(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2022-0064969
(43) 공개일자 2022년05월19일

(51) 국제특허분류(Int. Cl.)
H04N 19/117 (2014.01) *H04N 19/132* (2014.01)
H04N 19/159 (2014.01) *H04N 19/176* (2014.01)
H04N 19/186 (2014.01) *H04N 19/70* (2014.01)
H04N 19/82 (2014.01)
(52) CPC특허분류
H04N 19/117 (2015.01)
H04N 19/132 (2015.01)
(21) 출원번호 10-2022-7008823
(22) 출원일자(국제) 2020년09월23일
심사청구일자 없음
(85) 번역문제출일자 2022년03월16일
(86) 국제출원번호 PCT/US2020/052231
(87) 국제공개번호 WO 2021/061782
국제공개일자 2021년04월01일
(30) 우선권주장
62/904,508 2019년09월23일 미국(US)
17/028,209 2020년09월22일 미국(US)

(71) 출원인
퀄컴 인코포레이티드
미국 92121-1714 캘리포니아주 샌 디에고 모어하우스 드라이브 5775
(72) 발명자
후 난
미국 92121-1714 캘리포니아주 샌디에고 모어하우스 드라이브 5775
동 제
미국 92121-1714 캘리포니아주 샌디에고 모어하우스 드라이브 5775
(뒷면에 계속)
(74) 대리인
특허법인코리아나

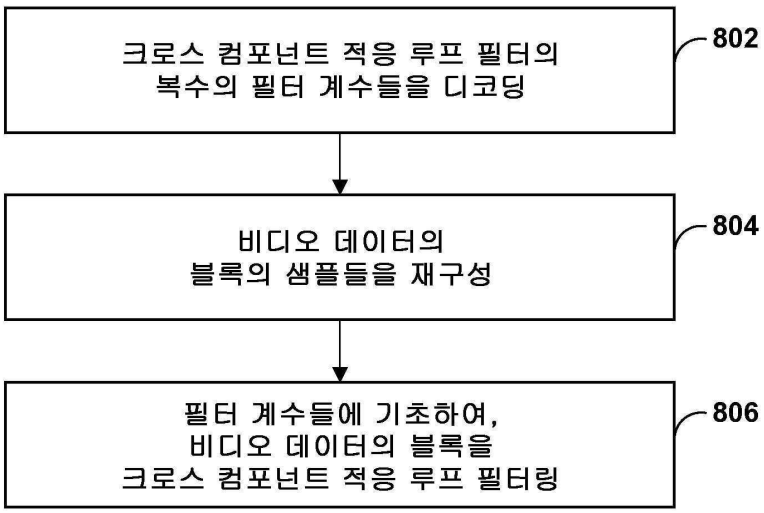
전체 청구항 수 : 총 20 항

(54) 발명의 명칭 **비디오 코딩을 위한 크로스-컴포넌트 적응형 루프 필터링을 위한 비트 시프팅**

(57) 요 약

예시적인 방법은, 크로스-컴포넌트 적응형 루프 필터의 복수의 필터 계수들을 디코딩하는 단계로서, 복수의 필터 계수들 중 특정 필터 계수를 디코딩하는 것은, 인코딩된 비디오 비트스트림으로부터, 로그 베이스 2 의 상기 특정 필터 계수의 절대 값을 표현하는 지수 값을 2 의 상기 지수 값의 거듭제곱으로서 특정하는 신택스 엘리먼트를 디코딩하는 것; 및, 상기 지수 값에 기초하여 상기 특정 필터 계수의 값을 결정하는 것을 포함하는, 상기 크로스-컴포넌트 적응형 루프 필터의 복수의 필터 계수들을 디코딩하는 단계; 비디오 데이터의 블록의 샘플들을 재구성하는 단계; 및, 상기 복수의 필터 계수들에 기초하여, 상기 비디오 데이터의 블록을 크로스-컴포넌트 적응형 루프 필터링하는 단계를 포함한다.

대 표 도 - 도8



(52) CPC특허분류
H04N 19/159 (2015.01)
H04N 19/176 (2015.01)
H04N 19/186 (2015.01)
H04N 19/70 (2015.01)
H04N 19/82 (2015.01)

(72) 발명자
세레긴 바딤
미국 92121-1714 캘리포니아주 샌디에고 모어하우스 드라이브 5775

카르체비츠 마르타
미국 92121-1714 캘리포니아주 샌디에고 모어하우스 드라이브 5775

명 세 서

청구범위

청구항 1

비디오 데이터를 디코딩하는 방법으로서,

상기 방법은,

크로스-컴포넌트 적응형 루프 필터의 복수의 필터 계수들을 디코딩하는 단계로서, 상기 복수의 필터 계수들 중 특정 필터 계수를 디코딩하는 것은:

인코딩된 비디오 비트스트림으로부터, 로그 베이스 2 의 상기 특정 필터 계수의 절대 값을 표현하는 지수 값을 2 의 상기 지수 값의 거듭제곱으로서 특정하는 신택스 엘리먼트를 디코딩하는 것; 및

상기 지수 값에 기초하여 상기 특정 필터 계수의 값을 결정하는 것

을 포함하는, 상기 크로스-컴포넌트 적응형 루프 필터의 복수의 필터 계수들을 디코딩하는 단계;

비디오 데이터의 블록의 샘플들을 재구성하는 단계; 및

상기 복수의 필터 계수들에 기초하여, 상기 비디오 데이터의 블록을 크로스-컴포넌트 적응형 루프 필터링하는 단계를 포함하는, 비디오 데이터를 디코딩하는 방법.

청구항 2

제 1 항에 있어서,

상기 복수의 필터 계수들 모두의 절대 값들은 0 또는 2의 거듭제곱으로 제한되는, 비디오 데이터를 디코딩하는 방법.

청구항 3

제 1 항에 있어서,

상기 특정 필터 계수를 디코딩하는 것은,

상기 지수 값이 0 이외의 값인 것에 응답하여, 상기 인코딩된 비디오 비트스트림으로부터, 상기 특정 필터 계수의 부호를 특정하는 신택스 엘리먼트를 디코딩하는 것을 더 포함하며,

상기 특정 필터 계수의 값을 결정하는 것은, 상기 부호에 기초하여 상기 특정 필터 계수의 값을 결정하는 것을 더 포함하는, 비디오 데이터를 디코딩하는 방법.

청구항 4

제 3 항에 있어서,

상기 특정 필터 계수의 값을 결정하는 것은, 다음 식:

$$c(i) = sign(i) * 2^{c'(i)+1}$$

에 따라 상기 특정 필터 계수의 값을 결정하는 것을 포함하고,

여기서, $c(i)$ 는 상기 특정 필터 계수의 값이고, $sign(i)$ 는 부호가 음인 경우 -1 이고 부호가 양인 경우 +1 이며, $c'(i)$ 는 상기 특정 필터 계수에 대한 상기 지수 값인, 비디오 데이터를 디코딩하는 방법.

청구항 5

제 1 항에 있어서,

상기 크로스-컴포넌트 적응형 루프 필터링은, 상기 복수의 필터 계수들의 값들에 기초하여, 곱셈을 수행함이 없이 상기 비디오 데이터의 블록의 샘플들을 비트-시프트하는 것을 포함하는, 비디오 데이터를 디코딩하는 방법.

청구항 6

제 1 항에 있어서,

상기 지수 값을 특정하는 상기 신택스 엘리먼트를 디코딩하는 것은, 고정 길이 코드를 사용하여 상기 지수 값을 특정하는 상기 신택스 엘리먼트를 디코딩하는 것을 포함하는, 비디오 데이터를 디코딩하는 방법.

청구항 7

비디오 데이터를 인코딩하는 방법으로서,

상기 방법은,

크로스-컴포넌트 적응형 루프 필터의 복수의 필터 계수들의 값들을 인코딩하는 단계로서, 상기 복수의 필터 계수들 중 특정 필터 계수의 값을 인코딩하는 것은:

인코딩된 비디오 비트스트림에서, 로그 베이스 2 의 상기 특정 필터 계수의 절대 값을 표현하는 지수 값을 2 의 상기 지수 값의 거듭제곱으로서 특정하는 신택스 엘리먼트를 인코딩하는 것을 포함하는, 상기 크로스-컴포넌트 적응형 루프 필터의 복수의 필터 계수들의 값들을 인코딩하는 단계;

비디오 데이터의 블록의 샘플들을 재구성하는 단계; 및

상기 복수의 필터 계수들의 값들에 기초하여, 상기 비디오 데이터의 블록을 크로스-컴포넌트 적응형 루프 필터링하는 단계를 포함하는, 비디오 데이터를 인코딩하는 방법.

청구항 8

제 7 항에 있어서,

상기 복수의 필터 계수들 모두의 절대 값들은 0 또는 2의 거듭제곱으로 제한되는, 비디오 데이터를 인코딩하는 방법.

청구항 9

제 7 항에 있어서,

상기 특정 필터 계수를 인코딩하는 것은,

상기 특정 필터 계수가 0 이외의 값을 갖는 것에 응답하여, 상기 인코딩된 비디오 비트스트림에서, 상기 특정 필터 계수의 부호를 특정하는 신택스 엘리먼트를 인코딩하는 것을 더 포함하는, 비디오 데이터를 인코딩하는 방법.

청구항 10

제 7 항에 있어서,

크로스-컴포넌트 적응형 루프 필터링은, 상기 복수의 필터 계수들의 값들에 기초하여, 곱셈을 수행함이 없이 상기 비디오 데이터의 블록의 샘플들을 비트-시프트하는 것을 포함하는, 비디오 데이터를 인코딩하는 방법.

청구항 11

제 7 항에 있어서,

상기 지수 값을 특정하는 상기 신택스 엘리먼트를 인코딩하는 것은, 고정 길이 코드를 사용하여 상기 지수 값을 특정하는 상기 신택스 엘리먼트를 인코딩하는 것을 포함하는, 비디오 데이터를 인코딩하는 방법.

청구항 12

비디오 데이터를 디코딩하기 위한 디바이스로서,

상기 디바이스는,

인코딩된 비디오 비트스트림의 적어도 부분을 저장하도록 구성된 메모리; 및

회로에서 구현되는 하나 이상의 프로세서들을 포함하고,

상기 하나 이상의 프로세서들은,

크로스-컴포넌트 적응형 루프 필터의 복수의 필터 계수들을 디코딩하는 것으로서, 상기 복수의 필터 계수들 중 특정 필터 계수를 디코딩하기 위해, 상기 하나 이상의 프로세서들은:

상기 인코딩된 비디오 비트스트림으로부터, 로그 베이스 2 의 상기 특정 필터 계수의 절대 값을 표현하는 지수 값을 2 의 상기 지수 값의 거듭제곱으로서 특정하는 신택스 엘리먼트를 디코딩하고; 그리고

상기 지수 값에 기초하여 상기 특정 필터 계수의 값을 결정하도록

구성되는, 상기 크로스-컴포넌트 적응형 루프 필터의 복수의 필터 계수들을 디코딩하는 것을 행하고;

비디오 데이터의 블록의 샘플들을 재구성하며; 그리고

상기 복수의 필터 계수들에 기초하여, 상기 비디오 데이터의 블록을 크로스-컴포넌트 적응형 루프 필터링하도록 구성되는, 비디오 데이터를 디코딩하기 위한 디바이스.

청구항 13

제 12 항에 있어서,

상기 복수의 필터 계수들 모두의 절대 값들은 0 또는 2의 거듭제곱으로 제한되는, 비디오 데이터를 디코딩하기 위한 디바이스.

청구항 14

제 12 항에 있어서,

상기 특정 필터 계수를 디코딩하기 위해, 상기 하나 이상의 프로세서들은 추가로,

상기 지수 값이 0 이외의 값인 것에 응답하여, 상기 인코딩된 비디오 비트스트림으로부터, 상기 특정 필터 계수의 부호를 특정하는 신택스 엘리먼트를 디코딩하도록 구성되며,

여기서, 상기 특정 필터 계수의 값을 결정하기 위해, 상기 하나 이상의 프로세서들은 추가로, 상기 부호에 기초하여 상기 특정 필터 계수의 값을 결정하도록 구성되는, 비디오 데이터를 디코딩하기 위한 디바이스.

청구항 15

제 14 항에 있어서,

상기 특정 필터 계수의 값을 결정하기 위해, 상기 하나 이상의 프로세서들은, 다음 식:

$$c(i) = sign(i) * 2^{c'(i)+1}$$

에 따라 상기 특정 필터 계수의 값을 결정하도록 구성되고,

여기서, $c(i)$ 는 상기 특정 필터 계수의 값이고, $sign(i)$ 는 부호가 음인 경우 -1 이고 부호가 양인 경우 +1 이며, $c'(i)$ 는 상기 특정 필터 계수에 대한 상기 지수 값인, 비디오 데이터를 디코딩하기 위한 디바이스.

청구항 16

제 12 항에 있어서,

상기 크로스-컴포넌트 적응형 루프 필터링하기 위해, 상기 하나 이상의 프로세서들은, 상기 복수의 필터 계수들의 값들에 기초하여, 곱셈을 수행함이 없이 상기 비디오 데이터의 블록의 샘플들을 비트-시프트하도록 구성되는, 비디오 데이터를 디코딩하기 위한 디바이스.

청구항 17

비디오 데이터를 인코딩하기 위한 디바이스로서,

상기 디바이스는,

인코딩된 비디오 비트스트림의 적어도 부분을 저장하도록 구성된 메모리; 및

회로에서 구현되는 하나 이상의 프로세서들을 포함하고,

상기 하나 이상의 프로세서들은,

크로스-컴포넌트 적응형 루프 필터의 복수의 필터 계수들의 값들을 인코딩하는 것으로서, 상기 복수의 필터 계수들 중 특정 필터 계수의 값을 인코딩하기 위해, 상기 하나 이상의 프로세서들은:

상기 인코딩된 비디오 비트스트림에서, 로그 베이스 2 의 상기 특정 필터 계수의 절대 값을 표현하는 지수 값을 2 의 상기 지수 값의 거듭제곱으로서 특정하는 신택스 엘리먼트를 인코딩하도록 구성되는, 상기 크로스-컴포넌트 적응형 루프 필터의 복수의 필터 계수들의 값들을 인코딩하는 것을 행하고;

비디오 데이터의 블록의 샘플들을 재구성하며; 그리고

상기 복수의 필터 계수들의 값들에 기초하여, 상기 비디오 데이터의 블록을 크로스-컴포넌트 적응형 루프 필터링하도록 구성되는, 비디오 데이터를 인코딩하기 위한 디바이스.

청구항 18

제 17 항에 있어서,

상기 복수의 필터 계수들 모두의 절대 값들은 0 또는 2의 거듭제곱으로 제한되는, 비디오 데이터를 인코딩하기 위한 디바이스.

청구항 19

제 17 항에 있어서,

상기 특정 필터 계수를 인코딩하기 위해, 상기 하나 이상의 프로세서들은 추가로,

상기 특정 필터 계수가 0 이외의 값을 갖는 것에 응답하여, 상기 인코딩된 비디오 비트스트림으로부터, 상기 특정 필터 계수의 부호를 특정하는 신택스 엘리먼트를 인코딩하도록 구성되는, 비디오 데이터를 인코딩하기 위한 디바이스.

청구항 20

제 17 항에 있어서,

상기 크로스-컴포넌트 적응형 루프 필터링하기 위해, 상기 하나 이상의 프로세서들은, 상기 복수의 필터 계수들의 값들에 기초하여, 곱셈을 수행함이 없이 상기 비디오 데이터의 블록의 샘플들을 비트-시프트하도록 구성되는, 비디오 데이터를 인코딩하기 위한 디바이스.

발명의 설명

기 술 분 야

[0001] 본 출원은 2019년 9월 23일자로 출원된 미국 가출원 제62/904,508호의 이익을 주장하는, 2020년 9월 22일자로 출원된 미국출원 제17/028,209호에 대해 우선권 주장하며, 이들 출원들의 전체 내용은 본원에 참조에 의해 통합된다.

[0002] **기술 분야**

[0003] 본 개시는 비디오 인코딩 및 비디오 디코딩을 포함하는 비디오 코딩에 관한 것이다.

배 경 기 술

[0004] **배경**

[0005] 디지털 비디오 능력들은 디지털 텔레비전들, 디지털 직접 브로드캐스트 시스템들, 무선 브로드캐스트 시스템들, 개인용 디지털 보조기들 (PDA들), 랩톱 또는 데스크톱 컴퓨터들, 태블릿 컴퓨터들, e-북 리더들, 디지털 카메라들, 디지털 레코딩 디바이스들, 디지털 미디어 플레이어들, 비디오 게이밍 디바이스들, 비디오 게임 콘솔들, 셀룰러 또는 위성 무선 전화기들, 소위 "스마트 폰들", 비디오 텔레컨퍼런싱 디바이스들, 비디오 스트리밍 디바이스들 등을 포함한, 광범위한 디바이스들에 통합될 수 있다. 디지털 비디오 디바이스들은, MPEG-2, MPEG-4, ITU-T H.263 또는 ITU-T H.264/MPEG-4, Part 10, AVC (Advanced Video Coding), ITU-T H.265, HEVC (High Efficiency Video Coding) 에 의해 정의되는 표준들, 및 그러한 표준들의 확장들에서 설명된 것들과 같은 비디오 코딩 기법들을 구현한다. 비디오 디바이스들은 그러한 비디오 코딩 기술들을 구현함으로써 디지털 비디오 정보를 더 효율적으로 송신, 수신, 인코딩, 디코딩, 및/또는 저장할 수도 있다.

[0006] 비디오 코딩 기법들은 비디오 시퀀스들에 내재한 리던던시를 감소 또는 제거하기 위해 공간 (인트라 픽처) 예측 및/또는 시간 (인터 픽처) 예측을 포함한다. 블록 기반 비디오 코딩에 대해, 비디오 슬라이스 (즉, 비디오 픽처 또는 비디오 픽처의 일부) 는 비디오 블록들로 파티셔닝될 수도 있으며, 이 비디오 블록들은 또한 코딩 트리 유닛들 (CTU들), 코딩 유닛들 (CU들) 및/또는 코딩 노드들로서 지칭될 수도 있다. 픽처의 인트라-코딩된 (I) 슬라이스에서의 비디오 블록들은 동일한 픽처에 있어서 이웃하는 블록들에서의 레퍼런스 샘플들에 대한 공간적 예측을 사용하여 인코딩된다. 픽처의 인터-코딩된 (P 또는 B) 슬라이스에서의 비디오 블록들은 동일 픽처의 이웃하는 블록들에서의 레퍼런스 샘플들에 대한 공간 예측, 또는 다른 레퍼런스 픽처들에서의 레퍼런스 샘플들에 대한 시간 예측을 이용할 수도 있다. 픽처들은 프레임들로 지칭될 수도 있고, 레퍼런스 픽처들은 레퍼런스 프레임들로 지칭될 수도 있다.

발명의 내용

[0007] **요약**

[0008] 일반적으로, 본 개시는 비디오 데이터의 크로스-컴포넌트 적응형 루프 필터링 (cross-component adaptive loop filtering; CC-ALF) 에 관련된 기술들을 설명한다. ALF 를 수행하기 위해, 비디오 코더는 상이한 계수 세트들을 사용하여 (예를 들어, 루마 블록을 필터링하기 위한 루마 계수 세트 및 크로마 블록들을 필터링하기 위한 하나 이상의 크로마 계수 세트들을 사용하여) 대응하는 루마 및 크로마 블록들을 개별적으로 필터링할 수도 있다. 그러나, 루마 블록은 코딩 루프에서 대응하는 크로마 블록에서 손실될 수도 있는 디테일들을 포함할 수도 있다. 이와 같이, 비디오 코더는 루마 블록으로부터의 정보가 대응하는 크로마 블록을 향상시키기 위해 사용되는 CC-ALF 를 수행할 수도 있다.

[0009] 예를 들어, 비디오 코더는 루마 블록을 크로마 필터 계수들의 제 1 세트로 필터링하여 제 1 크로마 컴포넌트 (예를 들어, Cb) 에 대한 중간 블록을 생성하고, 루마 블록을 크로마 필터 계수들의 제 2 세트로 필터링하여 제 2 크로마 컴포넌트 (예를 들어, Cr) 에 대한 중간 블록을 생성할 수도 있다. 비디오 인코더는 CC-ALF 에 대해 사용되는 필터 계수들의 값들 (예를 들어, 루마 블록을 필터링하는데 사용되는 크로마 필터 계수들의 적어도 제 1 및 제 2 세트들) 을 인코딩된 비디오 비트스트림에서의 하나 이상의 신택스 엘리먼트들로서 비디오 디코더에 시그널링할 수도 있다. 비디오 코더는 그 후 크로마 컴포넌트들의 ALF 필터링된 크로마 블록들에 각각의 중간 블록들을 부가할 수도 있다.

[0010] 크로마 컴포넌트에 대한 중간 블록을 생성하기 위해 루마 블록을 필터링하기 위해, 비디오 코더는 루마 블록의 각각의 샘플에 대해 복수의 곱셈 연산들을 수행할 수도 있다. 예를 들어, 비디오 코더는 루마 블록의 샘플들에 의해 승산된 크로마 필터 계수들의 합산으로서 루마 블록의 특정 샘플에 대한 필터링된 값을 계산할 수도 있다. 이와 같이, CC-ALF 를 수행하는 것은 많은 수의 곱셈 연산들 (예를 들어, 8x8 루마 블록에 대해 7개) 을 수반할 수도 있다. 이러한 높은 수의 곱셈 동작들의 수행은 비디오 코더에 대한 리소스 집약적 노력일 수도 있으며, 이는 코딩 시간 및/또는 전력 소비를 바람직하지 않게 증가시킬 수도 있다.

[0011] 본 개시의 하나 이상의 기법들에 따르면, 비디오 코더는 필터 계수들의 절대 값들이 제로 또는 2 의 거듭제곱이도록 제한되도록 CC-ALF에 대한 필터 계수들을 코딩할 수도 있다 (예를 들어, 비디오 인코더는 인코딩할 수도 있고 비디오 디코더는 디코딩할 수도 있다). 중간 크로마 블록들을 생성하기 위해 필터 계수들을 사용하여 루마 블록을 필터링할 때, 비디오 코더는 곱셈 연산들을 비트-시프트 연산들 (예를 들어, 좌측-시프트 및 우측-

시프트 연산들) 로 대체할 수도 있다. 필터 계수들의 절대값들이 제로 또는 2의 거듭제곱으로 제한되기 때문에, 비트-시프트 연산들로 곱셈 연산들의 대체는 수학적으로 동등할 수도 있다 (즉, 동일한 중간 크로마 블록을 산출할 수도 있다). 그러나, 수학적으로 동등하지만, 비트-시프트 연산들은 곱셈 연산들보다 실질적으로 덜 리소스-집약적일 수도 있다. 추가적으로, 전용 하드웨어(예를 들어, 애플리케이션 특정적 집적 회로(ASIC))로 구현될 때, 비트-시프트 연산들을 수행하는 데 필요한 하드웨어는 곱셈 연산들을 수행하는 데 필요한 하드웨어보다 단순할 수도 있다. 이러한 방식으로, 본 개시의 기법들은 CC-ALF 의 리소스 요건들을 감소시킨다.

[0012] 일 예로서, 방법은, 크로스-컴포넌트 적응형 루프 필터 (cross-component adaptive loop filter) 의 복수의 필터 계수들을 디코딩하는 단계로서, 복수의 필터 계수들 중 특정 필터 계수를 디코딩하는 것은: 인코딩된 비디오 비트스트림으로부터, 로그 베이스 2 의 상기 특정 필터 계수의 절대 값을 표현하는 지수 값을 2 의 상기 지수 값의 거듭제곱으로서 특정하는 신택스 엘리먼트를 디코딩하는 것; 및, 상기 지수 값에 기초하여 상기 특정 필터 계수의 값을 결정하는 것을 포함하는, 상기 크로스-컴포넌트 적응형 루프 필터의 복수의 필터 계수들을 디코딩하는 단계; 비디오 데이터의 블록의 샘플들을 재구성하는 단계; 및, 상기 복수의 필터 계수들에 기초하여, 상기 비디오 데이터의 블록을 크로스-컴포넌트 적응형 루프 필터링하는 단계를 포함한다.

[0013] 다른 예로서, 방법은, 크로스-컴포넌트 적응형 루프 필터의 복수의 필터 계수들의 값들을 인코딩하는 단계로서, 상기 복수의 필터 계수들 중 특정 필터 계수의 값을 인코딩하는 것은: 인코딩된 비디오 비트스트림에서, 로그 베이스 2 의 상기 특정 필터 계수의 절대 값을 표현하는 지수 값을 2 의 상기 지수 값의 거듭제곱으로서 특정하는 신택스 엘리먼트를 인코딩하는 것을 포함하는, 상기 크로스-컴포넌트 적응형 루프 필터의 복수의 필터 계수들의 값들을 인코딩하는 단계; 비디오 데이터의 블록의 샘플들을 재구성하는 단계; 및, 상기 복수의 필터 계수들의 값들에 기초하여, 상기 비디오 데이터의 블록을 크로스-컴포넌트 적응형 루프 필터링하는 단계를 포함한다.

[0014] 다른 예로서, 디바이스는, 인코딩된 비디오 비트스트림의 적어도 부분을 저장하도록 구성된 메모리; 및, 회로에서 구현된 하나 이상의 프로세서들을 포함하고, 상기 하나 이상의 프로세서들은: 크로스-컴포넌트 적응형 루프 필터의 복수의 필터 계수들을 디코딩하는 것으로서, 상기 복수의 필터 계수들 중 특정 필터 계수를 디코딩하기 위해, 상기 하나 이상의 프로세서들은: 상기 인코딩된 비디오 비트스트림으로부터, 로그 베이스 2 의 상기 특정 필터 계수의 절대 값을 표현하는 지수 값을 2 의 상기 지수 값의 거듭제곱으로서 특정하는 신택스 엘리먼트를 디코딩하고; 그리고, 상기 지수 값에 기초하여 상기 특정 필터 계수의 값을 결정하도록 구성되는, 상기 크로스-컴포넌트 적응형 루프 필터의 복수의 필터 계수들을 디코딩하는 것을 행하고; 비디오 데이터의 블록의 샘플들을 재구성하며; 그리고, 상기 복수의 필터 계수들에 기초하여, 상기 비디오 데이터의 블록을 크로스-컴포넌트 적응형 루프 필터링하도록 구성된다.

[0015] 다른 예로서, 디바이스는, 인코딩된 비디오 비트스트림의 적어도 부분을 저장하도록 구성된 메모리; 및, 회로에서 구현된 하나 이상의 프로세서들을 포함하고, 상기 하나 이상의 프로세서들은: 크로스-컴포넌트 적응형 루프 필터의 복수의 필터 계수들의 값들을 인코딩하는 것으로서, 상기 복수의 필터 계수들 중 특정 필터 계수의 값을 인코딩하기 위해, 상기 하나 이상의 프로세서들은: 상기 인코딩된 비디오 비트스트림에서, 로그 베이스 2 의 상기 특정 필터 계수의 절대 값을 표현하는 지수 값을 2 의 상기 지수 값의 거듭제곱으로서 특정하는 신택스 엘리먼트를 인코딩하도록 구성되는, 상기 크로스-컴포넌트 적응형 루프 필터의 복수의 필터 계수들의 값들을 인코딩하는 것을 행하고; 비디오 데이터의 블록의 샘플들을 재구성하며; 그리고, 상기 복수의 필터 계수들의 값들에 기초하여, 상기 비디오 데이터의 블록을 크로스-컴포넌트 적응형 루프 필터링하도록 구성된다.

[0016] 하나 이상의 예들의 상세들이 첨부 도면들 및 이하의 설명에서 전개된다. 다른 특징들, 목적들, 및 이점들은 설명, 도면들, 및 청구항들로부터 명백할 것이다.

도면의 간단한 설명

[0017] **도면들의 간단한 설명**

도 1 은 본 개시의 기법들을 수행할 수도 있는 예시적인 비디오 인코딩 및 디코딩 시스템을 나타내는 블록도이다.

도 2a 및 도 2b 는 예시적인 쿼드트리 바이너리 트리 (QTBT) 구조, 및 대응하는 코딩 트리 유닛 (CTU) 을 나타내는 개념도들이다.

도 3 은 본 개시의 기법들을 수행할 수도 있는 예시적인 비디오 인코더를 나타내는 블록도이다.

도 4 는 본 개시의 기법들을 수행할 수도 있는 예시적인 비디오 디코더를 나타내는 블록도이다.

도 5 는 본 개시의 하나 이상의 기법들에 따른, 예시적인 필터 유닛을 나타내는 블록도이다.

도 6 은 본 개시의 기법들에 따른, 현재 블록을 인코딩하기 위한 예시적인 방법을 나타내는 플로우차트이다.

도 7 은 본 개시의 기법들에 따른, 현재 블록을 디코딩하기 위한 예시적인 방법을 나타내는 플로우차트이다.

도 8 은 본 개시의 하나 이상의 기법들에 따른, 현재 블록에 대한 크로스-컴포넌트 적응형 루프 필터링 (CC-ALF) 을 위한 예시의 방법을 나타내는 플로우챠트이다.

발명을 실시하기 위한 구체적인 내용

[0018] **상세한 설명**

[0019] 도 1 은 본 개시의 기법들을 수행할 수도 있는 비디오 인코딩 및 디코딩 시스템 (100) 의 예를 나타내는 블록도이다. 본 개시의 기법들은 일반적으로 비디오 데이터를 코딩 (인코딩 및/또는 디코딩) 하는 것과 관련된다. 일반적으로, 비디오 데이터는 비디오를 프로세싱하기 위한 임의의 데이터를 포함한다. 따라서, 비디오 데이터는 원시, 코딩되지 않은 비디오, 인코딩된 비디오, 디코딩된 (예를 들어, 재구성된) 비디오, 및 비디오 메타데이터, 이를 테면 시그널링 데이터를 포함할 수도 있다.

[0020] 도 1 에 도시된 바와 같이, 시스템 (100) 은 이 예에서 목적지 디바이스 (116) 에 의해 디코딩 및 디스플레이될 인코딩된 비디오 데이터를 제공하는 소스 디바이스 (102) 를 포함한다. 특히, 소스 디바이스 (102) 는 컴퓨터 판독 가능 매체 (110) 를 통해 목적지 디바이스 (116) 에 비디오 데이터를 제공한다. 소스 디바이스 (102) 및 목적지 디바이스 (116) 는 데스크톱 컴퓨터들, 노트북 (즉, 랩톱) 컴퓨터들, 태블릿 컴퓨터들, 셋톱 박스들, 전화기 핸드셋, 예컨대 스마트폰들, 텔레비전들, 카메라들, 디스플레이 디바이스들, 디지털 미디어 플레이어들, 비디오 게이밍 콘솔들, 비디오 스트리밍 디바이스 등을 포함한, 광범위한 디바이스들 중 임의의 것을 포함할 수도 있다. 일부 경우들에서, 소스 디바이스 (102) 및 목적지 디바이스 (116) 는 무선 통신을 위해 장비될 수도 있고, 따라서 무선 통신 디바이스들로 지칭될 수도 있다.

[0021] 도 1 의 예에서, 소스 디바이스 (102) 는 비디오 소스 (104), 메모리 (106), 비디오 인코더 (200), 및 출력 인터페이스 (108) 를 포함한다. 목적지 디바이스 (116) 는 입력 인터페이스 (122), 비디오 디코더 (300), 메모리 (120), 및 디스플레이 디바이스 (118) 를 포함한다. 본 개시에 따르면, 소스 디바이스 (102) 의 비디오 인코더 (200) 및 목적지 디바이스 (116) 의 비디오 디코더 (300) 는 크로스-컴포넌트 적응형 루프 필터링을 수행위한 기법들을 적용하도록 구성될 수도 있다. 따라서, 소스 디바이스 (102) 는 비디오 인코딩 디바이스의 일 예를 나타내는 한편, 목적지 디바이스 (116) 는 비디오 디코딩 디바이스의 일 예를 나타낸다. 다른 예들에서, 소스 디바이스 및 목적지 디바이스는 다른 컴포넌트들 또는 배열들을 포함할 수도 있다. 예를 들어, 소스 디바이스 (102) 는 외부 카메라와 같은 외부 비디오 소스로부터 비디오 데이터를 수신할 수도 있다. 마찬가지로, 목적지 디바이스 (116) 는 통합된 디스플레이 디바이스를 포함하는 것보다는, 외부 디스플레이 디바이스와 인터페이스할 수도 있다.

[0022] 도 1 에서 도시된 시스템 (100) 은 단지 하나의 예일 뿐이다. 일반적으로, 임의의 디지털 비디오 인코딩 및/또는 디코딩 디바이스는 크로스-컴포넌트 적응형 루프 필터링을 위한 기법들을 수행할 수도 있다. 소스 디바이스 (102) 및 목적지 디바이스 (116) 는 소스 디바이스 (102) 가 목적지 디바이스 (116) 로의 송신을 위한 코딩된 비디오 데이터를 생성하는 이러한 코딩 디바이스들의 예들일 뿐이다. 본 개시는 데이터의 코딩 (인코딩 및/또는 디코딩) 을 수행하는 디바이스로서 "코딩" 디바이스를 지칭한다. 따라서, 비디오 인코더 (200) 및 비디오 디코더 (300) 는 코딩 디바이스들, 특히 각각 비디오 인코더 및 비디오 디코더의 예들을 나타낸다. 일부 예들에서, 소스 디바이스 (102) 및 목적지 디바이스 (116) 는 소스 디바이스 (102) 및 목적지 디바이스 (116) 의 각각이 비디오 인코딩 및 디코딩 컴포넌트들을 포함하도록 실질적으로 대칭 방식으로 동작할 수도 있다. 이로써, 시스템 (100) 은 예를 들어, 비디오 스트리밍, 비디오 플레이백, 비디오 브로드캐스팅 또는 비디오 텔레포니를 위해, 소스 디바이스 (102) 와 목적지 디바이스 (116) 사이의 일방향 또는 양방향 비디오 송신을 지원할 수도 있다.

[0023] 일반적으로, 비디오 소스 (104) 는 비디오 데이터 (즉, 원시, 코딩되지 않은 비디오 데이터) 의 소스를 나타내며 픽처들에 대한 데이터를 인코딩하는 비디오 인코더 (200) 에 비디오 데이터의 순차적인 일련의 픽처들 (또한

"프레임들" 로도 지칭됨) 을 제공한다. 소스 디바이스 (102) 의 비디오 소스 (104) 는 비디오 카메라와 같은 비디오 캡처 디바이스, 이전에 캡처된 원시 비디오를 포함하는 비디오 아카이브, 및/또는 비디오 컨텐츠 제공자로부터 비디오를 수신하기 위한 비디오 피드 인터페이스를 포함할 수도 있다. 추가의 대안으로서, 비디오 소스 (104) 는 컴퓨터 그래픽 기반 데이터를 소스 비디오로서, 또는 라이브 비디오, 아카이브된 비디오, 및 컴퓨터 생성된 비디오의 조합으로서 생성할 수도 있다. 각각의 경우에 있어서, 비디오 인코더 (200) 는 캡처된, 사전 캡처된, 또는 컴퓨터 생성된 비디오 데이터를 인코딩한다. 비디오 인코더 (200) 는 픽처들을 수신된 순서 (때때로 "디스플레이 순서" 로 지칭됨) 로부터 코딩을 위한 코딩 순서로 재배열할 수도 있다. 비디오 인코더 (200) 는 인코딩된 비디오 데이터를 포함하는 비트스트림을 생성할 수도 있다. 그 후, 소스 디바이스 (102) 는 예를 들어, 목적지 디바이스 (116) 의 입력 인터페이스 (122) 에 의한 수신 및/또는 취출을 위해 인코딩된 비디오 데이터를 출력 인터페이스 (108) 를 통해 컴퓨터 판독가능 매체 (110) 상으로 출력할 수도 있다.

[0024] 소스 디바이스 (102) 의 메모리 (106) 및 목적지 디바이스 (116) 의 메모리 (120) 는 범용 메모리들을 나타낸다. 일부 예들에서, 메모리들 (106, 120) 은 원시 비디오 데이터, 예를 들어, 비디오 소스 (104) 로부터의 원시 비디오 및 비디오 디코더 (300) 로부터의 원시, 디코딩된 비디오 데이터를 저장할 수도 있다. 추가적으로 또는 대안적으로, 메모리들 (106, 120) 은 예를 들어, 각각 비디오 인코더 (200) 및 비디오 디코더 (300) 에 의해 실행가능한 소프트웨어 명령들을 저장할 수도 있다. 이 예에서는 비디오 인코더 (200) 및 비디오 디코더 (300) 와 별도로 도시되어 있지만, 비디오 인코더 (200) 및 비디오 디코더 (300) 는 또한 기능적으로 유사하거나 또는 동등한 목적들을 위한 내부 메모리들을 포함할 수도 있음을 이해해야 한다. 또한, 메모리들 (106, 120) 은 예를 들어, 비디오 인코더 (200) 로부터 출력되고 비디오 디코더 (300) 에 입력되는 인코딩된 비디오 데이터를 저장할 수도 있다. 일부 예들에서, 메모리들 (106, 120) 의 부분들은 예를 들어, 원시, 디코딩된, 및/또는 인코딩된 비디오 데이터를 저장하기 위해 하나 이상의 비디오 버퍼들로서 할당될 수도 있다.

[0025] 컴퓨터 판독가능 매체 (110) 는 인코딩된 비디오 데이터를 소스 디바이스 (102) 로부터 목적지 디바이스 (116) 로 전송할 수도 있는 임의의 타입의 매체 또는 디바이스를 나타낼 수도 있다. 하나의 예에서, 컴퓨터 판독가능 매체 (110) 는, 소스 디바이스 (102) 로 하여금, 실시간으로, 예를 들어, 무선 주파수 네트워크 또는 컴퓨터 기반 네트워크를 통해 직접 목적지 디바이스 (116) 로 인코딩된 비디오 데이터를 송신할 수도 있게 하기 위한 통신 매체를 나타낸다. 무선 통신 프로토콜과 같은 통신 표준에 따라, 출력 인터페이스 (108) 는 인코딩된 비디오 데이터를 포함하는 송신 신호를 변조할 수도 있고, 입력 인터페이스 (122) 는 수신된 송신 신호를 복조할 수도 있다. 통신 매체는 임의의 무선 또는 유선 통신 매체, 이를 테면 라디오 주파수 (radio frequency; RF) 스펙트럼 또는 하나 이상의 물리적 송신 라인들을 포함할 수도 있다. 통신 매체는 로컬 영역 네트워크, 광역 네트워크, 또는 인터넷과 같은 글로벌 네트워크와 같은 패킷 기반 네트워크의 부분을 형성할 수도 있다. 통신 매체는 라우터들, 스위치들, 기지국들, 또는 소스 디바이스 (102) 로부터 목적지 디바이스 (116) 로의 통신을 가능하게 하는 데 유용할 수도 있는 임의의 다른 장비를 포함할 수도 있다.

[0026] 일부 예들에서, 소스 디바이스 (102) 는 출력 인터페이스 (108) 로부터 저장 디바이스 (112) 로 인코딩된 데이터를 출력할 수도 있다. 유사하게, 목적지 디바이스 (116) 는 입력 인터페이스 (122) 를 통해 저장 디바이스 (112) 로부터의 인코딩된 데이터에 액세스할 수도 있다. 저장 디바이스 (112) 는 하드 드라이브, 블루레이 디스크들, DVD들, CD-ROM들, 플래시 메모리, 휘발성 또는 비휘발성 메모리, 또는 인코딩된 비디오 데이터를 저장하기 위한 임의의 다른 적합한 디지털 저장 매체와 같은 다양한 분산된 또는 로컬 액세스된 데이터 저장 매체 중 임의의 것을 포함할 수도 있다.

[0027] 일부 예들에서, 소스 디바이스 (102) 는 소스 디바이스 (102) 에 의해 생성된 인코딩된 비디오를 저장할 수도 있는 파일 서버 (114) 또는 다른 중간 저장 디바이스에 인코딩된 비디오 데이터를 출력할 수도 있다. 목적지 디바이스 (116) 는 스트리밍 또는 다운로드를 통해 파일 서버 (114) 로부터 저장된 비디오 데이터에 액세스할 수도 있다. 파일 서버 (114) 는 인코딩된 비디오 데이터를 저장하고 그 인코딩된 비디오 데이터를 목적지 디바이스 (116) 에 송신할 수도 있는 임의의 타입의 서버 디바이스일 수도 있다. 파일 서버 (114) 는 (예를 들어, 웹 사이트를 위한) 웹 서버, 파일 전송 프로토콜 (FTP) 서버, 콘텐츠 전달 네트워크 디바이스, 또는 NAS (network attached storage) 디바이스를 나타낼 수도 있다. 목적지 디바이스 (116) 는 인터넷 접속을 포함한, 임의의 표준 데이터 접속을 통해 파일 서버 (114) 로부터 인코딩된 비디오 데이터에 액세스할 수도 있다. 이것은 파일 서버 (114) 상에 저장된 인코딩된 비디오 데이터에 액세스하기에 적합한, 무선 채널 (예를 들어, Wi-Fi 접속), 유선 접속 (예를 들어, 디지털 가입자 라인 (DSL), 케이블 모뎀 등), 또는 양자의 조합을 포함할 수도 있다. 파일 서버 (114) 및 입력 인터페이스 (122) 는 스트리밍 송신 프로토콜, 다운로드 송

신 프로토콜, 또는 이들의 조합에 따라 동작하도록 구성될 수도 있다.

[0028] 출력 인터페이스 (108) 및 입력 인터페이스 (122) 는 무선 송신기/수신기, 모뎀들, 유선 네트워킹 컴포넌트들 (예를 들어, 이더넷 카드들), 다양한 IEEE 802.11 표준들 중 임의의 것에 따라 동작하는 무선 통신 컴포넌트들, 또는 다른 물리적 컴포넌트들을 나타낼 수도 있다. 출력 인터페이스 (108) 및 입력 인터페이스 (122) 가 무선 컴포넌트들을 포함하는 예들에 있어서, 출력 인터페이스 (108) 및 입력 인터페이스 (122) 는 4G, 4G-LTE (Long-Term Evolution), LTE 어드밴스드, 5G 등과 같은 셀룰러 통신 표준에 따라, 인코딩된 비디오 데이터와 같은 데이터를 전송하도록 구성될 수도 있다. 출력 인터페이스 (108) 가 무선 송신기를 포함하는 일부 예들에 있어서, 출력 인터페이스 (108) 및 입력 인터페이스 (122) 는 IEEE 802.11 사양, IEEE 802.15 사양 (예를 들어, ZigBee™), Bluetooth™ 표준 등과 같은 다른 무선 표준들에 따라, 인코딩된 비디오 데이터와 같은 데이터를 전송하도록 구성될 수도 있다. 일부 예들에서, 소스 디바이스 (102) 및/또는 목적지 디바이스 (116) 는 개개의 시스템-온-칩 (SoC) 디바이스들을 포함할 수도 있다. 예를 들어, 소스 디바이스 (102) 는 비디오 인코더 (200) 및/또는 출력 인터페이스 (108) 에 기인하는 기능성을 수행하기 위한 SoC 디바이스를 포함할 수도 있고, 목적지 디바이스 (116) 는 비디오 디코더 (300) 및/또는 입력 인터페이스 (122) 에 기인하는 기능성을 수행하기 위한 SoC 디바이스를 포함할 수도 있다.

[0029] 본 개시의 기법들은 오버-디-에어 (over-the-air) 텔레비전 브로드캐스트들, 케이블 텔레비전 송신들, 위성 텔레비전 송신들, 인터넷 스트리밍 비디오 송신들, 이를 테면 DASH (dynamic adaptive streaming over HTTP), 데이터 저장 매체 상으로 인코딩되는 디지털 비디오, 데이터 저장 매체 상에 저장된 디지털 비디오의 디코딩, 또는 다른 애플리케이션들과 같은 다양한 멀티미디어 애플리케이션들 중 임의의 것을 지원하여 비디오 코딩에 적용될 수도 있다.

[0030] 목적지 디바이스 (116) 의 입력 인터페이스 (122) 는 컴퓨터 판독가능 매체 (110) (예를 들어, 저장 디바이스 (112), 파일 서버 (114) 등) 로부터 인코딩된 비디오 비트스트림을 수신한다. 인코딩된 비디오 비트스트림은 비디오 블록들 또는 다른 코딩된 유닛들 (예를 들어, 슬라이스들, 픽처들, 픽처들의 그룹들, 시퀀스들 등) 의 프로세싱 및/또는 특성들을 기술하는 값들을 갖는 신택스 엘리먼트들과 같은, 비디오 디코더 (300) 에 의해 또한 사용되는 비디오 인코더 (200) 에 의해 정의된 시그널링 정보를 포함할 수도 있다. 디스플레이 디바이스 (118) 는 디코딩된 비디오 데이터의 디코딩된 픽처들을 사용자에게 디스플레이한다. 디스플레이 디바이스 (118) 는 음극선관 (CRT), 액정 디스플레이 (LCD), 플라즈마 디스플레이, 유기 발광 다이오드 (OLED) 디스플레이, 또는 다른 타입의 디스플레이 디바이스와 같은 다양한 디스플레이 디바이스들 중 임의의 디스플레이 디바이스를 나타낼 수도 있다.

[0031] 도 1 에 도시되지는 않았지만, 일부 예들에서, 비디오 인코더 (200) 및 비디오 디코더 (300) 는 각각 오디오 인코더 및/또는 오디오 디코더와 통합될 수도 있고, 공통 데이터 스트림에서 오디오 및 비디오 양자 모두를 포함하는 멀티플렉싱된 스트림들을 핸들링하기 위해, 적절한 MUX-DEMUX 유닛들, 또는 다른 하드웨어 및/또는 소프트웨어를 포함할 수도 있다. 적용가능한 경우, MUX-DEMUX 유닛들은 ITU H.223 멀티플렉서 프로토콜, 또는 다른 프로토콜들, 이를 테면 사용자 데이터그램 프로토콜 (UDP) 에 따를 수도 있다.

[0032] 비디오 인코더 (200) 및 비디오 디코더 (300) 각각은 다양한 적합한 인코더 및/또는 디코더 회로부, 이를 테면 하나 이상의 마이크로프로세서들, 디지털 신호 프로세서들 (DSP들), 주문형 집적 회로들 (ASIC들), 필드 프로그래밍가능 게이트 어레이들 (FPGA들), 이산 로직, 소프트웨어, 하드웨어, 펌웨어 또는 이들의 임의의 조합들 중 임의의 것으로서 구현될 수도 있다. 기법들이 부분적으로 소프트웨어에서 구현되는 경우, 디바이스는 적합한 비일시적 컴퓨터 판독가능 매체에 소프트웨어에 대한 명령들을 저장하고, 본 개시의 기법들을 수행하기 위해 하나 이상의 프로세서들을 사용하는 하드웨어에서 그 명령들을 실행할 수도 있다. 비디오 인코더 (200) 및 비디오 디코더 (300) 의 각각은 하나 이상의 인코더들 또는 디코더들에 포함될 수도 있는데, 이들 중 어느 하나는 각각의 디바이스에서 결합된 인코더/디코더 (CODEC) 의 부분으로서 통합될 수도 있다. 비디오 인코더 (200) 및/또는 비디오 디코더 (300) 를 포함하는 디바이스는 집적 회로, 마이크로프로세서, 및/또는 무선 통신 디바이스, 예컨대 셀룰러 전화기를 포함할 수도 있다.

[0033] 비디오 인코더 (22) 및 비디오 디코더 (300) 는 고효율 비디오 코딩 (HEVC) 으로도 지칭되는 ITU-T H.265 와 같은 비디오 코딩 표준 또는 멀티-뷰 및/또는 스케일러블 비디오 코딩 확장들과 같은 그의 확장들에 따라 동작할 수도 있다. 대안적으로, 비디오 인코더 (200) 및 비디오 디코더 (300) 는 JEM (Joint Exploration Test Model) 또는 VVC (Versatile Video Coding) 로도 또한 지칭되는 ITU-T H.266 과 같은 다른 독점 또는 산업 표준들에 따라 동작할 수도 있다. VVC 표준의 최신 초안은 Bross 등의, "Versatile Video Coding (Draft

6)," Joint Video Experts Team (JVET) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11, 15th Meeting: Gothenburg, SE, 3-12 July 2019, JVET-O2001-vE (이하 "VVC 드래프트 6" 라 함) 에서 설명되어 있다. 하지만, 본 개시의 기법들은 임의의 특정 코딩 표준에 제한되지 않는다.

[0034] 일반적으로, 비디오 인코더 (200) 및 비디오 디코더 (300) 는 픽처들의 블록 기반 코딩을 수행할 수도 있다. 용어 "블록" 은 일반적으로 프로세싱될 (예를 들어, 인코딩될, 디코딩될, 또는 다르게는 인코딩 및/또는 디코딩 프로세스에서 사용될) 데이터를 포함하는 구조를 지칭한다. 예를 들어, 블록은 루미넌스 및/또는 크로미넌스 데이터의 샘플들의 2 차원 매트릭스를 포함할 수도 있다. 일반적으로, 비디오 인코더 (200) 및 비디오 디코더 (300) 는 YUV (예를 들어, Y, Cb, Cr) 포맷으로 표현된 비디오 데이터를 코딩할 수도 있다. 즉, 픽처의 샘플들에 대한 적색, 녹색, 및 청색 (RGB) 데이터를 코딩하는 것보다는, 비디오 인코더 (200) 및 비디오 디코더 (300) 는 루미넌스 및 크로미넌스 컴포넌트들을 코딩할 수도 있고, 여기서 크로미넌스 컴포넌트들은 적색 색조 및 청색 색조 크로미넌스 컴포넌트들 양자 모두를 포함할 수도 있다. 일부 예들에서, 비디오 인코더 (200) 는 인코딩 이전에 수신된 RGB 포맷팅된 데이터를 YUV 표현으로 변환하고, 비디오 디코더 (300) 는 YUV 표현을 RGB 포맷으로 변환한다. 대안적으로는, 사전 및 사후 프로세싱 유닛들 (도시되지 않음) 이 이들 변환들을 수행할 수도 있다.

[0035] 본 개시는 일반적으로 픽처의 데이터를 인코딩 또는 디코딩하는 프로세스를 포함하는 픽처들의 코딩 (예를 들어, 인코딩 및 디코딩) 을 언급할 수도 있다. 유사하게, 본 개시는, 예를 들어, 예측 및/또는 잔차 코딩과 같은, 블록들에 대한 데이터를 인코딩 또는 디코딩하는 프로세스를 포함하는 픽처의 블록들의 코딩을 언급할 수도 있다. 인코딩된 비디오 비트스트림은 일반적으로 코딩 결정들 (예를 들어, 코딩 모드들) 및 픽처들의 블록들로의 파티셔닝을 나타내는 신택스 엘리먼트들에 대한 일련의 값들을 포함한다. 따라서, 픽처 또는 블록을 코딩하는 것에 대한 참조들은 일반적으로 픽처 또는 블록을 형성하는 신택스 엘리먼트들에 대한 코딩 값들로서 이해되어야 한다.

[0036] HEVC 는 코딩 유닛들 (CU들), 예측 유닛들 (PU들), 및 변환 유닛들 (TU들) 을 포함한 다양한 블록들을 정의한다. HEVC 에 따르면, 비디오 코더 (이를 테면 비디오 인코더 (200)) 는 쿼드트리 구조에 따라 코딩 트리 유닛 (CTU) 을 CU들로 파티셔닝한다. 즉, 비디오 코더는 CTU들 및 CU들을 4 개의 동일한 비오버랩하는 정사각형들로 파티셔닝하고, 쿼드트리의 각각의 노드는 0 개 또는 4 개 중 어느 하나의 자식 노드들을 갖는다. 자식 노드들 없는 노드들은 "리프 노드들" 로 지칭될 수도 있고, 그러한 리프 노드들의 CU들은 하나 이상의 PU들 및/또는 하나 이상의 TU들을 포함할 수도 있다. 비디오 코더는 PU들 및 TU들을 추가로 파티셔닝할 수도 있다. 예를 들어, HEVC 에서, 잔차 쿼드트리 (RQT) 는 TU들의 파티셔닝을 나타낸다. HEVC 에서, PU들은 인터 예측 데이터를 나타내는 한편, TU들은 잔차 데이터를 나타낸다. 인트라 예측되는 CU들은 인트라 모드 표시와 같은 인트라 예측 정보를 포함한다.

[0037] 다른 예로서, 비디오 인코더 (200) 및 비디오 디코더 (300) 는 JEM 또는 VVC 에 따라 동작하도록 구성될 수도 있다. JEM 또는 VVC 에 따르면, 비디오 코더 (이를 테면 비디오 인코더 (200)) 는 픽처를 복수의 코딩 트리 유닛들 (CTU들) 로 파티셔닝한다. 비디오 인코더 (200) 는 쿼드트리 바이너리 트리 (QTBT) 구조 또는 멀티-타입 트리 (MTT) 구조와 같은 트리 구조에 따라 CTU 를 파티셔닝할 수도 있다. QTBT 구조는 HEVC 의 CU들, PU들, 및 TU들 간의 분리와 같은 다중 파티션 타입들의 개념들을 제거한다. QTBT 구조는 2 개의 레벨들: 쿼드트리 파티셔닝에 따라 파티셔닝된 제 1 레벨, 및 바이너리 트리 파티셔닝에 따라 파티셔닝된 제 2 레벨을 포함한다. QTBT 구조의 루트 노드는 CTU 에 대응한다. 바이너리 트리들의 리프 노드들은 코딩 유닛들 (CU들) 에 대응한다.

[0038] MTT 파티셔닝 구조에서, 블록들은 쿼드트리 (QT) 파티션, 바이너리 트리 (BT) 파티션, 및 하나 이상의 타입들의 트리플 트리 (TT) 파티션들을 사용하여 파티셔닝될 수도 있다. 트리플 트리 파티션은 블록이 3 개의 서브-블록들로 스플릿팅되는 파티션이다. 일부 예들에서, 트리플 트리 파티션은 센터를 통해 원래의 블록을 분할하지 않고 블록을 3 개의 서브-블록들로 분할한다. MTT (예를 들어, QT, BT, 및 TT) 에서의 파티셔닝 타입들은 대칭적 또는 비대칭적일 수도 있다.

[0039] 일부 예들에서, 비디오 인코더 (200) 및 비디오 디코더 (300) 는 루미넌스 및 크로미넌스 컴포넌트들의 각각을 나타내기 위해 단일 QTBT 또는 MTT 구조를 사용할 수도 있는 한편, 다른 예들에서, 비디오 인코더 (200) 및 비디오 디코더 (300) 는 2 개 이상의 QTBT 또는 MTT 구조들, 이를 테면 루미넌스 컴포넌트를 위한 하나의 QTBT/MTT 구조 및 양자의 크로미넌스 컴포넌트들을 위한 다른 QTBT/MTT 구조 (또는 각각의 크로미넌스 컴포넌트들을 위한 2 개의 QTBT/MTT 구조들) 를 사용할 수도 있다.

[0040] 비디오 인코더 (200) 및 비디오 디코더 (300) 는 HEVC 당 쿼드트리 파티셔닝, QTBT 파티셔닝, MTT 파티셔닝, 또는 다른 파티셔닝 구조들을 사용하도록 구성될 수도 있다. 설명의 목적들을 위해, 본 개시의 기법들의 설명은 QTBT 파티셔닝에 대하여 제시된다. 그러나, 본 개시의 기법들은 또한, 쿼드트리 파티셔닝, 또는 다른 타입들의 파티셔닝에도 사용하도록 구성된 비디오 코더들에 적용될 수도 있음이 이해되어야 한다.

[0041] 블록들 (예를 들어, CTU들 또는 CU들) 은 픽처에서 다양한 방식으로 그룹화될 수도 있다. 하나의 예로서, 브릭 (brick) 은 픽처에서의 특정 타일 내의 CTU 행들의 직사각형 영역을 지칭할 수도 있다. 타일은 픽처에서의 특정 타일 행 (row) 및 특정 타일 열 (column) 내의 CTU들의 직사각형 영역일 수도 있다. 타일 열은 (예를 들어, 픽처 파라미터 세트에서와 같은) 신택스 엘리먼트들에 의해 특정된 폭 및 픽처의 높이와 동일한 높이를 갖는 CTU들의 직사각형 영역을 지칭한다. 타일 행은, 픽처의 폭과 동일한 폭 및 (예를 들어, 픽처 파라미터 세트에서와 같은) 신택스 엘리먼트들에 의해 특정된 높이를 갖는 CTU들의 직사각형 영역을 지칭한다.

[0042] 일부 예들에서, 타일은 다중 브릭들로 파티셔닝될 수도 있고, 그 브릭들의 각각은 타일 내의 하나 이상의 CTU 행들을 포함할 수도 있다. 다중 브릭들로 파티셔닝되지 않는 타일은 또한 브릭으로 지칭될 수도 있다. 하지만, 타일의 진정한 서브세트인 브릭은 타일로서 지칭되지 않을 수도 있다.

[0043] 픽처에서의 브릭들은 또한 슬라이스로 배열될 수도 있다. 슬라이스는 단일 네트워크 추상화 계층 (NAL) 유닛에 배타적으로 포함될 수도 있는 픽처의 정수 개의 브릭들일 수도 있다. 일부 예들에서, 슬라이스는 다수의 완전한 타일들 또는 오직 하나의 타일의 연속적인 시퀀스의 완전한 브릭들 중 어느 하나를 포함한다.

[0044] 본 개시는 수직 및 수평 차원들의 관점에서 블록 (예컨대 CU 또는 다른 비디오 블록) 의 샘플 차원들을 지칭하기 위해 상호교환가능하게 "NxN" 및 "N 바이 N", 예를 들어, 16x16 샘플들 또는 16 바이 16 샘플들을 사용할 수도 있다. 일반적으로, 16x16 CU 는 수직 방향에서 16 샘플들 (y = 16) 그리고 수평 방향에서 16 샘플들 (x = 16) 을 가질 것이다. 마찬가지로, NxN CU 는 일반적으로 수직 방향에서 N 샘플들 및 수평 방향에서 N 샘플들을 갖고, 여기서 N 은 음이 아닌 정수 값을 나타낸다. CU 에서의 샘플들은 행들 (rows) 및 열들 (columns) 로 배열될 수도 있다. 더욱이, CU들은 수직 방향에서와 동일한 수의 샘플들을 수평 방향에서 반드시 가질 필요는 없다. 예를 들면, CU들은 NxM 샘플들을 포함할 수도 있고, 여기서 M 은 N 과 반드시 동일한 것은 아니다.

[0045] 비디오 인코더 (200) 는 예측 및/또는 잔차 정보를 나타내는 CU들에 대한 비디오 데이터, 및 다른 정보를 인코딩한다. 예측 정보는 CU 에 대한 예측 블록을 형성하기 위하여 CU 가 어떻게 예측될지를 표시한다. 잔차 정보는 일반적으로 인코딩 이전의 CU 의 샘플들과 예측 블록 사이의 샘플 별 (sample-by-sample) 차이들을 나타낸다.

[0046] CU 를 예측하기 위해, 비디오 인코더 (200) 는 일반적으로 인터 예측 또는 인트라 예측을 통해 CU 에 대한 예측 블록을 형성할 수도 있다. 인터 예측은 일반적으로 이전에 코딩된 픽처의 데이터로부터 CU 를 예측하는 것을 지칭하는 반면, 인트라 예측은 일반적으로 동일한 픽처의 이전에 코딩된 데이터로부터 CU 를 예측하는 것을 지칭한다. 인터 예측을 수행하기 위해, 비디오 인코더 (200) 는 하나 이상의 모션 벡터들을 사용하여 예측 블록을 생성할 수도 있다. 비디오 인코더 (200) 는 일반적으로 CU 와 레퍼런스 블록 사이의 차이들의 관점에서, CU 에 밀접하게 매칭하는 레퍼런스 블록을 식별하기 위해 모션 탐색을 수행할 수도 있다. 비디오 인코더 (200) 는 절대 차이의 합 (sum of absolute difference; SAD), 제곱 차이들의 합 (sum of squared differences; SSD), 평균 절대 차이 (mean absolute difference; MAD), 평균 제곱 차이들 (mean squared differences; MSD), 또는 레퍼런스 블록이 현재 CU 에 밀접하게 매칭하는지 여부를 결정하기 위한 다른 그러한 차이 계산들을 사용하여 차이 메트릭을 계산할 수도 있다. 일부 예들에서, 비디오 인코더 (200) 는 단방향 예측 또는 양방향 예측을 사용하여 현재 CU 를 예측할 수도 있다.

[0047] JEM 및 VVC 의 일부 예들은 또한, 인터 예측 모드로 고려될 수도 있는 아핀 모션 보상 모드 (affine motion compensation mode) 를 제공한다. 아핀 모션 보상 모드에서, 비디오 인코더 (200) 는 줌 인 또는 아웃, 회전, 원근 모션 (perspective motion), 또는 다른 불규칙한 모션 타입들과 같은 비-병진 모션을 나타내는 2 개 이상의 모션 벡터들을 결정할 수도 있다.

[0048] 인트라 예측을 수행하기 위해, 비디오 인코더 (200) 는 예측 블록을 생성하기 위해 인트라 예측 모드를 선택할 수도 있다. JEM 및 VVC 의 일부 예들은 다양한 방향 모드들 뿐만 아니라 평면 모드 및 DC 모드를 포함하여 67 개의 인트라 예측 모드들을 제공한다. 일반적으로, 비디오 인코더 (200) 는 현재 블록의 샘플들을 예측할 현재 블록 (예를 들어, CU 의 블록) 에 대한 이웃하는 샘플들을 기술하는 인트라 예측 모드를 선택한다.

그러한 샘플들은 일반적으로, 비디오 인코더 (200) 가 래스터 스캔 순서로 (좌측에서 우측으로, 상부에서 하부로) CTU들 및 CU들을 코딩하는 것을 가정하여, 현재 블록과 동일한 픽처에서 현재 블록의 상부, 상부 및 좌측에, 또는 좌측에 있을 수도 있다.

[0049] 비디오 인코더 (200) 는 현재 블록에 대한 예측 모드를 나타내는 데이터를 인코딩한다. 예를 들어, 인터 예측 모드들에 대해, 비디오 인코더 (200) 는 다양한 이용가능한 인터 예측 모드들 중 어느 것이 사용되는지를 나타내는 데이터 뿐만 아니라 대응하는 모드에 대한 모션 정보를 인코딩할 수도 있다. 단방향 또는 양방향 인터 예측을 위해, 예를 들어, 비디오 인코더 (200) 는 어드밴스드 모션 벡터 예측 (AMVP) 또는 병합 모드(merge mode)를 사용하여 모션 벡터들을 인코딩할 수도 있다. 비디오 인코더 (200) 는 유사한 모드들을 사용하여 아핀 모션 보상 모드에 대한 모션 벡터들을 인코딩할 수도 있다.

[0050] 블록의 인트라 예측 또는 인터 예측과 같은 예측에 이어, 비디오 인코더 (200) 는 블록에 대한 잔차 데이터를 계산할 수도 있다. 잔차 블록과 같은 잔차 데이터는 대응하는 예측 모드를 사용하여 형성되는, 블록과 블록에 대한 예측 블록 사이의 샘플 별 차이들을 나타낸다. 비디오 인코더 (200) 는 샘플 도메인 대신에 변환 도메인에서 변환된 데이터를 생성하기 위해, 잔차 블록에 하나 이상의 변환들을 적용할 수도 있다. 예를 들어, 비디오 인코더 (200) 는 이산 코사인 (DCT), 정수 변환, 웨이브릿 변환, 또는 개념적으로 유사한 변환을 잔차 비디오 데이터에 적용할 수도 있다. 추가적으로, 비디오 인코더 (200) 는 MDNSST (mode-dependent non-separable secondary transform), 신호 의존적 변환, 카루넨-루베 변환 (Karhunen-Loeve transform; KLT) 등과 같은 제 1 변환에 후속하는 세컨더리 변환을 적용할 수도 있다. 비디오 인코더 (200) 는 하나 이상의 변환들의 적용에 이어 변환 계수들을 생성한다.

[0051] 상기 언급된 바와 같이, 변환 계수들을 생성하기 위한 임의의 변환들에 이어, 비디오 인코더 (200) 는 변환 계수들의 양자화를 수행할 수도 있다. 양자화는 일반적으로, 변환 계수들이 그 계수들을 나타내는데 사용되는 데이터의 양을 가능하게는 감소시키도록 양자화되어, 추가 압축을 제공하는 프로세스를 지칭한다. 양자화 프로세스를 수행함으로써, 비디오 인코더 (200) 는 계수들의 일부 또는 전부와 연관된 비트 심도(bit depth)를 감소시킬 수도 있다. 예를 들어, 비디오 인코더 (200) 는 양자화 동안 n 비트 값을 m 비트 값으로 라운딩 다운할 수도 있고, 여기서 n 은 m 보다 크다. 일부 예들에서, 양자화를 수행하기 위해, 비디오 인코더 (200) 는 양자화될 값의 비트단위 우측 시프트를 수행할 수도 있다.

[0052] 양자화에 이어, 비디오 인코더 (200) 는 변환 계수들을 스캔하여, 양자화된 변환 계수들을 포함한 2 차원 매트릭스로부터 1 차원 벡터를 생성할 수도 있다. 스캔은 벡터의 전방에 더 높은 에너지 (및 따라서 더 낮은 주파수) 계수들을 배치하고 벡터의 후방에 더 낮은 에너지 (및 따라서 더 높은 주파수) 변환 계수들을 배치하도록 설계될 수도 있다. 일부 예들에서, 비디오 인코더 (200) 는 양자화된 변환 계수들을 스캔하기 위해 미리정의된 스캔 순서를 활용하여 직렬화된 벡터를 생성한 후, 벡터의 양자화된 변환 계수들을 엔트로피 인코딩할 수도 있다. 다른 예들에서, 비디오 인코더 (200) 는 적응적 스캔을 수행할 수도 있다. 1 차원 벡터를 형성하기 위해 양자화된 변환 계수들을 스캔한 후, 비디오 인코더 (200) 는, 예를 들어, 컨텍스트 적응 이진 산술 코딩 (CABAC) 에 따라, 1 차원 벡터를 엔트로피 인코딩할 수도 있다. 비디오 인코더 (200) 는 또한, 비디오 데이터를 디코딩하는데 있어서 비디오 디코더 (300) 에 의한 사용을 위해 인코딩된 비디오 데이터와 연관된 메타데이터를 기술하는 신택스 엘리먼트들에 대한 값들을 엔트로피 인코딩할 수도 있다.

[0053] CABAC 을 수행하기 위해, 비디오 인코더 (200) 는 송신될 심볼에 컨텍스트 모델 내의 컨텍스트를 할당할 수도 있다. 컨텍스트 (context) 는 예를 들어, 심볼의 이웃하는 값들이 제로 값인지 여부와 관련될 수도 있다. 확률 결정은 심볼에 할당된 컨텍스트에 기초할 수도 있다.

[0054] 비디오 인코더 (200) 는 신택스 데이터, 예컨대 블록-기반 신택스 데이터, 픽처-기반 신택스 데이터, 및 시퀀스-기반 신택스 데이터를, 비디오 디코더 (300) 에, 예를 들어, 픽처 헤더, 블록 헤더, 슬라이스 헤더, 또는 다른 신택스 데이터, 예컨대 시퀀스 파라미터 세트 (SPS), 픽처 파라미터 세트 (PPS), 또는 비디오 파라미터 세트 (VPS) 에서 추가로 생성할 수도 있다. 비디오 디코더 (300) 는 마찬가지로 대응하는 비디오 데이터를 디코딩하는 방법을 결정하기 위해 그러한 신택스 데이터를 디코딩할 수도 있다.

[0055] 이러한 방식으로, 비디오 인코더 (200) 는 인코딩된 비디오 데이터, 예를 들어, 픽처의 블록들 (예를 들어, CU들) 로의 파티셔닝을 기술하는 신택스 엘리먼트들 및 블록들에 대한 예측 및/또는 잔차 정보를 포함하는 비트스트림을 생성할 수도 있다. 궁극적으로, 비디오 디코더 (300) 는 비트스트림을 수신하고 인코딩된 비디오 데이터를 디코딩할 수도 있다.

[0056] 일반적으로, 비디오 디코더 (300) 는 비트스트림의 인코딩된 비디오 데이터를 디코딩하기 위해 비디오 인코더 (200) 에 의해 수행되는 것과 상반되는 프로세스를 수행한다. 예를 들어, 비디오 디코더 (300) 는 비디오 인코더 (200) 의 CABAC 인코딩 프로세스와 실질적으로 유사하지만, 상반되는 방식으로 CABAC 을 사용하여 비트스트림의 신택스 엘리먼트들에 대한 값들을 디코딩할 수도 있다. 신택스 엘리먼트들은 픽처의 CTU들로의 파티셔닝 정보, 및 QTBT 구조와 같은 대응하는 파티션 구조에 따른 각각의 CTU 의 파티셔닝을 정의하여, CTU 의 CU들을 정의할 수도 있다. 신택스 엘리먼트들은 비디오 데이터의 블록들 (예를 들어, CU들) 에 대한 예측 및 잔차 정보를 추가로 정의할 수도 있다.

[0057] 잔차 정보는 예를 들어 양자화된 변환 계수들에 의해 표현될 수도 있다. 비디오 디코더 (300) 는 블록에 대한 잔차 블록을 재생하기 위해 블록의 양자화된 변환 계수들을 역 양자화 및 역 변환할 수도 있다. 비디오 디코더 (300) 는 시그널링된 예측 모드 (인트라 또는 인터 예측) 및 관련된 예측 정보 (예를 들어, 인터 예측을 위한 모션 정보) 를 사용하여 블록에 대한 예측 블록을 형성한다. 비디오 디코더 (300) 는 그 후 예측 블록과 잔차 블록을 (샘플 별 기준으로) 조합하여 원래의 블록을 재생할 수도 있다. 비디오 디코더 (300) 는 블록의 경계들을 따라 시각적 아티팩트들을 감소시키기 위해 디블로킹 프로세스를 수행하는 것과 같은 추가적인 프로세싱을 수행할 수도 있다.

[0058] 위에서 논의된 바와 같이 그리고 본 개시의 하나 이상의 기법들에 따라, 비디오 인코더 (200) 및/또는 비디오 디코더 (300) 는 제로 또는 2 의 거듭제곱들로 제한된 절대값들로 CC-ALF에 대한 필터 계수들을 시그널링하도록 구성될 수도 있다. 이러한 방식으로, 비디오 인코더 (200) 및/또는 비디오 디코더 (300) 는 CC-ALF 의 퍼포먼스에서의 곱셈 연산들을 비트-시프트 연산들로 대체할 수도 있으며, 비트-시프트 연산들은 덜 리소스 집약적이다.

[0059] 본 개시는 일반적으로 신택스 엘리먼트들과 같은, 소정의 정보를 "시그널링(signaling)" 하는 것을 언급할 수도 있다. 용어 “시그널링” 은 일반적으로 인코딩된 비디오 데이터를 디코딩하는데 사용된 신택스 엘리먼트들 및/또는 다른 데이터에 대한 값들의 통신을 지칭할 수도 있다. 즉, 비디오 인코더 (200) 는 비트스트림에서 신택스 엘리먼트들에 대한 값들을 시그널링할 수도 있다. 일반적으로, 시그널링은 비트스트림에서 값을 생성하는 것을 지칭한다. 상기 언급된 바와 같이, 소스 디바이스 (102) 는 목적지 디바이스 (116) 에 의한 추후 취출을 위해 저장 디바이스 (112) 에 신택스 엘리먼트들을 저장할 때 발생할 수도 있는 바와 같은, 비실시간으로, 또는 실질적으로 실시간으로 비트스트림을 목적지 디바이스 (116) 로 전송할 수도 있다.

[0060] 도 2a 및 도 2b 는 예시적인 쿼드트리 바이너리 트리 (QTBT) 구조 (130), 및 대응하는 코딩 트리 유닛 (CTU) (132) 을 나타내는 개념도이다. 실선들은 쿼드트리 스플릿팅을 나타내고, 점선들은 바이너리 트리 스플릿팅을 나타낸다. 바이너리 트리의 각각의 스플릿팅된 (즉, 비-리프) 노드에서, 어느 스플릿팅 타입 (즉, 수평 또는 수직) 이 사용되는지를 나타내기 위해 하나의 플래그가 시그널링되며, 여기서 0 은 수평 스플릿팅을 표시하고 1 은 수직 스플릿팅을 표시한다. 쿼드트리 스플릿팅에 대해, 스플릿팅 타입을 표시할 필요는 없는데, 이는 쿼드트리 노드들이 동일한 사이즈를 가진 4 개의 서브-블록들로 수평으로 및 수직으로 블록을 스플릿팅하기 때문이다. 이에 따라, QTBT 구조 (130) 의 영역 트리 레벨 (즉, 실선들) 에 대한 신택스 엘리먼트들 (이를 테면 스플릿팅 정보) 및 QTBT 구조 (130) 의 예측 트리 레벨 (즉, 점선들) 에 대한 신택스 엘리먼트들 (이를 테면 스플릿팅 정보) 을, 비디오 인코더 (200) 가 인코딩할 수도 있고, 비디오 디코더 (300) 가 디코딩할 수도 있다. QTBT 구조 (130) 의 종단 리프 노드들에 의해 표현된 CU들에 대해, 예측 및 변환 데이터와 같은 비디오 데이터를, 비디오 인코더 (200) 가 인코딩할 수도 있고, 비디오 디코더 (300) 가 디코딩할 수도 있다.

[0061] 일반적으로, 도 2b 의 CTU (132) 는 제 1 및 제 2 레벨들에서 QTBT 구조 (130) 의 노드들에 대응하는 블록들의 사이즈들을 정의하는 파라미터들과 연관될 수도 있다. 이들 파라미터들은 CTU 사이즈 (샘플들에서 CTU (132) 의 사이즈를 나타냄), 최소 쿼드트리 사이즈 (MinQTSize, 최소 허용된 쿼드트리 리프 노드 사이즈를 나타냄), 최대 바이너리 트리 사이즈 (MaxBTSize, 최대 허용된 바이너리 트리 루트 노드 사이즈를 나타냄), 최대 바이너리 트리 심도 (MaxBTDepth, 최대 허용된 바이너리 트리 심도를 나타냄), 및 최소 바이너리 트리 사이즈 (MinBTSize, 최소 허용된 바이너리 트리 리프 노드 사이즈를 나타냄) 를 포함할 수도 있다.

[0062] CTU 에 대응하는 QTBT 구조의 루트 노드는 QTBT 구조의 제 1 레벨에서 4 개의 자식 노드들을 가질 수도 있고, 이들의 각각은 쿼드트리 파티셔닝에 따라 파티셔닝될 수도 있다. 즉, 제 1 레벨의 노드들은 리프 노드들 (자식 노드들이 없음) 이거나 또는 4 개의 자식 노드들을 갖는다. QTBT 구조 (130) 의 예는 그러한 노드들을 브랜치들에 대한 실선들을 갖는 자식 노드들 및 부모 노드를 포함하는 것으로서 나타낸다. 제 1 레벨의 노드들이 최대 허용된 바이너리 트리 루트 노드 사이즈 (MaxBTSize) 보다 더 크지 않으면, 그들은 개개의 바이

너리 트리들에 의해 추가로 파티셔닝될 수도 있다. 하나의 노드의 바이너리 트리 스플릿팅은 스플릿으로부터 발생하는 노드들이 최소 허용된 바이너리 트리 리프 노드 사이즈 (MinBTSize) 또는 최대 허용된 바이너리 트리 심도 (MaxBTDepth) 에 도달할 때까지 반복될 수도 있다. QTBT 구조 (130) 의 예는 그러한 노드들을 브랜치들에 대한 점선들을 갖는 것으로서 나타낸다. 바이너리 트리 리프 노드는, 임의의 추가 파티셔닝 없이, 예측 (예를 들어, 인트라 픽처 또는 인터 픽처 예측) 및 변환을 위해 사용되는 코딩 유닛 (CU) 으로 지칭된다. 상기 논의된 바와 같이, CU들은 또한, "비디오 블록들" 또는 "블록들" 로 지칭될 수도 있다.

[0063] QTBT 파티셔닝 구조의 하나의 예에서, CTU 사이즈는 128x128 (루마 샘플들 및 2 개의 대응하는 64x64 크로마 샘플들) 로서 설정되고, MinQTSize 는 16x16 으로서 설정되고, MaxBTSize 는 64x64 로서 설정되고, (폭 및 높이 양자 모두에 대한) MinBTSize 는 4 로서 설정되고, 그리고 MaxBTDepth 는 4 로서 설정된다. 쿼드트리 파티셔닝은 쿼드트리 리프 노드들을 생성하기 위해 먼저 CTU 에 적용된다. 쿼드트리 리프 노드들은 16x16 (즉, MinQTSize) 으로부터 128x128 (즉, CTU 사이즈) 까지의 사이즈를 가질 수도 있다. 리프 쿼드트리 노드가 128x128 인 경우, 사이즈가 MaxBTSize (즉, 이 예에서는 64x64) 를 초과하기 때문에 그것은 바이너리 트리에 의해 추가로 스플릿팅되지 않을 것이다. 그렇지 않으면, 리프 쿼드트리 노드는 바이너리 트리에 의해 추가로 파티셔닝될 것이다. 따라서, 쿼드트리 리프 노드는 또한 바이너리 트리에 대한 루트 노드이고 바이너리 트리 심도를 0 으로서 갖는다. 바이너리 트리 심도가 MaxBTDepth (이 예에서는 4) 에 도달할 때, 추가의 스플릿팅이 허용되지 않는다. 바이너리 트리 노드가 MinBTSize (이 예에서는 4) 와 동일한 폭을 가질 때, 그것은 추가의 수평 스플릿팅이 허용되지 않음을 암시한다. 유사하게, 높이가 MinBTSize 와 동일한 바이너리 트리 노드는 그 바이너리 트리 노드에 대해 추가의 수직 스플릿팅이 허용되지 않음을 암시한다. 상기 언급된 바와 같이, 바이너리 트리의 리프 노드들은 CU들로 지칭되고, 추가의 파티셔닝 없이 예측 및 변환에 따라 추가로 프로세싱된다.

[0064] 도 3 은 본 개시의 기법들을 수행할 수도 있는 예시적인 비디오 인코더 (200) 를 나타낸 블록도이다. 도 3 은 설명의 목적으로 제공되며 본 개시에 폭넓게 예시되고 기재되는 바와 같이 기법들을 제한하는 것으로 고려되지 않아야 한다. 설명의 목적으로, 본 개시는 HEVC 비디오 코딩 표준 및 개발 중인 H.266 비디오 코딩 표준과 같은 비디오 코딩 표준들의 맥락에서 비디오 인코더 (200) 를 설명한다. 그러나, 본 개시의 기법들은 이들 비디오 코딩 표준들에 제한되지 않으며, 일반적으로 비디오 인코딩 및 디코딩에 적용가능하다.

[0065] 도 3 의 예에서, 비디오 인코더 (200) 는 비디오 데이터 메모리 (230), 모드 선택 유닛 (202), 잔차 생성 유닛 (204), 변환 프로세싱 유닛 (206), 양자화 유닛 (208), 역 양자화 유닛 (210), 역 변환 프로세싱 유닛 (212), 재구성 유닛 (214), 필터 유닛 (216), 디코딩된 픽처 버퍼 (DPB)(218), 및 엔트로피 인코딩 유닛 (220) 을 포함한다. 비디오 데이터 메모리 (230), 모드 선택 유닛 (202), 잔차 생성 유닛 (204), 변환 프로세싱 유닛 (206), 양자화 유닛 (208), 역 양자화 유닛 (210), 역 변환 프로세싱 유닛 (212), 재구성 유닛 (214), 필터 유닛 (216), DPB (218), 및 엔트로피 인코딩 유닛 (220) 의 어느 것 또는 전부는 하나 이상의 프로세서들에서 또는 프로세싱 회로에서 구현될 수도 있다. 더욱이, 비디오 인코더 (200) 는 이들 및 다른 기능들을 수행하기 위해 추가적인 또는 대안적인 프로세서들 또는 프로세싱 회로를 포함할 수도 있다.

[0066] 비디오 데이터 메모리 (230) 는, 비디오 인코더 (200) 의 컴포넌트들에 의해 인코딩될 비디오 데이터를 저장할 수도 있다. 비디오 인코더 (200) 는 예를 들어, 비디오 소스 (104) (도 1) 로부터 비디오 데이터 메모리 (230) 에 저장된 비디오 데이터를 수신할 수도 있다. DPB (218) 는 비디오 인코더 (200) 에 의한 후속 비디오 데이터의 예측에 사용하기 위해 레퍼런스 비디오 데이터를 저장하는 레퍼런스 픽처 메모리로서 작용할 수도 있다. 비디오 데이터 메모리 (230) 및 DPB (218) 는 동기식 동적 랜덤 액세스 메모리 (SDRAM) 를 포함한 동적 랜덤 액세스 메모리 (DRAM), 자기저항성 RAM (MRAM), 저항성 RAM (RRAM), 또는 다른 타입들의 메모리 디바이스들과 같은 다양한 메모리 디바이스들 중 임의의 것에 의해 형성될 수도 있다. 비디오 데이터 메모리 (230) 및 DPB (218) 는 동일한 메모리 디바이스 또는 별도의 메모리 디바이스들에 의해 제공될 수도 있다. 다양한 예들에서, 비디오 데이터 메모리 (230) 는 예시된 바와 같이 비디오 인코더 (200) 의 다른 컴포넌트들과 온-칩이거나, 또는 그 컴포넌트들에 대하여 오프-칩일 수도 있다.

[0067] 본 개시에서, 비디오 데이터 메모리 (230) 에 대한 언급은 이처럼 구체적으로 기재되지 않으면 비디오 인코더 (200) 내부의 메모리 또는 이처럼 구체적으로 기재되지 않으면 비디오 인코더 (200) 외부의 메모리로 제한되는 것으로 해석되지 않아야 한다. 오히려, 비디오 데이터 메모리 (230) 에 대한 언급은 비디오 인코더 (200) 가 인코딩을 위해 수신하는 비디오 데이터 (예를 들어, 인코딩될 현재 블록에 대한 비디오 데이터) 를 저장하는 레퍼런스 메모리로서 이해되어야 한다. 도 1 의 메모리 (106) 는 또한 비디오 인코더 (200) 의 다양한 유닛

들로부터의 출력들의 일시적 저장을 제공할 수도 있다.

[0068] 도 3 의 다양한 유닛들은 비디오 인코더 (200) 에 의해 수행되는 동작들의 이해를 돕기 위해 도시된다. 그 유닛들은 고정 기능 회로들, 프로그래밍가능 회로들, 또는 이들의 조합으로서 구현될 수도 있다. 고정 기능 회로들은 특정 기능성을 제공하는 회로들을 지칭하며, 수행될 수도 있는 동작들에 대해 미리설정된다. 프로그래밍가능 회로들은 다양한 태스크들을 수행하도록 프로그래밍될 수도 있는 회로들을 지칭하고, 수행될 수도 있는 동작들에서 유연한 기능성을 제공한다. 예를 들어, 프로그래밍가능 회로들은, 프로그래밍가능 회로들이 소프트웨어 또는 펌웨어의 명령들에 의해 정의된 방식으로 동작하게 하는 소프트웨어 또는 펌웨어를 실행할 수도 있다. 고정 기능 회로들은 (예를 들어, 파라미터들을 수신하거나 또는 파라미터들을 출력하기 위해) 소프트웨어 명령들을 실행할 수도 있지만, 고정 기능 회로들이 수행하는 동작들의 타입들은 일반적으로 불변이다. 일부 예들에서, 유닛들 중 하나 이상은 별개의 회로 블록들 (고정 기능 또는 프로그래밍가능) 일 수도 있고, 일부 예들에 있어서, 하나 이상의 유닛들은 집적 회로들일 수도 있다.

[0069] 비디오 인코더 (200) 는 프로그래밍가능 회로들로부터 형성된, 산술 논리 유닛들 (arithmetic logic unit; ALU 들), 기본 함수 유닛들 (elementary function unit; EFU들), 디지털 회로들, 아날로그 회로들, 및/또는 프로그래밍가능 코어들을 포함할 수도 있다. 비디오 인코더 (200) 의 동작들이 프로그래밍가능 회로들에 의해 실행되는 소프트웨어에 의해 수행되는 예들에서, 메모리 (106) (도 1) 는 비디오 인코더 (200) 가 수신하고 실행하는 소프트웨어의 오브젝트 코드를 저장할 수도 있거나 또는 비디오 인코더 (200) 내의 다른 메모리 (미도시) 가 이러한 명령들을 저장할 수도 있다 .

[0070] 비디오 데이터 메모리 (230) 는 수신된 비디오 데이터를 저장하도록 구성된다. 비디오 인코더 (200) 는 비디오 데이터 메모리 (230) 로부터 비디오 데이터의 픽처를 취출하고 비디오 데이터를 잔차 생성 유닛 (204) 및 모드 선택 유닛 (202) 에 제공할 수도 있다. 비디오 데이터 메모리 (230) 에서의 비디오 데이터는 인코딩될 원시 비디오 데이터일 수도 있다.

[0071] 모드 선택 유닛 (202) 은 모션 추정 유닛 (222), 모션 보상 유닛 (224), 및 인트라-예측 유닛 (226) 을 포함한다. 모드 선택 유닛 (202) 은 다른 예측 모드들에 따라 비디오 예측을 수행하기 위해 부가적인 기능 유닛들을 포함할 수도 있다. 예를 들어, 모드 선택 유닛 (202) 은 팔레트 유닛, 인트라-블록 카피 유닛 (모션 추정 유닛 (222) 및/또는 모션 보상 유닛 (224) 의 일부일 수도 있음), 아핀 유닛, 선형 모델 (LM) 유닛 등을 포함할 수도 있다.

[0072] 모드 선택 유닛 (202) 은 일반적으로 인코딩 파라미터들의 조합들 및 그러한 조합들에 대한 결과의 레이트-왜곡 값들을 테스트하기 위해 다중 인코딩 패스들을 조정한다. 인코딩 파라미터들은 CTU들의 CU들로의 파티셔닝, CU들에 대한 예측 모드들, CU들의 잔차 데이터에 대한 변환 타입들, CU들의 잔차 데이터에 대한 양자화 파라미터들 등을 포함할 수도 있다. 모드 선택 유닛 (202) 은 궁극적으로 다른 테스트된 조합들보다 우수한 레이트-왜곡 값들을 갖는 인코딩 파라미터들의 조합을 선택할 수도 있다.

[0073] 비디오 인코더 (200) 는 비디오 데이터 메모리 (230) 로부터 취출된 픽처를 일련의 CTU들로 파티셔닝하고, 슬라이스 내에 하나 이상의 CTU들을 캡슐화할 수도 있다. 모드 선택 유닛 (202) 은 상기 설명된 HEVC 의 쿼드트리 구조 또는 QTBT 구조와 같은, 트리 구조에 따라 픽처의 CTU 를 파티셔닝할 수도 있다. 상기 설명된 바와 같이, 비디오 인코더 (200) 는 트리 구조에 따라 CTU 를 파티셔닝하는 것으로부터 하나 이상의 CU들을 형성할 수도 있다. 그러한 CU 는 일반적으로 "비디오 블록" 또는 "블록" 으로도 또한 지칭될 수도 있다.

[0074] 일반적으로, 모드 선택 유닛 (202) 은 또한 그것의 컴포넌트들 (예를 들어, 모션 추정 유닛 (222), 모션 보상 유닛 (224), 및 인트라 예측 유닛 (226)) 을 제어하여 현재 블록 (예를 들어, 현재 CU, 또는 HEVC 에서, PU 및 TU 의 오버랩하는 부분) 에 대한 예측 블록을 생성한다. 현재 블록의 인터 예측을 위해, 모션 추정 유닛 (222) 은 하나 이상의 레퍼런스 픽처들 (DPB (218) 에 저장된 하나 이상의 이전에 코딩된 픽처들) 에서 하나 이상의 밀접하게 매칭하는 레퍼런스 블록들을 식별하기 위해 모션 탐색을 수행할 수도 있다. 특히, 모션 추정 유닛 (222) 은, 예를 들어, 절대 차이의 합 (SAD), 제곱 차이들의 합 (SSD), 평균 절대 차이 (MAD), 평균 제곱 차이들 (MSD) 등에 따라, 잠재적 레퍼런스 블록이 현재 블록에 얼마나 유사한지를 나타내는 값을 계산할 수도 있다. 모션 추정 유닛 (222) 은 일반적으로 고려되는 레퍼런스 블록과 현재 블록 사이의 샘플 별 차이들을 사용하여 이들 계산들을 수행할 수도 있다. 모션 추정 유닛 (222) 은 현재 블록과 가장 근접하게 매칭하는 레퍼런스 블록을 표시하는, 이러한 계산들로부터 야기되는 최저 값을 갖는 레퍼런스 블록을 식별할 수도 있다.

[0075] 모션 추정 유닛 (222) 은 현재 픽처에서의 현재 블록의 포지션에 대한 레퍼런스 픽처들에서의 레퍼런스 블록들

의 포지션들을 정의하는 하나 이상의 모션 벡터 (MV) 들을 형성할 수도 있다. 모션 추정 유닛 (222) 은 그 후 모션 벡터들을 모션 보상 유닛 (224) 에 제공할 수도 있다. 예를 들어, 단방향 인터-예측에 대해, 모션 추정 유닛 (222) 은 단일 모션 벡터를 제공할 수도 있는 반면, 양방향 인터-예측에 대해, 모션 추정 유닛 (222) 은 2 개의 모션 벡터들을 제공할 수도 있다. 그 후, 모션 보상 유닛 (224) 은 모션 벡터들을 사용하여 예측 블록을 생성할 수도 있다. 예를 들어, 모션 보상 유닛 (224) 은 모션 벡터를 사용하여 레퍼런스 블록의 데이터를 취출 (retrieve) 할 수도 있다. 다른 예로서, 모션 벡터가 분수 샘플 정밀도를 갖는다면, 모션 보상 유닛 (224) 은 하나 이상의 보간 필터들에 따라 예측 블록에 대한 값들을 보간할 수도 있다. 또한, 양방향 인터 예측에 대해, 모션 보상 유닛 (224) 은 각각의 모션 벡터들에 의해 식별된 2 개의 레퍼런스 블록들에 대한 데이터를 취출하고, 예를 들어 샘플 별 평균화 또는 가중된 평균화를 통해 취출된 데이터를 결합할 수도 있다.

[0076] 다른 예로서, 인트라 예측, 또는 인트라 예측 코딩에 대해, 인트라 예측 유닛 (226) 은 현재 블록에 이웃하는 샘플들로부터 예측 블록을 생성할 수도 있다. 예를 들어, 방향성 모드들에 대해, 인트라 예측 유닛 (226) 은 일반적으로 이웃하는 샘플들의 값들을 수학적으로 결합하고 현재 블록에 걸쳐 정의된 방향에서 이들 계산된 값들을 팝퓰레이트 (populate) 하여 예측 블록을 생성할 수도 있다. 다른 예로서, DC 모드에 대해, 인트라 예측 유닛 (226) 은 현재 블록에 대한 이웃하는 샘플들의 평균을 계산하고 예측 블록을 생성하여 예측 블록의 각각의 샘플에 대해 이러한 결과의 평균을 포함할 수도 있다.

[0077] 모드 선택 유닛 (202) 은 예측 블록을 잔차 생성 유닛 (204) 에 제공한다. 잔차 생성 유닛 (204) 은 비디오 데이터 메모리 (230) 로부터의 현재 블록의 원시의, 코딩되지 않은 버전 및 모드 선택 유닛 (202) 으로부터의 예측 블록을 수신한다. 잔차 생성 유닛 (204) 은 현재 블록과 예측 블록 사이의 샘플 별 차이들을 계산한다. 결과의 샘플 별 차이들은 현재 블록에 대한 잔차 블록을 정의한다. 일부 예들에서, 잔차 생성 유닛 (204) 은 또한 잔차 차분 펄스 코드 변조 (residual differential pulse code modulation; RDPCM) 를 사용하여 잔차 블록을 생성하기 위해 잔차 블록에서의 샘플 값들 사이의 차이들을 결정할 수도 있다. 일부 예들에서, 잔차 생성 유닛 (204) 은 바이너리 감산을 수행하는 하나 이상의 감산 회로들을 사용하여 형성될 수도 있다.

[0078] 모드 선택 유닛 (202) 이 CU들을 PU들로 파티셔닝하는 예들에서, 각각의 PU 는 루마 예측 유닛 및 대응하는 크로마 예측 유닛들과 연관될 수도 있다. 비디오 인코더 (200) 및 비디오 디코더 (300) 는 다양한 사이즈를 갖는 PU들을 지원할 수도 있다. 상기 나타낸 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 2Nx2N 임을 가정하면, 비디오 인코더 (200) 는 인트라-예측을 위해 2Nx2N 또는 NxN 의 PU 사이즈들을 지원하고, 인터-예측을 위해 2Nx2N, 2NxN, Nx2N, NxN, 기타 등등의 대칭적인 PU 사이즈들을 지원할 수도 있다. 비디오 인코더 (200) 및 비디오 디코더 (300) 는 또한, 인터 예측을 위해 2NxnU, 2NxnD, nLx2N, 및 nRx2N 의 PU 크기들에 대한 비대칭적인 파티셔닝을 지원할 수도 있다.

[0079] 모드 선택 유닛 (202) 이 CU 를 PU들로 추가로 파티셔닝하지 않는 예들에 있어서, 각각의 CU 는 루마 코딩 블록 및 대응하는 크로마 코딩 블록들과 연관될 수도 있다. 위와 같이, CU 의 사이즈는 CU 의 루마 코딩 블록의 사이즈를 지칭할 수도 있다. 비디오 인코더 (200) 및 비디오 디코더 (300) 는 2Nx2N, 2NxN, 또는 Nx2N 의 CU 크기들을 지원할 수도 있다.

[0080] 인트라 블록 카피 모드 코딩, 아핀 모드 코딩 및 선형 모델 (LM) 모드 코딩과 같은 다른 비디오 코딩 기법들에 대해, 몇몇 예들에서와 같이, 모드 선택 유닛 (202) 은 코딩 기술과 연관된 개개의 유닛들을 통해, 인코딩될 현재 블록에 대한 예측 블록을 생성한다. 팔레트 모드 코딩과 같은 일부 예들에서, 모드 선택 유닛 (202) 은 예측 블록을 생성하지 않을 수도 있고, 대신에 선택된 팔레트에 기초하여 블록을 재구성하는 방식을 표시하는 신택스 엘리먼트들을 생성할 수도 있다. 이러한 모드들에서, 모드 선택 유닛 (202) 은 이들 신택스 엘리먼트들을 인코딩되도록 엔트로피 인코딩 유닛 (220) 에 제공할 수도 있다.

[0081] 상술한 바와 같이, 잔차 생성 유닛 (204) 은 현재 블록 및 대응하는 예측 블록에 대해 비디오 데이터를 수신한다. 잔차 생성 유닛 (204) 은 그 후 현재 블록에 대한 잔차 블록을 생성한다. 잔차 블록을 생성하기 위해, 잔차 생성 유닛 (204) 은 현재 블록과 예측 블록 사이의 샘플 별 차이들을 계산한다.

[0082] 변환 프로세싱 유닛 (206) 은 잔차 블록에 하나 이상의 변환들을 적용하여 변환 계수들의 블록 (본 명세서에서는 "변환 계수 블록" 으로 지칭됨) 을 생성한다. 변환 프로세싱 유닛 (206) 은 다양한 변환들을 잔차 블록에 적용하여 변환 계수 블록을 형성할 수도 있다. 예를 들어, 변환 프로세싱 유닛 (206) 은 이산 코사인 변환 (DCT), 방향성 변환, Karhunen-Loeve 변환 (KLT), 또는 개념적으로 유사한 변환을 잔차 블록에 적용할 수도

있다. 일부 예들에서, 변환 프로세싱 유닛 (206) 은 잔차 블록에 대한 다중 변환들, 예를 들어 1 차 변환 및 2 차 변환, 이를 테면 회전 변환을 수행할 수도 있다. 일부 예들에서, 변환 프로세싱 유닛 (206) 은 잔차 블록에 변환들을 적용하지 않는다.

[0083] 양자화 유닛 (208) 은 양자화된 변환 계수 블록을 생성하기 위해 변환 계수 블록에서의 변환 계수들을 양자화할 수도 있다. 양자화 유닛 (208) 은 현재 블록과 연관된 양자화 파라미터 (QP) 값에 따라 변환 계수 블록의 변환 계수들을 양자화할 수도 있다. 비디오 인코더 (200) 는 (예컨대, 모드 선택 유닛 (202) 을 통해) CU 와 연관된 QP 값을 조정함으로써 현재 블록과 연관된 변환 계수 블록들에 적용되는 양자화도를 조정할 수도 있다. 양자화는 정보의 손실을 도입할 수도 있으며, 따라서, 양자화된 변환 계수들은 변환 프로세싱 유닛 (206) 에 의해 생성된 오리지널 변환 계수들보다 더 낮은 정밀도를 가질 수도 있다.

[0084] 역 양자화 유닛 (210) 및 역 변환 프로세싱 유닛 (212) 은 각각 양자화된 변환 계수 블록에 역 양자화 및 역 변환들을 적용하여, 변환 계수 블록으로부터 잔차 블록을 재구성할 수도 있다. 재구성 유닛 (214) 은 모드 선택 유닛 (202) 에 의해 생성된 예측 블록 및 재구성된 잔차 블록에 기초하여 (잠재적으로 어느 정도의 왜곡을 가짐에도 불구하고) 현재 블록에 대응하는 재구성된 블록을 생성할 수도 있다 . 예를 들어, 재구성 유닛 (214) 은 재구성된 잔차 블록의 샘플들을, 모드 선택 유닛 (202) 에 의해 생성된 예측 블록으로부터의 대응하는 샘플들에 가산하여 재구성된 블록을 생성할 수도 있다.

[0085] 필터 유닛 (216) 은 재구성된 블록에 대해 하나 이상의 필터 동작들을 수행할 수도 있다. 예를 들어, 필터 유닛 (216) 은 CU들의 에지들을 따라 블록키니스 아티팩트들 (blockiness artifacts) 을 감소시키기 위해 디블록킹 동작들을 수행할 수도 있다. 필터 유닛 (216) 의 동작들은 일부 예들에서 생략될 수도 있다. 필터 유닛 (216) 은 단독으로 또는 임의의 조합으로 본 개시의 크로스-컴포넌트 적응형 루프 필터링 (CC-ALF) 기법들을 수행할 수도 있다. 예를 들어, 필터 유닛 (216) 은 도 5 를 참조하여 아래에서 논의되는 바와 같이 CC-ALF 를 수행할 수도 있다. 필터 유닛 (216) 은 CC-ALF에 대한 하나 이상의 계수들을 생성할 수도 있다. 예를 들어, 필터 유닛 (216) 은 루마 블록으로부터 제 1 중간 크로마 블록을 생성할 때 사용될 필터 계수들의 제 1 세트 및 루마 블록으로부터 제 2 중간 크로마 블록을 생성할 때 사용될 필터 계수들의 제 2 세트를 생성할 수도 있다. 위에서 논의된 바와 같이 그리고 본 개시의 하나 이상의 기법들에 따르면, 필터 유닛 (216) 은 생성된 필터 계수들의 절대 값들을 제로 또는 2 의 거듭제곱 (예를 들어, 1, 2, 4, 8, 16, 32, 64, 128, 256 등) 이 되도록 제한할 수도 있다. 마찬가지로, 엔트로피 인코딩 유닛 (220) 은 본 개시의 기법들에 따라 크로스-컴포넌트 적응형 루프 필터링 파라미터들을 엔트로피 인코딩하도록 구성될 수도 있다. 예를 들어, 필터 계수들의 실제 값들을 인코딩하는 것과는 대조적으로, 엔트로피 인코딩 유닛 (220) 은 필터 계수들의 지수 값을 인코딩할 수도 있고, 비디오 디코더는 그 지수 값에 기초하여 필터 계수들의 실제 값들을 재구성할 수도 있다.

[0086] 비디오 인코더 (200) 는 DPB (218) 에 재구성된 블록들을 저장한다. 예를 들어, 필터 유닛 (216) 의 동작들이 필요하지 않은 예들에서, 재구성 유닛 (214) 은 재구성된 블록들을 DPB (218) 에 저장할 수도 있다. 필터 유닛 (216) 의 동작들이 필요한 예들에서, 필터 유닛 (216) 은 필터링된 재구성된 블록들을 DPB (218) 에 저장할 수도 있다. 모션 추정 유닛 (222) 및 모션 보상 유닛 (224) 은 재구성된 (및 잠재적으로 필터링된) 블록들로부터 형성된 DPB (218) 로부터 레퍼런스 픽처를 취출하여, 후속 인코딩된 픽처들의 블록들을 인터 예측할 수도 있다. 또한, 인트라 예측 유닛 (226) 은 현재 픽처에서의 다른 블록들을 인트라 예측하기 위해 현재 픽처의 DPB (218) 에서 재구성된 블록들을 사용할 수도 있다.

[0087] 일반적으로, 엔트로피 인코딩 유닛 (220) 은 비디오 인코더 (200) 의 다른 기능 컴포넌트들로부터 수신된 신택스 엘리먼트들을 엔트로피 인코딩할 수도 있다. 예를 들어, 엔트로피 인코딩 유닛 (220) 은 양자화 유닛 (208) 으로부터 양자화된 변환 계수 블록들을 엔트로피 인코딩할 수도 있다. 다른 예로서, 엔트로피 인코딩 유닛 (220) 은 모드 선택 유닛 (202) 으로부터 예측 신택스 엘리먼트들 (예를 들어, 인터 예측에 대한 모션 정보 또는 인트라 예측에 대한 인트라 모드 정보) 을 엔트로피 인코딩할 수도 있다. 엔트로피 인코딩 유닛 (220) 은 엔트로피 인코딩된 데이터를 생성하기 위해, 비디오 데이터의 다른 예인, 신택스 엘리먼트들에 대해 하나 이상의 엔트로피 인코딩 동작들을 수행할 수도 있다. 예를 들어, 엔트로피 인코딩 유닛 (220) 은 컨텍스트 적응적 가변 길이 코딩 (CAVLC) 동작, CABAC 동작, V2V (variable-to-variable) 길이 코딩 동작, 신택스 기반 컨텍스트 적응 이진 산술 코딩 (SBAC) 동작, 확률 간격 파티셔닝 엔트로피 (PIPE) 코딩 동작, 지수-골롬 인코딩 동작, 또는 다른 타입의 엔트로피 인코딩 동작을 데이터에 대해 수행할 수도 있다. 일부 예들에서, 엔트로피 인코딩 유닛 (220) 은 신택스 엘리먼트들이 엔트로피 인코딩되지 않는 바이패스 모드에서 동작할 수도

있다.

[0088] 비디오 인코더 (200) 는 픽처 또는 슬라이스의 블록들을 재구성하는데 필요한 엔트로피 인코딩된 신택스 엘리먼트들을 포함하는 비트스트림을 출력할 수도 있다. 특히, 엔트로피 인코딩 유닛 (220) 이 비트스트림을 출력할 수도 있다.

[0089] 상기 설명된 동작들은 블록에 대하여 설명된다. 이러한 설명은 루마 코딩 블록 및/또는 크로마 코딩 블록들에 대한 동작들인 것으로 이해되어야 한다. 상술한 바와 같이, 일부 예들에서, 루마 코딩 블록 및 크로마 코딩 블록들은 CU 의 루마 및 크로마 컴포넌트들이다. 일부 예들에서, 루마 코딩 블록 및 크로마 코딩 블록들은 PU 의 루마 및 크로마 컴포넌트들이다.

[0090] 일부 예들에서, 루마 코딩 블록에 대해 수행되는 동작들은 크로마 코딩 블록들에 대해 반복될 필요가 없다. 일 예로서, 루마 코딩 블록에 대한 모션 벡터 (MV) 및 레퍼런스 픽처를 식별하기 위한 동작들이, 크로마 블록들에 대한 MV 및 레퍼런스 픽처를 식별하기 위해 반복될 필요는 없다. 오히려, 루마 코딩 블록에 대한 MV 는 크로마 블록들에 대한 MV 를 결정하도록 스케일링될 수도 있고, 레퍼런스 픽처는 동일할 수도 있다. 다른 예로서, 인트라-예측 프로세스는 루마 코딩 블록 및 크로마 코딩 블록들에 대해 동일할 수도 있다.

[0091] 비디오 인코더 (200) 는 비디오 데이터를 저장하도록 구성된 메모리, 및 회로에서 구현되고 본 개시의 크로스-컴포넌트 적응형 루프 필터링 기법들을 수행하도록 구성된 하나 이상의 프로세싱 유닛들을 포함하는 비디오 데이터를 인코딩하도록 구성된 디바이스의 예를 나타낸다.

[0092] 도 4 는 본 개시의 기법들을 수행할 수도 있는 예시적인 비디오 디코더 (300) 를 나타내는 블록도이다. 도 4 는 설명의 목적을 위해 제공되고 본 개시에 폭넓게 예시되고 설명된 기법들에 대해 한정하지 않는다. 설명의 목적으로, 본 개시는 JEM, VVC 및 HEVC 의 기법들에 따른 비디오 디코더 (300) 를 기술한다. 그러나, 본 개시의 기법들은 다른 비디오 코딩 표준들로 구성되는 비디오 코딩 디바이스들에 의해 수행될 수도 있다.

[0093] 도 4 의 예에서, 비디오 디코더 (300) 는 코딩된 픽처 버퍼 (CPB) 메모리 (320), 엔트로피 디코딩 유닛 (302), 예측 프로세싱 유닛 (304), 역 양자화 유닛 (306), 역 변환 프로세싱 유닛 (308), 재구성 유닛 (310), 필터 유닛 (312), 및 디코딩된 픽처 버퍼 (DPB) (314) 를 포함한다. CPB 메모리 (320), 엔트로피 디코딩 유닛 (302), 예측 프로세싱 유닛 (304), 역 양자화 유닛 (306), 역 변환 프로세싱 유닛 (308), 재구성 유닛 (310), 필터 유닛 (312), 및 DPB (314) 의 어느 것 또는 전부는 하나 이상의 프로세서들에서 또는 프로세싱 회로에서 구현될 수도 있다. 더욱이, 비디오 디코더 (300) 는 이들 및 다른 기능들을 수행하기 위해 추가적인 또는 대안적인 프로세서들 또는 프로세싱 회로를 포함할 수도 있다.

[0094] 예측 프로세싱 유닛 (304) 은 모션 보상 유닛 (316) 및 인트라 예측 프로세싱 유닛 (318) 을 포함한다. 예측 프로세싱 유닛 (304) 은 다른 예측 모드들에 따라 예측을 수행하기 위해 부가적인 유닛들을 포함할 수도 있다. 예들로서, 예측 프로세싱 유닛 (304) 은 팔레트 유닛, 인트라-블록 카피 유닛 (이는 모션 보상 유닛 (316) 의 부분을 형성할 수도 있음), 아핀 유닛, 선형 모델 (LM) 유닛 등을 포함할 수도 있다. 다른 예들에서, 비디오 디코더 (300) 는 더 많거나, 더 적거나, 또는 상이한 기능성 컴포넌트들을 포함할 수도 있다.

[0095] CPB 메모리 (320) 는, 비디오 디코더 (300) 의 컴포넌트들에 의해 디코딩될 인코딩된 비디오 비트스트림과 같은 비디오 데이터를 저장할 수도 있다. CPB 메모리 (320) 에 저장된 비디오 데이터는, 예를 들어 컴퓨터 판독가능 매체 (110) (도 1) 로부터 획득될 수도 있다. CPB 메모리 (320) 는 인코딩된 비디오 비트스트림으로부터 인코딩된 비디오 데이터 (예를 들어, 신택스 엘리먼트들) 를 저장하는 CPB 를 포함할 수도 있다. 또한, CPB 메모리 (320) 는 비디오 디코더 (300) 의 다양한 유닛들로부터의 출력들을 나타내는 일시적 데이터와 같은, 코딩된 픽처의 신택스 엘리먼트들 이외의 비디오 데이터를 저장할 수도 있다. DPB (314) 는 일반적으로, 인코딩된 비디오 비트스트림의 후속 데이터 또는 픽처들을 디코딩할 때, 레퍼런스 비디오 데이터로서 비디오 디코더 (300) 가 출력 및/또는 사용할 수도 있는 디코딩된 픽처들을 저장한다. CPB 메모리 (320) 및 DPB (314) 는 다양한 메모리 디바이스들, 예컨대 동기식 동적 랜덤 액세스 메모리 (SDRAM) 를 포함한 DRAM, 자기저항성 RAM (MRAM), 저항성 RAM (RRAM) 과 같은 다양한 메모리 디바이스들, 또는 다른 타입들의 메모리 디바이스들 중 임의의 것에 의해 형성될 수도 있다. CPB 메모리 (320) 및 DPB (314) 는 동일한 메모리 디바이스 또는 별도의 메모리 디바이스들에 의해 제공될 수도 있다. 여러 예들에서, APS 메모리 (320) 는 비디오 디코더 (300) 의 다른 컴포넌트들과 온-칩이거나 그 컴포넌트들에 대하여 오프-칩일 수도 있다.

[0096] 추가적으로 또는 대안적으로, 일부 예들에서, 비디오 디코더 (300) 는 메모리 (120) (도 1) 로부터 코딩된 비디오 데이터를 취출할 수도 있다. 즉, 메모리 (120) 는 CPB 메모리 (320) 로 상기 논의된 바와 같이 데이터를

저장할 수도 있다. 마찬가지로, 메모리 (120) 는 비디오 디코더 (300) 의 기능성의 일부 또는 전부가 비디오 디코더 (300) 의 프로세싱 회로에 의해 실행되는 소프트웨어에서 구현될 때, 비디오 디코더 (300) 에 의해 실행될 명령들을 저장할 수도 있다.

[0097] 도 4 에 도시된 다양한 유닛들은 비디오 디코더 (300) 에 의해 수행되는 동작들의 이해를 돕기 위해 예시된다. 이 유닛들은 고정 기능 회로들, 프로그래밍가능 회로들, 또는 이들의 조합으로서 구현될 수도 있다. 도 3 과 유사하게, 고정 기능 회로들은 특정 기능성을 제공하는 회로들을 지칭하며, 수행될 수도 있는 동작들에 대해 미리설정된다. 프로그래밍가능 회로들은 다양한 태스크들을 수행하도록 프로그래밍될 수도 있는 회로들을 지칭하고, 수행될 수도 있는 동작들에서 유연한 기능성을 제공한다. 예를 들어, 프로그래밍가능 회로들은, 프로그래밍가능 회로들이 소프트웨어 또는 펌웨어의 명령들에 의해 정의된 방식으로 동작하게 하는 소프트웨어 또는 펌웨어를 실행할 수도 있다. 고정 기능 회로들은 (예를 들어, 파라미터들을 수신하거나 또는 파라미터들을 출력하기 위해) 소프트웨어 명령들을 실행할 수도 있지만, 고정 기능 회로들이 수행하는 동작들의 타입들은 일반적으로 불변이다. 일부 예들에서, 하나 이상의 유닛들은 별개의 회로 블록들 (고정 기능 또는 프로그래밍가능) 일 수도 있고, 일부 예들에서, 하나 이상의 유닛들은 집적 회로들일 수도 있다.

[0098] 비디오 디코더 (300) 는 프로그래밍가능 회로들로부터 형성된, ALU 들, EFU들, 디지털 회로들, 아날로그 회로들, 및/또는 프로그래밍가능 코어들을 포함할 수도 있다. 비디오 디코더 (300) 의 동작들이 프로그래밍가능 회로들 상에서 실행하는 소프트웨어에 의해 수행되는 예들에서, 온-칩 또는 오프-칩 메모리는 비디오 디코더 (300) 가 수신하고 실행하는 소프트웨어의 명령들 (예를 들어, 오브젝트 코드) 을 저장할 수도 있다.

[0099] 엔트로피 디코딩 유닛 (302) 은 인코딩된 비디오 데이터를 CPB 로부터 수신하고, 비디오 데이터를 엔트로피 디코딩하여 신택스 엘리먼트들을 재생할 수도 있다. 예측 프로세싱 유닛 (304), 역 양자화 유닛 (306), 역 변환 프로세싱 유닛 (308), 복원 유닛 (310), 및 필터 유닛 (312) 은 비트스트림으로부터 추출된 신택스 엘리먼트들에 기초하여 디코딩된 비디오 데이터를 생성할 수도 있다.

[0100] 일반적으로, 비디오 디코더 (300) 는 블록 별 단위로 픽처를 재구성한다. 비디오 디코더 (300) 는 개별적으로 각각의 블록에 대해 재구성 동작을 수행할 수도 있다 (여기서 현재 재구성되는, 즉 디코딩되는 블록은 "현재 블록" 으로 지칭될 수도 있음).

[0101] 엔트로피 디코딩 유닛 (302) 은 양자화 파라미터 (QP) 및/또는 변환 모드 표시(들)와 같은 변환 정보 뿐만 아니라, 양자화된 변환 계수 블록의 양자화된 변환 계수들을 정의하는 신택스 엘리먼트들을 엔트로피 디코딩할 수도 있다. 역 양자화 유닛 (306) 은 양자화된 변환 계수 블록과 연관된 QP 를 사용하여, 양자화도 및 유사하게, 적용할 역 양자화 유닛 (306) 에 대한 역 양자화도를 결정할 수도 있다. 역 양자화 유닛 (306) 은 예를 들어, 양자화된 변환 계수들을 역 양자화하기 위해 비트단위 좌측-시프트 동작을 수행할 수도 있다. 따라서, 역 양자화 유닛 (306) 은 변환 계수들을 포함하는 변환 계수 블록을 형성할 수도 있다.

[0102] 역 양자화 유닛 (306) 이 변환 계수 블록을 형성한 후, 역변환 프로세싱 유닛 (308) 은 현재 블록과 연관된 잔차 블록을 생성하기 위해 변환 계수 블록에 하나 이상의 역 변환들을 적용할 수도 있다. 예를 들어, 역 변환 프로세싱 유닛 (308) 은 역 DCT, 역 정수 변환, 역 Karhunen-Loeve 변환 (KLT), 역 회전 변환, 역 방향성 변환, 또는 다른 역 변환을 변환 계수 블록에 적용할 수도 있다.

[0103] 더욱이, 예측 프로세싱 유닛 (304) 은, 엔트로피 디코딩 유닛 (302) 에 의해 엔트로피 디코딩된 예측 정보 신택스 엘리먼트들에 따라 예측 블록을 생성한다. 예를 들어, 예측 정보 신택스 엘리먼트들이 현재 블록이 인터 예측됨을 표시하면, 모션 보상 유닛 (316) 은 예측 블록을 생성할 수도 있다. 이 경우, 예측 정보 신택스 엘리먼트들은 레퍼런스 블록을 취출할 DPB (314) 에서의 레퍼런스 픽처뿐만 아니라 현재 픽처에서의 현재 블록의 위치에 대한 레퍼런스 픽처에서의 레퍼런스 블록의 위치를 식별하는 모션 벡터를 표시할 수도 있다. 모션 보상 유닛 (316) 은 일반적으로 모션 보상 유닛 (224) (도 3) 과 관련하여 설명된 것과 실질적으로 유사한 방식으로 인터 예측 프로세스를 수행할 수도 있다.

[0104] 다른 예로서, 예측 정보 신택스 엘리먼트가 현재 블록이 인트라 예측되는 것을 표시하면, 인트라 예측 유닛 (318) 은 예측 정보 신택스 엘리먼트들에 의해 표시된 인트라 예측 모드에 따라 예측 블록을 생성할 수도 있다. 다시, 인트라 예측 유닛 (318) 은 일반적으로 인트라 예측 유닛 (226)(도 3) 과 관련하여 설명된 것과 실질적으로 유사한 방식으로 인트라 예측 프로세스를 수행할 수도 있다. 인트라 예측 유닛 (318) 은 DPB (314) 로부터 현재 블록에 이웃하는 샘플들의 데이터를 취출할 수도 있다.

[0105] 재구성 유닛 (310) 은 예측 블록 및 잔차 블록을 사용하여 현재 블록을 재구성할 수도 있다. 예를 들어, 재

구성 유닛 (310) 은 잔차 블록의 샘플들을 예측 블록의 대응하는 샘플들에 가산하여 현재 블록을 재구성할 수도 있다.

[0106] 엔트로피 디코딩 유닛 (302) 은 본 개시의 기법들에 따라 크로스-컴포넌트 적응형 루프 필터 파라미터들을 추가로 엔트로피 디코딩할 수도 있다. 예를 들어, 본 개시의 하나 이상의 기법들에 따르면, 엔트로피 디코딩 유닛 (302) 은, 복수의 필터 계수들의 각각에 대해 그리고 인코딩된 비디오 비트스트림으로부터, 특정 필터 계수의 절대 값의 로그 베이스 2 를 나타내는 지수 값을 2 의 그 지수 값의 거듭제곱으로서 특정하는 신택스 엘리먼트를 디코딩할 수도 있다. 특정 필터 계수에 대한 지수 값이 0이 아닌 경우, 엔트로피 디코딩 유닛 (302) 은 인코딩된 비디오 비트스트림으로부터 그리고 특정 필터 계수에 대해, 특정 필터 계수의 부호 (예를 들어, 양 또는 음) 를 특정하는 값을 갖는 신택스 엘리먼트를 디코딩할 수도 있다. 엔트로피 디코딩 유닛 (302) 은 지수 값들에 기초하여 복수의 필터 계수들의 값들을 재구성할 수도 있다. 예를 들어, 엔트로피 디코딩 유닛 (302) 은 다음의 식에 따라 특정 필터 계수의 값을 재구성할 수도 있다:

[0107] $$c(i) = sign(i) * 2^{c'(i)+1}$$

[0108] 여기서, $c(i)$ 는 특정 필터 계수의 값이고, $sign(i)$ 는 시그널링된 부호가 음인 경우 -1 이고 시그널링된 부호가 양인 경우 +1 이며, $c'(i)$ 는 특정 필터 계수에 대한 시그널링된 지수 값이다.

[0109] 엔트로피 디코딩 유닛 (302) 은 재구성된 크로스-컴포넌트 적응형 루프 필터 계수들을 필터 유닛 (312)에 제공할 수도 있다. 필터 유닛 (312) 은 재구성된 블록에 대해 하나 이상의 필터 동작들을 수행할 수도 있다. 예를 들어, 필터 유닛 (312) 은 재구성된 블록들의 에지들을 따라 블록키니스 아티팩트들을 감소시키기 위해 디블록킹 동작들을 수행할 수도 있다. 필터 유닛 (312) 의 동작들이 모든 예들에서 반드시 수행되는 것은 아니다. 본 개시의 기법들에 따르면, 필터 유닛 (312) 은 비디오 데이터의 디코딩된 블록의 크로스-컴포넌트 적응형 루프 필터링을 수행하기 위해 크로스-컴포넌트 적응형 루프 필터 계수들을 사용할 수도 있다.

[0110] 비디오 디코더 (300) 는 DPB (314) 에 재구성된 블록들을 저장할 수도 있다. 위에 논의된 바와 같이, DPB (314) 는 예측 프로세싱 유닛 (304) 에 인트라-예측을 위한 현재 픽처의 샘플들 및 후속 모션 보상을 위해 이전에 디코딩된 픽처들와 같은 레퍼런스 정보를 제공할 수도 있다. 또한, 비디오 디코더 (300) 는 도 1 의 디스플레이 디바이스 (118) 와 같은 디스플레이 디바이스 상에의 후속 프리젠테이션을 위해 DPB (314) 로부터 디코딩된 픽처들을 출력할 수도 있다.

[0111] 이러한 방식으로, 비디오 디코더 (300) 는 비디오 데이터를 저장하도록 구성된 메모리, 및 회로에서 구현되고 본 개시의 크로스-컴포넌트 적응형 루프 필터링 기법들을 수행하도록 구성된 하나 이상의 프로세싱 유닛들을 포함하는 비디오 디코딩 디바이스의 예를 나타낸다.

[0112] 도 5 는 본 개시의 하나 이상의 기법들에 따른, 예시적인 필터 유닛을 나타내는 블록도이다. 도 5 의 필터 유닛 (500) 은 비디오 인코더 (200) 의 필터 유닛 (216) 또는 비디오 디코더 (300) 의 필터 유닛 (312) 의 일 예로서 고려될 수도 있다.

[0113] 필터 유닛 (500) 은 다양한 타입들의 필터링을 수행하도록 구성된 컴포넌트들을 포함할 수도 있다. 예를 들어, 도 5 에 도시된 바와 같이, 필터 유닛 (500) 은 SAO 루마 필터 (502), SAO Cb 필터 (504), 및 SAO Cr 필터 (506) 와 같은, 샘플 적응적 오프셋 (SAO) 필터링을 수행하도록 구성된 컴포넌트들을 포함할 수도 있다. 또한 도 5 에 도시된 바와 같이, 필터 유닛 (500) 은 크로스-컴포넌트 적응형 루프 필터링 (CC-ALF) 을 수행하도록 구성된 컴포넌트들, 이를테면 ALF 루마 필터 (508), CC ALF Cb 필터 (510), CC ALF Cr 필터 (512), ALF 크로마 필터 (514), 가산기 (516), 및 가산기 (518) 를 포함할 수도 있다.

[0114] 동작에서, SAO 루마 필터 (502) 는 비디오 데이터의 입력 루마 블록을 수신하고, 입력 루마 블록에 대해 SAO 필터링을 수행하여 비디오 데이터의 출력 루마 블록을 생성하고, 비디오 데이터의 출력 루마 블록을 ALF 루마 필터 (508), CC ALF Cb 필터 (510), 및 CC ALF Cr 필터 (512) 와 같은 하나 이상의 다른 필터 컴포넌트들에 제공할 수도 있다. SAO Cb 필터 (504) 는 비디오 데이터의 입력 Cb 크로마 블록을 수신하고, 입력 Cb 크로마 블록에 대해 SAO 필터링을 수행하여 비디오 데이터의 출력 Cb 크로마 블록을 생성하고, 비디오 데이터의 출력 Cb 크로마 블록을 ALF 크로마 필터 (514) 와 같은 하나 이상의 다른 필터 컴포넌트들에 제공할 수도 있다. 유

사하게, SAO Cr 필터 (506) 는 비디오 데이터의 입력 Cr 크로마 블록을 수신하고, 입력 Cr 크로마 블록에 대해 SAO 필터링을 수행하여 비디오 데이터의 출력 Cr 크로마 블록을 생성하고, 비디오 데이터의 출력 Cr 크로마 블록을 ALF 크로마 필터 (514) 와 같은 하나 이상의 다른 필터 컴포넌트들에 제공할 수도 있다.

[0115] ALF 컴포넌트들은 SAO 필터링 컴포넌트들에 의해 제공된 비디오 데이터의 블록들에 대해 ALF 를 수행할 수도 있다. 예를 들어, ALF 루마 필터 (508) 는 SAO 루마 필터 (502) 에 의해 제공된 루마 블록에 대해 적응적 루프 필터링을 수행하여, Y 로 표시된 출력 루마 블록을 생성할 수도 있다. 추가적으로, ALF 크로마 필터 (514) 는 SAO Cb 필터 (504) 및 SAO Cr 필터 (506) 에 의해 제공된 크로마 블록에 대해 적응적 루프 필터링을 수행하여, Cb' 및 Cr'으로 표시된 출력 크로마 블록들을 생성할 수도 있다.

[0116] Misra 외 "Cross-Component Adaptive Loop Filter for chroma" Joint Video Experts Team (JVET) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11, 15th Meeting: Gothenburg, SE, 3-12 July 2019, JVET-O0636 (이하 "JVET-O0636") 는 크로스-컴포넌트 적응형 루프 필터 (CC-ALF) 라고 불리는 툴을 제안하였다. CC-ALF 는 적응적 루프 필터 (adaptive loop filter; ALF) 의 일부로서 동작하고, 루마 샘플들을 이용하여 각각의 크로마 컴포넌트를 정제한다. 예를 들어, CC ALF Cb 필터 (510) 및 CC ALF Cr 필터 (512) 는 SAO 루마 필터 (502) 에 의해 제공된 루마 블록에 기초하여 향상/정제 크로마 블록을 각각 생성할 수도 있다 (예를 들어, CC ALF Cb 필터 (510) 는 향상 크로마 블록 Cb+ 를 생성할 수도 있고 CC ALF Cr 필터 (512) 는 향상 크로마 블록 Cr+ 를 생성할 수도 있다). CC ALF Cb 필터 (510) 및 CC ALF Cr 필터 (512) 의 각각은 필터 계수들의 각각의 세트에 기초하여 그들 각각의 향상 크로마 블록을 생성할 수도 있다 (예를 들어, CC ALF Cb 필터 (510) 는 필터 계수들의 제 1 세트를 사용할 수도 있고 CC ALF Cr 필터 (512) 는 필터 계수들의 제 2 세트를 사용할 수도 있다). 예를 들어, CC ALF Cb 필터 (510) 는 다음의 식에 따라 크로마 블록 Cb+ 를 생성할 수도 있다:

$$\Delta I_i(x,y) = \sum_{(x,y)\in S_i} I_0(x_C + x_0, y_C + y_0) c_i(x_0, y_0)$$

[0117]

[0118] 여기서, I_i 는 필터링된 블록이고, I_o 는 필터링되지 않은 블록이고, (x_C, y_C) 는 루마 위치 (x,y) 이고, S_i 는 컬러 컴포넌트 Cb 에 대한 루마에서의 필터 지원이고, $c_i(x_0, y_0)$ 는 필터 계수이다.

[0119] 상기 식에서 나타낸 바와 같이, CC ALF Cb 필터 (510) 및 CC ALF Cr 필터 (512) 의 각각은 많은 곱셈 연산들을 수행할 수도 있다. 위에서 논의된 바와 같이 그리고 본 개시의 하나 이상의 기법들에 따르면, 이들 곱셈 연산들은 비트-시프트 연산들에 의해 대체될 수도 있으며, 이는 곱셈 연산들보다 하드웨어에서 구현하기 위해 실질적으로 덜 리소스 집약적이고 및/또는 더 간단하다. 예를 들어, 비트-시프트 연산들을 사용하여 필터링을 수행하기 위해, CC ALF Cb 필터 (510) 및 CC ALF Cr 필터 (512) 각각은 다음의 식을 이용할 수도 있다:

$$\Delta I_i(x,y) = \sum_{(x,y)\in S_i} \{sign(c_i\ (x_0, y_0)) * [I_0(x_C + x_0, y_C + y_0) << c_i'(x_0, y_0)]\}$$

[0120]

[0121] CC-ALF 는 비트스트림에서의 정보에 의해 제어될 수도 있고, 이 정보는 (적응 파라미터 세트 (APS) 에서 시그널링될 수도 있는) 각각의 크로마 컴포넌트에 대한 전술된 필터 계수들 및 샘플들의 블록들에 대한 필터의 적용을 제어하는 마스크를 포함한다. JVET-O0636 에서, 필터 계수들 각각은 고정 소수점 십진수로 표현된다. 특히, 필터 계수는 하위 10비트를 사용하여 소수 부분을 나타낸다. 각 계수는 필터 템플릿에서 계수 위치에 따라 순서가 달라지는 지수-골롬 (exponential-Golomb; EG) 코딩으로 시그널링된다.

[0122] 위에서 언급된 바와 같이, 본 개시는, 예를 들어, 본 개시의 기법들 중 임의의 것 또는 전부에 따라, JVET-O0636에서 설명된 CC-ALF 툴의 승산들이 개선되고 단순화될 수도 있다는 것을 인식한다. 일반적으로, 비디오 인코더 (200) 및 비디오 디코더 (300) 는, 임의의 조합으로, 예를 들어 이하에서 설명되는 바와 같이, 본 개시의 기술들 중 임의의 기술 또는 그 모든 기술들에 따라 구성될 수도 있다.

[0123] 본 개시의 제 1 기법에 따르면, 비디오 코더 (예를 들어, 비디오 인코더 (200) 및/또는 비디오 디코더 (300)) 는 크로스-컴포넌트 적응형 루프 필터들 (510, 512) 에 대한 계수들의 일부 또는 전부의 값들을 제한할 수도 있다. 예를 들어, 비디오 코더는 (예를 들어, 이들 계수들에 대해 어떠한 곱셈들도 필요하지 않도록) 계수들의 일부 또는 전부의 값들을 제로 또는 2 의 거듭제곱수이도록 제한 (예를 들어, 가능한 선택을 제한) 할 수도

있다. 일부 예들에서, 곱셈들을 수행하는 것 대신에, 비디오 코더 (즉, 비디오 인코더 (200) 또는 비디오 디코더 (300)) 는 샘플들에 비트 시프팅을 적용할 수도 있다. 일 예에서, 비디오 코더는 모든 계수들의 절대값들이 단지 0 또는 2 의 거듭제곱들의 수들이도록 제한할 수도 있다. 다른 예에서, 비디오 코더는 일부 계수들의 절대값들을 단지 0 또는 2 의 거듭제곱들의 수들이도록 제한할 수도 있다. 어떤 필터의 계수들이 제한되는지에 대한 정보는 시그널링 없이 모든 필터들에 대해 동일할 수도 있다. 대안적으로 또는 추가적으로, 그 정보는 시그널링 없이 컬러 컴포넌트의 모든 필터들에 대해 동일할 수도 있다. 대안적으로 또는 추가적으로, 정보는 시퀀스, 픽처, 서브-픽처, 블록, 또는 컬러 컴포넌트에 대해 비트스트림에서 (예를 들어, 하나 이상의 신택스 엘리먼트들로서) 시그널링될 수도 있다.

[0124] 비디오 코더가 (예를 들어, 이들 제약된 계수들의 값들을 시그널링하기 위해) 비트스트림에서 정보를 시그널링하는 일부 예들에서, 비디오 코더는 (0이 아닌 계수들의 부호를 갖는 지수 값인) 맵핑된 값들만을 시그널링할 수도 있다. 제한된 계수들 c(i) 는 다음과 같이 c'(i) 에 맵핑될 수도 있다.

[0125] c(i) 가 0 과 동일한 경우, c'(i) 는 0;

[0126] 그렇지 않은 경우, $c'(i) = sign(c(i)) * (\log_2(abs(c(i)) + 1)$, 여기서, sign(c(i)) 는 c(i) 가 음인 경우 -1, 그렇지 않으면 1.

[0127] 일부 예들에서, 비디오 코더는 c'(i) 를 시그널링하기 위해 고정-오더 골룸 코드들, 고정-길이 코드 또는 단항 코드의 임의의 조합을 이용할 수도 있다.

[0128] 일부 예들에서, 비디오 코더는 고정-오더 골룸 코드들, 고정-길이 코드 또는 단항 코드의 임의의 조합을 이용함으로써 먼저 c'(i) 의 절대값을 시그널링 (또는 파싱) 할 수도 있다. c'(i) 가 0 이 아닌 경우, 비디오 코더는 후속하여 (예를 들어, c'(i) 의 절대 값을 시그널링한 후) 부호 정보를 시그널링 (또는 파싱) 할 수도 있다.

[0129] 일부 예들에서, 비디오 코더는 c'(i) 를 $c''(i) = c'(i) - c'_{min}(i)$ 에 의해 0이 아닌 값으로 변환하고, 그 변환된 값을 시그널링할 수도 있으며, 여기서 $c'_{min}(i)$ 는 i 번째 계수에 대한 최소 맵핑된 값이다. 비디오 디코더는 음이 아닌 값인 c''(i) 를 파싱할 수도 있다. c''(i) 에 기초하여, 비디오 디코더는 $c'(i) = c''(i) + c'_{min}(i)$ 를 계산할 수도 있다.

[0130] 본 개시의 제 2 기법에 따르면, 비디오 코더는 비용 승수를 감소시키기 위해 크로스-컴포넌트 적응형 루프 필터들 (510, 512)에 대한 필터 계수들의 다이내믹 레인지 (dynamic range) 를 제한하도록 구성될 수도 있다. k 를 계수의 소수점 부분을 나타내는 데 사용되는 비트 수라고 하자. 필터 계수 c(i) 의 다이내믹 레인지는 오픈 인터벌 (-(1<<(k-j)), (1<<(k-j))-1) 에서 제한될 수도 있다. 비디오 코더는 고정-오더 골룸 코드들, 고정-길이 코드들, 및/또는 단항 코드들의 임의의 조합을 사용하여 c(i) 를 시그널링할 수도 있다. 비디오 코더는 c(i) 의 절대값을 먼저 시그널링 (또는 파싱) 할 수도 있다. c 가 0 이 아닌 경우, 비디오 코더는 후속하여 c(i) 에 대한 부호 정보를 시그널링 (파싱) 할 수도 있다. 추가적으로 또는 대안적으로, 비디오 코더는 c(i) 를 $c'(i) = c(i) - c_{min}(i)$ 에 의해 0이 아닌 값으로 변환할 수도 있다. 비디오 코더는 그 후 변환된 값을 시그널링할 수도 있고, 여기서 $c_{min}(i)$ 는 i 번째 계수에 대한 min 값이다. 비디오 디코더는 음이 아닌 값인 c'(i) 를 파싱할 수도 있다. 비디오 디코더는 c(i) 의 값을 $c'(i) + c_{min}(i)$ 로서 계산할 수도 있다.

[0131] 도 6 은 본 개시의 기법들에 따른, 현재 블록을 인코딩하기 위한 예시적인 방법을 나타내는 플로우차트이다. 현재 블록은 현재 CU 를 포함할 수도 있다. 비디오 인코더 (200)(도 1 및 도 3) 와 관련하여 설명되지만, 도 6 의 것과 유사한 방법을 수행하도록 다른 디바이스들이 구성될 수도 있음을 이해해야 한다.

[0132] 본 예에서, 비디오 인코더 (200) 는 초기에 현재 블록을 예측한다 (350). 예를 들어, 비디오 인코더 (200) 는 현재 블록에 대한 예측 블록을 형성할 수도 있다. 그 다음, 비디오 인코더 (200) 는 현재 블록에 대한 잔차 블록을 계산할 수도 있다 (352). 잔차 블록을 계산하기 위해, 비디오 인코더 (200) 는 오리지널의 코딩되지 않은 블록과 현재 블록에 대한 예측 블록 사이의 차이를 계산할 수도 있다. 비디오 인코더 (200) 는 그 후 잔차 블록의 계수들을 변환하고 양자화할 수도 있다 (354). 다음으로, 비디오 인코더 (200) 는 잔차 블록의 양자화된 변환 계수들을 스캔할 수도 있다 (356). 스캔 동안 또는 스캔에 후속하여, 비디오 인코더 (200) 는 계수들을 엔트로피 인코딩할 수도 있다 (358). 예를 들어, 비디오 인코더 (200) 는 CAVLC 또는

CABAC 를 사용하여 계수들을 인코딩할 수도 있다. 비디오 인코더 (200) 는 그 후 블록의 엔트로피 인코딩된 데이터를 출력할 수도 있다 (360).

[0133] 비디오 인코더 (200) 는 그 후 현재 블록을 디코딩할 수도 있다 (362). 예를 들어, 비디오 인코더 (200) 는 양자화된 변환 계수들을 역 양자화 및 역 변환하여 잔차 블록을 재생성하고, 재생성된 잔차 블록을 예측 블록과 결합할 수도 있다. 비디오 인코더 (200) 는 그 후, 예를 들어, 본 개시에 따른 크로스-컴포넌트 적응형 루프 필터링 기법들을 사용하여, 디코딩된 블록을 필터링할 수도 있다 (364). 블록의 엔트로피 인코딩된 데이터는, 예를 들어, 어느 크로스-컴포넌트 적응형 루프 필터들이 블록에 대해 선택되는지를 표시하는 필터 인덱스들을 더 포함할 수도 있다. 비디오 인코더 (200) 는 그 후, 예를 들어, 인코딩될 (및 디코딩될) 장래의 블록을 예측할 때 참조를 위해 필터링된 블록을 저장할 수도 있다 (366).

[0134] 도 7 은 본 개시의 기법들에 따른, 현재 블록을 디코딩하기 위한 예시적인 방법을 나타내는 플로우차트이다. 현재 블록은 현재 CU 를 포함할 수도 있다. 비디오 디코더 (300)(도 1 및 도 4) 와 관련하여 설명되지만, 도 7 과 유사한 방법을 수행하도록 다른 디바이스들이 구성될 수도 있음을 이해해야 한다.

[0135] 비디오 디코더 (300) 는 현재 블록에 대응하는 잔차 블록의 계수들에 대한 엔트로피 인코딩된 예측 정보 및 엔트로피 인코딩된 데이터, 및 현재 블록에 대한 크로스-컴포넌트 적응형 루프 필터 정보와 같은, 현재 블록에 대한 엔트로피 인코딩된 데이터를 수신할 수도 있다 (370). 비디오 디코더 (300) 는 엔트로피 인코딩된 데이터를 엔트로피 디코딩하여 현재 블록에 대한 예측 정보를 결정하고 잔차 블록의 계수들을 재생성할 수도 있다 (372). 비디오 디코더 (300) 는 현재 블록에 대한 예측 블록을 계산하기 위해, 예를 들어 현재 블록에 대한 예측 정보에 의해 표시된 바와 같이 인트라- 또는 인터-예측 모드를 사용하여, 현재 블록을 예측할 수도 있다 (374). 비디오 디코더 (300) 는 그 후 양자화된 변환 계수들의 블록을 생성하기 위해 재생성된 계수들을 역 스캔할 수도 있다 (376). 비디오 디코더 (300) 는 그 후, 잔차 블록을 생성하기 위해 계수들을 역 양자화 및 역 변환할 수도 있다 (378). 비디오 디코더 (300) 는 예측 블록 및 잔차 블록을 조합함으로써 종국적으로 현재 블록을 디코딩할 수도 있다 (380).

[0136] 또한, 비디오 디코더 (300) 는, 예를 들어, 본 개시의 기법들 중 임의의 기법에 따른 크로스-컴포넌트 적응형 루프 필터링을 사용하여, 디코딩된 블록을 필터링할 수도 있다 (382). 비디오 디코더 (300) 는 그 후, 예를 들어, 디코딩될 장래 블록을 예측할 때 참조를 위해 필터링된 블록을 저장할 수도 있다 (384).

[0137] 도 8 은 본 개시의 하나 이상의 기법들에 따른, 현재 블록에 대한 크로스-컴포넌트 적응형 루프 필터링 (CC-ALF) 을 위한 예시의 방법을 나타내는 플로우챠트이다. 현재 블록은 현재 CU 를 포함할 수도 있다. 비디오 디코더 (300)(도 1 및 도 4) 와 관련하여 설명되지만, 도 8 과 유사한 방법을 수행하도록 다른 디바이스들이 구성될 수도 있음을 이해해야 한다.

[0138] 비디오 디코더 (300) 는 크로스-컴포넌트 적응형 루프 필터의 복수의 필터 계수들을 디코딩할 수도 있다 (802). 예를 들어, 복수의 필터 계수들 중 특정 필터 계수를 디코딩하기 위해, 엔트로피 디코딩 유닛 (302) 은 인코딩된 비디오 비트스트림으로부터, 로그 베이스 2 의 특정 필터 계수의 절대 값을 표현하는 지수 값을 2 의 상기 지수 값의 거듭제곱으로서 특정하는 신택스 엘리먼트를 디코딩할 수도 있다. 지수 값이 0이 아닌 경우 (즉, 0 이외의 값을 갖는 경우), 엔트로피 디코딩 유닛 (302) 은 인코딩된 비디오 비트스트림으로부터, 특정 필터 계수의 부호를 특정하는 신택스 엘리먼트를 디코딩할 수도 있다. 엔트로피 디코딩 유닛 (302) 은 지수 값 (및 존재하는 경우 부호 값)에 기초하여 특정 필터 계수의 값을 결정할 수도 있다. 예를 들어, 엔트로피 디코딩 유닛 (302) 은 다음의 식에 따라 특정 필터 계수의 값을 결정할 수도 있다:

[0139] $$c(i) = sign(i) * 2^{c'(i)+1}$$

[0140] 여기서, $c(i)$ 는 특정 필터 계수의 값이고, $sign(i)$ 는 부호가 음인 경우 -1 이고 부호가 양인 경우 +1 이며, $c'(i)$ 는 특정 필터 계수에 대한 지수 값이다.

[0141] 비디오 디코더 (300)는 비디오 데이터의 블록의 샘플들을 재구성할 수도 있다 (804). 예를 들어, 비디오 디코더 (300) 는 도 7 을 참조하여 상기 설명된 바와 같이 샘플들을 재구성할 수도 있다. 일 예로서, 비디오 디코더 (300) 는 블록의 샘플들을 재구성하기 위해 잔차 데이터와 함께 예측자 블록의 샘플들을 부가할 수도 있

다.

[0142] 비디오 디코더 (300) 는 비디오 데이터의 블록에 대해, 복수의 필터 계수들에 기초하여, 크로스-컴포넌트 적응형 루프 필터링을 수행할 수도 있다 (806). 예를 들어, 위에서 논의된 바와 같이, 필터 유닛 (312) 의 CC ALF Cb 필터 및 CC ALF Cr 필터 (예를 들어, CC ALF Cb 필터 (510) 및 CC ALF Cr 필터 (512)) 는 복수의 필터 계수들의 값들에 기초하여, 곱셈을 수행함이 없이 비디오 데이터의 블록의 샘플들을 비트-시프트함으로써 향상 크로마 블록들을 생성할 수도 있다. 이러한 방식으로, 본 개시의 기법들은 CC-ALF 를 수행하기 위해 요구되는 시스템 리소스들을 감소시킬 수도 있다.

[0143] 다음의 넘버링된 예들은 본 개시의 하나 이상의 양태들을 예시할 수도 있다:

[0144] 예 1. 비디오 데이터를 디코딩하는 방법으로서, 상기 방법은: 크로스-컴포넌트 적응형 루프 필터의 복수의 필터 계수들을 코딩하는 단계로서, 상기 복수의 필터 계수들 중 하나 이상의 필터 계수들의 값들은 제로 또는 2의 거듭제곱(power)으로 제한되는, 상기 코딩하는 단계; 상기 복수의 필터 계수들 중 하나 이상의 필터 계수들의 값들이 제로 또는 2의 거듭제곱으로 제한되는 비트-시프팅하는 단계; 비디오 데이터의 블록을 코딩하는 단계; 및, 상기 필터 계수들을 사용하여 상기 디코딩된 블록의 크로스-컴포넌트 적응형 루프 필터링을 수행하는 단계를 포함하는, 비디오 데이터를 디코딩하는 방법.

[0145] 예 2. 예 1 의 방법에 있어서, 상기 복수의 필터 계수들 모두의 값들은 제로 또는 2 의 거듭제곱으로 제한된다.

[0146] 예 3. 예 1 의 방법에 있어서, 상기 복수의 필터 계수들 중 적어도 하나의 값은 제로 또는 2의 거듭제곱으로 제한되지 않는다.

[0147] 예 4. 예 1 내지 예 3 중 어느 것의 방법에 있어서, 상기 크로스-컴포넌트 적응형 루프 필터링을 수행하는 단계는, 제로 또는 2 의 거듭제곱들의 값들을 갖는 필터 계수들을 디코딩된 블록의 샘플들에 의해 곱하지 않는 단계를 포함한다.

[0148] 예 5. 예 1 내지 예 4 중 어느 것의 방법에 있어서, 제한되는 상기 복수의 필터 계수들의 필터 계수들의 값들을 나타내는 하나 이상의 신택스 엘리먼트들을 코딩하는 단계를 더 포함한다.

[0149] 예 6. 비디오 데이터를 디코딩하는 방법으로서, 상기 방법은: 크로스-컴포넌트 적응형 루프 필터의 필터 계수에 대한 십진 값을 나타내는데 사용되는 비트 수 k 를 결정하는 단계; 상기 필터 계수의 다이내믹 레인지가 (-(1<<(k - j)), (1 << (k - j)) - 1) 를 포함하는 것을 결정하는 단계; 비디오 데이터의 블록을 코딩하는 단계; 및, 상기 필터 계수를 사용하여, 디코딩된 블록의 크로스-컴포넌트 적응형 루프 필터링을 수행하는 단계를 포함하는, 비디오 데이터를 디코딩하는 방법.

[0150] 예 7. 예 1 내지 예 6 중 어느 것의 방법에 있어서, 코딩은 디코딩을 포함한다.

[0151] 예 8. 예 1 내지 예 7 중 어느 것의 방법에 있어서, 코딩은 인코딩을 포함한다.

[0152] 예 9. 비디오 데이터를 코딩하기 위한 디바이스로서, 상기 디바이스는, 예들 1-8 중 어느 것의 방법을 수행하기 위한 하나 이상의 수단들을 포함하는, 비디오 데이터를 코딩하기 위한 디바이스.

[0153] 예 10. 예 9 의 디바이스에 있어서, 상기 하나 이상의 수단들은 회로에서 구현되는 하나 이상의 프로세서들을 포함한다.

[0154] 예 11. 예 9 및 예 10 중 어느 것의 디바이스에 있어서, 비디오 데이터를 저장하기 위한 메모리를 더 포함한다.

[0155] 예 12. 예 9 내지 예 11 중 어느 것의 디바이스에 있어서, 디코딩된 비디오 데이터를 디스플레이하도록 구성된 디스플레이를 더 포함한다.

[0156] 예 13. 예 9 내지 예 12 중 어느 것의 디바이스에 있어서, 상기 디바이스는 카메라, 컴퓨터, 모바일 디바이스, 브로드캐스트 수신기 디바이스, 또는 셋-톱 박스 중 하나 이상을 포함한다.

[0157] 예 14. 예 9 내지 예 13 중 어느 것의 디바이스에 있어서, 상기 디바이스는 비디오 디코더를 포함한다.

[0158] 예 15. 예 9 내지 예 14 중 어느 것의 디바이스에 있어서, 상기 디바이스는 비디오 인코더를 포함한다.

[0159] 예 16. 명령들이 저장된 컴퓨터 판독가능 저장 매체로서, 상기 명령들은, 실행될 때, 하나 이상의 프로세서들로 하여금 예들 1-8 중 어느 것의 방법을 수행하게 하는, 컴퓨터 판독가능 저장 매체.

[0160] 예에 의존하여, 본 명세서에서 설명된 기법들의 임의의 것의 특정 행위들 또는 이벤트들은 상이한 시퀀스로 수

행될 수도 있고, 전체적으로 부가되거나 병합되거나 또는 제거될 수도 있음 (예를 들어, 설명된 모든 행위들 또는 이벤트들이 그 기법들의 실시를 위해 필수적인 것은 아님) 이 인식되어야 한다. 더욱이, 특정 예들에 있어서, 행위들 또는 이벤트들은 순차적인 것보다는, 예를 들어, 다중-스레딩된 프로세싱, 인터럽트 프로세싱, 또는 다중의 프로세서들을 통해 동시에 수행될 수도 있다.

[0161] 하나 이상의 예들에서, 설명된 기능들은 하드웨어, 소프트웨어, 펌웨어, 또는 그 임의의 조합으로 구현될 수도 있다. 소프트웨어로 구현되면, 그 기능들은 컴퓨터 판독가능 매체 상의 하나 이상의 명령 또는 코드로서 저장되거나 송신될 수도 있고 하드웨어 기반 프로세싱 유닛에 의해 실행될 수도 있다. 컴퓨터 판독가능 매체는, 데이터 저장 매체와 같은 유형의 매체에 대응하는 컴퓨터 판독가능 저장 매체, 또는 예를 들어, 통신 프로토콜에 따라, 일 장소로부터 다른 장소로의 컴퓨터 프로그램의 전송을 용이하게 하는 임의의 매체를 포함하는 통신 매체를 포함할 수도 있다. 이러한 방식으로, 컴퓨터 판독가능 매체들은 일반적으로 (1) 비일시적인 유형의 컴퓨터 판독가능 저장 매체들 또는 (2) 신호 또는 캐리어파와 같은 통신 매체에 대응할 수도 있다. 데이터 저장 매체들은 본 개시에서 설명된 기법들의 구현을 위한 명령들, 코드 및/또는 데이터 구조들을 취출하기 위해 하나 이상의 컴퓨터들 또는 하나 이상의 프로세서들에 의해 액세스될 수도 있는 임의의 가용 매체들일 수도 있다. 컴퓨터 프로그램 제품이 컴퓨터 판독가능 매체를 포함할 수도 있다.

[0162] 제한이 아닌 예로서, 이러한 컴퓨터 판독가능 저장 매체들은 RAM, ROM, EEPROM, CD-ROM 또는 다른 광학 디스크 스토리지, 자기 디스크 스토리지, 또는 다른 자기 저장 디바이스들, 플래시 메모리, 또는 명령들 또는 데이터 구조들의 형태로 원하는 프로그램 코드를 저장하는데 사용될 수도 있고 컴퓨터에 의해 액세스될 수도 있는 임의의 다른 매체를 포함할 수도 있다. 또한, 임의의 커넥션이 컴퓨터 판독가능 매체로 적절히 명명된다. 예를 들어, 동축 케이블, 광섬유 케이블, 꼬임쌍선, 디지털 가입자 라인 (DSL), 또는 적외선, 무선, 및 마이크로파와 같은 무선 기술들을 이용하여 웹사이트, 서버, 또는 다른 원격 소스로부터 소프트웨어가 송신되는 경우, 그 동축 케이블, 광섬유 케이블, 꼬임쌍선, DSL, 또는 적외선, 무선, 및 마이크로파와 같은 무선 기술들은 매체의 정의에 포함된다. 하지만, 컴퓨터 판독가능 저장 매체들 및 데이터 저장 매체들은 커넥션들, 캐리어 파들, 신호들 또는 다른 일시적 매체들을 포함하는 것이 아니라, 대신에 비일시적, 유형의 저장 매체에 관련된다는 것이 이해되야 한다. 디스크 (disk) 및 디스크 (disc) 는, 본 명세서에서 사용된 바와 같이, 콤팩트 디스크 (CD), 레이저 디스크, 광학 디스크, 디지털 다용도 디스크 (DVD), 플로피 디스크 및 블루-레이 디스크를 포함하고, 여기서 디스크 (disk) 들은 보통 데이터를 자기적으로 재생하는 한편, 디스크 (disc) 들은 레이저들로 데이터를 광학적으로 재생한다. 상기의 조합들이 또한, 컴퓨터 판독가능 매체들의 범위 내에 포함되어야 한다.

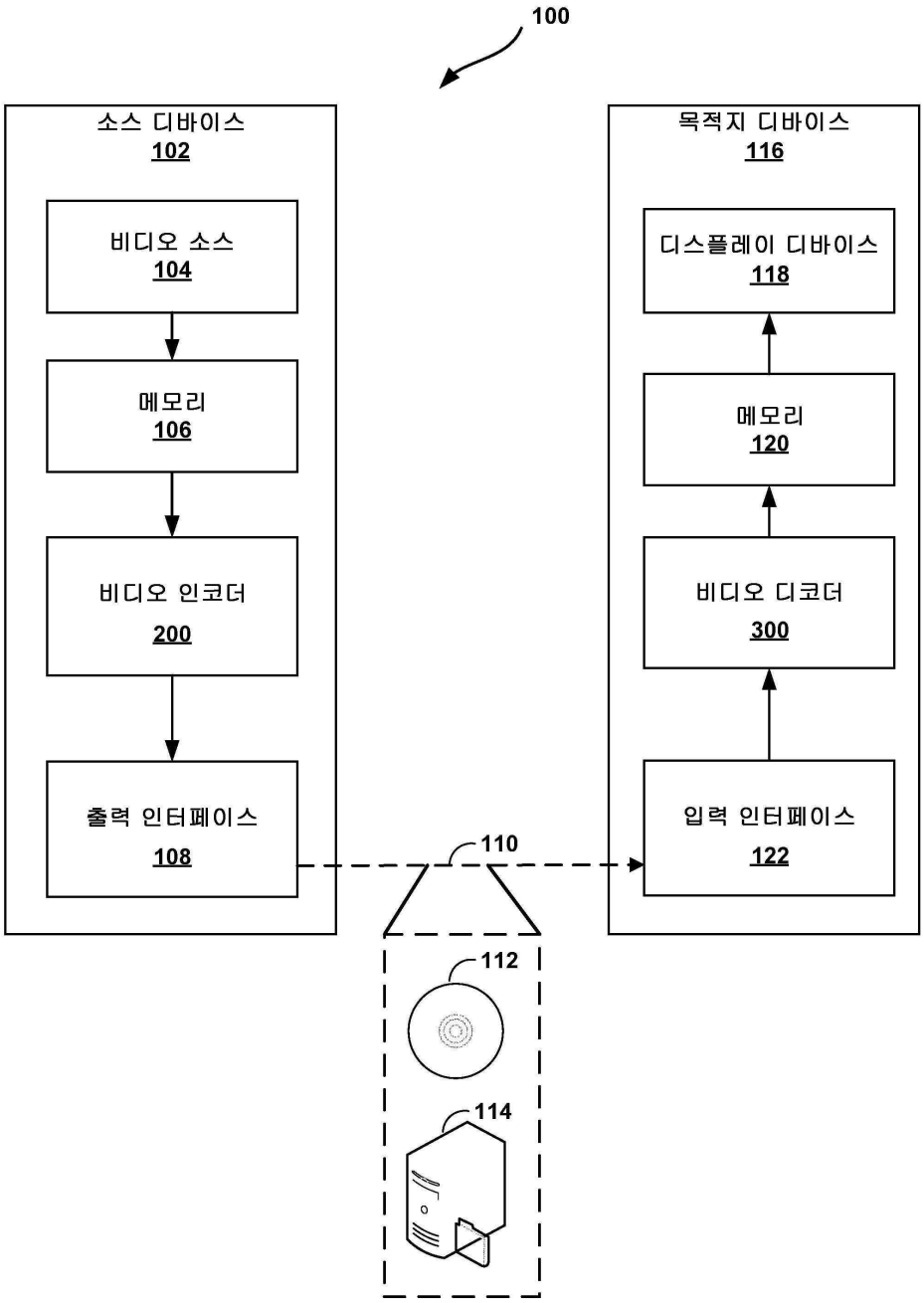
[0163] 명령들은 하나 이상의 프로세서, 이를테면 하나 이상의 DSP (digital signal processor), 범용 마이크로프로세서, ASIC (application specific integrated circuit), FPGA (field programmable logic array), 또는 다른 등가 집적 또는 이산 로직 회로에 의해 실행될 수도 있다. 이에 따라, 본 명세서에서 사용된 바와 같은 용어들 "프로세서" 및 "프로세싱 회로" 는 전술한 구조들 또는 본 명세서에서 설명된 기법들의 구현에 적합한 임의의 다른 구조 중 임의의 것을 지칭할 수도 있다. 부가적으로, 일부 양태들에 있어서, 본 명세서에서 설명된 기능성은 인코딩 및 디코딩을 위해 구성되거나 또는 결합된 코덱에서 통합된 전용 하드웨어 및/또는 소프트웨어 모듈들 내에 제공될 수도 있다. 또한, 그 기법들은 하나 이상의 회로들 또는 로직 엘리먼트들에서 완전히 구현될 수도 있다.

[0164] 본 개시의 기법들은 무선 핸드셋, 집적 회로 (IC) 또는 IC들의 세트 (예를 들어, 칩 세트) 를 포함하여, 광범위한 디바이스들 또는 장치들에서 구현될 수도 있다. 다양한 컴포넌트들, 모듈들, 또는 유닛들은 개시된 기법들을 수행하도록 구성된 디바이스들의 기능적 양태들을 강조하기 위해 본 개시에 설명되지만, 상이한 하드웨어 유닛들에 의한 실현을 반드시 필요로 하는 것은 아니다. 오히려, 상기 설명된 바와 같이, 다양한 유닛들은 코덱 하드웨어 유닛에서 결합되거나 또는 적합한 소프트웨어 및/또는 펌웨어와 함께, 상기 설명된 바와 같은 하나 이상의 프로세서들을 포함하는, 상호동작가능한 하드웨어 유닛들의 콜렉션에 의해 제공될 수도 있다.

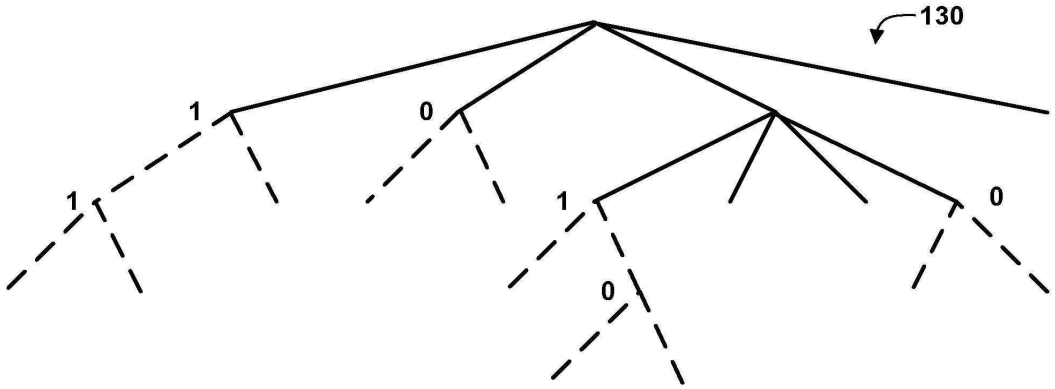
[0165] 다양한 예들이 설명되었다. 이들 및 다른 예들은 다음의 청구항들의 범위 내에 있다.

도면

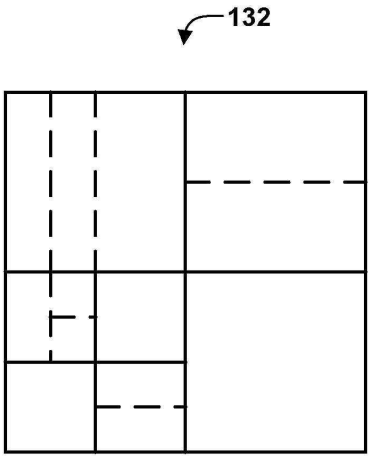
도면1


100
소스 디바이스
102
비디오 소스
104
메모리
106
비디오 인코더
200
출력 인터페이스
108
110
112
114
목적지 디바이스
116
디스플레이 디바이스
118
메모리
120
비디오 디코더
300
입력 인터페이스
122

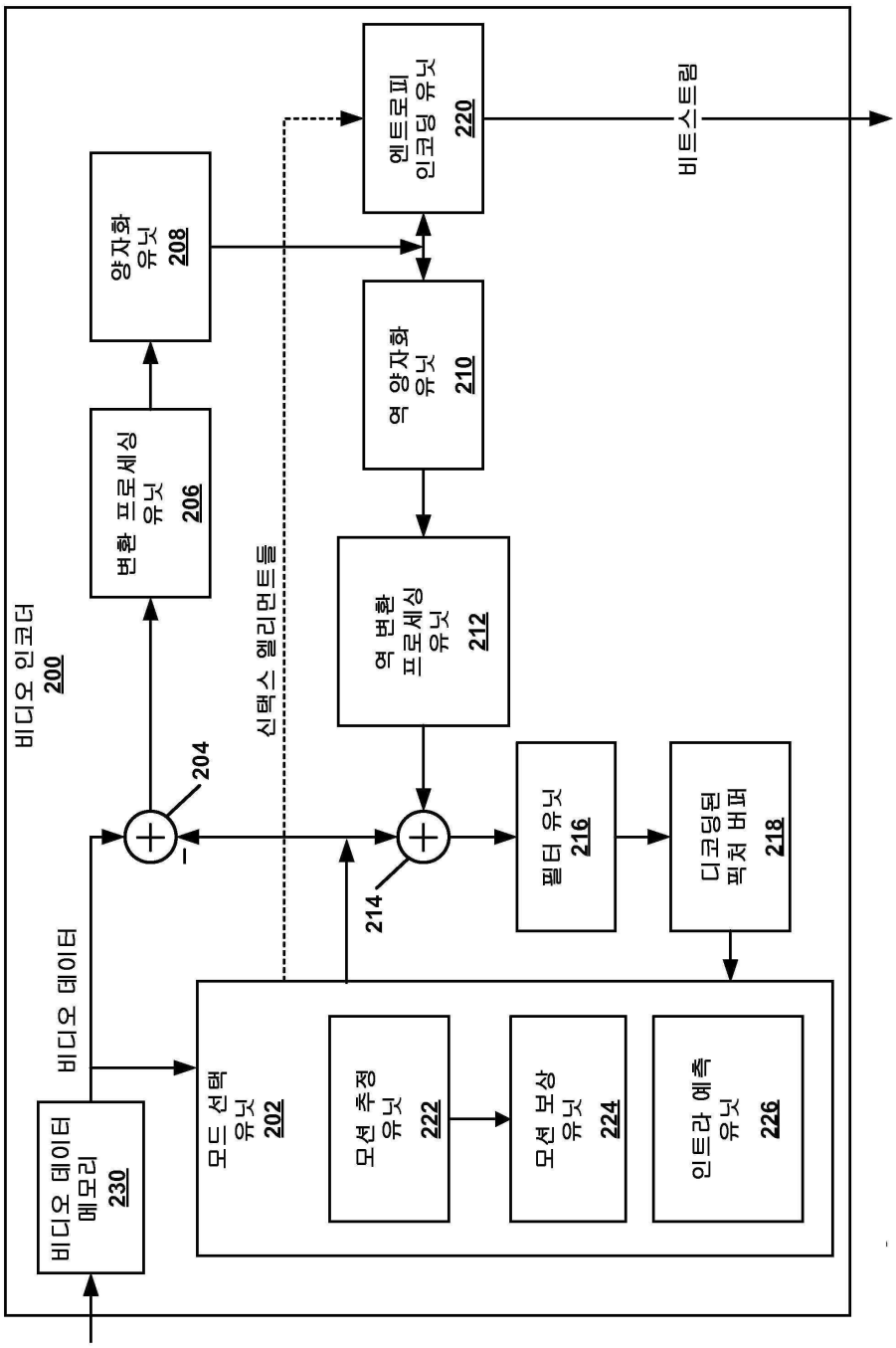
도면2a



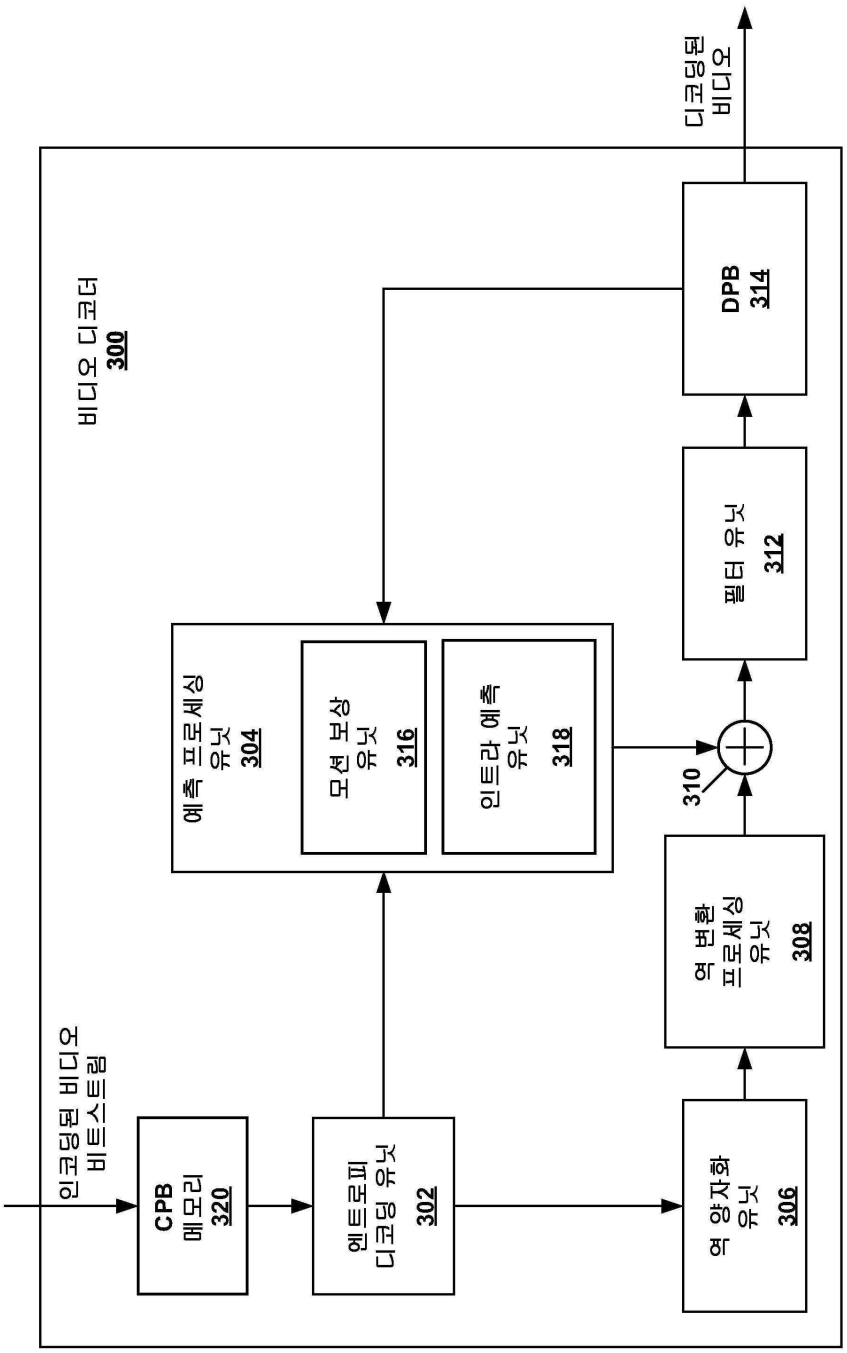
도면2b



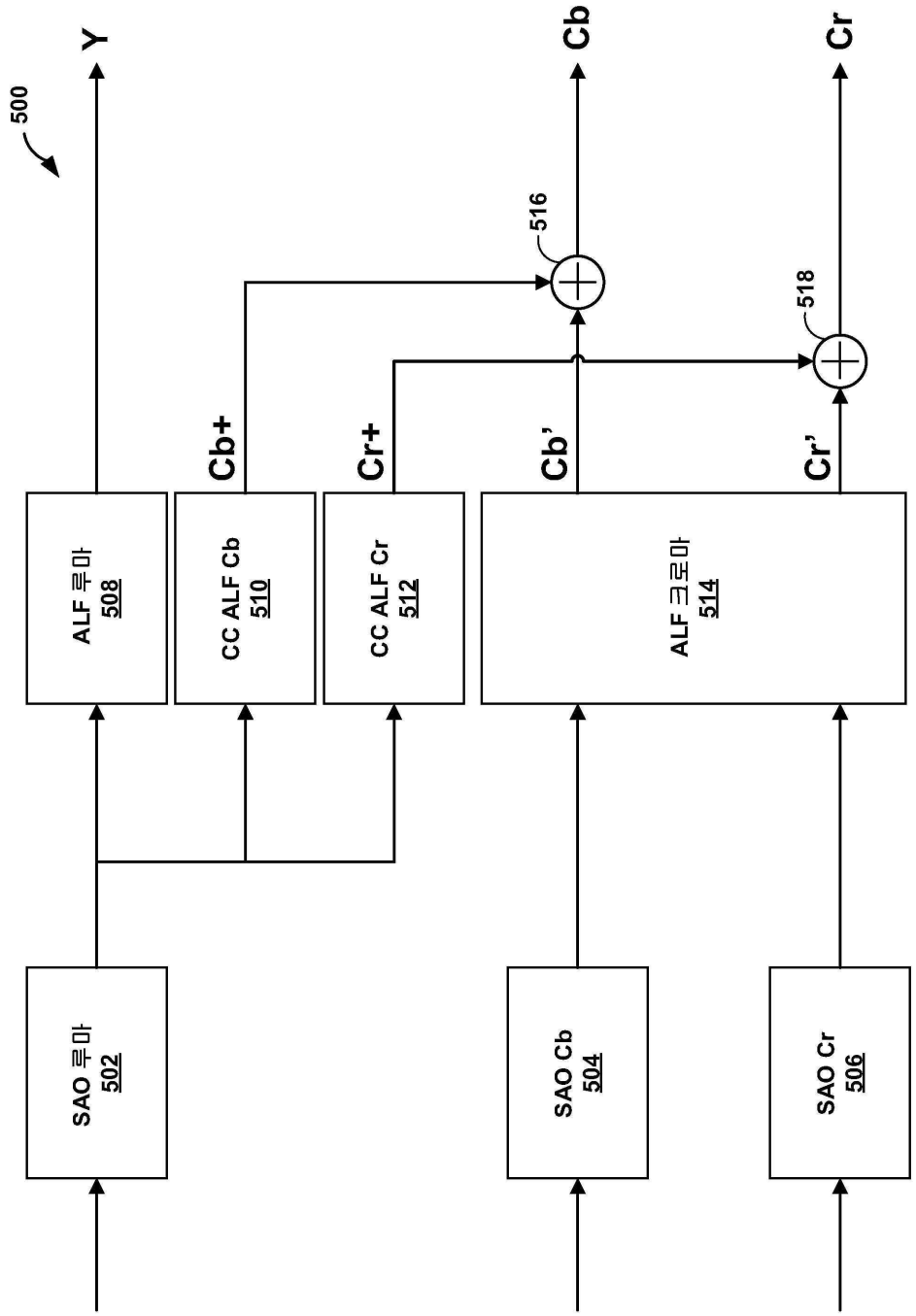
도면3


비디오 인코더
200
비디오 데이터
비디오 데이터 메모리
230
모드 선택 유닛
202
모션 추정 유닛
222
모션 보상 유닛
224
인트라 예측 유닛
226
204
214
신택스 엘리먼트들
변환 프로세싱 유닛
206
양자화 유닛
208
역 양자화 유닛
210
역 변환 프로세싱 유닛
212
필터 유닛
216
디코딩된 픽처 버퍼
218
엔트로피 인코딩 유닛
220
비트스트림

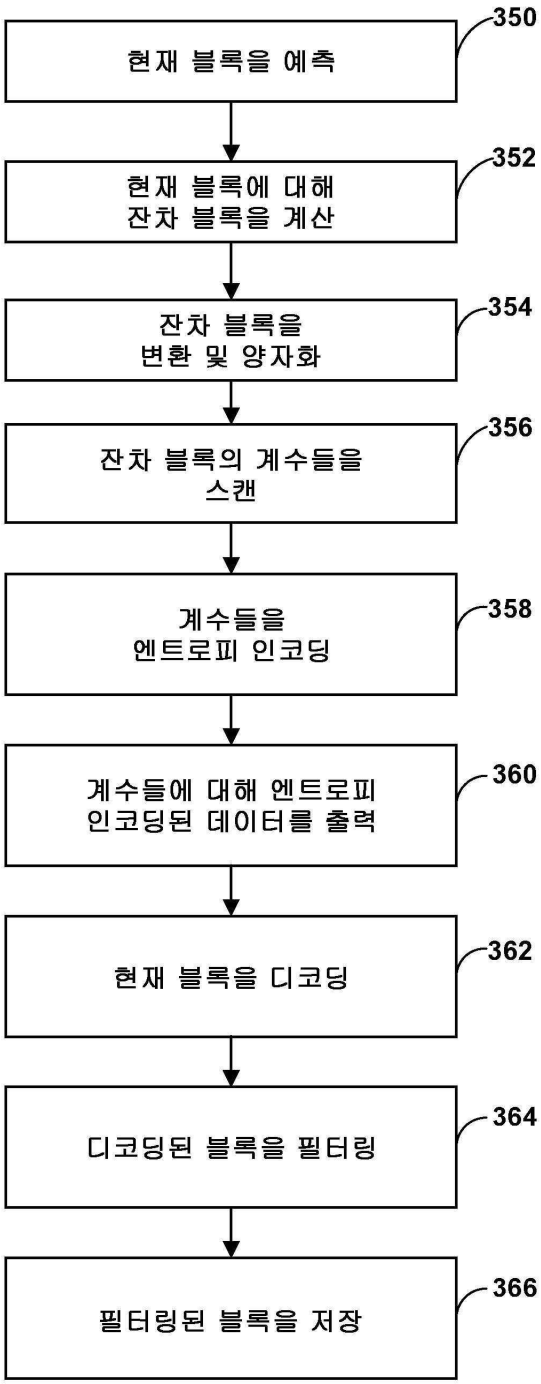
도면4


디코딩된
비디오
비디오 디코더
300
DPB
314
필터 유닛
312
예측 프로세싱
유닛
304
모션 보상
유닛
316
인트라 예측
유닛
318
310
역 변환
프로세싱
유닛
308
역 양자화
유닛
306
엔트로피
디코딩 유닛
302
CPB
메모리
320
인코딩된 비디오
비트스트림

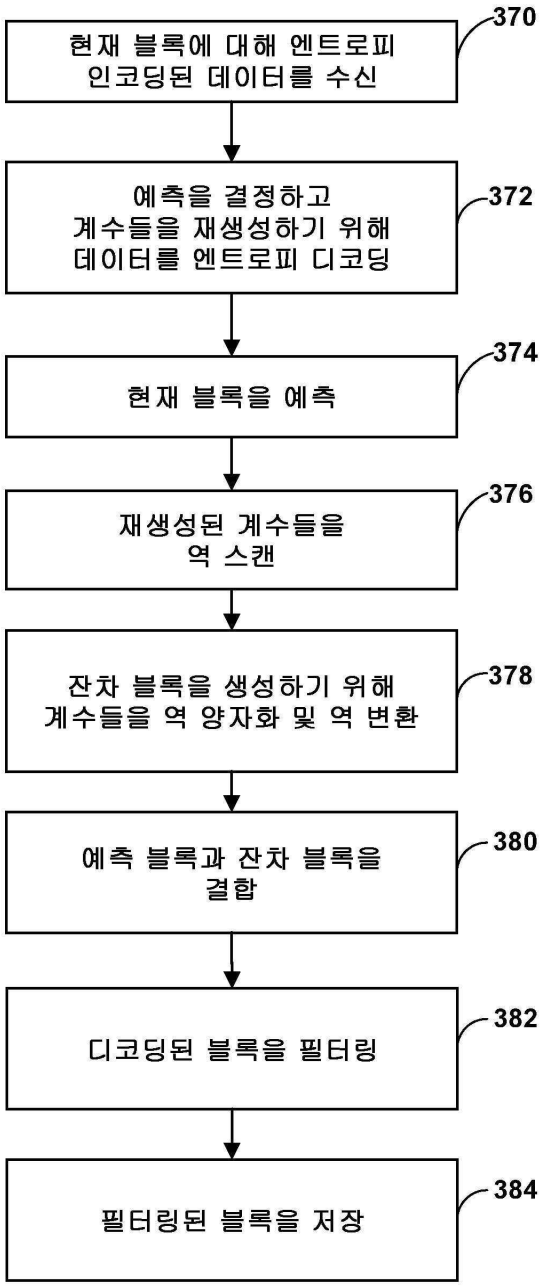
도면5


500
Y
Cb
Cr
516
518
Cb+
Cr+
Cb'
Cr'
ALF 루마
508
CC ALF Cb
510
CC ALF Cr
512
ALF 크로마
514
SAO 루마
502
SAO Cb
504
SAO Cr
506

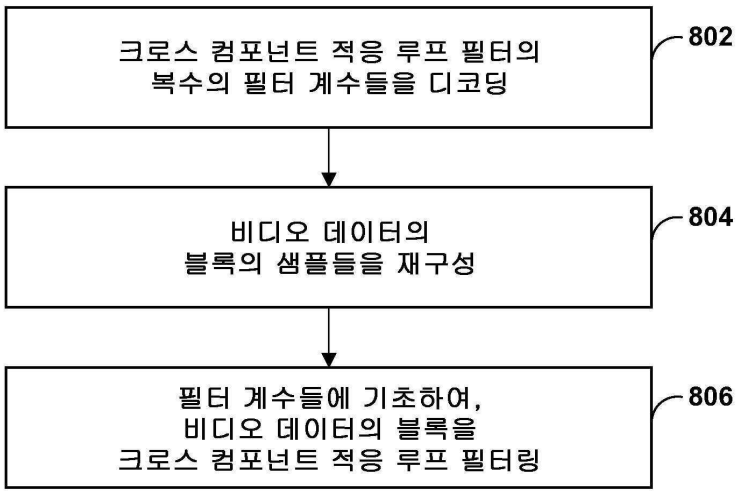
도면6


현재 블록을 예측
350
현재 블록에 대해
잔차 블록을 계산
352
잔차 블록을
변환 및 양자화
354
잔차 블록의 계수들을
스캔
356
계수들을
엔트로피 인코딩
358
계수들에 대해 엔트로피
인코딩된 데이터를 출력
360
현재 블록을 디코딩
362
디코딩된 블록을 필터링
364
필터링된 블록을 저장
366

도면7


현재 블록에 대해 엔트로피 인코딩된 데이터를 수신
370
예측을 결정하고 계수들을 재생성하기 위해 데이터를 엔트로피 디코딩
372
현재 블록을 예측
374
재생성된 계수들을 역 스캔
376
잔차 블록을 생성하기 위해 계수들을 역 양자화 및 역 변환
378
예측 블록과 잔차 블록을 결합
380
디코딩된 블록을 필터링
382
필터링된 블록을 저장
384

도면8


크로스 컴포넌트 적응 루프 필터의
복수의 필터 계수들을 디코딩
802
비디오 데이터의
블록의 샘플들을 재구성
804
필터 계수들에 기초하여,
비디오 데이터의 블록을
크로스 컴포넌트 적응 루프 필터링
806