HEVC 는 인터-뷰 모션 예측 및 인터-뷰 잔차 예측을 구현한다.

[0033] 비디오의 동일한 시간 인스턴스를 나타내는 각각의 뷰의 픽처들은 유사한 비디오 콘텐츠를 포함한다. 그러나, 뷰들의 비디오 콘텐츠는 서로에 대해 공간적으로 변위될 수도 있다. 특히, 뷰들의 비디오 콘텐츠는 동일한 장면 상의 상이한 관점들을 나타낼 수도 있다. 예를 들어, 제 1 뷰에서 픽처 내의 비디오 블록은 제 2 뷰에서 픽처 내의 비디오 블록과 유사한 비디오 콘텐츠를 포함할 수도 있다. 이 예에서, 제 1 뷰에서 픽처 내의 비디오 블록의 로케이션 및 제 2 뷰에서 픽처 내의 비디오 블록의 로케이션은 상이할 수도 있다. 예를 들어, 상이한 뷰들에서 비디오 블록들의 로케이션들 간에 일부 변위가 존재할 수도 있다.

[0034] 비디오 블록에 대한 디스패리티 벡터는 이 변위의 측정을 제공한다. 예를 들어, 제 1 뷰에서 픽처의 비디오 블록은 제 2 뷰에서 픽처 내의 대응하는 비디오 블록의 변위를 나타내는 디스패리티 벡터와 연관될 수도 있다.

[0035] 광원들로부터의 상이한 거리들 또는 상이한 카메라 설정들 때문에, 동일한 시간 인스턴스에 대응하지만 상이한 뷰들에 있는 픽처들은 거의 동일한 이미지를 포함할 수도 있지만, 픽처들 중 하나에서의 오브젝터들은 다른 픽처에서의 대응하는 오브젝트들 보다 더 밝을 수도 있다. 일루미네이션 보상 (IC) 은, 인터-뷰 예측을 수행하는 경우 뷰들 간의 일루미네이션에서의 이러한 차이들을 보상하기 위해 3D-HEVC 에서 구현된 기법이다. 3D-HEVC 에서, 비디오 코더는 현재 픽처의 현재 CU 의 현재 PU 에 대한 디스패리티 벡터를 결정한다. 또한, 비디오 코더는 현재 CU 에 대한 2 개의 IC 파라미터들을 계산할 수도 있다. 본 개시물은 IC 파라미터들을 *a* 및 *b* 로서 표기한다. 부가적으로, 현재 PU 의 루마 예측 블록의 각각의 개별의 샘플에 대해, 비디오 코더는 다음과 같이 계산한다:

[0036] $$p(i,j) = a * r\,(\,i + dv_x, j + dv_y + b\,)$$

[0037] 상기의 등식에서, $p(i,j)$ 는 현재 PU 의 루마 예측 블록의 개별의 샘플이고, (i,j) 는 현재 픽처의 상단-좌측 코너에 대한 개별의 샘플의 로케이션을 나타내는 좌표들이고, dv_x 는 현재 PU 에 대한 디스패리티 벡터의 수평

성분이고, dv_y 는 현재 PU 에 대한 디스패리티 벡터의 수직 성분이며, a 및 b 는 IC 파라미터들이다.

[0038] 본 개시물의 다른 곳에서 더 상세히 설명된 바와 같이, 3D-HEVC 에서 IC 파라미터 (b) 를 정의하는 수식은 현재의 CU의 상단 및 좌측 사이드들에 이웃하는 레퍼런스 샘플들의 수에 의한 나누기 연산을 수반한다. 3D-HEVC 에서, 현재 CU 의 상단 및 좌측 사이드들에 이웃하는 레퍼런스 샘플들의 수는 항상 2 의 거듭제곱이다. 결과적으로, IC 파라미터 (b) 를 정의하는 수식에서 나누기 연산은 우측-시프트 연산을 사용하여 구현될 수도 있다. 본 개시물의 다른 곳에서 설명된 바와 같이, 우측-시프트 연산은 나누기 연산 보다 구현하기가 상당히 덜 복잡할 수도 있고, 나누기 연산을 구현하는 것보다 상당히 더 빠를 수도 있다.

[0039] 2015년 6월 19-26일, 바르샤바, 폴란드, ITU - 전기통신 표준화 부문, 연구단 16 질문 6, 비디오 코딩 전문가 그룹 (VCEG), 52 차 회의, 문헌 VCEG-AZ06, H. Liu 의, "Local Illumination Compensation" (이하에서, "VCEG-AZ06") 에서 설명된 바와 같은, 2-차원 비디오 코딩에 대해, 로컬 일루미네이션 보상 (LIC) 이 각각의 인터-모드 코딩된 코딩 유닛 (CU) 에 대해 적응적으로 인에이블 또는 디스에이블되고, LIC 는 스케일링 팩터 (a) 및 오프셋 (b) 를 사용하는, 일루미네이션 변화들에 대한 선형 모델에 기초한다. LIC 가 CU 에 대해 적용하는 경우, CU 에 속하는 각각의 PU/서브-PU 에 대해, LIC 파라미터들은, 레퍼런스 픽처에서 (현재 PU/서브-PU 의 모션 정보에 의해 식별된) 대응하는 픽셀들 및 CU 의 서브샘플링된 (2:1 서브샘플링) 이웃하는 샘플들을 사용하는 방식으로 도출된다. N×N 과 동일한 사이즈를 갖는 CU 에 대해, 파라미터 계산에서 사용된 경계 픽셀들의 총 수는 2N 대신에 N 이다. 일 예가 도 20 에 예시된다. LIC 파라미터들은 각각의 예측 방향에 대해 별개로 도출 및 적용된다. 전술된 이웃하는 샘플들에 기초하여 파라미터들 (a 및 b) 을 도출하기 위해 최소 자승 오차법이 이용된다.

[0040] IC 는 단지, CU 가 단일의 PU 만을 갖는 경우 사용되었다. 그러나, CU 가 2 또는 3 개의 PU들로 파티셔닝되고/되거나 CU 가 비대칭적으로 파티셔닝되는 경우들을 포함하여, CU 가 다수의 PU들을 갖는 경우들에서 IC 를 사용하는 것이 바람직할 수도 있다. 이러한 경우들에서, 현재 CU 의 상단 및 좌측 사이드들에 이웃하는 레퍼런스 샘플들의 수는 더 이상 2 의 거듭제곱이 아닐 수도 있다. 따라서, 우측-시프트 연산을 사용하여 IC 파라미터 (b) 를 계산하는 것이 가능하지 않을 수도 있다. 차라리, 비디오 코더는 IC 파라미터 (b) 를 계산하기 위해 더 느리고 더 복잡한 나누기 연산을 사용할 필요가 있을 수도 있다.

[0041] 이 문제를 다루기 위해, 비디오 코더는 레퍼런스 샘플들의 제 1 세트를 서브-샘플링하여 총 2^m 개의 레퍼런스 샘플들을 포함하는 레퍼런스 샘플들의 제 1 서브-샘플링된 세트를 생성할 수도 있고, 여기서 m 은 정수이다. 본 개시물에서, 용어들 서브-샘플링은 샘플들의 세트로부터 하나 이상의 샘플들의 선택을 나타내고 다운 샘플링은, 여러 샘플들이 함께 사용되어 필터링된 샘플을 도출할 수도 있는 필터링 프로세스를 나타낸다. 레퍼런스 샘플들의 세트는 비-스퀘어 예측 블록의 좌측 사이드 및 상단 사이드를 따른 PU 의 비-스퀘어 예측 블록 밖의 샘플들을 포함할 수도 있다. 따라서, 레퍼런스 샘플들은 또한, 이웃 샘플들 또는 이웃하는 샘플들로서 본원에 지칭될 수도 있다. 부가적으로, 비디오 코더는 레퍼런스 샘플들의 제 2 세트를 서브-샘플링하여 총 2^m 개의 레퍼런스 샘플들을 포함하는 레퍼런스 샘플들의 제 2 서브-샘플링된 세트를 생성할 수도 있고, 여기서 m 은 정수이다. 레퍼런스 샘플들의 제 2 세트는 레퍼런스 블록의 좌측 사이드 및 상단 사이드를 따른 레퍼런스 블록 (예를 들어, 인터-뷰 레퍼런스 블록 또는 시간적 레퍼런스 블록) 밖의 샘플들을 포함할 수도 있다. 비디오 코더는 그 후, 레퍼런스 샘플들의 제 1 서브-샘플링된 세트 및 레퍼런스 샘플들의 제 2 서브-샘플링된 세트에 기초하여 적어도 IC 파라미터 (b) 를 결정할 수도 있다. 레퍼런스 샘플들의 제 1 서브-샘플링된 세트 및 레퍼런스 샘플들의 제 2 서브-샘플링된 세트 각각이 2^m 개의 샘플들을 포함하기 때문에, 비디오 코더는 나누기 연산 대신에 우측 시프트 연산을 사용하여 IC 파라미터 (b) 를 계산할 수도 있다. 이 방식으로, 본 개시물의 기법들은 비디오 코더의 복잡성을 감소시키고/시키거나 비디오 코딩을 가속화할 수도 있다.

[0042] 위에서 언급된 바와 같이, 비-스퀘어 TU들의 도입은 소정의 문제들을 도입할 수도 있다. 예를 들어, CU 의 PU들이 스퀘어가 아닌 경우에도 CU 를 TU들로 파티셔닝하는 이전의 기법들은 쿼드-트리 스플리팅 패턴을 따랐다. 본 개시물에서, 쿼드-트리는 또한, 쿼터-트리로서 지칭될 수도 있다. 쿼드-트리 스플리팅 패턴을 항상 사용하는 것은, 특히, 쿼드-트리 스플리팅 패턴이 CU 의 PU들의 사이드들과 TU들의 사이드들을 정렬시키지 않으면 차선의 비디오 데이터 압축 성능을 초래할 수도 있다.

[0043] 따라서, 본 개시물의 기법에 따르면, CU 의 변환 트리는 쿼드-트리 스플리팅 패턴에 제한되지 않는다. 차라리, 변환 트리에서의 노드는 2 개의 차일드 노드들을 가질 수도 있다. 따라서, 일 예에서, 비디오 디코더는 CU 가 트리 구조에 기초하여 TU들로 파티셔닝된다고 결정할 수도 있다. 이 예에서, 비디오 디코더는, 트리

구조에서의 노드가 트리 구조에서 2 개의 차일드 노드들을 갖는다는 것을 결정할 수도 있다. 이 예에서, 트리 구조의 루트 노드는 CU 의 코딩 블록에 대응하고, 트리 구조의 각각의 개별의 비-루트 노드는 개별의 비-루트 노드의 페어런트 노드에 대응하는 블록의 파티션인 개별의 블록에 대응하고, 트리 구조의 리프 노드들은 CU 의 TU들에 대응한다. 일부 예들에서, 트리 구조에서의 노드들은 2 또는 4 개의 차일드 노드들을 가질 수도 있다. 2 또는 4 개의 차일드 노드들을 갖는 노드의 유연성은 비디오 코딩 압축 성능을 증가시킬 수도 있다.

[0044] 도 1 은 본 개시물의 기법들을 이용할 수도 있는 예시의 비디오 인코딩 및 디코딩 시스템 (10) 을 예시하는 블록도이다. 도 1 에 도시된 바와 같이, 시스템 (10) 은 목적지 디바이스 (14) 에 의해 나중에 디코딩될 인코딩된 비디오 데이터를 제공하는 소스 디바이스 (12) 를 포함한다. 특히, 소스 디바이스 (12) 는 컴퓨터 판독가능 매체 (16) 를 통해 목적지 디바이스 (14) 에 비디오 데이터를 제공한다. 소스 디바이스 (12) 및 목적지 디바이스 (14) 는, 데스크톱 컴퓨터들, 노트북 (즉, 랩톱) 컴퓨터들, 태블릿 컴퓨터들, 셋톱 박스들, 소위 "스마트" 폰들과 같은 전화 핸드셋들, 태블릿 컴퓨터들, 텔레비전들, 카메라들, 디스플레이 디바이스들, 디지털 미디어 플레이어들, 비디오 게이밍 콘솔들, 비디오 스트리밍 디바이스 등을 포함하는 다양한 범위의 디바이스들 중 임의의 것을 포함할 수도 있다. 일부 경우들에서, 소스 디바이스 (12) 및 목적지 디바이스 (14) 는 무선 통신을 위해 구비될 수도 있다.

[0045] 도 1 의 예에서, 소스 디바이스 (12) 는 비디오 소스 (18), 비디오 데이터를 저장하도록 구성된 저장 매체 (19), 비디오 인코더 (20), 및 출력 인터페이스 (22) 를 포함한다. 목적지 디바이스 (14) 는 입력 인터페이스 (28), 비디오 데이터를 저장하도록 구성된 저장 매체 (29), 비디오 디코더 (30), 및 디스플레이 디바이스 (32) 를 포함한다. 다른 예들에서, 소스 디바이스 및 목적지 디바이스는 다른 컴포넌트들 또는 배열들을 포함할 수도 있다. 예를 들어, 소스 디바이스 (12) 는 외부 비디오 소스, 예컨대 외부 카메라로부터 비디오 데이터를 수신할 수도 있다. 마찬가지로, 목적지 디바이스 (14) 는 집적 디스플레이 디바이스를 포함하기보다는, 외부 디스플레이 디바이스와 인터페이스할 수도 있다.

[0046] 도 1 의 예시된 시스템 (10) 은 단지 일 예이다. 비디오 데이터를 프로세싱하기 위한 기법들은 임의의 디지털 비디오 인코딩 및/또는 디코딩 디바이스에 의해 수행될 수도 있다. 일반적으로, 본 개시물의 기법들은 비디오 인코딩 디바이스에 의해 수행되지만, 그 기법들은 또한, 통상적으로 "CODEC" 으로서 지칭된 비디오 인코더/디코더에 의해 수행될 수도 있다. 소스 디바이스 (12) 및 목적지 디바이스 (14) 는 단지, 이러한 코딩 디바이스들의 예들이고, 여기서 소스 디바이스 (12) 는 목적지 디바이스 (14) 로의 송신을 위해 코딩된 비디오 데이터를 생성한다. 일부 예들에서, 디바이스들 (12, 14) 은, 디바이스들 (12, 14) 각각이 비디오 인코딩 및 디코딩 컴포넌트들을 포함하도록 실질적으로 대칭적인 방식으로 동작할 수도 있다. 따라서, 시스템 (10) 은, 예를 들어 비디오 스트리밍, 비디오 플레이백, 비디오 브로드캐스팅, 또는 비디오 텔레포니를 위해 비디오 디바이스들 (12, 14) 간의 일방향 또는 양방향 비디오 송신을 지원할 수도 있다.

[0047] 소스 디바이스 (12) 의 비디오 소스 (18) 는 비디오 캡처 디바이스, 예컨대 비디오 카메라, 이전에 캡처된 비디오 데이터를 포함하는 비디오 아카이브, 및/또는 비디오 콘텐츠 제공자로부터 비디오를 수신하기 위한 비디오 피드 인터페이스를 포함할 수도 있다. 추가의 대안으로서, 비디오 소스 (18) 는 소스 비디오로서 컴퓨터 그래픽 기반 데이터, 또는 라이브 비디오, 아카이브된 비디오, 및 컴퓨터 생성된 비디오의 조합을 생성할 수도 있다. 일부 경우들에서, 소스 디바이스 (12) 및 목적지 디바이스 (14) 는 소위 카메라 폰들 또는 비디오 폰들을 형성할 수도 있다. 소스 디바이스 (12) 는 비디오 데이터를 저장하도록 구성된 하나 이상의 데이터 저장 매체들 (예를 들어, 저장 매체들 (19)) 을 포함할 수도 있다. 본 개시물에서 설명된 기법들은 일반적으로 비디오 코딩에 적용 가능할 수도 있으며, 무선 및/또는 유선 애플리케이션들에 적용될 수도 있다. 각각의 경우에서, 캡처된, 미리-캡처된, 또는 컴퓨터 생성된 비디오는 비디오 인코더 (20) 에 의해 인코딩될 수도 있다. 출력 인터페이스 (22) 는 그 후, 인코딩된 비디오 정보를 컴퓨터 판독가능 매체 (16) 상으로 출력할 수도 있다.

[0048] 출력 인터페이스 (22) 는 컴포넌트들 또는 디바이스들의 다양한 유형들을 포함할 수도 있다. 예를 들어, 출력 인터페이스 (22) 는 무선 송신기, 모뎀, 유선 네트워킹 컴포넌트 (예를 들어, 이더넷 카드), 또는 다른 물리적 컴포넌트를 포함할 수도 있다. 출력 인터페이스 (22) 가 무선 수신기를 포함하는 예들에서, 출력 인터페이스 (22) 는 4G, 4G-LTE, LTE 어드밴스드, 5G, 등과 같은 셀룰러 통신 표준에 따라 변조된, 비트스트림과 같은 데이터를 수신하도록 구성될 수도 있다. 출력 인터페이스 (22) 가 무선 수신기를 포함하는 일부 예들에서, 출력 인터페이스 (22) 는 IEEE 802.15 사양 (예를 들어, 지그비™, 블루투스™ 표준, 등과 같은 다른 무선 표준들에 따라 변조된, 비트스트림과 같은 데이터를 수신하도록 구성될 수도 있다. 일부 예들에서, 출력 인터페이

스 (22) 의 회로부는 비디오 인코더 (20) 의 회로부 및/또는 소스 디바이스 (12) 의 다른 컴포넌트들에 통합될 수도 있다. 예를 들어, 비디오 인코더 (20) 및 출력 인터페이스 (22) 는 시스템 온 칩 (SoC) 의 부분들일 수도 있다. SoC 는 또한, 다른 컴포넌트들, 예컨대 범용 마이크로프로세서, 그래픽 프로세싱 유닛, 등을 포함할 수도 있다.

[0049] 목적지 디바이스 (14) 는 디코딩될 인코딩된 비디오 데이터를 컴퓨터 판독가능 매체 (16) 를 통해 수신할 수도 있다. 컴퓨터 판독가능 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. 일 예에서, 컴퓨터 판독가능 매체 (16) 는 소스 디바이스 (12) 로 하여금 인코딩된 비디오 데이터를 목적지 디바이스 (14) 로 실시간으로 직접적으로 송신하게 하는 통신 매체를 포함할 수도 있다. 인코딩된 비디오 데이터는 무선 통신 프로토콜과 같은 통신 표준에 따라 변조되고, 목적지 디바이스 (14) 로 송신될 수도 있다. 통신 매체는 임의의 무선 또는 유선 통신 매체, 예컨대 무선 주파수 (RF) 스펙트럼 또는 하나 이상의 물리적 송신 라인들을 포함할 수도 있다. 통신 매체는 패킷 기반의 네트워크, 예컨대 근거리 통신망, 광대역 통신망, 또는 인터넷과 같은 글로벌 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/또는 디코딩된 비디오 데이터를 저장하도록 구성된 하나 이상의 데이터 저장 매체를 포함할 수도 있다.

[0050] 일부 예들에서, 출력 인터페이스 (22) 는 인코딩된 데이터를 저장 디바이스로 출력할 수도 있다. 유사하게, 입력 인터페이스 (28) 는 저장 디바이스로부터 인코딩된 데이터에 액세스할 수도 있다. 저장 디바이스는 하드 드라이브, 블루레이 디스크들, DVD들, CD-ROM들, 플래시 메모리, 휘발성 또는 비휘발성 메모리와 같은 임의의 다양한 분산된 또는 로컬하게 액세스된 데이터 저장 매체, 또는 인코딩된 비디오 데이터를 저장하기 위한 임의의 다른 적합한 디지털 저장 매체를 포함할 수도 있다. 추가의 예에서, 저장 디바이스는 소스 디바이스 (12) 에 의해 생성된 인코딩된 비디오를 저장할 수도 있는 다른 중간 저장 디바이스 또는 파일 서버에 대응할 수도 있다. 목적지 디바이스 (14) 는 스트리밍 또는 다운로드를 통해 저장 디바이스로부터의 저장된 비디오 데이터에 액세스할 수도 있다. 파일 서버는 인코딩된 비디오 데이터를 저장할 수 있고 그 인코딩된 비디오 데이터를 목적지 디바이스 (14) 로 송신할 수 있는 임의의 유형의 서버일 수도 있다. 예시의 파일 서버들은 (예를 들어, 웹사이트 용의) 웹서버, 파일 전송 프로토콜 (FTP) 서버, NAS (network attached storage) 디바이스들, 또는 로컬 디스크 드라이브를 포함한다. 목적지 디바이스 (14) 는 인터넷을 접속을 포함하는 임의의 표준 데이터 접속을 통해 인코딩된 비디오 데이터에 액세스할 수도 있다. 이것은 파일 서버에 저장된 인코딩된 비디오 데이터에 액세스하는데 적합한 무선 채널 (예를 들어, 와이파이 접속), 유선 접속 (예를 들어, DSL, 케이블 모뎀 등), 또는 이들의 조합을 포함할 수도 있다. 저장 디바이스로부터의 인코딩된 비디오 데이터의 송신은 스트리밍 송신, 다운로드 송신, 또는 이들의 조합일 수도 있다.

[0051] 기법들은 다양한 멀티미디어 애플리케이션들, 예컨대 지상파 (over-the-air) 텔레비전 방송들, 케이블 텔레비전 송신들, 위성 텔레비전 송신들, HTTP 를 통한 동적 적응형 스트리밍 (DASH) 과 같은 인터넷 스트리밍 비디오 송신들, 데이터 저장 매체 상에 인코딩되는 디지털 비디오, 데이터 저장 매체 상에 저장된 디지털 비디오의 디코딩, 또는 다른 애플리케이션들 중 임의의 것의 지원으로 비디오 코딩에 적용될 수도 있다. 일부 예들에서, 시스템 (10) 은, 비디오 스트리밍, 비디오 플레이백, 비디오 방송, 및/또는 비디오 텔레포니와 같은 애플리케이션들을 지원하기 위해 일방향 또는 양방향 비디오 송신을 지원하도록 구성될 수도 있다.

[0052] 컴퓨터 판독가능 매체 (16) 는 트랜션트 (transient) 매체, 예컨대 무선 브로드캐스트 또는 유선 네트워크 송신, 또는 저장 매체 (즉, 비일시적 저장 매체), 예컨대 하드 디스크, 플래시 드라이브, 컴팩트 디스크, 디지털 비디오 디스크, 블루레이 디스크, 또는 다른 컴퓨터 판독가능 매체를 포함할 수도 있다. 일부 예들에서, 네트워크 서버 (미도시) 는 소스 디바이스 (12) 로부터 인코딩된 비디오 데이터를 수신하고, 예를 들어 네트워크 송신을 통해 인코딩된 비디오 데이터를 목적지 디바이스 (14) 에 제공할 수도 있다. 유사하게, 디스크 스탬핑 설비와 같은 매체 생성 설비의 컴퓨팅 디바이스가 소스 디바이스 (12) 로부터 인코딩된 비디오 데이터를 수신하고, 인코딩된 비디오 데이터를 포함하는 디스크를 생성할 수도 있다. 따라서, 컴퓨터 판독가능 매체 (16) 는, 다양한 형태들의 하나 이상의 컴퓨터 판독가능 매체를 포함하는 것으로 이해될 수도 있다.

[0053] 목적지 디바이스 (14) 의 입력 인터페이스 (28) 는 컴퓨터 판독가능 매체 (16) 로부터 정보를 수신한다. 입력 인터페이스 (28) 는 컴포넌트들 또는 디바이스들의 다양한 유형들을 포함할 수도 있다. 예를 들어, 입력 인터페이스 (28) 는 무선 수신기, 모뎀, 유선 네트워킹 컴포넌트 (예를 들어, 이더넷 카드), 또는 다른 물리적 컴포넌트를 포함할 수도 있다. 입력 인터페이스 (28) 가 무선 수신기를 포함하는 예들에서, 입력 인터페이

스 (28) 는 4G, 4G-LTE, LTE 어드밴스드, 5G, 등과 같은 셀룰러 통신 표준에 따라 변조된, 비트스트림과 같은 데이터를 수신하도록 구성될 수도 있다. 입력 인터페이스 (28) 가 무선 수신기를 포함하는 일부 예들에서, 입력 인터페이스 (28) 는 IEEE 802.11 사양, IEEE 802.15 사양 (예를 들어, 지그비™), 블루투스™ 표준, 등과 같은 다른 무선 표준들에 따라 변조된, 비트스트림과 같은 데이터를 수신하도록 구성될 수도 있다. 일부 예들에서, 입력 인터페이스 (28) 의 회로부는 비디오 디코더 (30) 의 회로부 및/또는 목적지 디바이스 (14) 의 다른 컴포넌트들에 통합될 수도 있다. 예를 들어, 비디오 디코더 (30) 및 입력 인터페이스 (28) 는 시스템 온 칩 (SoC) 의 부분들일 수도 있다. SoC 는 또한, 다른 컴포넌트들, 예컨대 범용 마이크로프로세서, 그래픽 프로세싱 유닛, 등을 포함할 수도 있다.

[0054] 컴퓨터 판독가능 매체 (16) 의 정보는 비디오 인코더 (20) 에 의해 정의된 신택스를 포함할 수도 있는데, 이것은 비디오 디코더 (30) 에 의해 또한 사용되며, 블록들 및 다른 코딩된 유닛들, 예를 들어 픽처들의 그룹 (GOP) 들의 프로세싱 및/또는 특징들을 설명하는 신택스 엘리먼트들을 포함한다. 디스플레이 디바이스 (32) 는 디코딩된 비디오 데이터를 사용자에게 디스플레이할 수도 있다. 예를 들어, 목적지 디바이스 (14) 또는 비디오 디코더 (30) 는, 비디오 데이터의 복원된 픽처들을, 디스플레이 디바이스 (32) 에 의한 디스플레이를 위해 출력할 수도 있다. 이러한 복원된 픽처들은 복원된 블록들을 포함할 수도 있다. 디스플레이 디바이스 (32) 는 다양한 디스플레이 디바이스들, 예컨대 음극선관 (CRT), 액정 디스플레이 (LCD), 플라즈마 디스플레이, 유기 발광 다이오드 (OLED) 디스플레이, 또는 다른 유형의 디스플레이 디바이스 중 임의의 것을 포함할 수도 있다.

[0055] 비디오 인코더 (20) 및 비디오 디코더 유닛 (30) 각각은 임의의 다양한 적합한 인코더 회로부, 예컨대 하나 이상의 마이크로프로세서들, 디지털 신호 프로세서 (DSP)들, 주문형 반도체들 (ASIC)들, 필드 프로그램가능 게이트 어레이 (FPGA)들, 이산 로직, 소프트웨어, 하드웨어, 펌웨어 또는 이들의 임의의 조합들로서 구현될 수도 있다. 이 기법들이 부분적으로 소프트웨어로 구현되는 경우, 디바이스는 그 소프트웨어에 대한 명령들을 적합한, 비일시적 컴퓨터 판독가능 매체에 저장할 수도 있고, 본 개시물의 기법들을 수행하기 위해 하나 이상의 프로세서들을 사용하는 하드웨어에서 그 명령들을 실행할 수도 있다. 비디오 인코더 (20) 및 비디오 디코더 (30) 각각은 하나 이상의 인코더들 또는 디코더들에 포함될 수도 있고, 이들 중 어느 것도 결합된 인코더/디코더 (CODEC) 의 일부로서 개별의 디바이스에 통합될 수도 있다.

[0056] 일부 예들에서, 비디오 인코더 (20) 및 비디오 디코더 (30) 는 비디오 코딩 표준에 따라 동작할 수도 있다. 예시의 비디오 코딩 표준들은 ITU-T H.261, ISO/IEC MPEG-1 비주얼, ITU-T H.262 또는 ISO/IEC MPEG-2 비주얼, ITU-T H.263, ISO/IEC MPEG-4 비주얼 및 자신의 SVC (Scalable Video Coding) 및 MVC (Multi-view Video Coding) 확장안들을 포함하는 ITU-T H.264 (또한, ISO/IEC MPEG-4 AVC 로 알려짐) 을 포함한다. 또한, 새로운 비디오 코딩 표준, 즉 고효율 비디오 코딩 (HEVC) 은 ITU-T 비디오 코딩 전문가 그룹 (VCEG) 및 ISO/IEC 동화상 전문가 그룹 (MPEG) 의 JCT-VC (Joint Collaboration Team on Video Coding) 에 의해 최근에 개발되고 있다. 2013 년 7월 25일 - 8월 2일, 오스트리아, 비엔나, ITU-T SG 16 WP3 및 ISO/IEC JTC1/SC29/WG11 의 JCT-VC (Joint Collaborative Team in Video Coding), 14 차 회의, Wang 등의, "High Efficiency Video Coding (HEVC) Defect Report", 문헌 JCTVC-N1003_v1 (이하에서, "JCTVC-N1003") 는 HEVC 표준의 초안이다. JCTVC-N1003 은 http://phenix.int-evry.fr/jct/doc_end_user/documents/14_Vienna/wg11/JCTVC-N1003-v1.zip 로부터 입수 가능하다.

[0057] HEVC 및 다른 비디오 코딩 사양들에서, 비디오 시퀀스는 통상적으로 일련의 픽처들을 포함한다. 픽처들은 또한, "프레임들" 로서 지칭될 수도 있다. 픽처는 하나 이상의 샘플 어레이들을 포함할 수도 있다. 픽처의 각각의 개별의 샘플 어레이는 개별의 컬러 컴포넌트에 대한 샘플들의 어레이를 포함할 수도 있다. HEVC 에서, 픽처는 S_L, S_{Cb}, 및 S_{Cr} 로 표기된 3 개의 샘플 어레이들을 포함할 수도 있다. S_L 은 루마 샘플들의 2-차원 어레이 (즉, 블록) 이다. S_{Cb} 는 Cb 크로마 샘플들의 2-차원 어레이이다. S_{Cr} 은 Cr 크로마 샘플들의 2-차원 어레이이다. 다른 경우들에서, 픽처는 모노크롬일 수도 있고 단지 루마 샘플들의 어레이를 포함할 수도 있다.

[0058] 비디오 데이터를 인코딩하는 부분으로서, 비디오 인코더 (20) 는 비디오 데이터의 픽처들을 인코딩할 수도 있다. 다시 말해, 비디오 인코더 (20) 는 비디오 데이터의 픽처들의 인코딩된 표현들을 생성할 수도 있다. 픽처의 인코딩된 표현은 "코딩된 픽처" 또는 "인코딩된 픽처" 로서 지칭될 수도 있다.

[0059] 픽처의 인코딩된 표현을 생성하기 위해, 비디오 인코더 (20) 는 코딩 트리 유닛 (CTU)들의 세트를 생성할 수도

있다. CTU들 각각은 루마 샘플들의 CTB, 크로마 샘플들의 2 개의 대응하는 CTB들, 및 CTB들의 샘플들을 코딩하는데 사용된 신택스 구조들을 포함할 수도 있다. 3 개의 별개의 컬러 평면들을 갖는 픽처들 또는 모노크롬 픽처들에서, CTU 는 단일의 CTB 및 CTB 의 샘플들을 코딩하는데 사용된 신택스 구조들을 포함할 수도 있다. CTB 는 샘플들의 N×N 블록일 수도 있다. CTU 는 또한, "트리 블록" 또는 "최대 코딩 유닛" (LCU) 으로서 지칭될 수도 있다. 신택스 구조는 지정된 순서로 비트스트림에 함께 존재하는 제로 또는 더 많은 신택스 엘리먼트들로서 정의될 수도 있다. 슬라이스는 래스터 스캔 순서로 연속적으로 오더링된 정수의 CTU 들을 포함할 수도 있다.

[0060] 본 개시물은 샘플들의 하나 이상의 블록들의 샘플들을 코딩하는데 사용된 하나 이상의 샘플 블록들 및 신택스 구조들을 지칭하기 위해 용어 "비디오 유닛" 또는 "비디오 블록" 또는 "블록" 을 사용할 수도 있다. 비디오 유닛들의 예시의 유형들은 CTU들, CU들, PU들, 변환 유닛 (TU)들, 매크로블록들, 매크로블록 파티션들 등을 포함할 수도 있다. 일부 맥락들에서, PU들의 논의는 매크로블록들 또는 매크로블록 파티션들의 논의와 상호교환될 수도 있다.

[0061] HEVC 에서, 코딩된 CTU 를 생성하기 위해, 비디오 인코더 (20) 는 CTU 의 코딩 트리 블록들 상에서 쿼드-트리 파티셔닝을 재귀적으로 수행하여 코딩 트리 블록들을 코딩 블록들, 따라서 명칭 "코딩 트리 유닛들" 로 분할할 수도 있다. 코딩 블록은 샘플들의 N×N 블록이다. CU 는 루마 샘플 어레이, Cb 샘플 어레이, 및 Cr 샘플 어레이를 갖는 픽처의 루마 샘플들의 코딩 블록 및 크로마 샘플들의 2 개의 대응하는 코딩 블록들, 및 코딩 블록들의 샘플들을 코딩하는데 사용된 신택스 구조들을 포함할 수도 있다. 3 개의 별개의 컬러 평면들을 갖는 픽처들 또는 모노크롬 픽처들에서, CU 는 단일의 코딩 블록 및 코딩 블록의 샘플들을 코딩하는데 사용된 신택스 구조들을 포함할 수도 있다.

[0062] 비디오 인코더 (20) 는 비디오 데이터의 픽처의 CU들을 인코딩할 수도 있다. CU 를 인코딩하는 부분으로서, 비디오 인코더 (20) 는 CU 의 코딩 블록을 하나 이상의 예측 블록들로 파티셔닝할 수도 있다. 예측 블록은, 동일한 예측이 적용되는 샘플들의 직사각형 (즉, 스퀘어 또는 비-스퀘어) 블록이다. 예측 유닛 (PU) 은 루마 샘플들의 예측 블록, 크로마 샘플들의 2 개의 대응하는 예측 블록들, 및 예측 블록들을 예측하는데 사용된 신택스 구조들을 포함할 수도 있다. 모노크롬 픽처들 또는 3 개의 별개의 컬러 평면들을 갖는 픽처들에서, PU 는 단일의 예측 블록 및 예측 블록을 예측하는데 사용된 신택스 구조들을 포함할 수도 있다. 비디오 인코더 (20) 는 CU 의 각각의 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.

[0063] 따라서, 일반적으로, PU 는 예측 블록들을 예측하는데 사용된 신택스 구조들 및 샘플들의 하나 이상의 예측 블록들을 포함할 수도 있다. HEVC 와 같은 일부 예시의 코덱들에서, PU 는 CU 의 서브-유닛일 수도 있다. 다른 예시의 코덱들에서, CU 와 PU 간에 구별이 존재하지 않을 수도 있다. 일부 예들에서, 다른 용어들이 PU 에 대해 사용될 수도 있다.

[0064] 비디오 인코더 (20) 는 인트라 예측 또는 인터 예측을 사용하여 PU 의 예측 블록을 생성할 수도 있다. 비디오 인코더 (20) 가 인트라 예측을 사용하여 PU 의 예측 블록을 생성하는 경우, 비디오 인코더 (20) 는 PU 를 포함하는 픽처의 디코딩된 샘플들에 기초하여 PU 의 예측 블록들을 생성할 수도 있다. 비디오 인코더 (20) 가 인터 예측을 사용하여 현재 픽처의 PU 의 예측 블록을 생성하는 경우, 비디오 인코더 (20) 는 레퍼런스 픽처 (즉, 현재 픽처 이외의 픽처) 의 디코딩된 샘플들에 기초하여 PU 의 예측 블록을 생성할 수도 있다.

[0065] 비디오 인코더 (20) 가 CU 의 하나 이상의 PU들에 대한 예측 블록들 (예를 들어, 루마, Cb 및 Cr 예측 블록들) 을 생성한 후에, 비디오 인코더 (20) 는 CU 에 대한 하나 이상의 잔차 블록들을 생성할 수도 있다. 예를 들어, 비디오 인코더 (20) 는 CU 에 대한 루마 잔차 블록을 생성할 수도 있다. CU 의 루마 잔차 블록에서 각각의 샘플은, CU 의 예측 루마 블록들 중 하나에서의 루마 샘플과 CU 의 원래의 루마 코딩 블록에서의 대응하는 샘플 간의 차이를 나타낼 수도 있다. 또한, 비디오 인코더 (20) 는 CU 에 대한 Cb 잔차 블록을 생성할 수도 있다. CU 의 Cb 잔차 블록에서 각각의 샘플은, CU 의 예측 Cb 블록들 중 하나에서의 Cb 샘플과 CU 의 원래의 Cb 코딩 블록에서의 대응하는 샘플 간의 차이를 나타낼 수도 있다. 비디오 인코더 (20) 는 또한, CU 에 대한 Cr 잔차 블록을 생성할 수도 있다. CU 의 Cr 잔차 블록에서 각각의 샘플은, CU 의 예측 Cr 블록들 중 하나에서의 Cr 샘플과 CU 의 원래의 Cr 코딩 블록에서의 대응하는 샘플 간의 차이를 나타낸다.

[0066] 반복하기 위해, HEVC 에서, 슬라이스에서의 최대 코딩 유닛은 CTU 로 지칭된다. 각각의 픽처는, 특정 타일 또는 슬라이스에 대해 래스터 스캔 순서로 코딩될 수도 있는, CTU들로 분할된다. CTU 는 스퀘어 블록이고 쿼드트리의 루트, 즉 코딩 트리를 나타낸다. CTU 는 쿼드-트리를 포함하고, 이것의 노드들은 CU들이다.

일부 경우들에서, CTU 의 사이즈는 HEVC 메인 프로파일에서 16×16 내지 64×64 의 범위일 수 있다 (하지만, 통상적으로 8×8 CTU 사이즈들이 지원될 수 있다). 일부 경우들에서, CTU 사이즈는 8×8 내지 64×64 루마 샘플들의 범위에 있을 수도 있지만, 통상적으로 64×64 이 사용된다. 각각의 CTU 는 CU들로 지칭된 더 작은 스퀘어 블록들로 추가로 스플리팅될 수 있다. CU 는 CTU 의 동일한 사이즈일 수도 있지만, CU 는 8×8 만큼 작을 수 있다. 각각의 CU 는 하나의 모드로 코딩된다. 예를 들어, CU 는 인터 코딩 또는 인트라 코딩될 수도 있다. CU 가 인터 코딩되는 경우, CU 는 또한 2 또는 4 개의 PU들로 파티셔닝될 수도 있거나 또는 추가의 파티션이 적용되지 않는 경우 단지 하나의 PU 가 될 수도 있다. 하나의 CU 에서 2 개의 PU들이 존재하는 경우, 2 개의 PU들은 CU 의 절반 사이즈 직사각형들 또는 ¼ 또는 ¾ 사이즈를 갖는 2 개의 직사각형 사이즈일 수 있다. CU 가 인터 코딩되는 경우, 모션 정보의 하나의 세트가 각각의 PU 에 대해 존재한다. 또한, 각각의 PU 는 고유한 인터-예측 모드로 코딩되어 모션 정보의 세트를 도출한다. 다시 말해, 각각의 PU 는 모션 정보의 그 자신의 세트를 가질 수도 있다.

[0067] 또한, 비디오 인코더 (20) 는 CU 의 잔차 블록들을 하나 이상의 변환 블록들로 분해할 수도 있다. 예를 들어, 비디오 인코더 (20) 는 쿼드-트리 파티셔닝을 사용하여, CU 의 잔차 블록들 (예를 들어, 루마, Cb, 및 Cr 잔차 블록들) 을 하나 이상의 변환 블록들 (예를 들어, 루마, Cb, 및 Cr 변환 블록들) 로 분해할 수도 있다. 변환 블록은, 동일한 변환이 적용되는 샘플들의 직사각형 (예를 들어, 스퀘어 또는 비-스퀘어) 블록이다.

[0068] CU 의 변환 유닛 (TU) 은 루마 샘플들의 변환 블록, 크로마 샘플들의 2 개의 대응하는 변환 블록들, 및 변환 블록 샘플들을 변환하는데 사용된 신택스 구조들을 포함할 수도 있다. 따라서, CU 의 각각의 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. 3 개의 별개의 컬러 평면들을 갖는 픽처들 또는 모노크롬 픽처들에서, TU 는 단일의 변환 블록 및 변환 블록의 샘플들을 변환하는데 사용된 신택스 구조들을 포함할 수도 있다.

[0069] 비디오 인코더 (20) 는 TU 의 변환 블록에 하나 이상의 변환들을 적용하여 TU 에 대한 계수 블록을 생성할 수도 있다. 예를 들어, 비디오 인코더 (20) 는 TU 의 루마 변환 블록에 하나 이상의 변환들을 적용하여 TU 에 대한 루마 계수 블록을 생성할 수도 있다. 계수 블록은 변환 계수들의 2-차원 어레이일 수도 있다. 변환 계수는 스칼라 양일 수도 있다. 일부 예들에서, 하나 이상의 변환들은 변환 블록을 픽셀 도메인에서 주파수 도메인으로 컨버팅한다.

[0070] 일부 예들에서, 비디오 인코더 (20) 는 변환 블록에 변환을 적용하지 않는다. 다시 말해, 비디오 인코더 (20) 는 변환 블록에 대한 변환들의 적용을 스킵한다. 이러한 예들에서, 비디오 인코더 (20) 는 변환 계수들과 동일한 방식으로 잔차 샘플 값들을 취급할 수도 있다. 따라서, 비디오 인코더 (20) 가 변환들의 적용을 스킵하는 예들에서, 변환 계수들 및 계수 블록들의 다음의 논의는 잔차 샘플들의 변환 블록들에 적용 가능할 수도 있다.

[0071] 계수 블록 (예를 들어, 루마 계수 블록, Cb 계수 블록 또는 Cr 계수 블록) 을 생성한 후에, 비디오 인코더 (20) 는 계수 블록을 양자화할 수도 있다. 일부 예들에서, 비디오 인코더 (20) 는 계수 블록을 양자화하지 않는다. 비디오 인코더 (20) 가 변환 블록에 변환을 적용하지 않는 예들에서, 비디오 인코더 (20) 는 변환 블록의 잔차 샘플들을 양자화 또는 양자화하지 않을 수도 있다. 양자화는 일반적으로, 계수들을 표현하기 위해 사용된 데이터의 양을 가능하게는 감소시키기 위해 변환 계수들이 양자화되어, 추가의 압축을 제공하는 프로세스를 지칭한다. 비디오 인코더 (20) 가 계수 블록을 양자화한 후에, 비디오 인코더 (20) 는 양자화된 변환 계수들 또는 잔차 샘플들을 나타내는 신택스 엘리먼트들을 엔트로피 인코딩할 수도 있다. 예를 들어, 비디오 인코더 (20) 는 양자화된 변환 계수들 또는 잔차 샘플들을 나타내는 신택스 엘리먼트들 상에 콘텍스트-적응 바이너리 산술 코딩 (CABAC) 을 수행할 수도 있다. 일부 예들에서, 비디오 인코더 (20) 는 팔레트-기반의 코딩을 사용하여 CU들을 인코딩한다. 따라서, 인코딩된 블록 (예를 들어, 인코딩된 CU) 은 양자화된 변환 계수들을 나타내는 엔트로피 인코딩된 신택스 엘리먼트들을 포함할 수도 있다.

[0072] 비디오 인코더 (20) 는 비디오 데이터의 인코딩된 픽처들 및 연관된 데이터 (즉, 인코딩된 픽처들과 연관된 데이터) 의 표현을 형성하는 비트들의 시퀀스를 포함하는 비트스트림을 출력할 수도 있다. 따라서, 비트스트림은 비디오 데이터의 인코딩된 표현을 포함한다. 비트스트림은 네트워크 추상 계층 (NAL) 유닛들의 시퀀스를 포함할 수도 있다. NAL 유닛은, NAL 유닛에서의 데이터 유형의 표시를 포함하는 신택스 구조 및 에뮬레

이션 방지 비트들과 필요에 따라 배치된 RBSP (raw byte sequence payload) 의 형태의 그 데이터를 포함하는 바이트들이다. NAL 유닛들 각각은 NAL 유닛 헤더를 포함하고 RBSP 를 캡슐화할 수도 있다. NAL 유닛 헤더는 NAL 유닛 유형 코드를 나타내는 신택스 엘리먼트를 포함할 수도 있다. NAL 유닛의 NAL 유닛 헤더에 의해 지정된 NAL 유닛 유형 코드는 NAL 유닛의 유형을 나타낸다. RBSP 는 NAL 유닛 내에 캡슐화되는 정수의 바이트들을 포함하는 신택스 구조일 수도 있다. 일부 경우들에서, RBSP 는 제로 비트들을 포함한다.

[0073] 비디오 디코더 (30) 는 비디오 인코더 (20) 에 의해 생성된 비트스트림을 수신할 수도 있다. 또한, 비디오 디코더 (30) 는 비트스트림을 파싱 (parse) 하여 비트스트림으로부터 신택스 엘리먼트들을 획득할 수도 있다. 비디오 디코더 (30) 는 비트스트림으로부터 획득된 신택스 엘리먼트들에 적어도 부분적으로 기초하여 비디오 데이터의 픽처들을 복원할 수도 있다. 비디오 데이터의 픽처들을 복원하기 위한 프로세스는 픽처들을 인코딩하기 위해 비디오 인코더 (20) 에 의해 수행된 프로세스에 일반적으로 상반될 수도 있다. 예를 들어, 비디오 데이터의 픽처를 복원하기 위해, 비디오 디코더 (30) 는 비트스트림으로부터 획득된 신택스 엘리먼트들 및/또는 외부 소스들로부터의 데이터에 기초하여 픽처의 블록들, 예컨대 CU들을 디코딩할 수도 있다.

[0074] 일부 예들에서, 픽처의 현재 CU 를 디코딩하는 것의 부분으로서, 비디오 디코더 (30) 는 인터 예측 또는 인트라 예측을 사용하여 현재 CU 의 각각의 PU 에 대한 하나 이상의 예측 블록들을 생성할 수도 있다. 인터 예측을 사용하는 경우, 비디오 디코더 (30) 는 PU들의 모션 벡터들을 사용하여 현재 CU 의 PU들에 대한 예측 블록들을 결정할 수도 있다. 또한, 비디오 디코더 (30) 는, 일부 예들에서 현재 CU 의 TU들의 계수 블록들을 역 양자화할 수도 있다. 비디오 디코더 (30) 는, 일부 예들에서 계수 블록들 상에서 역 변환들을 수행하여 현재 CU 의 TU들의 변환 블록들을 복원할 수도 있다. 비디오 디코더 (30) 는 현재 CU 의 PU들에 대한 예측 블록들의 샘플들을 현재 CU 의 TU들의 변환 블록들의 대응하는 디코딩된 샘플들 (예를 들어, 잔차 샘플들) 에 추가함으로써 현재 CU 의 코딩 블록들을 복원할 수도 있다. 픽처의 각각의 CU 에 대한 코딩 블록들을 복원함으로써, 비디오 디코더 (30) 는 픽처를 복원할 수도 있다.

[0075] 또한, HEVC 에서, 픽처를 타일로 지칭된 직사각형 영역들로 파티셔닝하기 위한 옵션이 지정되어 있다. 타일들의 주요 목적은 에러 복원력을 제공하기 보다는 차라리 병렬 프로세싱에 대한 능력을 증가시키는 것이다. 타일들은 일부 공유된 헤더 정보와 인코딩되는 픽처의 영역들을 독립적으로 디코딩 가능하다. 타일들은 부가적으로, 비디오 픽처들의 로컬 영역들에 대한 공간적 랜덤 액세스의 목적을 위해 사용될 수 있다. 픽처의 통상적인 타일 구성은 각각의 타일에서 CTU들의 거의 동일한 수들을 갖는 직사각형 영역들로 픽처를 세그먼트하는 것으로 이루어진다. 타일들은 더 거친 레벨의 입도 (픽처/서브픽처) 로 병렬 처리를 제공하며, 그들의 사용을 위해 스레드들의 정교한 동기화가 필요하지 않다.

[0076] 잔차 블록들의 다양한 특징들을 적응시키기 위해, 잔차 쿼드트리 (RQT) 를 사용하는 변환 코딩 구조는 HEVC 에서 적용되고, 이것은 Marpe 등의, "Transform Coding Using the Residual Quadtree (RQT)" Fraunhofer Heinrich Hertz Institute 에서 간단히 설명되며, http://www.hhi.fraunhofer.de/fields-of-competence/image-processing/researchgroups/image-video-coding/hevc-high-efficiency-video-coding/transform-coding-using-the-residual-quadtree-rqt.html 에서 입수 가능하다. CTU 가 CU들로 재귀적으로 스플리팅된 후에, 각각의 CU 는 PU들 및 TU들로 추가로 분할된다. CU 의 TU들로의 파티셔닝은 쿼드트리 접근에 기초하여 재귀적으로 수행되고, 따라서 각각의 CU 의 잔차 신호는 트리 구조, 즉 잔차 쿼드트리 (RQT) 에 의해 코딩된다. RQT 는 4×4 에서부터 최대 32×32 까지의 루마 샘플들의 TU 사이즈들을 허용한다. 도 2a 및 도 2b 는 HEVC 에서의 잔차 쿼드트리에 기초하여 예시의 변환 스킴을 예시하는 개념도들이다. 특히, 도 2a 는 CU (40) 가 문자들 a 내지 j 로 라벨링된, 10 개의 TU들, 및 대응하는 블록 파티셔닝을 포함하는 예를 나타낸다. 도 2b 는 도 2a 의 CU 에 대한 RQT 를 예시하는 개념도이다.

[0077] 비디오 코더는, 심도-우선 횡단으로 재귀적 Z-스캔을 따르는, 알파벳 순서로 도 2a 에 예시되는 심도-우선 (depth-first) 트리 횡단 순서로 개별의 TU들을 프로세싱할 수도 있다. 쿼드트리 접근은 잔차 신호의 가변하는 공간-주파수 특징에 변환의 적응을 가능하게 한다. 통상적으로, 더 큰 공간 지원을 갖는 더 큰 변환 블록 사이즈들은 더 좋은 주파수 레졸루션을 제공한다. 그러나, 더 작은 공간 지원을 갖는 더 작은 변환 블록 사이즈들은 더 좋은 공간 레졸루션을 제공한다. 2 개의, 공간 및 주파수 레졸루션들 간의 트레이드-오프는, 예를 들어 레이트-왜곡 최적화 기법에 기초하여 인코더 모드 판정에 의해 선택된다. 레이트-왜곡 최적화 기법은, 각각의 코딩 모드 (예를 들어, 특정의 RQT 스플리팅 구조) 에 대한, 코딩 비트들의 가중된 합 및 복원 왜곡, 즉 레이트-왜곡 비용을 계산하고, 최소 레이트-왜곡 비용을 갖는 코딩 모드를 최선의 모드로서 선택한다.

[0078] 3 개의 파라미터들은 RQT 에서 정의된다: 트리의 최대 심도, 최소 허용된 변환 사이즈 및 최대 허용된 변환 사이즈. HEVC 에서, 최소 및 최대 변환 사이즈들은, 이전 단락에서 언급된 지원된 블록 변환들에 대응하는, 4×4 내지 32×32 샘플들의 범위 내에서 변할 수 있다. RQT 의 최대 허용된 심도는 TU들의 수를 제한한다. 0 과 동일한 최대 심도는, CTU 가, 각각의 포함된 TU 가 최대 허용된 변환 사이즈, 예를 들어 32×32 에 도달하면 더욱 더 스플리팅될 수 없다는 것을 의미한다. HEVC 에서, 더 큰 사이즈 변환들, 예를 들어 64×64 변환은, 상대적으로 더 작은 레졸루션 비디오들에 대한 상대적으로 더 높은 복잡성 및 고려 중인 그들의 제한된 이점으로 인해 주로 채택되지 않았다.

[0079] HEVC 에서, TU 의 사이즈에 관계 없이, TU 의 잔차 (예를 들어, TU 의 계수 블록) 는 비-오버랩된 계수 그룹들 (CG) 과 코딩된다. CG들 각각은 TU 의 4×4 블록의 계수들을 포함한다. 예를 들어, 32×32 TU 는 총 64 개의 CG들을 갖고, 16×16 TU 는 총 16 개의 CG들을 갖는다. TU 의 CG들은 소정의 미리-정의된 스캔 순서에 따라 코딩된다. 각각의 CG 를 코딩할 때, 현재 CG 내의 계수들은 4×4 블록에 대한 소정의 미리-정의된 스캔 순서에 따라 스캐닝 및 코딩된다. 도 3 은 HEVC 에서의 계수 그룹들에 기초하여 예시의 계수 스캔을 예시하는 개념도이다. 특히, 도 3 은 4 개의 4×4 CG들을 포함하는 8×8 TU 에 대한 계수 스캔을 예시한다.

[0080] 위에서 주목된 바와 같이, 비디오 인코더 (20) 및 비디오 디코더 (30) 는 인트라 예측을 수행하여 예측 블록을 생성할 수도 있다. 인트라 예측은 그 공간적으로 이웃하는 복원된 이미지 샘플들을 사용하여 이미지 블록 예측을 수행한다. 도 4 는 16×16 블록에 대한 인트라 예측의 예를 예시하는 개념도이다. 도 4 에서, 블록 스퀘어는 16×16 블록 (50) 을 포함한다. 도 4 에서, 블록 (50) 은 선택된 예측 방향을 따라 상부 및 좌측 이웃하는 복원된 샘플들 (52, 54)(즉, 레퍼런스 샘플들) 에 의해 예측된다. 도 4 에서, 블랙 박스 밖의 샘플들은 레퍼런스 샘플들이다. 도 4 에서 화이트 화살표는 선택된 예측 방향을 나타낸다.

[0081] 도 5 는 HEVC 에서 정의된 35 개의 인트라 예측 모드들의 예를 예시하는 개념도이다. 도 5 에 나타낸 바와 같이, HEVC 는 루마 블록의 인트라 예측에 대해 (평면 모드, DC 모드 및 33 개의 각 모드들을 포함하는) 35 개의 모드들을 정의한다. HEVC 에서 정의된 인트라 예측의 35 개의 모드들은 이하의 테이블에서 도시된 바와 같이 인덱싱된다:

[0082] **표 1 - 인트라 예측 모드 및 연관된 명칭들의 사양**

인트라 예측 모드	연관된 명칭
0	INTRA_PLANAR
1	INTRA_DC
2..34	INTRA_ANGULAR2..INTRA_ANGULAR34

[0083]

[0084] HEVC 인트라 코딩은 PU 분할의 2 개의 유형들, 2N×N 및 N×N 을 지원한다. 2N×N 은 CU 를 하나의 PU 로 스플리팅한다. 다시 말해, CU 는 CU 와 동일한 사이즈를 갖는 하나의 PU 를 갖는다. N×N 은 CU 를 4 개의 동일한-사이즈의 PU들로 스플리팅한다. 그러나, 파티셔닝 유형 PART_N×N 에 의해 지정된 4 개의 영역들은 파티셔닝 유형 PART_2N×2N 을 갖는 4 개의 더 작은 CU들에 의해 표현될 수 있다. 이 때문에, HEVC 는 인트라 CU 가 최소 CU 사이즈에서만 4 개의 PU들로 스플리팅되도록 허용한다.

[0085] 도 6 은 HEVC 에서 정의된 평면 모드를 예시하는 개념도이다. 평면 모드는 통상적으로, 가장 빈번하게 사용된 인트라 예측 모드이다. N×N 블록에 대한 평면 예측을 수행하기 위해, (x, y) 에 위치된 각각의 샘플 p_{xy} 에 대해, 예측 값은 쌍일차 필터 (bilinear filter) 로 4 개의 특정의 이웃하는 복원된 샘플들, 즉 레퍼런스 샘플들을 사용하여 계산된다. 4 개의 레퍼런스 샘플들은 상단-우측 복원된 샘플 TR, 하단-좌측 복원된 샘플 BL, 및 현재 샘플과 동일한 컬럼 ($r_{x,-1}$) 및 로우 ($r_{-1,y}$) 에 위치된 2 개의 복원된 샘플들 (60, 62) 을 포함한다.

평면 모드는 이하에서와 같이 수식화될 수 있다:

[0086] $$p_{xy} = ((N-1-x)\cdot L + (x+1)\cdot TR + (N-1-y)\cdot T + (y+1)\cdot BL + N) >> (Log2(N)+1) \quad (1)$$

[0087] 위의 수식 (1) 에서, L 은 복원된 샘플 (60) 에 대응하고 T 는 복원된 샘플 (62) 에 대응한다. DC 모드에

대해, 예측 블록은 이웃하는 복원된 샘플들의 평균 값으로 단순히 필터링된다. 일반적으로, 평면 및 DC 모드들 양자 모두는 평활하게 변하고 일정한 이미지 영역들을 모델링하기 위해 적용된다.

[0088] 도 7 은 HEVC 에서 정의된 예시의 각 모드의 개념도이다. HEVC 에서 각 인트라 예측 모드들에 대한 인트라 예측 프로세스는 다음과 같이 설명된다. 각각의 주어진 각 인트라 예측 모드에 대해, 인트라 예측 방향이 이에 따라 식별될 수 있다. 예를 들어, 주어진 각 인트라 예측 모드는 도 5 에 따라 식별될 수도 있다. 도 5 에 도시된 바와 같이, 인트라 모드 (18) 는 순 (pure) 수평 예측 방향에 대응하고, 인트라 모드 (26) 는 순 수직 예측 방향에 대응한다. 특정 인트라 예측 방향이 제공되면, 예측 블록의 각각의 개별의 샘플에 대해, 개별의 샘플의 좌표들 (x, y) 은 예측 방향을 따라 이웃하는 복원된 샘플들의 로우 또는 컬럼으로 먼저 프로젝팅된다. 예를 들어, 도 7 의 예에 도시된 바와 같이, 예측 블록 (72) 의 샘플 (70) 의 좌표들 (x, y) 은 특정 인트라 예측 방향 (74) 을 따라 프로젝팅된다. (x,y) 는 2 개의 이웃하는 복원된 샘플들 (L 과 R) 사이의 부분 포지션 (fractional position)(α) 로 프로젝팅된다고 가정한다. 그 후, (x,y) 에 대한 예측 값은 다음과 같이 수식화된, 2-탭 양방향-선형 보간 필터를 사용하여 계산된다:

[0089] $$p_{xy} = (1-\alpha)\cdot L + \alpha\cdot R.$$

[0090] HEVC 에서, 부동 소수점 연산들을 회피하기 위해, 상기의 계산은 정수 계산을 사용하여 다음과 같이 근사화된다:

[0091] $$p_{xy} = ((32-a)\cdot L + a\cdot R + 16) >> 5,$$

[0092] 여기서, a 는 32*α 와 동일한 정수이다.

[0093] 도 8 은 HEVC 에서 인터 예측을 위해 CU 를 스플리팅하기 위한 파티션 모드들의 개념도이다. 도 8 에 도시된 바와 같이, HEVC 에서, 인터-코딩된 CU 는 1, 2, 또는 4 개의 파티션들로 스플리팅될 수 있고, 이 스플리팅의 다양한 유형들이 가능하다. 인터-예측된 코딩 블록들에 대한 파티셔닝 가능성들은 도 8 에 도시된다. 상위 4 개의 파티션 유형들은 사이즈 N×N 의 CU 를 스플리팅하지 않는 경우, CU 를 사이즈 N×N/2 또는 N/2×N 의 2 개의 파티션들로 스플리팅하는 경우, 및 CU 를 사이즈 N/2×N/2 의 4 개의 파티션들로 스플리팅하는 경우를 각각 예시한다. 도 8 에서 하위 4 개의 파티션 유형들은 비대칭적인 모션 파티셔닝 (AMP) 으로서 지칭된다. AMP 모드의 하나의 파티션은 높이 또는 폭 N/4 및 폭 또는 높이 N 을 각각 갖고, 다른 파티션은 3N/4 의 높이 또는 폭 및 N 의 폭 또는 높이를 가짐으로써 CU 의 나머지를 채운다. 각각의 인터-코딩된 파티션에는 1 또는 2 개의 모션 벡터들 및 레퍼런스 픽처 인덱스들이 할당된다.

[0094] 인트라 슬라이스들에 대해, 단지 인트라 예측 모드가 허용된다. 따라서, 예측 모드를 시그널링할 필요가 없다. 그러나, 인터 슬라이스들 (P 또는 B 슬라이스들) 에 대해, 인트라 및 인터 예측 모드 양자 모두가 허용된다. 따라서, HEVC 에서, 각각의 CU 에 대해, 하나의 플래그 pred_mode_flag 가 비-스킵 모드에 대해 시그널링된다. CU 에 대한 HEVC 에서 정의된 신택스 및 시맨틱들의 부분 리스팅은 이하에서 다음과 같이 제시된다:

[0095] 7.3.8.5 코딩 유닛 신택스

coding_unit(x0, y0, log2CbSize) {	디스크립터
if(transquant_bypass_enabled_flag)	
cu_transquant_bypass_flag	ae(v)
if(slice_type != I)	
cu_skip_flag[x0][y0]	ae(v)
nCbS = (1 << log2CbSize)	
if(cu_skip_flag[x0][y0])	
prediction_unit(x0, y0, nCbS, nCbS)	
else {	
if(slice_type != I)	
pred_mode_flag	ae(v)
if(CuPredMode[x0][y0] != MODE_INTRA || log2CbSize == MinCbLog2SizeY)	
part_mode	ae(v)
…	
}	

[0096]

[0097] 1 과 동일한 **cu_skip_flag**[x0][y0] 는, 현재 코딩 유닛에 대해, P 또는 B 슬라이스를 디코딩하는 경우, cu_skip_flag[x0][y0] 후에 머징 후보 인덱스 merge_idx[x0][y0] 외의 어느 신택스 엘리먼트들도 파싱되지 않는다는 것을 지정한다. 0 과 동일한 cu_skip_flag[x0][y0] 는, 코딩 유닛이 스킵되지 않는다는 것을 지정한다. 어레이 인덱스들 x0, y0 는, 픽처의 상단-좌측 루마 샘플에 대한 고려된 코딩 블록의 상단-좌측 루마 샘플의 로케이션 (x0, y0) 을 지정한다. cu_skip_flag[x0][y0] 이 존재하지 않는 경우, cu_skip_flag[x0][y0] 는 0 과 동일한 것으로 추론된다.

[0098] 0 과 동일한 **pred_mode_flag** 는, 현재 코딩 유닛이 인터 예측 모드로 코딩된다는 것을 지정한다. 1 과 동일한 pred_mode_flag 는, 현재 코딩 유닛이 인트라 예측 모드로 코딩된다는 것을 지정한다. 변수 CuPredMode[x][y] 는 x = x0..x0 + nCbS - 1 및 y = y0..y0 + nCbS - 1 에 대해 다음과 같이 도출된다:

[0099] - pred_mode_flag 가 0 과 동일하면, CuPredMode[x][y] 는 MODE_INTER 와 동일하게 설정된다.

[0100] - 그렇지 않으면 (pred_mode_flag 가 1 과 동일하면), CuPredMode[x][y] 는 MODE_INTRA 와 동일하게 설정된다.

[0101] pred_mode_flag 가 존재하지 않는 경우, 변수 CuPredMode[x][y] 는 x = x0..x0 + nCbS - 1 및 y = y0..y0 + nCbS - 1 에 대해 다음과 같이 도출된다:

[0102] - slice_type 이 I 와 동일하면, CuPredMode[x][y] 는 MODE_INTRA 와 동일한 것으로 추론된다.

[0103] - 그렇지 않으면 (slice_type 이 P 또는 B 이면), cu_skip_flag[x0][y0] 이 1 과 동일한 경우, CuPredMode[x][y] 는 MODE_SKIP 과 동일한 것으로 추론된다.

[0104] HEVC 를 개발하는 프로세스 동안 및 그 후에 HEVC 를 강화시키기 위한 다양한 제안들이 이루어져 왔다. 예를 들어, 바르샤바, 폴란드, 52 차 회의: 2015년 6월 19-26일, 문헌: VCEG-AZ07_v2, Jianle Chen 등의, "Further improvements to HMKTA-1.0" (이하에서, "VCEG-AZ07") 은 단거리 인트라 코딩 스킴을 설명한다. 인트라 예측을 위해 스퀘어 블록들을 항상 생성하는 전통적인 블록 파티션 방법들과 달리, VCEG-AZ07 의 단거리 인트라 예측 (SDIP) 스킴은 HEVC 의 쿼드트리 기반의 블록 구조 하에서 비-스퀘어 블록 스플리팅을 이용한다.

VCEG-AZ07 에서 설명된 바와 같이, 블록은 1/4 폭 또는 높이를 갖는 4 개의 비-스퀘어 블록들을 스플리팅되고, 각각의 비-스퀘어 블록은 예측을 위한 기본 유닛으로서 취급된다. 비-스퀘어 블록들은 차례로 코딩 및 복원되고, 다음의 이웃하는 블록에 대한 인트라 예측을 위해 레퍼런스 픽셀들을 제공할 수 있다. 따라서,

레퍼런스 픽셀들과 로컬 픽셀들 간의 거리는 감소될 수 있고, 인트라 예측의 정확도는 많이 개선될 수 있다.

[0105] 도 9 는 SDIP 유닛 파티션들의 개념도이다. SDIP 스킴에서, 64×64 보다 더 작은 CU 는 사이즈들 N/2×2N 또는 2N×N/2 를 갖는 4 개의 수직 또는 수평 직사각형 PU들로 스플리팅될 수 있다 (이들 파티션 모드들은 본 개시물에서 hN×2N 및 2N×hN 으로서 지칭될 수도 있고, 여기서 h 는 절반을 의미한다). 4 개의 PU들은, hN×2N 모드에서 좌측에서 우측으로, 그리고 2N×hN 모드에서 상단에서 하단으로 차례로 코딩 및 복원된다. 도 9 에서, 점선들은 PU/TU 스플리팅을 나타내고, 음영진 영역 (78) 은 RQT 구조에서 4 개의 32×2 TU들로 스플리팅될 32×8 TU 를 가리킨다. 파티션 모드 2N×hN 을 갖는 상위 우측 32×32 CU 는 4 개의 32×8 PU들로 스플리팅되고, 모드 hN×2N 를 갖는 하위 좌측 16×16 CU 는 4 개의 4×16 PU들로 스플리팅되며, 하위 우측 8×8 CU 는 8×2 PU들로 스플리팅되는 등등이다. 2N×2N 모드에서 하위 좌측 16×16 CU 및 N×N 모드에서 하위 우측 8×8 CU 와 같은, HEVC 에서의 CU들의 스퀘어 스플리팅이 또한 존재할 수도 있다.

[0106] 또한, SDIP 스킴에서, M×N (M > N) TU 는 M < N 인 경우 사이즈 M×N/4, 또는 M/4×N 인 4 개의 TU 들로 스플리팅될 수 있다. 다시 말해, SDIP 스킴에서 스플릿은 항상, CU 에서 동일한 방향 (수직 또는 수평) 을 따라 수행되어야 한다. 표 2 는, 이하에서 SDIP 스킴 및 대응하는 RQT 심도에 존재하는 비-스퀘어 유닛들 모두를 나열한다.

[0107] **표 2 - SDIP 스킴에서의 유닛들 및 대응하는 RQT 심도의 리스트**

CU 사이즈	(PU 사이즈와 동일한) 심도 = 1 인 경우의 유닛 사이즈	심도 = 2 인 경우의 유닛 사이즈
32×32	32×8	32×2
	8×32	2×32
16×16	16×4	-
	4×16	-
8×8	8×2	-
	2×8	-

[0108]

[0109] 2011년 3월 16-23일, 제네바, ITU-T SG16 WP3 및 ISO/IEC JTC1/SC29/WG11 의 JCT-VC (Joint Collaborative Team on Video Coding), 5 차 회의, 문헌 JCTVC-E0278, Xiaoran Cao 등의, "CE6.b1 Report on Short Distance Intra Prediction Method" (이하에서, "Cao 1") 에서, 1×N 및 N×1 과 같은 파티션들이 더 포함되고, 대응하는 RQT 심도 및 변환 사이즈들은 이하의, 표 3 에 나열된다:

[0110] **표 3 - Cao 1 의 SDIP 에서의 변환 사이즈들의 리스트**

CU 사이즈	(PU 사이즈와 동일한) 심도 = 1 인 경우의 유닛 사이즈	심도 = 2 인 경우의 유닛 사이즈
32×32	32×8	32×2
	8×32	2×32
16×16	16×4	16×1
	4×16	1×16
8×8 (2N×2N)	8×2	-
	2×8	-
8×8 (N×N)	4×4	4×1
		1×4

[0111]

[0112] 또한, 일부 SDIP 스킴들은 비-스퀘어 변환 및 엔트로피 코딩을 사용한다. 예를 들어, n x m 변환이 비-스퀘

어 블록에 대해 사용된다. n x m (n > m) 블록에 대해, 순방향 변환이 다음과 같이 설명된다:

[0113] $$C_{nxm} = T_m \cdot B_{nxm} \cdot T_n^T \quad (1).$$

[0114] 상기의 등식에서, B_{nxm} 은 n 개의 로우들 및 m 개의 컬럼들을 갖는 블록을 가리키고, T_n 및 T_m 은 사이즈 n x n 및 m x m 의 변환 매트릭스들이며, C_{nxm} 은 변환 블록을 가리킨다. T_n 및 T_m 은 HEVC 에서의 변환 매트릭스들과 동일하다. 따라서, 하드웨어 구현을 위해, 변환 부분은 비-스퀘어 블록들에 대해 재사용될 수 있다. n x m (n < m) 블록에 대해, 블록은 먼저 m x n (m > n) 블록으로 트랜스포즈되고 그 후, 등식 (1) 에서와 같이 변환된다. 엔트로피 코딩에 있어서, 중복 구현들을 회피하기 위해, 스퀘어 블록의 계수 코딩이 또한, 재사용된다. 예를 들어, 도 10 은 8×8 매트릭스 (82) 로 스캐닝 및 재구성된 16×4 계수 매트릭스 (80) 의 개념도이다. 이 예에서, 계수 매트릭스 (80) 의 계수들은 도 10 에 도시된 바와 같이, 먼저 1D 버퍼 (84) 으로 고 주파수에서 저 주파수로 스캐닝되고, 그 후 HEVC 에서 기존의 방법을 사용하여 코딩되는, 지그재그 순서로 8×8 매트릭스 (82) 로 재구성된다.

[0115] HEVC 에 대한 제안된 강화의 다른 예에서, 7 차 회의: 제네바, 2011년 11월 21-30일, ITU-T SG16 WP3 및 ISO/IEC JTC1/SC29/WG11 의 JCT-VC (Joint Collaborative Team on Video Coding), 문헌 JCTVC-G135, Liu 등의, "Rectangular (2N×N, N×2N) Intra Prediction" (이하에서, "JCTVC-G135") 은 인터 코딩에 대한 2N×N 및 N×2N 파티션 사이즈들의 사용을 인트라 코딩으로 확장하는 것을 설명한다. 추가적인 PU 사이즈들 및 대응하는 TU들은 이하의 표 2 에서 제공된다. JCTVC-G135 에서, 종래의 변환 쿼드트리 구조가 이용된다.

[0116] **표 2- JCTVC-G135 에서의 변환 사이즈들의 리스트**

CU 사이즈	PU 사이즈	심도 = 1 인 경우의 유닛 사이즈	심도 = 2 인 경우의 유닛 사이즈
32×32	32×16	32×8	32×2
	16×32	8×32	2×32
16×16	16×8	16×4	-
	8×16	4×16	-
8×8	8×4	8×2	-
	4×8	2×8	-

[0117]

[0118] 다음의 기법들은 VCEG-AZ07 에서 설명되었다. 자연스러운 비디오들에서 제시된 미세한 에지 방향들을 캡처하기 위해, VCEG-AZ07 은 HEVC 에서 정의된 바와 같이 33 에서부터 65 로 방향성 인트라 모드들을 확장시키는 것을 제안하였다. 도 11 은 제안된 67 개의 인트라 예측 모드들을 예시하는 개념도이다. VCEG-AZ07 에서 설명된 방향성 모드들은 도 11 에서 점선 화살표들로 표시되고, 평면 및 DC 모드들이 동일한 채로 있다. VCEG-AZ07에서 제안된 조밀한 방향성 인트라 예측 모드들은 모든 PU 사이즈들 및 루마 및 크로마 인트라 예측들 양자 모두에 대해 적용한다.

[0119] 증가된 수의 방향성 인트라 모드들에 부합하기 위해, VCEG-AZ07 은, 6 개의 최대 확률 모드 (MPM)들을 사용하여, 개선된 인트라 모드 코딩 방법을 제안하였다. 2 개의 주요 기술적 양태들이 수반된다: 1) 6 MPM들의 도출, 및 2) 6 MPM들으 엔트로피 코딩. 6 개의 MPM들의 세트를 도출하는 경우, VCEG-AZ06 은 좌측 및 상부의 이웃하는 인트라 모드들의 정의를 변경하였다. HEVC 에서와 같이 상단 및 좌측 이웃하는 블록들로부터 직접적으로 인트라 모드들을 사용하는 대신에, 상단의 이웃하는 로우를 따라 및 좌측의 이웃하는 컬럼을 따라 가장 빈번하게 사용된 인트라 모드가 연산되고, 그 후 좌측 및 상부 이웃하는 모드들로서 각각 사용된다.

[0120] 또한, VCEG-AZ07 에서 설명된 바와 같이, 4-탭 인트라 보간 필터들이 이용되어 방향성 인트라 예측의 정확도를 개선시킨다. 예를 들어, 도 7 에 대하여 전술된 바와 같이, HEVC 는 방향성 예측 모드들 (즉, 평면 및 DC 예측자들을 배제하는 인트라 예측 모드들) 에서 인트라 예측 블록을 생성하기 위해 2-탭 선형 보간 필터를 사용한다. 특히, 도 7 의 예에서, 비디오 코더는 샘플들 (L 및 R) 에 2-탭 필터를 적용하여 샘플 (50) 에 대한

예측 값을 결정한다. 필터를 2 개의 레퍼런스 샘플들이 적용하여 예측 블록의 샘플에 대한 예측 값을 결정하는, HEVC 의 접근과 대조적으로, VCEG-AZ07 은 4 개의 레퍼런스 샘플들에 필터를 적용하여 예측 블록의 샘플에 대한 예측 값을 결정한다. VCEG-AZ07 에서, 4-탭 보간 필터들의 2 개의 유형들이 사용된다: 4×4 및 8×8 블록들에 대해 큐빅 보간 필터들, 및 16×16 및 더 큰 블록들에 대해 가우시안 보간 필터들. VCEG-AZ07 에서, 필터들의 파라미터들은 블록 사이즈에 따라 고정되고, 동일한 필터가 모든 예측된 픽셀들에 대해, 모든 방향성 모드들에서 사용된다.

[0121] HEVC 에서, 인트라 예측 블록이 수직 및 수평 인트라 모드들에 대해 생성된 후에, 최-좌측 컬럼 및 최-상단 로우의 예측 샘플들 (즉, 예측 블록의 샘플들) 이 각각 추가로 조정된다. 최대 4 개의 컬럼들 또는 로우들의 경계 샘플들은 (인트라 모드들 (2 및 34) 에 대해) 2-탭 또는 (인트라 모드들 (3-6 및 30-33) 에 대해) 3-탭 필터를 사용하여 추가로 조정된다.

[0122] 도 12a 및 도 12b 는 인트라 모드들 (30-34) 에 대한 경계 필터들의 개념도이다. 특히, 도 12a 는 인트라 모드 (34) 에 대한 경계 필터들의 개념도이다. 도 12b 는 인트라 모드 (30-33) 에 대한 경계 필터들의 개념도이다. 도 12a 및 도 12b 에서, 최 좌측 컬럼의 블록들은 레퍼런스 샘플들의 세트이고, 블록들의 나머지는 인트라 예측 블록의 샘플들이다. 비디오 코더는 종래의 방식으로 인트라 예측 블록의 샘플들을 생성할 수도 있다. 그러나, 인트라 예측 모드들 (30-34) 에 대해, 비디오 인코더는 하나 이상의 추가적인 필터들을 음영진 픽셀들에 적용할 수도 있다. 따라서, 인트라 모드 (34 및 30-33) 에 대한 경계 예측 필터들의 예들은 도 12a 및 도 12b 에 도시되고, 인트라 모드들 (2 및 3-6) 에 대한 경계 예측 필터들은 유사하다.

[0123] 특히, 도 12a 에서, 인트라 모드 (34) 에 대해, 비디오 코더는 개별의 샘플의 상부 및 우측의 레퍼런스 샘플들에 기초하여 인트라 예측 블록의 각각의 개별의 샘플을 생성한다. 그러나, 이것은 좌측 레퍼런스 샘플들로부터 이용 가능한 정보를 무시할 수도 있다. 따라서, 비디오 코더는 4 개의 상이한 필터들을 4 개의 최좌측 컬럼들에 적용할 수도 있다. 인트라 예측 블록의 라인들 (1-4) 에서의 각각의 개별의 샘플에 대해, 비디오 코더는 인트라 모드 (34) 의 반대 방향 (즉, 좌측 및 하방) 에서 레퍼런스 샘플 및 개별의 샘플에 기초하여 필터를 적용한다. 라인 1 에 대해, 결과의 샘플은 (8*a + 8*b) / 16 로서 계산될 수도 있고, 여기서 a 는 개별의 샘플이고 b 는 레퍼런스 샘플이다. 라인 2 에 대해, 결과의 샘플은 (12*a + 4*b) / 16 로서 계산될 수도 있고, 여기서 a 는 개별의 샘플이고 b 는 레퍼런스 샘플이다. 도 12b 의 예에서, 인트라 모드들 (30-33) 에 대한 방향들은 풀 정수 포지션 픽셀들과 함께 정렬되지 않는다. 차라리, 예측 블록의 각각의 샘플에 대해, 인트라 예측 모드들 (30-33) 에 대한 방향들은 레퍼런스 샘플들 중 2 개 사이의 부분 포지션들에서 레퍼런스 샘플들과 교차한다. 따라서, 인트라 예측 모드들 (30-33) 에 대해 경계 필터를 적용하는 경우, 예측 블록의 최 좌측 컬럼의 각각의 샘플에 대해 2 개의 레퍼런스 샘플들이 존재한다. 도 12b 의 예에서, 예측 블록의 최 좌측 컬럼의 각각의 개별의 샘플에 대해, 인트라 예측 모드가 33 인 경우, 비디오 코더는 개별의 샘플의 값을 (8*a + 8*b + 2*c) 로서 계산할 수도 있고, 여기서 a 는 개별의 샘플이고, b 는 레퍼런스 샘플들 중 하나이며, c 는 레퍼런스 샘플들 중 다른 하나이다.

[0124] 비디오 코딩은 컬러 공간 및 컬러 포맷에 기초하여 수행될 수도 있다. 예를 들어, 컬러 비디오는, 다양한 컬러 공간들이 사용되어 컬러를 효율적으로 표현하는 멀티미디어 시스템들에서 필수적인 역할을 한다. 컬러 공간은 다수의 컴포넌트들을 사용하여 수치로 컬러를 지정한다. 대중적인 컬러 공간은 RGB 컬러 공간이고, 여기서 컬러는 3 개의 프라이머리 컬러 컴포넌트 값들 (즉, 레드, 그린 및 블루) 의 조합으로서 표현된다. 1998년 8월, 런던, 웨스트민스터 대학, 공대, Ford 및 A. Roberts 의, "Colour space conversions" 에서 설명된 바와 같이, 컬러 비디오 압축을 위해, YCbCr 컬러 공간이 광범위하게 사용되고 있다.

[0125] YCbCr 은 선형 변환을 통해 RGB 컬러 공간으로부터 쉽게 컨버팅될 수 있고, 상이한 컴포넌트들 간의 리던던시, 즉 크로스-컴포넌트 리던던시는 YCbCr 컬러 공간에서 상당히 감소된다. YCbCr 의 하나의 이점은, Y 신호가 루미넌스 정보를 전달하기 때문에 블랙 및 화이트 TV 와의 역방향 호환성 (backward compatibility) 이다. 또한, 크로미넌스 대역폭은 RGB 에서의 서브-샘플링보다 주관적 영향이 상당히 적은 4:2:0 크로마 샘플링 포맷에서 Cb 및 Cr 컴포넌트들을 서브-샘플링함으로써 감소될 수 있다. 이들 이점들 때문에, YCbCr 은 비디오 압축에서 주요한 컬러 공간이었다. 또한, 비디오 압축에서 사용된 YCoCg 와 같은 다른 컬러 공간들이 존재한다. 본 개시물에서, 사용된 실제 컬러 공간에 관계 없이, YCbCr 컬러 공간이 사용되어 비디오 압축 스킴에서 3 개의 컬러 컴포넌트들을 표현한다.

[0126] 크로스-보완 리던던시가 YCbCr 컬러 공간에서 상당히 감소되지만, 3-컬러 컴포넌트들 간의 상관은 여전히 존재한다. 3 개의 컬러 컴포넌트들 간의 상관을 더 감소시킴으로써 비디오 코딩 성능을 개선시키기 위한 다양한

기법들이 연구되어 왔다.

[0127] 예를 들어, 2012년 5월 7-9일, 크라쿠프, 폴란드, 2012 PCS (Picture Coding Symposium), pp. 501-504, Xiaoran Cao 등의, "Short distance intra coding scheme for HEVC" (이하에서, "Cao 2") 은 단거리 인트라 코딩 스킴을 설명한다. Cao 2 는, HEVC 표준의 개발 동안 연구되었던, 선형 모델 (LM) 예측 모드로 명명된 4:2:0 크로마 비디오 코딩에서의 방법을 설명한다. 예를 들어, 이하에서 JCTVC-E266 로서 지칭된, 5 차 회의: 제네바, 2011년 3월 16-23일, ITU-T SG16 WP3 및 ISO/IEC JTC1/SC29/WG11 의 JCT-VC (Joint Collaborative Team on Video Coding), JCTVC-E266, J. Chen 등의, "CE6.a.4: Chroma intra prediction by reconstructed luma samples" 을 참조한다. 4:2:0 샘플링에서, 2 개의 크로마 어레이들 각각은 루마 어레이의 높이의 절반 및 폭의 절반을 갖는다. LM 예측 모드로, 크로마 샘플들은 다음과 같이 선형 모델을 사용함으로써 동일한 블록의 복원된 루마 샘플들에 기초하여 예측된다:

[0128] $$pred_C(i,j) = \alpha * rec_L(i,j) + \beta \quad (2)$$

[0129] 여기서, $pred_C(i,j)$ 는 현재 블록에서 크로마 샘플들의 예측을 나타내고, $rec_L(i,j)$ 는 현재 블록의 다운-샘플링된 복원된 루마 샘플들을 나타낸다. 파라미터들 (α 및 β) 은 현재 블록 주변의 인과관계 (causal) 복원된 샘플들로부터 도출된다. 블록의 인과관계 샘플들은 디코딩 순서에서 블록 이전에 발생하는 샘플들이다. 크로마 블록 사이즈가 N×N 으로 표기되면, i 및 j 양자 모두는 범위 [0, N) 내에 있다.

[0130] 등식 (2) 에서 파라미터들 (α 및 β) 은 현재 블록 주변의 이웃하는 복원된 루마 및 크로마 샘플들 간의 회귀 오차 (regression error) 를 최소화시킴으로써 도출된다.

[0131] $$E(\alpha,\beta) = \sum_i (y_i - (\alpha \cdot x_i + \beta))^2 \quad (3)$$

[0132] 파라미터들 (α 및 β) 은 다음과 같이 풀린다.

[0133] $$\alpha = \frac{I\sum x_i \cdot y_i - \sum x_i \cdot \sum y_i}{I\sum x_i \cdot x_i - \sum x_i \cdot \sum x_i} \quad (4)$$

[0134] $$\beta = \left(\sum y_i - \alpha \cdot \sum x_i\right) / I \quad (5)$$

[0135] 상기의 등식에서, x_i 는 컬러 포맷이 4:4:4 가 아닌 다운-샘플링된 복원된 루마 레퍼런스 샘플 (즉, 컬러 포맷이, 하나의 크로마 샘플이 다수의 루마 샘플들에 대응하는 포맷임) 이고, y_i 는 다운-샘플링 없이 복원된 크로마 레퍼런스 샘플들이며, I 는 레퍼런스 샘플들의 수이다. 다시 말해, 비디오 코더는 4:4:4 가 아닌 컬러 포맷에 기초하여 복원된 루마 레퍼런스 샘플들을 다운 샘플링 할 수도 있지만, 4:4:4 인 컬러 포맷에 기초하여 복원된 루마 레퍼런스 샘플들을 다운 샘플링하지 않는다. 타겟 $N \times N$ 크로마 블록에 대해, 좌측 및 상부의 인과관계 샘플들 양자 모두가 이용 가능한 경우, 수반된 샘플들의 총 수 (I) 는 2N 이다. 단지 좌측 또는 상부의 인과관계 샘플들이 이용 가능한 경우, 수반된 샘플들의 총 수 (I) 는 N 과 동일하다. 여기서, N 은 항상 2^m 과 동일한 (여기서, m 은 상이한 CU 사이즈들에 대해 상이할 수도 있다). 따라서, 복잡성을 감소시키기 위해, 시프팅 연산들은 등식들 (4) 및 (5) 에서 나누기 연산들을 구현하도록 사용될 수 있다.

[0136] 도 13 은 α 및 β 의 도출을 위해 사용된 샘플들의 예시의 로케이션들을 예시하는 개념도이다. 특히, 도 13 은 동일한 PU 의 루마 예측 블록 (92) 및 PU 의 크로마 예측 블록 (90) 을 예시한다. 크로마 샘플들이 루마 샘플들에 대해 다운-샘플링되기 때문에, 크로마 예측 블록 (90) 의 폭 및 높이 (즉, N) 는 루마 예측 블록 (92) 의 폭 및 높이 (즉, 2N) 의 절반이다. 도 13 의 예에서, 큰 다크 스퀘어 밖의 작은 스퀘어들은 레퍼런스 샘플들이다. 도 13 의 예에서, 작은 원들은 LM 파라미터들 (α 및 β) 을 결정하기 위해 사용된 샘플 값들을 나타낸다. 도 13 의 예에 도시된 바와 같이, LM 파라미터들 (α 및 β) 을 결정하기 위해 사용된 크로마 샘플 값들은 크로마 예측 블록 (90) 에 대한 레퍼런스 샘플들과 동일하다. 그러나, LM 파라미터들 (α 및 β) 을 결정하기 위해 사용된 루마 샘플 값들은 루마 레퍼런스 샘플들로부터 보간된다. LM 파라미터들 (α 및 β) 을 결정하기 위해 사용된 루마 샘플들의 결과의 세트의 총 수는 LM 파라미터들 (α 및 β) 을 결정하기 위해 사용된 크로마 샘플들의 수와 동일하다.

[0137] 일반적으로, LM 예측 모드가 현재의 PU 에 대해 적용되는 경우, 비디오 코더는 다음의 단계들을 수행할 수도 있다. 먼저, 비디오 코더는 현재 PU 에 대한 루마 블록을 복원할 수도 있다. 현재 PU 에 대한 루마 블록을 복원하는 것의 부분으로서, 비디오 코더는 인트라 예측을 수행하여 현재 PU 의 루마 예측 블록을 결정할 수도 있다. 또한, 현재 PU 에 대한 루마 블록을 복원하는 것의 부분으로서, 비디오 코더는 잔차 데이터를 현재 PU 의 루마 예측 블록에 추가하여 현재 PU 에 대한 루마 블록을 복원할 수도 있다. 두 번째, 비디오 코더는 현재 PU 의 상단 및 좌측 사이드들을 이웃하는 레퍼런스 루마 샘플들을 다운-샘플링할 수도 있다. 세 번째, 비디오 코더는 상기의 등식들 (4) 및 (5) 를 사용하여, 다운-샘플링된 루마 레퍼런스 샘플들 및 현재 PU 의 상단 및 좌측 사이드들을 이웃하는 크로마 레퍼런스 샘플들에 기초하여 선형 파라미터들 (즉, α 및 β) 를 도출할 수도 있다. 본 개시물은 또한, 선형 파라미터를 "스케일링 팩터들" 로서 지칭할 수도 있다. 네 번째, 비디오 코더는 현재 PU 에 대한 복원된 루마 블록을 다운-샘플링할 수도 있다. 다섯 번째, 비디오 코더는 상기의 등식 (2) 를 사용하여, 선형 파라미터들 및 현재 PU 에 대한 다운-샘플링된 루마 블록으로부터 크로마 샘플들을 예측 (예를 들어, 예측 크로마 블록을 도출) 할 수도 있다.

[0138] 상기에서 주목된 바와 같이, 비디오 코더는 현재 PU 의 복원된 루마 블록을 다운-샘플링할 수도 있다. 비디오 코더는 현재 PU 의 복원된 루마 블록을 다양한 방식들로 다운-샘플링할 수도 있다. 예를 들어, 크로마 컴포넌트들의 통상적인 샘플링 비가 루마 컴포넌트들의 것의 절반이고 4:2:0 샘플링에서 수직 방향에서 0.5 샘플 위상 차이를 갖기 때문에, 현재 PU 의 복원된 루마 샘플들은 수직 방향에서 다운-샘플링되고 수평 방향에서 서브-샘플링되어 크로마 신호의 사이즈 및 위상에 일치한다. 예를 들어, 0 에서부터 현재 PU 의 예측 크로마 블록의 폭 마이너스 1 까지의 각각의 값 (i) 및 0 에서부터 현재 PU 의 예측 크로마 블록의 높이 마이너스 1 까지의 각각의 값 (j) 에 대해, 비디오 코더는 다음과 같이 계산할 수도 있다:

[0139] $$rec_L(i,j) = (\mathrm{Rec}_{LOrig}[\,2i, 2j\,] + \mathrm{Rec}_{LOrig}[\,2i, 2j+1\,]) >> 1 \quad (6)$$

[0140] 상기의 등식에서, $rec_L(i,j)$ 는 현재 PU 의 다운-샘플링된 복원된 루마 블록의 상단-좌측 코너에 대한 포지션 (i,j) 에 대응하는 루마 샘플이다. $\mathrm{Rec}_{LOrig}[2i, 2j]$ 및 $\mathrm{Rec}_{LOrig}[2i, 2j+1]$ 은 현재 PU 의 원래의 복원된 루마 블록의 상단-좌측 코너에 대한 포지션들 ($2i, 2j$) 및 ($2i, 2j+1$) 에서 복원된 루마 샘플들이다. 따라서, 등식 (6) 에서, 현재 PU 의 다운-샘플링된 복원된 루마 블록의 포지션 (i,j) 에서 루마 샘플은 현재 PU 의 원래의 복원된 루마 블록의 포지션 ($2i, 2j$) 에서의 루마 샘플 및 현재 PU 의 원래의 복원된 루마 블록의 포지션 ($2i, 2j + 1$) 에서의 루마 샘플의 평균이다.

[0141] 도 14 는 현재 PU 의 복원된 루마 블록의 다운-샘플링 샘플들에 대한 루마 포지션들 및 크로마 포지션들의 예를 예시하는 개념도이다. 도 14 는 삼각형들로서 크로마 샘플들을 그리고 원들로서 루마 샘플들을 도시한다.

비디오 코더는, [1,1] 필터를 적용함으로써, (2 개의 채워진 원들로 도 14 에서 표현된) 2 개의 루마 샘플들로부터 (채워진 삼각형으로 도 14 에서 표현된) 현재 크로마 샘플의 값을 예측한다. [1, 1] 필터는 2-탭 필터의 일 예이다. [1, 1] 필터에서, 2 개의 탭들이 동일하게 가중화된다. 도 14 에서 각각의 개별의 삼각형에 대해, 비디오 코더는 상부의 원들 및 이하의 개별의 삼각형으로 표현된 샘플들에 등식 (6) 을 적용하여, 개별의 삼각형으로 표현된 샘플에 대한 개별의 루마 값을 결정할 수도 있다.

[0142] 또한, 상기에서 주목된 바와 같이, 비디오 코더는 루마 레퍼런스 샘플들을 다운-샘플링할 수도 있다. 비디오 코더는 루마 레퍼런스 샘플들을 다양한 방식들로 다운-샘플링할 수도 있다. 도 14 에 도시된 바와 같이, 현재 PU 의 예측 크로마 블록의 컬럼들은 현재 PU 의 예측 루마 블록의 컬럼들과 정렬된다. 4:2:0 컬러 포맷을 사용하는 일 예에서, 현재 루마 블록의 상단 사이드를 이웃하는 다운-샘플링된 루마 레퍼런스 샘플들은 루마 레퍼런스 샘플들의 세트에서 짝수 인덱싱된 포지션에서의 각각의 루마 레퍼런스 샘플로 이루어질 수도 있다.

따라서, 0 에서부터 현재 PU 의 예측 크로마 블록의 폭 마이너스 1 까지의 범위의 i 의 각각의 개별의 값에 대해, 다운-샘플링 프로세스는 다음과 같이 정의될 수도 있다:

[0143] $$rec_L(i,-1) = \mathrm{Rec}_{LOrig}[2i,-1] \quad (7)$$

[0144] 상기의 등식에서, $rec_L(i,-1)$ 은 현재 PU 의 크로마 예측 블록의 상단-좌측 코너에 대한 포지션 ($i, -1$) 에서의 다운-샘플링된 루마 레퍼런스 샘플이다. $\mathrm{Rec}_{LOrig}[2i, -1]$ 은 현재 PU 의 원래의 예측 루마 블록의 상단-좌측 코너에 대한 포지션 ($2i, -1$) 에서의 루마 레퍼런스 샘플이다.

[0145] 도 14 에 도시된 바와 같이, 현재 PU 의 예측 크로마 블록의 로우들은 4:2:0 컬러 포맷에서 현재 PU 의 예측 루

마 블록의 로우들과 정렬되지 않는다. 그러나, LM-기반 예측에 대한 파라미터들 (α 및 β) 을 계산하기 위한 등식들 (4) 및 (5) 는 각각의 크로마 레퍼런스 샘플에 대해 하나의 루마 레퍼런스 샘플이 존재한다는 것에 입각한다. 따라서, 현재 PU 에 대한 예측 크로마 블록의 개별의 로우에 대해, 비디오 코더는 예측 크로마 블록의 로우 아래의 현재 PU 의 예측 루마 블록의 로우에서의 루마 레퍼런스 샘플 및 위의 현재 PU 의 예측 루마 블록의 로우에서의 루마 레퍼런스 샘플의 평균을 계산할 수도 있다. 예를 들어, 0 에서부터 예측 크로마 블록에서 로우들의 수 마이너스 1 까지의 범위에 있는 각각의 값 (j) 에 대해, 비디오 코더는 다음과 같은 좌측-이웃하는 루마 레퍼런스 샘플의 값을 계산할 수도 있다:

[0146] $$rec_L(-1, j) = (\mathrm{Rec}_{LOrig}[-2, 2j] + \mathrm{Rec}_{LOrig}[-2, 2j+1]) >> 1 \quad (8)$$

[0147] 상기의 등식에서, $rec_L(-1, j)$ 은 현재 PU 의 예측 크로마 블록의 상단-좌측 코너에 대한 포지션 (-1, j) 에서의 다운-샘플링된 루마 레퍼런스 샘플이다. $\mathrm{Rec}_{LOrig}[-2, 2j]$ 및 $\mathrm{Rec}_{LOrig}[-2, 2j+1]$ 은 현재 PU 의 원래의 예측 루마 블록의 상단-좌측 코너에 대한 포지션들 (-2, $2j$) 및 (-2, $2j$+1) 에서의 원래의 루마 샘플들이다.

[0148] 다른 다운-샘플링 기법들이 또한, 제안되어 있다. 예를 들어, 6 차 회의: 토리노, 이탈리아, 2011년 7월 14-22일, ITU-T SG16 WP3 및 ISO/IEC JTC1/SC29/WG11 의 JCT-VC (Joint Collaborative Team on Video Coding), JCTVC-F502, Yi-Jen Chiu 등의, "Cross-channel techniques to improve intra chroma prediction" (본원에서 "JCTVC-F502" 로서 지칭됨) 에서, 2-탭 필터를 사용하는 대신에, 비디오 코더는 현재 루마 블록 및 이웃하는 루마 블록 양자 모두에 2-차원 6-탭 필터링을 적용한다. 2-차원 필터 계수 세트는 다음과 같다:

[0149] $$\begin{bmatrix} 1 & 2 & 1 \\ 1 & 2 & 1 \end{bmatrix} / 8 \quad (9)$$

[0150] 다운-샘플링된 루마 샘플들은 등식 (10) 에 의해 도출된다:

[0151] $$\begin{aligned} \mathrm{rec}_L(i,j) = (&\mathrm{Rec}_{LOrig}[2i, 2j]*2 + \mathrm{Rec}_{LOrig}[2i, 2j+1] + \mathrm{Rec}_{LOrig}[2i, 2j-1] \\ &+ \mathrm{Rec}_{LOrig}[2i+1, 2j]*2 + \mathrm{Rec}_{LOrig}[2i+1, 2j+1] + \mathrm{Rec}_{LOrig}[2i+1, 2j-1]) >> 3 \end{aligned} \quad (10)$$

[0152] 상기의 등식에서, $rec_L(i, j)$ 는 현재 PU 다운-샘플링된 복원된 루마 블록의 상단-좌측 코너에 대한 포지션 (i, j) 에서의 복원된 루마 샘플이고, $\mathrm{Rec}_{LOrig}[..]$ 는 현재 PU 의 원래의 복원된 루마 블록의 상단-좌측 코너에 대한 포지션들에서의 현재 PU 의 원래의 복원된 루마 블록의 복원된 루마 샘플들이다.

[0153] 예를 들어, 비디오 코더는 등식 (10) 의 연산들을 수행하여 다운-샘플링된 루마 블록을 결정할 수도 있다. 등식 (10) 은 6 개의 입력 샘플들로서 $\mathrm{Rec}_{LOrig}[2i, 2j]$, $\mathrm{Rec}_{LOrig}[2i, 2j+1]$, $\mathrm{Rec}_{LOrig}[2i, 2j-1]$, $\mathrm{Rec}_{LOrig}[2i+1, 2j]$, $\mathrm{Rec}_{LOrig}[2i+1, 2j+1]$, 및 $\mathrm{Rec}_{LOrig}[2i+1, 2j-1]$ 으로 [1, 2, 1; 1, 2, 1] 에 의해 표현된 바와 같이, 빌트인 6-탭 필터를 포함한다. 필터의 탭 수는, 얼마나 많은 입력 샘플들이 필터를 적용하기 위해 사용되는지를 나타낸다. 예를 들어, 등식 (10) 에서, 비디오 코더는 복원된 루마 블록으로부터의 6 개의 값들을 사용하여 다운-샘플링된 루마 블록을 생성한다.

[0154] 도 15 는 예측 블록을 생성하기 위해 루마 블록의 다운-샘플링 샘플들에 대한 루마 포지션들 및 크로마 포지션들의 예를 예시하는 개념도이다. 도 15 에 도시된 바와 같이, 비디오 코더는 6-탭 필터를 적용함으로써, 6 개의 채워진 원들로 표현된, 6 개의 루마 샘플들로부터, 채워진 삼각형으로 표현된, 크로마 샘플을 예측한다.

크로마 샘플의 예측자가 등식 (2) 에 정의된 바와 같이 선형 함수를 사용하여 도출되기 때문에, 6-탭 필터가 적용되는 경우, 하나의 크로마 샘플의 예측자는 6 개의 이웃하는 루마 샘플들에 의존한다는 것을 알 수 있다.

등식들 (2) 및 (10) 을 결합할 때, 결과는 다음의 등식 (11) 과 같다:

[0155] $$\begin{aligned} pred_C(i,j) = \alpha \cdot (&\mathrm{Rec}_{LOrig}[2i, 2j]*2 + \mathrm{Rec}_{LOrig}[2i, 2j+1] + \mathrm{Rec}_{LOrig}[2i, 2j-1] \\ &+ \mathrm{Rec}_{LOrig}[2i+1, 2j]*2 + \mathrm{Rec}_{LOrig}[2i+1, 2j+1] + \mathrm{Rec}_{LOrig}[2i+1, 2j-1]) >> 3) \\ &+ \beta \end{aligned} \quad (11)$$

[0156] 다음의 텍스트는 다운-샘플링된 복원된 루마 샘플 $rec_L(i, j)$ 을 (i,j) 에 위치된 크로마 샘플에 대한 대응하는 다운-샘플링된 루마 샘플로서 지칭한다. 예를 들어, 4:2:0 샘플링 때문에, 2N×2N 루마 블록은 N×N 크로마

블록에 대응한다. 다운-샘플링으로, 2N×2N 루마 블록들은 N×N 다운-샘플링된 루마 블록이 된다. 이 N×N 다운-샘플링된 루마 블록은 $rec_L(i, j)$ 로서 지칭되고, N×N 크로마 블록에 대응한다.

[0157] 또한, 상기의 예들이 4:2:0 샘플링에 대하여 설명되지만, 본 개시물에 설명된 기법들은 이에 제한되지 않는다. 예를 들어, 본 개시물에 설명된 기법들은 또한, 4:2:2 샘플링에 적용 가능할 수도 있다. 따라서, 4:2:0 에 대한 예들은 단지 이해를 돕기 위해 제공된다.

[0158] 또한, 일부 예들에서, 본 개시물에 설명된 기법들은 4:4:4 샘플링에 또한 적용 가능할 수도 있다. 예를 들어, 4:4:4 샘플링에서, 크로마 블록은 루마 블록에 대해 서브-샘플링되지 않는다. 그러나, 이러한 예들에서 또한 크로마 블록에 대한 예측 블록을 결정하는 것이 가능할 수도 있다. 예를 들어, 루마 블록은 필터링될 수도 있고, 필터링된 블록은 크로마 블록에 대한 예측 블록으로서 사용될 수도 있다. 이들 예들에서, 루마 블록의 다운-샘플링은 필요하지 않을 수도 있다. 더 상세히 설명되는 바와 같이, 예시의 기법들은 크로마 블록의 로케이션에 기초하여 루마 블록의 샘플들에 적용된 필터의 선택을 설명한다. 루마 블록의 샘플들에 적용된 필터를 선택하기 위한 기법들은 4:4:4 샘플링에 대한 것과 같이, 다운-샘플링이 LM 예측을 위해 필요하지 않는 예들로 확장될 수도 있다. 이러한 예들에서, 필터는 4:4:4 샘플링이 보존되도록 임의의 다운-샘플링을 포함하지 않을 수도 있다. 따라서, 4:2:0 샘플링에 대한 설명은 일 예이고, 이 기법들은 4:4:4 샘플링에 또한 적용 가능하다.

[0159] 예를 들어, 루마 블록을 다운-샘플링하기 위해 단지 2-탭 필터 또는 6-탭 필터를 사용하는 것에 제한되기 보다는 차라리, 비디오 코더 (예를 들어, 비디오 인코더 (20) 또는 비디오 디코더 (30)) 는 루마 블록을 다운-샘플링하기 위해 사용되는 필터들의 세트로부터 필터를 결정할 수도 있다. 일 예로서, 비디오 코더가 다운-샘플링을 위해 사용할 수 있는 다수 (X) 의 상이한 필터들이 존재할 수도 있다. 예를 들어, 1-탭 필터, 2-탭 필터, 3-탭 필터, 등이 존재할 수도 있다. 또한, 각각의 필터에 대해 특정 탭들이 상이할 수도 있다 (예를 들어, 제 1 의 2-탭 필터에 대해 사용된 루마 샘플들은 제 2 의 2-탭 필터에 대해 사용된 루마 샘플들과 상이하다). 본 개시물에 설명된 예들의 일부에서, 필터들의 세트는 2 개의 필터들을 포함한다; 그러나, 2 개 보다 많은 필터들로부터 비디오 코더가 루마 블록을 다운-샘플링하기 위해 어느 필터를 적용할지를 결정하는 것이 가능하다.

[0160] 비디오 코더는 어느 필터를 적용할지를 결정하기 위해 다양한 기준을 사용할 수도 있다. 일 예로서, 비디오 코더는 크로마 블록의 로케이션에 기초하여 필터들의 세트로부터 어느 필터를 적용할지를 결정한다. 크로마 블록이 픽처의 좌측 경계, CU, PU, 또는 TU 를 접하면 (예를 들어, 픽처의 좌측 경계, CU, PU, 또는 TU 가 크로마 블록 에지와 동일하면), 비디오 코더는 좌측 경계 상에 있는 크로마 블록의 크로마 샘플들에 대응하는 루마 블록의 루마 샘플들을 다운-샘플링하기 위해 제 1 필터를 사용할 수도 있다. 좌측 경계 상에 있는 크로마 블록의 샘플들은 경계 상에 직접적으로 있는 샘플들을 포함하는 좌측 경계에 가장 가까운 크로마 블록의 샘플들을 지칭한다. 제 1 필터는 경계에 가장 가까운 N 개의 샘플들 (예를 들어, 경계에 가장 가까운 샘플, 그 샘플 다음의 샘플, 및 N 개의 이러한 샘플들) 에 적용될 수도 있다.

[0161] 일부 경우들에서, 비디오 코더는 좌측 경계를 이웃하는 크로마 샘플들에 대응하는 단지 이들 샘플들보다는 차라리, 루마 블록의 모든 루마 샘플들에 대해 제 1 필터를 적용할 수도 있다. 그러나, 본 개시물에 설명된 기법들은 그렇게 제한되지 않는다. 모든 다른 경우들에 대해, 비디오 코더는 루마 블록을 다운-샘플링을 위해 제 2 의, 상이한 필터를 사용할 수도 있다.

[0162] 예를 들어, 4:2:0 샘플링에서, 4 개의 루마 샘플들은 하나의 크로마 샘플에 대응한다. 따라서, 비디오 코더는, 어느 크로마 샘플이 어느 루마 샘플들에 대응하는지를 결정할 수도 있다. 더 큰 탭 수들을 갖는 필터들이 사용되는 경우, 하나의 크로마 샘플은 4 개 보다 많은 루마 샘플들에 대응할 수도 있다. (다수의 샘플들에 바로 인접한 또는 이들 내의) 좌측 경계 상의 크로마 샘플에 대응하는 루마 샘플들에 대해, 비디오 코더는 대응하는 루마 샘플들에 제 1 필터를 적용하여 루마 블록을 다운-샘플링할 수도 있고, (다수의 샘플들에 바로 인접하지 않은 또는 이들 내에 있지 않은) 좌측 경계 상에 있지 않은 크로마 샘플에 대응하는 루마 샘플들에 대해, 비디오 코더는 대응하는 루마 샘플들에 제 2 필터를 적용하여 루마 블록을 다운-샘플링할 수도 있다.

[0163] 일부 예들에서, 제 1 필터는 제 2 필터보다 더 적은 탭들 (예를 들어, 필터가 걸쳐 있는 샘플들의 수) 을 포함할 수도 있다. 일 예로서, 제 1 필터는 2-탭 필터이고 제 2 필터는 6-탭 필터이다. 이 예에서, 비디오 코더는, 크로마 블록의 대응하는 크로마 샘플들이 좌측 경계 상에 있는 경우에서 루마 블록의 다운-샘플링된 루마 샘플들을 결정하도록 등식 (6) 의 연산들을 수행할 수도 있고, 크로마 블록의 대응하는 크로마 샘플들이 좌측 경계 상에 있지 않은 경우에서 루마 블록의 다운-샘플링된 루마 샘플들을 결정하도록 등식 (10) 의 연산들을

수행할 수도 있다. 따라서, 크로마 샘플들의 대응하는 다운-샘플링된 루마 샘플들의 도출 프로세스 동안, 비디오 코더는 CU, PU, 또는 TU 의 좌측 경계 또는 좌측 픽처 경계에 있지 않는 크로마 샘플들에 대응하는 루마 블록의 다른 샘플들에 적용된 필터와 비교하여, CU/PU/TU 의 좌측 픽처 경계, 또는 좌측 경계 (즉, 사이드) 에 위치된 크로마 블록의 크로마 샘플들에 대응하는 루마 블록의 루마 샘플들에 상이한 필터를 적용할 수도 있다.

좌측 경계에 있는 크로마 샘플들은 좌측 경계로부터 소정 수의 샘플들 내의 또는 좌측 경계에 바로 인접한 크로마 샘플들을 지칭한다.

[0164] 상이한 필터들을 사용하는 것은 비디오 코더가, 이용 가능한 샘플 값들을 적절히 사용하도록 허용한다. 예를 들어, 픽처의 좌측 경계에서 크로마 샘플에 대응하는 루마 샘플들에 대해 6-탭 필터를 사용하면, CU, PU, 또는 TU 는 다운-샘플링에 대한 루마 블록의 부분이 아닌 루마 샘플 값들을 사용하도록 비디오 코더에 요구할 수도 있고, 비디오 코더가 루마 샘플들의 부족을 어드레싱하기 위해 일부 추가적인 프로세싱 (예를 들어, 루마 블록의 부분이 아닌 샘플들에 대한 값들을 생성하기 위해 루마 샘플 값들을 패딩) 을 수행해야 할 수도 있다. 그러나, 좌측 경계에서 2-탭 필터를 사용하는 것은 다운-샘플링에 대한 루마 블록의 부분이 아닌 루마 샘플 값들을 비디오 코더가 사용할 것을 요구하지 않을 수도 있다. 따라서, 2-탭 및 6-탭 필터들이 설명되지만, 루마 블록의 부분이 아닌 루마 샘플들을 요구할 필요를 회피하기 위해 (예를 들어, 좌측 경계 상의 루마 샘플들을 패딩할 필요성을 회피하기 위해) 고려하여 다운-샘플링을 위한 다른 사이즈의 필터들이 가능할 수도 있다.

[0165] 일 예로서, 크로마 샘플들의 대응하는 다운-샘플링된 루마 샘플들의 도출 프로세스 동안, 비디오 코더는 좌측 픽처 경계에 위치되지 않는 크로마 샘플들에 대응하는 다른 루마 샘플들에 적용된 필터와 비교하여 좌측 픽처 경계에 위치된 크로마 샘플들에 대응하는 루마 샘플들에 상이한 필터를 적용한다. 일 예에서, 좌측 픽처 경계에서 크로마 샘플들의 대응하는 다운-샘플링된 루마 샘플들을 도출하기 위한 필터의 길이 (예를 들어, 탭)(즉, 필터가 걸쳐 있는 샘플들의 수) 는 좌측 픽처 경계에 있지 않은 크로마 샘플들의 대응하는 다운-샘플링된 루마 샘플들을 도출하기 위한 필터의 길이 (예를 들어, 좌측 경계에 대해 2-탭 및 모든 다른 것들에 대해 6-탭) 보다 더 작다.

[0166] 일 예로서, 크로마 샘플들의 대응하는 다운-샘플링된 루마 샘플들의 도출 프로세스 동안, 비디오 코더는 현재 CU 내의 다른 루마 샘플들에 적용된 필터와 비교하여 좌측 CU 경계에 위치된 크로마 샘플들의 루마 샘플들에 대한 상이한 필터를 적용한다. 일 예에서, 좌측 CU 경계에서 크로마 샘플들의 대응하는 다운-샘플링된 루마 샘플들을 도출하기 위한 필터의 길이 (예를 들어, 탭들)(즉, 필터가 걸쳐 있는 샘플들의 수) 는 좌측 CU 경계에 있지 않은 크로마 샘플들의 대응하는 다운-샘플링된 루마 샘플들을 도출하기 위한 필터의 길이 (예를 들어, 좌측 경계에 대해 2-탭 및 모든 다른 것들에 대해 6-탭) 보다 더 작다.

[0167] 일 예로서, 크로마 샘플들의 대응하는 다운-샘플링된 루마 샘플들의 도출 프로세스 동안, 비디오 코더는 현재 CU 내의 다른 샘플들에 적용된 필터와 비교하여 좌측 PU 경계에 위치된 크로마 샘플들에 대해 상이한 필터를 적용한다. 일 예에서, 좌측 PU 경계에서 크로마 샘플들의 대응하는 다운-샘플링된 루마 샘플들을 도출하기 위한 필터의 길이 (예를 들어, 탭들)(즉, 필터가 걸쳐 있는 샘플들의 수) 는 좌측 PU 경계에 있지 않은 크로마 샘플들의 대응하는 다운-샘플링된 루마 샘플들을 도출하기 위한 필터의 길이 (예를 들어, 좌측 경계에 대해 2-탭 및 모든 다른 것들에 대해 6-탭) 보다 더 작다.

[0168] 일 예로서, 크로마 샘플들의 대응하는 다운-샘플링된 루마 샘플들의 도출 프로세스 동안, 비디오 코더는 현재 TU 내의 다른 샘플들에 적용된 필터와 비교하여 좌측 TU 경계에 위치된 크로마 샘플들에 대해 상이한 필터를 적용할 수도 있다. 일 예에서, 좌측 TU 경계에서 크로마 샘플들의 대응하는 다운-샘플링된 루마 샘플들을 도출하기 위한 필터의 길이 (예를 들어, 탭들)(즉, 필터가 걸쳐 있는 샘플들의 수) 는 좌측 TU 경계에 있지 않은 크로마 샘플들의 대응하는 다운-샘플링된 루마 샘플들을 도출하기 위한 필터의 길이 (예를 들어, 좌측 경계에 대해 2-탭 및 모든 다른 것들에 대해 6-탭) 보다 더 작다.

[0169] 일부 경우들에서, 동일한 픽처에서 대응하는 루마 샘플들이 존재하지 않을 수도 있다. 다음은 이러한 상황들을 어드레싱하기 위한 일부 예시의 기법들을 설명한다. 예를 들어, 패딩을 회피하는 것이 일부 경우들에서 유리할 수도 있지만, 일부 경우들에서는 패딩을 회피하는 것이 가능하지 않을 수도 있다. 예를 들어, 일부 루마 샘플들이 (예를 들어, 오프 픽처 때문에) 이용 가능하지 않기 때문에, 비디오 코더는 이들 이용 가능하지 않은 샘플들에 대해 패딩 샘플 값들을 대체할 수도 있고 이들 패딩 샘플 값들로 다운-샘플링을 수행할 수도 있다 (예를 들어, 다운 샘플은 이용 가능한 루마 샘플들에 대해 실제 루마 샘플 값들 및 이용 가능하지 않은 루마 샘플들에 대해 패딩 샘플 값들을 사용). 패딩 샘플 값들은 디폴트 값들 (예를 들어, $2^{bitdepth}$, 여기서

bitdepth 는 루마 컴포넌트의 비트 심도를 나타냄), 비디오 인코더 (20) 에 의해 결정되고 비디오 디코더 (30) 로 시그널링된 값들, 또는 정보의 시그널링을 요구하지 않는 일부 내포적인 기법에 기초하여 결정된 값들일 수도 있다. 패딩 샘플 값들은 추가하는 것은, 별개의 필터들에 대한 필요성이 존재하지 않을 수도 있기 때문에 복잡성을 감소시킬 수도 있다.

[0170] 크로마 샘플들의 대응하는 다운-샘플링된 루마 샘플들의 도출 프로세스 동안, 루마 샘플들이 픽처의 밖에 있고, 또는 CU/PU/TU 가 다운-샘플링 프로세스에 수반될 필요가 있는 경우, 비디오 코더는 먼저 패딩 동작, 다음에 다운-샘플링 프로세스를 적용할 수도 있다. 샘플들의 패딩에서, 비디오 코더는 스크린 밖에 있는 이들 샘플들을 패딩 샘플 값들로 대체할 수도 있다.

[0171] 일 예로서, 크로마 샘플들의 대응하는 다운-샘플링된 루마 샘플들의 도출 프로세스 동안, 비디오 코더는 현재 픽처 밖에 위치되는 루마 샘플들 (예를 들어, 단지 루마 샘플들) 을 패딩할 수도 있다. 모든 다른 포지션들에 대해, 복원된 샘플들이 사용된다. 일 예로서, 크로마 샘플들의 대응하는 다운-샘플링된 루마 샘플들의 도출 프로세스 동안, 비디오 코더는 현재 CU 밖에 위치되는 루마 샘플들을 패딩할 수도 있다. 모든 다른 포지션들에 대해, 복원된 샘플들이 사용된다. 일 예로서, 크로마 샘플들의 대응하는 다운-샘플링된 루마 샘플들의 도출 프로세스 동안, 비디오 코더는 현재 PU 밖에 위치되는 루마 샘플들을 패딩할 수도 있다. 모든 다른 포지션들에 대해, 복원된 샘플들이 사용된다. 일 예로서, 크로마 샘플들의 대응하는 다운-샘플링된 루마 샘플들의 도출 프로세스 동안, 비디오 코더는 현재 TU 밖에 위치되는 루마 샘플들을 패딩할 수도 있다. 모든 다른 포지션들에 대해, 복원된 샘플들이 사용된다. 패딩에 대한 상기의 예들에서, 동일한 다운-샘플링 프로세스가 모든 포지션들에 적용된다.

[0172] LM 예측 모드에서 사용된 루마 복원된 샘플들의 포지션이 현재 슬라이스 또는 현재 타일 밖에 위치되는 경우, 비디오 코더는 이러한 샘플들을 이용 가능하지 않은 것으로서 마킹할 수도 있다 (예를 들어, 비디오 코더는 이러한 샘플들을 이용 가능하지 않은 것으로서 결정할 수도 있다). 샘플이 이용 가능하지 않은 것으로서 마킹되는 경우, 비디오 코더는 다음 중 하나 이상을 수행할 수도 있다.

[0173] 이용 가능하지 않은 샘플들은, 이웃하는 루마 블록에 대한 다운-샘플링 프로세스에서 사용된 경우, 이웃하는 루마 블록에 대한 다운-샘플링 프로세스에서 사용되지 않는다. 대안으로 또는 부가적으로, 필터는 다른 샘플들에 대해 사용된 필터와 상이할 수도 있다. 이용 가능하지 않은 샘플들은, 현재 루마 블록에 대한 다운-샘플링 프로세스에서 사용된 경우, 현재 루마 블록에 대한 다운-샘플링 프로세스에서 사용되지 않는다. 대안으로 또는 부가적으로, 필터는 다른 샘플들에 대해 사용된 필터와 상이할 수도 있다. 이용 가능하지 않은 샘플들은 이용 가능한 것으로 재-마킹된다; 그러나, 샘플 값은 패딩된 샘플 값 또는 디폴트 값이도록 수정된다. 대안으로 또는 부가적으로, 필터는 다른 샘플들에 대해 사용된 필터와 동일하게 유지된다. 일 예에서, 디폴트 값은 비트-심도에 의존적이다. 다른 예에서, 패딩은 이용 가능한 것으로서 마킹되는 좌측/우측/위/아래 샘플로부터의 것일 수 있다.

[0174] 일반적으로, 다른 타일에 있는 루마 샘플들에 대해, 비디오 코더는 타일 경계 밖의 픽셀들을 이용 가능하지 않은 것으로서 마킹하고, 이들을 다운-샘플링 프로세스에 포함하지 않을 수도 있다. 일부 예들에서, 비디오 코더는 다른 타일에서 루마 샘플들을 이용 가능한 것으로서 마킹하지만, 다른 타일에서 이러한 루마 샘플들에 대한 패딩된 픽셀들을 사용할 수도 있다. 다른 예로서, 비디오 코더는, 샘플들을 이용 가능하지 않은 것으로서 마킹하기 보다는 차라리, 다른 타일에서 루마 샘플들에 대해 패딩된 "확장된" 값들을 사용할 수도 있다 (예를 들어, 비트 심도에 기초한 1/2 가능한 값, 따라서 8 비트, 128 을 사용).

[0175] 일부 예들에서, 비디오 코더는 상이한 크로마 컬러 컴포넌트들 (Cb 또는 Cr) 에 상이한 필터들을 적용할 수도 있다. 일부 예들에서, LM 예측 모드가 인에이블되는 경우, 다운-샘플링 필터의 하나 이상의 세트들은 시퀀스 파라미터 세트 (SPS), 픽처 파라미터 세트 (PPS), 또는 슬라이스 헤더 중 어느 하나에서 더 시그널링될 수도 있다. 대안으로 또는 부가적으로, 보충 강화 정보 (SEI) 메시지 신택스가 도입되어 다운-샘플링 필터를 설명한다. 대안으로 또는 부가적으로, 또한, 디폴트 다운-샘플링 필터가, 예를 들어 시그널링 없이 6-탭 필터 [1, 2, 1; 1, 2, 1] 가 정의된다. 대안으로 또는 부가적으로, 하나의 PU/CU/최대 CU 는 LM 예측 모드에서 사용되는 필터의 인덱스를 시그널링할 수도 있다. 대안으로 또는 부가적으로, 필터 탭의 사용은 시그널링 없이 비디오 디코더 (30) 에 의해 온-더-플라이 (on-the-fly) 로 도출될 수도 있다. 필터 지원을 제공하기 위한 다른 방식들이 또한 존재할 수도 있다.

[0176] 일 예에서, 또한 a_i 는 $\alpha_{(i+3)}$ 와 동일하다는 제약이 적용된다. 일 예에서, 또한 a_i 는 $\alpha_{(i+2)}$ 와 동일하다는 제약이 적용되고 i 는 0 또는 3 과 동일하다. 일 예에서, 이 예시의 기법은 더 큰 코딩된 CU들, 예를 들

어 16×16 보다 큰 CU 사이즈에 대해서만 인에이블될 수도 있다. 일 예에서, 파라미터들 중 하나 이상은 0 인 것으로 제한된다.

[0177] 또한, 비디오 코더는 크로스 컴포넌트 잔차 예측에 대해 상기의 기법들 중 하나 이상을 또한 적용할 수도 있고, 여기서 다운-샘플링된 루마 잔차는 크로마 잔차를 예측하는데 사용된다. 이 경우에서, 다운-샘플링 프로세스는 일 예로서, 복원된 루마 잔차에 적용된다.

[0178] 다음은, 본 개시물에서 설명된 기법들이 비디오 코더에 의해 구현될 수도 있는 예시의 방식이다. 예시의 구현 기법은 제한하는 것으로 간주되지 않아야 한다.

[0179] 이하는 좌측 픽처 경계에서 샘플들에 대해 상이한 다운-샘플링 프로세스들을 적용하는 예이다. 현재 루마 블록에 대한 다운-샘플링 프로세스는 다음과 같이 정의된다:

[0180] - 크로마 샘플이 픽처의 좌측 경계에 위치되지 않으면, 6-탭 필터, 예를 들어 [1 2 1; 1 2 1] 는 대응하는 다운-샘플링된 루마 샘플을 도출하는데 적용된다:

$$rec_L(i,j) = (\mathrm{Re}\,c_{LOrig}[\;\;2i,2j]*2 + \mathrm{Re}\,c_{LOrig}[\;\;2i,2j+1] + \mathrm{Re}\,c_{LOrig}[\;\;2i,2j-1]$$

[0181] $$+ \mathrm{Re}\,c_{LOrig}[2i+1,2j]*2 + \mathrm{Re}\,c_{LOrig}[2i+1,2j+1] + \mathrm{Re}\,c_{LOrig}[2i+1,2j-1] + offset0) >> 3 \quad (13)$$

[0182] - 그렇지 않고, 크로마 샘플이 픽처의 좌측 경계에 위치되면, 2-탭 필터, 예를 들어 [1; 1] 는 대응하는 다운-샘플링된 루마 샘플을 도출하는데 적용된다:

[0183] $$rec_L(i,j) = (\mathrm{Rec}_{LOrig}[2i,2j] + \mathrm{Rec}_{Lorig}[2i,2j+1] + offset1) >> 1 \quad (14)$$

[0184] 일 예에서, offset0 및 offset1 양자 모두는 0 과 동일하게 설정된다. 다른 예에서, offset0 은 4 와 동일하게 설정되고 offset1 은 1 과 동일하게 설정된다.

[0185] HEVC 에서, 직사각형 PU들에 대해서도, 스퀘어 변환이 항상 적용된다. 예를 들어, 도 16 은 N×N 변환을 갖는 nR×2N 예측 모드를 예시하는 개념도이다. 도 16 의 예에서, nR×2N 예측 모드는 2N×2N 블록 사이즈를 갖는 코딩 블록 (100) 을 0.5N×2N 및 1.5N×2N 의 사이즈들을 갖는 2 개의 예측 블록들로 각각 파티셔닝한다. 그러나, 도 16 의 예에서, 변환 블록 사이즈는 N×N 이다.

[0186] 도 17 은 2N×N, 2N×nD 및 2N×nU 예측 모드들에 대해 비-스퀘어 쿼드트리 (NSQT) 를 예시하는 개념도이다. 도 17 에서, 레벨0 에서 2N×2N 블록은 레벨 1 에 위치된 4 개의 2N×0.5N 블록들로 스플리팅되고; 레벨 1 에서 블록은 또한, 레벨 2 에서 4 개의 N×0.25N 블록들로 스플리팅된다. 도 18 은 N×2N, nR×2N 및 nL×2N 예측 모드들에 대한 NSQT 를 예시하는 개념도이다. 도 18 에서, 레벨 0 에서 2N×2N 블록은 레벨 1 에 위치된 4 개의 0.5N×2N 블록들로 스플리팅되고; 레벨 1 에서 블록은 또한, 레벨 2 에서 4 개의 0.25N×N 블록들로 스플리팅된다.

[0187] 잔차들이 2 개의 결합 예측 블록 (connective prediction block) 들의 경계들에서 불연속적일 수도 있다는 것을 고려하면, 고 주파수 변환 계수들이 생성될 것이고 코딩 성능이 영향을 받을 것이다. 본 개시물에서, 결합 예측 블록들은 4 개의 경계들 중 적어도 하나를 공유하는 예측 블록들이다. 따라서, 2012년 5월 7-9일, 크라쿠프, 폴란드, 2012 PCS (Picture Coding Symposium), pp. 505-508, Yuan 등의,"Non-Square Quadtree Transform Structure for HEVC" (이하에서, "Yuan") 에서, 비-스퀘어 쿼드트리 변환 (NSQT) 구조가 설명된다.

[0188] NSQT 에서, 2 개의 추가적인 변환 블록 사이즈들이 추가된다: 2N×0.5N 및 0.5N×2N. 이 구조에서, 변환 블록은 2N×0.5N 및 0.5N×2N 으로 스플리팅되고, 변환 매트릭스는 0.5N×0.5N 및 2N×2N 변환 매트릭스들을 재사용함으로써 획득될 수 있다. 본 개시물에서, 변환 매트릭스는 또한, 변환 코어로서 지칭될 수도 있다. Yuan 에서, HEVC 의 N×N 양자화 테이블은 2N×0.5N 및 0.5N×2N 변환 블록들의 변환 계수들을 양자화하는데 재사용된다.

[0189] 위에서 언급된 바와 같이, 비디오 코더는 변환을 적용하여 샘플들을 주파수 도메인으로 컨버팅할 수도 있고, 또는 그 반대일 수도 있다. HEVC 에서 적용된 변환들의 특정 유형들은 이산 코사인 변환들의 2 개의 유형들, 즉 DCT-II 및 4×4 DST-VII 이다. Xin Zhao 등의, 미국 특허공개 2016/0219290 A1 은 인터 및 인트라 코딩된 블록들 양자 모두에 대해 DCT-II 및 4×4 DST-VII 에 추가하여 강화된 다중 변환 (EMT) 스킴을 제안하였다. EMT 스킴은 HEVC 에서의 현재 변환들 외에 DCT/이산 사인 변환 (DST) 패밀리들로부터 다수의 선택된 변환들

을 이용한다. 미국 특허 공개 제 2016/0219290 에서의 새롭게 도입된 변환 매트릭스들은 DST-Ⅶ, DCT-Ⅷ, DST-Ⅰ및 DCT-Ⅴ 이다.

[0190] 미국 특허공개 제 2016/0219290 A1 에서 제안된 EMT 는 64×64 보다 더 작은 CU들에 적용하고, EMT 가 적용하는지 또는 아닌지 여부는 CU 내의 모든 TU들에 대해, 플래그, 즉 EMT 플래그를 사용하여 CU 레벨에서 제어된다.

EMT-인에이블된 CU 내의 각각의 TU 에 대해, 사용될 수평 또는 수직 변환은 선택된 변환 세트, 즉 EMT 인덱스로 인덱스에 의해 시그널링된다. 각각의 변환 세트는 전술된 변환 매트릭스들로부터 2 개의 변환들을 선택함으로써 형성된다.

[0191] 인트라 예측 잔차에 대해, 변환 세트들은, 2012년 1월, IEEETrans. Circuits Syst. Video Technol., vol. 22, no. 1, pp. 138-151, X.Zhao 등의, "Video coding with rate-distortion optimized transform" 에서 설명된 바와 같이 인트라 예측 모드에 기초하여 미리-정의된다; 따라서 각각의 인트라 예측 모드는 그 자신의 변환 세트를 갖는다. 예를 들어, 하나의 변한 세트는 {DCT-Ⅷ, DST-Ⅶ} 일 수 있다. 수평 변환에 대한 변환 세트는, 동일한 인트라 예측 모드에 대해서도 수직 변환에 대한 변환 세트와 상이할 수도 있다는 것을 주목한다.

그러나, 모든 인트라 예측 모드들에 대한 상이한 변환 세트들의 총 수 뿐만 아니라 새롭게 도입된 변환들의 수는 제한된다. 그러나, 인터 예측 잔차에 대해, 단지 하나의 변환 세트가 모든 인터 모드들에 대해 그리고 수평 및 수직 변환들 양자 모두에 대해 사용된다.

[0192] 멀티-뷰 비디오 코딩에서 일루미네이션 보상 (IC) 은, 각각의 카메라가 광 소스에 대해 상이한 노출을 가질 수도 있기 때문에 상이한 뷰들 간의 일루미네이션 불일치를 보상하기 위해 사용된다. 통상적으로, 가중 팩터 및/또는 오프셋이 사용되어 상이한 뷰에서 예측 블록과 코딩된 블록 간의 차이들을 보상한다. 일루미네이션 보상은 인터-뷰 레퍼런스 픽처들로부터 예측된 블록들에 대한 코딩 효율성을 개선시키도록 도입되었다. 따라서, 일루미네이션 보상은 인터-뷰 레퍼런스 픽처에 의해 예측된 블록들에만 적용할 수도 있다.

[0193] 2012년 7월 16-20일, 스톡홀름, 스웨덴, 1 차 회의, ITU-T SG 16 WP3 및 ISO/IEC JTC1/SC29/WG11 의 3D 비디오 코딩 확장 개발에 대한 공동 협력 팀, Liu 등의, "3D-CE1.h related: Illumination Compensation for Inter-View Prediction", 문헌 JCT3V-A0086 (이하, JCT3V-A0086) 은 일루미네이션 보상 (IC) 을 설명한다. JCT3V-A0086 에서, IC 는 인터-뷰 예측에 대해 인에이블된다. 또한, JCT3V-A0086 에서 설명된 바와 같이, IC 프로세스는 레퍼런스 블록의 이웃하는 샘플들 및 현재 CU 의 이웃하는 샘플들에 기초하여 IC 파라미터들을 도출한다. JCT3V-A0086 에서, IC 는 단지 2N×2N 파티션 모드에 적용한다. 또한, JCT3V-A0086 에서, AMVP 모드에 대해, 하나의 IC 플래그는 인터-뷰 레퍼런스 픽처로부터 예측되는 각각의 CU 에 대해 시그널링된다. 머지 모드에 있어서, 비트들을 절감하기 위해, IC 플래그는, PU 의 머지 인덱스가 0 과 동일하지 않은 경우에만 시그널링된다. IC 플래그는, IC 가 CU 에 대해 사용되는지 여부를 나타낸다. IC 는 시간적 레퍼런스 픽처들로부터 단지 예측되는 CU들에 적용하지 않는다.

[0194] JCT3V-A0086 에서 설명된 바와 같이, 인터-뷰 예측에서 사용된 선형 IC 모델은 등식 (6) 에 도시된다:

[0195] $$p(i,j) = a * r(i + dv_x, j + dv_y + b), \text{ 여기서 } (i,j) \in PU_c \quad (15)$$

[0196] 여기서, PU_c 는 현재 PU 이고, (i, j) 는 PU_c 에서 픽셀들의 좌표들이고, (dv_x, dv_y) 는 PU_c 의 디스패리티 벡터이고, $p(i, j)$ 는 PU_c 의 예측이며, r 은 이웃하는 뷰로부터의 현재 PU 의 레퍼런스 픽처이다. a 및 b 는 선형 IC 모델의 파라미터들이다.

[0197] JCT3V-A0086 에서, 도 19 에 도시된 바와 같은 픽셀들의 2 개의 세트들은 현재 PU 에 대한 파라미터들 a 및 b 를 추정하는데 사용된다. 픽셀들의 제 1 세트는 현재 CU (즉, 현재 PU 를 포함하는 CU) 의 좌측 컬럼 및 상부 로우에서 이용 가능한 복원된 이웃하는 픽셀들을 포함한다. 픽셀들의 제 2 세트는 현재 CU 의 레퍼런스 블록의 대응하는 이웃하는 픽셀들을 포함한다. 현재 PU 의 디스패리티 벡터는 현재 CU 의 레퍼런스 블록을 찾는데 사용된다.

[0198] 도 19 는 IC 모델에서 파라미터들을 추정하는데 사용된 이웃하는 픽셀들을 예시한다. 특히, 도 19 는 현재 CU (110) 및 레퍼런스 블록 (112) 을 포함한다. 도 19 의 각각의 개별의 스퀘어는 개별의 샘플에 대응한다.

따라서, 현재 CU (110) 및 레퍼런스 블록 (112) 은 각각 64 개의 샘플들을 포함한다. 현재 CU (110) 에 인접한 원들을 인클로징하는 스퀘어들은 현재 CU (110) 의 이웃하는 샘플들 (즉, Rec_{neigh}) 에 대응한다. 레퍼런스 블록 CU (112) 에 인접한 원들을 인클로징하는 스퀘어들은 이웃하는 블록 (112) 의 이웃하는 샘플들

(즉, $Rec_{refneigh}$) 에 대응한다. 본 개시물의 다른 곳에서 설명된 바와 같이, 비디오 코더는 Rec_{neigh} 및 $Rec_{refneigh}$ 을 사용하여 IC 에 대한 파라미터들을 추정할 수도 있다.

[0199] 또한, JCT3V-A0086 에서 설명된 바와 같이, Rec_{neigh} 는 현재 CU 에 의해 사용된 이웃하는 픽셀 세트를 가리킨다고 하자. $Rec_{refneigh}$ 는 현재 CU 의 레퍼런스 블록에 의해 사용된 이웃하는 픽셀 세트를 가리킨다고 하자.

현재 CU 의 사이즈 및 현재 CU 의 레퍼런스 블록의 사이즈 양자 모두는 N×N 과 동일하다고 하자. 2N 은 Rec_{neigh} 및 $Rec_{refneigh}$ 에서의 픽셀들의 수를 가리킨다고 하자. 그러면, a 및 b 는 다음과 같이 계산될 수 있다:

[0200]
$$a = \frac{2N \cdot \sum_{i=0}^{2N-1} \mathrm{Rec}_{neig}(i) \cdot \mathrm{Rec}_{refneig}(i) - \sum_{i=0}^{2N-1} \mathrm{Rec}_{neig}(i) \cdot \sum_{i=0}^{2N-1} \mathrm{Rec}_{refneig}(i)}{2N \cdot \sum_{i=0}^{2N-1} \mathrm{Rec}_{refneig}(i) \cdot \mathrm{Rec}_{refneig}(i) - \left(\sum_{i=0}^{2N-1} \mathrm{Rec}_{refneig}(i) \right)^2} \quad (16)$$

[0201]
$$b = \frac{\sum_{i=0}^{2N-1} \mathrm{Rec}_{neig}(i) - a \cdot \sum_{i=0}^{2N-1} \mathrm{Rec}_{refneig}(i)}{2N} \quad (17)$$

[0202] 일부 경우들에서, 단지 a 는 선형 모델에 사용되고 b 는 항상 0 과 동일하게 설정되며, 또는 단지 b 가 사용되고 a 는 항상 1 과 동일하게 설정된다.

[0203] VCEG-AZ06 에서, 로컬 일루미네이션 보상 (LIC) 은 각각의 인터-모드 코딩된 CU 에 대해 적응적으로 인에이블 또는 디스에이블된다. VCEG-AZ06 에서, LIC 는 스케일링 팩터 (a) 및 오프셋 (b) 를 사용하여, 일루미네이션 변화들에 대한 선형 모델에 기초한다. 도 20 은 VCEG-AZ06 에서 설명된 바와 같이 IC 파라미터들을 도출하기 위해 사용된 예시의 이웃하는 샘플들을 예시하는 개념도이다.

[0204] VCEG-AZ06 에서, LIC 가 CU 에 대해 적용하는 경우, CU 에 속하는 각각의 PU/서브-PU 에 대해, 비디오 코더는 CU 의 서브-샘플링된 (2:1 서브-샘플링) 이웃하는 샘플들 및 레퍼런스 픽처에서 (현재 PU/서브-PU 의 모션 정보에 의해 식별된) 대응하는 픽셀들을 사용하는 방식으로 LIC 파라미터들을 도출한다. N×N 과 동일한 사이즈를 갖는 CU 에 대해, 등식들 (16) 및 (17) 에서 사용된 경계 픽셀들의 총 수는 2N 대신에 N 이다. 일 예가 도 20 에 예시된다. 따라서, 도 20 은 일루미네이션 보상 모델에서 파라미터들을 추정하는데 사용된 예시의 이웃하는 픽셀들을 예시하는 개념도이고, 여기서 현재 CU (116) 의 레퍼런스 블록 (114) 은 현재 PU 의 디스패리티 벡터를 사용하여 발견된다. VCEG-AZ06 에서, IC 파라미터들은 각각의 예측 방향에 대해 별개로 도출 및 적용된다. 비디오 코더는 전술된 이웃하는 샘플들에 기초하여 파라미터들 (a 및 b) 를 도출하도록 최소 자승 오차를 이용할 수도 있다.

[0205] HEVC 에서의 현재의 RQT 설계, 및 NSQT 및 IC 와 같은 다른 기법들은 다음의 단점들을 가질 수도 있다. 예를 들어, NSQT 또는 HEVC 의 변환 트리가 사용되는지 여부에 관계 없이, PU 정보를 고려하지 않고 차선일 수도 있는 쿼드-트리 구조가 항상 이용된다. 그러나, HEVC 는 단지, 인트라 예측 모드들에 대한 스퀘어 PU 들을 지원한다.

[0206] JCTVC-G135 에서 행해진 것과 같이, 2N×N 및 N×2N 파티션들을 인트라 모드들에 도입하는 것은 다음의 문제들을 가질 수도 있다. 먼저, AMP 가 허용되지 않는다. 두 번째, 고 코딩 효율성을 달성하기 위해 변환 트리 구조를 정의하는 방법이 연구되어 있지 않다. 세 번째, LM 예측 모드는 단지 스퀘어 PU들과 사용되고 있고, 비-스퀘어 PU들을 갖는 LM 예측 모드에서 사용된 파라미터들 (α 및 β) 을 도출하는 방법이 알려져 있지 않다. 네 번째, 이전의 기법들에서, 계수들은, 이웃하는 계수들 간의 상관을 감소시킬 수도 있는, 스퀘어 형태에 있도록 재구성되어야 한다. 또한, 현재 EMT 설계는, EMT 가 CU 레벨에서 제어된다는 문제를 갖는다.

그러나, CU 레벨에서 EMT 를 제어하는 것은, CU 에서 각각의 PU 의 잔차 특징 (예를 들어, 분포들) 이 상이하면 효율적이지 않다.

[0207] 전술된 문제들을 해결하기 위해, 본 개시물은 다음의 기법들을 제안한다. 다음의 아이템화된 기법들은 개별적으로 적용될 수도 있다. 대안으로, 이들의 임의의 조합이 적용될 수도 있다. 다음의 설명에서, CU 사이즈는 MxM 에 의해 표기되고 PU 사이즈는 K×L 에 의해 표기되며, K 및 L 양자 모두는 M 보다 더 이상 길지 않

다.

[0208] 본 개시물의 제 1 예시의 기법에 따르면, 변환 트리가 쿼터 트리인 것에 제한되지 않는다는 것이 제안된다. 예를 들어, 변환 쿼드-트리 및 변환 바이너리 트리가 결합될 수도 있다. 즉, 적어도 소정의 변환 심도에 대해, 하나의 TU 는 2 개의 더 작은 TU들 또는 4 개의 더 작은 TU들로 스플리팅될 수도 있다. 본 개시물에서, 변환 트리의 각각의 개별의 노드에 대해, 개별의 노드의 개별의 변환 심도는 변환 트리의 루트 노드와 개별의 노드 간의 변환 트리에서의 노드들의 수를 지칭한다. TU 를 2 개의 TU들 또는 4 개의 TU들로 스플리팅하기 위한 유연성은, PU 경계들과 TU 경계들을 정렬하는 방식으로 변환 트리를 구조화하도록 비디오 인코더 (20) 의 능력을 강화시킬 수도 있다. PU 경계들과 TU 경계들을 정렬시키는 것은 압축 성능을 증가시킬 수도 있다.

[0209] 따라서, 이 예에서, 비디오 인코더 (20) 는 트리 구조에 기초하여 비디오 데이터의 CU 를 CU 의 TU들로 파티셔닝할 수도 있다. 이 예에서, 트리 구조의 루트 노드는 CU 의 코딩 블록에 대응한다. 또한, 이 예에서, 트리 구조의 각각의 개별의 비-루트 노드는 개별의 비-루트 노드의 페어런트 노드에 대응하는 블록의 파티션인 개별의 블록에 대응한다. 이 예에서, 트리 구조의 리프 노드들은 CU의 TU들에 대응한다. 이 예에서, 트리 구조에서의 적어도 하나의 노드는 트리 구조에서 정확히 2 개의 차일드 노드들을 갖는다. 일부 경우들에서, 트리 구조에서의 적어도 하나의 노드는 트리 구조에서 정확히 4 개의 차일드 노드들을 가질 수도 있다. 이 예에서, 비디오 인코더 (20) 는, 비디오 데이터의 인코딩된 표현을 포함하는 비트스트림에서, CU 의 TU들의 하나 이상을 나타내는 데이터를 포함할 수도 있다.

[0210] 대응하는 예에서, 비디오 디코더 (30) 는 트리 구조에 기초하여 CU 가 CU 의 TU들로 파티셔닝된다고 결정할 수도 있다. 이 예에서, 트리 구조의 루트 노드는 CU 의 코딩 블록에 대응한다. 또한, 이 예에서, 트리 구조의 각각의 개별의 비-루트 노드는 개별의 비-루트 노드의 페어런트 노드에 대응하는 블록의 파티션인 개별의 블록에 대응한다. 이 예에서, 트리 구조의 리프 노드들은 CU 의 TU들에 대응한다. 이 예에서, 트리 구조에서 적어도 하나의 노드는 트리 구조에서 정확히 2 개의 차일드 노드들을 갖고, 트리 구조에서 적어도 하나의 노드는 트리 구조에서 정확히 4 개의 차일드 노드들을 갖는다. 이 예에서, 비디오 디코더 (30) 는 CU 의 TU들 중 적어도 하나에 대한 데이터에 기초하여, CU 의 코딩 블록을 복원할 수도 있다.

[0211] 또한, 변환 트리가 쿼터 트리인 것에 제한되는 (즉, 모든 비-리프 노드들이 4 개의 차일드 노드들을 갖는 트리이도록 요구되지 않는) 예들에서, 0 과 동일한 변환 심도에 대해, MxM 과 동일한 사이즈를 갖는 스퀘어 변환이 적용된다. 1 과 동일한 변환 심도에 대해, 변환은 (PU들의 수에 따라) 2 또는 4 개로 스플리팅되고, 변환 사이즈는 K×L 와 동일하다. 나머지 변환 심도들에 대해, 쿼드-트리 구조가 여전히 적용되고, 여기서 하나의 TU 는 4 개의 더 작은 것들로 스플리팅된다, 즉 2 와 동일한 변환 심도에 대해, 변환 사이즈는 K/2 x L/2 로 설정된다. 일 예가 도 21 에 주어진다. 변환 심도 1 에서 2 또는 4 로의 변환의 스플리팅을 제한하는 하나의 이유는 PU 사이즈들과 변환 사이즈들을 정렬시키기 위한 것이다, 예를 들어 PU 사이즈가 2N×N 또는 N×2N 이면, 2 로의 스플리팅이 바람직할 수도 있다. PU 가 N×N 파티션이면, 4-웨이 변환 스플리팅은 더 좋은 결과들을 산출할 수도 있다. 다른 이유로, 대응하는 변환 매트릭스가 알려져 있지 않으면, 예를 들어 AMP 가 사용되면, 하나의 16×16 CU 는 4×16 및 12×16 PU들로 스플리팅될 수도 있는 한편, 12×12 변환은 정의되지 않고, 따라서 4 로의 스플리팅이 이 경우에 대해 사용될 수도 있다.

[0212] 도 21 은 2N×N 와 동일한 파티션 사이즈에 대한 예시의 변환 구조를 예시하는 개념도이다. 도 21 및 다음의 도면들에서, 점선들은 다음의 변환 심도에 대한 스플리팅 정보를 나타낸다. 특히, 도 21 에서, 변환 블록 (130) 은 CU 의 코딩 블록과 동일한 사이즈를 갖는다. 변환 블록 (130) 은 변환 블록들 (132 및 134) 로 파티셔닝된다. 또한, 도 21 의 예에서, 변환 블록 (132) 은 변환 블록들 (136, 137, 138, 및 139) 로 파티셔닝된다. 변환 블록 (134) 은 변환 블록들 (140, 141, 142 및 143) 로 파티셔닝된다. 따라서, 도 21 에 도시된 바와 같이, 루트 노드는 2 개의 차일드 노드들을 가질 수도 있지만, 다른 변환 심도들에서 노드들은 0 또는 4 개의 차일드 노드들을 갖도록 요구될 수도 있다.

[0213] CU 의 변환 트리가 쿼터 트리인 것에 제한되지 않는 일부 예들에서, 바이너리 트리 또는 쿼터 트리 중 어느 하나가 적용된다. 비디오 코더는, CU 에서의 PU들의 수에 기초하여 바이너리 트리 또는 쿼터 트리가 적용되는지 여부를 결정할 수도 있다. 예를 들어, 2 개의 PU들이 존재하는 경우, 비디오 코더는 바이너리 변환 트리를 이용한다. CU 가 4 개의 PU들을 가지면, 비디오 코더는 쿼터 트리 구조를 사용하여 CU 를 TU들로 파티셔닝할 수도 있다. 일 예에서, 바이너리 트리 또는 쿼터 트리 중 어느 하나를 선택하는 방법은 단지, 1 과 같은 소정의 변환 심도들에 적용된다.

[0214] 따라서, 이 예에서, 비디오 인코더 (20) 는 트리 구조에 기초하여 비디오 데이터의 CU 를 CU 의 TU들로 파티셔닝할 수도 있다. 이 예에서, 트리 구조의 루트 노드는 CU 의 코딩 블록에 대응하고, 트리 구조의 각각의 개별의 비-루트 노드는 개별의 비-루트 노드의 페어런트 노드에 대응하는 블록의 파티션인 개별의 블록에 대응하고, 트리 구조의 리프 노드들은 CU들의 TU들에 대응하며, CU 는 하나 이상의 PU들을 갖는다. 또한, 이 예에서, CU 의 PU들의 수에 따라, 다음 중 정확히 하나가 적용된다: 트리 구조에서 각각의 노드는 트리 구조에서 정확히 2 개의 차일드 노드들을 갖고, 또는 트리 구조에서 각각의 노드는 트리 구조에서 정확히 4 개의 차일드 노드들을 갖는다. 이 예에서, 비디오 인코더 (20) 는, 비디오 데이터의 인코딩된 표현을 포함하는 비트스트림에서, CU 의 TU들의 하나 이상을 나타내는 데이터를 포함할 수도 있다.

[0215] 대응하는 예에서, 비디오 디코더 (30) 는 트리 구조에 기초하여 비디오 데이터의 CU 가 CU 의 TU들로 파티셔닝된다고 결정할 수도 있다. 이 예에서, 트리 구조의 루트 노드는 CU 의 코딩 블록에 대응하고, 트리 구조의 각각의 개별의 비-루트 노드는 개별의 비-루트 노드의 페어런트 노드에 대응하는 블록의 파티션인 개별의 블록에 대응하고, 트리 구조의 리프 노드들은 CU들의 TU들에 대응하며, CU 는 하나 이상의 PU들을 갖는다. 이 예에서, CU 의 PU들의 수에 따라, 다음 중 정확히 하나가 적용된다: 트리 구조에서 각각의 노드는 트리 구조에서 정확히 2 개의 차일드 노드들을 갖고, 또는 트리 구조에서 각각의 노드는 트리 구조에서 정확히 4 개의 차일드 노드들을 갖는다. 또한, 이 예에서, 비디오 디코더 (30) 는 CU 의 TU들 중 적어도 하나에 대한 데이터에 기초하여, CU 의 코딩 블록을 복원할 수도 있다.

[0216] 변환 트리가 쿼터 트리인 것에 제한되지 않는 일부 예들에서, 바이너리 트리 또는 쿼터 트리 중 어느 하나의 스플리팅 방법이 시그널링된다. 예를 들어, 비디오 인코더 (20) 는, CU 가 바이너리 트리에 따라 또는 쿼터 트리에 따라 TU들로 파티셔닝되는지 여부를 나타내는 신택스 엘리먼트를 나타내는 데이터를 비트스트림에서 포함할 수도 있다. 이 예에서, 비디오 디코더 (30) 는, 비트스트림에서의 데이터에 기초하여, 신택스 엘리먼트의 값을 결정할 수도 있다. 또한, 이 예에서, 비디오 디코더 (30) 는, 신택스 엘리먼트의 값에 기초하여, 바이너리 트리에 따라 또는 쿼터 트리에 따라 CU 가 TU들로 파티셔닝되는지 여부를 결정할 수도 있다.

[0217] 대안으로, 또한 시그널링은 소정의 PU 파티션들에 대해 스킵될 수도 있다. 다시 말해, 비디오 인코더 (20) 는, 블록이 PU들로 스플리팅되는 방법에 기초하여 블록에 대한 변환 트리 스플리팅의 시그널링을 스킵할 수도 있다. 예를 들어, 일 예에서 2N×N 과 동일한 PU 파티션들에 대해, 바이너리 트리가 항상 사용되고, 따라서 바이너리 또는 쿼터 트리 스플리팅이 대응하는 변환 트리에서 사용된다는 것을 시그널링할 필요가 없다.

[0218] 본 개시물의 제 2 기법에 따르면, 소정의 변환 심도에서, 변환 사이즈는 직사각형 PU들에 대한 PU 사이즈와 동일하다는 것이 제안된다. 본 개시물에서, 변환 사이즈는 TU 의 변환 블록의 사이즈를 지칭한다. 따라서, 이 예에서, CU 의 변환 트리의 특정 심도에서 노드들에 대응하는 변환 블록들은 CU 의 PU들의 예측 블록들과 동일한 사이즈를 갖는다. 이전에 논의된 바와 같이, TU 경계들을 PU 경계들과 정렬시키는 것은 압축 성능을 개선시킬 수도 있다.

[0219] 제 2 기법의 일 예에서, 상기의 방법은 단지, 인터 코딩된 CU들에 적용된다. 다시 말해, 비디오 코딩 표준은, 변환 사이즈들이 인터 코딩된 CU들에 대한 PU들과 동일하다는 것을 보장하도록 비디오 인코더 (20) 에 요구할 수도 있지만, 이 요건은 인트라 코딩된 CU들에 적용하지 않는다.

[0220] 또한, 일부 예들에서, 비디오 인코더 (20) 는, 3 개의 컬러 컴포넌트들 (예를 들어, Y, Cb, 및 Cr) 에 대한 적어도 하나의 넌-제로 계수가 존재하는지 여부를 나타내도록 각각의 PU 에 대해 하나의 플래그를 시그널링할 수도 있다. 위에서 언급된 바와 같이, 본 개시물의 제 2 기법은 CU 의 TU들의 사이즈들이 변환 트리에서의 특정 심도에서 CU 의 PU들의 사이즈들과 동일하도록 요구한다. 따라서, 특정 심도에서, 변환 트리는 CU 의 각각의 개별의 PU 에 대한 개별의 변환 트리 노드를 포함한다. 특정 심도에서 변환 트리의 각각의 개별의 변환 트리 노드에 대해, 개별의 변환 트리 노드는 대응하는 PU 의 루마 예측 블록 및 크로마 예측 블록들과 동일한 사이즈들 및 형상들을 갖는 루마 변환 블록 및 크로마 변환 블록들에 대응한다. 따라서, 인코더 (20) 는 대응하는 PU 에서 정보를 시그널링함으로써 특정 심도에서 변환 트리 노드 (및 특정 심도에서 변환 트리 노드의 후손의 변환 트리 노드들) 에 관한 정보를 시그널링할 수도 있다. 예를 들어, 비디오 인코더 (20) 는, 비트스트림에서, PU 에 대한 제 1 신택스 엘리먼트, PU 에 대한 제 2 신택스 엘리먼트, 및 PU 에 대한 제 3 신택스 엘리먼트를 시그널링할 수도 있다. 이 예에서, PU 에 대한 제 1 신택스 엘리먼트는 대응하는 변환 트리 노드 또는 그 후손의 변환 트리 노드의 루마 계수 블록에서 넌-제로 변환 계수가 존재하는지 여부를 나타내고, PU 에 대한 제 2 신택스 엘리먼트는 대응하는 변환 트리 노드 또는 그 후손의 변환 트리 노드의 Cb 계수 블록에서 넌-제로 변환 계수가 존재하는지 여부를 나타내며, PU 에 대한 제 3 신택스 엘리먼트는 대응하는 변환 트리 노

드 또는 그 후손의 변환 트리 노드의 Cr 계수 블록에서 넌-제로 변환 계수가 존재하는지 여부를 나타낸다.

[0221] 이전의 기법들에서, 직사각형의 PU들은 단지, 인터 예측된 CU들에 대해 허용되었다. 그러나, 본 개시물의 제 2 기법의 일부 예들에서, 직사각형 PU들 (예컨대, 2N×N, N×2N) 이 인트라 코딩된 CU들에 도입되는 경우, 상기의 방법 (즉, 특정 변환 심도에서 변환 사이즈가 PU 사이즈와 동일하도록 요구함) 이 또한, 적용된다.

[0222] 제 3 기법에 따르면, 하나의 TU 는 다수의 더 작은 TU들로 스플리팅될 수도 있는 한편, 더 작은 TU들의 사이즈들은 상이할 수도 있다. 다시 말해, 비디오 코더는 TU 를 2 개의 상이한-사이즈의 차일드 TU들로 스플리팅할 수도 있다. 일부 경우들에서, TU 를 2 이상의 상이한-사이즈의 차일드 TU들로 스플리팅하는 것은, TU 를 2 이상의 상이한-사이즈의 차일드 TU들로 스플리팅하는 것이 PU 경계들과 차일드 TU들의 경계들을 더 잘 정렬시킬 수도 있기 때문에 AMP 가 인에이블되는 경우들에서 비디오 코딩 성능을 개선시킬 수도 있다. 본 개시물의 다른 곳에서 논의된 바와 같이, TU 의 경계들을 PU들의 경계들과 정렬시키는 것은 예측 블록들 간의 경계들에서의 불연속성과 연관된 고 주파수 변환 계수들의 발생을 감소시키고, 따라서 압축 효율성을 증가시킬 수도 있다. 예를 들어, 블록 (예를 들어, CU) 이 12×16 PU 를 가지면, 12×16 PU 에 대응하는 블록의 일부는 2 개의 8×8 TU들 플러스 2 개의 4×4 TU들, 또는 2 개의 8×8 TU들 플러스 하나의 4×16 TU 로 스플리팅될 수도 있다.

[0223] 제 3 기법의 일 예에서, AMP 모드가 하나의 CU 에 대해 인에이블되는 경우, 더 큰 PU 에 대한 변환 트리는 더 작은 PU 와 동일한 것을 갖는 2 개의 부분들로 스플리팅되고, 나머지는 다른 TU 로서 스플리팅될 수도 있다. 일 예가 도 22 에 주어진다. 도 22 는 본 개시물의 기법에 따라, N×N/4(U) 와 동일한 파티션 사이즈에 대한 변환 구조를 예시하는 개념도이다. 도 22 의 예에서, 변환 블록 (150) 은 변환 블록들 (152 및 154) 로 파티셔닝된다. 또한, 도 22 의 예에서, 변환 블록 (152) 은 변환 블록들 (156, 158, 160, 및 162) 로 파티셔닝된다. 변환 블록 (154) 은 변환 블록들 (164 및 166) 로 파티셔닝된다. 변환 심도 2 에서 우측 분기는, 2 개의 스플릿 변환 사이즈들이 상이하다는 것을 나타낸다. 즉, 변환 블록 (164) 및 변환 블록 (166) 은 상이한 사이즈들을 갖는다.

[0224] 제 3 기법의 일부 예들에서, TU 의 비대칭 스플리팅은 단지 AMP 경우에 적용 가능하고, 여기서 2 개의 PU 사이즈들은 상이하고 또는 하나의 CU 는 상이한 사이즈들을 갖는 PU들 중 적어도 2 개를 갖는 다수의 PU들을 포함한다. 다시 말해, 비디오 코더는 단지, TU 를 포함하는 CU 가 상이한 사이즈들의 PU들로 스플리팅되는 경우 TU 를 상이한 사이즈들의 차일드 TU들로 스플리팅할 수도 있다.

[0225] 따라서, TU 가 다수의 상이한-사이즈의 TU들로 스플리팅될 수도 있는 예에서, 비디오 인코더 (20) 는 트리 구조에 기초하여 CU 를 CU 의 TU들로 파티셔닝할 수도 있다. 이 예에서, 트리 구조의 루트 노드는 CU 의 코딩 블록에 대응한다. 또한, 이 예에서, 트리 구조의 각각의 개별의 비-루트 노드는 개별의 비-루트 노드의 페어런트 노드에 대응하는 블록의 파티션인 개별의 블록에 대응한다. 이 예에서, 트리 구조의 리프 노드들은 CU들의 TU들에 대응한다. 이 예에서, 트리 구조의 적어도 하나의 노드의 차일드 노드들은 상이한 사이즈들의 블록들에 대응한다. 또한, 이 예에서, 비디오 인코더 (20) 는, 비디오 데이터의 인코딩된 표현을 포함하는 비트스트림에서, CU 의 TU들의 하나 이상을 나타내는 데이터를 포함할 수도 있다.

[0226] 대응하는 예에서, 비디오 디코더 (30) 는 트리 구조에 기초하여 CU 가 CU 의 TU들로 파티셔닝된다고 결정할 수도 있다. 이 예에서, 트리 구조의 루트 노드는 CU 의 코딩 블록에 대응한다. 또한, 이 예에서, 트리 구조의 각각의 개별의 비-루트 노드는 개별의 비-루트 노드의 페어런트 노드에 대응하는 블록의 파티션인 개별의 블록에 대응한다. 이 예에서, 트리 구조의 리프 노드들은 CU들의 TU들에 대응한다. 이 예에서, 트리 구조의 적어도 하나의 노드의 차일드 노드들은 상이한 사이즈들의 블록들에 대응한다. 또한, 이 예에서, 비디오 디코더 (30) 는 CU 의 TU들 중 적어도 하나에 대한 데이터에 기초하여, CU 의 코딩 블록을 복원할 수도 있다.

[0227] 본 개시물의 제 4 기법에 따르면, 변환의 스플릿은 CU 에서 동일한 방향 (수직 또는 수평) 을 따르지 않고 수행된다는 것이 허용된다. 다시 말해, CU 에 대한 변환 트리는 수평으로 스플리팅되는 변환 블록들 및 수직으로 스플리팅되는 변환 블록들을 포함할 수도 있다. 변환 블록들의 수평 및 수직 스플리팅 양자 모두를 허용하는 것은 CU 의 PU들의 경계들과 CU 의 TU들의 경계들을 더 잘 정렬시킬 수도 있다. 본 개시물의 다른 곳에서 논의된 바와 같이, CU 의 TU들의 경계들을 CU 의 PU들의 경계들과 정렬시키는 것은 예측 블록들 간의 경계들에서의 불연속성과 연관된 고 주파수 변환 계수들의 발생을 감소시키고, 따라서 압축 효율성을 증가시킬 수도 있다. 제 4 기법의 일 예에서, CU 의 TU들의 수평 및 수직 스플리팅은 소정의 파티션 모드들, 예를 들어

AMP 에만 적용 가능하다.

[0228] 도 23 은 본 개시물의 기법에 따라, N×N/4(U) 와 동일한 파티션 사이즈에 대한 변환 구조를 예시하는 개념도이다. 도 23 의 예에서, CU 파티션은 수평 방향을 따르고 TU 파티션은 수평 및/또는 수직 방향들 중 어느 하나로부터의 것일 수 있다. 특히, TU (180) 는 TU (182) 및 TU (184) 로 수평으로 스플리팅된다. TU (182) 는 TU들 (186, 188, 190, 및 192) 로 스플리팅된다. TU (184) 는 TU들 (194, 196, 및 198) 로 수평으로 그리고 수직으로 스플리팅된다.

[0229] CU 에서 상이한 방향들을 따른 변환 블록들의 스플리팅이 허용되는 예에서, 비디오 인코더 (20) 는 트리 구조에 기초하여 CU 를 CU 의 TU들로 파티셔닝할 수도 있다. 이 예에서, 트리 구조의 루트 노드는 CU 의 코딩 블록에 대응한다. 또한, 이 예에서, 트리 구조의 각각의 개별의 비-루트 노드는 개별의 비-루트 노드의 페어런트 노드에 대응하는 블록의 파티션인 개별의 블록에 대응한다. 이 예에서, 트리 구조의 리프 노드들은 CU들의 TU들에 대응한다. 이 예에서, 트리 구조에서 제 1 노드는 정확히 2 개의 차일드 노드들을 갖고, 제 1 노드의 차일드 노드들에 대응하는 블록들 간의 경계는 수직적이다. 부가적으로, 이 예에서, 트리 구조에서 제 2 노드는 정확히 2 개의 차일드 노드들을 갖고, 제 2 노드의 차일드 노드들에 대응하는 블록들 간의 경계는 수평적이다. 이 예에서, 비디오 인코더 (20) 는, 비디오 데이터의 인코딩된 표현을 포함하는 비트스트림에서, CU 의 TU들의 하나 이상을 나타내는 데이터를 포함할 수도 있다.

[0230] 유사하게, 비디오 디코더 (30) 는 트리 구조에 기초하여 CU 가 CU 의 TU들로 파티셔닝된다고 결정할 수도 있다. 이 예에서, 트리 구조의 루트 노드는 CU 의 코딩 블록에 대응하고, 트리 구조의 각각의 개별의 비-루트 노드는 개별의 비-루트 노드의 페어런트 노드에 대응하는 블록의 파티션인 개별의 블록에 대응하고, 트리 구조의 리프 노드들은 CU들의 TU들에 대응한다. 또한, 이 예에서, 트리 구조에서 제 1 노드는 정확히 2 개의 차일드 노드들을 갖고, 제 1 노드의 차일드 노드들에 대응하는 블록들 간의 경계는 수직적이다. 이 예에서, 트리 구조에서 제 2 노드는 정확히 2 개의 차일드 노드들을 갖고, 제 2 노드의 차일드 노드들에 대응하는 블록들 간의 경계는 수평적이다. 이 예에서, 비디오 디코더 (30) 는 CU 의 TU들 중 적어도 하나에 대한 데이터에 기초하여, CU 의 코딩 블록을 복원할 수도 있다.

[0231] 본 개시물의 제 5 기법에 따르면, 하나의 CU 는 다음의 설명들에서 *comb_mode* 로 지칭되는 인트라 및 인터 PU들 양자 모두를 포함할 수도 있다. 일부 경우들에서, *comb_mode* 의 사용은 CU 의 예측 블록들의 정확도를 증가시키고 따라서, 궁극적으로 증가된 압축 성능을 초래할 수도 있다. CU 의 PU 의 예측 블록의 정확도는 CU 의 코딩 블록의 샘플들과 PU 의 예측 블록의 대응하는 샘플들 간의 차이들의 측정치이다.

[0232] 따라서, 제 5 기법에 따르면, 비디오 인코더 (20) 는 인트라 예측을 수행하여 CU 의 제 1 PU 에 대한 제 1 예측 블록을 획득할 수도 있다. 부가적으로, 이 예에서, 비디오 인코더 (20) 는 인터 예측을 수행하여 동일한 CU 의 제 2 PU 에 대한 제 2 예측 블록을 획득할 수도 있다. 이 예에서, 비디오 인코더 (20) 는, 제 1 예측 블록 및 제 2 예측 블록에 기초하여, CU 에 대한 잔차 데이터를 획득할 수도 있다. 또한, 이 예에서, 비디오 인코더 (20) 는, 비디오 데이터의 인코딩된 표현을 포함하는 비트스트림에서, CU 에 대한 잔차 데이터를 나타내는 데이터를 포함할 수도 있다.

[0233] 유사하게, 제 5 기법에 따르면, 비디오 디코더 (30) 는 인트라 예측을 수행하여 CU 의 제 1 PU 에 대한 제 1 예측 블록을 획득할 수도 있다. 이 예에서, 비디오 디코더 (30) 는 인터 예측을 수행하여 동일한 CU 의 제 2 PU 에 대한 제 2 예측 블록을 획득할 수도 있다. 또한, 이 예에서, 비디오 디코더 (30) 는, 제 1 예측 블록 및 제 2 예측 블록에 기초하여, CU 의 코딩 블록을 복원할 수도 있다.

[0234] 하나의 CU 가 *comb_mode* 으로 코딩되고, CU 가 수직 방향에서 2 개의 PU들로 스플리팅되는 경우, 예컨대 N×2N 파티셔닝 모드에서, 비디오 코더는 다음과 같이 CU 의 변환 트리의 변환 심도를 결정할 수도 있다: 좌측 PU 가 인트라-코딩된 PU 이면, 변환 심도는 0 에서부터 일 수 있다. 다시 말해, 인트라-코딩되는 좌측 PU 에 기초하여, CU 의 변환 트리의 심도는 0 이상이도록 허용된다. 따라서, CU 의 변환 트리의 심도가 0 과 동일한 경우들에서, TU 사이즈는 CU 사이즈와 동일할 수 있고 하나의 TU 는 2 개의 PU들을 커버, 즉 PU 경계들을 가로지를 수도 있다. 그렇지 않으면 (좌측 PU 가 인터-코딩된 PU 이면), 변환 심도는 1 에서부터이도록 제한된다. 다시 말해, CU 의 변환 트리의 심도는 1 이상일 수도 있지만, 0 과 동일하지는 않다. 이 예에서, 좌측 PU 가 인터-코딩된 PU 인 경우, TU 사이즈는 PU 사이즈보다 크지 않아야 한다.

[0235] 따라서, 이 예에서, 비디오 인코더 (20) 는 비디오 코딩 표준을 따르는 비트스트림을 생성할 수도 있다. 이 예에서, CU 가 수직 경계를 따라 제 1 PU 및 제 2 PU 로 스플리팅되고 CU 의 좌측 PU 가 인트라-코딩된 PU 인

것에 기초하여, 비디오 코딩 표준은 제 1 및 제 2 PU들 양자 모두를 커버하도록 CU 의 TU 를 허용한다. 유사한 예에서, 비디오 디코더 (30) 는 비디오 데이터의 인코딩된 표현을 포함하는 비트스트림을 획득할 수도 있다. 이 예에서, 비트스트림은, CU 가 수직 경계를 따라 제 1 PU 및 제 2 PU 로 스플리팅되고 CU 의 좌측 PU 가 인트라-코딩된 PU 인 것에 기초하여, CU 의 TU 가 제 1 및 제 2 PU들 양자 모두를 커버하도록 허용하는 비디오 코딩 표준을 따를 수도 있다. 비디오 인코더 (20) 및 비디오 디코더 (30) 의 양자 모두의 예에서, 비디오 코딩 표준은, CU 가 수직 경계를 따라 제 1 PU 및 제 2 PU 로 스플리팅되고 CU 의 좌측 PU 가 인터-코딩된 PU 인 것에 기초하여, CU 의 TU 의 TU 사이즈가 제 1 PU 또는 제 2 PU 의 사이즈보다 크지 않다는 것을 요구하는 제한을 제공할 수도 있다.

[0236] 또한, 하나의 CU 가 수평 방향에서 2 개의 PU들로 스플리팅되는 경우, 예컨대 2N×N 파티션 모드가 사용되는 경우, 비디오 코더는 CU 가 인트라 및 인터 PU들 양자 모두를 포함하는 경우 사용된 변환 심도를 다음과 같이 결정할 수도 있다: 상부 PU 가 인트라-코딩된 PU 이면, 변환 심도는 0 에서부터 일 수 있다. 다시 말해, CU 의 변환 트리의 심도는 0 이상이다. 이 예에서, 상부 PU 는 수평으로 분할된 CU 의 상위 PU 이다. 변환 심도가 0 인 경우들에서, TU 사이즈는 CU 사이즈와 동일하고 하나의 TU 는 2 개의 PU들을 커버한다. 그렇지 않으면 (즉, 상부 PU 가 인터-코딩된 PU 이면), 변환 심도는 1 에서부터 이도록 제한된다. 다시 말해, CU 의 변환 트리의 심도는 1 이상이지만, 0 일 수 없다. 이 예에서, 상부 PU 가 인터-코딩된 PU 인 경우, TU 사이즈는 PU 사이즈보다 크지 않아야 한다.

[0237] 따라서, 이 예에서, 비디오 인코더 (20) 는 비디오 코딩 표준을 따르는 비트스트림을 생성할 수도 있다. 이 예에서, CU 가 수평 경계를 따라 제 1 PU 및 제 2 PU 로 스플리팅되고 CU 의 상부 PU 가 인트라-코딩된 PU 인 것에 기초하여, 비디오 코딩 표준은 CU 의 TU 가 제 1 및 제 2 PU들 양자 모두를 커버하도록 허용한다. 또한, 일부 예들에서, 비트스트림은, CU 가 수평 경계를 따라 제 1 PU 및 제 2 PU 로 스플리팅되고 CU 의 상부 PU 가 인터-코딩된 PU 인 것에 기초하여, CU 의 TU 의 TU 사이즈가 제 1 PU 또는 제 2 PU 의 사이즈보다 크지 않다는 것을 요구하는 제한을 제공하는 비디오 코딩 표준을 따른다.

[0238] 유사한 예에서, 비디오 디코더 (30) 는 비디오 데이터의 인코딩된 표현을 포함하는 비트스트림을 획득할 수도 있다. 이 예에서, 비트스트림은, CU 가 수평 경계를 따라 제 1 PU 및 제 2 PU 로 스플리팅되고 CU 의 상부 PU 가 인트라-코딩된 PU 인 경우, CU 의 TU 가 제 1 및 제 2 PU들 양자 모두를 커버하도록 허용하는 비디오 코딩 표준을 따른다. 또한, 일부 예들에서, 비디오 디코더 (30) 는 CU 가 수평 경계를 따라 제 1 PU 및 제 2 PU 로 스플리팅되고 CU 의 상부 PU 가 인터-코딩된 PU 인 경우, CU 의 TU 의 TU 사이즈가 제 1 PU 또는 제 2 PU 의 사이즈보다 더 이상 크지 않다는 것을 요구하는 제한을 제공하는 비디오 코딩 표준을 따르는 비트스트림을 획득한다.

[0239] 일부 예들에서, 하나의 CU 가 *comb_mode* 로 코딩되는 경우, TU 가 PU 경계들을 가로지르지 않도록 제한이 추가된다. 이 제한은, TU들이 PU 경계들을 가로지를 수 있는 경우의 레이트-왜곡 비용을 체크할 필요가 없기 때문에 인코더 복잡성을 감소시킬 수도 있다. 따라서, 이 예에서, 비디오 인코더 (20) 는, 인트라-코딩된 PU 및 인터-코딩된 PU 를 갖는 CU 에 기초하여, CU 의 어떤 TU 도 CU 의 PU 경계들을 가로지르지 않는다는 것을 요구하는 제한을 제공하는 비디오 코딩 표준을 따르는 비트스트림을 생성할 수도 있다. 유사한 예에서, 비디오 디코더 (30) 는 비디오 데이터의 인코딩된 표현을 포함하는 비트스트림을 획득할 수도 있다. 이 예에서, 비트스트림은, 인트라-코딩된 PU 및 인터-코딩된 PU 를 갖는 CU 에 기초하여, CU 의 어떤 TU 도 CU 의 PU 경계들을 가로지르지 않는다는 것을 요구하는 제한을 제공하는 비디오 코딩 표준을 따른다.

[0240] 또한, *comb_mode* 를 수반하는 일부 예들에서, *comb_mode* 는 8×8 과 같은 소정 사이즈보다 큰 (이를 포함하지 않는) CU 에 대해서만 적용된다고 제한된다. 더 작은 블록들에 대해, *comb_mode* 가 CU 에 적용되는지 여부를 시그널링하는 비트들을 증가시키는 것은 *comb_mode* 에 의해 도입된 절감된 레이트-왜곡 비용을 보상하지 않을 수도 있다. 따라서, 소정의 작은 사이즈들에 대해, *comb_mode* 는 추가적인 시그널링 없이 항상 디스에이블될 수도 있다. 본 개시물에서, 제한은 비디오 인코더가 일부 방식으로 일부 액션을 수행하고 또는 비트스트림을 생성하는 것을 방지할 수도 있다. 예를 들어, 비디오 인코더 (20) 는, 특정 사이즈보다 더 작은 어떤 CU 도 인트라-코딩된 PU 및 인터-코딩된 PU 양자 모두를 갖도록 허용되지 않는다는 것을 요구하는 제한을 제공하는 비디오 코딩 표준을 따르는 비트스트림을 생성할 수도 있다. 유사한 예에서, 비디오 디코더 (30) 는 비디오 데이터의 인코딩된 표현을 포함하는 비트스트림을 획득할 수도 있다. 이 예에서, 비트스트림은, 특정 사이즈보다 더 작은 어떤 CU 도 인트라-코딩된 PU 및 인터-코딩된 PU 양자 모두를 갖도록 허용되지 않는다는 것을 요구하는 제한을 제공하는 비디오 코딩 표준을 따른다.

[0241] *comb_mode* 를 수반하는 일부 예들에서, 하나의 8×8 CU 는 하나의 8×4 인트라 및 하나의 8×4 인터 PU, 또는 하나의 4×8 인트라 및 하나의 4×8 인터 PU 를 갖는 *comb_mode* 로 코딩될 수 있다. 이 예에서, (4:2:0 컬러 포맷에서) 이 8×8 CU 의 대응하는 4×4 크로마 블록은 인터 코딩된 루마 PU 의 인터 예측 모드만을 사용하여, 또는 인트라 코딩된 루마 PU 의 인트라 예측 모드만을 사용하여 코딩될 수 있고, 또는 4×4 크로마 블록은 또한, 루마 PU 파티션에 기초하여 2 개의 4×2 또는 2×4 블록들로서 상응하여 파티셔닝되며, 2 개의 4×2 또는 2×4 의 각각은 대응하는 루마 예측 모드에 의해 예측되며, 4×4 잔차 블록이 생성되고 생성된 44 개의 잔차 블록 상에 4×4 변환이 수행되어 2×2 변환의 도입을 회피한다. 2×2 변환의 도입은 복잡성을 불필요하게 증가시킬 수도 있다.

[0242] 따라서, 상기의 예에서, 비디오 인코더 (20) 는 인트라 예측을 수행하여 비디오 데이터의 CU 의 제 1 PU 에 대한 제 1 예측 블록을 획득할 수도 있다. 부가적으로, 이 예에서, 비디오 인코더 (20) 는 인터 예측을 수행하여 동일한 CU 의 제 2 PU 에 대한 제 2 예측 블록을 획득할 수도 있다. 이 예에서, CU 의 사이즈는 2N×2N 이고, 제 1 PU 의 사이즈는 2N×N 이고, 제 2 PU 의 사이즈는 N×2N 이고 또는 제 1 PU 의 사이즈는 N×2N 이며 제 2 PU 의 사이즈는 2N×N 이다. 또한, 이 예에서, CU 는 4:2:0 컬러 포맷을 사용하여 코딩된다. 이 예에서, 제 1 PU 에 대한 제 1 예측 블록은 제 1 PU 에 대한 루마 예측 블록이다. 이 예에서, 제 2 PU 에 대한 제 2 예측 블록은 제 2 PU 에 대한 루마 예측 블록이다. 이 예에서, 비디오 인코더 (20) 는 인터 예측만을 사용하여 제 3 예측 블록을 획득하고, 제 3 예측 블록은 사이즈 N×N 의 크로마 예측 블록이다. 이 예에서, 비디오 인코더 (20) 는 제 1, 제 2, 및 제 3 예측 블록들에 기초하여 CU 에 대한 잔차 데이터를 획득한다. 제 1, 제 2, 및 제 3 예측 블록들에 기초하여 CU 에 대한 잔차 데이터를 획득하는 비디오 인코더 (20) 대신에, 유사한 예는 인트라 예측만을 사용하여 인트라 예측 대신에 제 3 예측 블록을 획득하는 비디오 인코더 (20) 를 대체한다.

[0243] 더욱이, 대응하는 예에서, 비디오 디코더 (30) 는 인트라 예측을 수행하여 비디오 데이터의 CU 의 제 1 PU 에 대한 제 1 예측 블록을 획득할 수도 있다. 이 예에서, 비디오 디코더 (30) 는 인터 예측을 수행하여 동일한 CU 의 제 2 PU 에 대한 제 2 예측 블록을 획득할 수도 있다. 이 예에서, CU 의 사이즈는 2N×2N 이고, 제 1 PU 의 사이즈는 2N×N 이고, 제 2 PU 의 사이즈는 N×2N 이며 또는 제 1 PU 의 사이즈는 N×2N 이며 제 2 PU 의 사이즈는 2N×N 이고, CU 는 4:2:0 컬러 포맷을 사용하여 코딩된다. 또한, 이 예에서, 제 1 PU 에 대한 제 1 예측 블록은 제 1 PU 에 대한 루마 예측 블록이고, 제 2 PU 에 대한 제 2 예측 블록은 제 2 PU 에 대한 루마 예측 블록이다. 이 예에서, 비디오 디코더 (30) 는 인터 예측만을 사용하여 제 3 예측 블록을 획득하고, 제 3 예측 블록은 사이즈 N×N 의 크로마 예측 블록이다. 또한, 이 예에서, 비디오 디코더 (30) 는, 제 1, 제 2, 및 제 3 예측 블록들에 기초하여, CU 의 코딩 블록을 복원할 수도 있다. 유사한 예는, 인트라 예측 대신에 제 3 예측 블록을 획득하기 위해 인트라 예측만을 사용하는 비디오 디코더 (30) 의 상이한 구성을 대체한다.

[0244] 위에서 언급된 바와 같이, 일부 예들에서, 하나의 8×8 CU 는 하나의 8×4 인트라 및 하나의 8×4 인터 PU, 또는 하나의 4×8 인트라 및 하나의 4×8 인터 PU 를 갖고 *comb_mode* 로 코딩될 수 있고, 이 8×8 CU 의 대응하는 4×4 크로마 블록은 인터 코딩된 루마 PU 의 인터 예측 모드만을 사용하여 코딩될 수 있고, 비디오 코더는 루마 PU 파티션에 기초하여 4×4 크로마 블록을 상응하여 2 개의 4×2 또는 2×4 블록들로서 파티셔닝할 수도 있고, 비디오 코더는 대응하는 루마 예측 모드에 의해 2 개의 4×2 또는 2×4 의 각각을 예측하며, 비디오 코더는 4×4 잔차 블록을 생성하고 생성된 4×4 잔차 블록 상에서 4×4 변환을 수행한다.

[0245] 따라서, 이러한 예에서, 비디오 인코더 (20) 는 인트라 예측을 수행하여 비디오 데이터의 CU 의 제 1 PU 에 대한 제 1 예측 블록을 획득할 수도 있다. 부가적으로, 이 예에서, 비디오 인코더 (20) 는 인터 예측을 수행하여 동일한 CU 의 제 2 PU 에 대한 제 2 예측 블록을 획득할 수도 있다. 이 예에서, CU 의 사이즈는 2N×2N 이고, 제 1 PU 의 사이즈는 2N×N 이고, 제 2 PU 의 사이즈는 N×2N 이며 또는 제 1 PU 의 사이즈는 N×2N 이며 제 2 PU 의 사이즈는 2N×N 이고, CU 는 4:2:0 컬러 포맷을 사용하여 코딩된다. 또한, 제 1 PU 에 대한 제 1 예측 블록은 제 1 PU 에 대한 루마 예측 블록이고, 제 2 PU 에 대한 제 2 예측 블록은 제 2 PU 에 대한 루마 예측 블록이다. 이 예에서, 비디오 인코더 (20) 는 제 1 PU 의 인트라 예측 모드를 사용하여 제 1 PU 에 대한 크로마 예측 블록들을 생성할 수도 있다. 또한, 이 예에서, 비디오 인코더 (20) 는 인터 예측을 사용하여 제 2 PU 에 대한 크로마 예측 블록을 생성할 수도 있다. 이 예에서, 비디오 인코더 (20) 는, 제 1 예측 블록, 제 2 예측 블록들, 제 1 PU 에 대한 크로마 예측 블록 및 제 2 PU 에 대한 크로마 예측 블록에 기초하여 CU 에 대한 잔차 데이터를 획득할 수도 있다.

[0246] 유사한 예에서, 비디오 디코더 (30) 는 인트라 예측을 수행하여 비디오 데이터의 CU 의 제 1 PU 에 대한 제 1

예측 블록을 획득할 수도 있다. 이 예에서, 비디오 디코더 (30) 는 인터 예측을 수행하여 동일한 CU 의 제 2 PU 에 대한 제 2 예측 블록을 획득할 수도 있다. 이 예에서, CU 의 사이즈는 2N×2N 이고, 제 1 PU 의 사이즈는 2N×N 이고 제 2 PU 의 사이즈는 N×2N 이며 또는 제 1 PU 의 사이즈는 N×2N 이고 제 2 PU 의 사이즈는 2N×N 이며, CU 는 4:2:0 컬러 포맷을 사용하여 코딩된다. 또한, 이 예에서, 제 1 PU 에 대한 제 1 예측 블록은 제 1 PU 에 대한 루마 예측 블록이고, 제 2 PU 에 대한 제 2 예측 블록은 제 2 PU 에 대한 루마 예측 블록이다. 이 예에서, 비디오 디코더 (30) 는 제 1 PU 의 인트라 예측 모드를 사용하여 제 1 PU 에 대한 크로마 예측 블록을 생성할 수도 있다. 비디오 디코더 (30) 는 인터 예측을 사용하여 제 2 PU 에 대한 크로마 예측 블록을 생성할 수도 있다. 또한, 비디오 디코더 (30) 는 제 1 예측 블록, 제 2 예측 블록들, 제 1 PU 에 대한 크로마 예측 블록 및 제 2 PU 에 대한 크로마 예측 블록에 기초하여 CU 의 코딩 블록을 복원할 수도 있다.

[0247] 부가적으로, *comb_mode* 를 수반하는 일부 예들에서, 하나의 CU 가 2 이상의 PU들을 사용하여 코딩되고 인터 및 인트라 예측 모드들 양자 모두가 사용되는 경우, 인터-코딩된 PU들에 대해, CU 는 현재 HEVC 설계와 동일한 방식으로 처리된다. 즉, 가능한 역 양자화/변환 및 그 모션 정보를 사용하는 모션-보상된 예측 블록 후에 디코딩된 잔차의 합계로서 복원이 정의된다. 또한, 인트라-코딩된 PU들에 대해, 비디오 코더는 2 개의 예측자들을 수반하는 프로세스를 사용한다, 즉 복원은 가능한 역 양자화/변환 및 그 이웃하는 인터-코딩된 PU 로부터의 모션 정보를 사용하는 모션-보상된 예측 블록 및 현재 PU 와 연관된 인트라 예측 모드들을 사용하는 인트라 예측 블록 후에 디코딩된 잔차의 합계로서 정의된다.

[0248] 따라서, 이 예에서, 비디오 인코더 (20) 는 인트라 예측을 수행하여 비디오 데이터의 CU 의 제 1 PU 에 대한 제 1 예측 블록을 획득할 수도 있다. 이 예에서, 비디오 인코더 (20) 는 인터 예측을 수행하여 동일한 CU 의 제 2 PU 에 대한 제 2 예측 블록을 획득할 수도 있다. 또한, 이 예에서 CU 에 대한 잔차 데이터를 획득하는 것의 부분으로서, 비디오 인코더 (20) 는, 제 1 PU 에 대응하는 잔차 데이터의 각각의 개별의 샘플에 대해, 개별의 샘플이 CU 의 코딩 블록의 개별의 샘플 마이너스 제 2 PU 의 모션 정보를 사용하여 획득된 예측 샘플 마이너스 제 1 예측 블록의 샘플과 동일하도록 개별의 샘플을 획득할 수도 있다. 모션 정보를 사용하여 획득된 예측 샘플은 인터-예측 PU 의 예측 블록의 샘플일 수도 있다.

[0249] 대응하는 예에서, 비디오 디코더 (30) 는 인트라 예측을 수행하여 비디오 데이터의 CU 의 제 1 PU 에 대한 제 1 예측 블록을 획득할 수도 있다. 이 예에서, 비디오 디코더 (30) 는 인터 예측을 수행하여 동일한 CU 의 제 2 PU 에 대한 제 2 예측 블록을 획득할 수도 있다. 또한, 이 예에서 CU 의 코딩 블록을 복원하는 것의 부분으로서, 비디오 디코더 (30) 는, 제 1 PU 에 대응하는 코딩 블록의 각각의 개별의 샘플에 대해, 개별의 샘플이 개별의 디코딩된 잔차 샘플, 제 2 PU 의 모션 정보를 사용하여 획득된 예측 샘플, 및 제 1 예측 블록의 샘플의 합계와 동일하도록 개별의 샘플을 획득할 수도 있다.

[0250] 대안으로, *comb_mode* 를 수반하는 일부 예들에서, 하나의 CU 가 2 이상의 PU들을 사용하여 코딩되고 인터 및 인트라 예측 모드들 양자 모두가 사용되는 경우, 인트라-코딩된 PU들에 대해, CU 의 코딩 블록들을 복원하기 위한 프로세스는 현재의 HEVC 설계와 동일하다, 즉 복원은 가능한 역 양자화/변환 및 그 인트라 예측 모드를 사용하는 인트라 예측 블록 후의 디코딩된 잔차의 합계로서 정의된다. 또한, CU 의 인터-코딩된 PU 에 대해, 인터-코딩된 PU 에 대응하는 코딩 블록들의 일부들을 복원하는 프로세스는, 2 개의 예측자들이 인터-코딩된 PU 에 대해 정의된다는 점에서 HEVC 에서의 복원 프로세스와 상이하다. 또한, 인터-코딩된 PU 의 샘플에 대응하는 CU 의 코딩 블록의 각각의 샘플에 대해, 샘플은 (예를 들어, 가능한 역 양자화/변환 후의) 디코딩된 잔차 샘플 및 인터-코딩된 PU 의 모션 정보를 사용하여 생성된 인터-코딩된 PU 의 모션-보상된 예측 블록의 샘플 및 인터-코딩된 PU 를 이웃하는 인트라-코딩된 PU 와 연관된 인트라 예측 모드를 사용하여 생성된 인트라 예측 블록의 샘플의 합계로서 정의된다.

[0251] 따라서, 이 예에서, 비디오 인코더 (20) 는 인트라 예측을 수행하여 비디오 데이터의 CU 의 제 1 PU 에 대한 제 1 예측 블록을 획득할 수도 있다. 이 예에서, 비디오 인코더 (20) 는 인터 예측을 수행하여 동일한 CU 의 제 2 PU 에 대한 제 2 예측 블록을 획득할 수도 있다. 또한, 이 예에서 CU 에 대한 잔차 데이터를 획득하는 것의 부분으로서, 비디오 인코더 (20) 는, 제 2 PU 에 대응하는 잔차 데이터의 각각의 개별의 샘플에 대해, 개별의 샘플이 CU 의 코딩 블록의 개별의 샘플 마이너스 제 1 PU 의 인트라 예측 모드를 사용하여 획득된 예측 샘플 마이너스 제 2 예측 블록의 샘플과 동일하도록 개별의 샘플을 획득할 수도 있다.

[0252] 대응하는 예에서, 비디오 디코더 (30) 는 인트라 예측을 수행하여 비디오 데이터의 CU 의 제 1 PU 에 대한 제 1 예측 블록을 획득할 수도 있다. 이 예에서, 비디오 디코더 (30) 는 인터 예측을 수행하여 동일한 CU 의 제

2 PU 에 대한 제 2 예측 블록을 획득할 수도 있다. 또한, 이 예에서 CU 의 코딩 블록을 복원하는 것의 부분으로서, 비디오 디코더 (30) 는, 제 2 PU 에 대응하는 코딩 블록의 각각의 개별의 샘플에 대해, 개별의 샘플이 개별의 디코딩된 잔차 샘플, 제 1 PU 의 인트라 예측 모드를 사용하여 획득된 예측 샘플, 및 제 2 예측 블록의 샘플의 합계와 동일하도록 개별의 샘플을 획득할 수도 있다.

[0253] 일 예에서, 2 개의 예측자들을 허용하는 경우, 2 개의 예측 블록들은 선형 가중화 함수, 예를 들어 2 의 평균과 결합된다. 예를 들어, 비디오 인코더 (20) 또는 비디오 디코더 (30) 와 같은 비디오 코더는 인트라 예측을 사용하여 PU 의 제 1 예측 블록을 생성할 수도 있고, 인터 예측을 사용하여 PU 에 대한 제 2 예측 블록을 생성할 수도 있다. 이 예에서, 비디오 코더는, 최종 예측 블록의 각각의 개별의 샘플에 대해, 최종 예측 블록의 개별의 샘플에 대응하는 제 1 및 제 2 예측 블록들의 샘플들의 가중화된 평균을 결정함으로써 PU 에 대한 최종 예측 블록을 결정할 수도 있다. 이 예에서, 가중화된 평균에서 사용된 가중치들은 인터 예측된 예측 블록에 비해 인트라 예측된 예측 블록을 선호할 수도 있고, 또는 그 반대일 수도 있다. 일부 경우들에서, 이러한 선형 가중화 함수를 사용하는 것은 더 정확한 최종 예측 블록을 초래할 수도 있고, 이것은 궁극적으로 압축 성능을 증가시킬 수도 있다. 선형 가중화 팩터들은 사이드 정보로서 시그널링되거나 또는 소정의 코딩된 정보로부터 도출될 수도 있다.

[0254] 예를 들어, CU 의 PU 에 대응하는 CU 의 코딩 블록의 각각의 개별의 샘플에 대해, 비디오 인코더 (20) 는 개별의 샘플에 대한 제 1 예측 샘플 및 개별의 샘플에 대한 제 2 예측 샘플을 획득할 수도 있다. 예를 들어, 개별의 샘플에 대한 제 1 예측 샘플은 인터 예측을 사용하여 생성될 수도 있고, 개별의 샘플에 대한 제 2 예측 샘플은 인트라 예측을 사용하여 생성될 수도 있다. 이 예에서, 비디오 인코더 (20) 는 개별의 샘플에 대한 제 1 예측 샘플 및 개별의 샘플에 대한 제 2 예측 샘플에 선형 가중화 함수를 적용함으로써 개별의 샘플에 대한 가중화된 예측 샘플을 결정할 수도 있다. 부가적으로, 이 예에서, 비디오 인코더 (20) 는 개별의 샘플에 대한 잔차 샘플이 개별의 샘플의 원래의 값과 개별의 샘플의 가중화된 예측 샘플 간의 차이와 동일하다고 결정할 수도 있다.

[0255] 유사하게, 코딩 블록의 각각의 개별의 샘플에 대해, 그 비디오 디코더 (30) 는 개별의 샘플에 대한 잔차 샘플을 획득할 수도 있다. 예를 들어, 비디오 디코더 (30) 는, 비트스트림으로부터, 변환 계수들을 나타내는 신택스 엘리먼트들을 획득하고, 변환 계수들에 역 양자화를 적용하며, 변환 계수들에 역 변환을 적용하여 잔차 샘플들을 획득할 수도 있다. 또한, 이 예에서, 비디오 디코더 (30) 는 개별의 샘플에 대한 제 1 예측 샘플 및 개별의 샘플에 대한 제 2 예측 샘플을 결정할 수도 있다. 예를 들어, 개별의 샘플에 대한 제 1 예측 샘플은 인터 예측을 사용하여 생성될 수도 있고, 개별의 샘플에 대한 제 2 예측 샘플은 인트라 예측을 사용하여 생성될 수도 있다. 이 예에서, 비디오 디코더 (30) 는 개별의 샘플에 대한 제 1 예측 샘플 및 개별의 샘플에 대한 제 2 예측 샘플에 선형 가중화 함수를 적용함으로써 개별의 샘플에 대한 가중화된 예측 샘플을 결정할 수도 있다. 부가적으로, 이 예에서, 비디오 디코더 (30) 는 개별의 샘플에 대한 가중화된 예측 샘플 및 개별의 샘플에 대한 잔차 샘플의 합계로서 개별의 샘플을 복원할 수도 있다.

[0256] 다음의 예들은, *comb_mode* 가 하나의 슬라이스, 픽처, 또는 시퀀스 (즉, 코딩된 비디오 시퀀스) 에 대해 인에이블되는 경우 *comb_mode* 의 사용을 나타낸다. HEVC 에서, CU 는 1-비트 pred_mode_flag 신택스 엘리먼트를 포함한다. 0 과 동일한 CU 의 pred_mode_flag 신택스 엘리먼트는, CU 가 인터 예측 모드에서 코딩된다는 것을 지정한다. 1 과 동일한 CU 의 pred_mode_flag 신택스 엘리먼트는, CU 가 인트라 예측 모드에서 코딩된다는 것을 지정한다. 본 개시물의 하나의 예에 따르면, CU 의 1-비트 pred_mode_flag 는 3 개의 가능한 값들로 신택스 엘리먼트에 의해 대체된다. 이 예에서, 3 개의 값들은 종래의 인트라 모드, 종래의 인터 모드, 및 *comb_mode* 에 각각 대응한다. 이 예에서, 종래의 인트라 모드는, CU 의 모든 PU들이 인트라 예측 모드를 사용하여 코딩되는 경우들을 지칭한다. 또한, 이 예에서, 종래의 인터 예측 모드는, CU 의 모든 PU들이 인터 예측 모드를 사용하여 코딩되는 경우들을 지칭한다. 일부 예들에서, *comb_mode* 가 CU 에 대해 인에이블되는 경우, CU 의 단지 하나의 PU 에 대해, 비디오 인코더 (20) 는, PU 가 인트라 예측 모드 또는 인터 예측 모드에서 코딩되는지 여부를 나타내기 위해 1-비트 값을 시그널링한다. CU 가 *comb_mode* 에서 코딩되기 때문에, 다른 PU 는 1-비트 값이 시그널링되었던 PU 와는 상이한 예측 모드이다. 다른 예에서, *comb_mode* 는 종래의 인터 모드로서 취급된다. 이 예에서, CU 의 각각의 PU 에 대해, 추가적인 플래그가 추가되어 인트라 또는 인터 예측 모드의 사용을 나타낸다.

[0257] 본 개시물의 제 7 기법에서, 하나의 PU 는 인트라 예측 및 인터 예측 양자 모두로부터 예측될 수 있고, 인트라 예측 및 인터 예측으로부터의 2 개의 예측 블록들이 사용되어 PU 에 대한 최종 예측 블록을 도출한다. 이 방식으로 최종 예측 블록을 도출하는 것은 PU 에 대한 더 정확한 예측 블록을 초래할 수도 있고, 이것은 압축

성능을 증가시킬 수도 있다.

[0258] 따라서, 제 7 기법에 따르면, 비디오 인코더 (20) 는 인트라 예측을 수행하여 CU 의 PU 에 대한 제 1 예측 블록을 획득할 수도 있다. 부가적으로, 이 예에서, 비디오 인코더 (20) 는 인터 예측을 수행하여 동일한 CU 의 동일한 PU 에 대한 제 2 예측 블록을 획득할 수도 있다. 이 예에서, 비디오 인코더 (20) 는, 제 1 예측 블록 및 제 2 예측 블록에 기초하여, PU 에 대한 최종 예측 블록을 도출할 수도 있다. 또한, 비디오 인코더 (20) 는, PU 에 대한 최종 예측 블록에 기초하여, CU 에 대한 잔차 데이터를 획득할 수도 있다. 예를 들어, 비디오 인코더 (20) 는 PU 에 대한 최종 예측 블록의 샘플들과 CU 의 코딩 블록의 대응하는 샘플들 간의 차이들을 계산함으로써 CU 에 대한 잔차 데이터의 적어도 일부를 생성할 수도 있다. 이 예에서, 비디오 인코더 (20) 는, 비디오 데이터의 인코딩된 표현을 포함하는 비트스트림에서, CU 에 대한 잔차 데이터를 나타내는 데이터를 포함할 수도 있다.

[0259] 대응하는 예에서, 비디오 디코더 (30) 는 인트라 예측을 수행하여 CU 의 PU 에 대한 제 1 예측 블록을 획득할 수도 있다. 부가적으로, 이 예에서, 비디오 디코더 (30) 는 인터 예측을 수행하여 동일한 CU 의 동일한 PU 에 대한 제 2 예측 블록을 획득할 수도 있다. 이 예에서, 비디오 디코더 (30) 는, 제 1 예측 블록 및 제 2 예측 블록에 기초하여, PU 에 대한 최종 예측 블록을 도출할 수도 있다. 또한, 이 예에서, 비디오 디코더 (30) 는, PU 에 대한 최종 예측 블록에 기초하여, CU 의 코딩 블록을 복원할 수도 있다. 예를 들어, 비디오 디코더 (30) 는 PU 에 대한 최종 예측 블록을 CU 에 대한 잔차 데이터에 추가하여, CU 의 코딩 블록의 적어도 일부를 복원할 수도 있다.

[0260] 또한, 제 7 기법의 일부 예들에서, 비디오 코더는 선형 가중화 함수를 2 개의 예측 블록들에 적용한다, 예를 들어 2 개의 예측 블록들의 동일한 상대적인 포지션들에 위치된 픽셀들의 가중화 팩터들은 고정된다. 일부 예들에서, 상이한 포지션들에 대한 가중화 팩터들은 가변적일 수도 있다. 또한, 일부 예들에서, 가중화 팩터들은 인트라 예측 모드에 의존한다. 일 예에서, 블록 내의 상단-좌측 포지션에 대해, 인트라 예측 모드가 DC 모드이면, 인터 및 인트라 예측 블록들에서의 상단-좌측 샘플들의 가중치들은 동일한, 즉 (0.5, 0.5) 인 한편, 인트라 예측이 수직 예측 모드이면, 인트라 예측 블록에서의 상단-좌측 샘플의 가중치는 인터 예측 블록에서 상단-좌측 샘플의 것보다 더 클 수도 있다.

[0261] 제 7 기법의 일부 예들에서, 하나 이상의 포지션들에 있지만, 모든 포지션들은 아닌 포지션들에서의 하나의 픽셀의 최종 예측 값은 인트라 예측된 블록 또는 인터 예측된 블록 중 어느 하나로부터 복사될 수도 있다, 즉 2 개의 가중화 팩터들 중 하나는 0 이고 다른 하나는 1 이다.

[0262] 일부 예들에서, 제 7 기법은 특정 파티션 사이즈들 (예를 들어, 2N×2N), 및/또는 특정 예측 모드들 (MERGE/SKIP 모드) 에 적용된다. 또한, 일부 예들에서, 제 7 기법이 적용되는 경우, 인트라 예측 모드들은 종래의 인트라 예측에 대해 사용된 인트라 예측 모드들의 서브세트이도록 제한된다. 일 예에서, 서브세트는 단지 MPM (most probable mode)들을 포함하도록 정의된다.

[0263] 위에서 논의된 바와 같이, 비디오 코더는 동일한 블록의 복원된 루마 샘플들에 기초하여 블록의 크로마 샘플들을 예측하도록 선형 모델 (LM) 예측 모드를 사용할 수도 있다. 또한, 전술된 바와 같이, LM 예측 모드는 비-스퀘어 PU들과 사용되지 않았다. 본 개시물의 제 8 기법에 따르면, 비디오 코더는 비-스퀘어 PU들과 LM 예측 모드를 사용할 수도 있다. 보다 일반적으로, LM 예측 모드를 적용하기 위한 동일한 기법들은 비-스퀘어 루마 및 크로마 블록들과 작업한다. 따라서, 비-스퀘어 PU들에 대하여 제 8 기법에 관한 본 개시물에서의 논의는 PU들의 루마 예측 블록들 및 크로마 예측 블록들과 같은, 비-스퀘어 루마 및 크로마 블록들에 더 일반적으로 적용할 수도 있다. 또한, 제 8 기법의 예들은 여러 방식들로 LM 예측 모드에서 사용된 파라미터들을 도출할 수도 있다.

[0264] 예를 들어, 제 8 기법의 일부 예들에서, 비-스퀘어 PU 의 더 긴 사이드에서의 경계는, 다운-샘플링된 또는 서브-샘플링된 경계에서의 픽셀들의 수가 더 짧은 경계에서의 픽셀들의 수와 동일하도록 다운-샘플링 또는 서브-샘플링된다. 프로세스는 데시메이션 또는 보간된 샘플링일 수 있다. 비디오 디코더 (30) 가 데시메이션을 사용하여 서브-샘플링을 수행하는 예들에서, 비디오 디코더 (30) 는 규칙적인 인터벌들로 (예를 들어, 2 샘플들마다 한번씩) 샘플들을 제거하여 나머지 샘플들의 값을 변화시키지 않고 샘플들의 수를 감소시킬 수도 있다.
다른 예에서, 비디오 디코더 (30) 는 보간을 사용하여 서브-샘플링을 수행할 수도 있다. 비디오 디코더 (30) 가 보간을 사용하여 서브-샘플링을 수행하는 예들에서, 인접한 샘플들의 개별의 쌍들에 대해, 비디오 디코더 (30) 는 개별의 쌍의 샘플들 간의 값을 보간할 수도 있고 샘플들의 서브-샘플링된 세트에서 보간된 값을 포함할 수도 있다.

[0265] 따라서, 본 개시물의 제 8 기법의 예에서, 비디오 인코더 (20) 는 선형 모델 예측 동작을 수행하여 PU 의 다운-샘플링된 또는 서브-샘플링된 복원된 루마 샘플들로부터 CU 의 비-스퀘어 PU 에 대한 예측 크로마 블록을 예측할 수도 있다. 또한, 이 예에서, 비디오 인코더 (20) 는, 예측 크로마 블록에 기초하여, CU 에 대한 잔차 데이터를 획득할 수도 있다. 이 예에서, 비디오 인코더 (20) 는, 비디오 데이터의 인코딩된 표현을 포함하는 비트스트림에서, CU 에 대한 잔차 데이터를 나타내는 데이터를 포함할 수도 있다. 대응하는 예에서, 비디오 디코더 (30) 는 선형 모델 예측 동작을 수행하여 PU 의 다운-샘플링된 복원된 루마 샘플들로부터 비디오 데이터의 현재 픽처의 CU 의 비-스퀘어 PU 에 대한 예측 크로마 블록을 예측할 수도 있다. 이 예에서, 비디오 디코더 (30) 는, 예측 크로마 블록에 적어도 부분적으로 기초하여 CU 에 대한 코딩 블록을 복원할 수도 있다. 단락의 예들 중 어느 하나에서, 비디오 인코더 (20) 또는 비디오 디코더 (30) 는, 비-스퀘어 PU 의 더 긴 사이드 상의 다운-샘플링된 또는 서브-샘플링된 루마 샘플들의 수가 비-스퀘어 PU 의 더 짧은 사이드 상의 루마 샘플들과 동일하도록 비-스퀘어 PU 의 더 긴 사이드의 루마 샘플들을 다운-샘플링 또는 서브-샘플링할 수도 있다.

[0266] 일 예에서, 선형 모델 파라미터들을 계산하기 위해 등식 (4) 및 등식 (5) 를 사용하는 경우, 루마 및 크로마 컴포넌트들 양자 모두에 대해, 비-스퀘어 PU 의 더 긴 사이드에서 경계의 픽셀들은, 다운-샘플링된 또는 서브-샘플링된 경계에서의 픽셀들의 수가 더 짧은 경계 (즉, min(K, L)) 에서 픽셀들의 수와 동일하도록 서브-샘플링된다. 서브-샘플링 프로세스는 데시메이션 또는 보간된 샘플링일 수 있다.

[0267] 따라서, 이 예에서, LM 예측 동작을 수행하는 것의 부분으로서, 비디오 코더는 예측 크로마 샘플이 제 1 파라미터 곱하기 병치된 루마 샘플, 플러스 제 2 파라미터와 동일하도록 예측 크로마 샘플을 획득할 수도 있고, 여기서 제 1 파라미터는 다음과 동일하고:

[0268] $$\alpha = \frac{I\sum x_i \cdot y_i - \sum x_i \cdot \sum y_i}{I\sum x_i \cdot x_i - \sum x_i \cdot \sum x_i}$$

[0269] 제 2 파라미터는 다음과 동일하다:

[0270] $$\beta = \left(\sum y_i - \alpha \cdot \sum x_i\right) / I$$

[0271] 여기서, I 은 비-스퀘어 PU 의 좌측 및 상단 경계에서의 레퍼런스 샘플들의 수이고, x_i 는 다운-샘플링된 또는 서브-샘플링된 복원된 루마 레퍼런스 샘플이며, y_i 는 복원된 크로마 레퍼런스 샘플이다.

[0272] 대안으로, 일부 예들에서, PU 의 더 긴 및 더 짧은 사이드들에 위치된 픽셀들은 서브-샘플링될 수도 있고, 서브-샘플링 비들은 상이할 수도 있다. 서브-샘플링 비는, 서브-샘플링 후의 샘플들에 대한 서브-샘플링 이전의 샘플들의 비이다. 그러나, 서브-샘플링 후의 2 개의 사이드들에서 픽셀들의 총 수는 2^m 과 동일해야 한다는 것이 요구될 수도 있다 (여기서, m 은 정수이고, m 은 루마 및 크로마 컴포넌트들에 대해 상이할 수도 있다). m 의 값은 K 및 L 의 블록 사이즈에 의존적일 수 있다.

[0273] 따라서, 이 예에서, LM 예측 동작을 수행하는 것의 부분으로서, 비디오 코더는 예측 크로마 샘플이 제 1 파라미터 곱하기 병치된 루마 샘플, 플러스 제 2 파라미터와 동일하도록 예측 크로마 샘플을 획득할 수도 있고, 여기서 제 1 파라미터는 다음과 동일하고:

[0274] $$\alpha = \frac{I\sum x_i \cdot y_i - \sum x_i \cdot \sum y_i}{I\sum x_i \cdot x_i - \sum x_i \cdot \sum x_i}$$

[0275] 제 2 파라미터는 다음과 동일하다:

[0276] $$\beta = \left(\sum y_i - \alpha \cdot \sum x_i\right) / I$$

[0277] 여기서, I 는 레퍼런스 샘플들의 세트에서의 레퍼런스 샘플들의 수이고, x_i 는 복원된 루마 레퍼런스 샘플이며, y_i 는 복원된 크로마 레퍼런스 샘플이다. 이 예에서, 레퍼런스 샘플들의 세트는 좌측 레퍼런스 샘플들 및 상부 레퍼런스 샘플들의 서브-샘플링된 세트이고, 좌측 레퍼런스 샘플들은 현재 PU 의 좌측 경계의 바로 좌측에 있고, 상부 레퍼런스 샘플들은 현재 PU 의 상단 경계 바로 위에 있다.

[0278] 제 8 기법의 다른 예에서, 등식들 (4) 및 (5) 에서 픽셀들의 수 (*I*) 는 경계에서의 픽셀들의 실제 수에 기초하여 조정된다. 예를 들어, 2N×N PU 에 대해, *I*=3N 이다. 단지 좌측 또는 상부의 인과관계 샘플들이 이용 가능한 경우, 총 수반된 샘플들의 수 (*I*) 는 좌측 또는 상부 경계의 길이와 동일하다. 따라서, 비디오 코더는 다음과 같이 α 를 계산할 수도 있다:

[0279] $$\alpha = \frac{3N\sum x_i \cdot y_i - \sum x_i \cdot \sum y_i}{3N\sum x_i \cdot x_i - \sum x_i \cdot \sum x_i}$$

[0280] 부가적으로, 비디오 코더는 다음과 같이 β 를 계산할 수도 있다:

[0281] $$\beta = \left(\sum y_i - \alpha \cdot \sum x_i\right)/3N$$

[0282] LM 이 하나의 비-스퀘어 크로마 PU 에 대해 인에이블되는 경우 (K×L 과 동일한 사이즈를 갖고, 여기서 K 는 는 L 과 동일하지 않음), 파라미터들 (즉, a 및 b) 은 다양한 방식들로 도출될 수 있다. 예를 들어, 선형 모델 파라미터들을 계산하기 위해 등식 (4) 및 등식 (5) 를 사용하는 경우, 루마 및 크로마 컴포넌트들 양자 모두에 대해, 비-스퀘어 PU 의 더 짧은 사이드에서 경계의 픽셀들은, 업-샘플링된 경계에서의 픽셀들의 수가 더 긴 경계 (즉, max(K, L)) 에서 픽셀들의 수와 동일하도록 업-샘플링된다. 업-샘플링 프로세스는 듀플리케이터 (duplicator) 또는 보간된 샘플링일 수 있다. 듀플리케이터 업-샘플링 프로세스는, 기존의 샘플들이 복제되어 새로운 샘플들을 생성하는 업-샘플링 프로세스이다. 보간된 업-샘플링 프로세스는 2 이상의 기존의 샘플들에 기초하여 새로운 샘플의 값을 보간함으로써 샘플들의 수를 증가시킨다.

[0283] 따라서, 이 예에서, LM 예측 동작을 수행하는 것의 부분으로서, 비디오 코더는, 예측 크로마 샘플이 제 1 파라미터 곱하기 병치된 루마 샘플, 플러스 제 2 파라미터와 동일하도록 예측 크로마 샘플을 획득할 수도 있고, 여기서 제 1 파라미터 및 제 2 파라미터는 등식들 (4) 및 (5) 에서 정의된다. 이 예에서, 레퍼런스 샘플들의 세트는 좌측 레퍼런스 샘플들 및 상부 레퍼런스 샘플들의 업-샘플링된 세트이고, 좌측 레퍼런스 샘플들은 현재 PU 의 좌측 경계의 바로 좌측에 있고 상부 레퍼런스 샘플들은 현재 PU 의 상단 경계 바로 위에 있다. 이 예에서, 비디오 코더는 좌측 레퍼런스 샘플들 및/또는 상부 레퍼런스 샘플들에 업-샘플링 방법을 적용함으로써 레퍼런스 샘플들의 세트를 결정할 수도 있다. 예를 들어, 업-샘플링 방법은 좌측 레퍼런스 샘플들 또는 상부 레퍼런스 샘플들 중 어느 것이든 현재 PU 의 좌측 경계 및 현재 PU 의 상단 경계 중 더 짧은 것에 대응하는 것을 업-샘플링하지만, 어느 것이든 현재 PU 의 좌측 레퍼런스 샘플들 및 현재 PU 의 상부 레퍼런스 샘플들 중 더 긴 것을 업-샘플링하지 않을 수도 있다.

[0284] 일부 예들에서, PU 의 더 긴 및 더 짧은 사이드들에 위치된 픽셀들은 업-샘플링될 수도 있고, 업-샘플링 비들은 상이할 수도 있다. 그러나, 업-샘플링 후의 2 개의 사이드들에서 픽셀들의 총 수는 2^m 과 동일해야 하는 것이 요구될 수도 있다 (여기서, m 은 정수이고, m 은 루마 및 크로마 컴포넌트들에 대해 상이할 수도 있다). m 의 값은 K 및 L 의 블록 사이즈에 의존적일 수도 있다. 다시 말해, m 은 PU 의 높이 및/또는 폭에 의존적이다. 예를 들어, PU 는, PU 의 좌측 사이드를 따라 32 개의 레퍼런스 샘플들 및 PU 의 상단 사이드를 따라 32 개의 레퍼런스 샘플들이 존재하도록 레퍼런스 샘플들을 업-샘플링할 수도 있다. 이 예에서, m 은 6 과 동일하다. 다른 예에서, PU 는 4×8 일 수도 있고, 비디오 코더는, PU 의 좌측 사이드를 따라 16 개의 레퍼런스 샘플들 및 PU 의 상단 사이드를 따라 16 개의 레퍼런스 샘플들이 존재하도록 레퍼런스 샘플들을 업-샘플링할 수도 있다. 이 예에서, m 은 4 와 동일하다.

[0285] 또한, 제 8 기법의 일부 예들에서, LM 파라미터들을 계산하기 위해 등식 (4) 및 등식 (5) 를 사용하는 경우, 루마 및 크로마 컴포넌트들 양자 모두에 대해, 비-스퀘어 PU 의 더 짧은 사이드에서 경계의 픽셀들은 업-샘플링되고, 더 긴 경계 (즉, max(K, L)) 의 픽셀들은, 업-샘플링된 더 짧은 경계에서의 픽셀들의 수가 서브-샘플링된 더 긴 경계에서의 픽셀들의 수와 동일하도록 서브-샘플링된다. 업-샘플링 프로세스는 듀플리케이터 또는 보간된 샘플링일 수 있다. 서브-샘플링 프로세스는 데시메이션 또는 보간된 샘플링일 수 있다.

[0286] 따라서, 이 예에서, LM 예측 동작을 수행하는 것의 부분으로서, 비디오 코더는 예측 크로마 샘플이 제 1 파라미터 곱하기 병치된 루마 샘플, 플러스 제 2 파라미터와 동일하도록 예측 크로마 샘플을 획득할 수도 있고, 여기서 제 1 파라미터 및 제 2 파라미터는 등식들 (4) 및 (5) 에서와 같이 정의된다. 이 예에서, 레퍼런스 샘플들의 세트는 레퍼런스 샘플들의 업-샘플링된 세트 및 레퍼런스 샘플들의 서브-샘플링된 세트의 조합이고, 레퍼런스 샘플들의 업-샘플링된 세트는 어느 것이든 좌측 레퍼런스 샘플들 및 상부 레퍼런스 샘플들 중 더 적은 샘

플들을 포함하는 것의 업-샘플링된 버전이다. 이 예에서, 레퍼런스 샘플들의 서브-샘플링된 세트는, 어느 것이든 좌측 레퍼런스 샘플들 및 상부 레퍼런스 샘플들 중 더 많은 샘플들을 포함하는 것의 서브-샘플링된 버전이다. 이 예에서, 좌측 레퍼런스 샘플들은 현재 PU 의 좌측 경계의 바로 좌측에 있고, 상부 레퍼런스 샘플들은 현재 PU 의 상단 경계 바로 위에 있다.

[0287] 일부 예들에서, 위에서 언급된 제 8 기법의 예들에 대해, 서브-샘플링 또는 업-샘플링 프로세스 후에, 루마 컴포넌트에 대해서만 (예를 들어, 본 개시물의 다른 곳에서 설명된 바와 같은) 다운-샘플링 프로세스는 컬러 포맷이 4:4:4 가 아닌 경우를 커버하도록 또한 적용될 수도 있다. 따라서, 4:4:4 외의 현재 픽처의 컬러 포맷에 기초하여, 비디오 코더는 예측 블록의 루마 샘플들을 서브-샘플링 또는 다운-샘플링할 수도 있다. 일부 예들에서, 루마 샘플들의 2 개의 다운-샘플링 프로세스들은 하나로 머지될 수 있다.

[0288] 또한, 제 8 기법의 일부 예들에서, 경계 픽셀들에 대한 서브-샘플링/업-샘플링의 다양한 방식들이 적용될 수도 있다. 일 예에서, 서브-샘플링/업-샘플링 방법은 PU 사이즈에 (즉, K 및 L 의 값들에) 의존적이다. 다른 예에서, 서브-샘플링/업-샘플링에 대한 방법들은 시퀀스 파라미터 세트, 픽처 파라미터 세트, 슬라이스 헤더, 또는 다른 신택스 구조에서 시그널링될 수도 있다.

[0289] 제 8 기법의 일부 예들에서, 업-샘플링/다운-샘플링 (또는 서브-샘플링) 은 내포적인 방식으로 구현된다. 다시 말해, 업-샘플링 또는 서브-샘플링 기법은 내포적으로 결정된다. 즉, 좌측 사이드 경계 또는/및 상위 사이드 경계의 등식 (4) 및 등식 (5) 의 $\sum x_i \cdot y_i$, $\sum x_i$ 및 $\sum y_i$ 에서와 같은 합계 값은 팩터 (S) 에 의해 곱해지거나 나누어진다. S 의 값은 좌측 사이드 경계 또는/및 상위 사이드 경계에서의 픽셀 수의 비에 의존적일 수 있다.

[0290] 따라서, 이 예에서, 예측 크로마 블록을 예측하기 위해 LM 예측 동작을 수행하는 것의 부분으로서, 비디오 코더는 예측 크로마 샘플이 제 1 파라미터 곱하기 병치된 루마 샘플, 플러스 제 2 파라미터와 동일하도록 예측 크로마 샘플을 획득할 수도 있고, 여기서 제 1 파라미터는 다음과 동일하다:

[0291] $$\propto = \frac{I \cdot S \cdot \sum x_i \cdot y_i - S \cdot \sum x_i \cdot S \cdot \sum y_i}{I \cdot S \cdot \sum x_i \cdot x_i - S \cdot \sum x_i \cdot S \cdot \sum x_i}$$

[0292] 여기서 S 는 비-스퀘어 PU 의 좌측 경계 또는/및 상위 경계에서의 픽셀 수의 비에 의존적이고, I 은 서브-샘플링 방법에 따라 결정된 현재 PU 의 좌측 및 상단 경계에서 샘플들의 서브세트에서의 레퍼런스 샘플들의 수이고, x_i 는 서브-샘플링된 복원된 루마 레퍼런스 샘플이며, y_i 는 복원된 크로마 레퍼런스 샘플이다. 일부 예들에서, K×L 크로마 블록에 대해 S = max(K, L)/min(K, L) 이다.

[0293] 전술된 바와 같이, DST-Ⅶ, DCT-Ⅷ, DST-Ⅰ 및 DCT-Ⅴ 를 사용하는 강화 다중 변환 (EMT) 스킴이 제안되어 있다. 또한, 위에서 논의된 바와 같이, EMT 가 적용하는지 또는 아닌지 여부는 CU 내의 모든 TU들에 대해, 플래그, 즉 EMT 플래그를 사용하는 CU 레벨에서 제어된다. EMT-인에이블된 CU 내의 각각의 TU 에 대해, 사용될 수평 또는 수직 변환은 선택된 변환 세트, 즉 EMT 인덱스로 인덱스에 의해 시그널링된다.

[0294] 그러나, 이전에-제안된 바와 같이 EMT 스킴을 제어하는 것은, CU 에서의 각각의 PU 의 잔차 특징이 상이하다면 효율적이지 않을 수도 있다. 예를 들어, 이전에-제안된 EMT 스킴을 제어하는 것은 CU 내의 인트라-코딩된 PU 및 인터-코딩된 PU 에 대해 효율적이지 않을 수도 있다. 따라서, 본 개시물의 제 9 기법에 따르면, EMT 가 하나의 슬라이스, 픽처, 또는 시퀀스에 대해 인에이블되고 하나의 CU 가 수직 방향에서 2 개의 PU 들 (예를 들어, N×2N 파티션) 로 스플리팅되는 경우, EMT 플래그의 시그널링은 다음의 방식으로 수정된다: 좌측 PU 가 인트라-코딩된 PU 이면, 변환 심도는 0 일 수 있다. 다시 말해, CU 의 변환 트리는 0 이상의 심도를 가질 수도 있다. 이 경우에서, EMT 플래그는 CU 레벨에서 시그널링될 수도 있다. 변환 심도가 0 이 아니면, EMT 플래그는 PU 레벨에서 시그널링될 수도 있다. 따라서, EMT 는 각각의 PU 에 대해 인에이블되거나 또는 인에이블되지 않을 수도 있다.

[0295] 또한, 본 개시물의 제 9 기법에 따르면, EMT 가 하나의 슬라이스, 픽처, 또는 시퀀스에 대해 인에이블되는 경우 및 하나의 CU 가 수평 방향에서 2 개의 PU 들 (예를 들어, 2N×N 파티션) 로 스플리팅되는 경우, EMT 플래그의 시그널링은 다음의 방식으로 수정된다: 상부 PU 가 인트라-코딩된 PU 이면, 변환 심도는 0 일 수도 있다. 다시 말해, CU 의 변환 트리는 0 이상의 심도를 가질 수도 있다. 이 경우에서, EMT 플래그는 CU 레벨에서 시그널링될 수도 있다. 변환 심도가 0 이 아니면, EMT 플래그는 PU 레벨에서 시그널링될 수도 있다.

즉, 각각의 PU 는 EMT 인에이블되거나 또는 인에이블되지 않을 수도 있다.

[0296] 따라서, 제 9 기법에 따르면, 비디오 인코더 (20) 는, 비디오 데이터의 인코딩된 표현을 포함하는 비트스트림에서, 제 1 신택스 엘리먼트를 포함할 수도 있다. 제 1 신택스 엘리먼트는, EMT 가 경계를 따라 2 개의 PU들로 정확히 파티셔닝되는 특정 CU 에 대해 인에이블되는지 여부를 나타낸다. 이 예에서, 제 1 신택스 엘리먼트가 특정 CU 또는 2 개의 PU들의 특정 PU 에 있는지 여부는 CU 의 PU들의 스플리팅 방향에 의존한다. 또한, 이 예에서, 특정 CU 에 대해 인에이블되는 EMT 에 기초하여, 특정 CU 의 각각의 개별의 TU 에 대해, 비디오 인코더 (20) 는, 비트스트림에서 개별의 TU 에 대한 개별의 선택된 변환 세트를 나타내는 개별의 신택스 엘리먼트를 포함할 수도 있다. 이 예에서, 특정 CU 에 대해 인에이블되는 EMT 에 기초하여, 비디오 인코더 (20) 는 개별의 TU 의 변환 계수들에 개별의 선택된 변환 세트의 하나 이상의 변환들을 적용하여 샘플 도메인에서 개별의 TU 에 대한 개별의 변환 블록을 획득할 수도 있다. 이 예에서, 비디오 인코더 (20) 는, 비디오 데이터의 인코딩된 표현을 포함하는 비트스트림에서, CU 의 TU들의 하나 이상을 나타내는 데이터를 포함할 수도 있다. 이 예에서, 경계는 수평 경계일 수도 있고, 또는 경계는 수직 경계일 수도 있다.

[0297] 대응하는 예에서, 비디오 디코더 (30) 는 제 1 신택스 엘리먼트를 획득할 수도 있다. 제 1 신택스 엘리먼트는, EMT 가 경계를 따라 2 개의 PU들로 정확히 파티셔닝되는 특정 CU 에 대해 인에이블되는지 여부를 나타낸다. 이 예에서, 제 1 신택스 엘리먼트가 특정 CU 또는 2 개의 PU들의 특정 PU 에 있는지 여부는 CU 의 PU들의 스플리팅 방향에 의존한다. 이 예에서, EMT 가 특정 CU 에 대해 인에이블된다는 결정에 응답하여, 특정 CU 의 각각의 개별의 TU 에 대해, 비디오 디코더 (30) 는 개별의 TU 에 대한 개별의 선택된 변환 세트를 나타내는 개별의 신택스 엘리먼트를 획득할 수도 있다. 부가적으로, 특정 CU 에 대해 EMT 가 인에이블된다는 결정에 응답하여, 특정 CU 의 각각의 개별의 TU 에 대해, 비디오 디코더 (30) 는 개별의 TU 의 변환 계수들에 개별의 선택된 변환 세트의 하나 이상의 변환들의 역을 적용하여 샘플 도메인에서 개별의 TU 에 대한 개별의 변환 블록을 획득할 수도 있다. 이 예에서, 비디오 디코더 (30) 는, CU 의 TU들에 대한 변환 블록들에 적어도 부분적으로 기초하여 CU 의 코딩 블록을 복원할 수도 있다. 이 예에서, 경계는 수평 경계일 수도 있고, 또는 경계는 수직 경계일 수도 있다.

[0298] 본 개시물의 제 10 기법에 따르면, 여러 변환 트리 구조들이 하나의 슬라이스, 픽처, 또는 시퀀스를 코딩하기 위해 적용될 수도 있다. 예를 들어, 일 예에서, 변환 트리 구조들은 미리-정의된다. 일부 예들에서, 각각의 픽처, 슬라이스, 최대 코딩 유닛, CU, 또는 PU 에 대해, 비디오 인코더 (20) 는 선택된 변환 트리 구조를 시그널링할 수도 있다. 대안으로, 일부 예들에서, 비디오 디코더 (30) 는 예측 모드들/파티션 사이즈들과 같은, 코딩된 정보로부터 선택된 변환 트리를 도출할 수도 있다.

[0299] 따라서, 제 10 기법에 따르면, 비디오 인코더 (20) 는 복수의 미리정의된 트리 구조들 중에서부터의 특정 트리 구조에 기초하여 비디오 데이터의 CU 를 CU 의 TU들로 파티셔닝할 수도 있다. 이 예에서, 트리 구조의 루트 노드는 CU 의 코딩 블록에 대응한다. 또한, 이 예에서, 트리 구조의 각각의 개별의 비-루트 노드는 개별의 비-루트 노드의 페어런트 노드에 대응하는 블록의 파티션인 개별의 블록에 대응한다. 이 예에서, 트리 구조의 리프 노드들은 CU들의 TU들에 대응한다. 부가적으로, 이 예에서, 비디오 인코더 (20) 는, 비디오 데이터의 인코딩된 표현을 포함하는 비트스트림에서, CU 의 TU들의 하나 이상을 나타내는 데이터를 포함한다. 일부 예들에서, 비디오 인코더 (20) 는, 비트스트림에서, 특정 트리 구조를 식별하는 하나 이상의 신택스 엘리먼트들을 더 포함할 수도 있다. 일부 예들에서, CU들에 적용 가능한 특정 트리 구조를 나타내는 하나 이상의 신택스 엘리먼트들은 픽처, 슬라이스, LCU, CU, 및 PU 중 하나에 있다. 또한, 일부 예들에서, CU 가 TU들로 파티셔닝된다고 결정하는 것의 부분으로서, 비디오 인코더 (20) 는 특정 트리 구조를 명시적으로 시그널링하지 않고 코딩된 정보로부터 특정 트리 구조를 결정한다. 이러한 예들에서, 코딩된 정보는 예측 모드들 및 파티션 사이즈들 중 적어도 하나를 포함할 수도 있다.

[0300] 대응하는 예에서, 비디오 디코더 (30) 는 복수의 미리정의된 트리 구조들 중에서부터의 특정 트리 구조에 기초하여 비디오 데이터의 CU 가 CU 의 TU들로 파티셔닝된다고 결정할 수도 있다. 이 예에서, 트리 구조의 루트 노드는 CU 의 코딩 블록에 대응한다. 트리 구조의 각각의 개별의 비-루트 노드는 개별의 비-루트 노드의 페어런트 노드에 대응하는 블록의 파티션인 개별의 블록에 대응한다. 이 예에서, 트리 구조의 리프 노드들은 CU들의 TU들에 대응한다. 부가적으로, 이 예에서, 비디오 디코더 (30) 는 CU 의 TU들 중 적어도 하나에 대한 데이터에 기초하여, CU 의 코딩 블록을 복원할 수도 있다. 일부 예들에서, 비디오 디코더 (30) 는 인코딩된 비디오 데이터를 포함하는 비트스트림으로부터, 특정 트리 구조를 식별하는 하나 이상의 신택스 엘리먼트를 획득할 수도 있다. CU들에 적용 가능한 특정 트리 구조를 나타내는 하나 이상의 신택스 엘리먼트들은 픽처, 슬라이스, LCU, CU, 및 예측 유닛 중 하나에 있다. 또한, 일부 예들에서, CU 가 TU들로 파티셔닝된다고

결정하는 것의 부분으로서, 비디오 디코더 (30) 는 특정 트리 구조의 명시적인 시그널링 없이 코딩된 정보로부터 특정 트리 구조를 결정할 수도 있다. 코딩된 정보는 예측 모드들 및 파티션 사이즈들 중 적어도 하나를 포함할 수도 있다.

[0301] 본 개시물의 제 11 예에서, 1×N 및 N×1 과 동일한 사이즈를 갖는 변환들은 또한, 인터 코딩된 블록들에 적용될 수도 있다. 예를 들어, 일 예에서, 이러한 TU들은 단지, 특정 변환 심도, 예를 들어 최고 변환 심도에 대해서만 허용된다. 일부 예들에서, 이러한 TU들은 단지, 8×8 과 동일한 CU 사이즈와 같은 특정 코딩 블록들에 대해서만 허용된다. 또한, 일부 예들에서, 제 11 기법은 단지, 루마와 같은 특정 컬러 컴포넌트에 대해서만 적용 가능하다.

[0302] 따라서, 제 11 기법에 따르면, 비디오 인코더 (20) 는 1×N 또는 N×1 변환을 인터 코딩된 블록의 잔차 데이터에 적용함으로써 변환-도메인 데이터 (예를 들어, 변환 계수들) 를 결정할 수도 있다. 이 예에서, 비디오 인코더 (20) 는, 비디오 데이터의 인코딩된 표현을 포함하는 비트스트림에서, 변환-도메인 데이터를 나타내는 데이터를 포함할 수도 있다. 대응하는 예에서, 비디오 디코더 (30) 는 인터 코딩된 블록의 변환 계수들에 1×N 또는 N×1 변환을 적용함으로써 샘플-도메인 데이터를 결정할 수도 있다. 이 예에서, 비디오 디코더 (30) 는, 샘플-도메인 데이터에 부분적으로 기초하여, 비디오 데이터의 CU 의 코딩 블록을 복원할 수도 있다. 예를 들어, 비디오 디코더 (30) 는 샘플-도메인 데이터의 샘플들을 잔차 데이터의 대응하는 샘플들에 추가하여 CU 의 코딩 블록을 복원할 수도 있다. 일부 경우들에서, 비디오 인코더 (20) 및 비디오 디코더 (30) 를 수반하는 제 11 기법의 상기 예들에 대해, 1×N 및 N×1 변환들은 특정 변환 심도에 대해서만 허용된다. 부가적으로, 일부 경우들에서, 비디오 인코더 (20) 및 비디오 디코더 (30) 를 수반하는 제 11 기법의 상기 예들에 대해, 1×N 및 N×1 변환들은 특정 사이즈들의 CU들에 대해서만 허용된다.

[0303] 본 개시물의 제 12 기법에 따르면, 인터 코딩된 CU들에 대한 HEVC 에서 정의된 비대칭 모션 파티셔닝은 또한, 인트라 코딩된 CU들에 적용된다. 인트라 예측된 CU들을 PU들로 비대칭적으로 파티셔닝하는 것은, 비디오 인코더 (20) 가, 압축 성능을 증가시킬 수도 있는, 상이한 오브젝트들에 대응하는 영역들로 CU 를 더 정확하게 분할하게 할 수도 있다. 따라서, 제 12 기법의 예에 따르면, 비디오 인코더 (20) 는 비디오 데이터의 인트라 예측된 CU 를 PU들로 비대칭적으로 파티셔닝할 수도 있다. 이 예에서, 비디오 인코더 (20) 는 인트라 예측된 CU 의 각각의 개별의 PU 에 대해 개별의 예측 블록을 결정할 수도 있다. 또한, 이 예에서, 비디오 인코더 (20) 는 인트라 예측된 CU 의 PU들에 대한 예측 블록들 및 인트라 예측된 CU 의 코딩 블록에 기초하여 잔차 데이터를 획득할 수도 있다. 부가적으로, 이 예에서, 비디오 인코더 (20) 는, 비디오 데이터의 인코딩된 표현을 포함하는 비트스트림에서, 잔차 데이터를 나타내는 데이터를 포함할 수도 있다.

[0304] 제 12 기법의 대응하는 예에서, 비디오 디코더 (30) 는 비디오 데이터의 인트라 예측된 CU 가 PU들로 비대칭적으로 파티셔닝된다고 결정할 수도 있다. 이 예에서, 비디오 디코더 (30) 는 인트라 예측된 CU 의 각각의 개별의 PU 에 대해 개별의 예측 블록을 결정할 수도 있다. 부가적으로, 이 예에서, 비디오 디코더 (30) 는 인트라 예측된 CU 의 PU들에 대한 예측 블록들에 기초하여, 인트라 예측된 CU 의 코딩 블록을 복원할 수도 있다.

[0305] 본 개시물의 제 13 기법에 따르면, 하나의 인트라 코딩된 CU 가 다수의 PU들을 포함하는 경우, 각각의 PU 는 그 자신의 크로마 예측 모드를 가질 수도 있다. 다시 말해, PU 는 제 1 인트라 예측 모드 (즉, 루마 예측 모드) 및 제 2 인트라 예측 모드 (즉, 크로마 예측 모드) 를 가질 수도 있다. 비디오 코더는 루마 예측 모드를 사용하여 PU 의 루마 예측 블록을 결정할 수도 있고, 크로마 예측 모드를 사용하여 PU 의 크로마 예측 블록들을 결정할 수도 있다. 따라서, 제 13 기법에 따르면, 비디오 인코더 (20) 는 비디오 데이터의 인트라 예측된 CU 가 적어도 제 1 PU 및 제 2 PU 를 갖는다고 결정할 수도 있다. 이 예에서, 제 1 PU 및 제 2 PU 는 상이한 크로마 예측 모드들을 갖는다. 또한, 이 예에서, 비디오 인코더 (20) 는 비디오 데이터의 인코딩된 표현을 포함하는 비트스트림에서, 적어도 제 1 PU 및 제 2 PU 의 예측 블록들 및 CU 의 코딩 블록에 기초하여 잔차 데이터를 나타내는 데이터를 포함할 수도 있다.

[0306] 제 13 기법의 대응하는 예에서, 비디오 디코더 (30) 는 비디오 데이터의 인트라 예측된 CU 가 적어도 제 1 PU 및 제 2 PU 를 갖는다고 결정할 수도 있다. 이 예에서, 제 1 PU 및 제 2 PU 는 상이한 크로마 예측 모드들을 갖는다. 또한, 이 예에서, 비디오 디코더 (30) 는, 제 1 PU 및 제 2 PU 의 예측 블록들에 적어도 기초하여, CU 의 코딩 블록을 복원할 수도 있다.

[0307] 또한, 제 13 기법의 예에 따르면, 이전에 코딩된 PU 의 크로마 인트라 예측 모드들이 다음의 PU 를 코딩하기 위해 고려될 수도 있다. 이, 비디오 코더는, 코딩 순서에서 현재 PU 이전의 PU 의 크로마 예측 모드에 적어도 부분적으로 기초하여, 현재 PU 의 크로마 예측 모드를 결정할 수도 있다. 예를 들어, 비디오 코더는 현재

PU 의 크로마 인트라 예측 모드에 대한 콘텍스트 모델링에서 이전에 코딩된 PU 의 크로마 인트라 예측 모드들을 사용할 수도 있다. 콘텍스트 모델링은 콘텍스트-적응 엔트로피 코딩에 대한 코딩 콘텍스트의 식별을 포함할 수도 있다. 코딩 콘텍스트는 값의 확률들을 나타낼 수도 있다. 다른 예에서, 비디오 코더는 크로마 인트라 예측 모드 리스트에 대한 하나의 새로운 후보로서 이전에 코딩된 PU 의 크로마 인트라 예측 모드들을 추가할 수도 있다.

[0308] 제 13 기법의 일부 예들에서, 하나의 플래그는 먼저, 모든 PU들이 동일한 크로마 인트라 예측 모드들을 공유하는지 여부를 나타내도록 CU 레벨에서 코딩될 수도 있다. 따라서, 이 예에서, 비디오 인코더 (20) 는, 비트스트림에서, 인트라 예측된 CU 의 모든 PU들이 동일한 크로마 인트라 예측 모드들을 공유하는지 여부를 나타내는 신택스 엘리먼트를 포함할 수도 있다. 유사하게, 비디오 인코더 (20) 는, 비디오 데이터의 인코딩된 표현을 포함하는 비트스트림으로부터, 인트라 예측된 CU 의 모든 PU들이 동일한 크로마 인트라 예측 모드들을 공유하는지 여부를 나타내는 신택스 엘리먼트를 획득할 수도 있다.

[0309] 또한, 제 13 기법의 일부 예들에서, 하나의 CU 내의 모든 크로마 PU들은 동일한 변환 트리를 따르도록 제한된다. 하나의 CU 내의 크로마 PU들 모두가 동일한 변환 트리를 따르도록 제한함으로써, 비디오 인코더 (20) 가 상이한 크로마 PU들에 대한 상이한 변환 트리들의 구조들을 나타내는 비트스트림에서 데이터를 포함할 필요가 없을 수도 있다. 따라서, 비디오 인코더 (20) 는, CU 의 크로마 PU들이 상이하게 구조화된 변환 트리들을 갖는 비트스트림들을 비디오 인코더가 생성하는 것을 제한하는 비디오 코딩 표준에 따르는 비트스트림을 생성할 수도 있다. 유사하게, 비디오 디코더 (30) 는 비디오 데이터의 인코딩된 표현을 포함하는 비트스트림을 획득할 수도 있다. 이 예에서, 비트스트림은, CU 의 크로마 PU들이 상이하게 구조화된 변환 트리들을 갖는 비트스트림들을 비디오 인코더가 생성하는 것을 제한하는 비디오 코딩 표준에 따른다.

[0310] 본 개시물의 제 14 예에서, 하나의 인트라 코딩된 CU 가 다수의 직사각형 PU들을 포함하는 경우, 비디오 코더는 모드-의존 스캔을 적용할 수도 있다. 모드-의존 스캔은 TU 에 대한 2-차원 계수 블록에서의 변환 계수들을 엔트로피 인코딩을 위한 1-차원 계수 벡터로 스캐닝하는데 사용된 스캐닝 순서이다. 비디오 인코더 (20) 는, 인트라 예측 모드가 TU 에 대응하는 PU 에 대해 사용되는 것에 기초하여, 복수의 이용 가능한 스캐닝 순서들 중에서부터 TU 의 변환 계수들을 스캐닝하기 위해 사용할 모드-의존 스캔을 선택할 수도 있다. TU 에 대응하는 PU 는 TU 와 공존할 수도 있거나 또는 TU 와 연관된 영역을 포함할 수도 있다. 모드-의존 스캔을 사용하여 CABAC 에 대한 변환 계수를 더 잘 배열할 수도 있다. HEVC 에서, 모드-의존 스캔들은, 8×8 및 4×4 TU들에 대해서만 허용된다.

[0311] 따라서, 제 14 기법의 예에 따르면, 비디오 인코더 (20) 는 2-차원 변환 계수 블록들에 기초하여 잔차 데이터를 획득할 수도 있다. 이 예에서, 비디오 인코더 (20) 는 비디오 데이터의 인트라 예측된 CU 의 복수의 직사각형 PU들 각각에 대해 예측 블록들을 획득할 수도 있다. 또한, 이 예에서, 비디오 인코더 (20) 는 변환 계수들의 2-차원 블록들을 CU 의 TU들에 대응하는 변환 계수들의 1-차원 어레이들로 배열하도록 모드-의존 스캔을 적용할 수도 있다. 이 예에서, 비디오 인코더 (20) 는, 비디오 데이터의 인코딩된 표현을 포함하는 비트스트림에서, 변환 계수들의 1-차원 어레이들을 나타내는 데이터를 포함할 수도 있다.

[0312] 유사한 예에서, 비디오 디코더 (30) 는 변환 계수들의 1-차원 어레이를 비디오 데이터의 인트라 예측된 CU 의 TU들에 대응하는 2-차원 변환 계수 블록들로 배열하도록 모드-의존 스캔을 적용할 수도 있다. 이 예에서, 인트라 예측된 CU 는 다수의 직사각형 PU들을 갖는다. 또한, 이 예에서, 비디오 디코더 (30) 는 변환 계수 블록들에 기초하여 잔차 데이터를 획득할 수도 있다. 부가적으로, 비디오 디코더 (30) 는 PU들 각각에 대한 예측 블록들을 획득할 수도 있다. 이 예에서, 비디오 디코더 (30) 는, 잔차 데이터 및 예측 블록들에 기초하여 CU 의 코딩 블록을 복원할 수도 있다.

[0313] 제 14 기법의 일 예에서, 모드-의존 스캔의 적용은 소정의 TU 사이즈들, 예컨대 8×4 또는 4×8 에 제한된다. 일부 예들에서, 모드-의존 스캔은 소정의 CU 사이즈들, 예컨대 단지 8×8, 또는 8×8 및 16×16 에만 제한된다. 또한, 일부 예들에서, HEVC 에서 8×8 및 4×4 와 동일한 TU 사이즈들에 대해 사용된 스캔 패턴과 인트라 예측 모드 간의 맵핑의 규칙이 재사용될 수도 있다. 일부 예들에서, 직사각형 TU 사이즈들에 의존적인 상이한 맵핑 함수들이 적용될 수도 있다.

[0314] 본 개시물의 다른 곳에서 사용된 바와 같이, VCEG-AZ07 은 HEVC 에서 사용된 2-탭 인트라 보간 필터에 대해 방향성 인트라 예측의 정확도를 개선시키기 위해 4-탭 인트라 보간 필터를 사용하는 것을 제안하였다. 그러나, VCEG-AZ07 은, 비디오 코더가 비-스퀘어 인트라 코딩된 PU 에 대해 4-탭 인트라 보간 필터를 선택하는 방법을 나타내지 않는다. 차라리, VCEG-AZ07 은, 비디오 코더가 4×4 및 8×8 블록들에 대해 큐빅 보간 필

터들을 사용하고, 16×16 및 더 큰 블록들에 대해 가우시안 보간 필터들을 사용한다는 것을 지정한다. 본 개시물의 제 15 기법에서, K×L 와 동일한 사이즈를 갖는 비-스퀘어 인트라 코딩된 PU 에 대해, 4-탭 인트라 보간 필터들에 대하여 본 개시물의 다른 곳에서 설명된 바와 같이 스캔 패턴들 또는 4-탭 필터 유형을 결정하는 경우 비-스퀘어 인트라 코딩된 PU 는 N×N 와 동일한 변환 사이즈로서 취급되고, 여기서 log2(N*N) = ((log2(K) + log2(L)) >> 1) << 1) 이고, 여기서 log2 는 바이너리 알고리즘이며, >> 및 << 은 각각 로직 우측 및 좌측 시프트이다.

[0315] 따라서, 제 15 기법의 예에서, 비디오 인코더 (20) 는 비디오 데이터의 CU 의 비-스퀘어 인트라 코딩된 PU들에 대해 4-탭 보간 필터를 결정할 수도 있다. 또한, 이 예에서, 비디오 인코더 (20) 는 비-스퀘어 인트라 코딩된 PU 에 대한 예측 블록을 획득하는 것의 부분으로서 결정된 4-탭 보간 필터를 적용할 수도 있다. 예를 들어, 비디오 인코더 (20) 는 2 개의 정수-포지션 레퍼런스 샘플들 (즉, 픽처의 상단-좌측 샘플에 대해 정수 좌표들에서의 레퍼런스 샘플들) 간에 있는 레퍼런스 샘플의 값을 결정하는 경우 4-탭 필터를 적용할 수도 있다. 부가적으로, 이 예에서, 비디오 인코더 (20) 는 비디오 데이터의 인코딩된 표현을 포함하는 비트스트림에서, 비-스퀘어 PU 에 대한 예측 블록 및 CU 의 코딩 블록에 적어도 부분적으로 기초하여 잔차 데이터를 나타내는 데이터를 포함할 수도 있다. 이 예에서, 4-탭 보간 필터를 결정하는 것의 부분으로서, 비디오 인코더 (20) 는 스퀘어 PU 의 사이즈에 기초하여 4-탭 보간 필터를 결정할 수도 있고, 여기서 스퀘어 PU 의 사이즈는 비-스퀘어 인트라 코딩된 PU 의 높이 및 폭에 기초한다.

[0316] 제 15 기법의 대응하는 예에서, 비디오 디코더 (30) 는 비디오 데이터의 CU 의 비-스퀘어 인트라 코딩된 PU 에 대한 4-탭 보간 필터를 결정할 수도 있다. 부가적으로, 이 예에서, 비디오 디코더 (30) 는 비-스퀘어 인트라 코딩된 PU 에 대한 예측 블록을 획득하는 것의 부분으로서 결정된 4-탭 보간 필터를 적용할 수도 있다. 또한, 비디오 디코더 (30) 는 비-스퀘어 PU 에 대한 예측 블록에 적어도 부분적으로 기초하여, CU 의 코딩 블록을 복원할 수도 있다. 이 예에서, 4-탭 보간 필터를 결정하는 것의 부분으로서, 비디오 디코더 (30) 는 스퀘어 PU 의 사이즈에 기초하여 4-탭 보간 필터를 결정할 수도 있고, 여기서 스퀘어 PU 의 사이즈는 비-스퀘어 인트라 코딩된 PU 의 높이 및 폭에 기초한다.

[0317] 제 15 기법의 일부 예들에서, 새로운 4-탭 필터가 비-스퀘어 인트라 코딩된 PU들에 대해 적용될 수도 있다. 즉, 비-스퀘어 인트라 코딩된 PU들에 대해서도, 4-탭 필터가 적용될 수도 있고 이 필터는 스퀘어 PU들에 대해 정의되는 것과 상이할 수도 있다. 또한, 제 15 기법의 일부 예들에서, 인트라 예측 모드와 스캔 패턴 인덱스 간의 상이한 맵핑 테이블은 비-스퀘어 인트라 코딩된 PU들에 대해 적용될 수도 있다.

[0318] K×L 변환 블록은 N×N 와 동일한 변환 사이즈로서 취급되고, 여기서 log2(N*N) = (log2(K) + log2(L) + 1) 이다. 따라서, K×L 변환 블록에 대한 스캔 패턴의 선택은 N×N 블록과 동일할 수도 있다.

[0319] 상기 논의된 바와 같이, HEVC 에서의 변환 블록들은 사이즈 N×N 이고, 여기서 $N = 2^m$ 이고 m 은 정수이다. 또한, HEVC 에서, 비디오 인코더는 2-차원 N×N 변환을 변환 블록에 적용하여 변환 계수들을 생성한다. 보다 구체적으로, 비디오 인코더는 변환 블록의 각각의 로우 및 변환 블록의 각각의 컬럼에 N-포인트 1-차원 변환을 별개로 적용함으로써 2-차원 N×N 변환을 적용한다. 이 방식에서 변환을 적용하는 것은 변환 계수들의 N×N 블록을 초래한다.

[0320] HEVC 에서, 비디오 인코더는 다음을 계산함으로써 변환 블록의 샘플들의 i-번째 로우 또는 컬럼 (w_i) 에 N-포인트 1-차원 DCT 변환을 적용할 수도 있다:

[0321] $$w_i = \sum_{j=0}^{N-1} u_j c_{ij} \quad (18)$$

[0322] 상기의 등식에서, $i = 0, \ldots, N-1$ 이다. DCT 변환 매트릭스 (C) 의 엘리먼트들 (c_{ij}) 은 다음과 같이 정의된다:

[0323] $$c_{ij} = \frac{A \cdot \cos\left[\frac{\pi}{N}\left(j+\frac{1}{2}\right)i\right]}{\sqrt{N}} \quad (19)$$

[0324] 위의 등식에서, $i, j = 0, \ldots, N-1$ 이고 여기서, A 는 $i = 0$ 및 $i > 0$ 각각에 대해 1 및 $2^{1/2}$ 과 동일하다.

[0325] 등식 (19) 에서, $A \cdot \cos\left[\frac{\pi}{N}\left(j+\frac{1}{2}\right)i\right]$ 를 X_{ij} 로 표기하자. 따라서, 등식 (18) 은 $w_i = \sum_{j=0}^{N-1} \frac{u_j X_{ij}}{\sqrt{N}}$ 으로서 재기입될 수도 있다. 비디오 인코더가 수평 및 수직 방향들 양자 모두에서 1-차원 DCT 변환을 적용하기 때문에, 변환 계수 (w_i) 는 궁극적으로 다음과 같이 재기입된다:

[0326] $$w_i = \sum_{k=0}^{N-1}\left(\left(\sum_{j=0}^{N-1}\frac{u_j X_{ij}}{\sqrt{N}}\right)\cdot\frac{X_{ik}}{\sqrt{N}}\right) \quad (20)$$

[0327] 이것은 또한, 다음과 같이 재기입될 수 있다:

[0328] $$\frac{\sum_{k=0}^{N-1}\left(\left(\sum_{j=0}^{N-1}u_j X_{ij}\right)\cdot X_{ik}\right)}{\sqrt{N}\cdot\sqrt{N}} \quad (21)$$

[0329] 따라서, 변환은 $\sqrt{N}\cdot\sqrt{N}$ 의 "정규화 팩터" 를 갖는 것으로서 궁극적으로 고려될 수 있다. N = 2^m 이기 때문에, $\sqrt{N}\cdot\sqrt{N}$ 은 또한 2 의 거듭제곱이다. 따라서, 변환 계수의 값은 나누기 연산 대신에 우측-시프트 연산에 의해 구현될 수 있다. 본 개시물의 다른 곳에서 논의된 바와 같이, 나누기 연산 대신에 우측-시프트 연산들을 사용하는 것은 복잡성을 감소시키고 코딩 속도를 개선시킬 수도 있다.

[0330] 그러나, TU 의 비-스퀘어 변환 블록으로 등식 (19) 를 재사용하는 경우 문제들이 발생할 수도 있다. K*L 변환을 고려하는, (수평 및 수직 변환들 양자 모두를 포함하는) 2-D 변환에 대해, 정규화 팩터는 ($\sqrt{K}*\sqrt{L}$) 일 것이다. N 이 등식 log2(N*N) = ((log2(K) + log2(L)) >> 1) << 1) 을 충족시키는 값으로서 정의되면, 이용된 정규화 팩터 ($\sqrt{N}*\sqrt{N}$) 및 실제 정규화 팩터 ($\sqrt{K}*\sqrt{L}$) 의 비는 $1/\sqrt{2}$ 일 것이다. 다시 말해, 양자화 프로세스에서 N×N 변환 블록으로부터 도출된 동일한 정규화 팩터를 재사용하는 경우, 에너지 (즉, 양자화된 변환 계수들의 제곱들의 합) 가 $\sqrt{2}$ 에 의해 변경된다.

[0331] 본 개시물의 제 16 기법은 이 문제를 다룰 수도 있다. 예를 들어, 본 개시물의 제 16 기법에서, K×L 와 동일한 사이즈를 갖는 비-스퀘어 변환 블록에 대해, (log2(K) + log2(L)) 이 홀수인 경우, HEVC 에서 변환 및 양자화 프로세스는 변경되지 않게 유지되고 비-스퀘어 변환 블록은 N×N 과 동일한 사이즈를 갖는 변환 블록으로서 취급되고, 여기서 log2(N*N) = ((log2(K) + log2(L)) >> 1) << 1) 이다. 다시 말해, (log2(K) + log2(L)) 이 홀수인 것에 기초하여, 비디오 인코더 (20) 는 log2(N*N) = ((log2(K) + log2(L)) >> 1) << 1) 이도록 값 N 을 결정할 수도 있다. 비디오 인코더 (20) 는 그 후, "정규화 팩터" 에서 N 의 결정된 값을 사용하여 등식 (19) 에 따라 정의되는 DCT 변환 매트릭스 (C) 의 엘리먼트들을 사용할 수도 있다. 따라서, 비디오 인코더 (20) 는 등식 (21) 에서 "정규화 팩터" 에 의한 나누기를 위해 우측-시프트 연산을 계속해서 사용할 수도 있다.

[0332] 또한, 본 개시물의 제 16 기법에 따르면, 변환 프로세스 후에 및 양자화 프로세스 전에, 변환 계수들은, $\sqrt{2}$ 의 팩터에 의해 곱해져 수정된다. 다시 말해, 비-스퀘어 변환 블록에 변환을 적용하여 계수 블록을 생성한 후에, 비디오 인코더 (20) 는 계수 블록의 각각의 변환 계수에 $\sqrt{2}$ 의 팩터를 곱한다. 이것은, 정규화 팩터 (즉, $\sqrt{N}*\sqrt{N}$) 대 실제 정규화 팩터 (즉, $\sqrt{K}*\sqrt{L}$) 의 비가 $\frac{1}{\sqrt{2}}$ 와 동일하기 때문이다.

[0333] 예를 들어, K = 8 및 L = 4 라 하자. 이 예에서, log2(4*4) = ((log2(8) + log2(4)) >> 1) << 1) 이므로, N = 4 이다. $\frac{\sqrt{8}\times\sqrt{4}}{\sqrt{4}\times\sqrt{4}}$ 은 $\frac{\sqrt{8\times4}}{\sqrt{4\times4}}$ 와 동일하고, 이것은 $\frac{\sqrt{32}}{\sqrt{16}}$ 와 동일하고, 이것은 $\sqrt{2}$ 와 동일하다.

(log2(K) + log2(L)) 이 짝수인 경우 K 및 L 의 값들에 대해, 사용된 정규화 팩터 (즉, $\sqrt{N}*\sqrt{N}$) 대 실제 정규화 팩터 (즉, $\sqrt{K}*\sqrt{L}$) 의 비는 1 과 동일하다는 것에 주목한다. 따라서, (log2(K) + log2(L)) 이 짝수인 경우, 비디오 인코더 (20) 가 변환 계수들에 $\sqrt{2}$ 의 팩터를 곱할 필요가 없을 수도 있다.

[0334] 탈양자화 프로세스 후에, 탈양자화된 계수들은 또한, $\sqrt{2}$ 의 팩터에 의해 나누어져, 수정된다. 양자화 전에 변환 계수들에 $\sqrt{2}$ 을 곱하고 변환 계수들을 $\sqrt{2}$ 으로 나누는 것은, 양자화 프로세스에서 그렇지 않으면 손실될 정보를 보존할 수도 있다. 이 정보를 보존하는 것은 원래의 변환 블록의 더 정확한 복원을 보장할 수도 있다.

[0335] 제 16 기법의 다른 예에서, HEVC 에서의 변환 및 양자화 프로세스는 변경되지 않게 유지되고, 이것은 N×N 과 동일한 변환 사이즈로서 취급되며, 여기서 log2(N*N) = (log2(K) + log2(L) + 1) 이다. 변환 후에 그리고 양자화 프로세스 전에, 변환 계수들은 $\sqrt{2}$ 의 팩터로 나누어져, 수정된다. 탈양자화 프로세스 후에, 탈양자화된 계수들은 또한, $\sqrt{2}$ 의 팩터로 곱해져, 수정된다.

[0336] 상기의 제 16 기법의 예에서, $\sqrt{2}$ 의 팩터는 그 근사치에 의해 표현될 수도 있다. 예를 들어, ($x*\sqrt{2}$) 의 프로세스는 (x*181) >> 7 에 의해 근사화될 수 있고, 여기서 >> 는 우측 시프트 연산을 나타낸다. ($x/\sqrt{2}$) 의 프로세스는 ($x*\sqrt{2}$)/2, 즉, (x*181) >> 8 에 의해 근사화될 수 있고, 여기서 >> 는 우측 시프트 연산을 나타낸다.

[0337] 따라서, 상기에서 제시된 제 16 기법의 예에서, 비디오 인코더 (20) 는 CU 의 비-스퀘어 TU 의 변환 블록에 변환을 적용하여 변환 계수들의 블록을 생성할 수도 있다. 부가적으로, 비디오 인코더 (20) 는, 변환 계수들의 블록의 각각의 개별의 변환 계수가 개별의 탈양자화된 변환 계수 곱하기 $\sqrt{2}$ 의 근사치에 기초하도록 변환 계수들을 수정할 수도 있다. 이 예에서, 변환 계수들을 수정한 후에, 비디오 인코더 (20) 는 CU 의 비-스퀘어 PU 의 수정된 변환 계수들에 양자화 프로세스를 적용한다. 또한, 이 예에서, 비디오 인코더 (20) 는, 비디오 데이터의 인코딩된 표현을 포함하는 비트스트림에서, 양자화된 변환 계수들에 기초하는 데이터를 포함할 수도 있다. 일부 예들에서, 비-스퀘어 TU 의 변환 블록에 변환을 적용하는 것의 부분으로서, 비디오 인코더 (20) 는, 탈양자화된 변환 계수들에, 사이즈 N×N 를 갖는 변환을 적용할 수도 있고, 여기서 log2(N*N) = ((log2(K) + log2(L)) >> 1) << 1) 이다.

[0338] 대응하는 예에서, 비디오 디코더 (30) 는 비디오 데이터의 CU 의 비-스퀘어 PU 의 변환 계수들에 탈양자화 프로세스를 적용할 수도 있다. 이 예에서, 변환 계수들에 탈양자화 프로세스를 적용한 후에, 비디오 디코더 (30) 는, 탈양자화된 변환 계수들의 각각의 개별의 탈양자화된 변환 계수가 개별의 탈양자화된 변환 계수 나누기 $\sqrt{2}$ 의 근사치에 기초하도록 탈양자화된 변환 계수들을 수정할 수도 있다. 일부 예들에서, 수정된 탈양자화된 변환 계수들에 역 변환을 적용하는 것의 부분으로서, 비디오 디코더 (30) 는, 수정된 탈양자화된 변환 계수들에, 사이즈 N×N 를 갖는 변환들을 적용할 수도 있고, 여기서 log2(N*N) = ((log2(K) + log2(L)) >> 1) << 1) 이다.

[0339] HEVC 에서, 비디오 인코더 (20) 는 다음의 등식을 사용하여 양자화된 변환 계수 (즉, 레벨 (level)) 를 계산할 수도 있다:

[0340] $$level = \left(\left(coeff \times f_{QP\%6} + offset_Q \right) \gg \frac{QP}{6} \right) \gg shift2 \quad (22)$$

[0341] 여기서 $coeff$ 는 변환 계수이고, $offset_q$ 는 오프셋 값이고, QP 는 양자화 파라미터이고, $shift\ 2 = 29 - M - B$ 이고, B 는 비트 심도이고, M= log2(N) 이며,

[0342] $$f = [f_0, \ldots f_5]^T = [26214, 23302, 20560, 18396, 16384, 14564]^T \quad (23)$$

[0343] 이다.

[0344] 또한, HEVC 에서, 비디오 디코더 (30) 는 다음의 등식을 사용하여 양자화된 변환 계수를 역 양자화할 수도 있다:

[0345] $$coeff_Q = \left(\left(level \times \left(g_{QP\%6} \ll \frac{QP}{6} \right) \right) + offset_{IQ} \right) \gg shift1 \quad (24)$$

[0346] 등식 (24) 에서, $coeff_Q$ 는 역 양자화된 변환 계수이고, $level$ 은 양자화된 변환 계수이고, $offset_{IQ}$ 는 오프셋 값 = 1 << (M - 10 + B) 이고, $shift1$ = (M - 9 + B) 이며, g 는 이하에서, 등식 (25) 에 도시된 바와 같이 정의된다:

[0347] $$g = [f_0, \ldots f_5]^T = [40, 45, 51, 57, 64, 72]^T \quad (25)$$

[0348] 본 개시물의 기법에 따르면, 비디오 인코더 (20) 는, ($\log_2$(W)+$\log_2$(H)) 이 홀수 또는 짝수인지 여부에 따라 상이한 양자화 매트릭스들 (즉, f 의 버전들) 을 사용할 수도 있다. 유사하게, 비디오 디코더 (30) 는, ($\log_2$(W)+$\log_2$(H)) 이 홀수 또는 짝수인지 여부에 따라 상이한 역 양자화 매트릭스들 (즉, g 의 버전들) 을 사용할 수도 있다. g 의 예는 다음과 같이 정의된다:

[0349] [40,45,51,57,64,72], // 합이 짝수인 경우

[0350] [7240,8145,9231,10317,11584,13032] // 합이 홀수인 경우

[0351] 이것이 짝수인 경우에 대해 g 의 각각의 대응하는 값은 181 에 의해 곱해진다는 것을 주목한다. 이 예에서, $\sqrt{2}$ 의 보상이 g 에서 이미 고려되었기 때문에 양자화 단계들 전에 또는 후에 곱셈 또는 나눗셈 프로세스들을 수행할 필요가 없다.

[0352] 또한, 등식들 (22) 및 (24) 에서, 양자화 매트릭스들 (f 및 g) 에서 선택된 값은 양자화 파라미터 (QP) 에 기초하여 선택된다. 양자화 매트릭스들 (f 및 g) 에서 선택된 값들은 양자화 매트릭스 계수들로서 본원에 지칭될 수도 있다. 본 개시물의 일부 예들에서, 비디오 인코더 (20) 및 비디오 디코더 (30) 는, 양자화 파라미터에 기초하여 그리고 또한, (log2(W)+log2(H)) 이 홀수 또는 짝수인지 여부에 기초하여 양자화 매트릭스 계수들을 선택할 수도 있다.

[0353] 상기에서 간략히 설명되고 도 2a 및 도 2b 에서 예시된 것과 같이, HEVC 에서, 비디오 코더는 항상, 재귀적 z-스캔 순서로 CU 의 TU들을 프로세싱한다. 따라서, 도 2a 에 도시된 바와 같이, TU 의 변환 계수들에 대응하는 데이터 "a" 는 TU 의 변환 계수들에 대응하는 데이터 "b" 전에 비트스트림에서 나타나는, 등등이다. 본 개시물의 제 17 기법은, 인터 모드 또는 *comb_mode* 또는 인트라 모드로 하지만 비-스퀘어 파티션들을 갖는 CU들 대해, 변환 계수의 코딩 순서가 항상 재귀적 z-스캔을 사용하는 대신에 PU 코딩 순서에 의존한다는 것을 제안한다. 제 17 기법의 일부 예들에서, 하나의 PU 내의 모든 계수들은 다른 PU 에서의 계수들을 코딩하기 전에 함께 코딩될 것이다. 따라서, 비디오 인코더 (20) 에 대해, CU 의 PU들 중 하나 내의 모든 계수들은 CU 의 PU들 중 다른 하나의 계수들을 인코딩하기 전에 함께 인코딩된다. 유사하게, 이 예에서, 비디오 디코더 (30) 에 대해, CU 의 하나의 PU 내의 모든 계수들은 CU 의 PU들 중 다른 하나의 계수들을 디코딩하기 전에 함께 디코딩된다.

[0354] 이 예에서, PU 의 변환 계수들은, 그 변환 블록들이 PU 의 예측 블록의 영역 내에 있는 TU들의 변환 계수들이다. 예를 들어, CU 의 코딩 블록의 상단-좌측 코너에 대해 16×16 CU 의 PU 의 예측 블록의 상단-좌측 코너의 좌표들을 (0,0) 이라 하고, PU 의 예측 블록의 하단-우측 코너의 좌표들을 (7,15) 라 하자. 또한, 이 예에서, CU 의 TU 의 변환 블록의 상단-좌측 코너의 좌표들을 (4,0) 이라 하고 TU 의 변환 블록의 하단-우측 코너의 좌표들을 (7,15) 라 하자. 이 예에서, TU 의 변환 계수들은 PU 의 변환 계수들이다. 그러나, 이 예에서, TU 의 변환 블록의 상단-좌측 코너가 (8, 0) 이고 TU 의 변환 블록의 하단-우측 코너가 (15, 7) 이면, TU 의 변환 계수들은 PU 의 변환 계수들이 아니다.

[0355] 예를 들어, 도 2a 에 대하여, CU (40) 는, CU (40) 를 CU (40) 의 센터를 수직으로 통과하여 파티셔닝하는 2 개

의 PU들을 갖는다고 가정한다. 따라서, CU (40) 의 제 1 PU 의 변환 계수들은 TU 들의 변환 계수들 "a," "b," "c," "d," 및 "f" 을 포함한다. CU (40) 의 제 2 PU 의 변환 계수들은 TU 들의 변환 계수들 "e," "g," "h," "i," 및 "j" 을 포함한다. 이 예에서, 비디오 인코더 (20) 는 TU 의 변환 계수들을 나타내는 데이터 "f" 다음에 TU 의 변환 계수들을 나타내는 데이터 "e" 를 포함할 수도 있다. 반대로, HEVC 에서, TU 의 변환 계수들을 나타내는 데이터 "f" 는, CU (40) 의 PU들의 사이즈 및 형상에 관계 없이, TU 의 변환 계수들을 나타내는 데이터 "e" 를 뒤따른다. 다시 말해, 하나의 PU 가 다수의 TU들을 포함하는 경우, 심도-우선 횡단을 하는 재귀적 Z-스캔이 PU 내의 이들 계수들을 코딩하기 위해 적용된다. 도 2 를 일 예로서 취하면, 파티션 사이즈가 N×2N 과 동일하면, 코딩 순서는 a, b, c, d, f, e, g, h, i, j 일 수도 있다.

[0356] 제 17 기법의 일부 예들은 단지, 변환 심도가 0 과 동일하지 않은, 즉 PU 사이즈들 보다 더 이상 크지 않은 변환 사이즈인 경우 적용 가능하다. 전술된 AMP 는 HEVC 에서 정의된 4 개의 경우들에서 뿐만 아니라, 다른 비대칭 파티션들을 포함할 수도 있다.

[0357] 위에서 간략히 언급된 바와 같이, HEVC 에서 IC 설계 만이 스퀘어 PU들을 지원한다. 본 개시물 전에, 비-스퀘어 PU들에 대한 IC 파라미터들 (*a* 및 *b*) 을 도출하는 방법이 알려져 있지 않았다. 본 개시물의 제 18 기법은 IC 로 하여금 비-스퀘어 PU들과 사용되게 할 수 있다. 예를 들어, 비디오 인코더 (20) 는 IC 를 사용하여 비디오 데이터의 픽처의 현재 PU 의 비-스퀘어 예측 블록을 생성할 수도 있다. 부가적으로, 비디오 인코더 (20) 는 예측 블록에 기초하여 잔차 데이터를 생성할 수도 있다. 예를 들어, 비디오 인코더 (20) 는 잔차 데이터의 각각의 개별의 샘플이 현재 CU 의 코딩 블록의 개별의 샘플과 예측 블록의 대응하는 개별의 샘플 간의 차이와 동일하도록 잔차 데이터를 생성할 수도 있다. 또한, 비디오 인코더 (20) 는 잔차 데이터에 기초하여 데이터를 포함하는 비트스트림을 출력할 수도 있다. 예를 들어, 비디오 인코더 (20) 는 잔차 데이터에 변환을 적용하여 계수 블록을 생성하고, 계수 블록의 계수들을 양자화하며, 양자화된 계수들의 각각을 나타내는 하나 이상의 신택스 엘리먼트들을 비트스트림에 포함할 수도 있다. 이 예에서, 비디오 인코더 (20) 는 각각의 양자화된 계수에 대한 신택스 엘리먼트들 중 하나 이상을 엔트로피 인코딩할 수도 있다. 다른 예들에서, 비디오 인코더 (20) 는 변환 및/또는 양자화의 적용을 스킵할 수도 있다.

[0358] 또한, 상기에서 제공된 IC 에 관련된 예들 중 하나 이상에 따르면, 비디오 디코더 (30) 는 IC 를 사용하여 비디오 데이터의 픽처의 현재 CU 의 현재 PU 의 비-스퀘어 예측 블록을 생성할 수도 있다. 부가적으로, 비디오 디코더 (30) 는, 예측 블록에 기초하여, 픽처의 블록 (예를 들어, 코딩 블록) 을 복원할 수도 있다. 예를 들어, 비디오 디코더 (30) 는 대응하는 잔차 샘플들에 예측 블록의 샘플들을 추가함으로써 블록의 샘플들을 복원할 수도 있다.

[0359] 제 18 기법의 예들에서, 비-스퀘어 예측 블록을 생성하기 위해 IC 를 사용하는 것의 부분으로서, 비디오 코더 (예를 들어, 비디오 인코더 (20) 및/또는 비디오 디코더 (30)) 는 다음과 같이 예측 블록의 샘플을 결정할 수도 있다:

[0360] $$p(i,j) = a * r(i + dv_x, j + dv_y + b), \text{ 여기서 } (i,j) \in PU_c$$

[0361] 여기서, PU_c 는 현재 PU 이고, (i,j) 는 예측 블록에서의 픽셀들의 좌표이고, (dv_x, dv_y) 는 PU_c 의 벡터 (예를 들어, 디스패리티 벡터) 이다. p(i,j) 는 PU_c 의 예측이고, *r* 은 인터-뷰 레퍼런스 픽처이고, *a* 는 제 1 IC 파라미터이며 *b* 는 제 2 IC 파라미터이다. 또한, 비-스퀘어 예측 블록을 생성하기 위해 IC 를 사용하는 것의 부분으로서, 비디오 코더는 제 1 IC 파라미터를 다음과 같이 계산할 수도 있다:

[0362] $$a = \frac{2N \cdot \sum_{i=0}^{2N-1} Rec_{neig}(i) \cdot Rec_{refneig}(i) - \sum_{i=0}^{2N-1} Rec_{neig}(i) \cdot \sum_{i=0}^{2N-1} Rec_{refneig}(i)}{2N \cdot \sum_{i=0}^{2N-1} Rec_{refneig}(i) \cdot Rec_{refneig}(i) - \left(\sum_{i=0}^{2N-1} Rec_{refneig}(i) \right)^2}$$

[0363] 부가적으로, 비디오 코더는 제 2 IC 파라미터를 다음과 같이 계산할 수도 있다:

[0364] $$b = \frac{\sum_{i=0}^{2N-1} Rec_{neig}(i) - a \cdot \sum_{i=0}^{2N-1} Rec_{refneig}(i)}{2N},$$

[0365] 위의 등식들에서, Rec_{neig} 및 $Rec_{refneig}$ 는 현재 CU 및 레퍼런스 블록의 이웃하는 픽셀 세트 각각을 가리키고, 2N 은 Rec_{neig} 및 $Rec_{refneig}$ 에서의 픽셀 넘버를 가리키며, 현재 CU 는 N×N 와 동일한 사이즈를 갖는다. 다른 예들은 상기에서 표시된 수식들에 대한 변형예들을 사용할 수도 있다.

[0366] 또한, 제 18 기법의 예들에서, IC 가 K×L (K 는 L 과 동일하지 않음) 과 동일한 사이즈를 갖는 하나의 비-스퀘어 PU 에 대해 인에이블되는 경우, 파라미터들은 다양한 방식들로 도출될 수 있다. 예를 들어, 선형 모델 파라미터들을 계산하기 위해 등식 (16) 및 등식 (17) 을 사용하는 경우, PU 의 더 긴 및 더 짧은 사이드들 양자 모두에 위치된 픽셀들은 상이한 서브-샘플링 비들과 같은, 상이한 방식들로 서브-샘플링될 수도 있다. 그러나, 2 개의 사이드들 함께에서 픽셀들의 총 수는 2^m 과 동일해야 한다는 것이 요구될 수도 있다 (여기서, m 은 정수이고, 그 값은 블록 사이즈에 의존적일 수도 있다). 따라서, 이 예에서, Rec_{neigh} 는 현재 CU 의 바로 위 및 현재 CU 의 바로 좌측의 픽셀들의 서브세트이고, $Rec_{refneigh}$ 는 레퍼런스 블록의 바로 위 및 레퍼런스 블록의 바로 좌측의 픽셀들의 서브세트이며, Rec_{neigh} 및 $Rec_{refneigh}$ 에서의 픽셀들의 총 수는 2^m 과 동일하고, 여기서 m 은 정수이다. 서브-샘플링 프로세스는 데시메이션 또는 보간된 샘플링일 수 있다.

[0367] IC 에 대한 파라미터들을 도출하는 다른 예에서, 선형 모델 파라미터들을 계산하기 위해 등식 (16) 및 등식 (17) 를 사용하는 경우, 비-스퀘어 PU 의 더 짧은 사이드에서의 경계의 픽셀들은, 업-샘플링된 경계에서의 픽셀들의 수가 더 긴 경계 (즉, max(K, L)) 에서 픽셀들의 수와 동일하도록 업-샘플링된다. 업-샘플링 프로세스는 듀플리케이터 또는 보간된 샘플링일 수 있다. 따라서, 이 예에서, 예측 블록의 생성하기 위해 IC 를 사용하는 것의 부분으로서, 비디오 코더는, 어느 것이든 현재 CU 의 좌측 사이드 및 상단 사이드 중 더 짧은 것에서 Rec_{neigh} 가 업-샘플링된 픽셀들을 포함하도록 Rec_{neigh} 를 생성할 수도 있다. 부가적으로, 이 예에서, 비디오 코더는, 어느 것이든 레퍼런스 블록의 상단 사이드 및 좌측 사이드 중 더 짧은 것에서 $Rec_{refneigh}$ 가 업-샘플링된 픽셀들을 포함하도록 $Rec_{refneigh}$ 를 생성할 수도 있다.

[0368] 대안으로, PU 의 더 긴 및 더 짧은 사이드들 양자 모두에 위치된 픽셀들이 업-샘플링될 수도 있고, 업-샘플링 비들은 상이할 수도 있다. 따라서, 이 예에서, 예측 블록의 생성하기 위해 IC 를 사용하는 것의 부분으로서, 비디오 코더는, 어느 것이든 현재 CU 의 좌측 사이드 및 상단 사이드 중 더 긴 것에서 Rec_{neigh} 가 업-샘플링된 픽셀들을 포함하도록 Rec_{neigh} 를 생성할 수도 있다. 부가적으로, 비디오 코더는, 어느 것이든 레퍼런스 블록의 상단 사이드 및 좌측 사이드 중 더 긴 것에서 $Rec_{refneigh}$ 가 업-샘플링된 픽셀들을 포함하도록 $Rec_{refneigh}$ 를 생성할 수도 있다. 그러나, 2 개의 사이드들 함께에서 픽셀들의 총 수는 2^m 과 동일해야 하는 것이 요구될 수도 있다 (여기서, m 은 정수이고, m 은 루마 및 크로마 컴포넌트들에 대해 상이할 수도 있다).

[0369] 또한, IC 에 대한 파라미터들을 도출하는 일부 예들에서, 경계 픽셀들에 대한 서브-샘플링/업-샘플링의 다양한 방식들이 적용될 수도 있다. 일 예에서, 서브-샘플링/업-샘플링 방법은 PU 사이즈에 (즉, K 및 L 의 값들에) 의존적이다. 따라서, 비디오 코더는 현재 PU 의 사이즈에 기초하여, Rec_{neigh} 및 $Rec_{refneigh}$ 를 생성하기 위해 사용할 서브-샘플링 방법 또는 업-샘플링 방법을 결정할 수도 있다. 다른 예에서, 서브-샘플링/업-샘플링에 대한 방법들은 시퀀스 파라미터 세트, 픽처 파라미터 세트, 및/또는 슬라이스 헤더에서 시그널링될 수도 있다. 따라서, 일부 예들에서, 비디오 인코더 (20) 는 비트스트림에서, 사용할 서브-샘플링 방법을 나타내는 신택스 엘리먼트를 포함할 수도 있고, 비디오 디코더 (30) 는, 비트스트림으로부터 이를 획득하여 Rec_{neigh} 및 $Rec_{refneigh}$ 를 생성할 수도 있다. 일부 예들에서, 비디오 인코더 (20) 는, 비트스트림에서 사용할 업-샘플링 방법을 나타내는 신택스 엘리먼트를 포함할 수도 있고, 비디오 디코더 (30) 는 비트스트림으로부터 이를 획득하여 업-샘플링된 픽셀들을 생성할 수도 있다.

[0370] IC 에 대한 파라미터들을 도출하는 일부 예들에서, 업-샘플링/다운-샘플링 (또는 서브-샘플링) 은 내포적인 방식으로 구현된다. 예를 들어, 좌측 사이드 경계 및/또는 상부 사이드 경계의 등식 (16) 및 등식 (17) 에서의 합계 값은 팩터 (S) 에 의해 곱해지거나 또는 나누어질 수도 있다. S 의 값은 좌측 사이드 경계 또는/및 상부 사이드 경계에서의 픽셀 넘버의 비에 의존적일 수 있다.

[0371] 제 18 기법의 일부 예들에서, 동일한 서브-샘플링/업-샘플링 방법은 또한, 레퍼런스 블록 (즉, $Rec_{refneigh}$) 의 경

계 픽셀들에 적용될 것이다. 예를 들어, Rec_{neigh} 및 $Rec_{refneigh}$ 양자 모두를 서브-샘플링하기 위해 데시메이션이 사용될 수도 있다.

[0372] 또한, 본 개시물의 특정 기법들에 따르면, LM 이 스퀘어 PU 에 대해 인에이블되는 경우, 루마 및 크로마 경계 픽셀들은, 예를 들어 등식들 (16) 및 (17) 을 사용하여 파라미터들을 도출하도록 먼저, 서브-샘플링될 수도 있다. 서브-샘플링 방법은 시퀀스 파라미터 세트, 픽처 파라미터 세트 또는 슬라이스 헤더에서 미리정의 또는 시그널링될 수도 있다. 서브-샘플링 방법은 예측 유닛 사이즈에 의존적일 수도 있다.

[0373] 따라서, 비디오 코더 (예를 들어, 비디오 인코더 (20) 또는 비디오 디코더 (30)) 는 PU 의 서브-샘플링된 복원된 루마 샘플들로부터 현재 PU 에 대한 예측 크로마 블록을 예측하도록 선형 모델 예측 동작을 수행할 수도 있다. 부가적으로, 비디오 코더는, 예측 크로마 블록에 부분적으로 기초하여, 픽처의 블록을 복원할 수도 있다. 선형 모델 예측 동작을 수행하는 것의 부분으로서, 비디오 코더는 예측 크로마 샘플이 제 1 파라미터 곱하기 병치된 루마 샘플, 플러스 제 2 파라미터와 동일하도록 예측 크로마 샘플을 획득할 수도 있고, 여기서 제 1 파라미터는 다음과 동일하고:

[0374] $$\alpha = \frac{I\sum x_i \cdot y_i - \sum x_i \cdot \sum y_i}{I\sum x_i \cdot x_i - \sum x_i \cdot \sum x_i}$$

[0375] 제 2 파라미터는 다음과 동일하다:

[0376] $$\beta = \left(\sum y_i - \alpha \cdot \sum x_i\right) / I$$.

[0377] 상기의 등식들에서, I 는 서브-샘플링된 방법에 따라 결정된 현재 PU 의 좌측 및 상단 경계에서의 샘플들의 서브세트에서의 레퍼런스 샘플들의 수이고, x_i 는 서브-샘플링된 복원된 루마 레퍼런스 샘플이고, y_i 는 복원된 크로마 레퍼런스 샘플이다. 이 예의 일부 경우들에서, 비디오 인코더 (20) 는, 비트스트림에서, 서브-샘플링 방법을 나타내는 신택스 엘리먼트를 포함할 수도 있고, 비디오 디코더 (30) 는, 비트스트림으로부터 이를 획득할 수도 있다. 이 예의 일부 경우들에서, 비디오 인코더 (20) 및 비디오 디코더 (30) 는 현재 PU 의 사이즈에 기초하여, 서브-샘플링 방법을 결정할 수도 있다.

[0378] 다양한 예들이 설명되었다. 본 개시물의 특정 예들은 별개로 또는 서로와 결합하여 사용될 수도 있다.

[0379] 도 24 는 본 개시물에서 기법들을 구현할 수도 있는 예시의 비디오 인코더 (20) 를 예시하는 블록도이다. 도 24 는 설명의 목적을 위해 제공되고, 본 개시물에 광범위하게 예시 및 설명된 바와 같은 기법들의 제한으로 간주되지 않아야 한다. 설명의 목적을 위해, 본 개시물은 HEVC 코딩의 맥락에서 비디오 인코더 (20) 를 설명한다. 그러나, 본 개시물의 기법들은 다른 코딩 표준들 또는 방법들에 적용 가능할 수도 있다.

[0380] 비디오 인코더 (20) 는 프로세싱 회로부를 포함하고, 비디오 인코더 (20) 는 본 개시물에 설명된 예시의 기법들 중 하나 이상을 수행하도록 구성된다. 이러한 프로세싱 회로부는 고정된 기능 및/또는 프로그램가능 회로부를 포함할 수도 있다. 예를 들어, 비디오 인코더 (20) 는 집적된 회로부를 포함하고, 도 24 에 예시된 다양한 유닛들은 회로 버스와 상호접속되는 하드웨어 회로 블록들로서 형성될 수도 있다. 이들 하드웨어 회로 블록들은 별개의 회로 블록들일 수도 있고, 또는 유닛들 중 2 이상은 공통의 하드웨어 회로 블록 안에 결합될 수도 있다. 하드웨어 회로 블록들은 산술 논리 유닛들 (ALUs), 기본 함수 유닛들 (EFUs), 뿐만 아니라 로직 블록들, 예컨대 AND, OR, NAND, NOR, XOR, XNOR, 및 다른 유사한 로직 블록들과 같은 연산 블록들을 형성하는 전자 컴포넌트들의 조합들로서 형성될 수도 있다.

[0381] 도 24 의 예에서, 비디오 인코더 (20) 는 예측 프로세싱 유닛 (200), 비디오 데이터 메모리 (201), 잔차 생성 유닛 (202), 변환 프로세싱 유닛 (204), 양자화 유닛 (206), 역 양자화 유닛 (208), 역변환 프로세싱 유닛 (210), 복원 유닛 (212), 필터 유닛 (214), 디코딩된 픽처 버퍼 (216), 및 엔트로피 인코딩 유닛 (218) 을 포함한다. 예측 프로세싱 유닛 (200) 은 인터-예측 프로세싱 유닛 (220) 및 인트라-예측 프로세싱 유닛 (226) 을 포함한다. 인터-예측 프로세싱 유닛 (220) 은 모션 추정 유닛 및 모션 보상 유닛 (미도시) 을 포함할 수도 있다. 일부 예들에서, 예측 프로세싱 유닛 (200) 은 본 개시물의 일루미네이션 보상 기법들을 수행한다.

일부 예들에서, 예측 프로세싱 유닛 (200) 은 본 개시물의 LM 기법들을 수행하여 비-스퀘어 크로마 예측 블록들을 생성한다. 또한, 일부 예들에서, 예측 프로세싱 유닛 (200) 은 본 개시물의 IC 기법들을 수행하여

비-스퀘어 예측 블록들을 생성한다.

[0382] 비디오 데이터 메모리 (201) 는 비디오 인코더 (20) 의 컴포넌트들에 의해 인코딩될 비디오 데이터를 저장하도록 구성될 수도 있다. 비디오 데이터 메모리 (201) 에 저장된 비디오 데이터는, 예를 들어 비디오 소스 (18)(도 1) 로부터 획득될 수도 있다. 디코딩된 픽처 버퍼 (216) 는, 예를 들어 인트라- 또는 인터-코딩 모드들에서 비디오 인코더 (20) 에 의해 비디오 데이터를 인코딩하는데 있어서 사용하기 위한 레퍼런스 비디오 데이터를 저장하는 레퍼런스 픽처 메모리일 수도 있다. 비디오 데이터 메모리 (201) 및 디코딩된 픽처 버퍼 (216) 는 다양한 메모리 디바이스들 중 임의의 것, 예컨대 동기식 DRAM (SDRAM) 을 포함하는 동적 랜덤 액세스 메모리 (DRAM), 자기저항 RAM (MRAM), 저항성 RAM (RRAM), 또는 다른 유형들의 메모리 디바이스들에 의해 생성될 수도 있다. 비디오 데이터 메모리 (201) 및 디코딩된 픽처 버퍼 (216) 는 동일한 메모리 디바이스 또는 별개의 메모리 디바이스들에 의해 제공될 수도 있다. 다양한 예들에서, 비디오 데이터 메모리 (201) 는 비디오 인코더 (20) 의 다른 컴포넌트들과 온-칩, 또는 이들 컴포넌트들에 대한 오프-칩일 수도 있다.

[0383] 비디오 인코더 (20) 는 비디오 데이터를 수신한다. 비디오 인코더 (20) 는 비디오 데이터의 픽처의 슬라이스에서 각각의 CTU 를 인코딩할 수도 있다. CTU들 각각은 동일한 크기의 루마 코딩 트리 블록 (CTB)들 및 픽처의 대응하는 CTB들과 연관될 수도 있다. CTU 를 인코딩하는 것의 부분으로서, 예측 프로세싱 유닛 (200) 은 쿼드-트리 파티셔닝을 수행하여 CTU 의 CTB들을 점진적으로 더 작은 블록들로 분할할 수도 있다. 더 작은 블록은 CU들의 코딩 블록들일 수도 있다. 예를 들어, 예측 프로세싱 유닛 (200) 은 CTU 와 연관된 CTB 를 4 개의 동일한 사이즈의 서브-블록들로 파티셔닝하고, 그 서브-블록들 중 하나 이상을 4 개의 동일한 사이즈의 서브-서브-블록들로 파티셔닝하는 등일 수도 있다.

[0384] 비디오 인코더 (20) 는 CTU 의 CU들을 인코딩하여, CU들의 인코딩된 표현들 (즉, 코딩된 CU들) 을 생성할 수도 있다. CU 를 인코딩하는 것의 부분으로서, 예측 프로세싱 유닛 (200) 은 CU 의 하나 이상의 PU들 중 CU 와 연관된 코딩 블록들을 파티셔닝할 수도 있다. 따라서, 각각의 PU 는 루마 예측 블록 및 대응하는 크로마 예측 블록들과 연관될 수도 있다. 비디오 인코더 (20) 및 비디오 디코더 (30) 는 다양한 사이즈들을 갖는 PU들을 지원할 수도 있다. 위에 표시된 바와 같이, CU 의 사이즈는 CU 의 루마 코딩 블록의 사이즈를 지칭할 수도 있고, PU 의 사이즈는 PU 의 루마 예측 블록의 사이즈를 지칭할 수도 있다. 특정 CU 의 사이즈가 2N×2N 이라고 가정하면, 비디오 인코더 (20) 및 비디오 디코더 (30) 는 인트라 예측을 위해 2N×2N 또는 N×N 의 PU 사이즈들, 및 2N×2N, 2N×N, N×2N, N×N 의 대칭적인 PU 사이즈들을 지원할 수도 있고, 또는 인터 예측에 대해 유사하다. 비디오 인코더 (20) 및 비디오 디코더 (30) 는 또한, 인터 예측을 위해 2N×nU, 2N×nD, nL×2N, 및 nR×2N 의 PU 사이즈들에 대한 비대칭적 파티셔닝을 지원할 수도 있다.

[0385] 인터-예측 프로세싱 유닛 (220) 은 CU 의 각각의 PU 상에서 인터 예측을 수행함으로써 PU 에 대한 예측 데이터를 생성할 수도 있다. PU 에 대한 예측 데이터는 PU 에 대한 모션 정보 및 PU 의 예측 블록들을 포함할 수도 있다. 인터-예측 프로세싱 유닛 (220) 은, PU 가 I 슬라이스, P 슬라이스, 또는 B 슬라이스인지 여부에 따라 CU 의 PU 에 대한 상이한 동작들을 수행할 수도 있다. I 슬라이스에서, 모든 PU들은 인트라 예측된다. 따라서, PU 가 I 슬라이스 내에 있으면, 인터-예측 프로세싱 유닛 (220) 은 PU 상에서 인터 예측을 수행하지 않는다. 따라서, I-모드에서 인코딩된 블록들에 대해, 예측 블록은 동일한 프레임 내의 이전에-인코딩된 이웃하는 블록들로부터의 공간 예측을 사용하여 형성된다. PU 가 P 슬라이스에 있으면, 인터-예측 프로세싱 유닛 (220) 은 단-방향 인터 예측을 사용하여 PU 의 예측 블록을 생성할 수도 있다. PU 가 B 슬라이스에 있으면, 인터-예측 프로세싱 유닛 (220) 은 단-방향 또는 양-방향 인터 예측을 사용하여 PU 의 예측 블록을 생성할 수도 있다.

[0386] 인트라-예측 프로세싱 유닛 (226) 은 PU 상에서 인트라 예측을 수행함으로써 PU 에 대한 예측 데이터를 생성할 수도 있다. PU 에 대한 예측 데이터는 다양한 신택스 엘리먼트들 및 PU 의 예측 블록들을 포함할 수도 있다. 인트라-예측 프로세싱 유닛 (226) 은 I 슬라이스들, P 슬라이스들, 및 B 슬라이스들에서의 PU들 상에서 인트라 예측을 수행할 수도 있다.

[0387] PU 상에서 인트라 예측을 수행하기 위해, 인트라-예측 프로세싱 유닛 (226) 은 다수의 인트라 예측 모드들을 사용하여 PU 에 대한 예측 데이터의 다수의 세트들을 생성할 수도 있다. 인트라-예측 프로세싱 유닛 (226) 은 이웃하는 PU들의 샘플 블록들로부터 샘플들을 사용하여 PU 에 대한 예측 블록을 생성할 수도 있다. PU들, CU들, 및 CTU들에 대한 좌측에서 우측으로, 상부에서 하부로의 인코딩 순서를 가정하면, 이웃하는 PU들은 PU 의 위, 위 및 우측, 위 및 좌측, 또는 좌측에 있을 수도 있다. 인트라-예측 프로세싱 유닛 (226) 은 다양한 수들의 인트라 예측 모드들, 예를 들어 33 개의 방향성 인트라 예측 모드들을 사용할 수도 있다. 일부 예들에

서, 인트라 예측 모드들의 수는 PU 와 연관된 영역의 사이즈에 의존할 수도 있다.

[0388] 예측 프로세싱 유닛 (200) 은 PU들에 대한 인터-예측 프로세싱 유닛 (220) 에 의해 생성된 예측 데이터 또는 PU들에 대한 인트라-예측 프로세싱 유닛 (226) 에 의해 생성된 예측 데이터 중에서부터 CU 의 PU들에 대한 예측 데이터를 선택할 수도 있다. 일부 예들에서, 예측 프로세싱 유닛 (200) 은 예측 데이터의 세트들의 레이트/왜곡 메트릭들에 기초하여 CU 의 PU 에 대한 예측 데이터를 선택한다. 선택된 예측 데이터의 예측 블록들은 선택된 예측 블록들로서 본원에 지칭될 수도 있다.

[0389] 잔차 생성 유닛 (202) 은, CU 에 대한 코딩 블록들 (예를 들어, 루마, Cb 및 Cr 코딩 블록들) 및 CU 의 PU들에 대한 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. 예를 들어, 잔차 생성 유닛 (202) 은, 잔차 블록들 내의 각각의 샘플이 CU 의 코딩 블록에서의 샘플과 CU 의 PU 의 대응하는 선택된 예측 블록 내의 대응하는 샘플 간의 차이와 동일한 값을 갖도록 CU 의 잔차 블록들을 생성할 수도 있다.

[0390] 변환 프로세싱 유닛 (204) 은 CU 와 연관된 잔차 블록들을 CU 의 TU들과 연관된 변환 블록들로 파티셔닝하도록 파티셔닝 (예를 들어, 쿼드-트리 파티셔닝) 을 수행할 수도 있다. 따라서, TU 는 루마 변환 블록 및 2 개의 크로마 변환 블록들과 연관될 수도 있다. CU 의 TU들의 루마 및 크로마 변환 블록들의 사이즈들 및 포지션들은 CU 의 PU들의 예측 블록들의 사이즈들 및 포지션들에 기초하거나 기초하지 않을 수도 있다. "잔차 쿼드-트리 (RQT)" 로서 알려진 쿼드-트리 구조는 영역들 각각과 연관된 노드들을 포함할 수도 있다. CU 의 TU들은 RQT 의 리프 노드들에 대응할 수도 있다.

[0391] 일부 예들에서, 변환 프로세싱 유닛 (204) 은 2 (및/또는 4) 개의 차일드 노드들을 갖는 노드들을 포함하는 잔차 트리 구조를 결정하기 위해 본 개시물의 기법들을 수행할 수도 있다. 예를 들어, 비디오 데이터 메모리 (201) 는 비디오 데이터를 수신할 수도 있고, 변환 프로세싱 유닛 (204) 은 트리 구조에 기초하여 비디오 데이터의 CU 를 CU 의 TU들로 파티셔닝할 수도 있다. 이 예에서, 트리 구조에 기초하여 CU 를 CU 의 TU들로 파티셔닝하는 것의 부분으로서, 변환 프로세싱 유닛 (204) 은 트리 구조에서 노드가 트리 구조에서 정확히 2 개의 차일드 노드들을 갖는다고 결정할 수도 있다. 일부 경우들에서, 변환 프로세싱 유닛 (204) 은 또한, 트리 구조에서 제 2 노드가 트리 구조에서 정확히 4 개의 차일드 노드들을 갖는다고 결정할 수도 있다. CU 의 TU들 중 적어도 하나에 대해, 변환 프로세싱 유닛 (204) 은 TU 에 대한 잔차 블록에 변환을 적용하여 TU 에 대한 변환 계수들의 블록을 생성할 수도 있다.

[0392] 변환 프로세싱 유닛 (204) 은 TU 의 변환 블록들에 하나 이상의 변환들을 적용함으로써 CU 의 각각의 TU 에 대한 변환 계수 블록들을 생성할 수도 있다. 변환 프로세싱 유닛 (204) 은 TU 와 연관된 변환 블록에 다양한 변환들을 적용할 수도 있다. 예를 들어, 변환 프로세싱 유닛 (204) 은 이산 코사인 변환 (DCT), 방향성 변환, 또는 개념적으로 유사한 변환을 변환 블록에 적용할 수도 있다. 일부 예들에서, 변환 프로세싱 유닛 (204) 은 변환 블록에 변환들을 적용하지 않는다. 이러한 예들에서, 변환 블록은 변환 계수 블록으로서 취급될 수도 있다. 일부 예들에서, 변환 프로세싱 유닛 (204) 은 본 개시물의 EMT 기법들을 수행한다.

[0393] 양자화 유닛 (206) 은 계수 블록에서 변환 계수들을 양자화할 수도 있다. 양자화 프로세스는 변환 계수들의 일부 또는 전부와 연관된 비트 심도를 감소시킬 수도 있다. 예를 들어, n-비트 변환 계수는 양자화 동안 m-비트 변환 계수로 내림 (rounded down) 될 수도 있고, 여기서 n 은 m 보다 크다. 양자화 유닛 (206) 은 CU 와 연관된 양자화 파라미터 (QP) 값에 기초하여 CU 의 TU 와 연관된 계수 블록을 양자화할 수도 있다. 비디오 인코더 (20) 는 CU 와 연관된 QP 값을 조정함으로써 CU 와 연관된 계수 블록들에 적용된 양자화의 정도를 조정할 수도 있다. 양자화는 정보의 손실을 도입할 수도 있고, 따라서 양자화된 변환 계수들은 원래의 변환 계수들보다 더 낮은 정확도를 가질 수도 있다.

[0394] 일부 예들에서, 양자화 유닛 (206) 은, 변환 계수들의 블록의 각각의 개별의 변환 계수가 개별의 탈양자화된 변환 계수 곱하기 $\sqrt{2}$ 의 근사치에 기초하도록 변환 계수들을 수정한다. 이 예에서, 변환 계수들을 수정한 후에, 양자화 유닛 (206) 은 CU 의 비-스퀘어 PU 의 수정된 변환 계수들에 양자화 프로세스를 적용한다.

[0395] 역 양자화 유닛 (208) 및 역변환 프로세싱 유닛 (210) 은 계수 블록에 역 양자화 및 역변환들을 각각 적용하여, 계수 블록으로부터 잔차 블록을 복원할 수도 있다. 복원 유닛 (212) 은 예측 프로세싱 유닛 (200) 에 의해 생성된 하나 이상의 예측 블록들로부터 대응하는 샘플들에 복원된 잔차 블록을 추가하여, TU 와 연관된 복원된 변환 블록을 생성할 수도 있다. 이 방식에서 CU 의 각각의 TU 에 대한 변환 블록들을 복원함

으로써, 비디오 인코더 (20) 는 CU 의 코딩 블록들을 복원할 수도 있다.

[0396] 필터 유닛 (214) 은 CU 와 연관된 코딩 블록들에서 블록킹 아티팩트들을 감소시키도록 하나 이상의 디블록킹 동작들을 수행할 수도 있다. 디코딩된 픽처 버퍼 (216) 는, 필터 유닛 (214) 이 복원된 코딩 블록들 상에서 하나 이상의 디블록킹 동작들을 수행한 후에 복원된 코딩 블록들을 저장할 수도 있다. 인터-예측 프로세싱 유닛 (220) 은 복원된 코딩 블록들을 포함하는 레퍼런스 픽처를 사용하여, 다른 픽처들의 PU들 상에서 인터 예측을 수행할 수도 있다. 또한, 인트라-예측 프로세싱 유닛 (226) 은 디코딩된 픽처 버퍼 (216) 에서 복원된 코딩 블록들을 사용하여 CU 와 동일한 픽처에서의 다른 PU들 상에서 인트라 예측을 수행할 수도 있다.

[0397] 엔트로피 인코딩 유닛 (218) 은 비디오 인코더 (20) 의 다른 기능적 컴포넌트들로부터 데이터를 수신할 수도 있다. 예를 들어, 엔트로피 인코딩 유닛 (218) 은 양자화 유닛 (206) 으로부터 계수 블록들을 수신할 수도 있고, 예측 프로세싱 유닛 (200) 으로부터 신택스 엘리먼트들을 수신할 수도 있다. 엔트로피 인코딩 유닛 (218) 은 데이터 상에서 하나 이상의 엔트로피 인코딩 동작들을 수행하여, 엔트로피-인코딩된 데이터를 생성할 수도 있다. 예를 들어, 엔트로피 인코딩 유닛 (218) 은, CABAC 동작, 콘텍스트-적응 가변 길이 코딩 (CAVLC) 동작, 변수-대-변수 (V2V) 길이 코딩 동작, 신택스 기반 콘텍스트 적응적 이진 산술 코딩 (SBAC) 동작, 확률 구간 파티셔닝 엔트로피 (PIPE) 코딩 동작, 지수-골롬 인코딩 동작, 또는 다른 유형의 엔트로피 인코딩 동작을 데이터 상에서 수행할 수도 있다. 비디오 인코더 (20) 는 엔트로피 인코딩 유닛 (218) 에 의해 생성된 엔트로피-인코딩된 데이터를 포함하는 비트스트림을 출력할 수도 있다. 예를 들어, 비트스트림은 CU 에 대한 RQT 를 나타내는 데이터를 포함할 수도 있다.

[0398] 도 25 는 본 개시물의 기법들을 구현하도록 구성되는 예시의 비디오 디코더 (30) 를 예시하는 블록도이다. 도 25 는 설명의 목적을 위해 제공되고, 본 개시물에 광범위하게 예시 및 설명된 바와 같은 기법들에 대한 제한이 아니다. 설명의 목적을 위해, 본 개시물은 HEVC 코딩의 맥락에서 비디오 디코더 (30) 를 설명한다. 그러나, 본 개시물의 기법들은 다른 코딩 표준들 또는 방법들에 적용 가능할 수도 있다.

[0399] 비디오 디코더 (30) 는 프로세싱 회로부를 포함하고, 비디오 디코더 (30) 는 본 개시물에 설명된 예시의 기법들 중 하나 이상을 수행하도록 구성된다. 예를 들어, 비디오 디코더 (30) 는 집적된 회로부를 포함하고, 도 25 에 예시된 다양한 유닛들은 회로 버스와 상호접속되는 하드웨어 회로 블록들로서 형성될 수도 있다. 이들 하드웨어 회로 블록들은 별개의 회로 블록들일 수도 있고, 또는 유닛들 중 2 이상은 공통의 하드웨어 회로 블록 안에 결합될 수도 있다. 하드웨어 회로 블록들은 산술 논리 유닛들 (ALUs), 기본 함수 유닛들 (EFUs), 뿐만 아니라 로직 블록들, 예컨대 AND, OR, NAND, NOR, XOR, XNOR, 및 다른 유사한 로직 블록들과 같은 연산 블록들을 형성하는 전자 컴포넌트들의 조합들로서 형성될 수도 있다.

[0400] 일부 예들에서, 도 25 에 예시된 유닛들 중 하나 이상은 프로세싱 회로부 상에서 실행되는 소프트웨어 유닛들에 의해 제공될 수도 있다. 이러한 예들에서, 이들 소프트웨어 유닛들에 대한 오브젝트 코드는 메모리에 저장된다. 운영 시스템은, 비디오 디코더 (30) 로 하여금 오브젝트 코드를 취출하고 오브젝트 코드를 실행하게 할 수도 있고, 이것은 비디오 디코더 (30) 로 하여금 예시의 기법들을 구현하기 위한 동작들을 수행하게 한다. 일부 예들에서, 소프트웨어 유닛들은 비디오 디코더 (30) 가 시작 시에 실행하는 펌웨어일 수도 있다. 따라서, 비디오 디코더 (30) 는 예시의 기법들을 수행하도록 하드웨어를 특수화하기 위해 하드웨어 상에서 실행되는 소프트웨어/펌웨어를 갖거나 예시의 기법들을 수행하는 하드웨어를 갖는 구조적 컴포넌트이다.

[0401] 도 25 의 예에서, 비디오 디코더 (30) 는 엔트로피 디코딩 유닛 (250), 비디오 데이터 메모리 (251), 예측 프로세싱 유닛 (252), 역 양자화 유닛 (254), 역변환 프로세싱 유닛 (256), 복원 유닛 (258), 필터 유닛 (260), 및 디코딩된 픽처 버퍼 (262) 를 포함한다. 예측 프로세싱 유닛 (252) 은 모션 보상 유닛 (264) 및 인트라-예측 프로세싱 유닛 (266) 을 포함한다. 다른 예들에서, 비디오 디코더 (30) 는 더 많은, 더 적은, 또는 상이한 기능적 컴포넌트들을 포함할 수도 있다. 일부 예들에서, 예측 프로세싱 유닛 (266) 은 본 개시물의 일루미네이션 보상 기법들을 수행한다. 일부 예들에서, 예측 프로세싱 유닛 (266) 은 본 개시물의 LM 기법들을 수행한다.

[0402] 비디오 데이터 메모리 (251) 는 비디오 디코더 (30) 의 컴포넌트들에 의해 디코딩될 인코딩된 비디오 데이터, 예컨대 인코딩된 비디오 비트스트림을 저장할 수도 있다. 비디오 데이터 메모리 (251) 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 액세스함으로써 획득될 수도 있다. 비디오 데이터 메모리 (251) 는 인코딩된 비디오 비트스트림으로부터 인코딩된 비디오 데이터를 저장하는 코딩된 픽처 버퍼 (CPB) 를 형성할 수도 있다. 디코딩된 픽처 버퍼 (262) 는, 예를 들어 인트라-

또는 인터- 코딩 모드들에서 비디오 디코더 (30) 에 의해 비디오 데이터를 디코딩하는데 있어서 사용하기 위해, 또는 출력을 위해 레퍼런스 비디오 데이터를 저장하는 레퍼런스 픽처 메모리일 수도 있다. 비디오 데이터 메모리 (251) 및 디코딩된 픽처 버퍼 (262) 는 다양한 메모리 디바이스들 중 임의의 것, 예컨대 동기식 DRAM (SDRAM) 을 포함하는 동적 랜덤 액세스 메모리 (DRAM), 자기저항 RAM (MRAM), 저항성 RAM (RRAM), 또는 다른 유형들의 메모리 디바이스들에 의해 형성될 수도 있다. 비디오 데이터 메모리 (251) 및 디코딩된 픽처 버퍼 (262) 는 동일한 메모리 디바이스 또는 별개의 메모리 디바이스들에 의해 제공될 수도 있다. 다양한 예들에서, 비디오 데이터 메모리 (251) 는 비디오 디코더 (30) 의 다른 컴포넌트들과 온-칩, 또는 이들 컴포넌트들에 대한 오프-칩일 수도 있다.

[0403] 비디오 데이터 메모리 (251) 는 비트스트림의 인코딩된 비디오 데이터 (예를 들어, NAL 유닛들) 를 수신 및 저장한다. 엔트로피 디코딩 유닛 (250) 은 비디오 데이터 메모리 (251) 로부터 인코딩된 비디오 데이터 (예를 들어, NAL 유닛들) 를 수신할 수도 잇고, NAL 유닛들을 파싱하여 신택스 엘리먼트들을 획득할 수도 있다. 엔트로피 디코딩 유닛 (250) 은 NAL 유닛들에서 엔트로피-인코딩된 신택스 엘리먼트들을 엔트로피 디코딩할 수도 있다. 예측 프로세싱 유닛 (252), 역 양자화 유닛 (254), 역변환 프로세싱 유닛 (256), 복원 유닛 (258), 및 필터 유닛 (260) 은 비트스트림으로부터 추출된 신택스 엘리먼트들에 기초하여 디코딩된 비디오 데이터를 생성할 수도 있다. 엔트로피 디코딩 유닛 (250) 은 엔트로피 인코딩 유닛 (218) 의 프로세스와 일반적으로 상반된 프로세스를 수행할 수도 있다.

[0404] 비트스트림으로부터 신택스 엘리먼트들을 획득하는 것에 추가하여, 비디오 디코더 (30) 는 비-파티셔닝된 CU 상에서 복원 동작을 수행할 수도 있다. CU 상에서 복원 동작을 수행하기 위해, 비디오 디코더 (30) 는 CU 의 각각의 TU 상에서 복원 동작을 수행할 수도 있다. CU 의 각각의 TU 에 대한 복원 동작을 수행함으로써, 비디오 디코더 (30) 는 CU 의 잔차 블록들을 복원할 수도 있다.

[0405] CU 의 TU 상에서 복원 동작을 수행하는 것의 부분으로서, 역 양자화 유닛 (254) 은 TU 와 연관된 변환 블록들을 역 양자화, 즉 탈-양자화할 수도 있다. 역 양자화 유닛 (254) 이 계수 블록을 역 양자화한 후에, 역변환 프로세싱 유닛 (256) 은 계수 블록에 하나 이상의 역변환들을 적용하여 TU 와 연관된 잔차 블록을 생성할 수도 있다. 예를 들어, 역변환 프로세싱 유닛 (256) 은 역 DCT, 역정수 변환, 역 카루넨-루베 변환 (KLT), 역회전 변환, 역방향 변환, 또는 다른 역변환을 계수 블록에 적용할 수도 있다. 일부 예들에서, 역 변환 프로세싱 유닛 (256) 은 본 개시물의 EMT 기법들을 수행한다.

[0406] 본 개시물의 일부 예들에 따르면, 역 양자화 유닛 (254) 은 비디오 데이터의 CU 의 비-스퀘어 TU 의 변환 계수들에 탈양자화 프로세스를 적용할 수도 있다. 또한, 변환 계수들에 탈양자화 프로세스를 적용한 후에, 역 양자화 유닛 (254) 은, 탈양자화된 변환 계수들의 각각의 개별의 탈양자화된 변환 계수가 개별의 탈양자화된 변환 계수 나누기 $\sqrt{2}$ 의 근사치에 기초하도록 탈양자화된 변환 계수들을 수정할 수도 있다.

[0407] 일부 예들에서, 역 변환 프로세싱 유닛 (256) 은 2 (및/또는 4) 개의 차일드 노드들을 갖는 노드들을 포함하는 잔차 트리 구조를 결정하기 위해 본 개시물의 기법들을 수행할 수도 있다. 예를 들어, 역 변환 프로세싱 유닛 (256) 은 트리 구조에 기초하여 비디오 데이터의 CU 가 CU 의 TU들로 파티셔닝된다고 결정할 수도 있다. 이 예에서, 트리 구조에 기초하여 CU 를 CU 의 TU들로 파티셔닝하는 것의 부분으로서, 역 변환 프로세싱 유닛 (256) 은 트리 구조에서 노드가 트리 구조에서 정확히 2 개의 차일드 노드들을 갖는다고 결정할 수도 있다. 일부 예들에서, 역 변환 프로세싱 유닛 (256) 은, 트리 구조에서 제 2 노드가 트리 구조에서 정확히 4 개의 차일드 노드들을 갖는다고 결정할 수도 있다. 또한, 이 예에서, CU 의 TU들 중 적어도 하나에 대해, 역 변환 프로세싱 유닛 (256) 은 TU 에 대한 계수 블록에 변환을 적용하여 TU 에 대한 잔차 블록을 생성할 수도 있다.

[0408] PU 가 인트라 예측을 사용하여 인코딩되면, 인트라 예측 프로세싱 유닛 (266) 은 인트라 예측을 수행하여 PU 의 예측 블록들을 생성할 수도 있다. 인트라-예측 프로세싱 유닛 (266) 은 공간적으로-이웃하는 블록들의 샘플들에 기초하여 PU 의 예측 블록을 생성하도록 인트라 예측 모드를 사용할 수도 있다. 인트라-예측 프로세싱 유닛 (266) 은 비트스트림으로부터 획득된 하나 이상의 신택스 엘리먼트들에 기초하여 PU 에 대한 인트라 예측 모드를 결정할 수도 있다.

[0409] PU 가 인터 예측을 사용하여 인코딩되면, 엔트로피 디코딩 유닛 (250) 은 PU 에 대한 모션 정보를 결정할 수도 있다. 모션 보상 유닛 (264) 은, PU 의 모션 정보에 기초하여 하나 이상의 레퍼런스 블록들을 결정할 수도 있다. 모션 보상 유닛 (264) 은, 하나 이상의 레퍼런스 블록들에 기초하여, PU 에 대한 예측 블록들 (예를

들어, 예측 루마, Cb 및 Cr 블록들) 을 생성할 수도 있다.

[0410] 복원 유닛 (258) 은 CU 의 TU들에 대한 변환 블록들 (예를 들어, 루마, Cb 및 Cr 변환 블록들) 및 CU 의 PU들의 예측 블록들 (예를 들어, 루마, Cb 및 Cr 블록들), 즉 적용 가능한 바와 같이 인트라-예측 데이터 또는 인터-예측 데이터를 사용하여, CU 에 대한 코딩 블록들 (예를 들어, 루마, Cb 및 Cr 코딩 블록들) 을 복원할 수도 있다. 예를 들어, 복원 유닛 (258) 은 예측 블록들 (예를 들어, 루마, Cb 및 Cr 예측 블록들) 의 대응하는 샘플들에 변환 블록들 (예를 들어, 루마, Cb 및 Cr 변환 블록들) 의 샘플들을 추가하여, CU 의 코딩 블록들 (예를 들어, 루마, Cb 및 Cr 코딩 블록들) 을 복원할 수도 있다.

[0411] 필터 유닛 (260) 은 CU 의 코딩 블록들에 하나 이상의 필터들을 적용할 수도 있다. 예를 들어, 필터 유닛 (260) 은 CU 의 코딩 블록들과 연관된 블록킹 아티팩트들을 감소시키도록 디블록킹 동작을 수행할 수도 있다.

비디오 디코더 (30) 는 디코딩된 픽처 버퍼 (262) 에 CU 의 코딩 블록들을 저장할 수도 있다. 따라서, 디코딩된 픽처 버퍼 (262) 는 비디오 데이터의 디코딩된 블록들을 저장할 수도 있다. 디코딩된 픽처 버퍼 (262) 는 도 1 의 디스플레이 디바이스 (32) 와 같은 디스플레이 디바이스 상의 프리젠테이션, 인트라 예측, 및 후속의 모션 보상을 위해 레퍼런스 픽처들을 제공할 수도 있다. 예를 들어, 비디오 디코더 (30) 는 디코딩된 픽처 버퍼 (262) 에서의 블록들에 기초하여 다른 CU들의 PU들에 대한 인트라 예측 또는 인터 예측 동작들을 수행할 수도 있다.

[0412] 도 26 은 본 개시물의 기법에 따라 LM-기반 인코딩을 지원하는 예시의 비디오 인코더 (20) 를 예시하는 블록도이다. 도 26 의 예에서, 비디오 인코더 (20) 의 컴포넌트는 도 24 의 비디오 인코더 (20) 의 대응하는 컴포넌트들과 동일한 방식으로 동작한다. 그러나, 도 26 의 비디오 인코더 (20) 는 또한, LM-기반 인코딩 유닛 (222) 을 포함한다.

[0413] LM-기반 인코딩 유닛 (222) 은 본 개시물의 다른 곳에서 설명된 예들에 따라 LM 예측 인코딩을 수행할 수도 있다. 예를 들어, 역 양자화 유닛 (208), 역 변환 프로세싱 유닛 (210), 복원 유닛 (212), 및 필터 유닛 (214) 은 루마 레퍼런스 샘플들의 세트, 크로마 레퍼런스 샘플들의 세트를 복원할 수도 있고, 또한 비-스퀘어 PU 의 루마 샘플들을 복원할 수도 있다. LM-기반 인코딩 유닛 (222) 은, 비-스퀘어 PU 의 더 긴 사이드를 이웃하는 루마 레퍼런스 샘플들의 세트에서 루마 레퍼런스 샘플들의 총 수가 비-스퀘어 PU 의 더 짧은 사이드를 이웃하는 루마 레퍼런스 샘플들의 세트의 루마 레퍼런스 샘플들의 총 수와 동일하도록 루마 레퍼런스 샘플들의 세트를 다운-샘플링 또는 서브-샘플링할 수도 있다. 부가적으로, LM-기반 디코딩 유닛 (222) 은, 제 1 파라미터가 다음과 동일하도록 제 1 파라미터를 결정할 수도 있다:

[0414] $$\left(\sum y_i - \alpha \cdot \sum x_i\right) / I$$

[0415] 여기서, I 는 루마 레퍼런스 샘플들의 세트에서 레퍼런스 샘플들의 총 수이고, x_i 는 루마 레퍼런스 샘플들의 세트에서 i-번째 루마 레퍼런스 샘플이며, y_i 는 크로마 레퍼런스 샘플들의 세트에서 i-번째 크로마 레퍼런스 샘플이다. 비-스퀘어 PU 의 예측 크로마 블록의 각각의 개별의 크로마 샘플에 대해, LM-기반 인코딩 유닛 (222) 은, 개별의 크로마 샘플의 값이 제 2 파라미터 곱하기 개별의 크로마 샘플에 대응하는 개별의 복원된 루마 샘플, 플러스 제 1 파라미터와 동일하도록 개별의 크로마 샘플의 값을 결정할 수도 있고, 개별의 크로마 샘플에 대응하는 복원된 루마 샘플은 비-스퀘어 PU 의 복원된 루마 샘플들 중 하나이다. LM-기반 인코딩 유닛 (222) 은, 제 2 파라미터가 다음과 동일하도록 제 1 파라미터를 결정할 수도 있다:

[0416] $$\frac{I\sum x_i \cdot y_i - \sum x_i \cdot \sum y_i}{I\sum x_i \cdot x_i - \sum x_i \cdot \sum x_i}$$.

[0417] LM-기반 인코딩 유닛 (222) 은 잔차 생성 유닛 (202) 으로 예측 블록을 출력할 수도 있다. 잔차 생성 유닛 (202) 은 예측 블록 및 크로마 블록으로부터 잔차 블록을 생성한다. 결과의 잔차 블록은 변환 프로세싱 유닛 (103) 에 의해 변환되고, 양자화 유닛 (206) 에 의해 양자화되며, 엔트로피 인코딩 유닛 (218) 에 의해 엔트로피 인코딩된다. 결과는 그 후, 비트스트림을 통해 시그널링되고, 비디오 디코더 (30) 는 비트스트림에서의 정보를 사용하여 크로마 블록을 복원할 수도 있다.

[0418] 도 27 은 본 개시물의 기법에 따라 LM-기반 디코딩을 지원하는 예시의 비디오 디코더 (30) 를 예시하는 블록도이다. 도 27 의 예에서, 비디오 디코더 (30) 의 컴포넌트들은 도 27 의 비디오 디코더 (30) 의 대응하는 컴포넌트들과 동일한 방식으로 동작한다. 그러나, 도 27 의 비디오 디코더 (30) 는 LM-기반 디코딩 유닛

(265) 을 포함한다.

[0419] 본 개시물의 다양한 예들에 따르면, 비디오 디코더 (30) 는 본 개시물의 다른 곳에서 제공된 예들에 따라 LM-기반의 코딩을 수행하도록 구성될 수도 있다. 예를 들어, 역 양자화 유닛 (254), 역 변환 프로세싱 유닛 (256), 복원 유닛 (258), 및 필터 유닛 (260) 은 루마 레퍼런스 샘플들의 세트, 크로마 레퍼런스 샘플들의 세트를 복원할 수도 있고, 또한 비-스퀘어 PU 의 루마 샘플들을 복원할 수도 있다. LM-기반 디코딩 유닛 (265) 은, 비-스퀘어 PU 의 더 긴 사이드를 이웃하는 루마 레퍼런스 샘플들의 세트에서 루마 레퍼런스 샘플들의 총 수가 비-스퀘어 PU 의 더 짧은 사이드를 이웃하는 루마 레퍼런스 샘플들의 세트의 루마 레퍼런스 샘플들의 총 수와 동일하도록 루마 레퍼런스 샘플들의 세트를 다운-샘플링 또는 서브-샘플링할 수도 있다. 부가적으로, LM-기반 디코딩 유닛 (265) 은, 제 1 파라미터가 다음과 동일하도록 제 1 파라미터를 결정할 수도 있다:

[0420] $$\left(\sum y_i - \alpha \cdot \sum x_i\right) / I$$

[0421] 여기서, I 는 루마 레퍼런스 샘플들의 세트에서 레퍼런스 샘플들의 총 수이고, x_i 는 루마 레퍼런스 샘플들의 세트에서 i-번째 루마 레퍼런스 샘플이며, y_i 는 크로마 레퍼런스 샘플들의 세트에서 i-번째 크로마 레퍼런스 샘플이다. 비-스퀘어 PU 의 예측 크로마 블록의 각각의 개별의 크로마 샘플에 대해, LM-기반 디코딩 유닛 (266) 은, 개별의 크로마 샘플의 값이 제 2 파라미터 곱하기 개별의 크로마 샘플에 대응하는 개별의 복원된 루마 샘플, 플러스 제 1 파라미터와 동일하도록 개별의 크로마 샘플의 값을 결정할 수도 있고, 개별의 크로마 샘플에 대응하는 복원된 루마 샘플은 비-스퀘어 PU 의 복원된 루마 샘플들 중 하나이다. LM-기반 디코딩 유닛 (266) 은, 제 2 파라미터가 다음과 동일하도록 제 1 파라미터를 결정할 수도 있다:

[0422] $$\frac{I\sum x_i \cdot y_i - \sum x_i \cdot \sum y_i}{I\sum x_i \cdot x_i - \sum x_i \cdot \sum x_i}$$

[0423] LM-기반 디코딩 유닛 (265) 은 예측 블록을 복원 유닛 (258) 으로 출력할 수도 있다. 복원 유닛 (258) 은 또한, (예를 들어, 잔차 블록에 대한 비트스트림에서의 정보가 엔트로피 디코딩 유닛 (250) 으로 엔트로피 디코딩되고, 역 양자화 유닛 (254) 으로 역 양자화되고, 역 변환 프로세싱 유닛 (256) 으로 역 변환된 후에) 잔차 블록을 수신한다. 복원 유닛 (258) 은 예측 블록과 함께 잔차 블록을 추가하여 크로마 블록을 복원한다.

[0424] 도 28 은 본 개시물의 LM-기반의 코딩 기법에 따라 비디오 인코더 (20) 의 예시의 동작을 예시하는 플로우차트이다. 본 개시물의 플로우차트들은 예들로서 제공된다. 본 개시물의 범위 내의 다른 예들은 더 많은, 더 적은, 또는 상이한 액션들을 포함할 수도 있다. 본 개시물의 범위 내의 다른 예들은 상이한 순서들로 액션들을 포함하고 또는 병렬로 수행될 수도 있다.

[0425] 도 28 의 예에서, 비디오 인코더 (20) 는 비디오 데이터를 수신할 수도 있다 (300). 예를 들어, 비디오 인코더 (20) 는 비디오 소스 (18)(도 1), 또는 다른 소스로부터 비디오 데이터를 수신할 수도 있다.

[0426] 부가적으로, 도 28 의 예에서, 비디오 인코더 (20) 는 루마 레퍼런스 샘플들의 세트 및 크로마 레퍼런스 샘플들의 세트를 복원할 수도 있다 (302). 루마 레퍼런스 샘플들의 세트는 비디오 데이터의 현재 픽처의 비-스퀘어 루마 블록의 상단 사이드를 이웃하는 상부 루마 샘플들 및 비-스퀘어 루마 블록의 좌측 사이드를 이웃하는 좌측 루마 샘플들을 포함할 수도 있다. 일부 예들에서, 비-스퀘어 루마 블록은 비-스퀘어 PU 의 루마 예측 블록이다. 크로마 레퍼런스 샘플들의 세트는 비-스퀘어 크로마 블록의 상단 사이드를 이웃하는 크로마 샘플들 및 비-스퀘어 크로마 블록의 좌측 사이드를 이웃하는 크로마 샘플들을 포함할 수도 있다. 일부 예들에서, 비-스퀘어 루마 블록은 비-스퀘어 PU 의 루마 예측 블록이다.

[0427] 또한, 비디오 인코더 (20) 는 비-스퀘어 루마 블록의 루마 샘플들을 복원할 수도 있다 (304). 예를 들어, 비디오 인코더 (20) 는 본 개시물의 다른 곳에서 설명된 바와 같이 CU 에 대한 루마 잔차 샘플들을 생성할 수도 있다. 이 예에서, 비디오 인코더 (20) 는 비-스퀘어 루마 블록의 루마 예측 블록의 샘플들을 루마 잔차 샘플들의 대응하는 샘플들에 추가하여, 비-스퀘어 루마 블록의 루마 샘플들을 복원할 수도 있다.

[0428] 일부 예들에서, 비디오 인코더 (20) 는 비-스퀘어 루마 블록의 루마 샘플들을 다운-샘플링 또는 서브-샘플링할 수도 있다. 비-스퀘어 루마 블록의 루마 샘플들을 다운-샘플링 또는 서브-샘플링함으로써, 비디오 인코더 (20) 는 크로마 예측 블록 (예를 들어, 루마 블록과 동일한 PU 의 크로마 예측 블록) 의 각각의 크로마 샘플에 대한 하나의 루마 샘플을 갖는 루마 샘플들의 다운-샘플링된 또는 서브-샘플링된 세트를 획득할 수도 있다.

비디오 인코더 (20) 는, 현재 픽처의 컬러 포맷이 4:4:4 가 아니라는 결정에 응답하여 비-스퀘어 루마 블록의 루마 샘플들을 다운-샘플링 또는 서브-샘플링할 수도 있다.

[0429] 부가적으로, 비디오 인코더 (20) 는, 비-스퀘어 루마 블록의 더 긴 사이드를 이웃하는 루마 레퍼런스 샘플들의 세트에서 루마 레퍼런스 샘플들의 총 수가 비-스퀘어 루마 블록의 더 짧은 사이드를 이웃하는 루마 레퍼런스 샘플들의 세트의 루마 레퍼런스 샘플들의 총 수와 동일하도록 루마 레퍼런스 샘플들의 세트를 다운-샘플링 또는 서브-샘플링할 수도 있다 (306). 비디오 인코더 (20) 는 본 개시물의 다른 곳에서 설명된 기법들에 따라 루마 레퍼런스 샘플들의 세트를 다운-샘플링 또는 서브-샘플링할 수도 있다. 예를 들어, 비디오 인코더 (20) 는, 비-스퀘어 루마 블록의 더 긴 사이드를 이웃하는 루마 레퍼런스 샘플들의 세트가 비-스퀘어 루마 블록의 더 짧은 사이드를 이웃하는 루마 레퍼런스 샘플들의 세트의 루마 레퍼런스 샘플들의 총 수와 동일하도록 루마 레퍼런스 샘플들의 세트를 데시메이팅 (decimate) 할 수도 있다. 일부 예들에서, 비디오 인코더 (20) 는 좌측 레퍼런스 샘플들 또는 상부 레퍼런스 샘플들 중 어느 것이든 루마의 상단 경계 및 루마의 좌측 경계 중 더 긴 것에 대응하는 것을 다운-샘플링 또는 서브-샘플링하지만, 어느 것이든 루마 블록의 좌측 경계 및 루마 블록의 상단 경계 중 더 짧은 것을 다운-샘플링 또는 서브-샘플링하지 않을 수도 있다. 일부 예들에서, 루마 레퍼런스 샘플들의 세트에서 레퍼런스 샘플들의 총 수는 2^m 과 동일하고, 여기서 m 은 비-스퀘어 루마 블록의 높이 및/또는 폭에 의존하는 정수이다.

[0430] 일부 예들에서, 비디오 인코더 (20) 는, 비-스퀘어 크로마 블록의 더 긴 사이드를 이웃하는 크로마 레퍼런스 샘플들의 세트에서 크로마 레퍼런스 샘플들의 총 수가 비-스퀘어 크로마 블록의 더 짧은 사이드를 이웃하는 크로마 레퍼런스 샘플들의 세트의 크로마 레퍼런스 샘플들의 총 수와 동일하도록 크로마 레퍼런스 샘플들의 세트를 다운-샘플링 또는 서브-샘플링할 수도 있다.

[0431] 도 28 의 액션 (308) 에서, 비디오 인코더 (20) 는, 제 1 파라미터가 다음에 기초하도록 제 1 파라미터 (β) 를 결정할 수도 있다:

[0432] $$\left(\sum y_i - \alpha \cdot \sum x_i\right) / I$$

[0433] 상기 등식에서, I 는 루마 레퍼런스 샘플들의 세트에서 레퍼런스 샘플들의 총 수이고, x_i 는 루마 레퍼런스 샘플들의 세트에서 i-번째 루마 레퍼런스 샘플이며, y_i 는 크로마 레퍼런스 샘플들의 세트에서 i-번째 크로마 레퍼런스 샘플이다. 비디오 인코더 (20) 는, 비디오 인코더 (20) 가 추가적인 상수들 또는 계수들을 포함하는 것과 같이, 상기 수식에 대한 변형 또는 상기 수식을 직접적으로 사용한다는 점에서 상기 수식에 기초하여 제 1 파라미터를 결정할 수도 있다.

[0434] 일부 예들에서, 비디오 인코더 (20) 는 또한, 제 2 파라미터가 다음에 기초하도록 제 2 파라미터 (α) 를 결정할 수도 있다:

[0435] $$\frac{I\sum x_i \cdot y_i - \sum x_i \cdot \sum y_i}{I\sum x_i \cdot x_i - \sum x_i \cdot \sum x_i}$$

[0436] 비디오 인코더 (20) 는, 비디오 인코더 (20) 가 추가적인 상수들 또는 계수들을 포함하는 것과 같이, 상기 수식에 대한 변형 또는 상기 수식을 직접적으로 사용한다는 점에서 상기 수식에 기초하여 제 2 파라미터를 결정할 수도 있다.

[0437] 부가적으로, 도 28 의 예에서, 예측 크로마 블록의 각각의 개별의 크로마 샘플에 대해, 비디오 인코더 (20) 는, 개별의 크로마 샘플의 값이 제 2 파라미터 곱하기 개별의 크로마 샘플에 대응하는 개별의 복원된 루마 샘플, 플러스 제 1 파라미터와 동일하도록 개별의 크로마 샘플의 값을 결정할 수도 있다 (310). 개별의 크로마 샘플에 대응하는 복원된 루마 샘플은 비-스퀘어 루마 블록의 복원된 루마 샘플들 중 하나이다.

[0438] 또한, 비디오 인코더 (20) 는, 예측 크로마 블록에 기초하여, 잔차 데이터를 획득할 수도 있다 (312). 예를 들어, 비디오 인코더 (20) 는 잔차 데이터의 크로마 샘플들의 값들이 CU 의 크로마 코딩 블록의 샘플들과 비-스퀘어 예측 블록의 크로마 블록의 샘플들 간의 차이들과 동일하다고 결정할 수도 있다.

[0439] 부가적으로, 비디오 인코더 (20) 는, 비디오 데이터의 인코딩된 표현을 포함하는 비트스트림에서, 잔차 데이터를 나타내는 데이터를 포함할 수도 있다 (314). 예를 들어, 비디오 인코더 (20) 는 하나 이상의 변환들을 잔차 데이터에 적용하여 하나 이상의 계수 블록들을 생성하고; 계수 블록들을 양자화하고; 변환 계수가 넌-제로

인지 여부, 변환 계수가 1 보다 큰지 여부, 변환 계수가 2 보다 큰지 여부, 변환 계수의 부호, 및 변환 계수에 대한 나머지를 나타내는 신택스 엘리먼트들을 생성할 수도 있다. 이 예에서, 비디오 인코더 (20) 는 이들 신택스 엘리먼트들 중 하나 이상에 CABAC 코딩을 적용하고, 결과의 값들을 비트스트림에 포함할 수도 있다.

[0440] 도 29 는 본 개시물의 LM-기반의 코딩 기법에 따라 비디오 디코더 (30) 의 예시의 동작을 예시하는 플로우차트이다. 도 29 의 예에서, 비디오 디코더 (30) 는 비디오 데이터의 인코딩된 표현을 포함하는 비트스트림을 수신한다 (350).

[0441] 또한, 도 29 의 예에서, 비디오 디코더 (30) 는 루마 레퍼런스 샘플들의 세트 및 크로마 레퍼런스 샘플들의 세트를 복원할 수도 있다 (352). 루마 레퍼런스 샘플들의 세트는 비디오 데이터의 현재 픽처의 비-스퀘어 루마 블록의 상단 사이드를 이웃하는 상부 루마 샘플들 및 비-스퀘어 루마 블록의 좌측 사이드를 좌측 이웃하는 루마 샘플들을 포함한다. 일부 예들에서, 비-스퀘어 루마 블록은 비-스퀘어 PU 의 루마 예측 블록이다. 크로마 레퍼런스 샘플들의 세트는 비-스퀘어 크로마 블록의 상단 사이드를 이웃하는 크로마 샘플들 및 비-스퀘어 크로마 블록의 좌측 사이드를 이웃하는 크로마 샘플들을 포함한다. 일부 예들에서, 비-스퀘어 루마 블록은 비-스퀘어 PU 의 루마 예측 블록이다.

[0442] 비디오 디코더 (30) 는 비-스퀘어 루마 블록의 루마 샘플들을 복원할 수도 있다 (354). 예를 들어, 비-스퀘어 루마 블록의 루마 샘플들을 복원하는 것의 부분으로서, 비디오 디코더 (30) 는 인트라 예측 또는 인터 예측을 사용하여 비-스퀘어 루마 블록에 대한 루마 예측 블록을 생성할 수도 있다. 부가적으로, 이 예에서, 비디오 디코더 (30) 는 비-스퀘어 루마 블록에 대한 루마 예측 블록의 샘플들을 대응하는 잔차 샘플들에 추가하여 루마 샘플들을 복원할 수도 있다.

[0443] 일부 예들에서, 비디오 디코더 (30) 는 비-스퀘어 루마 블록의 루마 샘플들을 다운-샘플링 또는 서브-샘플링할 수도 있다. 비-스퀘어 루마 블록의 루마 샘플들을 다운-샘플링 또는 서브-샘플링함으로써, 비디오 디코더 (30) 는 크로마 예측 블록 (예를 들어, 루마 블록과 동일한 PU 의 크로마 예측 블록) 의 각각의 크로마 샘플에 대한 하나의 루마 샘플을 갖는 루마 샘플들의 다운-샘플링된 또는 서브-샘플링된 세트를 획득할 수도 있다. 비디오 디코더 (30) 는, 현재 픽처의 컬러 포맷이 4:4:4 가 아니라는 결정에 응답하여 비-스퀘어 루마 블록의 루마 샘플들을 다운-샘플링 또는 서브-샘플링할 수도 있다.

[0444] 부가적으로, 도 29 의 예에서, 비디오 디코더 (30) 는, 비-스퀘어 루마 블록의 더 긴 사이드를 이웃하는 루마 레퍼런스 샘플들의 세트에서 루마 레퍼런스 샘플들의 총 수가 비-스퀘어 루마 블록의 더 짧은 사이드를 이웃하는 루마 레퍼런스 샘플들의 세트의 루마 레퍼런스 샘플들의 총 수와 동일하도록 루마 레퍼런스 샘플들의 세트를 다운-샘플링 또는 서브-샘플링할 수도 있다 (356). 비디오 디코더 (30) 는 본 개시물의 다른 곳에서 설명된 기법들에 따라 루마 레퍼런스 샘플들의 세트를 다운-샘플링 또는 서브-샘플링할 수도 있다. 예를 들어, 비디오 디코더 (30) 는, 비-스퀘어 루마 블록의 더 긴 사이드를 이웃하는 루마 레퍼런스 샘플들의 세트가 비-스퀘어 루마 블록의 더 짧은 사이드를 이웃하는 루마 레퍼런스 샘플들의 세트의 루마 레퍼런스 샘플들의 총 수와 동일하도록 루마 레퍼런스 샘플들의 세트를 데시메이팅할 수도 있다. 일부 예들에서, 비디오 디코더 (30) 는 좌측 레퍼런스 샘플들 또는 상부 레퍼런스 샘플들 중 어느 것이든 루마 블록의 상단 경계 및 루마 블록의 좌측 경계 중 더 긴 것에 대응하는 것을 다운-샘플링 또는 서브-샘플링하지만, 어느 것이든 루마 블록의 좌측 경계 및 루마 블록의 상단 경계 중 더 짧은 것을 다운-샘플링 또는 서브-샘플링하지 않을 수도 있다. 일부 예들에서, 루마 레퍼런스 샘플들의 세트에서 레퍼런스 샘플들의 총 수는 2^{m} 과 동일하고, 여기서 m 은 비-스퀘어 루마 블록의 높이 및/또는 폭에 의존하는 정수이다.

[0445] 일부 예들에서, 비디오 디코더 (30) 는 또한, 비-스퀘어 크로마 블록의 더 긴 사이드를 이웃하는 크로마 레퍼런스 샘플들의 세트에서 크로마 레퍼런스 샘플들의 총 수가 비-스퀘어 크로마 블록의 더 짧은 사이드를 이웃하는 크로마 레퍼런스 샘플들의 세트의 크로마 레퍼런스 샘플들의 총 수와 동일하도록 크로마 레퍼런스 샘플들의 세트를 다운-샘플링 또는 서브-샘플링할 수도 있다.

[0446] 부가적으로, 도 29 의 액션 (358) 에서, 비디오 디코더 (30) 는, 제 1 파라미터가 다음에 기초하도록 제 1 파라미터 (β) 를 결정할 수도 있다:

[0447] $$\left(\sum y_i - \alpha \cdot \sum x_i\right)/I$$

[0448] 상기 등식에서, I 는 루마 레퍼런스 샘플들의 세트에서 레퍼런스 샘플들의 총 수이고, x_i 는 루마 레퍼런스 샘플

들의 세트에서 i-번째 루마 레퍼런스 샘플이며, y_i 는 크로마 레퍼런스 샘플들의 세트에서 i-번째 크로마 레퍼런스 샘플이다. 본 개시물에서, 비디오 인코더 (20) 및/또는 비디오 디코더 (30) 는, 비디오 인코더 (20) 및/또는 비디오 디코더 (30) 가 추가적인 상수들 또는 계수들을 포함하는 것과 같이, 수식에 대한 변형 또는 수식을 직접적으로 사용한다는 점에서 수식에 기초하여 값을 결정할 수도 있다.

[0449] 일부 예들에서, 비디오 디코더 (30) 는 또한, 제 2 파라미터가 다음에 기초하도록 제 2 파라미터 (α) 를 결정할 수도 있다:

[0450]
$$\frac{I\sum x_i \cdot y_i - \sum x_i \cdot \sum y_i}{I\sum x_i \cdot x_i - \sum x_i \cdot \sum x_i}$$

[0451] 도 29 의 예에서, 예측 크로마 블록의 각각의 개별의 크로마 샘플에 대해, 비디오 디코더 (30) 는, 개별의 크로마 샘플의 값이 제 2 파라미터 곱하기 개별의 크로마 샘플에 대응하는 개별의 복원된 루마 샘플, 플러스 제 1 파라미터와 동일하도록 개별의 크로마 샘플의 값을 결정할 수도 있다 (360). 이 예에서, 개별의 크로마 샘플에 대응하는 복원된 루마 샘플은 비-스퀘어 루마 블록의 복원된 루마 샘플들 중 하나이다.

[0452] 또한, 비디오 디코더 (30) 는, 예측 크로마 블록에 적어도 부분적으로 기초하여 코딩 블록을 복원할 수도 있다 (362). 예를 들어, 비디오 디코더 (30) 는 예측 크로마 블록의 샘플들을 CU 의 대응하는 잔차 크로마 샘플들에 추가하여, CU 의 코딩 블록의 샘플들을 결정할 수도 있다.

[0453] 도 30 은 본 개시물의 양자화 기법에 따른, 비디오 인코더 (20) 의 예시의 동작을 예시하는 플로우차트이다. 도 30 의 예에서, 비디오 인코더 (20) 는 비디오 데이터를 수신한다 (400). 또한, 비디오 인코더 (20) 는, 잔차 블록의 각각의 잔차 샘플이 CU 의 코딩 블록의 대응하는 샘플들과 CU 의 PU 의 예측 블록 간의 차이를 나타내도록 CU 의 비-스퀘어 TU 에 대한 잔차 블록을 생성할 수도 있다 (402).

[0454] 비디오 인코더 (20) 는 변환 계수들의 블록을 생성하기 위해 잔차 블록에 변환을 적용할 수도 있다 (404). 예를 들어, 비디오 인코더 (20) 는 DCT 변환을 잔차 블록에 적용할 수도 있다. 또한, 비디오 인코더 (20) 는, 변환 계수들의 블록의 각각의 개별의 변환 계수가 개별의 탈양자화된 변환 계수 곱하기 $\sqrt{2}$ 의 근사치에 기초하도록 변환 계수들을 수정할 수도 있다 (406). 예를 들어, 비디오 인코더 (20) 는, 각각의 개별의 변환 계수가 개별의 변환 계수의 원래의 값 곱하기 $\sqrt{2}$ 의 근사치와 동일하도록 변환 계수들을 수정할 수도 있다. 본 개시물에서, $\sqrt{2}$ 의 근사치는 $\sqrt{2}$ 의 표현 (예를 들어, $\sqrt{2}$ 의 부동 소수점 표현) 일 수도 있다. 일부 예들에서, 각각의 개별의 변환 계수가 개별의 변환 계수의 원래의 값 곱하기 $\sqrt{2}$ 의 근사치와 동일하도록 변환 계수들을 수정하는 것은 하나 이상의 수학적 연산들을 수행하여 변환 계수 곱하기 $\sqrt{2}$ 을 근사화하는 값들을 결정하는 것을 포함할 수도 있다.

[0455] 비-스퀘어 TU 가 사이즈 K×L 을 갖는 일부 예들에서, 잔차 블록에 변환을 적용하는 것의 부분으로서, 비디오 인코더 (20) 는, 잔차 블록에 사이즈 N×N 를 갖는 변환을 적용할 수도 있고, 여기서 $\log_2(N*N) = ((\log_2(K) + \log_2(L)) >> 1) << 1)$ 및 $((\log_2(K) + \log_2(L))$ 는 홀수이다. 예를 들어, 비디오 인코더 (20) 는 상기의 등식 (18) 에 도시된 바와 같이 잔차 블록의 로우들 및 컬럼들에 N-포인트 1-차원 DCT 변환들을 적용할 수도 있다.

[0456] 또한, 도 30 의 예에서, 변환 계수들을 수정한 후에, 비디오 인코더 (20) 는 CU 의 비-스퀘어 예측 블록의 수정된 변환 계수들에 양자화 프로세스를 적용할 수도 있다 (408). 예를 들어, 비디오 인코더 (20) 는 상기의 등식 (22) 에서 설명된 바와 같이 수정된 변환 계수들을 양자화할 수도 있다.

[0457] 비디오 인코더 (20) 는, 비디오 데이터의 인코딩된 표현을 포함하는 비트스트림에서, 양자화된 변환 계수들에 기초하는 데이터를 포함할 수도 있다 (410). 예를 들어, 비디오 인코더 (20) 는 양자화된 변환 계수가 넌-제로인지 여부, 양자화된 변환 계수가 1 보다 큰지 여부, 양자화된 변환 계수가 2 보다 큰지 여부, 양자화된 변환 계수의 부호, 및 양자화된 변환 계수에 대한 나머지를 나타내는 신택스 엘리먼트들을 생성할 수도 있다.

이 예에서, 비디오 인코더 (20) 는 이들 신택스 엘리먼트들 중 하나 이상에 CABAC 코딩을 적용하고, 결과의 값들을 비트스트림에 포함할 수도 있다.

[0458] 비-스퀘어 TU 가 사이즈 K×L 을 갖는 일부 예들에서, 비디오 인코더 (20) 는, 홀수인 (($\log_2(K) + \log_2(L)$) 에 기초하여, 탈양자화된 변환 계수들의 각각의 개별의 탈양자화된 변환 계수가 개별의 탈양자화된 변환 계수 곱하기 $\sqrt{2}$ 의 근사치에 기초하도록 탈양자화된 변환 계수들을 수정할 수도 있다. 일부 예들에서, (($\log_2(K) + \log_2(L)$) 이 짝수인 경우, 비디오 인코더 (20) 는, 탈양자화된 변환 계수들의 각각의 개별의 탈양자화된 변환 계수가 개별의 탈양자화된 변환 계수 곱하기 $\sqrt{2}$ 의 근사치에 기초하도록 탈양자화된 변환 계수들을 수정하지 않는다.

[0459] 도 31 은 본 개시물의 양자화 기법에 따른, 비디오 디코더 (30) 의 예시의 동작을 예시하는 플로우차트이다. 도 31 의 예에서, 비디오 디코더 (30) 는 비디오 데이터의 인코딩된 표현을 포함하는 비트스트림을 수신할 수도 있다 (450). 또한, 비디오 디코더 (30) 는 비디오 데이터의 CU 의 비-스퀘어 TU 의 변환 계수들에 탈양자화 프로세스를 적용할 수도 있다 (452). 예를 들어, 비디오 디코더 (30) 는 상기의 등식 (24) 를 적용함으로써 변환 계수들을 탈양자화 (즉, 역 양자화) 할 수도 있다.

[0460] 변환 계수들에 탈양자화 프로세스를 적용한 후에, 비디오 디코더 (30) 은, 탈양자화된 변환 계수들의 각각의 개별의 탈양자화된 변환 계수가 개별의 탈양자화된 변환 계수 나누기 $\sqrt{2}$ 의 근사치에 기초하도록 탈양자화된 변환 계수들을 수정할 수도 있다 (454). 예를 들어, 비디오 디코더 (30) 는, 각각의 개별의 수정된 변환 계수가 변환 계수 나누기 $\sqrt{2}$ 의 근사치와 동일하다고 결정할 수도 있다. 본 개시물에서, $\sqrt{2}$ 의 근사치는 $\sqrt{2}$ 의 표현 (예를 들어, $\sqrt{2}$ 의 부동 소수점 표현) 일 수도 있다. 일부 예들에서, 각각의 개별의 변환 계수가 개별의 변환 계수의 원래의 값 곱하기 $\sqrt{2}$ 의 근사치와 동일하도록 변환 계수들을 수정하는 것은 하나 이상의 수학적 연산들을 수행하여 변환 계수 나누기 $\sqrt{2}$ 을 근사화하는 값들을 결정하는 것을 포함할 수도 있다.

[0461] 또한, 비디오 디코더 (30) 는 잔차 블록을 복원하기 위해, 수정된 탈양자화된 변환 계수들에 역 변환을 적용할 수도 있다 (456). 예를 들어, 비디오 디코더 (30) 는 변환 매트릭스 C (또는 정수 정확도로 표현된 그 근사치) 의 트랜스포즈로 등식 (18) 을 적용하여 수정된 탈양자화된 변환 계수들에 역 변환을 적용할 수도 있다. 비-스퀘어 TU 가 사이즈 K×L 를 갖는 일부 예들에서, 잔차 블록에 변환을 적용하는 것의 부분으로서, 비디오 디코더 (30) 는 잔차 블록에, 사이즈 N×N 을 갖는 변환을 적용할 수도 있고, 여기서 $\log_2(N*N) = (((\log_2(K) + \log_2(L)) >> 1) << 1)$ 및 (($\log_2(K) + \log_2(L)$) 는 홀수이다. 비디오 디코더 (30) 는 예측 블록의 샘플들을 CU 의 TU 에 대한 잔차 블록의 대응하는 샘플들에 추가함으로써 코딩 블록의 샘플들을 복원할 수도 있다 (458).

[0462] 비-스퀘어 TU 가 사이즈 K×L 을 갖는 일부 예들에서, 비디오 디코더 (30) 는, 홀수인 (($\log_2(K) + \log_2(L)$) 에 기초하여, 탈양자화된 변환 계수들의 각각의 개별의 탈양자화된 변환 계수가 개별의 탈양자화된 변환 계수 나누기 $\sqrt{2}$ 의 근사치에 기초하도록 탈양자화된 변환 계수들을 수정할 수도 있다. 일부 예들에서, (($\log_2(K) + \log_2(L)$) 이 짝수인 경우, 비디오 디코더 (30) 는, 탈양자화된 변환 계수들의 각각의 개별의 탈양자화된 변환 계수가 개별의 탈양자화된 변환 계수 나누기 $\sqrt{2}$ 의 근사치에 기초하도록 탈양자화된 변환 계수들을 수정하지 않는다.

[0463] 도 32 는 IC 를 사용하는 본 개시물의 기법에 따른, 비디오 인코더 (20) 의 예시의 동작을 예시하는 플로우차트이다. 도 32 의 예에서, 비디오 인코더 (20) 는 비디오 데이터를 수신한다 (500). 예를 들어, 비디오 인코더 (20) 는 비디오 소스 (18)(도 1), 또는 다른 곳으로부터 비디오 데이터를 수신할 수도 있다. 또한, 비디오 인코더 (20) 는 비디오 데이터의 현재 픽처의 현재 CU 의 현재 PU 의 비-스퀘어 예측 블록을 생성하기

위해 IC 를 사용할 수도 있다 (502).

[0464] 비-스퀘어 예측 블록을 생성하기 위해 IC 를 사용하는 것의 부분으로서, 비디오 인코더 (20) 는, 현재 PU 의 벡터에 기초하여 레퍼런스 픽처에서 레퍼런스 블록을 결정할 수도 있다 (504). 일부 예들에서, 벡터는 디스패리티 벡터이고 레퍼런스 픽처는 인터-뷰 레퍼런스 픽처이다. 일부 예들에서, 벡터는 모션 벡터이고 레퍼런스 픽처는 시간 모션 벡터이다. 레퍼런스 블록 및 비-스퀘어 예측 블록은 동일한 사이즈 및 형상일 수도 있다. 일부 예들에서, 현재 PU 의 벡터에 기초하여 레퍼런스 블록을 결정하기 위해, 비디오 인코더 (20) 는 비-스퀘어 예측 블록의 상단-좌측 코너의 x 좌표에 벡터의 수평 컴포넌트를 추가하고 비-스퀘어 예측 블록의 상단-좌측 코너의 y 좌표에 벡터의 수직 컴포넌트를 추가함으로써 레퍼런스 블록의 상단-좌측 코너의 레퍼런스 픽처에서의 포지션을 결정할 수도 있다. 이 예에서, 레퍼런스 블록의 상단-좌측 코너의 표시된 포지션이 정수 픽셀의 레퍼런스 픽처에서의 포지션을 나타내지 않으면, 비디오 인코더 (20) 는 레퍼런스 블록의 샘플들을 보간하여 레퍼런스 블록을 결정할 수도 있다.

[0465] 또한, 도 32 의 예에서, 비-스퀘어 예측 블록을 생성하기 위해 IC 를 사용하는 것의 부분으로서, 비디오 인코더 (20) 는 제 1 서브-샘플링 비로 레퍼런스 샘플들의 제 1 서브-샘플링된 세트를 생성하기 위해 레퍼런스 샘플들의 제 1 세트를 서브-샘플링할 수도 있다 (506). 이 예에서, 레퍼런스 샘플들의 제 1 세트에서의 레퍼런스 샘플들의 총 수는 2^m 과 동일하지 않고, 레퍼런스 샘플들의 제 1 서브-샘플링된 세트에서의 레퍼런스 샘플들의 총 수는 2^m 과 동일하다. 또한, 이 예에서, 레퍼런스 샘플들의 제 1 세트는 비-스퀘어 예측 블록의 좌측 사이드 및 상단 사이드를 따라 비-스퀘어 예측 블록 밖의 샘플들을 포함하고, m 은 정수이다.

[0466] 부가적으로, 비-스퀘어 예측 블록을 생성하기 위해 IC 를 사용하는 것의 부분으로서, 비디오 인코더 (20) 는 제 2 서브-샘플링 비로 레퍼런스 샘플들의 제 2 서브-샘플링된 세트를 생성하기 위해 레퍼런스 샘플들의 제 2 세트를 서브-샘플링할 수도 있다 (508). 제 1 서브-샘플링 비는 제 2 서브-샘플링 비와 동일하거나 또는 상이할 수도 있다. 이 예에서, 레퍼런스 샘플들의 제 2 세트에서의 레퍼런스 샘플들의 총 수는 2^m 과 동일하지 않고, 레퍼런스 샘플들의 제 2 서브-샘플링된 세트에서의 레퍼런스 샘플들의 총 수는 2^m 과 동일하다. 또한, 이 예에서, 레퍼런스 샘플들의 제 2 세트는 레퍼런스 블록의 좌측 사이드 및 상단 사이드를 따라 레퍼런스 블록 밖의 샘플들을 포함한다.

[0467] 액션들 (506) 및 (508) 에서, 비디오 인코더 (20) 는 다양한 방식들로 서브-샘플링을 수행할 수도 있다. 예를 들어, 비디오 인코더 (20) 는 데시메이션을 사용하여 서브-샘플링을 수행할 수도 있다. 비디오 인코더 (20) 가 데시메이션을 사용하여 서브-샘플링을 수행하는 예들에서, 비디오 인코더 (20) 는 규칙적인 인터벌들로 (예를 들어, 2 샘플들마다 한번씩) 샘플들을 제거하여 나머지 샘플들의 값을 변화시키지 않고 샘플들의 수를 감소시킬 수도 있다. 따라서, 이 예에서, 비디오 인코더 (20) 는, 레퍼런스 샘플들의 제 1 서브-샘플링된 세트를 생성하기 위해 레퍼런스 샘플들의 제 1 세트를 데시메이팅하는 것; 및 레퍼런스 샘플들의 제 2 서브-샘플링된 세트를 생성하기 위해 레퍼런스 샘플들의 제 2 세트를 데시메이팅하는 것 중 적어도 하나를 수행할 수도 있다.

[0468] 다른 예에서, 비디오 인코더 (20) 는 보간을 사용하여 서브-샘플링을 수행할 수도 있다. 비디오 인코더 (20) 가 보간을 사용하여 서브-샘플링을 수행하는 예들에서, 인접한 샘플들의 개별의 쌍들에 대해, 비디오 인코더 (20) 는 개별의 쌍의 샘플들 간의 값을 보간할 수도 있고 샘플들의 서브-샘플링된 세트에서 보간된 값을 포함할 수도 있다. 따라서, 이 예에서, 비디오 인코더 (20) 는, 레퍼런스 샘플들의 제 1 서브-샘플링된 세트를 생성하기 위해 레퍼런스 샘플들의 제 1 세트의 보간된 샘플링을 수행하는 것; 및 레퍼런스 샘플들의 제 2 서브-샘플링된 세트를 생성하기 위해 레퍼런스 샘플들의 제 2 세트의 보간된 샘플링을 수행하는 것 중 적어도 하나를 수행할 수도 있다.

[0469] 다른 예에서, 비디오 인코더 (20) 는 비트스트림에서 신택스 엘리먼트에 의해 표시된 서브-샘플링 방법을 사용하여 서브-샘플링을 수행할 수도 있다. 따라서, 이 예에서, 비디오 인코더 (20) 는, 비트스트림에서, 서브-샘플링 방법을 표시하는 신택스 엘리먼트를 포함할 수도 있다. 이 예에서, 비디오 인코더 (20) 는, 레퍼런스 샘플들의 제 1 서브-샘플링된 세트를 생성하기 위해 레퍼런스 샘플들의 제 1 세트를 서브-샘플링하기 위한 표시된 서브-샘플링 방법을 사용하는 것; 및 레퍼런스 샘플들의 제 2 서브-샘플링된 세트를 생성하기 위해 레퍼런스 샘플들의 제 2 세트를 서브-샘플링하기 위한 표시된 서브-샘플링 방법을 사용하는 것 중 적어도 하나를 수행할 수도 있다.

[0470] 다른 예에서, 비디오 인코더 (20) 는, 현재 PU 의 사이즈에 기초하여, 서브-샘플링 방법을 결정할 수도 있다.

이 예에서, 비디오 인코더 (20) 는, 레퍼런스 샘플들의 제 1 서브-샘플링된 세트를 생성하기 위해 레퍼런스 샘플들의 제 1 세트를 서브-샘플링하기 위한 결정된 서브-샘플링 방법을 사용하는 것; 및 레퍼런스 샘플들의 제 2 서브-샘플링된 세트를 생성하기 위해 레퍼런스 샘플들의 제 2 세트를 서브-샘플링하기 위해 결정된 서브-샘플링 방법을 사용하는 것 중 적어도 하나를 수행할 수도 있다.

[0471] 비-스퀘어 예측 블록을 생성하기 위해 IC 를 사용하는 것의 부분으로서, 도 32 의 액션 (510) 에서, 비디오 인코더 (20) 는 레퍼런스 샘플들의 제 1 서브-샘플링된 세트에서의 레퍼런스 샘플들의 총 수 및 레퍼런스 샘플들의 제 2 서브-샘플링된 세트에서의 레퍼런스 샘플들의 총 수, 레퍼런스 샘플들의 제 1 서브-샘플링된 세트, 및 레퍼런스 샘플들의 제 2 서브-샘플링된 세트에 기초하여 제 1 IC 파라미터를 결정할 수도 있다. 예를 들어, 비디오 인코더 (20) 는, 제 1 IC 파라미터가 다음에 기초하도록 제 1 IC 파라미터를 결정할 수도 있다:

[0472] $$\frac{\sum_{i=0}^{2N-1} \mathrm{Rec_{neigh}(i)} - a \cdot \sum_{i=0}^{2N-1} \mathrm{Rec_{refneigh}(i)}}{2N}$$

[0473] 상기 등식에서, 2N 은 레퍼런스 샘플들의 제 1 서브-샘플링된 세트에서의 레퍼런스 샘플들의 총 수 및 레퍼런스 샘플들의 제 2 서브-샘플링된 세트에서의 레퍼런스 샘플들의 총 수를 가리키고, $Rec_{neigh}(i)$ 는 레퍼런스 샘플들의 제 1 서브-샘플링된 세트에서의 i-번째 레퍼런스 샘플을 가리키며, $Rec_{refneigh}(i)$ 는 레퍼런스 샘플들의 제 2 서브-샘플링된 세트에서의 i-번째 레퍼런스 샘플을 가리킨다.

[0474] 일부 예들에서, 비디오 인코더 (20) 는 제 2 IC 파라미터가 다음에 기초하도록 제 2 IC 파라미터를 결정할 수도 있다:

[0475] $$\frac{2N \cdot \sum_{i=0}^{2N-1} Rec_{neigh}(i) \cdot Rec_{refneigh}(i) - \sum_{i=0}^{2N-1} Rec_{neigh}(i) \cdot \sum_{i=0}^{2N-1} Rec_{refneigh}(i)}{2N \cdot \sum_{i=0}^{2N-1} Rec_{refneigh}(i) \cdot Rec_{neigh}(i) - \left(\sum_{i=0}^{2N-1} Rec_{refneigh}(i)\right)^2}$$

[0476] 또한, 비-스퀘어 예측 블록을 생성하기 위해 IC 를 사용하는 것의 부분으로서, 도 32 의 액션 (512) 에서, 비디오 인코더 (20) 는 제 1 IC 파라미터에 기초하여 비-스퀘어 예측 블록의 샘플을 결정할 수도 있다. 예를 들어, 샘플은 현재 픽처의 상단-좌측 코너에 대한 좌표들 (i, j) 에 있을 수도 있고, 비디오 인코더 (20) 는 샘플이 다음에 기초하도록 샘플을 결정할 수도 있다:

[0477] $$a * r(i + dv_x, j + dv_y + b)$$

[0478] 상기 등식에서, b 는 제 1 IC 파라미터이고, a 는 제 2 IC 파라미터이고, r 은 레퍼런스 픽처이고, dv_x 는 현재 PU 의 벡터 (예를 들어, 디스패리티 벡터, 모션 벡터) 의 수평 컴포넌트이며, dv_y 는 현재 PU 의 벡터의 수직 컴포넌트이다.

[0479] 도 32 의 예에서, 비디오 인코더 (20) 는 비-스퀘어 예측 블록에 기초하여 잔차 데이터를 생성할 수도 있다 (514). 예를 들어, 비디오 인코더 (20) 는 잔차 데이터의 샘플들이 현재 CU 의 코딩 블록의 샘플들과 비-스퀘어 예측 블록의 샘플들 간의 차이들과 동일하도록 잔차 데이터를 생성할 수도 있다. 부가적으로, 비디오 인코더 (20) 는 잔차 데이터에 기초하는 데이터를 포함하는 비트스트림을 출력할 수도 있다 (516). 예를 들어, 비디오 인코더 (20) 는 잔차 데이터를 나타내는 엔트로피 인코딩된 신택스 엘리먼트들 (예를 들어, 1 보다 큰, 2 보다 큰, 나머지 등을 나타내는 신택스 엘리먼트들) 을 포함하는 비트스트림을 출력할 수도 있다.

[0480] 도 33 은 IC 를 사용하여 본 개시물의 기법들에 따라, 비디오 데이터를 인코딩하기 위한 비디오 디코더 (30) 의 예시의 동작을 예시하는 플로우차트이다. 도 32 의 예에서, 비디오 디코더 (30) 는 비디오 데이터의 인코딩된 표현을 포함하는 비트스트림을 수신한다 (550). 또한, 비디오 디코더 (30) 는 비디오 데이터의 현재 픽처의 현재 CU 의 현재 PU 의 비-스퀘어 예측 블록을 생성하기 위해 IC 를 사용할 수도 있다 (552).

[0481] 비-스퀘어 예측 블록을 생성하기 위해 IC 를 사용하는 것의 부분으로서, 비디오 디코더 (30) 는, 현재 PU 의 벡터에 기초하여 레퍼런스 픽처에서 레퍼런스 블록을 결정할 수도 있다 (554). 일부 예들에서, 벡터는 디스패리티 벡터이고 레퍼런스 픽처는 인터-뷰 레퍼런스 픽처이다. 일부 예들에서, 벡터는 모션 벡터이고 레퍼런스 픽처는 시간 모션 벡터이다. 레퍼런스 블록 및 비-스퀘어 예측 블록은 동일한 사이즈 및 형상이다. 현재 PU 의 디스패리티 벡터에 기초하여 레퍼런스 블록을 결정하기 위해, 비디오 디코더 (30) 는 비-스퀘어 예

측 블록의 상단-좌측 코너의 x 좌표에 벡터의 수평 컴포넌트를 추가하고 비-스퀘어 예측 블록의 상단-좌측 코너의 y 좌표에 벡터의 수직 컴포넌트를 추가함으로써 레퍼런스 블록의 상단-좌측 코너의 레퍼런스 픽처에서의 포지션을 결정할 수도 있다. 이 예에서, 레퍼런스 블록의 상단-좌측 코너의 표시된 포지션이 정수 픽셀의 레퍼런스 픽처에서의 포지션을 나타내지 않으면, 비디오 디코더 (30) 는 레퍼런스 블록의 샘플들을 보간하여 레퍼런스 블록을 결정할 수도 있다.

[0482] 또한, 비-스퀘어 예측 블록을 생성하기 위해 IC 를 사용하는 것의 부분으로서, 비디오 디코더 (30) 는 제 1 서브-샘플링 비로 레퍼런스 샘플들의 제 1 서브-샘플링된 세트를 생성하기 위해 레퍼런스 샘플들의 제 1 세트를 서브-샘플링할 수도 있다 (556). 이 예에서, 레퍼런스 샘플들의 제 1 세트에서의 레퍼런스 샘플들의 총 수는 2^m 과 동일하지 않고, 레퍼런스 샘플들의 제 1 서브-샘플링된 세트에서의 레퍼런스 샘플들의 총 수는 2^m 과 동일하다. 이 예에서, 레퍼런스 샘플들의 제 1 세트는 비-스퀘어 예측 블록의 좌측 사이드 및 상단 사이드를 따라 비-스퀘어 예측 블록 밖의 샘플들을 포함할 수도 있고, m 은 정수이다.

[0483] 부가적으로, 비-스퀘어 예측 블록을 생성하기 위해 IC 를 사용하는 것의 부분으로서, 비디오 디코더 (30) 는 제 2 서브-샘플링 비로 레퍼런스 샘플들의 제 2 서브-샘플링된 세트를 생성하기 위해 레퍼런스 샘플들의 제 2 세트를 서브-샘플링할 수도 있다 (558). 제 1 서브-샘플링 비는 제 2 서브-샘플링 비와 동일하거나 또는 상이할 수도 있다. 이 예에서, 레퍼런스 샘플들의 제 2 세트에서의 레퍼런스 샘플들의 총 수는 2^m 과 동일하지 않고, 레퍼런스 샘플들의 제 2 서브-샘플링된 세트에서의 레퍼런스 샘플들의 총 수는 2^m 과 동일하다. 또한, 이 예에서, 레퍼런스 샘플들의 제 2 세트는 레퍼런스 블록의 좌측 사이드 및 상단 사이드를 따라 레퍼런스 블록 밖의 샘플들을 포함할 수도 있다.

[0484] 액션들 (556) 및 (558) 에서, 비디오 디코더 (30) 는 다양한 방식들로 서브-샘플링을 수행할 수도 있다. 예를 들어, 비디오 디코더 (30) 는 데시메이션을 사용하여 서브-샘플링을 수행할 수도 있다. 비디오 디코더 (30) 가 데시메이션을 사용하여 서브-샘플링을 수행하는 예들에서, 비디오 디코더 (30) 는 규칙적인 인터벌들로 (예를 들어, 2 샘플들마다 한번씩) 샘플들을 제거하여 나머지 샘플들의 값을 변화시키지 않고 샘플들의 수를 감소시킬 수도 있다. 따라서, 이 예에서, 비디오 디코더 (30) 는, 레퍼런스 샘플들의 제 1 서브-샘플링된 세트를 생성하기 위해 레퍼런스 샘플들의 제 1 세트를 데시메이팅하는 것; 및 레퍼런스 샘플들의 제 2 서브-샘플링된 세트를 생성하기 위해 레퍼런스 샘플들의 제 2 세트를 데시메이팅하는 것 중 적어도 하나를 수행할 수도 있다.

[0485] 다른 예에서, 비디오 디코더 (30) 는 보간을 사용하여 서브-샘플링을 수행할 수도 있다. 비디오 디코더 (30) 가 보간을 사용하여 서브-샘플링을 수행하는 예들에서, 인접한 샘플들의 개별의 쌍들에 대해, 비디오 디코더 (30) 는 개별의 쌍의 샘플들 간의 값을 보간할 수도 있고 샘플들의 서브-샘플링된 세트에서 보간된 값을 포함할 수도 있다. 따라서, 이 예에서, 비디오 디코더 (30) 는, 레퍼런스 샘플들의 제 1 서브-샘플링된 세트를 생성하기 위해 레퍼런스 샘플들의 제 1 세트의 보간된 샘플링을 수행하는 것; 및 레퍼런스 샘플들의 제 2 서브-샘플링된 세트를 생성하기 위해 레퍼런스 샘플들의 제 2 세트의 보간된 샘플링을 수행하는 것 중 적어도 하나를 수행할 수도 있다.

[0486] 다른 예에서, 비디오 디코더 (30) 는 비트스트림에서 신택스 엘리먼트에 의해 표시된 서브-샘플링 방법을 사용하여 서브-샘플링을 수행할 수도 있다. 따라서, 이 예에서, 비디오 디코더 (30) 는, 비트스트림으로부터, 서브-샘플링 방법을 표시하는 신택스 엘리먼트를 획득할 수도 있다. 이 예에서, 비디오 디코더 (30) 는, 레퍼런스 샘플들의 제 1 서브-샘플링된 세트를 생성하기 위해 레퍼런스 샘플들의 제 1 세트를 서브-샘플링하기 위한 표시된 서브-샘플링 방법을 사용하는 것; 및 레퍼런스 샘플들의 제 2 서브-샘플링된 세트를 생성하기 위해 레퍼런스 샘플들의 제 2 세트를 서브-샘플링하기 위한 표시된 서브-샘플링 방법을 사용하는 것 중 적어도 하나를 수행할 수도 있다.

[0487] 다른 예에서, 비디오 디코더 (30) 는, 현재 PU 의 사이즈에 기초하여, 서브-샘플링 방법을 결정할 수도 있다. 이 예에서, 비디오 디코더 (30) 는, 레퍼런스 샘플들의 제 1 서브-샘플링된 세트를 생성하기 위해 레퍼런스 샘플들의 제 1 세트를 서브-샘플링하기 위한 결정된 서브-샘플링 방법을 사용하는 것; 및 레퍼런스 샘플들의 제 2 서브-샘플링된 세트를 생성하기 위해 레퍼런스 샘플들의 제 2 세트를 서브-샘플링하기 위한 결정된 서브-샘플링 방법을 사용하는 것 중 적어도 하나를 수행할 수도 있다.

[0488] 또한, 도 33 의 액션 (560) 에서, 비디오 디코더 (30) 는 레퍼런스 샘플들의 제 1 서브-샘플링된 세트에서의 레퍼런스 샘플들의 총 수 및 레퍼런스 샘플들의 제 2 서브-샘플링된 세트에서의 레퍼런스 샘플들의 총 수, 레퍼런

스 샘플들의 제 1 서브-샘플링된 세트, 및 레퍼런스 샘플들의 제 2 서브-샘플링된 세트에 기초하여 제 1 IC 파라미터를 결정할 수도 있다. 예를 들어, 비디오 디코더 (30) 는, 제 1 IC 파라미터가 다음에 기초하도록 제 1 IC 파라미터를 결정할 수도 있다:

[0489] $$\frac{\sum_{i=0}^{2N-1} Rec_{neigh}(i) - a \cdot \sum_{i=0}^{2N-1} Rec_{refneigh}(i)}{2N}$$

[0490] 상기의 등식에서, 2N 은 레퍼런스 샘플들의 제 1 서브-샘플링된 세트에서의 레퍼런스 샘플들의 총 수 및 레퍼런스 샘플들의 제 2 서브-샘플링된 세트에서의 레퍼런스 샘플들의 총 수를 가리키고, $Rec_{neigh}(i)$ 은 레퍼런스 샘플들의 제 1 서브-샘플링된 세트에서의 i-번째 레퍼런스 샘플을 가리키며, $Rec_{refneigh}(i)$ 는 레퍼런스 샘플들의 제 2 서브-샘플링된 세트에서의 i-번째 레퍼런스 샘플을 가리킨다.

[0491] 일부 예들에서, 비디오 디코더 (30) 는 제 2 IC 파라미터가 다음에 기초하도록 제 2 IC 파라미터를 결정할 수도 있다:

[0492] $$\frac{2N \cdot \sum_{i=0}^{2N-1} Rec_{neigh}(i) \cdot Rec_{refneigh}(i) - \sum_{i=0}^{2N-1} Rec_{neigh}(i) \cdot \sum_{i=0}^{2N-1} Rec_{refneigh}(i)}{2N \cdot \sum_{i=0}^{2N-1} Rec_{refneigh}(i) \cdot Rec_{neigh}(i) - \left(\sum_{i=0}^{2N-1} Rec_{refneigh}(i)\right)^2}$$

[0493] 부가적으로, 도 33 의 액션 (562) 에서, 비디오 디코더 (30) 는 제 1 IC 파라미터에 기초하여 비-스퀘어 예측 블록의 샘플을 결정할 수도 있다. 예를 들어, 샘플은 현재 픽처의 상단-좌측 코너에 대한 좌표들 (i,j) 에 있을 수도 있고, 비디오 디코더 (30) 는 샘플이 다음에 기초하도록 샘플을 결정할 수도 있다:

[0494] $$a * r(i + dv_x, j + dv_y + b)$$

[0495] 상기 등식에서, b 는 제 1 IC 파라미터이고, a 는 제 2 IC 파라미터이고, r 은 레퍼런스 픽처이고, dv_x 는 현재 PU 의 벡터의 수평 컴포넌트이며, dv_y 는 현재 PU 의 벡터의 수직 컴포넌트이다.

[0496] 비디오 디코더 (30) 는 비-스퀘어 예측 블록에 기초하여, 현재 CU 의 코딩 블록을 복원할 수도 있다 (564). 예를 들어, 비디오 디코더 (30) 는 비-스퀘어 예측 블록의 샘플들을 CU 의 TU 에 대한 잔차 블록의 대응하는 샘플들에 추가함으로써 코딩 블록의 샘플들을 복원할 수도 있다.

[0497] 도 34 는 플렉서블 잔차 트리를 사용하는 본 개시물의 기법에 따라, 비디오 데이터를 인코딩하기 위한 비디오 인코더 (20) 의 예시의 동작을 예시하는 플로우차트이다. 도 34 의 예에서, 비디오 인코더 (20) 는 비디오 데이터를 수신할 수도 있다 (600). 또한, 비디오 인코더 (20) 는 트리 구조에 기초하여 비디오 데이터의 CU 를 CU 의 TU들로 파티셔닝할 수도 있다 (602). 일부 예들에서, 비디오 인코더 (20) 는 트리 구조의 각각의 개별의 노드에 대해, 개별의 노드에 대한 스플리팅 표시자의 값을 결정할 수도 있다. 개별의 노드에 대한 스플리팅 표시자는, 개별의 노드가 얼마나 많은 차일드 노드들을 갖는지를 나타낼 수도 있다. 일부 경우들에서, 비디오 인코더 (20) 는, 비트스트림에서, 개별의 노드의 스플리팅 표시자의 값을 명시적으로 나타내는 신택스 엘리먼트를 시그널링할 수도 있다. 다른 경우들에서, 비디오 디코더 (30) 는 (예를 들어, 트리 구조에서 노드의 심도, 페어런트 노드들의 스플리팅 노드들의 값들, 개별의 노드에 대응하는 예측 블록들의 사이즈들 및/또는 형상들, 등에 기초하여) 개별의 노드에 대한 스플리팅 표시자의 값을 추론할 수도 있다.

[0498] 트리 구조에 기초하여 CU 를 CU 의 TU들로 파티셔닝하는 것의 부분으로서, 비디오 인코더 (20) 는 트리 구조에서 노드가 트리 구조에서 정확히 2 개의 차일드 노드들을 갖는다고 결정할 수도 있다 (604). 이 예에서, 트리 구조의 루트 노드는 CU 의 코딩 블록에 대응한다. 트리 구조의 각각의 개별의 비-루트 노드는 개별의 비-루트 노드의 페어런트 노드에 대응하는 블록의 파티션인 개별의 블록에 대응한다. 트리 구조의 리프 노드들은 CU들의 TU들에 대응한다.

[0499] 예를 들어, 비디오 인코더 (20) 는, CU 의 PU들의 총 수에 기초하여, 트리 구조가 바이너리 트리인지 또는 쿼터 트리인지 여부를 결정할 수도 있다. 이 예에서, 2 개의 PU들을 갖는 CU 에 기초하여, 노드는 트리 구조에서 정확히 2 개의 차일드 노드들을 갖는다. 다시 말해, 비디오 인코더 (20) 는, 정확히 2 개의 PU들을 갖는 CU 에 기초하여, 노드가 트리 구조에서 정확히 2 개의 차일드 노드들을 갖는다고 결정할 수도 있다.

[0500] 일부 예들에서, 비디오 인코더 (20) 는, 정확히 2 개의 PU들을 갖는 CU 에 기초하여, 노드가 트리 구조에서 정

확히 2 개의 차일드 노드들을 갖는다고 결정할 수도 있다.

[0501] 또한, 도 34 의 예에서, CU 의 TU들 중 적어도 하나에 대해, 비디오 인코더 (20) 는 TU 에 대한 변환 계수들의 블록을 생성하기 위해 TU 에 대한 잔차 블록에 변환을 적용할 수도 있다 (606). 예를 들어, 비디오 인코더 (20) 는 변환 계수들의 블록을 생성하기 위해 이산 코사인 변환 (DCT), 이산 사인 변환 (DST), 또는 다른 유형의 변환을 TU 에 대한 잔차 블록에 적용할 수도 있다. 부가적으로, 비디오 인코더 (20) 는 TU 에 대한 변환 계수들을 나타내는 신택스 엘리먼트들을 엔트로피 인코딩할 수도 있다 (608). 예를 들어, 비디오 인코더 (20) 는 변환 계수가 넌-제로인지 여부, 변환 계수가 1 보다 큰지 여부, 변환 계수가 2 보다 큰지 여부, 변환 계수의 부호, 및 변환 계수에 대한 나머지를 나타내는 신택스 엘리먼트들을 생성할 수도 있다. 이 예에서, 비디오 인코더 (20) 는 이들 신택스 엘리먼트들 중 하나 이상에 CABAC 코딩을 적용할 수도 있다.

[0502] 도 35 는 플렉서블 잔차 트리를 사용하는 본 개시물의 기법에 따라, 비디오 데이터를 디코딩하기 위한 비디오 디코더 (30) 의 예시의 동작을 예시하는 플로우차트이다. 도 35 의 예에서, 비디오 디코더 (30) 는 비디오 데이터의 인코딩된 표현을 포함하는 비트스트림을 수신한다 (650). 부가적으로, 비디오 디코더 (30) 는, 비디오 데이터의 CU 가 트리 구조에 기초하여 CU 의 TU들로 파티셔닝된다고 결정할 수도 있다 (652). 일부 예들에서, 비디오 디코더 (30) 는 트리 구조의 각각의 개별의 노드에 대해, 개별의 노드에 대한 스플리팅 표시자의 값을 결정할 수도 있다. 개별의 노드에 대한 스플리팅 표시자는, 개별의 노드가 얼마나 많은 차일드 노드들을 갖는지를 나타낼 수도 있다. 일부 경우들에서, 비디오 디코더 (30) 는, 비트스트림으로부터 개별의 노드의 스플리팅 표시자의 값을 명시적으로 나타내는 신택스 엘리먼트를 획득할 수도 있다. 다른 경우들에서, 비디오 디코더 (30) 는 (예를 들어, 트리 구조에서 노드의 심도, 페어런트 노드들의 스플리팅 노드들의 값들, 개별의 노드에 대응하는 예측 블록들의 사이즈들 및/또는 형상들, 등에 기초하여) 개별의 노드에 대한 스플리팅 표시자의 값을 추론할 수도 있다.

[0503] CU 가 트리 구조에 기초하여 CU 의 TU들로 파티셔닝된다고 결정하는 것의 부분으로서, 비디오 디코더 (30) 는 트리 구조에서 노드가 트리 구조에서 정확히 2 개의 차일드 노드들을 갖는다고 결정할 수도 있다 (654). 이 예에서, 트리 구조의 루트 노드는 CU 의 코딩 블록에 대응한다. 트리 구조의 각각의 개별의 비-루트 노드는 개별의 비-루트 노드의 페어런트 노드에 대응하는 블록의 파티션인 개별의 블록에 대응한다. 트리 구조의 리프 노드들은 CU 의 TU들에 대응한다. 본 개시물의 다른 곳에서 설명된 바와 같이, 비디오 디코더 (30) 는 CU 에서의 PU들의 수에 기초하여, 트리 구조에서 노드의 심도에 기초하여, 시그널링된 신택스 엘리먼트에 기초하여, 또는 다른 데이터에 기초하여 트리 구조에서의 노드가 정확히 2 개의 차일드 노드들을 갖는다고 결정할 수도 있다.

[0504] 예를 들어, 비디오 디코더 (30) 는, CU 의 PU들의 총 수에 기초하여, 트리 구조가 바이너리 트리인지 또는 쿼터 트리인지 여부를 결정할 수도 있다. 이 예에서, 2 개의 PU들을 갖는 CU 에 기초하여, 노드는 트리 구조에서 정확히 2 개의 차일드 노드들을 갖는다. 다시 말해, 비디오 디코더 (30) 는, 정확히 2 개의 PU들을 갖는 CU 에 기초하여, 노드가 트리 구조에서 정확히 2 개의 차일드 노드들을 갖는다고 결정할 수도 있다.

[0505] CU 의 TU들 중 적어도 하나에 대해, 비디오 디코더 (30) 는 TU 에 대한 잔차 블록을 생성하기 위해 TU 에 대한 계수 블록에 변환을 적용할 수도 있다 (656). 예를 들어, 비디오 디코더 (30) 는 TU 에 대한 잔차 블록을 생성하기 위해, 역 DCT, 역 DST, 또는 다른 유형의 변환을 TU 에 대한 계수 블록에 적용할 수도 있다. 부가적으로, 비디오 디코더 (30) 는 예측 블록의 샘플들을 CU 의 TU 에 대한 잔차 블록의 대응하는 샘플들에 추가함으로써 코딩 블록의 샘플들을 복원할 수도 있다 (658).

[0506] 본 개시물의 소정 양태들은 예시의 목적을 위해 HEVC 표준의 확장들에 대하여 설명되어 있다. 그러나, 본 개시물에 설명된 기법들은, 아직 개발되지 않은 다른 표준 또는 사설 비디오 코딩 프로세스들을 포함하여, 다른 비디오 코딩 프로세스들에 유용할 수도 있다.

[0507] 본 개시물에 설명된 바와 같이, 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 적용 가능한 바와 같이, 비디오 인코딩 또는 비디오 디코딩을 지칭할 수도 있다.

[0508] 예에 따라, 본원에서 설명된 임의의 기법들의 소정의 액트들 또는 이벤트들은 상이한 시퀀스로 수행될 수도 있거나, 추가, 머지될 수도 있거나, 또는 함께 제거될 수도 있다 (예를 들어, 설명된 모든 액트들 또는 이벤트들이 기법들의 실시에 반드시 필요한 것은 아님) 는 것으로 인식되어야 한다. 또한, 소정 예들에서, 액트들 또는 이벤트들은, 순차적이기 보다는 예를 들어 멀티-스레디드 프로세싱, 인터럽트 프로세싱, 또는 다중 프로세

서들을 통해 동시에 수행될 수도 있다.

[0509] 하나 이상의 예들에서, 설명된 기능들은 하드웨어, 소프트웨어, 펌웨어, 또는 이들의 임의의 조합으로 구현될 수도 있다. 소프트웨어로 구현되는 경우, 이 기능들은 하나 이상의 명령들 또는 코드로서 컴퓨터 판독 가능 매체 상에 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 들어 통신 프로토콜에 따라, 한 곳에서 다른 곳으로 컴퓨터 프로그램의 전송을 용이하게 하는 임의의 매체를 포함하는 통신 매체를 포함할 수도 있다. 이 방식으로, 컴퓨터 판독 가능 매체는 일반적으로 (1) 비일시적인 유형의 컴퓨터 판독가능 저장 매체 또는 (2) 신호 또는 반송파와 같은 통신 매체에 대응할 수도 있다. 데이터 저장 매체는 본 개시물에 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 매체일 수도 있다. 컴퓨터 프로그램 제품은 컴퓨터 판독가능 매체를 포함할 수도 있다.

[0510] 비제한적인 예로서, 이러한 컴퓨터 판독가능 저장 매체는 RAM, ROM, EEPROM, CD-ROM 또는 다른 광학 디스크 저장 디바이스, 자기 디스크 저장 디바이스 또는 다른 자기 저장 디바이스, 플래시 메모리, 또는 원하는 프로그램 코드를 명령들 또는 데이터 구조들의 형태로 저장하는데 사용될 수 있으며 컴퓨터에 의해 액세스될 수 있는 임의의 다른 매체를 포함할 수 있다. 또한, 임의의 접속이 컴퓨터 판독가능 매체라고 적절히 칭해진다. 예를 들어, 동축 케이블, 광섬유 케이블, 연선, 디지털 가입자 회선 (DSL), 또는 적외선, 무선, 및 마이크로파와 같은 무선 기술들을 사용하여 웹사이트, 서버, 또는 다른 원격 소스로부터 명령들이 송신되면, 동축 케이블, 광섬유 케이블, 연선, DSL, 또는 적외선, 무선, 및 마이크로파와 같은 무선 기술들은 매체의 정의 내에 포함된다. 그러나, 컴퓨터 판독가능 저장 매체 및 데이터 저장 매체는 접속들, 반송파들, 신호들, 또는 다른 일시적 매체를 포함하지 않고, 대신에 비일시적인, 유형의 저장 매체에 관련된다. 본원에서 사용된 디스크 (disk) 와 디스크 (disc) 는, 컴팩트 디스크 (CD), 레이저 디스크, 광학 디스크, 디지털 다기능 디스크 (DVD), 플로피 디스크, 및 블루레이 디스크를 포함하며, 여기서 디스크 (disk)들은 통상 자기적으로 데이터를 재생하는 반면, 디스크 (disc)들은 레이저들을 이용하여 광학적으로 데이터를 재생한다. 상기의 조합들이 또한, 컴퓨터 판독가능 매체의 범위 내에 포함되어야 한다.

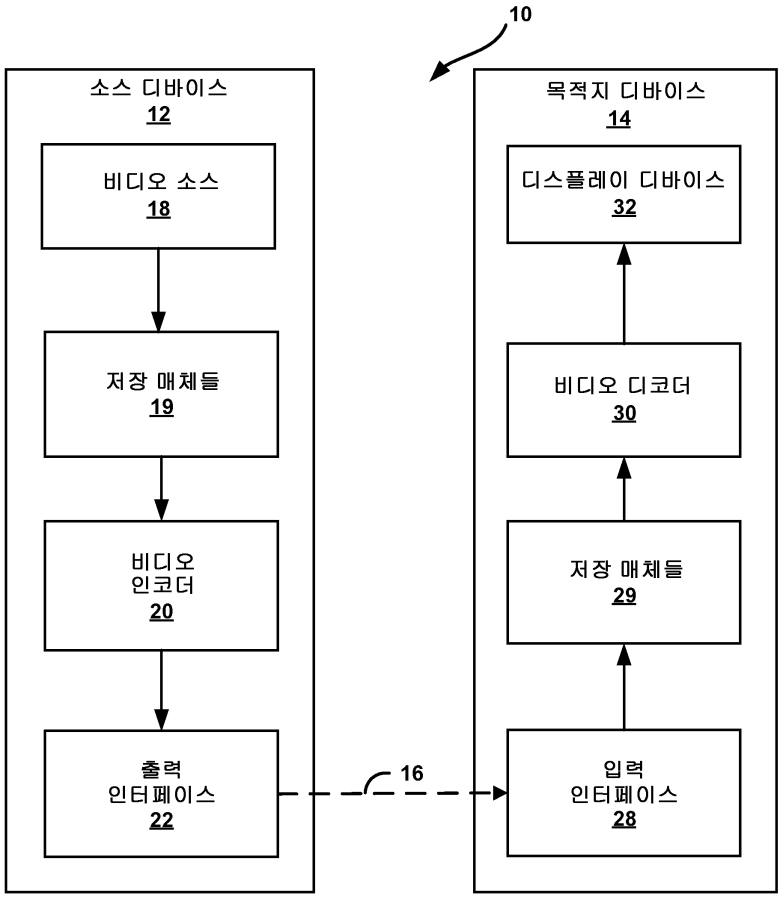
[0511] 명령들은, 하나 이상의 디지털 신호 프로세서 (DSP)들, 범용 마이크로프로세서들, 주문형 집적 회로 (ASIC)들, 필드 프로그램 가능 로직 어레이 (FPGA)들, 또는 다른 등가의 집적 또는 이산 로직 회로와 같은, 하나 이상의 프로세서들에 의해 실행될 수도 있다. 따라서, 본원에서 사용되는 바와 같은 용어 "프로세서" 는 상기의 구조 또는 본원에 설명된 기법들의 구현에 적합한 임의의 다른 구조 중 임의의 것을 지칭할 수도 있다. 또한, 일부 양태들에서, 본원에 설명된 기능성은 인코딩 및 디코딩을 위해 구성된 전용 하드웨어 및/또는 소프트웨어 모듈들 내에 제공될 수도 있고, 또는 결합형 코덱에 통합될 수도 있다. 또한, 본 기법들은 하나 이상의 회로들 또는 로직 엘리먼트들에서 완전히 구현될 수 있다.

[0512] 본 개시물의 기법들은 무선 핸드셋, 집적 회로 (IC) 또는 IC 들의 세트 (예를 들어, 칩 세트) 를 포함하는 광범위한 디바이스들 또는 장치들로 구현될 수도 있다. 개시된 기법들을 수행하도록 구성된 디바이스들의 기능적 양태들을 강조하기 위해 다양한 컴포넌트들, 모듈들, 또는 유닛들이 본 개시물에서 설명되었지만, 반드시 상이한 하드웨어 유닛들에 의해 실현될 필요는 없다. 차라리, 전술된 바와 같이 다양한 유닛들은 적합한 소프트웨어 및/또는 펌웨어와 관련되어, 전술된 하나 이상의 프로세서들을 포함하는 상호 동작적인 하드웨어 유닛들의 집합에 의해 제공되고 또는 코덱 하드웨어 유닛에 결합될 수도 있다.

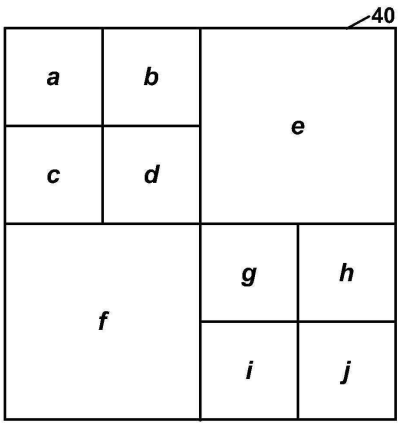
[0513] 다양한 예들이 설명되었다. 이들 및 다른 예들은 다음의 청구항들의 범위 내에 있다.

도면

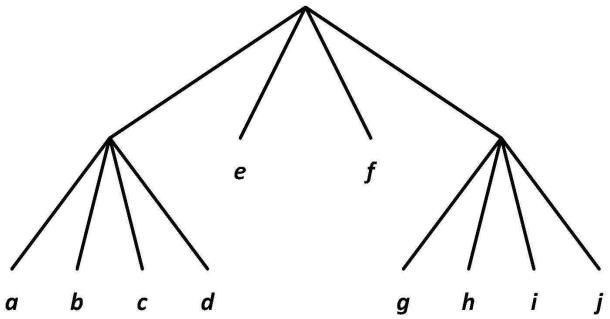
도면1


10
소스 디바이스
12
비디오 소스
18
저장 매체들
19
비디오
인코더
20
출력
인터페이스
22
16
목적지 디바이스
14
디스플레이 디바이스
32
비디오 디코더
30
저장 매체들
29
입력
인터페이스
28

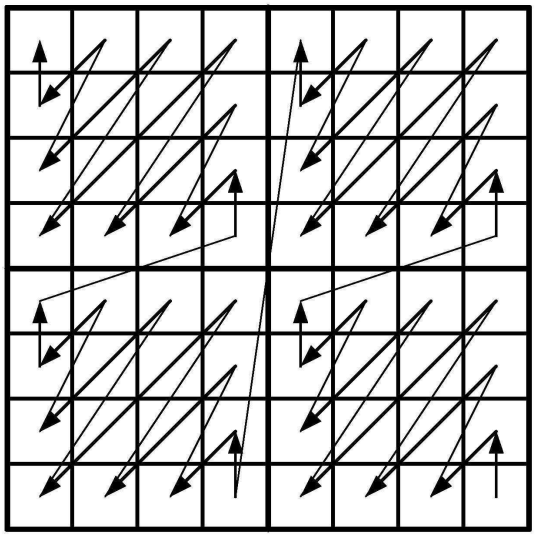
도면2a


40
a
b
c
d
e
f
g
h
i
j

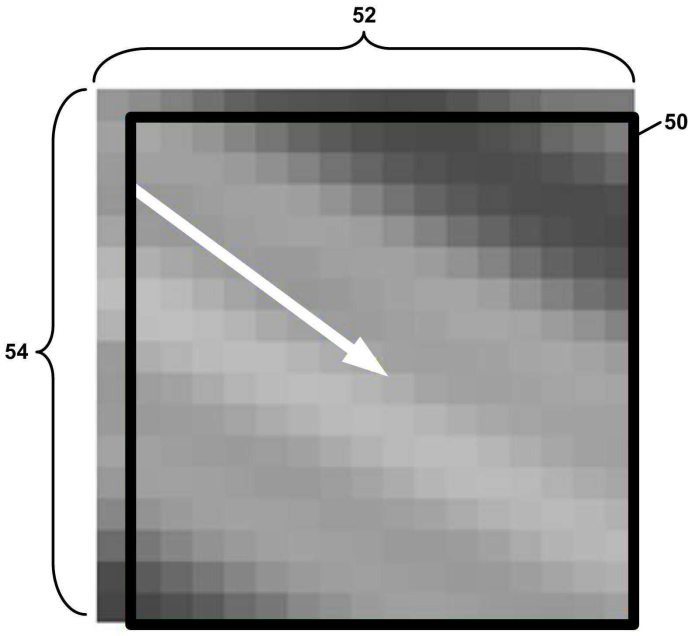
도면2b



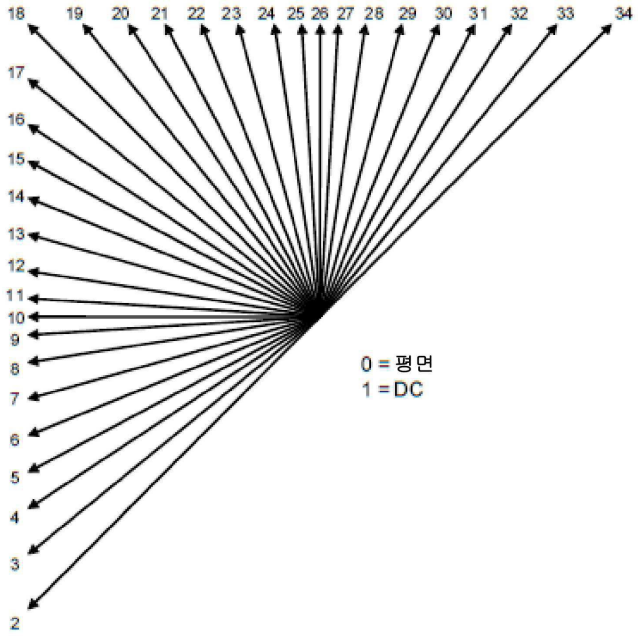
도면3



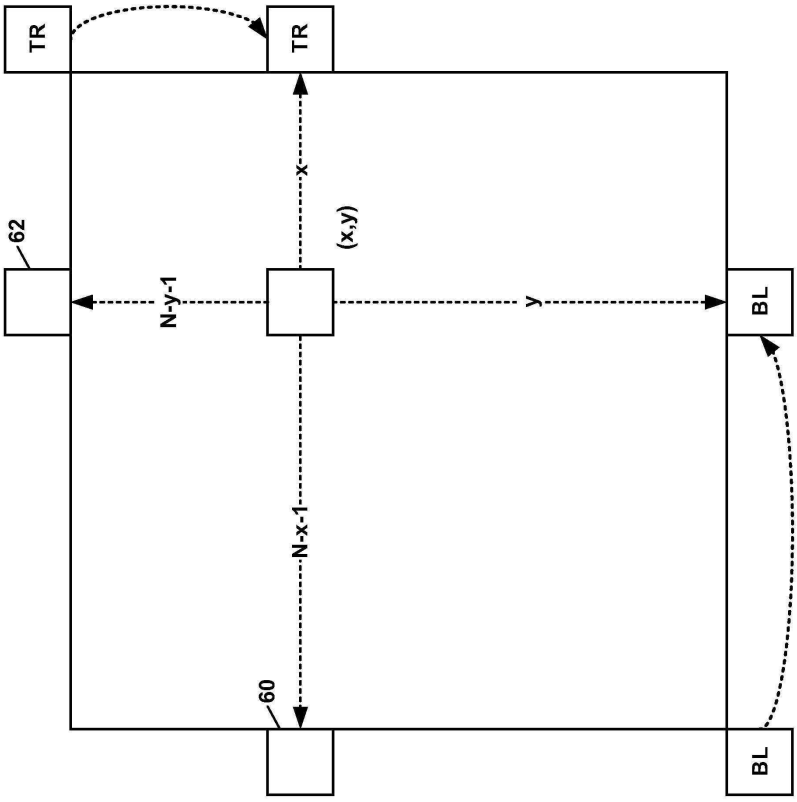
도면4



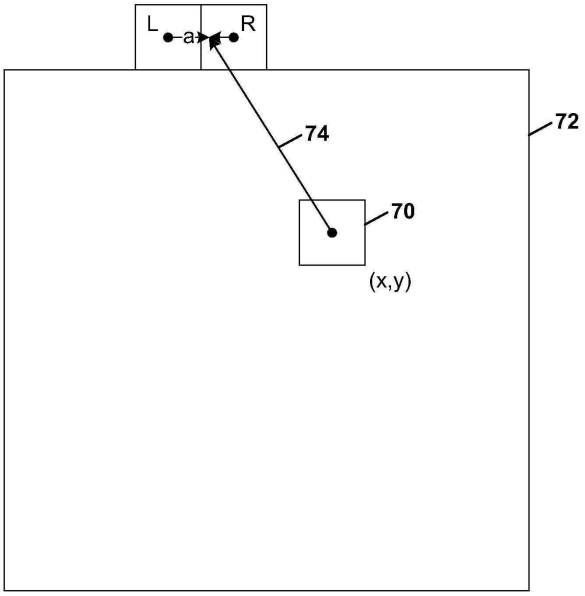
도면5


18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
17
16
15
14
13
12
11
10
9
8
7
6
5
4
3
2
0 = 평면
1 = DC

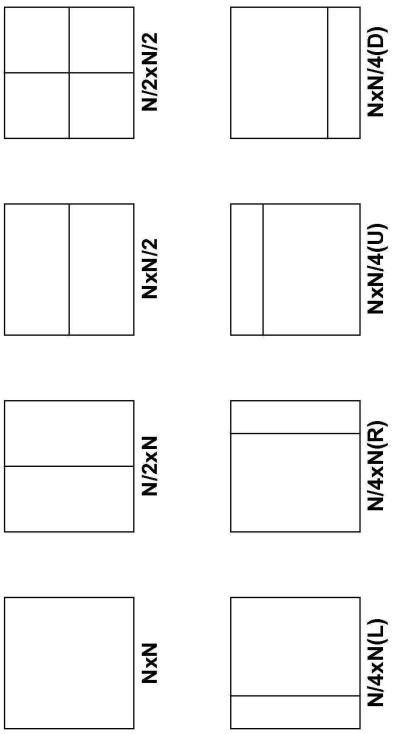
도면6


TR
TR
x
(x,y)
62
N-y-1
y
BL
N-x-1
60
BL

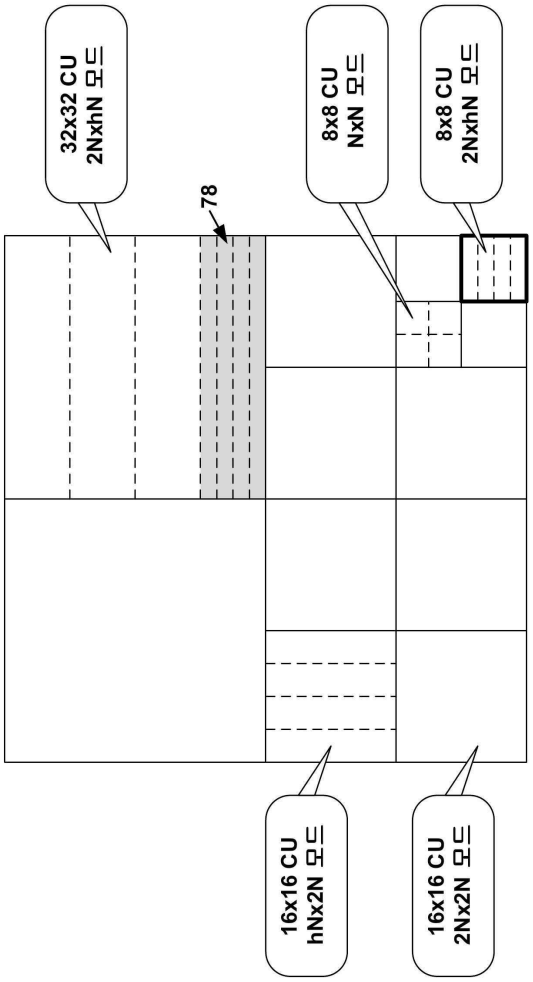
도면7



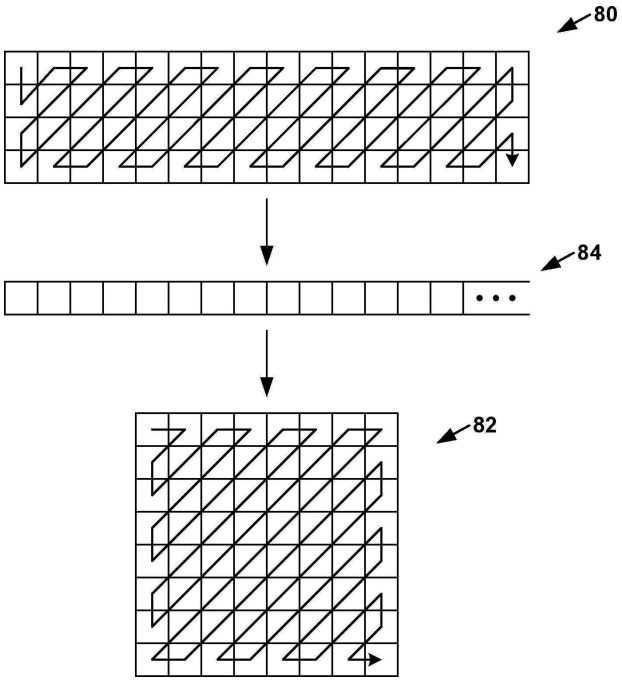
도면8



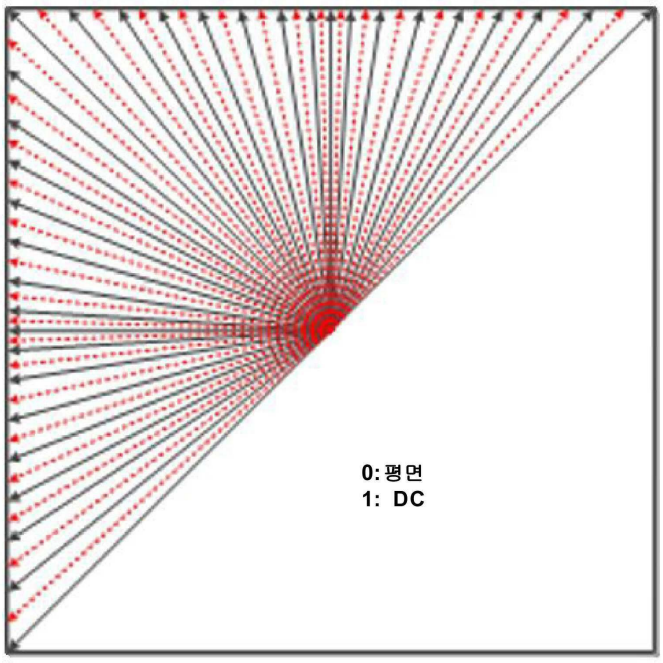
도면9


32x32 CU
2NxhN 모드
8x8 CU
NxN 모드
8x8 CU
2NxhN 모드
78
16x16 CU
hNx2N 모드
16x16 CU
2Nx2N 모드

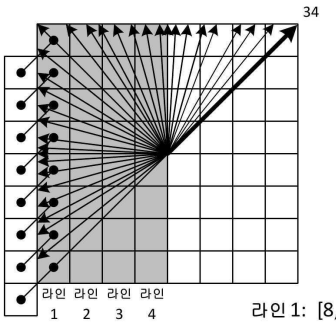
도면10


80
84
82

도면11

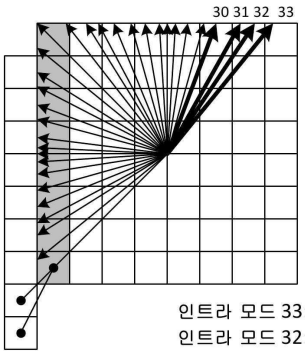


도면12a



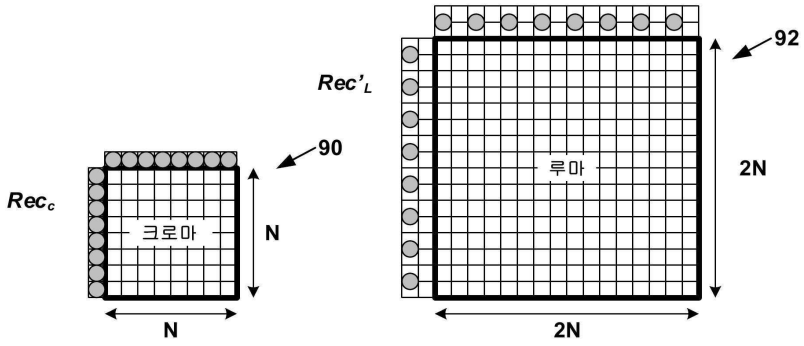
인트라 모드 34
라인 1: [8, 8]
라인 2: [12, 4]
라인 3: [14, 2]
라인 4: [15, 1]

도면12b

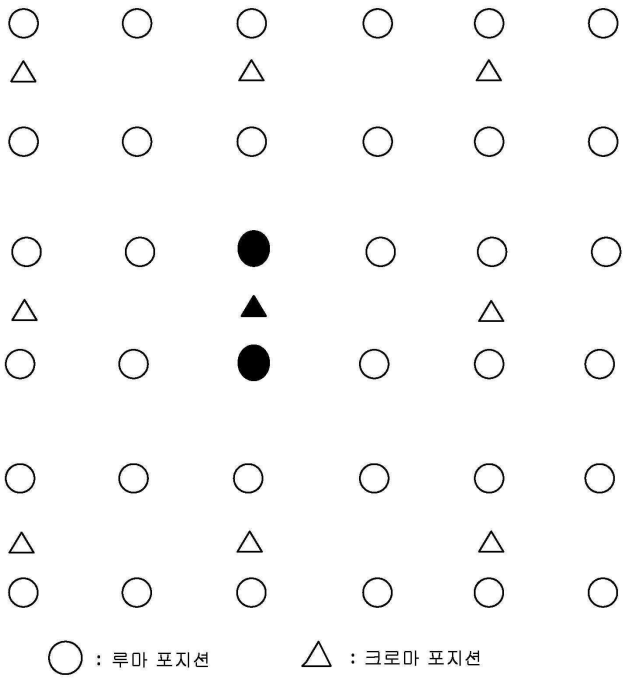


인트라 모드 33: [6, 8, 2]
인트라 모드 32: [2, 12, 2]
인트라 모드 31: [1, 12, 3]
인트라 모드 30: [3, 12, 1]

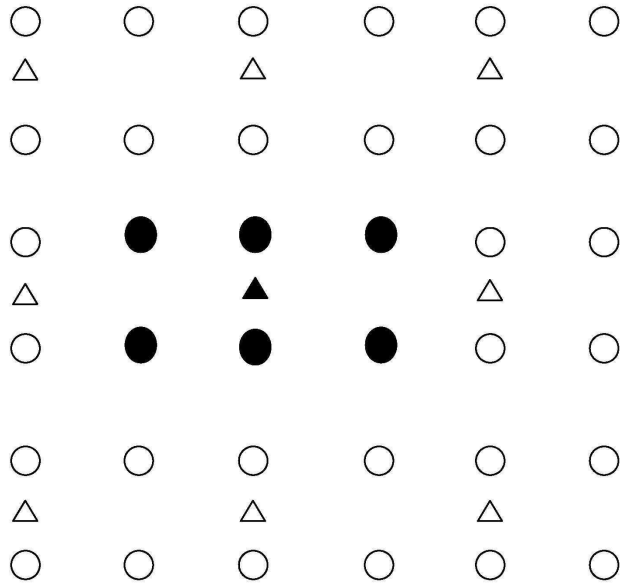
도면13



도면14

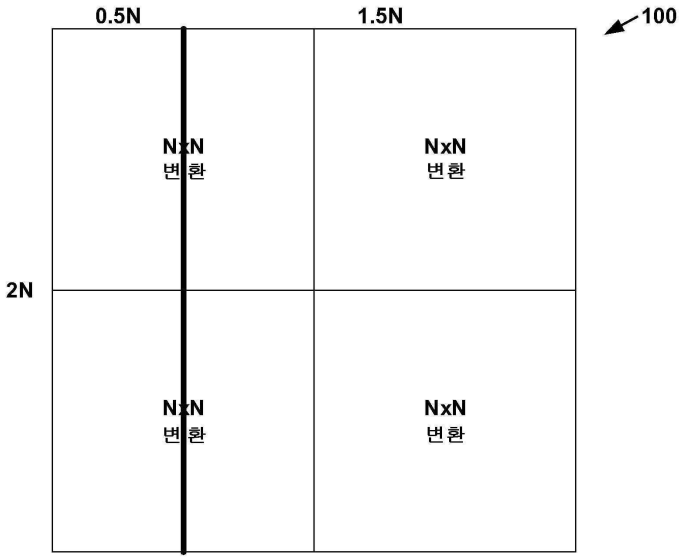


도면15

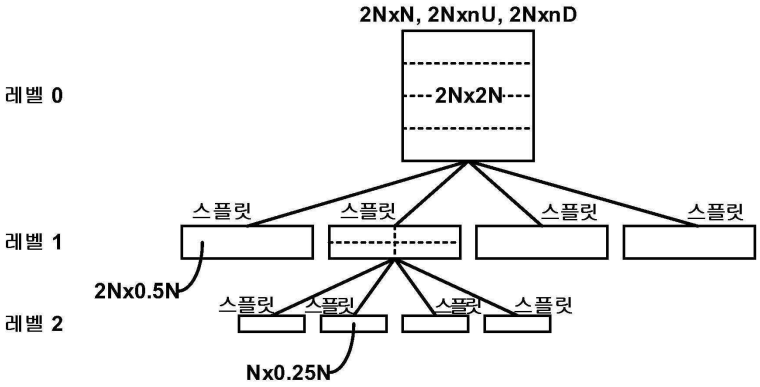


○ : 루마 포지션 △ : 크로마 포지션

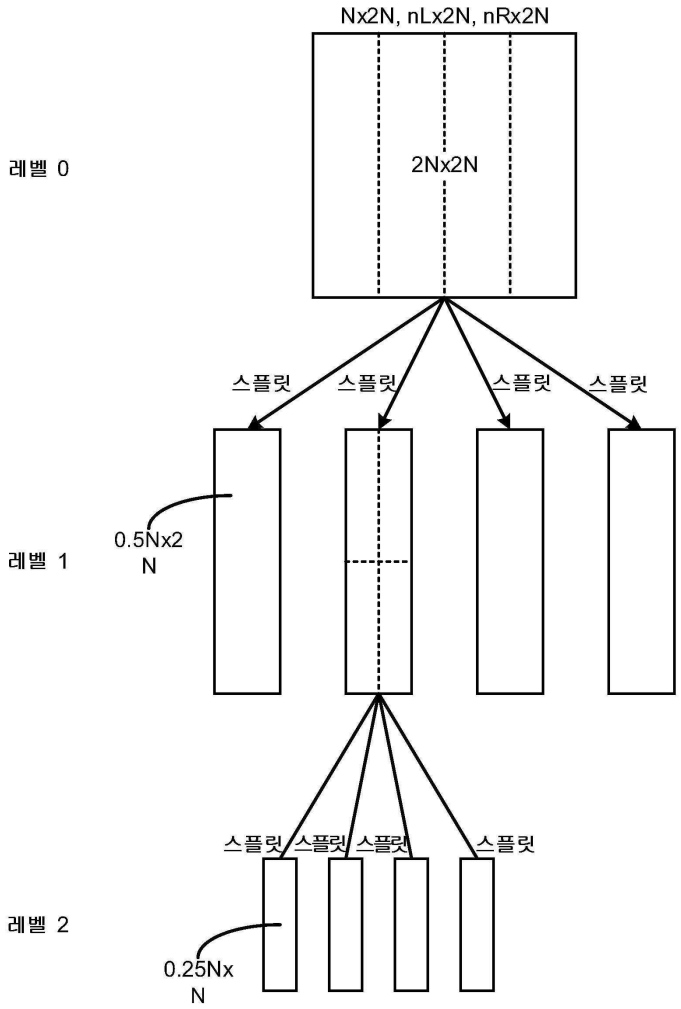
도면16



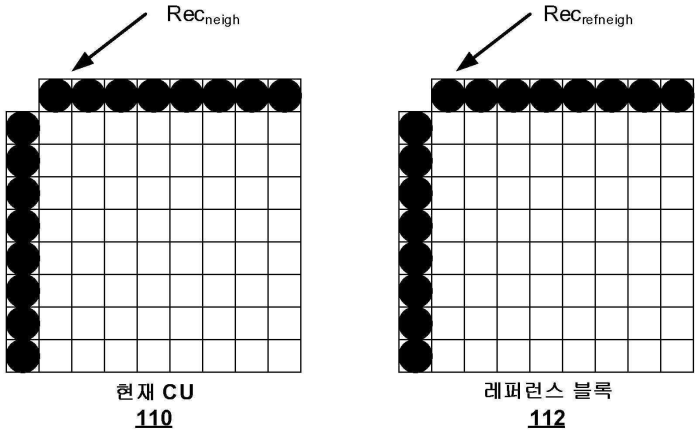
도면17


2NxN, 2NxnU, 2NxnD
2Nx2N
레벨 0
스플릿
스플릿
스플릿
스플릿
레벨 1
2Nx0.5N
스플릿
스플릿
스플릿
스플릿
레벨 2
Nx0.25N

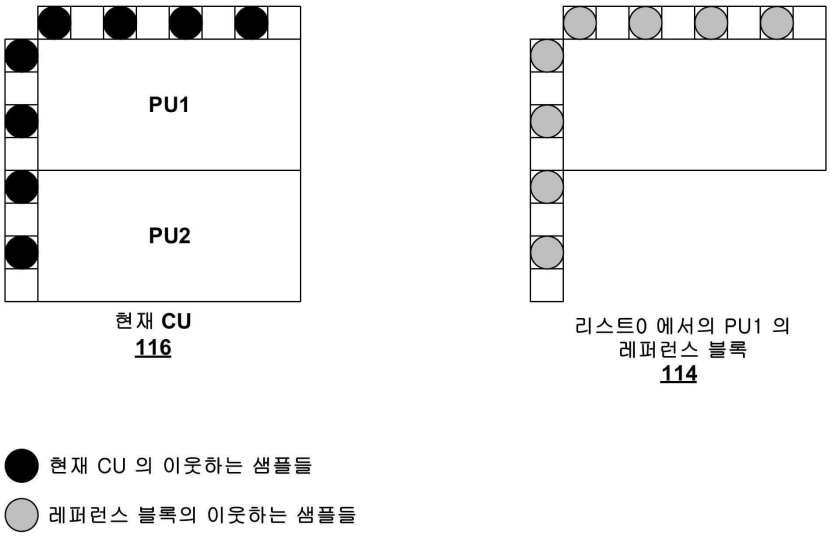
도면18


Nx2N, nLx2N, nRx2N
2Nx2N
레벨 0
스플릿
스플릿
스플릿
스플릿
0.5Nx2
N
레벨 1
스플릿
스플릿
스플릿
스플릿
0.25Nx
N
레벨 2

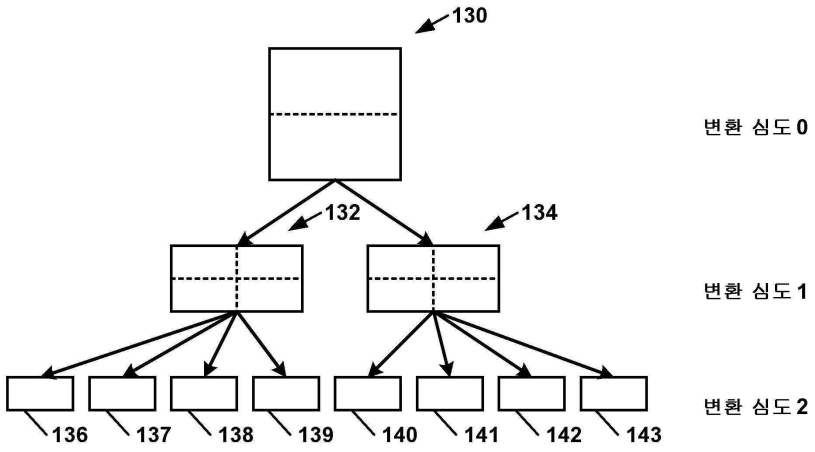
도면19



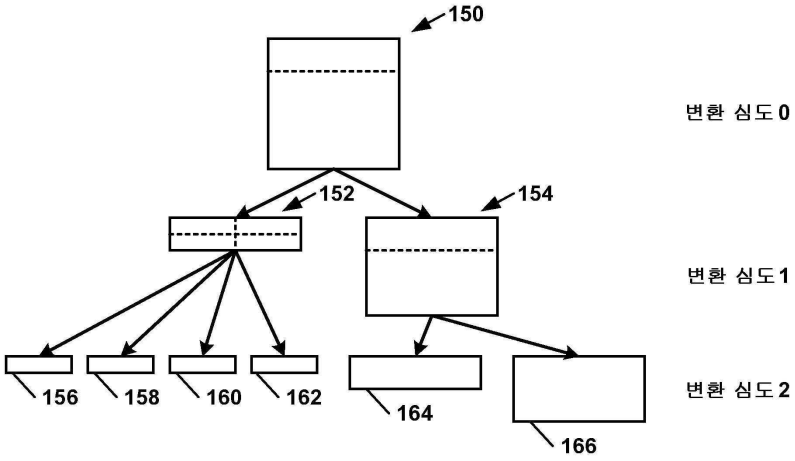
도면20



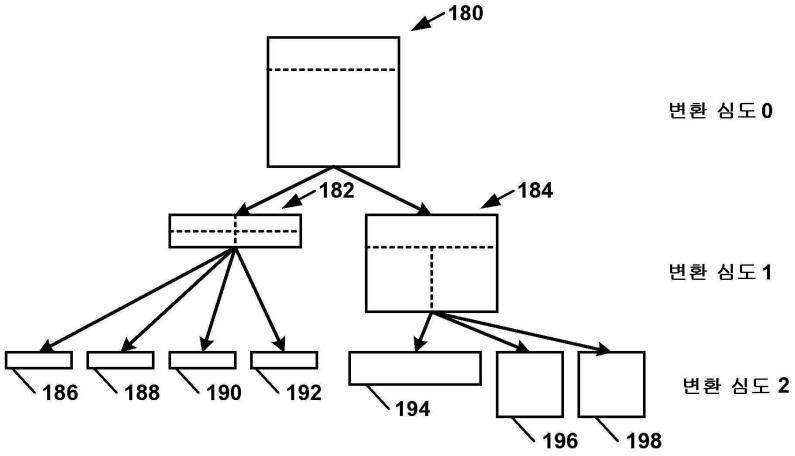
도면21



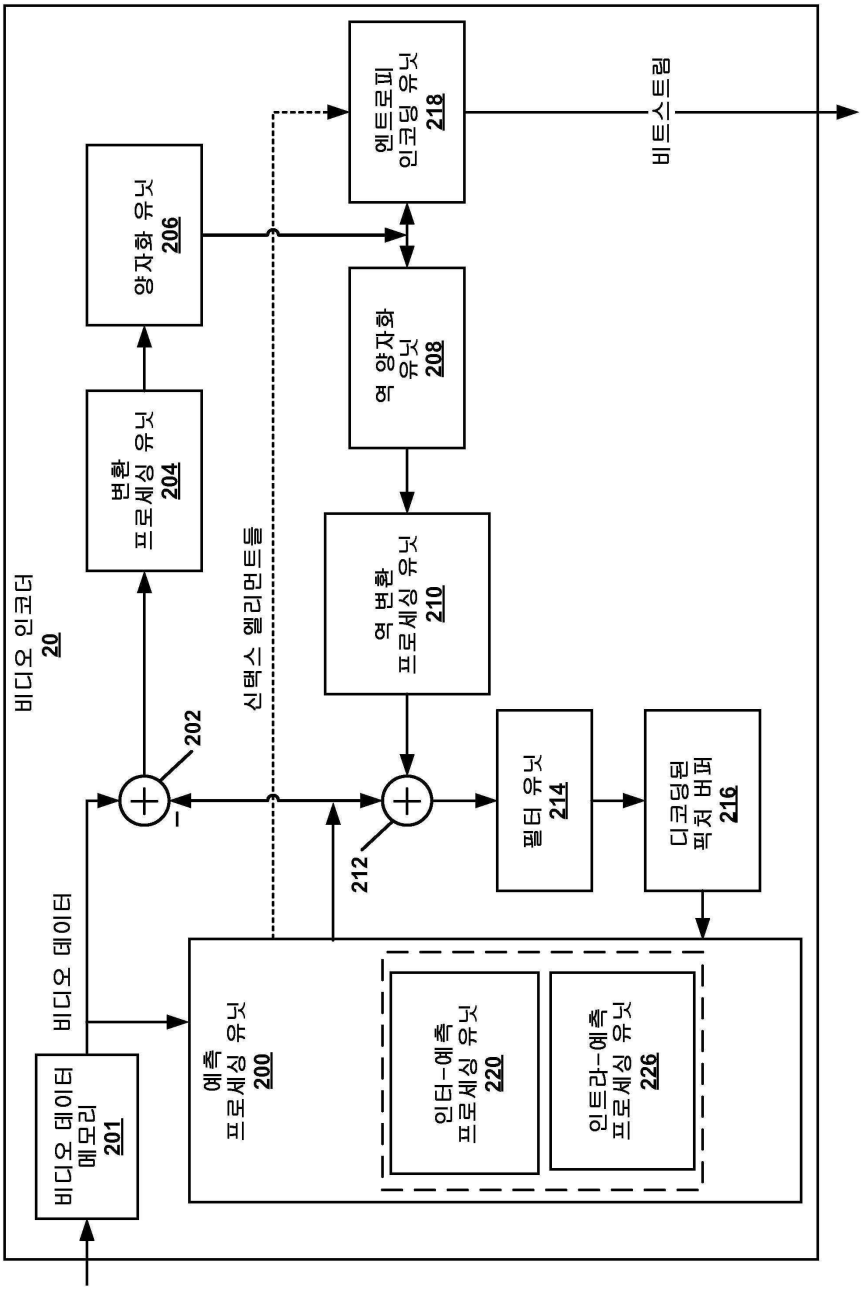
도면22


150
152
154
156
158
160
162
164
166
변환 심도 0
변환 심도 1
변환 심도 2

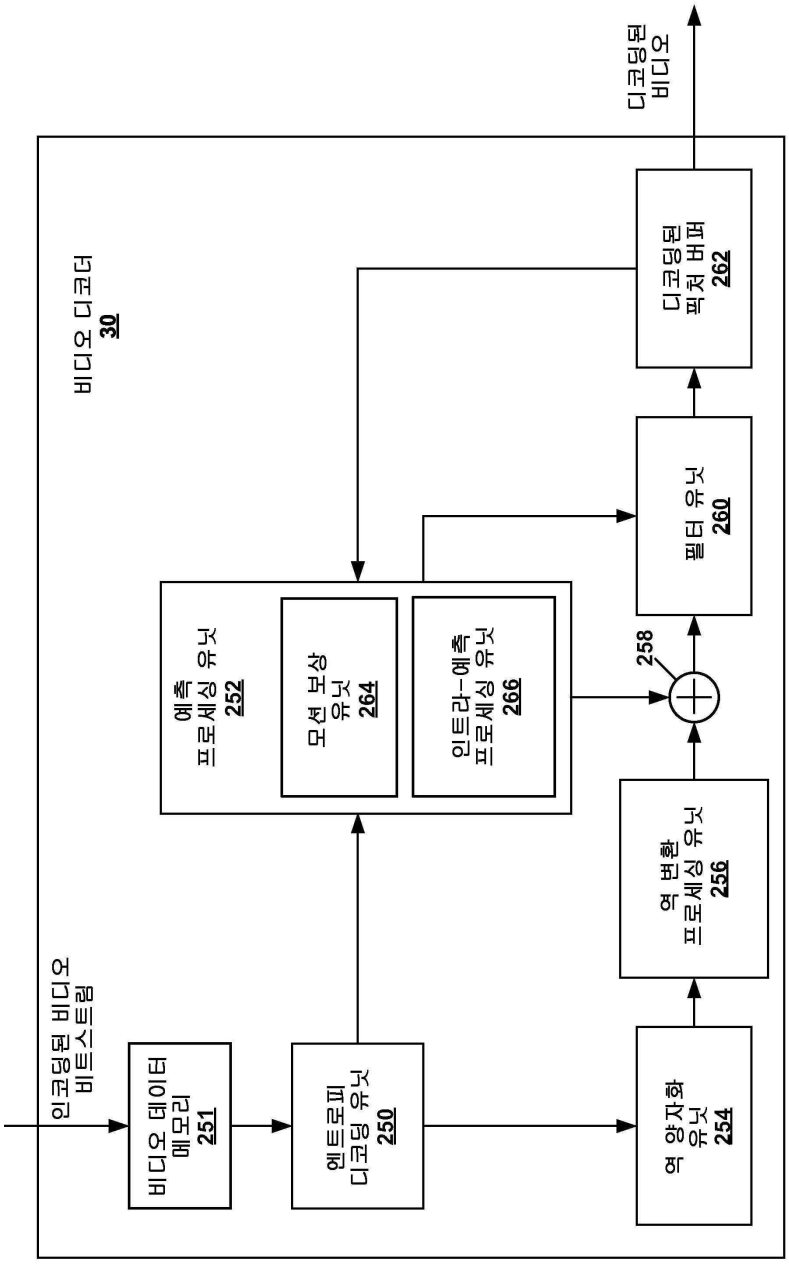
도면23


180
182
184
186
188
190
192
194
196
198
변환 심도 0
변환 심도 1
변환 심도 2

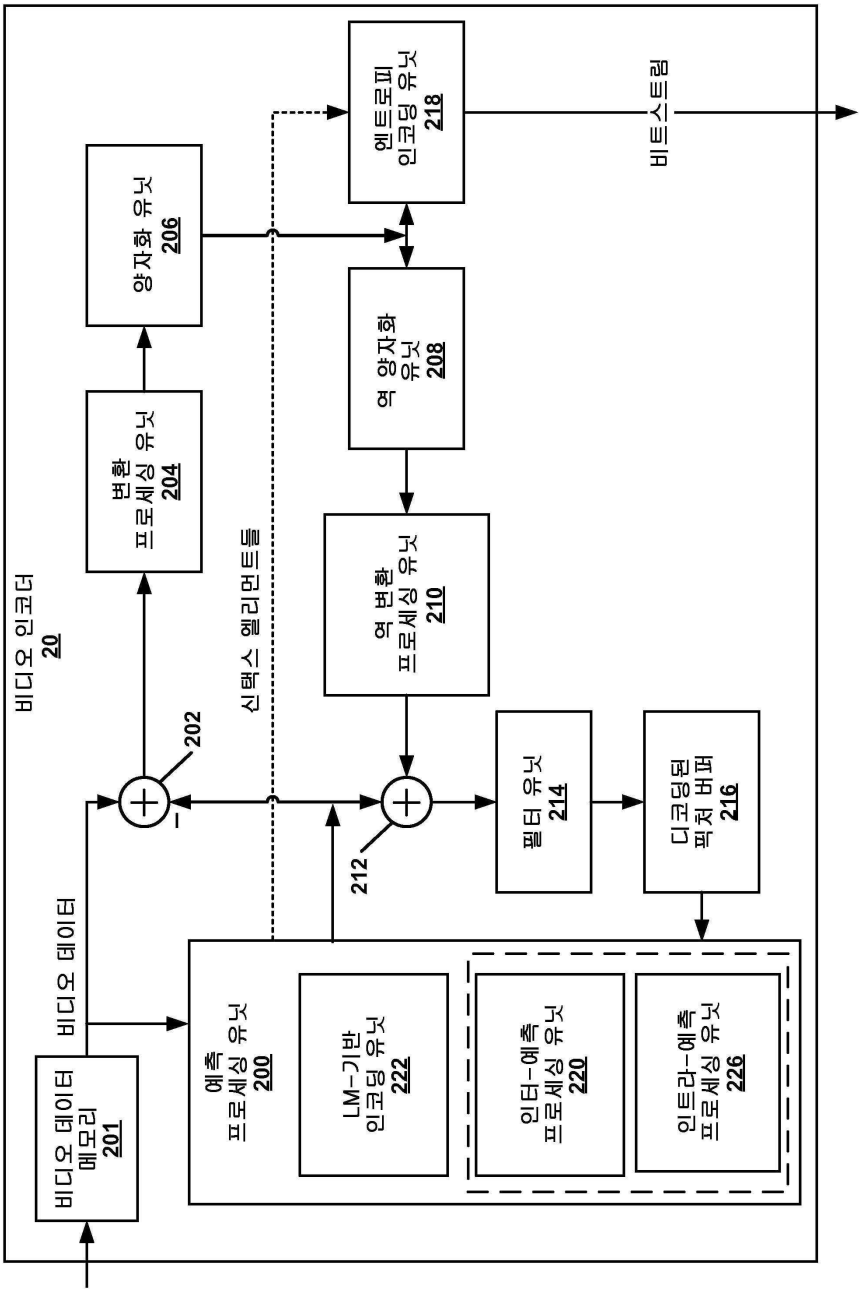
도면24


비디오 인코더
20
비디오 데이터
메모리
201
예측
프로세싱 유닛
200
인터-예측
프로세싱 유닛
220
인트라-예측
프로세싱 유닛
226
202
212
신택스 엘리먼트들
변환
프로세싱 유닛
204
양자화 유닛
206
역 양자화
유닛
208
역 변환
프로세싱 유닛
210
필터 유닛
214
디코딩된
픽처 버퍼
216
엔트로피
인코딩 유닛
218
비트스트림

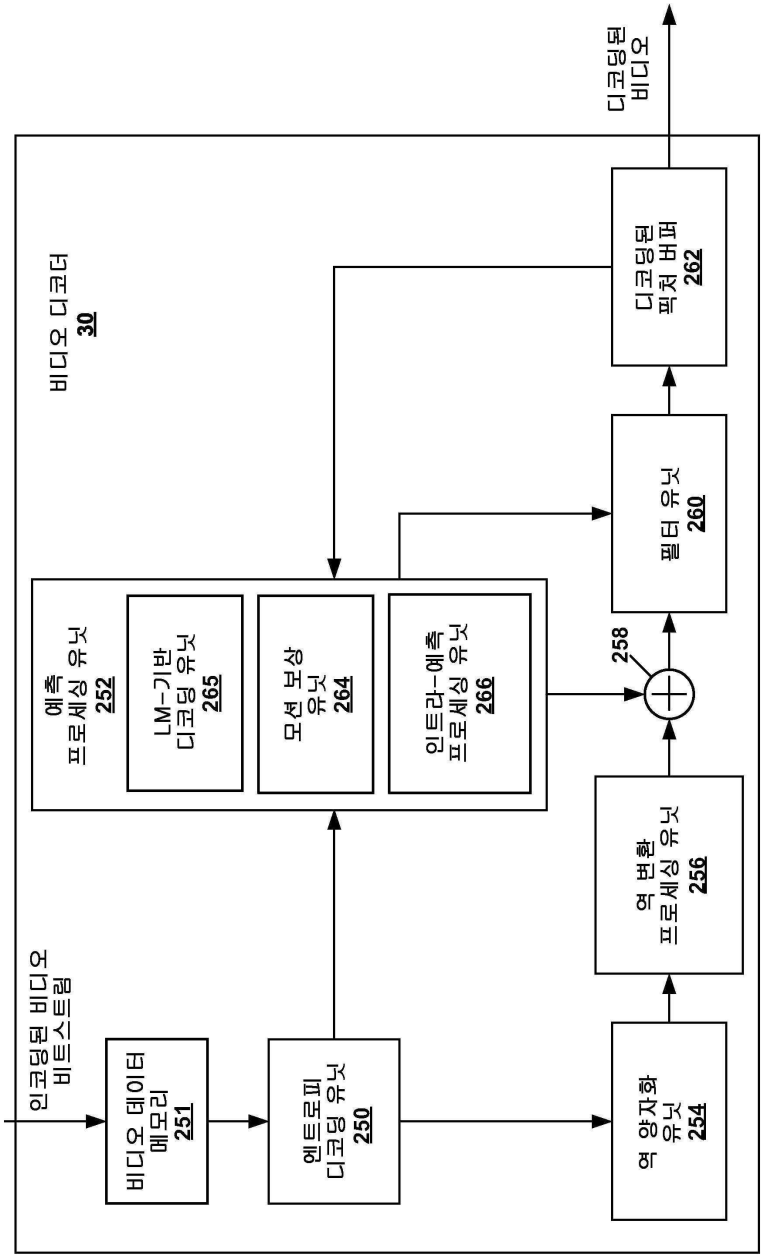
도면25


디코딩된 비디오
비디오 디코더 30
디코딩된 픽처 버퍼 262
필터 유닛 260
258
예측 프로세싱 유닛 252
모션 보상 유닛 264
인트라-예측 프로세싱 유닛 266
역 변환 프로세싱 유닛 256
역 양자화 유닛 254
엔트로피 디코딩 유닛 250
비디오 데이터 메모리 251
인코딩된 비디오 비트스트림

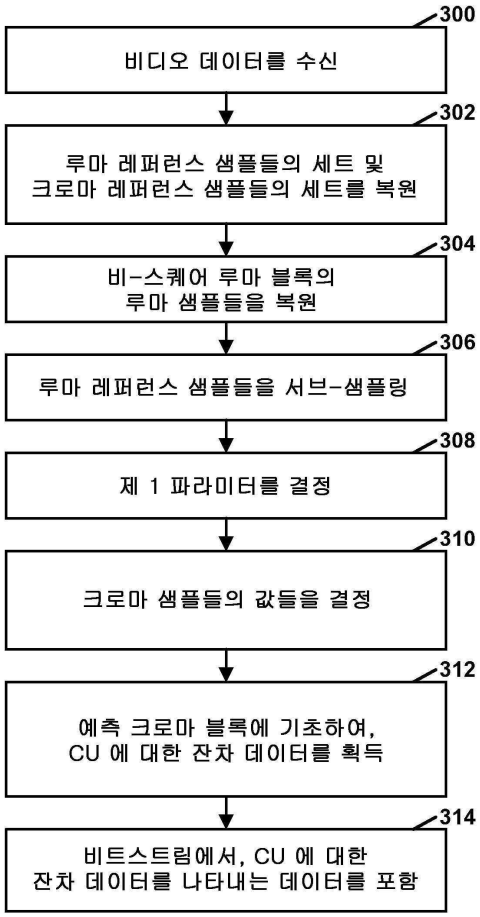
도면26


비디오 인코더
20
비디오 데이터
비디오 데이터
메모리
201
예측
프로세싱 유닛
200
LM-기반
인코딩 유닛
222
인터-예측
프로세싱 유닛
220
인트라-예측
프로세싱 유닛
226
202
-
212
신택스 엘리먼트들
변환
프로세싱 유닛
204
양자화 유닛
206
역 양자화
유닛
208
역 변환
프로세싱 유닛
210
필터 유닛
214
디코딩된
픽처 버퍼
216
엔트로피
인코딩 유닛
218
비트스트림

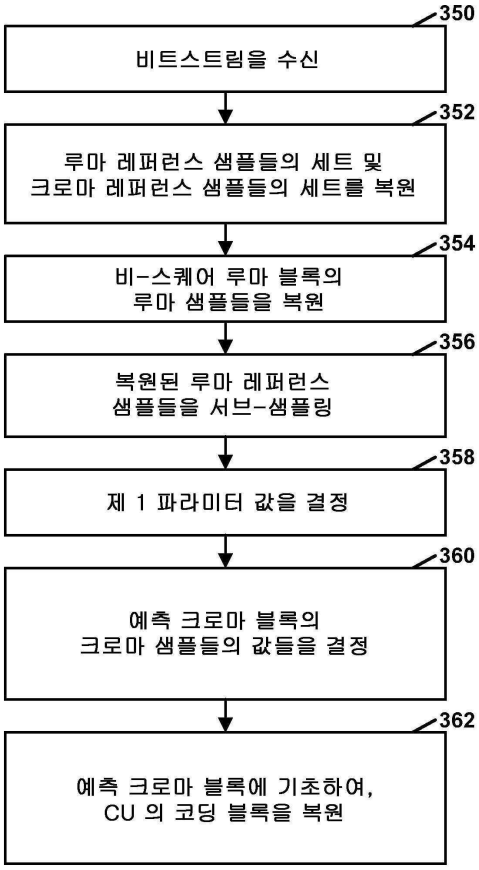
도면27


디코딩된
비디오
비디오 디코더
30
디코딩된
픽처 버퍼
262
필터 유닛
260
258
예측
프로세싱 유닛
252
LM-기반
디코딩 유닛
265
모션 보상
유닛
264
인트라-예측
프로세싱 유닛
266
역 변환
프로세싱 유닛
256
역 양자화
유닛
254
엔트로피
디코딩 유닛
250
비디오 데이터
메모리
251
인코딩된 비디오
비트스트림

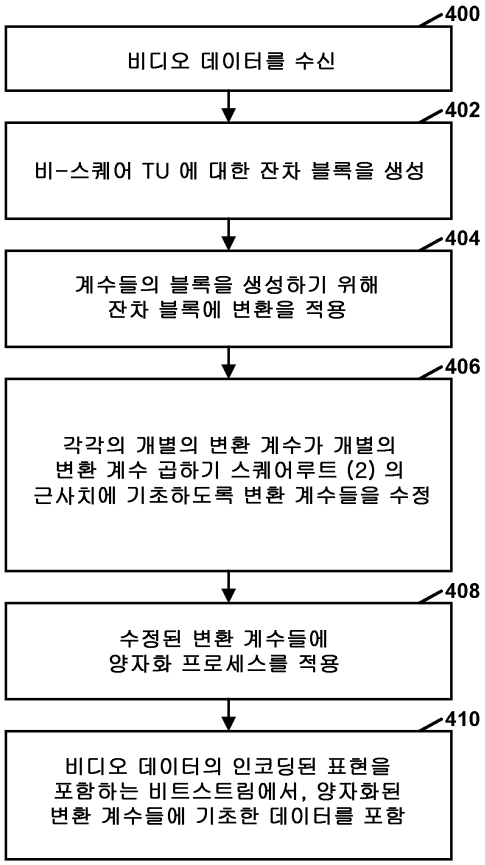
도면28


300
비디오 데이터를 수신
302
루마 레퍼런스 샘플들의 세트 및
크로마 레퍼런스 샘플들의 세트를 복원
304
비-스퀘어 루마 블록의
루마 샘플들을 복원
306
루마 레퍼런스 샘플들을 서브-샘플링
308
제 1 파라미터를 결정
310
크로마 샘플들의 값들을 결정
312
예측 크로마 블록에 기초하여,
CU 에 대한 잔차 데이터를 획득
314
비트스트림에서, CU 에 대한
잔차 데이터를 나타내는 데이터를 포함

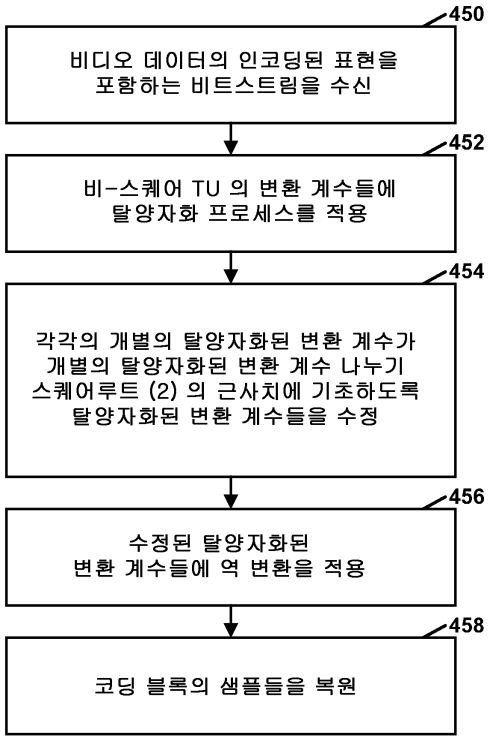
도면29


350
비트스트림을 수신
352
루마 레퍼런스 샘플들의 세트 및
크로마 레퍼런스 샘플들의 세트를 복원
354
비-스퀘어 루마 블록의
루마 샘플들을 복원
356
복원된 루마 레퍼런스
샘플들을 서브-샘플링
358
제 1 파라미터 값을 결정
360
예측 크로마 블록의
크로마 샘플들의 값들을 결정
362
예측 크로마 블록에 기초하여,
CU 의 코딩 블록을 복원

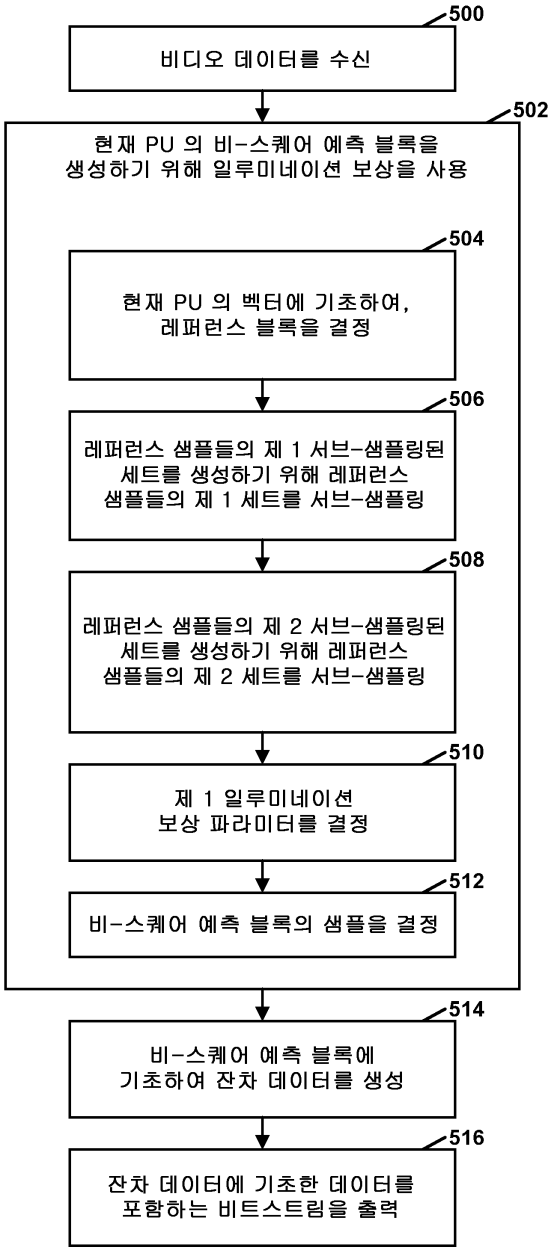
도면30


400
비디오 데이터를 수신
402
비-스퀘어 TU 에 대한 잔차 블록을 생성
404
계수들의 블록을 생성하기 위해
잔차 블록에 변환을 적용
406
각각의 개별의 변환 계수가 개별의
변환 계수 곱하기 스퀘어루트 (2) 의
근사치에 기초하도록 변환 계수들을 수정
408
수정된 변환 계수들에
양자화 프로세스를 적용
410
비디오 데이터의 인코딩된 표현을
포함하는 비트스트림에서, 양자화된
변환 계수들에 기초한 데이터를 포함

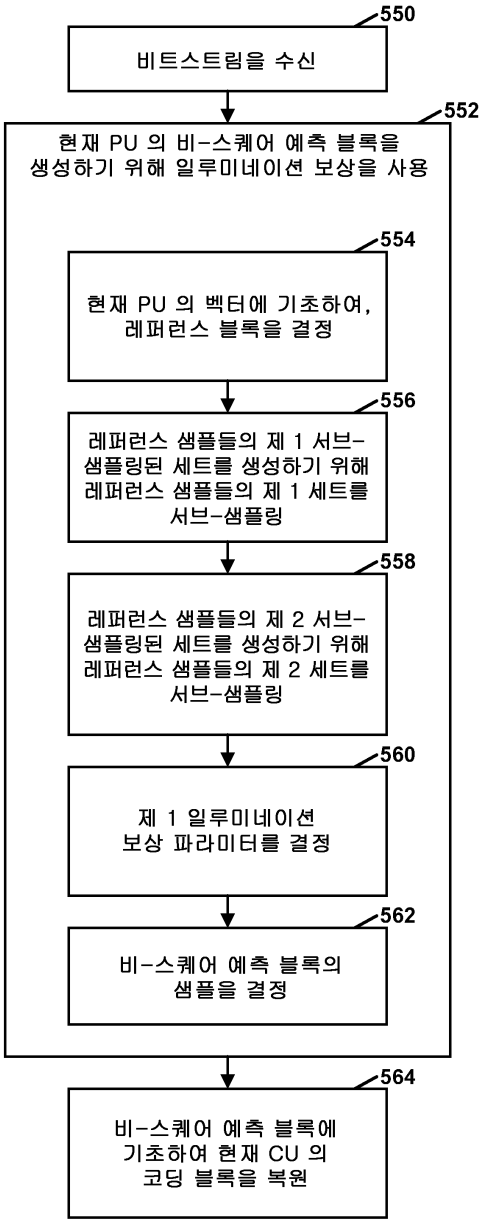
도면31


450
비디오 데이터의 인코딩된 표현을
포함하는 비트스트림을 수신
452
비-스퀘어 TU 의 변환 계수들에
탈양자화 프로세스를 적용
454
각각의 개별의 탈양자화된 변환 계수가
개별의 탈양자화된 변환 계수 나누기
스퀘어루트 (2) 의 근사치에 기초하도록
탈양자화된 변환 계수들을 수정
456
수정된 탈양자화된
변환 계수들에 역 변환을 적용
458
코딩 블록의 샘플들을 복원

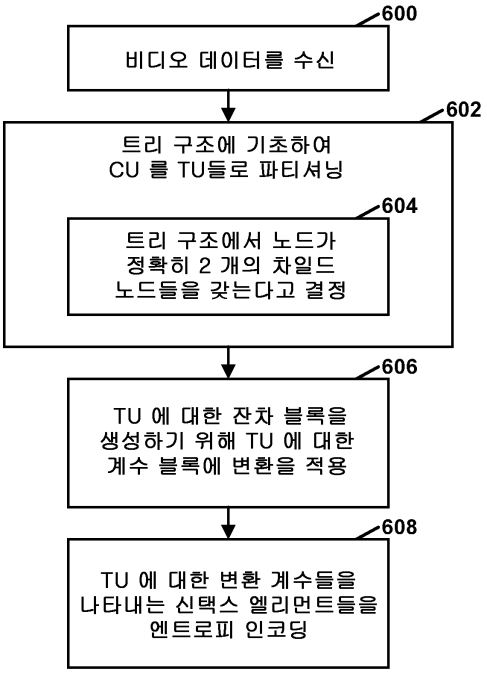
도면32


500
비디오 데이터를 수신
502
현재 PU 의 비-스퀘어 예측 블록을 생성하기 위해 일루미네이션 보상을 사용
504
현재 PU 의 벡터에 기초하여, 레퍼런스 블록을 결정
506
레퍼런스 샘플들의 제 1 서브-샘플링된 세트를 생성하기 위해 레퍼런스 샘플들의 제 1 세트를 서브-샘플링
508
레퍼런스 샘플들의 제 2 서브-샘플링된 세트를 생성하기 위해 레퍼런스 샘플들의 제 2 세트를 서브-샘플링
510
제 1 일루미네이션 보상 파라미터를 결정
512
비-스퀘어 예측 블록의 샘플을 결정
514
비-스퀘어 예측 블록에 기초하여 잔차 데이터를 생성
516
잔차 데이터에 기초한 데이터를 포함하는 비트스트림을 출력

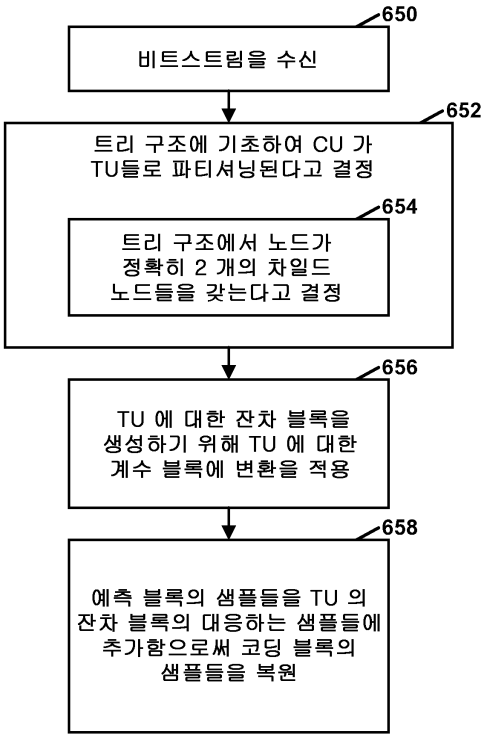
도면33


550
비트스트림을 수신
552
현재 PU 의 비-스퀘어 예측 블록을 생성하기 위해 일루미네이션 보상을 사용
554
현재 PU 의 벡터에 기초하여, 레퍼런스 블록을 결정
556
레퍼런스 샘플들의 제 1 서브-샘플링된 세트를 생성하기 위해 레퍼런스 샘플들의 제 1 세트를 서브-샘플링
558
레퍼런스 샘플들의 제 2 서브-샘플링된 세트를 생성하기 위해 레퍼런스 샘플들의 제 2 세트를 서브-샘플링
560
제 1 일루미네이션 보상 파라미터를 결정
562
비-스퀘어 예측 블록의 샘플을 결정
564
비-스퀘어 예측 블록에 기초하여 현재 CU 의 코딩 블록을 복원

도면34


600
비디오 데이터를 수신
602
트리 구조에 기초하여
CU 를 TU들로 파티셔닝
604
트리 구조에서 노드가
정확히 2 개의 차일드
노드들을 갖는다고 결정
606
TU 에 대한 잔차 블록을
생성하기 위해 TU 에 대한
계수 블록에 변환을 적용
608
TU 에 대한 변환 계수들을
나타내는 신택스 엘리먼트들을
엔트로피 인코딩

도면35


650
비트스트림을 수신
652
트리 구조에 기초하여 CU 가
TU들로 파티셔닝된다고 결정
654
트리 구조에서 노드가
정확히 2 개의 차일드
노드들을 갖는다고 결정
656
TU 에 대한 잔차 블록을
생성하기 위해 TU 에 대한
계수 블록에 변환을 적용
658
예측 블록의 샘플들을 TU 의
잔차 블록의 대응하는 샘플들에
추가함으로써 코딩 블록의
샘플들을 복원