(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2025-0096879
(43) 공개일자 2025년06월27일

(51) 국제특허분류(Int. Cl.)
H04N 19/132 (2014.01) *H04N 19/105* (2014.01)
H04N 19/119 (2014.01) *H04N 19/157* (2014.01)
H04N 19/186 (2014.01) *H04N 19/423* (2014.01)
(52) CPC특허분류
H04N 19/132 (2015.01)
H04N 19/105 (2015.01)
(21) 출원번호 10-2025-7020365(분할)
(22) 출원일자(국제) 2019년11월29일
심사청구일자 2025년06월18일
(62) 원출원 특허 10-2023-7028951
원출원일자(국제) 2019년11월29일
심사청구일자 2023년08월24일
(85) 번역문제출일자 2025년06월18일
(86) 국제출원번호 PCT/US2019/063845
(87) 국제공개번호 WO 2020/113156
국제공개일자 2020년06월04일
(30) 우선권주장
62/774,148 2018년11월30일 미국(US)
(뒷면에 계속)

(71) 출원인
텐센트 아메리카 엘엘씨
미국 94306 캘리포니아주 팔로 알토 파크 블러바드 2747
(72) 발명자
쉬, 샤오중
미국 94306 캘리포니아주 팔로 알토 파크 블러바드 2747 텐센트 아메리카 엘엘씨 내
리, 샹
미국 94306 캘리포니아주 팔로 알토 파크 블러바드 2747 텐센트 아메리카 엘엘씨 내
류, 산
미국 94306 캘리포니아주 팔로 알토 파크 블러바드 2747 텐센트 아메리카 엘엘씨 내
(74) 대리인
양영준, 김연송, 백만기

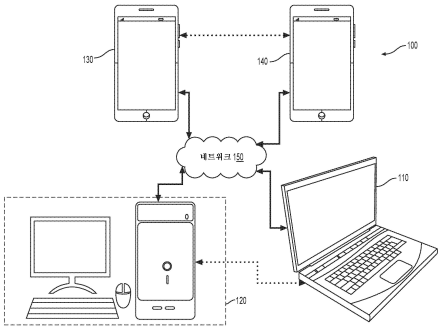
전체 청구항 수 : 총 14 항

(54) 발명의 명칭 **비디오 코딩을 위한 방법 및 장치**

(57) 요 약

본 개시내용의 양태들은 비디오 인코딩/디코딩을 위한 방법들 및 장치들을 제공한다. 일부 예들에서, 비디오 디코딩을 위한 장치는 수신 회로 및 처리 회로를 포함한다. 일부 실시예들에서, 처리 회로는 코딩된 비디오 비트스트림으로부터 현재 코딩 내의 현재 블록의 예측 정보를 디코딩한다. 예측 정보는 인트라 블록 카피 모드를 지시한다. 그 후, 처리 회로는 현재 블록과 동일한 화상 내의 참조 블록을 가리키는 블록 벡터를 결정한다. 참조 블록은 참조 샘플 메모리에 버퍼링된 재구성된 샘플들을 갖는 코딩 영역 내에서 제한된다. 코딩 영역은 코딩 트리 유닛(coding tree unit)(CTU)의 다수의 미리 정의된 영역 중 하나이다. 그 후, 처리 회로는 참조 샘플 메모리로부터 검색되는 참조 블록의 재구성된 샘플들에 기초하여 적어도 현재 블록의 샘플을 재구성한다.

대 표 도



(52) CPC특허분류
H04N 19/119 (2015.01)
H04N 19/157 (2015.01)
H04N 19/186 (2015.01)
H04N 19/423 (2015.01)

(30) 우선권주장
62/790,454 2019년01월09일 미국(US)
16/699,235 2019년11월29일 미국(US)

명 세 서

청구범위

청구항 1

디코더 내의 비디오 디코딩을 위한 방법으로서,

코딩된 비디오 비트스트림으로부터 현재 코딩 영역 내의 현재 블록의 예측 정보를 디코딩하는 단계 - 상기 예측 정보는 인트라 블록 카피 모드를 지시하고, 상기 현재 블록이 속하는 CTU가 128x128 미만의 루마 샘플들을 포함할 때, 상기 코딩 영역은 크기가 상기 CTU와 동일하고 그렇지 않으면 상기 CTU는 복수의 코딩 영역으로 분할되고, 상기 현재 코딩 영역은 상기 현재 블록이 속하는 상기 복수의 코딩 영역 중 하나의 코딩 영역임 -;

상기 현재 블록과 동일한 화상 내의 참조 블록을 가리키는 블록 벡터를 결정하는 단계 - 상기 참조 블록은 참조 샘플 메모리에 버퍼링된 재구성된 샘플들을 갖는 코딩 영역을 포함하는 검색 범위 내에 있음 -; 및

상기 참조 샘플 메모리로부터 검색되는 상기 참조 블록의 재구성된 샘플들에 기초하여 적어도 상기 현재 블록의 샘플을 재구성하는 단계

를 포함하고,

상기 검색 범위는 상기 코딩 영역의 유닛 내에서 업데이트되는,

방법.

청구항 2

제1항에 있어서,

상기 코딩 영역은 정사각형 형상을 갖는, 방법.

청구항 3

제1항에 있어서,

이동 전에 상기 참조 블록이 부분적으로 상기 코딩 영역에 있을 때, 상기 참조 블록을 상기 코딩 영역으로 이동시키는 단계를 추가로 포함하는, 방법.

청구항 4

제3항에 있어서,

상기 참조 블록과 상기 현재 블록 사이의 거리를 감소시키는 방향으로 상기 참조 블록을 이동시키는 단계를 추가로 포함하는, 방법.

청구항 5

제3항에 있어서,

상기 코딩 영역에서 참조 샘플들의 비율을 증가시키는 방향으로 상기 참조 블록을 이동시키는 단계를 추가로 포함하는, 방법.

청구항 6

제1항에 있어서,

상기 참조 블록이 부분적으로 상기 코딩 영역에 있을 때 상기 코딩 영역 내에 있는 제1 참조 샘플의 정보를 상기 코딩 영역 외부에 있는 제2 참조 샘플의 정보로서 사용하는 단계를 추가로 포함하는, 방법.

청구항 7

제1항에 있어서,

유연한 블록 분할은 상기 CTU의 분할에서 금지되는, 방법.

청구항 8

제1항에 있어서,

상기 참조 블록의 다수의 코너가 상기 코딩 영역 내에 있는지를 체크하는 단계를 추가로 포함하는, 방법.

청구항 9

비디오 디코딩 장치로서,

제1항 내지 제8항 중 어느 한 항에 따른 방법을 수행하도록 구성된 처리 회로를 포함하는, 장치.

청구항 10

비디오 인코더 내에서 수행되는 비디오 인코딩을 위한 방법으로서,

현재 코딩 영역 내의 현재 블록의 예측 정보를 코딩된 비디오 비트스트림으로 인코딩하는 단계 - 상기 예측 정보는 인트라 블록 카피 모드를 지시하고, 상기 현재 블록이 속하는 CTU가 128x128 미만의 루마 샘플들을 포함할 때, 상기 코딩 영역은 크기가 상기 CTU와 동일하고 그렇지 않으면 상기 CTU는 복수의 코딩 영역으로 분할되고, 상기 현재 코딩 영역은 상기 현재 블록이 속하는 상기 복수의 코딩 영역 중 하나의 코딩 영역임 -;

상기 현재 블록과 동일한 화상 내의 참조 블록을 가리키는 블록 벡터를 결정하는 단계 - 상기 참조 블록은 참조 샘플 메모리에 버퍼링된 재구성된 샘플들을 갖는 코딩 영역을 포함하는 검색 지역 내에 있음 -; 및

상기 참조 샘플 메모리로부터 검색되는 상기 참조 블록의 재구성된 샘플들에 기초하여 적어도 상기 현재 블록의 샘플을 재구성하는 단계

를 포함하고,

상기 검색 지역은 상기 코딩 영역의 유닛 내에서 업데이트되는,

방법.

청구항 11

비디오 인코더 내에서 수행되는 비디오 인코딩을 위한 방법으로서,

현재 코딩 영역 내의 현재 블록이 병합 모드에서 코딩되지 않을 때, 인트라 블록 카피(IBC) 모드를 지시하는 상기 현재 블록의 예측 정보를 시그널링하는 단계 - 상기 현재 블록이 속하는 CTU가 128x128 미만의 루마 샘플들을 포함할 때, 상기 코딩 영역은 크기가 상기 CTU와 동일하고 그렇지 않으면 상기 CTU는 복수의 코딩 영역으로 분할되고, 상기 현재 코딩 영역은 상기 현재 블록이 속하는 상기 복수의 코딩 영역 중 하나의 코딩 영역임 -;

상기 현재 블록과 동일한 화상 내의 참조 블록을 가리키는 블록 벡터를 결정하는 단계 - 상기 참조 블록은 참조 샘플 메모리에 버퍼링된 재구성된 샘플들을 갖는 코딩 영역을 포함하는 검색 지역 내에 있음 -; 및

상기 참조 샘플 메모리로부터 검색되는 상기 참조 블록의 재구성된 샘플들에 기초하여 적어도 상기 현재 블록의 샘플을 재구성하는 단계

를 포함하고,

상기 검색 지역은 상기 코딩 영역의 유닛 내에서 업데이트되는,

방법.

청구항 12

비디오 인코딩 장치로서,

제10항 또는 제11항에 따른 방법을 수행하도록 구성된 처리 회로를 포함하는, 장치.

청구항 13

코딩된 비디오 비트스트림을 저장 또는 송신하는 방법으로서,

제10항 또는 제11항에 따른 비디오 인코딩을 위한 방법을 수행하여, 상기 코딩된 비디오 비트스트림을 생성하는 단계를 포함하는, 방법.

청구항 14

비일시적 컴퓨터 판독가능 매체로서,

컴퓨터에 의해 실행될 때, 상기 컴퓨터로 하여금 제1항 내지 제11항 중 어느 한 항에 따른 방법을 수행하게 하는 명령어들을 저장하는 비일시적 컴퓨터 판독가능 매체.

발명의 설명

기 술 분 야

[0001] 인용에 의한 통합

[0002] 본 출원은, 2018년 11월 30일에 출원된 미국 가출원 제62/774,148호, "CONSTRAINTS ON REFERENCE BLOCK LOCATIONS FOR INTRA BLOCK COPY" 및 2019년 1월 9일에 출원된 미국 가출원 제62/790,454호, "INTRA PICTURE BLOCK COMPENSATION BUFFER REUSE WITH NON SQUARE BLOCK PARTITIONS"에 대한 우선권의 이익을 주장하는, 2019년 11월 29일에 출원된 미국 특허 출원 제16/699,235호, "METHOD AND APPARATUS FOR VIDEO CODING"의 우선권의 이익을 주장한다. 이 이전 출원들의 전체 개시내용은 이로써 그 전체가 참고로 포함된다.

[0003] 기술 분야

[0004] 본 개시내용은 일반적으로 비디오 코딩에 관련된 실시예들을 설명한다.

배 경 기 술

[0005] 본 명세서에 제공되는 배경기술 설명은, 본 개시내용의 맥락을 일반적으로 제시하기 위한 것이다. 현재 호명된 발명자들의 연구- 그 연구가 이 배경기술 부분에서 설명되는 한 - 뿐만 아니라 출원 시에 종래 기술로서의 자격이 없을 수 있는 설명의 양태들은 명백하게도 또는 암시적으로도 본 개시내용에 대한 종래 기술로서 인정되지 않는다.

[0006] 비디오 코딩 및 디코딩은 움직임 보상을 갖는 인터-화상 예측(inter-picture prediction)을 사용하여 수행될 수 있다. 압축되지 않은 디지털 비디오는 일련의 화상들을 포함할 수 있고, 각각의 화상은, 예를 들어, 1920x1080 루미넌스 샘플들 및 연관된 크로미넌스 샘플들의 공간 차원을 갖는다. 이 일련의 화상들은, 예를 들어, 초당 60개 화상 또는 60 Hz의, 고정된 또는 가변 화상 레이트(비공식적으로 프레임 레이트로도 알려져 있음)를 가질 수 있다. 압축되지 않은 비디오는 상당한 비트레이트 요건들을 갖는다. 예를 들어, 샘플당 8 비트에서의 1080p60 4:2:0 비디오(60 Hz 프레임 레이트에서의 1920x1080 루미넌스 샘플 해상도)는 1.5 Gbit/s 대역폭에 가까울 것을 요구한다. 그러한 비디오의 시간은 600 기가바이트보다 많은 저장 공간을 필요로 한다.

[0007] 비디오 코딩 및 디코딩의 하나의 목적은, 압축을 통한, 입력 비디오 신호에서의 중복성의 감소일 수 있다. 압축은 전술한 대역폭 또는 저장 공간 요건들을, 일부 경우들에서, 2 자릿수 이상 감소시키는 데 도움이 될 수 있다. 무손실 및 손실 압축 둘 다뿐만 아니라 이들의 조합이 사용될 수 있다. 무손실 압축은 압축된 원래 신호로부터 원래 신호의 정확한 카피가 재구성될 수 있는 기법들을 지칭한다. 손실 압축을 사용할 때, 재구성된 신호는 원래 신호와 동일하지 않을 수 있지만, 원래 신호와 재구성된 신호 사이의 왜곡은 재구성된 신호를 의도된 애플리케이션에 유용하게 만들 정도로 충분히 작다. 비디오의 경우, 손실 압축이 널리 사용된다. 용인되는 왜곡의 양은 애플리케이션에 의존한다; 예를 들어, 특정 소비자 스트리밍 애플리케이션들의 사용자들은 텔레비전 배포 애플리케이션들의 사용자들보다 더 높은 왜곡을 용인할 수 있다. 달성가능한 압축비는 더 높은 허용가능한/용인가능한 왜곡이 더 높은 압축비를 산출할 수 있다는 것을 반영할 수 있다.

[0008] 비디오 인코더 및 디코더는, 예를 들어, 움직임 보상, 변환, 양자화, 및 엔트로피 코딩을 포함하는, 수개의 넓은 카테고리로부터의 기법들을 사용할 수 있다.

[0009] 비디오 코덱 기법들은 인트라 코딩으로 알려진 기법들을 포함할 수 있다. 인트라 코딩에서, 샘플 값들은 이전에 재구성된 참조 화상들로부터의 샘플들 또는 다른 데이터를 참조하지 않고 표현된다. 일부 비디오 코덱들에서, 화상은 샘플들의 블록들로 공간적으로 세분된다. 샘플들의 모든 블록들이 인트라 모드에서 코딩될 때, 그

화상은 인트라 화상일 수 있다. 인트라 화상들 및 독립적인 디코더 리프레시 화상들과 같은 그들의 파생들은 디코더 상태를 리셋하기 위해 사용될 수 있고, 따라서 코딩된 비디오 비트스트림 및 비디오 세션에서의 제1 화상으로서 또는 정지 이미지로서 사용될 수 있다. 인트라 블록의 샘플들은 변환에 노출될 수 있고, 변환 계수들은 엔트로피 코딩 전에 양자화될 수 있다. 인트라 예측은 사전 변환 도메인에서 샘플 값들을 최소화하는 기법일 수 있다. 일부 경우들에서, 변환 후의 DC 값이 더 작을수록, 그리고 AC 계수들이 더 작을수록, 엔트로피 코딩 후에 블록을 표현하기 위해 주어진 양자화 스텝 사이즈에서 요구되는 비트들이 더 적다.

[0010] 예를 들면, MPEG-2세대 코딩 기법들로부터 알려진 것과 같은 전통적인 인트라 코딩은 인트라 예측을 사용하지 않는다. 그러나, 일부 더 새로운 비디오 압축 기법들은, 예를 들어, 데이터의 공간적으로 이웃하는, 그리고 디코딩 순서에서 선행하는 블록들의 인코딩/디코딩 동안 획득된 주위의 샘플 데이터 및/또는 메타데이터로부터 시도하는 기법들을 포함한다. 이러한 기법들은 이후 "인트라 예측" 기법들로 불린다. 적어도 일부 경우들에서, 인트라 예측은 참조 화상들로부터가 아니라 재구성 중인 현재 화상으로부터의 참조 데이터만을 사용한다는 점에 유의한다.

[0011] 많은 상이한 형태들의 인트라 예측이 있을 수 있다. 주어진 비디오 코딩 기법에서 그러한 기법들 중 하나보다 많은 기법이 사용될 수 있을 때, 사용 중인 기법은 인트라 예측 모드에서 코딩될 수 있다. 특정 경우들에서, 모드들은 서브모드들 및/또는 파라미터들을 가질 수 있고, 이들은 개별적으로 코딩되거나 모드 코드워드에 포함될 수 있다. 주어진 모드/서브모드/파라미터 조합에 사용할 코드워드는 인트라 예측을 통해 코딩 효율 이득에 영향을 미칠 수 있고, 따라서 코드워드들을 비트스트림으로 변환하기 위해 사용되는 엔트로피 코딩 기법에 영향을 미칠 수 있다.

[0012] 인트라 예측의 특정 모드가 H.264로 도입되었고, H.265에서 정제되고, JEM(joint exploration model), VVC(versatile video coding), 및 BMS(benchmark set)와 같은 더 새로운 코딩 기법들에서 추가로 정제되었다. 예측자 블록은 이미 이용가능한 샘플들에 속하는 이웃 샘플 값들을 사용하여 형성될 수 있다. 이웃 샘플들의 샘플 값들은 방향에 따라 예측자 블록 내로 카피된다. 사용 중인 방향에 대한 참조는 비트스트림에서 코딩될 수 있거나, 그 자체로 예측될 수 있다.

발명의 내용

[0013] 본 개시내용의 양태들은 비디오 인코딩/디코딩을 위한 방법들 및 장치들을 제공한다. 일부 예들에서, 비디오 디코딩을 위한 장치는 수신 회로 및 처리 회로를 포함한다. 일부 실시예들에서, 처리 회로는 코딩된 비디오 비트스트림으로부터 현재 코딩 내의 현재 블록의 예측 정보를 디코딩한다. 예측 정보는 인트라 블록 카피 모드를 지시한다. 그 후, 처리 회로는 현재 블록과 동일한 화상 내의 참조 블록을 가리키는 블록 벡터를 결정한다. 참조 블록은 참조 샘플 메모리에 버퍼링된 재구성된 샘플들을 갖는 코딩 영역 내에서 제한된다. 코딩 영역은 코딩 트리 유닛(coding tree unit)(CTU)의 다수의 미리 정의된 영역 중 하나이다. 그 후, 처리 회로는 참조 샘플 메모리로부터 검색되는 참조 블록의 재구성된 샘플들에 기초하여 적어도 현재 블록의 샘플을 재구성한다.

[0014] 일부 예들에서, 코딩 영역은 정사각형 형상을 갖는다. 일 예에서, CTU는 루마 컴포넌트의 128 샘플들x128 샘플들을 갖도록 구성된다.

[0015] 일 실시예에서, 참조 샘플 메모리는 코딩 영역의 유닛에서 참조 샘플 메모리에 버퍼링된 샘플들을 업데이트하도록 구성된다.

[0016] 일부 실시예들에서, 처리 회로는 참조 블록의 다수의 코너가 코딩 영역 내에 있는지를 체크한다.

[0017] 일 예에서, CTU가 128x128 루마 샘플들을 포함할 때, 코딩 영역 크기는 크기가 64x64 루마 샘플들이다. 다른 예에서, CTU가 128x128 미만의 루마 샘플들을 포함할 때, 코딩 영역은 크기가 CTU와 동일하다.

[0018] 일부 실시예들에서, 처리 회로는 참조 블록이 이동 전에 코딩 영역에 부분적으로 있을 때 참조 블록을 코딩 영역으로 이동시킨다. 일 예에서, 처리 회로는 참조 블록과 현재 블록 사이의 거리를 감소시키는 방향으로 참조 블록을 이동시킨다. 다른 예에서, 처리 회로는 코딩 영역에서 참조 샘플들의 비율을 증가시키는 방향으로 참조 블록을 이동시킨다.

[0019] 일부 실시예들에서, 처리 회로는, 참조 블록이 코딩 영역 내에 부분적으로 있을 때 코딩 영역 내에 있는 제1 참조 샘플의 정보를 코딩 영역 외부에 있는 제2 참조 샘플의 정보로서 사용한다. 일부 예들에서, 유연한 블록 분할(flexible block partition)은 CTU의 분할에서 금지된다.

[0020] 본 개시내용의 양태들은 또한, 비디오 디코딩을 위해 컴퓨터에 의해 실행될 때, 컴퓨터로 하여금 비디오 디코딩을 위한 방법을 수행하게 하는 명령어들을 저장하는 비일시적 컴퓨터 판독가능 매체를 제공한다.

도면의 간단한 설명

[0021] 개시된 주제의 추가의 특징들, 본질 및 다양한 이점들이 다음의 상세한 설명 및 첨부 도면들로부터 더 명백할 것이고, 이 도면들에서:

도 1은 일 실시예에 따른 통신 시스템(100)의 단순화된 블록도의 개략 예시이다.

도 2는 일 실시예에 따른 통신 시스템(200)의 단순화된 블록도의 개략 예시이다.

도 3은 일 실시예에 따른 디코더의 단순화된 블록도의 개략 예시이다.

도 4는 일 실시예에 따른 인코더의 단순화된 블록도의 개략 예시이다.

도 5는 다른 실시예에 따른 인코더의 블록도를 도시한다.

도 6은 다른 실시예에 따른 디코더의 블록도를 도시한다.

도 7은 본 개시내용의 일 실시예에 따른 인트라 블록 카피의 예를 도시한다.

도 8a 내지 도 8d는 본 개시내용의 실시예에 따른 참조 샘플 메모리 업데이트들 및 인트라 블록 카피 모드에 대한 유효 검색 범위들의 예들을 도시한다.

도 9a 내지 도 9h는 비-정사각형 블록들을 초래할 수 있는 분할들의 일부 예들을 도시한다.

도 10a 및 도 10b는 코딩 순서들 및 참조 샘플 메모리 사용의 2가지 예를 도시한다.

도 11a 및 도 11b는 2개의 상이한 코딩 영역에서 나오는 참조 블록 내의 샘플들의 예들을 도시한다.

도 12a 내지 도 12f는 참조 샘플 메모리 재사용 전략들의 예들을 도시한다.

도 13a 내지 도 13f는 본 개시내용의 일부 실시예들에 따른 업데이트 프로세스의 예들을 도시한다.

도 14는 본 개시내용의 일부 실시예들에 따른 프로세스 예를 약술하는 흐름도를 도시한다.

도 15는 일 실시예에 따른 컴퓨터 시스템의 개략 예시이다.

발명을 실시하기 위한 구체적인 내용

[0022] 도 1은 본 개시내용의 일 실시예에 따른 통신 시스템(100)의 단순화된 블록도를 도시한다. 통신 시스템(100)은, 예를 들어, 네트워크(150)를 통해, 서로 통신할 수 있는 복수의 단말 디바이스를 포함한다. 예를 들어, 통신 시스템(100)은 네트워크(150)를 통해 상호연결되는 제1 쌍의 단말 디바이스들(110 및 120)을 포함한다. 도 1의 예에서, 제1 쌍의 단말 디바이스들(110 및 120)은 데이터의 단방향 송신을 수행한다. 예를 들어, 단말 디바이스(110)는 네트워크(150)를 통해 다른 단말 디바이스(120)로의 송신을 위해 비디오 데이터(예를 들어, 단말 디바이스(110)에 의해 캡처되는 비디오 화상들의 스트림)를 코딩할 수 있다. 인코딩된 비디오 데이터는 하나 이상의 코딩된 비디오 비트스트림의 형식으로 송신될 수 있다. 단말 디바이스(120)는 네트워크(150)로부터 코딩된 비디오 데이터를 수신하고, 코딩된 비디오 데이터를 디코딩하여 비디오 화상들을 복구하고 복구된 비디오 데이터에 따라 비디오 화상들을 디스플레이할 수 있다. 단방향 데이터 송신은 미디어 서빙 애플리케이션들 등에서 일반적일 수 있다.

[0023] 다른 예에서, 통신 시스템(100)은, 예를 들어, 영상 회의 동안 발생할 수 있는 코딩된 비디오 데이터의 양방향 송신을 수행하는 제2 쌍의 단말 디바이스들(130 및 140)을 포함한다. 데이터의 양방향 송신을 위해, 일 예에서, 단말 디바이스들(130 및 140) 중의 각각의 단말 디바이스는 네트워크(150)를 통해 단말 디바이스들(130 및 140) 중의 다른 단말 디바이스로의 송신을 위해 비디오 데이터(예를 들어, 단말 디바이스에 의해 캡처되는 비디오 화상들의 스트림)를 코딩할 수 있다. 단말 디바이스들(130 및 140) 중의 각각의 단말 디바이스는 또한 단말 디바이스들(130 및 140) 중의 다른 단말 디바이스에 의해 송신된 코딩된 비디오 데이터를 수신할 수 있고, 코딩된 비디오 데이터를 디코딩하여 비디오 화상들을 복구할 수 있고, 복구된 비디오 데이터에 따라 액세스 가능한 디스플레이 디바이스에서 비디오 화상들을 디스플레이할 수 있다.

[0024] 도 1의 예에서, 단말 디바이스들(110, 120, 130 및 140)은 서버들, 개인용 컴퓨터들 및 스마트 폰들로서 예시될

수 있지만, 본 개시내용의 원리들은 그렇게 제한되지 않는다. 본 개시내용의 실시예들은 랩톱 컴퓨터들, 태블릿 컴퓨터들, 미디어 플레이어들 및/또는 전용 영상 회의 장비를 사용한 애플리케이션을 찾는다. 네트워크(150)는 예를 들어 와이어라인(유선) 및/또는 무선 통신 네트워크들을 포함하여, 단말 디바이스들(110, 120, 130 및 140) 사이에 코딩된 비디오 데이터를 전달하는 임의의 수의 네트워크들을 지시한다. 통신 네트워크(150)는 회선 교환 및/또는 패킷 교환 채널들에서 데이터를 교환할 수 있다. 대표적인 네트워크들은 통신 네트워크들, 로컬 영역 네트워크들, 광역 네트워크들 및/또는 인터넷을 포함한다. 본 논의의 목적을 위해, 네트워크(150)의 아키텍처 및 토폴로지는 아래에서 본 명세서에서 설명되지 않는 한 본 개시내용의 동작에 중요하지 않을 수 있다.

[0025] 도 2는, 개시된 주제를 위한 응용에 대한 예로서, 스트리밍 환경에서의 비디오 인코더 및 비디오 디코더의 배치를 예시한다. 개시된 주제는, 예를 들어, 영상 회의, 디지털 TV, CD, DVD, 메모리 스틱 등을 포함하는 디지털 미디어 상의 압축된 비디오의 저장 등을 포함하여, 다른 비디오 인에이블 애플리케이션들에 동등하게 적용가능할 수 있다.

[0026] 스트리밍 시스템은, 예를 들어 압축되지 않은 비디오 화상들의 스트림(202)을 생성하는 비디오 소스(201), 예를 들어 디지털 카메라를 포함할 수 있는 캡처 서브시스템(213)을 포함할 수 있다. 일 예에서, 비디오 화상들의 스트림(202)은 디지털 카메라에 의해 촬영되는 샘플들을 포함한다. 인코딩된 비디오 데이터(204)(또는 코딩된 비디오 비트스트림들)와 비교할 때 많은 데이터 용량을 강조하기 위해 굵은 라인으로 묘사된 비디오 화상들의 스트림(202)은 비디오 소스(201)에 결합된 비디오 인코더(203)를 포함하는 전자 디바이스(220)에 의해 처리될 수 있다. 비디오 인코더(203)는 아래에서 더 상세히 설명되는 바와 같이 개시된 주제의 양태들을 가능하게 하거나 구현하기 위해 하드웨어, 소프트웨어, 또는 이들의 조합을 포함할 수 있다. 비디오 화상들의 스트림(202)과 비교할 때 적은 데이터 용량을 강조하기 위해 얇은 라인으로서 묘사된 인코딩된 비디오 데이터(204)(또는 인코딩된 비디오 비트스트림(204))는 미래의 사용을 위해 스트리밍 서버(205) 상에 저장될 수 있다. 도 2에서의 클라이언트 서브시스템들(206 및 208)과 같은 하나 이상의 스트리밍 클라이언트 서브시스템들은 스트리밍 서버(205)에 액세스하여 인코딩된 비디오 데이터(204)의 카피들(207 및 209)을 검색할 수 있다. 클라이언트 서브시스템(206)은, 예를 들어, 전자 디바이스(230) 내에 비디오 디코더(210)를 포함할 수 있다. 비디오 디코더(210)는 인코딩된 비디오 데이터의 착신 카피(207)를 디코딩하고 디스플레이(212)(예를 들어, 디스플레이 스크린) 또는 다른 렌더링 디바이스(묘사되지 않음) 상에 렌더링될 수 있는 비디오 화상들의 발신 스트림(211)을 생성한다. 일부 스트리밍 시스템들에서, 인코딩된 비디오 데이터(204, 207, 및 209)(예를 들어, 비디오 비트스트림들)는 특정 비디오 코딩/압축 표준들에 따라 인코딩될 수 있다. 해당 표준들의 예들은 ITU-T 권고안(Recommendation) H.265를 포함한다. 일 예에서, 개발 중인 비디오 코딩 표준은 VVC(Versatile Video Coding)로서 널리 알려져 있다. 개시된 주제는 VVC의 맥락에서 사용될 수 있다.

[0027] 전자 디바이스들(220 및 230)은 다른 컴포넌트들(도시되지 않음)을 포함할 수 있다는 점에 유의한다. 예를 들어, 전자 디바이스(220)는 비디오 디코더(도시되지 않음)도 포함할 수 있고 전자 디바이스(230)는 비디오 인코더(도시되지 않음)도 포함할 수 있다.

[0028] 도 3은 본 개시내용의 일 실시예에 따른 비디오 디코더(310)의 블록도를 도시한다. 비디오 디코더(310)는 전자 디바이스(330)에 포함될 수 있다. 전자 디바이스(330)는 수신기(331)(예를 들어, 수신 회로)를 포함할 수 있다. 비디오 디코더(310)는 도 2의 예에서의 비디오 디코더(210) 대신에 사용될 수 있다.

[0029] 수신기(331)는 비디오 디코더(310)에 의해 디코딩될 하나 이상의 코딩된 비디오 시퀀스를 수신할 수 있다; 동일한 또는 다른 실시예에서, 한 번에 하나의 코딩된 비디오 시퀀스- 각각의 코딩된 비디오 시퀀스의 디코딩은 다른 코딩된 비디오 시퀀스들과 독립적이다. 코딩된 비디오 시퀀스는, 인코딩된 비디오 데이터를 저장하는 저장 디바이스에 대한 하드웨어/소프트웨어 링크일 수 있는, 채널(301)로부터 수신될 수 있다. 수신기(331)는 인코딩된 비디오 데이터를 다른 데이터, 예를 들어, 코딩된 오디오 데이터 및/또는 보조 데이터 스트림들과 함께 수신할 수 있고, 이들은 그것들 각각의 사용 엔티티들(묘사되지 않음)에 포워딩될 수 있다. 수신기(331)는 코딩된 비디오 시퀀스를 다른 데이터로부터 분리할 수 있다. 네트워크 지터를 방지하기 위해, 수신기(331)와 엔트로피 디코더/파서(320)(이후 "파서(320)") 사이에 버퍼 메모리(315)가 결합될 수 있다. 특정 응용들에서, 버퍼 메모리(315)는 비디오 디코더(310)의 일부이다. 다른 것들에서, 그것은 비디오 디코더(310)(묘사되지 않음) 외부에 있을 수 있다. 또 다른 것들에서, 예를 들어 네트워크 지터를 방지하기 위해, 비디오 디코더(310) 외부의 버퍼 메모리(묘사되지 않음), 그리고 추가로, 예를 들어 재생 타이밍을 핸들링하기 위해, 비디오 디코더(310) 내부의 다른 버퍼 메모리(315)가 존재할 수 있다. 수신기(331)가 충분한 대역폭 및 제어가능성의 저장/포워드 디바이스로부터, 또는 동기식 네트워크(isosynchronous network)로부터 데이터를 수신하고 있을 때, 버퍼 메모

리(315)는 필요하지 않을 수 있거나, 작을 수 있다. 인터넷과 같은 최선 노력 패킷 네트워크들 상에서의 사용을 위해, 버퍼 메모리(315)는 요구될 수 있고, 비교적 클 수 있고, 유리하게는 적응적 크기일 수 있고, 비디오 디코더(310) 외부의 운영 체제 또는 유사한 요소들(묘사되지 않음)에서 적어도 부분적으로 구현될 수 있다.

[0030] 비디오 디코더(310)는 코딩된 비디오 시퀀스로부터 심벌들(321)을 재구성하기 위해 파서(320)를 포함할 수 있다. 해당 심벌들의 카테고리들은 비디오 디코더(310)의 동작을 관리하기 위해 사용되는 정보, 및 잠재적으로, 도 3에 도시된 바와 같이, 전자 디바이스(330)의 일체 부분(integral part)은 아니지만 전자 디바이스(330)에 결합될 수 있는 렌더링 디바이스(312)(예를 들어, 디스플레이 스크린)와 같은 렌더링 디바이스를 제어하기 위한 정보를 포함한다. 렌더링 디바이스(들)에 대한 제어 정보는 SEI 메시지(Supplemental Enhancement Information) 또는 VUI(Video Usability Information) 파라미터 세트 프래그먼트들(묘사되지 않음)의 형식일 수 있다. 파서(320)는 수신되는 코딩된 비디오 시퀀스를 파싱/엔트로피 디코딩할 수 있다. 코딩된 비디오 시퀀스의 코딩은 비디오 코딩 기법 또는 표준에 따를 수 있고, 가변 길이 코딩, 허프만 코딩, 맥락 민감성(context sensitivity)을 갖거나 갖지 않는 산술 코딩 등을 포함하는 다양한 원리들을 따를 수 있다. 파서(320)는, 코딩된 비디오 시퀀스로부터, 그룹에 대응하는 적어도 하나의 파라미터에 기초하여, 비디오 디코더 내의 픽셀들의 서브그룹들 중 적어도 하나에 대한 서브그룹 파라미터들의 세트를 추출할 수 있다. 서브그룹들은 화상 그룹들(Groups of Pictures, GOPs), 화상들, 타일들, 슬라이스들, 매크로블록들, 코딩 유닛들(Coding Units, CUs), 블록들, 변환 유닛들(Transform Units, TUs), 예측 유닛들(Prediction Units, PUs) 등을 포함할 수 있다. 파서(320)는 또한 코딩된 비디오 시퀀스로부터 변환 계수들, 양자화기 파라미터 값들, 움직임 벡터들 등과 같은 정보를 추출할 수 있다.

[0031] 파서(320)는 버퍼 메모리(315)로부터 수신된 비디오 시퀀스에 대해 엔트로피 디코딩/파싱 동작을 수행하여, 심벌들(321)을 생성할 수 있다.

[0032] 심벌들(321)의 재구성은 코딩된 비디오 화상 또는 그것의 부분들의 타입(예컨대: 인터 및 인트라 화상, 인터 및 인트라 블록), 및 다른 인자들에 의존하여 다수의 상이한 유닛들을 수반할 수 있다. 어느 유닛들이 수반되는지, 그리고 어떻게 되는지는 파서(320)에 의해 코딩된 비디오 시퀀스로부터 파싱된 서브그룹 제어 정보에 의해 제어될 수 있다. 파서(320)와 아래의 다수의 유닛 사이의 그러한 서브그룹 제어 정보의 흐름은 명확성을 위해 묘사되어 있지 않다.

[0033] 이미 언급된 기능 블록들 이외에, 비디오 디코더(310)는 아래에 설명되는 바와 같이 개념적으로 다수의 기능 유닛으로 세분될 수 있다. 상업적 제약 하에서 동작하는 실제 구현에서, 이들 유닛 중 다수는 서로 밀접하게 상호작용하고, 적어도 부분적으로 서로 통합될 수 있다. 그러나, 개시된 주제를 설명하기 위해, 아래의 기능 유닛들로의 개념적 세분이 적절하다.

[0034] 제1 유닛은 스케일러/역변환 유닛(351)이다. 스케일러/역변환 유닛(351)은, 파서(320)로부터의 심벌(들)(321)로서, 어느 변환을 사용할지, 블록 크기, 양자화 인자, 양자화 스케일링 행렬들 등을 포함하여, 제어 정보뿐만 아니라 양자화된 변환 계수를 수신한다. 스케일러/역변환 유닛(351)은 집계기(aggregator)(355)에 입력될 수 있는 샘플 값들을 포함하는 블록들을 출력할 수 있다.

[0035] 일부 경우들에서, 스케일러/역변환(351)의 출력 샘플들은 인트라 코딩된 블록에 관련될 수 있다; 즉, 이전에 재구성된 화상들로부터의 예측 정보를 사용하는 것이 아니고, 현재 화상의 이전에 재구성된 부분들로부터의 예측 정보를 사용할 수 있는 블록. 그러한 예측 정보는 인트라 화상 예측 유닛(352)에 의해 제공될 수 있다. 일부 경우들에서, 인트라 화상 예측 유닛(352)은 현재 화상 버퍼(358)로부터 페치된 주위의 이미 재구성된 정보를 사용하여, 재구성 중인 블록의 동일한 크기 및 형상의 블록을 생성한다. 현재 화상 버퍼(358)는, 예를 들어, 부분적으로 재구성된 현재 화상 및/또는 완전히 재구성된 현재 화상을 버퍼링한다. 집계기(355)는, 일부 경우들에서, 샘플당 기준으로, 인트라 예측 유닛(352)이 생성한 예측 정보를 스케일러/역변환 유닛(351)에 의해 제공된 출력 샘플 정보에 추가한다.

[0036] 다른 경우들에서, 스케일러/역변환 유닛(351)의 출력 샘플들은 인터 코딩되고, 잠재적으로 움직임 보상된 블록에 관련될 수 있다. 그러한 경우에, 움직임 보상 예측 유닛(353)은 참조 화상 메모리(357)에 액세스하여 예측에 사용되는 샘플들을 페치할 수 있다. 블록에 관련된 심벌들(321)에 따라 페치된 샘플들을 움직임 보상한 후에, 이들 샘플은 집계기(355)에 의해 스케일러/역변환 유닛(351)의 출력(이 경우 잔차 샘플들 또는 잔차 신호라고 불림)에 추가되어 출력 샘플 정보를 생성할 수 있다. 움직임 보상 예측 유닛(353)이 예측 샘플들을 페치하는 참조 화상 메모리(357) 내의 어드레스들은, 예를 들어 X, Y, 및 참조 화상 컴포넌트들을 가질 수 있는 심벌들(321)의 형식으로 움직임 보상 예측 유닛(353)에 이용가능한 움직임 벡터들에 의해 제어될 수 있다. 움직임

보상은 또한 서브샘플 정확한 움직임 벡터들이 사용 중일 때 참조 화상 메모리(357)로부터 페치된 샘플 값들의 보간, 움직임 벡터 예측 메커니즘 등을 포함할 수 있다.

[0037] 집계기(355)의 출력 샘플들에 대해 루프 필터 유닛(356) 내의 다양한 루프 필터링 기법들이 수행될 수 있다. 비디오 압축 기법들은, 파서(320)로부터의 심벌들(321)로서 루프 필터 유닛(356)에 이용가능하게 되고 코딩된 비디오 시퀀스(코딩된 비디오 비트스트림이라고도 지칭됨)에 포함된 파라미터들에 의해 제어되지만, 코딩된 화상 또는 코딩된 비디오 시퀀스의 이전(디코딩 순서로) 부분들의 디코딩 동안 획득된 메타-정보에 응답할 뿐만 아니라, 이전에 재구성된 및 루프-필터링된 샘플 값들에 응답할 수도 있는 인-루프 필터(in-loop filter) 기법들을 포함할 수 있다.

[0038] 루프 필터 유닛(356)의 출력은 렌더링 디바이스(312)에 출력될 뿐만 아니라 미래의 인터-화상 예측에서 사용하기 위해 참조 화상 메모리(357)에 저장될 수도 있는 샘플 스트림일 수 있다.

[0039] 특정 코딩된 화상들은, 완전히 재구성되면, 미래 예측을 위한 참조 화상들로서 사용될 수 있다. 예를 들어, 현재 화상에 대응하는 코딩된 화상이 완전히 재구성되고 코딩된 화상이 참조 화상으로서 식별되면(예를 들어, 파서(320)에 의해), 현재 화상 버퍼(358)는 참조 화상 메모리(357)의 일부가 될 수 있고, 다음의 코딩된 화상의 재구성을 개시하기 전에 새로운 현재 화상 버퍼가 재할당될 수 있다.

[0040] 비디오 디코더(310)는 ITU-T Rec. H.265와 같은 표준에서의 미리 결정된 비디오 압축 기법에 따라 디코딩 동작들을 수행할 수 있다. 코딩된 비디오 시퀀스는 비디오 압축 기법 또는 표준의 구문, 또는 비디오 압축 기법 또는 표준에서 문서화된 프로파일들 둘 다를 고수한다는 점에서, 코딩된 비디오 시퀀스는 사용 중인 비디오 압축 기법 또는 표준에 의해 특정된 구문을 따를 수 있다. 구체적으로, 프로파일은 비디오 압축 기법 또는 표준에서 이용가능한 모든 툴들로부터 해당 프로파일 하에서 사용하기 위해 이용가능한 유일한 툴들로서 특정 툴들을 선택할 수 있다. 또한 준수를 위해 필요한 것은 코딩된 비디오 시퀀스의 복잡성이 비디오 압축 기법 또는 표준의 레벨에 의해 정의된 경계 내에 있는 것일 수 있다. 일부 경우들에서, 레벨들은 최대 화상 크기, 최대 프레임 레이트, 최대 재구성 샘플 레이트(예를 들어, 초당 메가샘플수로 측정됨), 최대 참조 화상 크기 등을 제한한다. 레벨들에 의해 설정된 한계들은, 일부 경우들에서, HRD(Hypothetical Reference Decoder) 사양들 및 코딩된 비디오 시퀀스에서 시그널링된 HRD 버퍼 관리를 위한 메타데이터를 통해 추가로 제한될 수 있다.

[0041] 일 실시예에서, 수신기(331)는 인코딩된 비디오와 함께 추가적인(중복) 데이터를 수신할 수 있다. 이 추가적인 데이터는 코딩된 비디오 시퀀스(들)의 일부로서 포함될 수 있다. 이 추가적인 데이터는 데이터를 적절히 디코딩하고/하거나 원래의 비디오 데이터를 더 정확하게 재구성하기 위해 비디오 디코더(310)에 의해 사용될 수 있다. 추가적인 데이터는 예를 들어, 시간, 공간, 또는 신호 잡음 비(SNR) 향상 계층들, 중복 슬라이스들, 중복 화상들, 순방향 오류 정정 코드들 등의 형식일 수 있다.

[0042] 도 4은 본 개시내용의 일 실시예에 따른 비디오 인코더(403)의 블록도를 도시한다. 비디오 인코더(403)는 전자 디바이스(420)에 포함된다. 전자 디바이스(420)는 송신기(440)(예를 들어, 송신 회로)를 포함한다. 비디오 인코더(403)는 도 2의 예에서의 비디오 인코더(203) 대신에 사용될 수 있다.

[0043] 비디오 인코더(403)는 비디오 인코더(403)에 의해 코딩될 비디오 이미지(들)를 캡처할 수 있는 비디오 소스(401)(도 4의 예에서는 전자 디바이스(420)의 일부가 아님)로부터 비디오 샘플들을 수신할 수 있다. 다른 예에서, 비디오 소스(401)는 전자 디바이스(420)의 일부이다.

[0044] 비디오 소스(401)는, 임의의 적합한 비트 심도(예를 들어: 8 비트, 10 비트, 12 비트, ...), 임의의 색공간(예를 들어, BT.601 Y CrCB, RGB, ...), 및 임의의 적합한 샘플링 구조(예를 들어, Y CrCb 4:2:0, Y CrCb 4:4:4)일 수 있는 디지털 비디오 샘플 스트림의 형식으로 비디오 인코더(403)에 의해 코딩될 소스 비디오 시퀀스를 제공할 수 있다. 미디어 서빙 시스템에서, 비디오 소스(401)는 이전에 준비된 비디오를 저장하는 저장 디바이스일 수 있다. 영상 회의 시스템에서, 비디오 소스(401)는 비디오 시퀀스로서 로컬 이미지 정보를 캡처하는 카메라일 수 있다. 비디오 데이터는 순차적으로 볼 때 움직임을 부여하는 복수의 개별 화상으로서 제공될 수 있다. 화상들 자체는 픽셀들의 공간 어레이로서 조직될 수 있고, 여기서 각각의 픽셀은 사용 중인 샘플링 구조, 색 공간 등에 의존하여 하나 이상의 샘플을 포함할 수 있다. 본 기술분야의 통상의 기술자는 픽셀들과 샘플들 사이의 관계를 쉽게 이해할 수 있다. 아래의 설명은 샘플들에 초점을 맞춘다.

[0045] 일 실시예에 따르면, 비디오 인코더(403)는 소스 비디오 시퀀스의 화상들을 실시간으로 또는 응용에 의해 요구되는 임의의 다른 시간 제약들 하에서 코딩된 비디오 시퀀스(443)로 코딩 및 압축할 수 있다. 적절한 코딩 속도를 시행하는 것이 컨트롤러(450)의 하나의 기능이다. 일부 실시예들에서, 컨트롤러(450)는 아래 설명되는 바

와 같이 다른 기능 유닛들을 제어하고 다른 기능 유닛들에 기능적으로 결합된다. 결합은 명확성을 위해 묘사되어 있지 않다. 컨트롤러(450)에 의해 설정된 파라미터들은 레이트 제어 관련 파라미터들(화상 스킵, 양자화기, 레이트-왜곡 최적화 기법들의 람다 값들, ...), 화상 크기, 화상 그룹(GOP) 레이아웃, 최대 움직임 벡터 검색 범위 등을 포함할 수 있다. 컨트롤러(450)는 특정 시스템 설계에 대해 최적화된 비디오 인코더(403)에 관련된 다른 적합한 기능들을 갖도록 구성될 수 있다.

[0046] 일부 실시예들에서, 비디오 인코더(403)는 코딩 루프에서 동작하도록 구성된다. 과도하게 단순화된 설명으로서, 일 예에서, 코딩 루프는 소스 코더(430)(예를 들어, 코딩될 입력 화상, 및 참조 화상(들)에 기초하여 심벌 스트림과 같은 심벌들을 생성하는 것을 담당함), 및 비디오 인코더(403)에 임베드된 (로컬) 디코더(433)를 포함할 수 있다. 디코더(433)는 (원격) 디코더가 또한 생성하는 것과 유사한 방식으로 샘플 데이터를 생성하기 위해 심벌들을 재구성한다(심벌들과 코딩된 비디오 비트스트림 사이의 임의의 압축이 개시된 주제에서 고려되는 비디오 압축 기법들에서 무손실이기 때문에). 재구성된 샘플 스트림(샘플 데이터)은 참조 화상 메모리(434)에 입력된다. 심벌 스트림의 디코딩이 디코더 위치(로컬 또는 원격)와는 독립적으로 비트-정확한 결과들을 야기하기 때문에, 참조 화상 메모리(434) 내의 콘텐츠도 또한 로컬 인코더와 원격 인코더 사이에서 비트 정확하다. 다시 말해서, 인코더의 예측 부분은 디코딩 동안 예측을 사용할 때 디코더가 "보는" 것과 정확히 동일한 샘플 값들을 참조 화상 샘플로서 "본다". 참조 화상 동기성의 이 기본적인 원리(그리고 결과적인 드리프트, 예를 들어, 채널 오류들 때문에 동기성이 유지될 수 없는 경우)는 일부 관련 기법들에서도 사용된다.

[0047] "로컬" 디코더(433)의 동작은 도 3과 관련하여 위에서 이미 상세히 설명된 비디오 디코더(310)와 같은 "원격" 디코더와 동일할 수 있다. 그러나, 또한 도 3을 잠시 참조하면, 심벌들이 이용가능하고 엔트로피 코더(445) 및 파서(320)에 의한 코딩된 비디오 시퀀스로의 심벌들의 인코딩/디코딩이 무손실일 수 있기 때문에, 버퍼 메모리(315), 및 파서(320)를 포함하는, 비디오 디코더(310)의 엔트로피 디코딩 부분들은 로컬 디코더(433)에서 완전히 구현되지 않을 수 있다.

[0048] 이 시점에서 이루어질 수 있는 관찰은, 디코더에 존재하는 파싱/엔트로피 디코딩을 제외한 임의의 디코더 기법이 또한 필연적으로, 대응하는 인코더에서, 실질적으로 동일한 기능 형식으로 존재할 필요가 있다는 점이다. 이러한 이유로, 개시된 주제는 디코더 동작에 초점을 맞춘다. 인코더 기법들은 포괄적으로 설명된 디코더 기법들의 역이기 때문에 그것들에 대한 설명은 축약될 수 있다. 특정 영역들에서만 더 상세한 설명이 요구되고 아래에 제공된다.

[0049] 동작 동안, 일부 예들에서, 소스 코더(430)는, "참조 화상"으로 지정된 비디오 시퀀스로부터의 하나 이상의 이전에 코딩된 화상을 참조하여 예측적으로 입력 화상을 코딩하는, 움직임 보상된 예측 코딩을 수행할 수 있다. 이러한 방식으로, 코딩 엔진(432)은 입력 화상의 픽셀 블록들과 입력 화상에 대한 예측 참조(들)로서 선택될 수 있는 참조 화상(들)의 픽셀 블록들 사이의 차이들을 코딩한다.

[0050] 로컬 비디오 디코더(433)는, 소스 코더(430)에 의해 생성된 심벌들에 기초하여, 참조 화상들로서 지정될 수 있는 화상들의 코딩된 비디오 데이터를 디코딩할 수 있다. 코딩 엔진(432)의 동작들은 유리하게는 손실 프로세스들일 수 있다. 코딩된 비디오 데이터가 비디오 디코더(도 4에 도시되지 않음)에서 디코딩될 수 있는 경우, 재구성된 비디오 시퀀스는 전형적으로 일부 오류들을 갖는 소스 비디오 시퀀스의 복제본일 수 있다. 로컬 비디오 디코더(433)는 참조 화상들에 대해 비디오 디코더에 의해 수행될 수 있는 디코딩 프로세스들을 복제하고 재구성된 참조 화상들이 참조 화상 캐시(434)에 저장되게 할 수 있다. 이러한 방식으로, 비디오 인코더(403)는 (송신 오류들이 없이) 원단(far-end) 비디오 디코더에 의해 획득될 재구성된 참조 화상으로서 공통 콘텐츠를 갖는 재구성된 참조 화상들의 카피들을 저장할 수 있다.

[0051] 예측기(435)는 코딩 엔진(432)에 대한 예측 검색들을 수행할 수 있다. 즉, 코딩될 새로운 화상에 대해, 예측기(435)는 새로운 화상들에 대한 적절한 예측 참조로서 역할할 수 있는 참조 화상 움직임 벡터들, 블록 형상들 등과 같은 특정 메타데이터 또는 샘플 데이터(후보 참조 픽셀 블록들로서)에 대해 참조 화상 메모리(434)를 검색할 수 있다. 예측기(435)는 적절한 예측 참조들을 찾기 위해 샘플 블록 바이 픽셀 블록(sample block-by-pixel block) 기준으로 동작할 수 있다. 일부 경우들에서, 예측기(435)에 의해 획득된 검색 결과들에 의해 결정된 바와 같이, 입력 화상은 참조 화상 메모리(434)에 저장된 다수의 참조 화상으로부터 인출된 예측 참조들을 가질 수 있다.

[0052] 컨트롤러(450)는, 예를 들어, 비디오 데이터를 인코딩하기 위해 사용되는 파라미터들 및 서브그룹 파라미터들의 설정을 포함하여, 소스 코더(430)의 코딩 동작을 관리할 수 있다.

[0053] 전술한 모든 기능 유닛들의 출력은 엔트로피 코더(445)에서 엔트로피 코딩을 겪을 수 있다. 엔트로피 코더(445)는 다양한 기능 유닛들에 의해 생성된 심벌들을, 허프만 코딩, 가변 길이 코딩, 산술 코딩 등과 같은 기법들에 따라 심벌들을 무손실 압축함으로써, 코딩된 비디오 시퀀스로 변환한다.

[0054] 송신기(440)는, 인코딩된 비디오 데이터를 저장할 저장 디바이스에 대한 하드웨어/소프트웨어 링크일 수 있는, 통신 채널(460)을 통한 송신을 준비하기 위해 엔트로피 코더(445)에 의해 생성된 코딩된 비디오 시퀀스(들)를 버퍼링할 수 있다. 송신기(440)는 비디오 코더(403)로부터의 코딩된 비디오 데이터를 송신될 다른 데이터, 예를 들어, 코딩된 오디오 데이터 및/또는 보조 데이터 스트림들(소스들이 도시되지 않음)과 병합할 수 있다.

[0055] 컨트롤러(450)는 비디오 인코더(403)의 동작을 관리할 수 있다. 코딩 동안, 컨트롤러(450)는, 각각의 화상에 적용될 수 있는 코딩 기법들에 영향을 미칠 수 있는, 특정 코딩된 화상 타입을 각각의 코딩된 화상에 할당할 수 있다. 예를 들어, 화상들은 종종 다음 화상 타입들 중 하나로서 할당될 수 있다:

[0056] 인트라 화상(I 화상)은 예측의 소스로서 시퀀스 내의 임의의 다른 화상을 사용하지 않고 코딩되고 디코딩될 수 있는 것일 수 있다. 일부 비디오 코덱들은, 예를 들어, "IDR"(Independent Decoder Refresh) 화상들을 포함하는, 상이한 타입의 인트라 화상들을 허용한다. 본 기술분야의 통상의 기술자는 I 화상들의 해당 변형들 및 그것들 각각의 애플리케이션들 및 특징들을 인식한다.

[0057] 예측 화상(P 화상)은 각각의 블록의 샘플 값들을 예측하기 위해 많아야 하나의 움직임 벡터 및 참조 인덱스를 사용하여 인트라 예측 또는 인터 예측을 사용하여 코딩되고 디코딩될 수 있는 것일 수 있다.

[0058] 양방향 예측 화상(B 화상)은 각각의 블록의 샘플 값들을 예측하기 위해 많아야 2개의 움직임 벡터 및 참조 인덱스를 사용하여 인트라 예측 또는 인터 예측을 사용하여 코딩되고 디코딩될 수 있는 것일 수 있다. 유사하게, 다중-예측 화상들은 단일 블록의 재구성을 위해 2개보다 많은 참조 화상 및 연관된 메타데이터를 사용할 수 있다.

[0059] 소스 화상들은 일반적으로 복수의 샘플 블록(예를 들어, 각각 4x4, 8x8, 4x8, 또는 16x16 샘플들의 블록들)으로 공간적으로 세분되고 블록 바이 블록(block-by-block) 기준으로 코딩될 수 있다. 블록들은 블록들의 각각의 화상들에 적용되는 코딩 할당에 의해 결정된 다른(이미 코딩된) 블록들을 참조하여 예측적으로 코딩될 수 있다. 예를 들어, I 화상들의 블록들은 비예측적으로 코딩될 수 있거나 그것들은 동일한 화상의 이미 코딩된 블록들을 참조하여 예측적으로 코딩될 수 있다(공간 예측 또는 인트라 예측). P 화상의 픽셀 블록들은, 하나의 이전에 코딩된 참조 화상을 참조하여 공간 예측을 통해 또는 시간 예측을 통해, 예측적으로 코딩될 수 있다. B 화상들의 블록들은, 하나 또는 2개의 이전에 코딩된 참조 화상을 참조하여 공간 예측을 통해 또는 시간 예측을 통해, 예측적으로 코딩될 수 있다.

[0060] 비디오 인코더(403)는 ITU-T Rec. H.265와 같은 표준에서의 미리 결정된 비디오 코딩 기법에 따라 코딩 동작들을 수행할 수 있다. 그것의 동작 중에, 비디오 인코더(403)는, 입력 비디오 시퀀스에서 시간 및 공간 중복성을 사용하는 예측 코딩 동작들을 포함하여, 다양한 압축 동작들을 수행할 수 있다. 따라서, 코딩된 비디오 데이터는 사용 중인 비디오 코딩 기법 또는 표준에 의해 특정된 구문을 따를 수 있다.

[0061] 일 실시예에서, 송신기(440)는 인코딩된 비디오와 함께 추가적인 데이터를 송신할 수 있다. 소스 코더(430)는 코딩된 비디오 시퀀스의 일부로서 그러한 데이터를 포함할 수 있다. 추가적인 데이터는 시간/공간/SNR 향상 계층들, 중복 화상들 및 슬라이스들과 같은 다른 형식의 중복 데이터, SEI 메시지들, VUI 파라미터 세트 프래그먼트들 등을 포함할 수 있다.

[0062] 비디오는 시간 시퀀스에서 복수의 소스 화상(비디오 화상)으로서 캡처될 수 있다. 인트라-화상 예측(종종 인트라 예측으로 축약됨)은 주어진 화상에서 공간 상관을 사용하고, 인터-화상 예측은 화상들 사이의 (시간 또는 다른) 상관을 사용한다. 일 예에서, 현재 화상이라고 지칭되는, 인코딩/디코딩 중인 특정 화상이 블록들로 분할된다. 현재 화상 내의 블록이 비디오 내의 이전에 코딩되고 여전히 버퍼링된 참조 화상 내의 참조 블록과 유사할 때, 현재 화상 내의 블록은 움직임 벡터라고 지칭되는 벡터에 의해 코딩될 수 있다. 움직임 벡터는 참조 화상 내의 참조 블록을 가리키고, 다수의 참조 화상이 사용 중인 경우, 참조 화상을 식별하는 제3의 차원을 가질 수 있다.

[0063] 일부 실시예들에서, 인터-화상 예측에서 양방향 예측(bi-prediction) 기법이 사용될 수 있다. 양방향 예측 기법에 따르면, 둘 다 비디오 내의 현재 화상에 디코딩 순서에서 앞서는(그러나, 디스플레이 순서에서, 과거 및 미래에 각각 있을 수 있는) 제1 참조 화상 및 제2 참조 화상과 같은 2개의 참조 화상이 사용된다. 현재 화상

내의 블록은 제1 참조 화상 내의 제1 참조 블록을 가리키는 제1 움직임 벡터, 및 제2 참조 화상 내의 제2 참조 블록을 가리키는 제2 움직임 벡터에 의해 코딩될 수 있다. 블록은 제1 참조 블록과 제2 참조 블록의 조합에 의해 예측될 수 있다.

[0064] 또한, 코딩 효율을 개선하기 위해 인터-화상 예측에서 병합 모드 기법이 사용될 수 있다.

[0065] 본 개시내용의 일부 실시예들에 따르면, 인터-화상 예측들 및 인트라-화상 예측들과 같은 예측들이 블록들의 유닛으로 수행된다. 예를 들어, HEVC 표준에 따르면, 비디오 화상들의 시퀀스 내의 화상은 압축을 위해 코딩 트리 유닛들(CTU)로 분할되고, 화상 내의 CTU들은 64x64 픽셀들, 32x32 픽셀들, 또는 16x16 픽셀들과 같은 동일한 크기를 갖는다. 일반적으로, CTU는 3개의 코딩 트리 블록(CTB)을 포함하는데, 이는 하나의 루마 CTB 및 2개의 크로마 CTB이다. 각각의 CTU는 하나 또는 다수의 코딩 유닛(CU)들로 재귀적으로 쿼드트리 분열될 수 있다. 예를 들어, 64x64 픽셀들의 CTU는 64x64 픽셀들의 하나의 CU, 또는 32x32 픽셀들의 4개의 CU, 또는 16x16 픽셀들의 16개의 CU로 분열될 수 있다. 일 예에서, 각각의 CU는, 인터 예측 타입 또는 인트라 예측 타입과 같은, CU에 대한 예측 타입을 결정하기 위해 분석된다. CU는 시간 및/또는 공간 예측성에 의존하여 하나 이상의 예측 유닛(PU)으로 분열된다. 일반적으로, 각각의 PU는 루마 예측 블록(PB), 및 2개의 크로마 PB를 포함한다. 일 실시예에서, 코딩(인코딩/디코딩)에서의 예측 동작은 예측 블록의 유닛으로 수행된다. 예측 블록의 예로서 루마 예측 블록을 사용하여, 예측 블록은, 8x8 픽셀들, 16x16 픽셀들, 8x16 픽셀들, 16x8 픽셀들 등과 같은, 픽셀들에 대한 값들(예를 들어, 루마 값들)의 행렬을 포함한다.

[0066] 도 5는 본 개시내용의 다른 실시예에 따른 비디오 인코더(503)의 도면을 도시한다. 비디오 인코더(503)는 비디오 화상들의 시퀀스에서 현재 비디오 화상 내의 샘플 값들의 처리 블록(예를 들어, 예측 블록)을 수신하고, 처리 블록을 코딩된 비디오 시퀀스의 일부인 코딩된 화상 내에 인코딩하도록 구성된다. 일 예에서, 비디오 인코더(503)는 도 2의 예에서의 비디오 인코더(203) 대신에 사용된다.

[0067] HEVC 예에서, 비디오 인코더(503)는 8x8 샘플들 등의 예측 블록과 같은 처리 블록에 대한 샘플 값들의 행렬 등을 수신한다. 비디오 인코더(503)는 처리 블록이, 예를 들어, 레이트-왜곡 최적화를 사용하여 인트라 모드, 인터 모드, 또는 양방향 예측 모드 중 어느 것을 사용하여 최선으로 코딩되는지를 결정한다. 처리 블록이 인트라 모드로 코딩되어야 할 때, 비디오 인코더(503)는 인트라 예측 기법을 사용하여 처리 블록을 코딩된 화상 내에 인코딩할 수 있다; 그리고 처리 블록이 인터 모드 또는 양방향 예측 모드로 코딩되어야 할 때, 비디오 인코더(503)는 인터 예측 또는 양방향 예측 기법을 각각 사용하여 처리 블록을 코딩된 화상 내에 인코딩할 수 있다. 특정 비디오 코딩 기법들에서, 병합 모드는 예측자들 외부의 코딩된 움직임 벡터 성분의 혜택 없이 하나 이상의 움직임 벡터 예측자들로부터 움직임 벡터가 도출되는 인터 화상 예측 서브모드일 수 있다. 특정 다른 비디오 코딩 기법들에서, 대상 블록에 적용가능한 움직임 벡터 성분이 존재할 수 있다. 일 예에서, 비디오 인코더(503)는 처리 블록들의 모드를 결정하기 위한 모드 결정 모듈(도시되지 않음)과 같은 다른 컴포넌트들을 포함한다.

[0068] 도 5의 예에서, 비디오 인코더(503)는 도 5에 도시된 바와 같이 함께 결합된 인터 인코더(530), 인트라 인코더(522), 잔차 계산기(523), 스위치(526), 잔차 인코더(524), 일반 컨트롤러(521), 및 엔트로피 인코더(525)를 포함한다.

[0069] 인터 인코더(530)는 현재 블록(예를 들어, 처리 블록)의 샘플들을 수신하고, 블록을 참조 화상들 내의 하나 이상의 참조 블록(예를 들어, 이전 화상들 및 나중 화상들 내의 블록들)과 비교하고, 인터 예측 정보(예를 들어, 인터 인코딩 기법에 따른 중복 정보의 설명, 움직임 벡터들, 병합 모드 정보)를 생성하고, 임의의 적합한 기법을 사용하여 인터 예측 정보에 기초하여 인터 예측 결과들(예를 들어, 예측된 블록)을 계산하도록 구성된다. 일부 예들에서, 참조 화상들은 인코딩된 비디오 정보에 기초하여 디코딩되는 디코딩된 참조 화상들이다.

[0070] 인트라 인코더(522)는 현재 블록(예를 들어, 처리 블록)의 샘플들을 수신하고, 일부 경우들에서 블록을 동일한 화상 내의 이미 코딩된 블록들과 비교하고, 변환 후 양자화된 계수들을 생성하고, 일부 경우들에서 또한 인트라 예측 정보(예를 들어, 하나 이상의 인트라 인코딩 기법에 따라 인트라 예측 방향 정보)를 수신하도록 구성된다. 일 예에서, 인트라 인코더(522)는 또한 동일한 화상 내의 참조 블록들 및 인트라 예측 정보에 기초하여 인트라 예측 결과들(예를 들어, 예측 블록)을 계산한다.

[0071] 일반 컨트롤러(521)는 일반 제어 데이터를 결정하고 일반 제어 데이터에 기초하여 비디오 인코더(503)의 다른 컴포넌트들을 제어하도록 구성된다. 일 예에서, 일반 컨트롤러(521)는 블록의 모드를 결정하고, 모드에 기초하여 스위치(526)에 제어 신호를 제공한다. 예를 들어, 모드가 인트라 모드일 때, 일반 컨트롤러(521)는 잔차 계

산기(523)에 의한 사용을 위해 인트라 모드 결과를 선택하도록 스위치(526)를 제어하고, 인트라 예측 정보를 선택하고 인트라 예측 정보를 비트스트림에 포함시키도록 엔트로피 인코더(525)를 제어한다; 그리고 모드가 인터 모드일 때, 일반 컨트롤러(521)는 잔차 계산기(523)에 의한 사용을 위해 인터 예측 결과를 선택하도록 스위치(526)를 제어하고, 인터 예측 정보를 선택하고 인터 예측 정보를 비트스트림에 포함시키도록 엔트로피 인코더(525)를 제어한다.

[0072] 잔차 계산기(523)는 수신된 블록과 인트라 인코더(522) 또는 인터 인코더(530)로부터 선택된 예측 결과들 사이의 차이(잔차 데이터)를 계산하도록 구성된다. 잔차 인코더(524)는 잔차 데이터에 기초하여 동작하여 잔차 데이터를 인코딩하여 변환 계수들을 생성하도록 구성된다. 일 예에서, 잔차 인코더(524)는 잔차 데이터를 공간 도메인으로부터 주파수 도메인으로 변환하고, 변환 계수들을 생성하도록 구성된다. 그 후 변환 계수들에 대해 양자화 처리를 수행하여 양자화된 변환 계수들을 획득한다. 다양한 실시예들에서, 비디오 인코더(503)는 잔차 디코더(528)를 또한 포함한다. 잔차 디코더(528)는 역변환을 수행하고, 디코딩된 잔차 데이터를 생성하도록 구성된다. 디코딩된 잔차 데이터는 인트라 인코더(522) 및 인터 인코더(530)에 의해 적합하게 사용될 수 있다. 예를 들어, 인터 인코더(530)는 디코딩된 잔차 데이터 및 인터 예측 정보에 기초하여 디코딩된 블록들을 생성할 수 있고, 인트라 인코더(522)는 디코딩된 잔차 데이터 및 인트라 예측 정보에 기초하여 디코딩된 블록들을 생성할 수 있다. 디코딩된 블록들은 디코딩된 화상들을 생성하기 위해 적합하게 처리되고 디코딩된 화상들은 메모리 회로(도시되지 않음)에 버퍼링되고 일부 예들에서 참조 화상들로서 사용될 수 있다.

[0073] 엔트로피 인코더(525)는 인코딩된 블록을 포함하도록 비트스트림을 포맷하도록 구성된다. 엔트로피 인코더(525)는 HEVC 표준과 같은 적합한 표준에 따라 다양한 정보를 포함하도록 구성된다. 일 예에서, 엔트로피 인코더(525)는 일반 제어 데이터, 선택된 예측 정보(예를 들어, 인트라 예측 정보 또는 인터 예측 정보), 잔차 정보, 및 다른 적합한 정보를 비트스트림 내에 포함시키도록 구성된다. 개시된 주제에 따르면, 인터 모드 또는 양방향 예측 모드의 병합 서브모드에서 블록을 코딩할 때, 잔차 정보가 존재하지 않는다는 점에 유의한다.

[0074] 도 6은 본 개시내용의 다른 실시예에 따른 비디오 디코더(610)의 도면을 도시한다. 비디오 디코더(610)는 코딩된 비디오 시퀀스의 일부인 코딩된 화상들을 수신하고, 코딩된 화상들을 디코딩하여 재구성된 화상들을 생성하도록 구성된다. 일 예에서, 비디오 디코더(610)는 도 2의 예에서의 비디오 디코더(210) 대신에 사용된다.

[0075] 도 6의 예에서, 비디오 디코더(610)는 도 6에 도시된 바와 같이 함께 결합된 엔트로피 디코더(671), 인터 디코더(680), 잔차 디코더(673), 재구성 모듈(674), 및 인트라 디코더(672)를 포함한다.

[0076] 엔트로피 디코더(671)는, 코딩된 화상으로부터, 코딩된 화상이 구성되는 구문 요소들을 나타내는 특정 심벌들을 재구성하도록 구성될 수 있다. 그러한 심벌들은, 예를 들어, 블록이 코딩되는 모드(예컨대, 예를 들어, 인트라 모드, 인터 모드, 양방향 예측(bi-predicted) 모드, 후자의 둘은 병합 서브모드 또는 다른 서브모드에서), 인트라 디코더(672) 또는 인터 디코더(680) 각각에 의한 예측을 위해 사용되는 특정 샘플 또는 메타데이터를 식별할 수 있는 예측 정보(예컨대, 예를 들어, 인트라 예측 정보 또는 인터 예측 정보), 예를 들어, 양자화된 변환 계수들의 형식으로 된 잔차 정보 등을 포함할 수 있다. 일 예에서, 예측 모드가 인터 또는 양방향 예측 모드일 때, 인터 예측 정보가 인터 디코더(680)에 제공된다; 그리고 예측 타입이 인트라 예측 타입일 때, 인트라 예측 정보가 인트라 디코더(672)에 제공된다. 잔차 정보에 대해 역양자화가 수행될 수 있고 이는 잔차 디코더(673)에 제공된다.

[0077] 인터 디코더(680)는 인터 예측 정보를 수신하고, 인터 예측 정보에 기초하여 인터 예측 결과들을 생성하도록 구성된다.

[0078] 인트라 디코더(672)는 인트라 예측 정보를 수신하고, 인트라 예측 정보에 기초하여 예측 결과들을 생성하도록 구성된다.

[0079] 잔차 디코더(673)는 역양자화를 수행하여 탈양자화된 변환 계수들을 추출하고, 탈양자화된 변환 계수들을 처리하여 잔차를 주파수 도메인으로부터 공간 도메인으로 변환하도록 구성된다. 잔차 디코더(673)는 또한(양자화기 파라미터(QP)를 포함하도록) 특정 제어 정보를 요구할 수 있고, 그 정보는 엔트로피 디코더(671)에 의해 제공될 수 있다(이는 단지 저용량 제어 정보일 수 있으므로 데이터 경로가 묘사되지 않음).

[0080] 재구성 모듈(674)은, 공간 도메인에서, 잔차 디코더(673)에 의해 출력된 잔차와 예측 결과들(경우에 따라 인터 또는 인트라 예측 모듈에 의해 출력된 것)을 조합하여 재구성된 블록을 형성하도록 구성하고, 재구성된 블록은 재구성된 화상의 일부일 수 있고, 재구성된 화상은 결국 재구성된 비디오의 일부일 수 있다. 시각적 품질을 개선하기 위해 디블로킹 동작 등과 같은 다른 적합한 동작들이 수행될 수 있다는 점에 유의한다.

[0081] 비디오 인코더들(203, 403, 및 503), 및 비디오 디코더들(210, 310, 및 610)은 임의의 적합한 기법을 사용하여 구현될 수 있다는 점에 유의한다. 일 실시예에서, 비디오 인코더들(203, 403, 및 503), 및 비디오 디코더들(210, 310, 및 610)은 하나 이상의 집적 회로를 사용하여 구현될 수 있다. 다른 실시예에서, 비디오 인코더들(203, 403, 및 503), 및 비디오 디코더들(210, 310, 및 610)은 소프트웨어 명령어들을 실행하는 하나 이상의 프로세서를 사용하여 구현될 수 있다.

[0082] 본 개시내용의 양태들은 인트라 블록 카피 모드에 대한 참조 블록 위치들을 제한하기 위한 기법들을 제공하고, 비-정사각형 블록 분할들을 갖는 인트라 화상 블록 보상 버퍼를 재사용하기 위한 기법들을 제공한다.

[0083] 블록 기반 보상은 인터 예측 및 인트라 예측에 사용될 수 있다. 인터 예측에 대해, 상이한 화상으로부터의 블록 기반 보상은 움직임 보상으로 알려져 있다. 인트라 예측의 경우, 블록 기반 보상은 또한 동일한 화상 내의 이전에 재구성된 영역으로부터 행해질 수 있다. 동일한 화상 내의 재구성된 영역으로부터의 블록 기반 보상은 인트라 화상 블록 보상, 현재 화상 참조(CPR) 또는 인트라 블록 카피(IBC)로 지칭된다. 동일한 화상 내의 현재 블록과 참조 블록 사이의 오프셋을 지시하는 변위 벡터는 블록 벡터(또는 줄여서 BV)로서 지칭된다. 임의의 값(x 또는 y 방향에서 포지티브 또는 네거티브)일 수 있는 움직임 보상에서의 움직임 벡터와는 상이하게, 블록 벡터는 참조 블록이 이용가능하고 이미 재구성되는 것을 보장하기 위해 몇 가지 제약을 갖는다. 또한, 일부 예들에서, 병렬 처리 고려사항을 위해, 타일 경계 또는 파면 래더 형상 경계(wavefront ladder shape boundary)인 일부 참조 영역이 배제된다.

[0084] 블록 벡터의 코딩은 명시적이거나 암시적일 수 있다. (인터 코딩에서의 AMVP(advanced motion vector prediction) 모드와 유사한) 명시적 모드에서, 블록 벡터와 그의 예측자 사이의 차이가 시그널링된다; 암시적 모드에서, 블록 벡터는 병합 모드에서 움직임 벡터와 유사한 방식으로 예측자(블록 벡터 예측자라고 지칭됨)로부터 복구된다. 일부 구현들에서, 블록 벡터의 해상도는 정수 위치들로 제한되고; 다른 시스템들에서, 블록 벡터는 분수 위치들을 가리키도록 허용된다.

[0085] 일부 예들에서, 블록 레벨에서의 인트라 블록 카피의 사용은 IBC 플래그로서 지칭되는 블록 레벨 플래그를 사용하여 시그널링될 수 있다. 일 실시예에서, IBC 플래그는 현재 블록이 병합 모드에서 코딩되지 않을 때 시그널링된다. 다른 실시예에서, IBC 플래그는 참조 인덱스 접근법을 사용하여 시그널링된다. 디코딩 중인 현재 화상은 그 후 참조 화상으로서 취급된다. 일 예에서, 이러한 참조 화상은 참조 화상들의 리스트의 마지막 위치에 놓인다. 이 특수 참조 화상은 또한 DPB(decoded picture buffer)와 같은 버퍼 내의 다른 시간 참조 화상들과 함께 관리된다.

[0086] 플립된 인트라 블록 카피(참조 블록은 현재 블록을 예측하기 위해 사용되기 전에 수평으로 또는 수직으로 플립됨), 또는 라인 기반 인트라 블록 카피(MxN 코딩 블록 내의 각각의 보상 유닛은 Mx1 또는 1xN 라인임)와 같은, 인트라 블록 카피에 대한 일부 변형들이 또한 존재한다.

[0087] 도 7은 본 개시내용의 일 실시예에 따른 인트라 블록 카피의 예를 도시한다. 현재 화상(700)은 디코딩 중이다. 현재 화상(700)은 재구성된 영역(710)(회색 영역) 및 디코딩될 영역(720)(백색 영역)을 포함한다. 현재 블록(730)은 디코더에 의해 재구성 중이다. 현재 블록(730)은 재구성된 영역(710)에 있는 참조 블록(740)으로부터 재구성될 수 있다. 참조 블록(740)과 현재 블록(730) 사이의 위치 오프셋은 블록 벡터(750)(또는 BV(750))로서 지칭된다.

[0088] 일부 구현 예들에서, 현재 화상의 재구성된 샘플들은 전용 메모리에 버퍼링될 수 있다. 구현 비용을 고려하여, 참조 영역은 전체 프레임만큼 클 수 없고, 전용 메모리의 메모리 크기까지이다. 일부 실시예들에서, 인트라 블록 카피에서의 블록 벡터는 전체 화상이 아니라 단지 일부 이웃 영역을 참조하도록 허용된다. 일 예에서, 전용 메모리의 메모리 크기는 하나의 CTU이고, 그 후 참조 블록이 현재 블록과 동일한 CTU 내에 있을 때, 인트라 블록 카피 모드가 사용될 수 있다. 다른 예에서, 메모리 크기는 2개의 CTU이고, 그 후 참조 블록이 현재 CTU 내에 있거나, 현재 CTU의 좌측에 있는 CTU 내에 있을 때, 인트라 블록 카피 모드가 사용될 수 있다. 참조 블록이 지정된 로컬 영역 외부에 있을 때, 참조 블록이 재구성되었더라도, 참조 샘플들은 일 예에서 인트라 화상 블록 보상을 위해 사용될 수 없다.

[0089] 제한된 참조 영역에 의해, 인트라 블록 카피의 효율은 제한된다. 제한된 참조 영역을 갖는 인트라 블록 카피의 효율을 더욱 개선시킬 필요가 있다.

[0090] VVC의 버전에서와 같은 비디오 코딩 표준의 일부 버전들에서, 화상은 비-중첩 CTU들의 어레이로 분할된다. CTU의 크기는 128x128 루마 샘플들(및 컬러 포맷에 따라 대응하는 크로마 샘플들)로 설정된다. CTU는 이진-트리

(BT) 분열, 삼진-트리(TT) 분열 등과 같은 트리 분할 방법들 중 하나 또는 이들의 조합을 사용하여 코딩 유닛(CU)들로 분열될 수 있다. BT 분열은 부모 블록(parent block)을 수평 또는 수직 방향으로 절반으로 분열한다. BT 분열의 결과적인 2개의 더 작은 분할은 부모 블록에 비해 크기가 절반이다. TT 분열은 부모 블록을 수평 또는 수직 방향으로 3개의 부분으로 분할한다. 그 3개의 중간 부분은 다른 2개의 부분보다 2배 크다. TT 분열의 결과적인 3개의 더 작은 분할은 부모 블록과 비교하여 크기가 1/4, 1/2 및 1/4이다.

[0091] 일부 실시예들에서, 128x128 레벨에서의 블록의 분할들은 1개의 128x128 블록, 2개의 128x64 블록, 2개의 64x128 블록 또는 4개의 64x64 블록과 같은 특정 결과들을 허용하도록 제한된다. 일부 예들에서, 128x64 또는 64x128의 블록 레벨에서, 어느 한쪽(수평 또는 수직) 방향으로의 TT 분열들이 허용되지 않도록 분할들이 추가로 제한된다. 또한, 일 예에서, 128x64 또는 64x128은 임의의 추가 분열이 있을 때 단지 2개의 64x64 블록으로 분열되도록 허용된다. 제한된 분할 타입들은 정규 분할 타입들로서 지칭된다.

[0092] 일부 예들에서, 인트라 블록 카피는 제한된 메모리 요건으로 구현된다. 구현 예들에서, 미래의 인트라 블록 카피 참조를 위해 이전에 코딩된 CU들의 참조 샘플들을 저장하기 위한 지정된 메모리는 참조 샘플 메모리로서 지칭된다. 예를 들어, 인트라 블록 카피 모드에 대한 필터링되지 않은 샘플들을 저장하기 위해 사용되는 참조 샘플 메모리(버퍼라고도 지칭됨)의 최대 크기는 1 CTU(128x128 루마 샘플들 및 대응하는 크로마 샘플들) 크기이다. 이러한 제한된 메모리를 완전히 이용하기 위해, 일부 예들에서, 버퍼는 64x64 기준으로 업데이트된다. 참조 샘플 메모리의 일부 부분들이 현재 CTU로부터의 재구성된 샘플들로 아직 업데이트되지 않았을 때, 메모리의 그 부분들에 저장된 데이터는 일 예에서 실제로 좌측 CTU로부터의 재구성된 샘플들이다. 이러한 샘플들은 인트라 블록 카피 모드에서 현재 블록을 예측하기 위한 참조로서 사용될 수 있다. 그 후, 유효 검색 범위는 현재 CTU 내로부터 좌측 CTU의 일부 부분으로 확장된다.

[0093] 따라서, 일부 실시예들에서, 참조 샘플 메모리는, 참조 샘플 메모리가 현재 CTU에서의 재구성된 샘플들로 업데이트되기 전에, 좌측 CTU 내에 이전에 코딩된 샘플들을 저장하는 데 재사용될 수 있다. 일 실시예에서, 참조 샘플 메모리의 가용성은 64x64 기준으로 업데이트된다. 일 예에서, 64x64 블록 내의 참조 샘플들 중 임의의 것이 업데이트되었다면, 전체 64x64 블록은 현재 CTU로부터의 재구성된 샘플들로 업데이트된 것으로 간주되고 더 이상 업데이트 전에 샘플들을 저장하는 것으로 간주되지 않을 것이다. 일부 구현 예들에서, 메모리(예를 들어, 버퍼 메모리, 온 칩 메모리 등) 내의 연속적인 메모리 공간은 64x64 블록의 재구성된 샘플들을 저장하기 위해 할당되고, 연속적인 메모리 공간으로부터 재구성된 샘플들을 저장 및 리드-백(read-back)하기 위한 어드레스 동작들은 비교적 용이한 계산들로 수행될 수 있다. 일부 예들에서, 64x64 코딩 영역과 같은 코딩 영역의 재구성된 샘플들을 저장하는 연속적인 메모리 공간은 메모리 업데이트 유닛이라고 지칭된다. 메모리 업데이트 유닛 내의 샘플들에 액세스하는 것은 감소된 어드레스 계산들로 수행될 수 있고, 상이한 메모리 업데이트 유닛들 내의 샘플들에 액세스하는 것은 증가된 수의 어드레스 계산들을 요구할 수 있다.

[0094] 도 8a 내지 도 8d는 본 개시내용의 일 실시예에 따른, 64x64 기준으로의 참조 샘플 메모리 업데이트들 및 인트라 블록 카피 모드에 대한 유효 검색 범위들의 예들을 도시한다. 일부 예들에서, 인코더/디코더는 128x128 샘플들과 같은, 하나의 CTU의 샘플들을 저장할 수 있는 캐시 메모리를 포함한다. 또한, 도 8a 내지 8d의 예들에서, CTU는 4개의 64x64 코딩 영역으로 분할된다. 예측을 위한 현재 블록(코딩 영역과 동일하거나 그보다 작을 수 있음)을 갖는 코딩 영역은 현재 코딩 영역("Curr"로 라벨링됨)으로 지칭된다. 예들은 다른 적절한 크기들의 코딩 영역들에 대해 적절하게 수정될 수 있다는 점에 유의한다.

[0095] 도 8a 내지 도 8d 각각은 현재 CTU(820) 및 좌측 CTU(810)를 도시한다. 좌측 CTU(810)는 4개의 코딩 영역(811-814)을 포함하고, 각각의 코딩 영역은 64x64 샘플들의 샘플 크기를 갖는다. 현재 CTU(820)는 4개의 코딩 영역(821-824)을 포함하고, 각각의 코딩 영역은 64x64 샘플들의 샘플 크기를 갖는다. 현재 CTU(820)는 재구성 중인 현재 코딩 영역(라벨 "Curr" 및 수직 스트라이프 패턴에 의해 도시되는 바와 같음)을 포함하는 CTU이다. 좌측 CTU(810)는 현재 CTU(820)의 좌측 상의 바로 인접한 이웃이다. 도 8a 내지 도 8d에서, 회색 블록들은 이미 재구성된 코딩 영역들이고, 백색 블록들은 재구성될 코딩 영역들이라는 점에 유의한다.

[0096] 도 8a에서, 재구성 중인 현재 코딩 영역은 코딩 영역(821)이다. 캐시 메모리는 코딩 영역들(812, 813 및 814)에서의 재구성된 샘플들을 저장하고, 캐시 메모리는 현재 코딩 영역(821)의 재구성된 샘플들을 저장하기 위해 사용될 것이다. 도 8a의 예에서, 현재 코딩 영역(821)에 대한 유효 검색 범위는, 캐시 메모리에 저장된 재구성된 샘플들을 갖는 좌측 CTU(810)에서의 코딩 영역들(812, 813 및 814)을 포함한다. 일 예에서, 캐시 메모리가 현재 코딩 영역(821)으로부터의 하나 이상의 재구성된 샘플로 업데이트될 때, 따라서 코딩 영역(811)은 코딩 영역(811)의 일부 부분이 현재 CTU로부터의 재구성된 샘플들에 의해 업데이트되었다는 사실로 인해 CPR 예측을 위

해 캐시 메모리에서 이용가능하지 않는 것("X"로 마킹됨)으로 간주된다. 일 실시예에서, 코딩 영역(811)의 재구성된 샘플들은, 캐시 메모리보다 느린 액세스 속도를 갖는 메인 메모리에 저장된다(예를 들어, 코딩 영역(821)의 재구성 전에 캐시 메모리로부터 메인 메모리로 카피된다)는 점에 유의한다.

[0097] 도 8b에서, 재구성 중인 현재 코딩 영역은 코딩 영역(822)이다. 캐시 메모리는 코딩 영역들(813, 814 및 821)에 재구성된 샘플들을 저장하고, 캐시 메모리는 현재 코딩 영역(822)의 재구성된 샘플들을 저장하기 위해 사용될 것이다. 도 8b의 예에서, 현재 코딩 영역(822)에 대한 유효 검색 범위는 캐시 메모리에 저장된 재구성된 샘플들을 갖는 현재 CTU(820) 내의 좌측 CTU(810 및 821)의 코딩 영역들(813 및 814)을 포함한다. 일 예에서, 캐시 메모리가 현재 코딩 영역(822)으로부터의 하나 이상의 재구성된 샘플들로 업데이트될 때, 따라서 코딩 영역(812)은 코딩 영역(812)의 일부 부분이 현재 CTU로부터의 재구성된 샘플들에 의해 업데이트되었다는 사실로 인해, CPR 예측을 위해 캐시 메모리에서 이용가능하지 않는 것("X"로 마킹됨)으로 간주된다. 일 실시예에서, 코딩 영역(812)의 재구성된 샘플들은 캐시 메모리보다 느린 액세스 속도를 갖는 메인 메모리에 저장된다(예를 들어, 코딩 영역(822)의 재구성 전에 캐시 메모리로부터 메인 메모리로 카피된다).

[0098] 도 8c에서, 재구성 중인 현재 코딩 영역은 코딩 영역(823)이다. 캐시 메모리는 코딩 영역들(814, 821 및 822)에 재구성된 샘플들을 저장하고, 캐시 메모리는 현재 코딩 영역(823)의 재구성된 샘플들을 저장하기 위해 사용될 것이다. 도 8c의 예에서, 현재 블록(823)에 대한 유효 검색 범위는 캐시 메모리에 저장된 재구성된 샘플들을 갖는 현재 CTU(820) 내의 코딩 영역들(821 및 822) 및 좌측 CTU(810) 내의 코딩 영역(814)을 포함한다. 일 예에서, 캐시 메모리가 현재 코딩 영역(823)으로부터의 하나 이상의 재구성된 샘플로 업데이트될 때, 따라서 코딩 영역(813)은 코딩 영역(813)의 일부 부분이 현재 CTU로부터의 재구성된 샘플들에 의해 업데이트되었다는 사실로 인해, CPR 예측을 위해 캐시 메모리에서 이용가능하지 않는 것("X"로 마킹됨)으로 간주된다. 일 실시예에서, 코딩 영역(813)의 재구성된 샘플들은 캐시 메모리보다 느린 액세스 속도를 갖는 메인 메모리에 저장된다(예를 들어, 코딩 영역(823)의 재구성 전에 캐시 메모리로부터 메인 메모리로 카피된다).

[0099] 도 8d에서, 재구성 중인 현재 코딩 영역은 코딩 영역(824)이다. 캐시 메모리는 코딩 영역들(821, 822 및 823)에 재구성된 샘플들을 저장하고, 캐시 메모리는 현재 코딩 영역(824)의 재구성된 샘플들을 저장하기 위해 사용될 것이다. 도 8d의 예에서, 현재 코딩 영역(824)에 대한 유효 검색 범위는 캐시 메모리에 저장된 재구성된 샘플들을 갖는 현재 CTU(820) 내의 블록들(821, 822 및 823)을 포함한다. 일 예에서, 캐시 메모리가 현재 코딩 영역(824)으로부터의 하나 이상의 재구성된 샘플들로 업데이트될 때, 따라서 코딩 영역(814)은 코딩 영역(814)의 일부 부분이 현재 CTU로부터의 재구성된 샘플들에 의해 업데이트되었다는 사실로 인해 CPR 예측을 위해 캐시 메모리에서 이용가능하지 않는 것("X"로 마킹됨)으로 간주된다. 일 실시예에서, 코딩 영역(814)의 재구성된 샘플들은 캐시 메모리보다 느린 액세스 속도를 갖는 메인 메모리에 저장된다(예를 들어, 코딩 영역(824)의 재구성 전에 캐시 메모리로부터 메인 메모리로 카피된다).

[0100] 도 8a 내지 도 8d의 예에서, 4x64x64 루마 샘플들(대응하는 크로마 샘플들과 함께)의 전체 메모리가 인트라 블록 카피 보상에 사용된다는 것에 유의한다. 도 8a 내지 도 8d의 예에서 사용되는 기법은 3x64x64, 2x64x64 루마 샘플들의 전체 메모리와 같은 다른 메모리 크기들에 대해 적절히 수정될 수 있다.

[0101] 일부 실시예들에서, 128x128 레벨 이하에서 삼진 트리 분할이 허용되고, 분할 결과들은 비-정사각형 블록들일 수 있다는 점에 유의한다.

[0102] 일부 예들에서, 128x128 레벨에서, CTU는 삼진-트리 분열을 사용하여 분할될 수 있다. 수평 TT 분열의 경우, 분할의 결과적인 블록들은 128x32, 128x64 및 128x32일 것이다. 수직 분열의 경우, 분할의 결과적인 블록들은 32x128, 64x128 및 32x128일 것이다.

[0103] 또한, 일부 예들에서, 128x64/64x128 레벨에서, 삼진-트리 분열이 사용될 수 있고, 결과적인 블록들은 비-정사각형 블록들일 수 있다.

[0104] 도 9a 내지 도 9h는 비-정사각형 블록들을 초래할 수 있는 분할들의 일부 예들을 도시한다.

[0105] 본 개시내용의 일부 양태들에 따르면, 128x128 레벨에서의 TT 분열 등과 같은 일부 분할 타입들은 64x64 코딩 영역에 포함될 수 없거나 하나 이상의 64x64 코딩 영역을 포함할 수 없는 코딩 블록을 초래할 수 있다. 이러한 분할 타입들은 유연한 분할 타입들이라고 지칭한다.

[0106] 이 유연한 분할 타입들이 그러한 정규 분할 타입들의 상단에 허용될 때. 참조 샘플 업데이트 프로세스는 더 이상 64x64 코딩 영역들로 하나씩 정렬되지 않을 수 있다. 위에 언급된 바와 같이 참조 샘플 메모리 버퍼 재사용

전략을 가능하게 하는 방법들을 개발할 필요가 있다.

[0107] 본 개시내용의 양태에 따르면, 특정 참조 영역 제약들은 인트라 블록 카피 성능을 개선하기 위해 사용될 수 있다. 보다 구체적으로, 일 예에서, 참조 샘플 메모리 내의 참조 샘플 블록의 위치는 동일한 64x64 코딩 영역 내에 있도록 제한된다. 일부 실시예들에서, 참조 샘플 메모리의 크기는 2개의 64x64 루마 샘플들(대응하는 크로마 샘플들과 함께)로 고정된다. 본 개시내용에서 사용되는 기법들은 3개의 64x64 루마 샘플들 플러스 대응하는 크로마 샘플들, 4개의 64x64 루마 샘플들 플러스 대응하는 크로마 샘플들(즉, 1개의 CTU 크기) 등을 사용하는 것과 같이, 상이한 참조 샘플 메모리 크기들에 사용하기 위해 적절히 수정될 수 있다는 점에 유의한다.

[0108] 일반적으로, IBC 코딩된 블록의 크기는 임의의 정규 인터 코딩된 블록만큼 클 수 있다. 본 개시내용의 일 양태에 따르면, 참조 샘플 메모리를 더 효율적으로 이용하기 위해, IBC 코딩된 블록의 크기가 제한된다. 일 예에서, IBC 코딩된 블록의 크기는 폭 또는 높이에서 64개의 루마 샘플과 같은 더 높은 한계들보다 크지 않도록 제한된다(대응하는 크기 제약들이 컬러 포맷에 따라, 크로마 샘플들에 적용되고; 예를 들어, 4:2:0 포맷에서, IBC 모드에서의 크로마 블록의 크기는 각각의 측면에서 32개의 샘플보다 크지 않아야 함). 또한, 일부 예들에서, IBC 코딩된 블록의 크기는 폭 또는 높이에서 32개의 루마 샘플과 같은 하한들보다 낮지 않도록 제한된다. 64x64 루마 샘플들의 최대 IBC 크기가 설명에서 사용되지만, 기법들은 다른 최대 IBC 크기에 대해 적절히 수정될 수 있다는 점에 유의한다.

[0109] 이하의 설명에서, 인트라 블록 카피 참조 샘플들을 저장하기 위해 사용되는 참조 샘플 메모리의 최대 크기는 2개의 64x64 루마 샘플 및 대응하는 크로마 샘플들을 저장하기 위한 메모리 크기에 대응한다.

[0110] 본 개시내용의 일부 양태들에 따르면, 참조 샘플 메모리는 2개의 코딩 영역의 샘플 정보를 저장하기 위한 크기를 가지며, 각각의 코딩 영역은 64x64 루마 샘플들의 크기를 가지며, 샘플 정보는 64x64 루마 샘플들의 루마 샘플 정보 및 대응하는 크로마 샘플들의 크로마 샘플 정보를 포함한다. 그 후, 참조 샘플 메모리는 현재 CTU 내의 현재 코딩 영역의 샘플 정보, 및 일부 이전에 코딩된 64x64 코딩 영역들 중 하나인 다른 코딩 영역의 샘플 정보를 저장한다.

[0111] 도 10a 및 도 10b는 코딩 순서들 및 참조 샘플 메모리 사용의 2가지 예를 도시한다. 도 10a 및 10b에서, 회색을 갖는 영역들은 코딩된 코딩 영역들(예를 들어, 64x64 루마 샘플들)이고; 백색을 갖는 영역들은 아직 코딩되지 않은 영역들이고; 그리고, 사선들을 갖는 영역(들)은 현재 코딩 블록이 위치하는 코딩 영역(들)이다. 또한, 도 10a 및 10b에서는, 참조 샘플 메모리 내의 2개의 코딩 영역이 점선 박스(들)로 도시된다.

[0112] 도 10a에서, 일 예에서, CTU는 128x128 레벨에서 수평 이진 분열 또는 쿼드-트리 분열 중에 있다. 그 후, 64x64 코딩 영역들의 코딩 순서는 (1001) - (1004)를 따른다.

[0113] 도 10b에서, 일 예에서, CTU는 128x128 레벨에서 수직 이진 분열 중에 있다. 그 후, 64x64 코딩 영역들의 코딩 순서는 (1011) - (1014)를 따른다.

[0114] 도 10a 및 10b의 예들에 대해, 현재 코딩 블록이 현재 CTU 내의 4개의 64x64 코딩 영역 중 하나에 위치할 때, 참조 샘플 메모리는 인트라 블록 카피 모드의 참조가 되도록 할당된 다른 64x64 코딩 영역을 저장할 수 있다.

[0115] 도 10a 및 도 10b는 단지 예들이고, 다른 가능한 참조 영역 할당들이 또한 가능하다는 점에 유의한다. 일 실시예에서, 우측 상단 64x64 코딩 영역은 도 10a 및 10b에 도시된 바와 같이 우측 하단 64x64 코딩 영역에 대한 참조 영역으로서 사용된다. 다른 실시예에서, 좌측 상단 64x64 코딩 영역은 우측 하단 64x64 코딩 영역에 대한 참조 영역으로서 사용된다.

[0116] 이상적으로, 참조 샘플 메모리 내의 재구성된 샘플들은 인트라 블록 카피 모드에서 현재 블록을 예측하기 위해 사용될 수 있다. 그러나, 때로는 참조 블록 내의 샘플들이 2개의 상이한 64x64 코딩 영역에서 나올 수 있다.

[0117] 도 11a 및 11b는 2개의 상이한 64x64 코딩 영역에서 나오는 참조 블록 내의 샘플의 예들을 도시한다. 도 11a에서, 참조 블록의 일부는 하나의 64x64 코딩된 코딩 영역에서 나오는 반면, 다른 부분은, 이미 재구성되었어도, 다른 64x64 이전에 코딩된 코딩 영역에 속한다. 도 11b에서, 참조 블록의 일부는 하나의 64x64 코딩된 영역에서 나오는 반면, 다른 부분은, 이미 재구성되었어도, 현재 64x64 코딩 영역에 속한다. 참조 샘플 메모리 내의 64x64 코딩 영역들 각각은 개별적으로 동작될 것이기 때문에, 메모리 내의 2개의 64x64 코딩 영역은 물리적으로 연결되지 않을 수 있다. 도 11a 및 11b에 도시된 바와 같이, 상이한 코딩 영역들에서의 참조 블록의 이러한 위치들은 바람직하지 않은 여분의 메모리 액세스 또는 동작들을 가져올 수 있다.

[0118] 본 개시내용의 일부 양태들에 따르면, 특정 비디오/이미지 코딩 시스템에서 인트라 블록 카피 작업을 행하는 다

른 적절한 블록 벡터 제약들과 함께 추가적인 제약들이 사용된다. 인트라 블록 카피 모드에 대한 전체 참조 블록이 동일한 64x64 코딩 영역 내에 위치해야 하도록, 인트라 블록 카피 모드의 블록 벡터에 추가적인 제약들이 부과된다. 64x64 코딩 영역은 128x128 루마 샘플들(플러스 대응하는 크로마 샘플들)을 갖는 각각의 CTU가 4개의 비-중첩 64x64 영역으로 분할될 수 있다는 사실을 지칭한다. 이러한 64x64 영역들 각각은 일부 예들에서 제약들에서의 64x64 코딩 영역으로 간주된다.

[0119] 일 실시예에서, 예를 들어, CTU의 상단-레벨 코너가 (0, 0)라고 가정하여, 참조 블록의 좌측 상단 코너는 (Ref_TLx, Ref_TLy)로서 표기되고; 참조 블록의 우측 하단 코너는 (Ref_BRx, Ref_BRy)로 표기된다. 추가적인 제약들은 다음의 조건들 (수학식 1) 및 (수학식 2)가 충족되어야 하는 비트스트림 적합성 요건들의 형태이다:

[0120] Ref_BRx/64 = Ref_TLx/64 (수학식 1)

[0121] Ref_BRy/64 = Ref_TLy/64 (수학식 2)

[0122] (수학식 1)이 만족될 때, Ref_BRx/64의 정수 부분은 Ref_TLx/64의 정수 부분과 동일하다. (수학식 2)가 만족될 때, Ref_BRy/64의 정수 부분은 Ref_TLy/64의 정수 부분과 동일하다. 분할 동작들은 6비트의 우측 시프팅에 의해 수행될 수 있다는 점에 유의한다.

[0123] 일부 실시예들에서, 블록 벡터가 참조 블록 위치를 가리킬 때- 이 참조 블록 내의 모든 샘플들이 동일한 64x64 코딩 영역에 속하는 것은 아님 -, 블록 벡터는 동일한 64x64 코딩 영역 요건을 충족시키는 인근 참조 블록 위치로 클리핑(clip)될 수 있다. 일 실시예에서, 클리핑은 (적어도 2개의 64x64 영역을 커버하는) 원래의 참조 블록 위치를 수직으로 또는 수평으로 하나의 방향을 향해 이동시키는 것을 지칭한다. 일 예에서, 참조 블록과 현재 블록 사이의 거리를 감소시키는 방향으로 이동이 있다. 또한, 동일한 64x64 샘플 영역에 속하는 참조 샘플들의 비율을 증가시키고 결국 전체 참조 블록이 동일한 64x64 영역에 속하게 하는 방향으로 이동이 있다.

[0124] 도 11a에 도시된 예의 경우, 참조 블록은 모든 샘플들이 상단 64x64 코딩 영역에 속할 때까지 수직으로 위로 이동될 수 있다. 도 11a의 예에서, 참조 블록을 아래쪽으로 이동시키는 것은 2개의 블록 사이의 거리를 증가시킬 수 있고, 참조 블록을 수평으로 이동시키는 것은 각각의 64x64 코딩 영역에서의 샘플들의 비율을 변경하지 않는다는 점에 유의한다.

[0125] 도 11b에 도시된 예의 경우, 참조 블록은 모든 샘플들이 현재 64x64 코딩 영역에 속할 때까지 수평 우측으로 이동될 수 있다. 도 11b의 예에서, 참조 블록을 좌측으로 이동시키는 것이 2개의 블록 사이의 거리를 증가시킬 수 있고; 수직으로 이동시키는 것은 각각의 64x64 코딩 영역에서 샘플들의 비율을 변경하지 않는다는 점에 유의한다.

[0126] 일부 실시예들에서, 블록 벡터가 참조 블록 위치를 가리킬 때- 이 참조 블록 내의 모든 샘플들이 동일한 64x64 코딩 영역에 속하는 것은 아님 -, 64x64 코딩 영역 외부의 샘플들은 샘플 패딩에 의해 생성될 수 있다(64x64 코딩 영역의 경계 상의 샘플들을 수직으로 및/또는 수평으로 복제함). 패딩 프로세스는 움직임 보상에서 참조 화상에 대한 경계 확장과 유사한 방식으로 수행될 수 있다.

[0127] 일 예에서, 참조 블록의 좌측 상단 코너는 64x64 코딩 영역에 위치하고, 그 후 64x64 코딩 영역은 참조 블록이 나와야 하는 코딩 영역으로서 간주된다. 참조 블록의 임의의 부분이 그 코딩 영역 외부에 있는 경우, 그 부분의 샘플들을 결정하기 위해 패딩이 사용될 수 있다.

[0128] 다른 예에서, 참조 블록의 우측 하단 코너는 64x64 코딩 영역에 위치하고, 그 후 64x64 코딩 영역은 참조 블록이 나와야 하는 곳으로서 고려된다. 참조 블록의 임의의 부분이 그 코딩 영역 외부에 있는 경우, 그 부분의 샘플들을 결정하기 위해 패딩이 수행될 수 있다.

[0129] 다른 예에서, 참조 블록의 좌측 상단 코너는 64x64 코딩 영역에 위치하고, 64x64 코딩 영역은 참조 블록이 어느 코딩 영역에서 나오는지를 결정하기 위해 먼저 사용된다. 위치가 유효한 참조 위치가 아닐 때, 그 후 참조 블록의 우측 하단 코너는 참조 블록이 어느 코딩 영역에서 나오는지를 결정하는 것으로 간주된다.

[0130] 다른 예에서, 참조 블록의 우측 하단 코너는 64x64 코딩 영역에 위치하고, 64x64 코딩 영역은 참조 블록이 어느 코딩 영역에서 나오는지를 결정하기 위해 먼저 사용된다. 위치가 유효한 참조 위치가 아닌 경우, 참조 블록의 좌측 상단 코너는 참조 블록이 어느 코딩 영역에서 나오는지를 결정하는 것으로 간주된다.

[0131] 다른 실시예에서, 블록 벡터가 참조 블록 위치를 가리킬 때- 여기서 참조 블록 내의 모든 샘플들이 유효 참조

영역(현재 64x64 코딩 영역 또는 이전에 코딩된 64x64 코딩 영역(들)) 외부에 있음 -, 64x64 코딩 영역 외부의 샘플들은 샘플 패딩에 의해 생성될 수 있다(가장 가까운 유효 참조 영역의 경계 상의 샘플들을 수직으로 및/또는 수평으로 복제함). 패딩 프로세스는 움직임 보상에서의 참조 화상에 대한 경계 확장과 동일한 방식으로 수행될 수 있으며, 여기서 경계는 허용된 참조 영역(들)의 경계로서 간주될 수 있다.

[0132] 본 개시내용의 양태들은 또한 인트라 블록 카피를 위한 버퍼 재사용 메커니즘 및 유연한 블록 분할 전략들을 제공한다.

[0133] 일부 실시예들에서, 인트라 블록 카피가 주어진 코딩 레벨에 대해(예컨대 시퀀스에 대해, 화상에 대해 또는 슬라이스/타일/타일 그룹에 대해) 사용될 때, 128x32/32x128과 같은 유연한 블록 분할 타입들이 사용될 수 없다.

[0134] 실시예에서, 주어진 코딩 레벨에 대해(예컨대 시퀀스에 대해, 화상에 대해 또는 슬라이스/타일/타일 그룹에 대해), 유연한 블록 분할 타입들이 사용되지 않을 때, 인트라 블록 카피 모드는 참조 샘플 메모리 재사용 전략들을 사용할 수 있다. 예를 들어, 64x64 코딩 영역 기반 업데이트 프로세스 및 버퍼 재사용 전략들이 사용될 수 있다.

[0135] 일 실시예에서, 주어진 코딩 레벨에 대해(예컨대 시퀀스에 대해, 화상에 대해 또는 슬라이스/타일/타일 그룹에 대해), 유연한 블록 분할 타입들이 사용 중일 때, 인트라 블록 카피 모드는 위의 개시내용에 개시된 참조 샘플 메모리 재사용 전략들을 사용할 수 없다. 대신에, 일 예에서, 블록 벡터는 현재 CTU의 재구성된 부분들을 참조하도록 제한된다.

[0136] 일 실시예에서, 주어진 코딩 레벨에 대해(예컨대 시퀀스에 대해, 화상에 대해 또는 슬라이스/타일/타일 그룹에 대해), 위에서 언급된 유연한 블록 분할 타입들이 사용 중일 때, 위에서 언급된 참조 샘플 메모리 재사용 전략들이 수정될 수 있다.

[0137] 일부 예들에서, 코딩 블록이 CTU(128x128) 레벨에서 제1 TT 분할(예를 들어, 좌측 32x128 또는 상단 128x32) 또는 BT 분할(상단 128x64 또는 좌측 64x128) 또는 QT 분할(좌측 상단 64x64)에 있을 때, 좌측 CTU 내의 참조 샘플들의 일부가 사용될 수 있다.

[0138] 도 12a 내지 도 12f는 참조 샘플 메모리 재사용 전략들의 예들을 도시한다. 도 12a 내지 도 12f의 예들 각각에서, 현재 코딩 블록은 사선들로 도시되고, 이용가능한 참조 영역은 회색으로 도시된다.

[0139] 일 예에서, 도 12a 내지 도 12c의 예들에서 회색으로 도시된 바와 같이, 좌측 CTU에서의 우측 상단 코너 64x64(샘플들) 영역은 이용가능한 참조 영역이다.

[0140] 다른 예에서, 도 12d 내지 도 12f의 예들에서 회색 컬러로 도시된 바와 같이, 좌측 CTU에서의 최우측의 32x128(샘플들) 영역은 이용가능한 참조 영역이다.

[0141] 다른 예에서, 좌측 CTU에서의 상단 128x32 영역은 이용가능한 참조 영역이다.

[0142] 다른 예에서, 영역의 선택은 좌측 CTU의 분할들에 의해 결정된다. 예를 들어, 좌측 CTU가 128x128 레벨에서 쿼드트리 분할을 사용할 때, 그 후 좌측 CTU에서의 우측 상단 64x64(샘플들) 영역은 현재 CTU의 제1 분할에서의 코딩 블록을 예측하기 위해 사용될 수 있다.

[0143] 다른 예에서, 총 3x64x64 크기의 메모리가 사용될 때, 현재 CTU의 제1 분할에서의 코딩 블록에 대해, 좌측 CTU로부터의 참조 샘플들을 저장하기 위해 사용될 수 있는 메모리의 크기는 64x64 크기이다(반드시 정사각형일 필요는 없음). 메모리 공간은 반드시 샘플들을 정사각형 형상으로 저장하는 것은 아니라는 점에 유의한다.

[0144] 다른 실시예에서, 총 4x64x64 크기의 메모리가 사용될 때, 현재 CTU의 제1 분할에서의 코딩 블록에 대해, 좌측 CTU로부터의 참조 샘플들을 저장하기 위해 사용될 수 있는 메모리의 크기는 크기가 2x64x64이다(반드시 2개의 정사각형일 필요는 없음). 메모리 공간은 반드시 2개의 정사각형으로 샘플들을 저장하는 것은 아니라는 점에 유의한다.

[0145] 일부 실시예들에서, 유연한 블록 분할 타입들 및 인트라 블록 카피 모드가 둘 다 주어진 코딩 레벨에서(예를 들어, 시퀀스에 대해, 화상에 대해 또는 슬라이스/타일/타일 그룹에 대해) 사용될 때, 기법들은 메모리 요건이 동일하게 유지되거나 감소되면서 인트라 블록 카피의 검색 범위는 효과적으로 증가될 수 있도록 참조 샘플 메모리를 재사용하기 위해 사용된다.

[0146] 일부 예들에서, 이전에 64x64 기반 업데이트 프로세스를 사용하는 대신에, 32x32 또는 32x64 또는 64x32 또는

16x128의 형상들에서와 같이, 더 많은 블록 형상들을 갖는 더 작거나 동일한 입도(granularity) 업데이트 프로세스가 제안될 수 있다. 일 예에서, 업데이트 프로세스는 32x32(샘플들) 유닛을 사용할 수 있고, 현재 CTU는 128x128 레벨에서 TT를 사용하여 분할된다.

[0147] 도 13a 내지 도 13c는 32x32(샘플들) 유닛을 사용하는 업데이트 프로세스의 일 예를 나타내고, 도 13d 내지 도 13f는 32x32(샘플들)를 사용하는 업데이트 프로세스의 다른 예를 나타낸다. 각각의 32x32(샘플들) 유닛은 작은 정사각형으로서 도시된다. 도 13a 내지 도 13f의 예들에서, 참조 샘플 메모리 크기는 총 128x128이다. 이어서, 3x64x64 샘플들의 제1 메모리 공간은 인트라 블록 카피 참조 목적을 위한 참조 샘플들일 수 있는 샘플들을 저장하기 위해 할당되고, 64x64 샘플들의 제2 메모리 공간은 현재 64x64 크기의 유닛에 대해 할당된다(반드시 정사각형일 필요는 없고, 루마 샘플들의 총수는 64x64 샘플들과 동일하다). 각각의 64x64 또는 64x128 크기의 분할 및 그의 서브-분할들(정사각형일 필요는 없고, 루마 샘플들의 총 수는 64x64와 동일함)에 대해, 현재 CTU에서의 재구성된 샘플들 및 도 13a 내지 도 13f에서 회색 영역들로 표시된 것과 같은 일부 다른 이전에 코딩된 영역들은 참조 샘플 메모리에 저장될 수 있고 인트라 블록 카피 예측을 위해 참조될 수 있다.

[0148] 도 14는 본 개시내용의 일 실시예에 따른 프로세스(1400)를 약술하는 흐름도를 도시한다. 프로세스(1400)는, 재구성 중인 블록에 대한 예측 블록을 생성하기 위해, 인트라 모드에서 코딩된 블록의 재구성에서 사용될 수 있다. 다양한 실시예들에서, 프로세스(1400)는, 단말 디바이스들(110, 120, 130 및 140) 내의 처리 회로, 비디오 인코더(203)의 기능들을 수행하는 처리 회로, 비디오 디코더(210)의 기능들을 수행하는 처리 회로, 비디오 디코더(310)의 기능들을 수행하는 처리 회로, 비디오 인코더(403)의 기능들을 수행하는 처리 회로 등과 같은 처리 회로에 의해 실행된다. 일부 실시예들에서, 프로세스(1400)는 소프트웨어 명령어들로 구현되고, 따라서 처리 회로가 소프트웨어 명령어들을 실행할 때, 처리 회로는 프로세스(1400)를 수행한다. 프로세스는 (S1401)에서 시작되어 (S1410)으로 진행한다.

[0149] (S1410)에서, 현재 블록의 예측 정보가 코딩된 비디오 비트스트림으로부터 디코딩된다. 예측 정보는 인트라 블록 카피 모드를 지시한다.

[0150] (S1420)에서, 현재 블록과 동일한 화상에서의 참조 블록을 가리키는 블록 벡터가 결정된다.

[0151] (S1430)에서, 참조 블록이 완전히 코딩 영역 내에 있는지를 결정하기 위해 참조 블록이 체크된다. 일부 실시예들에서, 제한된 저장 공간을 갖는 참조 샘플 메모리는 이전 코딩된 샘플들을 저장하기 위해 재사용된다. 예를 들어, 참조 샘플 메모리가 현재 CTU에서의 재구성된 샘플들로 업데이트되기 전에, 참조 샘플 메모리는 현재 CTU의 좌측 CTU에 샘플들을 저장한다. 참조 샘플 메모리는 코딩 영역의 유닛으로 현재 CTU에서의 재구성된 샘플들로 업데이트된다. 일 실시예에서, 참조 샘플 메모리의 가용성은 64x64 기준으로 업데이트된다. 일 예에서, 64x64 블록 내의 참조 샘플들 중 임의의 것이 업데이트되었다면, 전체 64x64 블록은 현재 CTU로부터의 재구성된 샘플들로 업데이트된 것으로 간주되고 더 이상 업데이트 전에 샘플들을 저장하는 것으로 간주되지 않을 것이다. 일 실시예에서, 128x128 샘플들의 CTU는 코딩 영역들이라고 지칭될 수 있는 4개의 64x64 영역을 포함한다.

[0152] 일 실시예에서, 참조 블록의 다수의 코너는 참조 블록이 완전히 코딩 영역 내에 있는지를 결정하기 위해 체크된다. 참조 블록이 완전히 코딩 영역에 있을 때, 프로세스는 (S1450)으로 진행한다; 그렇지 않으면, 프로세스는 (S1440)으로 진행한다.

[0153] (S1440)에서, 참조 블록은 코딩 영역에서 적절히 제한된다. 일부 실시예들에서, 참조 블록은 코딩 영역 내로 적절히 이동된다. 일부 실시예들에서, 코딩 영역에 있지 않는 일부 참조 샘플들은 코딩 영역에 있는 다른 참조 샘플로 패딩될 수 있다. 그 후, 일 예에서, 현재 블록의 재구성을 위해 참조 샘플들을 검색하기 위한 메모리 액세스는 단지 하나의 메모리 업데이트 유닛만을 액세스함으로써 달성될 수 있다.

[0154] (S1450)에서, 참조 샘플 메모리로부터 검색되는 참조 블록의 재구성된 샘플들에 기초하여 현재 블록이 재구성된다. 예를 들어, 참조 샘플 메모리는 참조 블록의 재구성된 샘플들을 검색하기 위해 액세스되고, 그 후 현재 블록의 샘플들은 참조 샘플 메모리로부터 검색되는 재구성된 샘플들에 기초하여 재구성된다. 참조 블록은 참조 블록의 참조 샘플들이 완전히 하나의 코딩 영역에 있거나, 하나의 코딩 영역에 기초하여 결정될 수 있는 것으로 제한되기 때문에, 참조 샘플들을 검색하기 위한 어드레스 동작들은 용이하고 높은 효율로 수행될 수 있다. 그 후, 프로세스는 (S1499)로 진행하고 종료된다.

[0155] 위에서 설명된 기법들은 컴퓨터 판독가능 명령어들을 사용하여 컴퓨터 소프트웨어로서 구현되고 하나 이상의 컴퓨터 판독가능 매체에 물리적으로 저장될 수 있다. 예를 들어, 도 15는 개시된 주제의 특정 실시예들을 구현하기에 적합한 컴퓨터 시스템(1500)을 도시한다.

[0156] 컴퓨터 소프트웨어는, 하나 이상의 컴퓨터 중앙 처리 유닛(CPU)들, 그래픽 처리 유닛(GPU)들 등에 의해, 직접, 또는 해석, 마이크로-코드 실행 등을 통해 실행될 수 있는 명령어들을 포함하는 코드를 생성하기 위해 어셈블리, 컴파일, 링킹, 또는 유사한 메커니즘들이 수행될 수 있는 임의의 적합한 머신 코드 또는 컴퓨터 언어를 사용하여 코딩될 수 있다.

[0157] 명령어들은, 예를 들어, 개인용 컴퓨터, 태블릿 컴퓨터, 서버, 스마트폰, 게이밍 디바이스, 사물 인터넷 디바이스 등을 포함하여, 다양한 타입의 컴퓨터들 또는 그것의 컴포넌트들 상에서 실행될 수 있다.

[0158] 컴퓨터 시스템(1500)에 대한 도 15에 도시된 컴포넌트들은 사실상 예시적인 것이고, 본 개시내용의 실시예들을 구현하는 컴퓨터 소프트웨어의 사용 또는 기능성의 범위에 대한 임의의 제한을 암시하도록 의도되지 않는다. 컴포넌트들의 구성이 컴퓨터 시스템(1500)의 예시적인 실시예에서 예시된 컴포넌트들 중 임의의 하나 또는 이들의 조합과 관련하여 임의의 종속성 또는 요건을 갖는 것으로 해석되어서도 안 된다.

[0159] 컴퓨터 시스템(1500)은 특정 휴먼 인터페이스 입력 디바이스들을 포함할 수 있다. 그러한 휴먼 인터페이스 입력 디바이스는, 예를 들어, 촉각 입력(예컨대: 키스트로크, 스와이프, 데이터 글러브 움직임), 오디오 입력(예컨대: 음성, 손뼉), 시각적 입력(예컨대, 제스처), 후각적 입력(묘사되지 않음)을 통한 하나 이상의 인간 사용자에 의한 입력에 응답할 수 있다. 휴먼 인터페이스 디바이스들은 또한 오디오(예컨대: 음성, 음악, 주변 사운드), 이미지들(예컨대: 스캐닝된 이미지들, 스틸 이미지 카메라로부터 획득된 사진 이미지들), 비디오(예컨대 2차원 비디오, 입체적 비디오를 포함하는 3차원 비디오)와 같은, 인간에 의한 의식적인 입력과 반드시 직접적으로 관련되는 것은 아닌 특정 미디어를 캡처하기 위해 사용될 수 있다.

[0160] 입력 휴먼 인터페이스 디바이스들은: 키보드(1501), 마우스(1502), 트랙패드(1503), 터치 스크린(1510), 데이터-글러브(도시되지 않음), 조이스틱(1505), 마이크로폰(1506), 스캐너(1507), 카메라(1508) 중 하나 이상(각각의 하나만이 묘사됨)을 포함할 수 있다.

[0161] 컴퓨터 시스템(1500)은 특정 휴먼 인터페이스 출력 디바이스들을 또한 포함할 수 있다. 그러한 휴먼 인터페이스 출력 디바이스들은, 예를 들어, 촉각 출력, 사운드, 광, 및 냄새/맛을 통해 하나 이상의 인간 사용자의 감각들을 자극하고 있을 수 있다. 그러한 휴먼 인터페이스 출력 디바이스들은 촉각 출력 디바이스들(예를 들어 터치-스크린(1510), 데이터-글러브(도시되지 않음), 또는 조이스틱(1505)에 의한 촉각 피드백이지만, 입력 디바이스들로서 역할하지 않는 촉각 피드백 디바이스들도 있을 수 있음), 오디오 출력 디바이스들(예컨대: 스피커들(1509), 헤드폰들(묘사되지 않음)), 시각적 출력 디바이스들(예컨대 CRT 스크린들, LCD 스크린들, 플라즈마 스크린들, OLED 스크린들을 포함하는 스크린들(1510), 각각은 터치-스크린 입력 능력이 있거나 없고, 각각은 촉각 피드백 능력이 있거나 없고- 이들 중 일부는 스테레오그래픽 출력; 가상 현실 안경(묘사되지 않음), 홀로그래픽 디스플레이들 및 연기 탱크들(묘사되지 않음)), 및 프린터들(묘사되지 않음)과 같은 수단을 통해 2차원 시각적 출력 또는 3개보다 많은 차원의 출력을 출력할 수 있다.

[0162] 컴퓨터 시스템(1500)은 인간 액세스 가능한 저장 디바이스들 및 그것들과 연관된 매체들, 예컨대 CD/DVD 등의 매체(1521)를 갖는 CD/DVD ROM/RW(1520)를 포함하는 광학 매체, 썸-드라이브(1522), 이동식 하드 드라이브 또는 솔리드 스테이트 드라이브(1523), 테이프 및 플로피 디스크(묘사되지 않음)와 같은 레거시 자기 매체, 보안 동글들(묘사되지 않음)과 같은 특수화된 ROM/ASIC/PLD 기반 디바이스들 등을 또한 포함할 수 있다.

[0163] 본 기술분야의 통상의 기술자들은 또한, 현재 개시된 주제와 관련하여 사용되는 용어 "컴퓨터 판독가능 매체"가 송신 매체, 반송파들, 또는 다른 일시적 신호들을 포함하지 않는다는 것을 이해해야 한다.

[0164] 컴퓨터 시스템(1500)은 하나 이상의 통신 네트워크에 대한 인터페이스를 또한 포함할 수 있다. 네트워크들은 예를 들어 무선, 와이어라인, 광학일 수 있다. 네트워크들은 추가로 로컬, 광역, 대도시, 차량 및 산업, 실시간, 지연-허용(delay-tolerant) 등일 수 있다. 네트워크들의 예들은 로컬 영역 네트워크들, 예컨대 이더넷, 무선 LAN들, GSM, 3G, 4G, 5G, LTE 등을 포함하는 셀룰러 네트워크들, 케이블 TV, 위성 TV 및 지상파 브로드캐스트 TV를 포함하는 TV 와이어라인 또는 무선 광역 디지털 네트워크들, CANBus를 포함하는 차량 및 산업 등을 포함한다. 특정 네트워크들은 일반적으로 특정 범용 데이터 포트들 또는 주변 버스들(1549)(예컨대, 예를 들어, 컴퓨터 시스템(1500)의 USB 포트들)에 부착된 외부 네트워크 인터페이스 어댑터들을 요구한다; 다른 것들은 일반적으로 아래에 설명되는 바와 같은 시스템 버스로의 부착에 의해 컴퓨터 시스템(1500)의 코어에 통합된다(예를 들어, PC 컴퓨터 시스템으로의 이더넷 인터페이스는 또는 스마트폰 컴퓨터 시스템으로의 셀룰러 네트워크 인터페이스). 이들 네트워크들 중 임의의 것을 사용하여, 컴퓨터 시스템(1500)은 다른 엔티티들과 통신할 수 있다. 그러한 통신은 단방향성 수신 전용(예를 들어, 브로드캐스트 TV), 단방향성 송신 전용(예를 들어, CANbus

대 특정 CANbus 디바이스들), 또는 예를 들어 로컬 또는 광역 디지털 네트워크들을 사용하는 다른 컴퓨터 시스템들과의 양방향성일 수 있다. 위에서 설명된 바와 같은 네트워크들 및 네트워크 인터페이스들 각각에 대해 특정 프로토콜들 및 프로토콜 스택들이 사용될 수 있다.

[0165] 전술한 휴먼 인터페이스 디바이스들, 인간-액세스 가능한 저장 디바이스들, 및 네트워크 인터페이스들은 컴퓨터 시스템(1500)의 코어(1540)에 부착될 수 있다.

[0166] 코어(1540)는 하나 이상의 중앙 처리 유닛(CPU)(1541), 그래픽 처리 유닛(GPU)(1542), 필드 프로그래머블 게이트 영역(FPGA)(1543)의 형식으로 특수화된 프로그래머블 처리 유닛들, 특정 태스크들에 대한 하드웨어 가속기들(1544) 등을 포함할 수 있다. 이들 디바이스는, 판독 전용 메모리(ROM)(1545), 랜덤 액세스 메모리(1546), 내부 비-사용자 액세스 가능 하드 드라이브들, SSD들 등과 같은 내부 대용량 저장소(1547)와 함께, 시스템 버스(1548)를 통해 연결될 수 있다. 일부 컴퓨터 시스템들에서, 시스템 버스(1548)는 추가적인 CPU들, GPU들 등에 의한 확장을 가능하게 하기 위해 하나 이상의 물리적 플러그의 형식으로 액세스 가능할 수 있다. 주변 디바이스들은 코어의 시스템 버스(1548)에 직접, 또는 주변 버스(1549)를 통해 부착될 수 있다. 주변 버스를 위한 아키텍처들은 PCI, USB 등을 포함한다.

[0167] CPU들(1541), GPU들(1542), FPGA들(1543), 및 가속기들(1544)은, 조합하여, 전술한 컴퓨터 코드를 구성할 수 있는 특정 명령어들을 실행할 수 있다. 해당 컴퓨터 코드는 ROM(1545) 또는 RAM(1546)에 저장될 수 있다. 과도적인 데이터가 또한 RAM(1546)에 저장될 수 있는 반면, 영구 데이터는, 예를 들어, 내부 대용량 저장소(1547)에 저장될 수 있다. 메모리 디바이스들 중 임의의 것에 대한 고속 저장 및 검색은, 하나 이상의 CPU(1541), GPU(1542), 대용량 저장소(1547), ROM(1545), RAM(1546) 등과 밀접하게 연관될 수 있는, 캐시 메모리의 사용을 통해 가능하게 될 수 있다.

[0168] 컴퓨터 판독가능 매체는 다양한 컴퓨터 구현 동작들을 수행하기 위한 컴퓨터 코드를 그 위에 가질 수 있다. 매체 및 컴퓨터 코드는 본 개시내용의 목적을 위해 특별히 설계되고 구성된 것들일 수 있거나, 또는 그것들은 컴퓨터 소프트웨어 기술분야의 기술자들에게 잘 알려져 있고 이용가능한 종류의 것일 수 있다.

[0169] 제한이 아니라 예로서, 아키텍처를 갖는 컴퓨터 시스템(1500), 및 구체적으로 코어(1540)는 프로세서(들)(CPU들, GPU들, FPGA, 가속기들 등을 포함함)가 하나 이상의 유형의(tangible) 컴퓨터 판독가능 매체에 구현된 소프트웨어를 실행하는 결과로서 기능성을 제공할 수 있다. 그러한 컴퓨터 판독가능 매체는 위에 소개된 바와 같은 사용자-액세스 가능한 대용량 저장소뿐만 아니라, 코어-내부 대용량 저장소(1547) 또는 ROM(1545)과 같은 비일시적인 본질의 것인 코어(1540)의 특정 저장소와 연관된 매체일 수 있다. 본 개시내용의 다양한 실시예들을 구현하는 소프트웨어가 그러한 디바이스들에 저장되고 코어(1540)에 의해 실행될 수 있다. 컴퓨터 판독가능 매체는 특정 필요에 따라 하나 이상의 메모리 디바이스 또는 칩을 포함할 수 있다. 소프트웨어는 코어(1540) 및 구체적으로 그 내부의 프로세서들(CPU, GPU, FPGA 등을 포함함)로 하여금, RAM(1546)에 저장된 데이터 구조들을 정의하는 것 및 소프트웨어에 의해 정의된 프로세스들에 따라 그러한 데이터 구조들을 수정하는 것을 포함하여, 본 명세서에 설명된 특정 프로세스들 또는 특정 프로세스들의 특정 부분들을 실행하게 할 수 있다. 추가로 또는 대안으로서, 컴퓨터 시스템은, 본 명세서에 설명된 특정 프로세스들 또는 특정 프로세스들의 특정 부분들을 실행하기 위해 소프트웨어 대신에 또는 그와 함께 동작할 수 있는, 회로(예를 들어: 가속기(1544))에 하드와이어링되거나 다른 방식으로 구현된 로직의 결과로서 기능성을 제공할 수 있다. 소프트웨어에 대한 참조는, 적절한 경우, 로직을 포함할 수 있고, 그 반대도 가능하다. 컴퓨터 판독가능 매체에 대한 참조는, 적절한 경우, 실행을 위한 소프트웨어를 저장하는 회로(예컨대 집적 회로(IC)), 또는 실행을 위한 로직을 구현하는 회로, 또는 둘 다를 포함할 수 있다. 본 개시내용은 하드웨어 및 소프트웨어의 임의의 적합한 조합을 포함한다.

[0170] 부록 A: 두문자어들

[0171] JEM: 조인트 탐색 모델

[0172] VVC: 다목적 비디오 코딩

[0173] BMS: 벤치마크 세트

[0174] MV: 움직임 벡터

[0175] HEVC: 고효율 비디오 코딩

[0176] SEI: 보조 향상 정보

[0177] VUI: 비디오 사용성 정보

[0178] GOP들: 화상들의 그룹들

[0179] TU들: 변환 유닛들

[0180] PU들: 예측 유닛들

[0181] CTU들: 코딩 트리 유닛들

[0182] CTB들: 코딩 트리 블록들

[0183] PB들: 예측 블록들

[0184] HRD: 가상 참조 디코더

[0185] SNR: 신호 잡음비

[0186] CPU들: 중앙 처리 유닛들

[0187] GPU들: 그래픽 처리 유닛들

[0188] CRT: 음극선관

[0189] LCD: 액정 디스플레이

[0190] OLED: 유기 발광 다이오드

[0191] CD: 콤팩트 디스크

[0192] DVD: 디지털 비디오 디스크

[0193] ROM: 판독 전용 메모리

[0194] RAM: 랜덤 액세스 메모리

[0195] ASIC: 애플리케이션-특정 집적 회로

[0196] PLD: 프로그래머블 로직 디바이스

[0197] LAN: 로컬 영역 네트워크

[0198] GSM: 모바일 통신을 위한 글로벌 시스템

[0199] LTE: 롱 텀 에볼루션

[0200] CANBus: 컨트롤러 영역 네트워크 버스

[0201] USB: 범용 직렬 버스

[0202] PCI: 주변 컴포넌트 인터커넥트

[0203] FPGA: 필드 프로그래머블 게이트 영역들

[0204] SSD: 솔리드 스테이트 드라이브

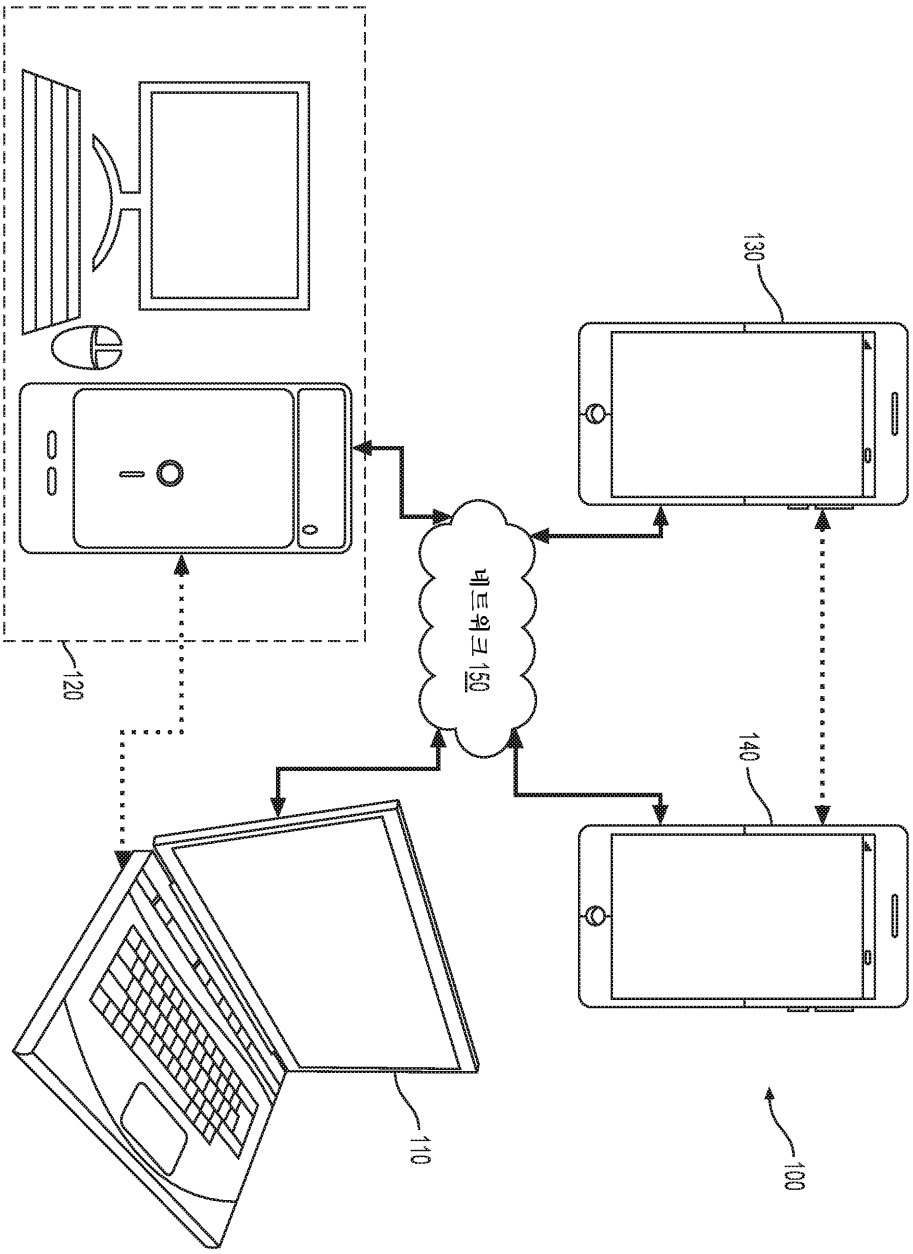
[0205] IC: 집적 회로

[0206] CU: 코딩 유닛

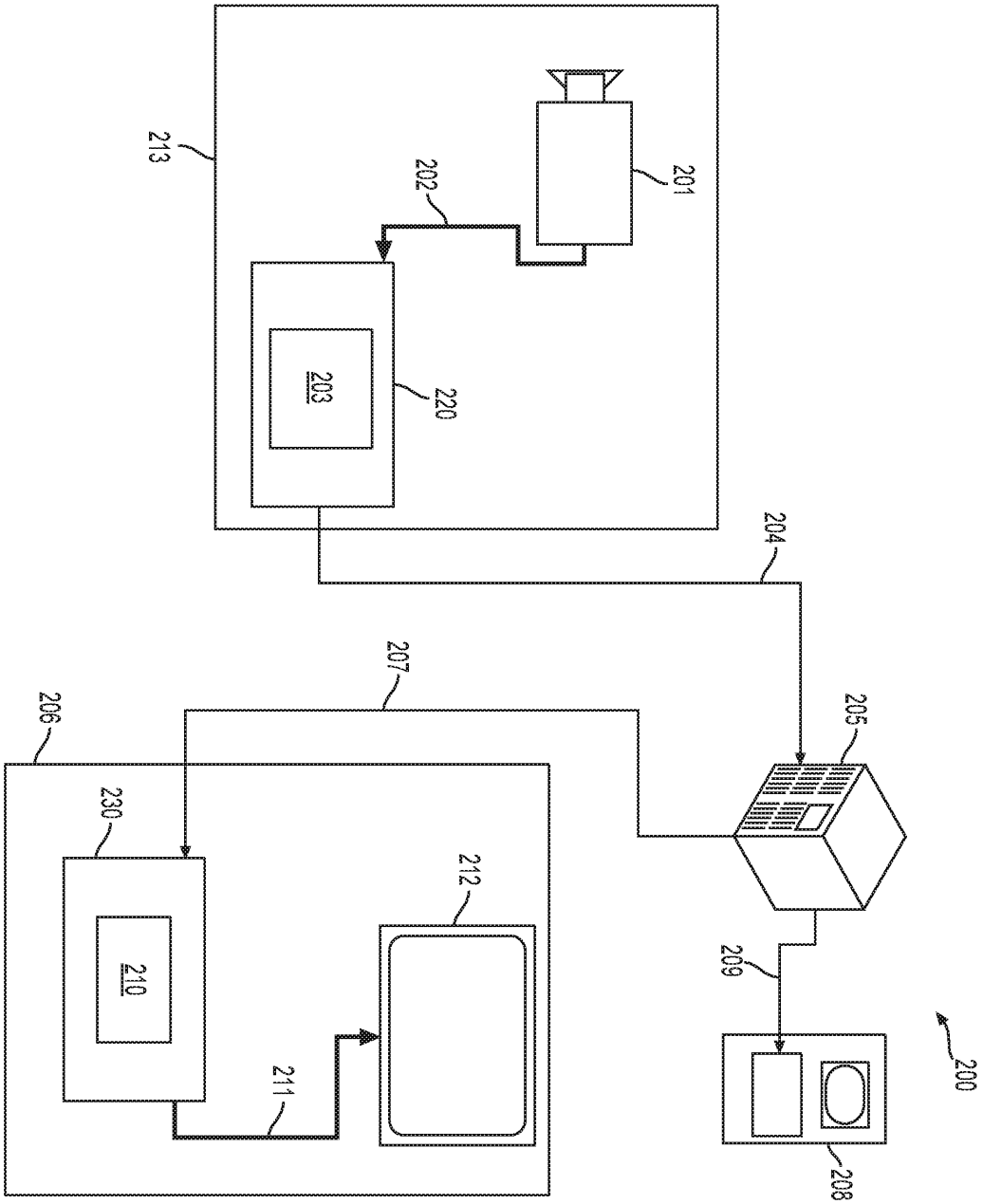
[0207] 본 개시내용이 여러 예시적인 실시예들을 설명하였지만, 본 개시내용의 범위 내에 속하는 변경들, 치환들, 및 다양한 대체 균등물들이 존재한다. 따라서, 본 기술분야의 통상의 기술자들은, 비록 본 명세서에 명시적으로 도시되거나 설명되지는 않았지만, 본 개시내용의 원리들을 구현하고 따라서 그것의 진의 및 범위 내에 있는, 다수의 시스템들 및 방법들을 고안할 수 있을 것이라는 점이 인정될 것이다.

도면

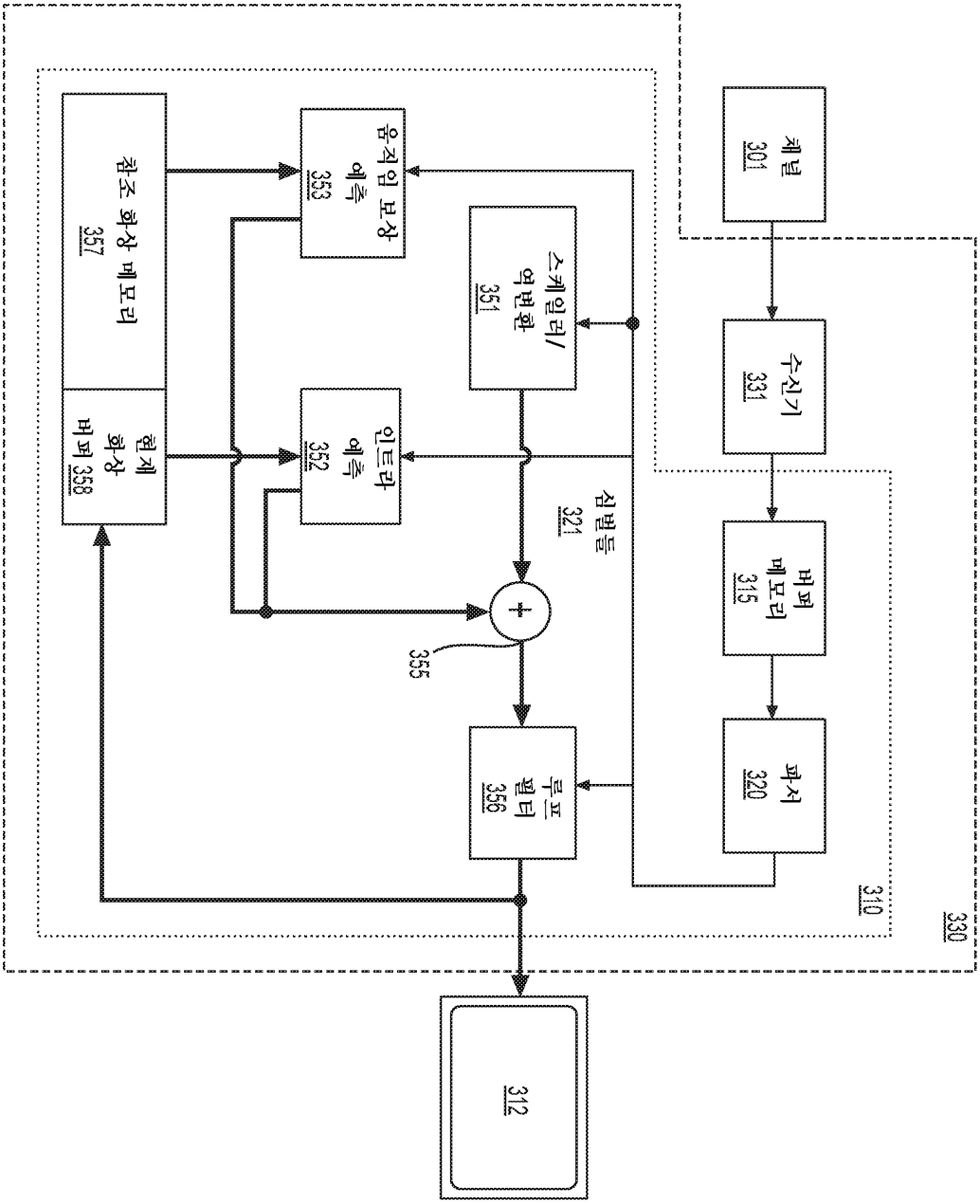
도면1



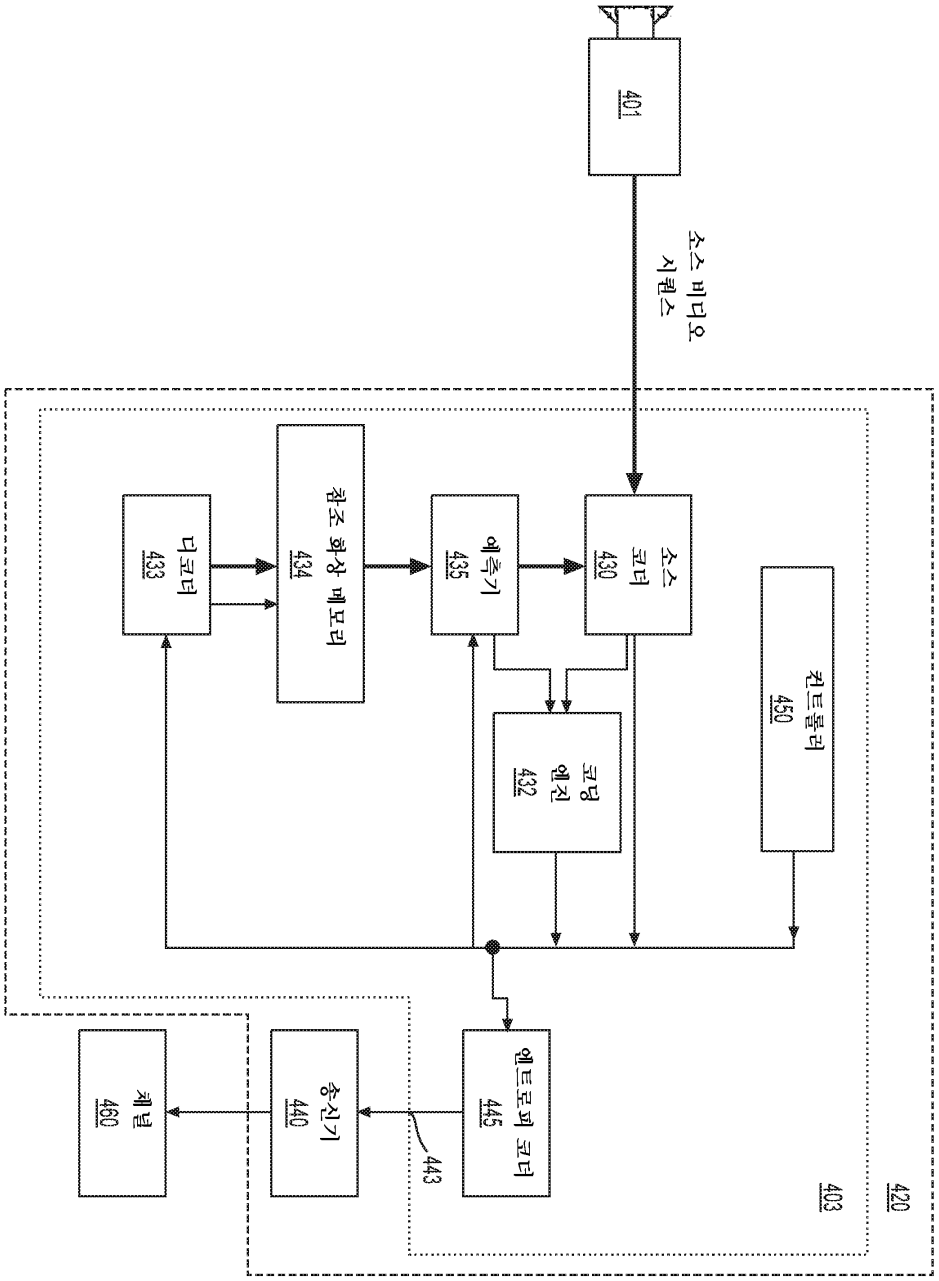
도면2



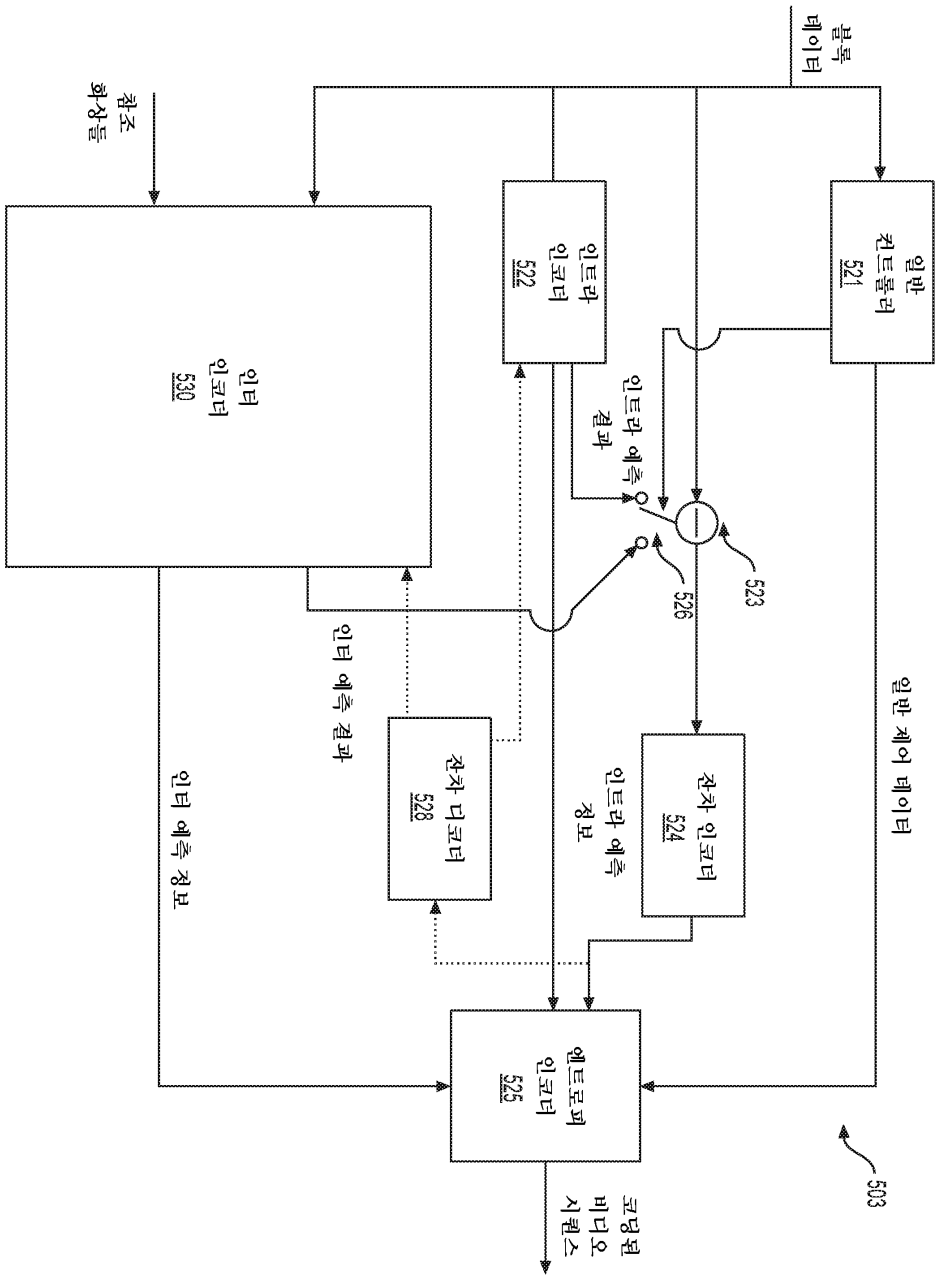
도면3


330
310
채널
301
수신기
331
버퍼
메모리
315
파서
320
심벌들
321
스케일러/
역변환
351
움직임 보상
예측
353
인트라
예측
352
+
355
루프
필터
356
참조 화상 메모리
357
현재
화상
버퍼 358
312

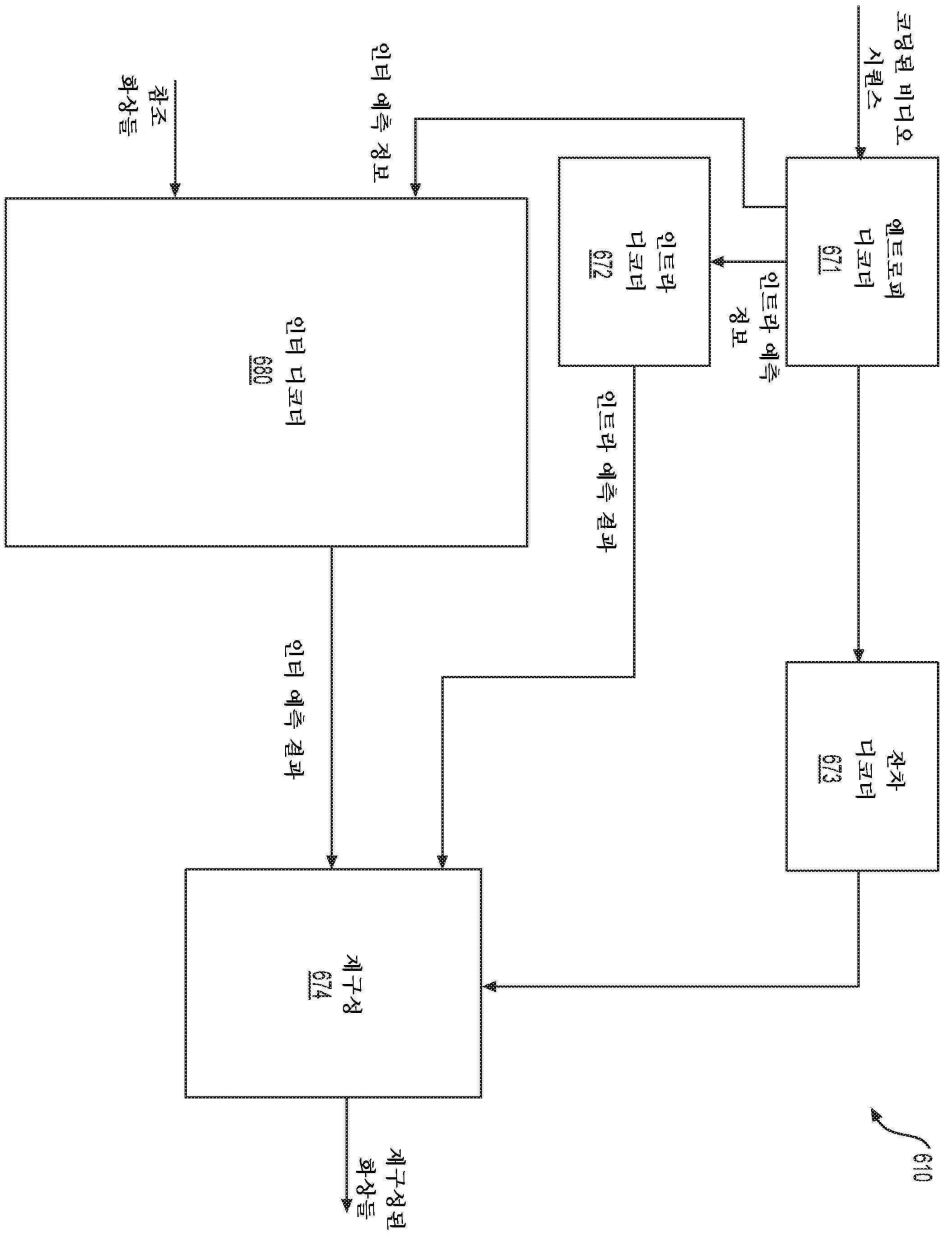
도면4


401
소스 비디오 시퀀스
420
403
소스 코더 430
컨트롤러 450
예측기 435
코딩 엔진 432
참조 화상 메모리 434
디코더 433
443
엔트로피 코더 445
송신기 440
채널 460

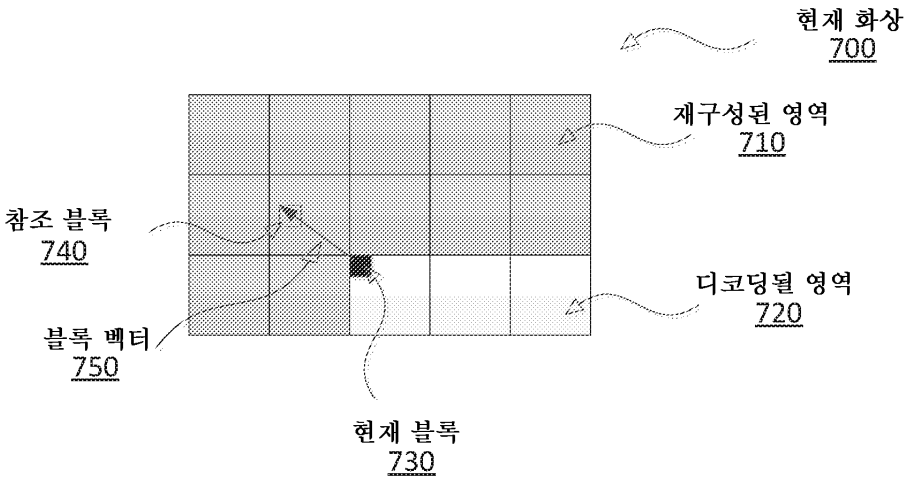
도면5


블록
데이터
일반
컨트롤러
521
일반 제어 데이터
인트라
인코더
522
인트라 예측
결과
523
526
전자 인코더
524
인트라 예측
정보
참조
화상들
인터
인코더
530
인터 예측 결과
전자 디코더
528
인터 예측 정보
엔트로피
인코더
525
503
코딩된
비디오
시퀀스

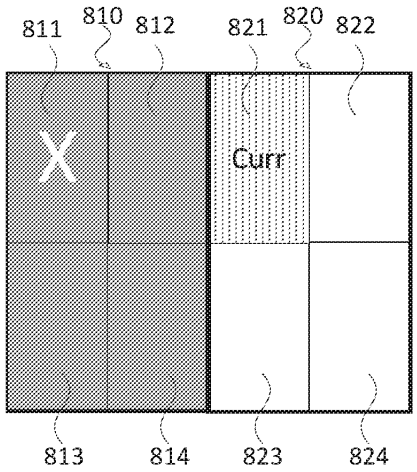
도면6


610
코딩된 비디오 시퀀스
엔트로피 디코더 671
인트라 예측 정보
인트라 디코더 672
인트라 예측 결과
잔차 디코더 673
인터 예측 정보
참조 화상들
인터 디코더 680
인터 예측 결과
재구성 674
재구성된 화상들

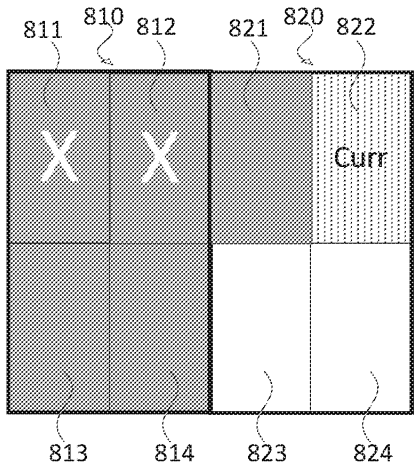
도면7



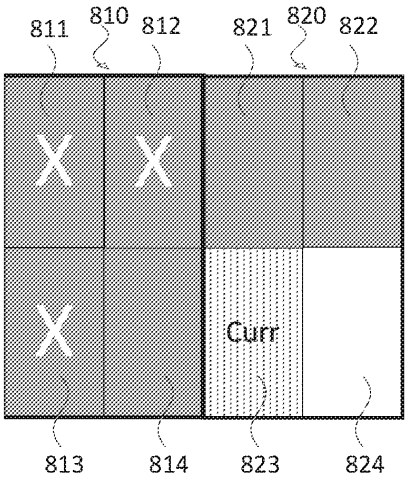
도면8a



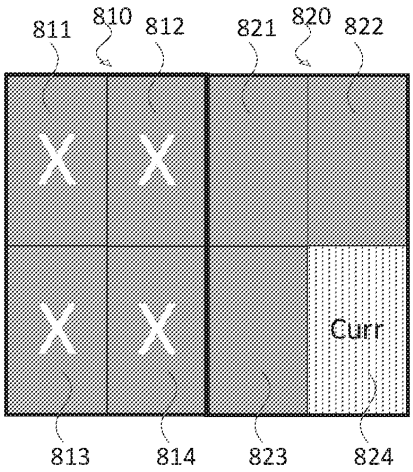
도면8b



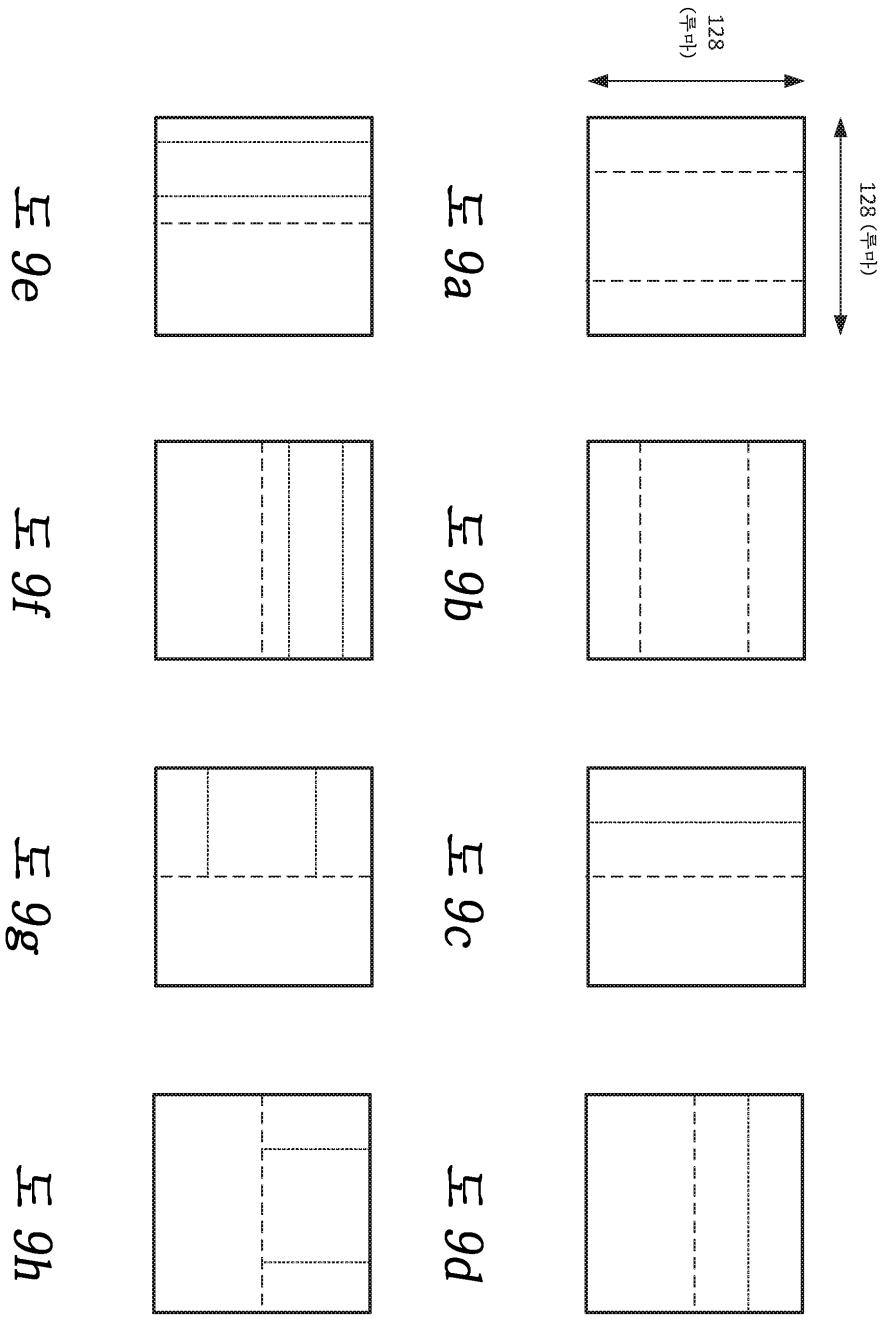
도면8c


811
810
812
821
820
822
Curr
813
814
823
824

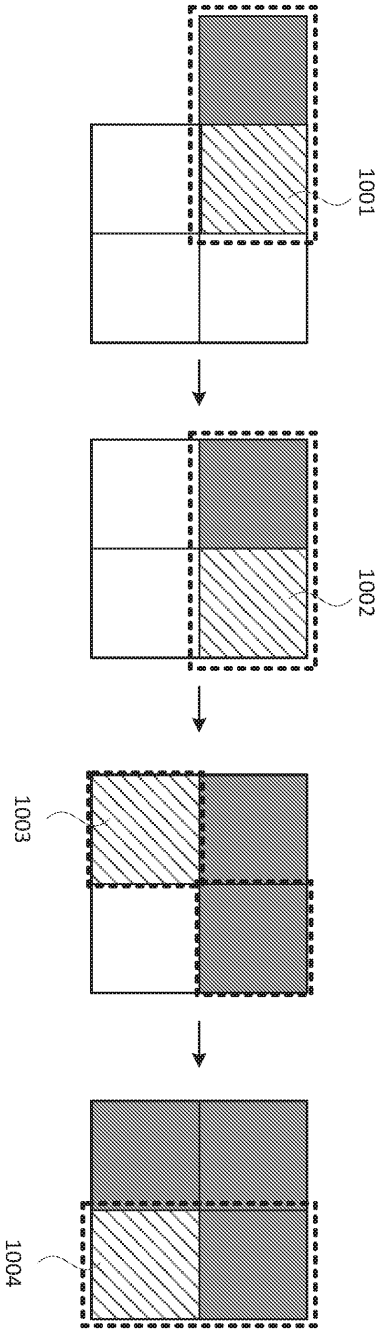
도면8d


811
810
812
821
820
822
Curr
813
814
823
824

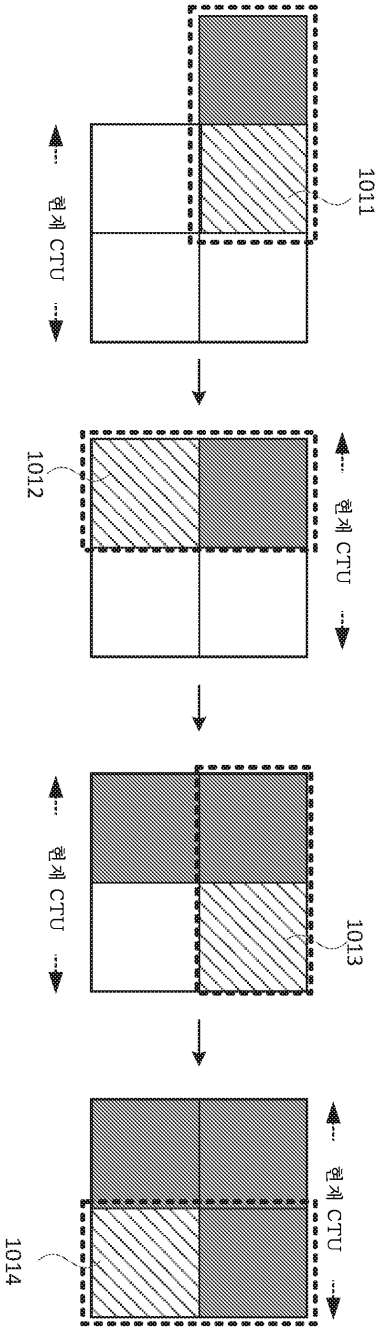
도면9



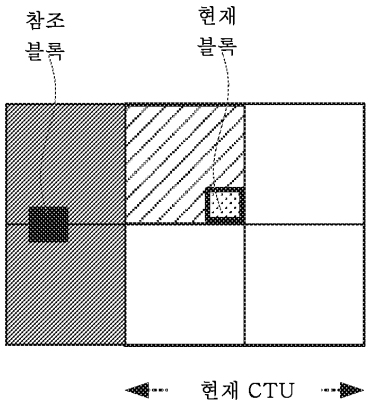
도면10a



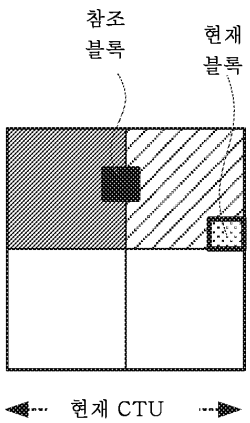
도면10b



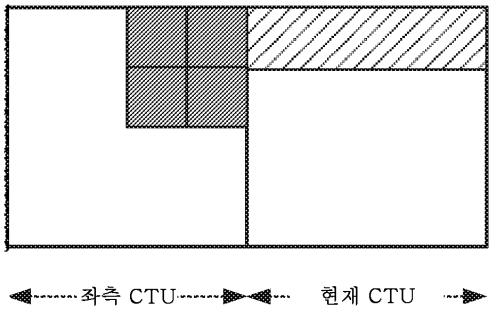
도면11a



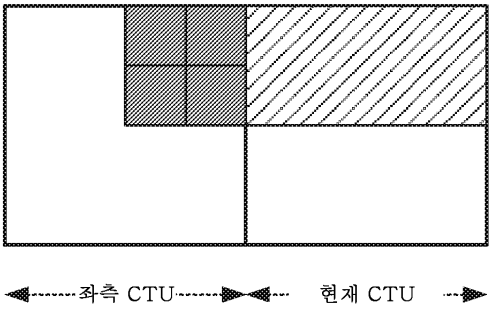
도면11b



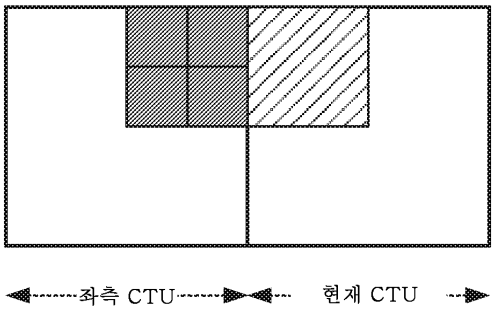
도면12a



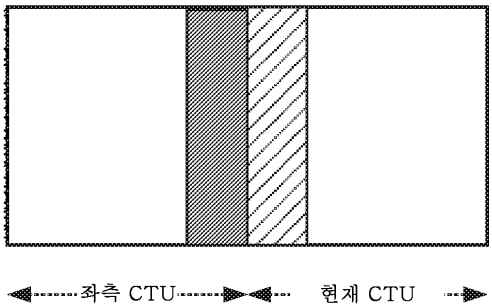
도면12b



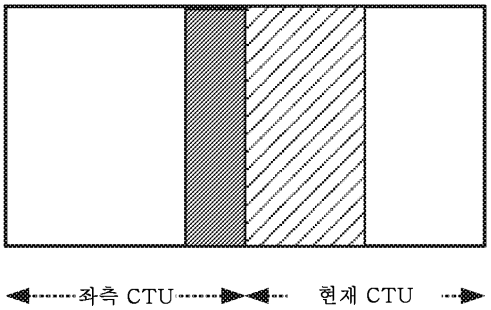
도면12c



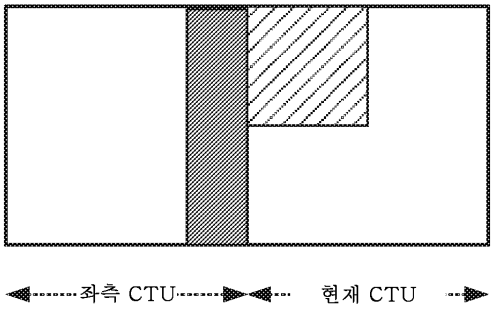
도면12d



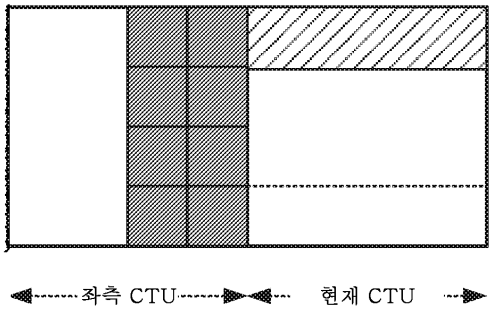
도면12e



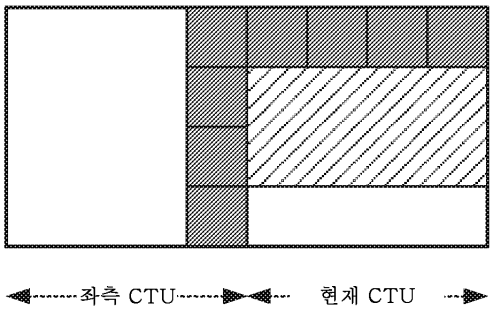
도면12f



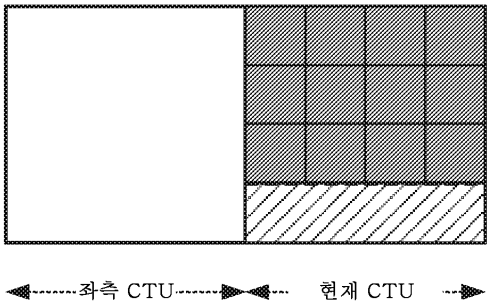
도면13a



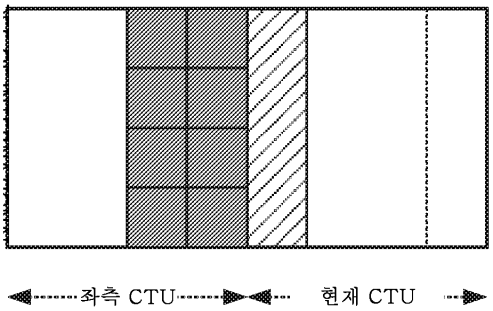
도면13b



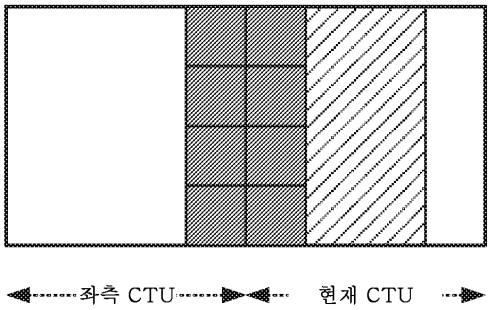
도면13c



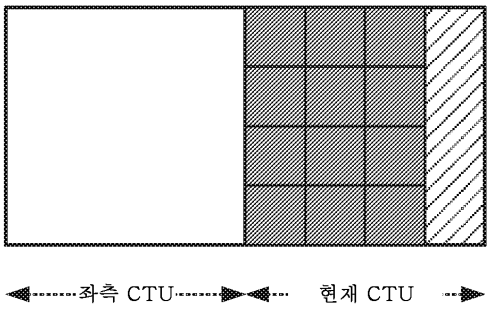
도면13d



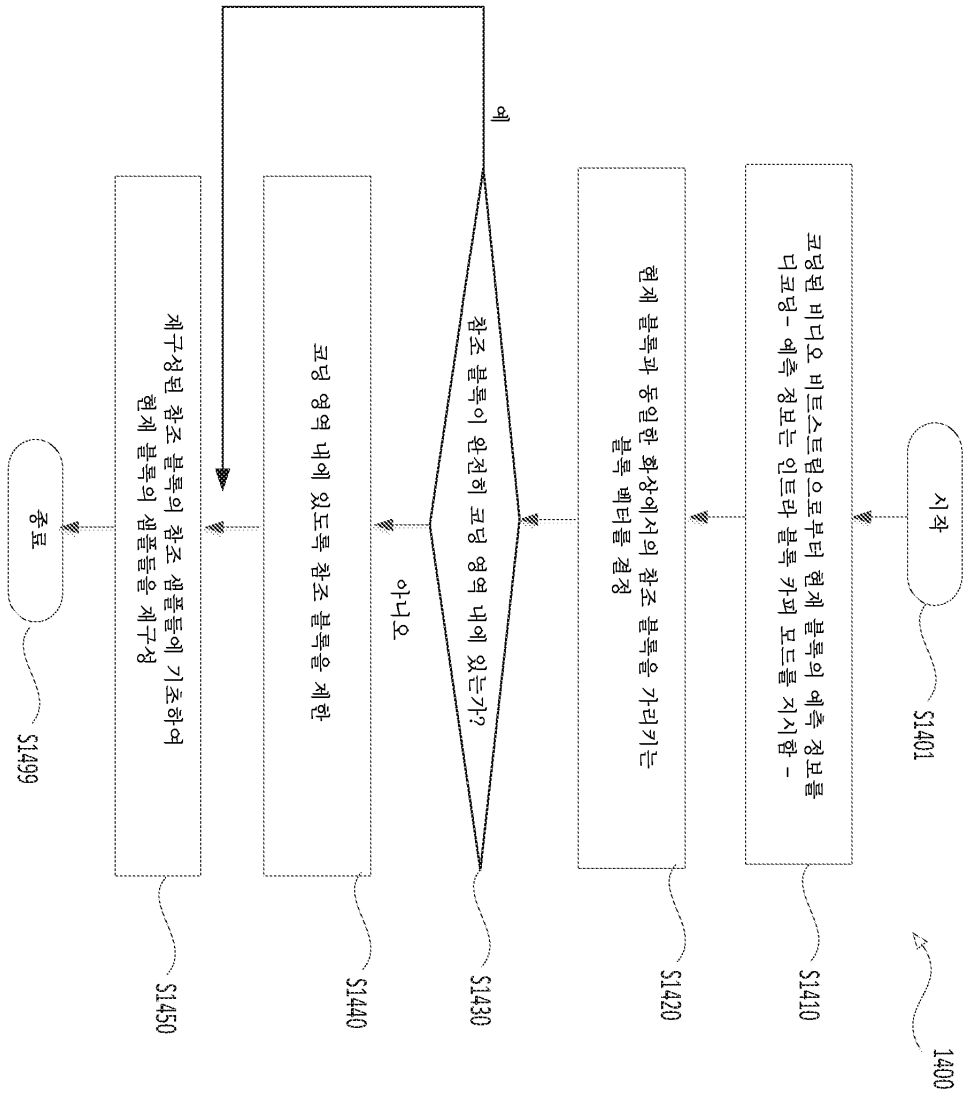
도면13e



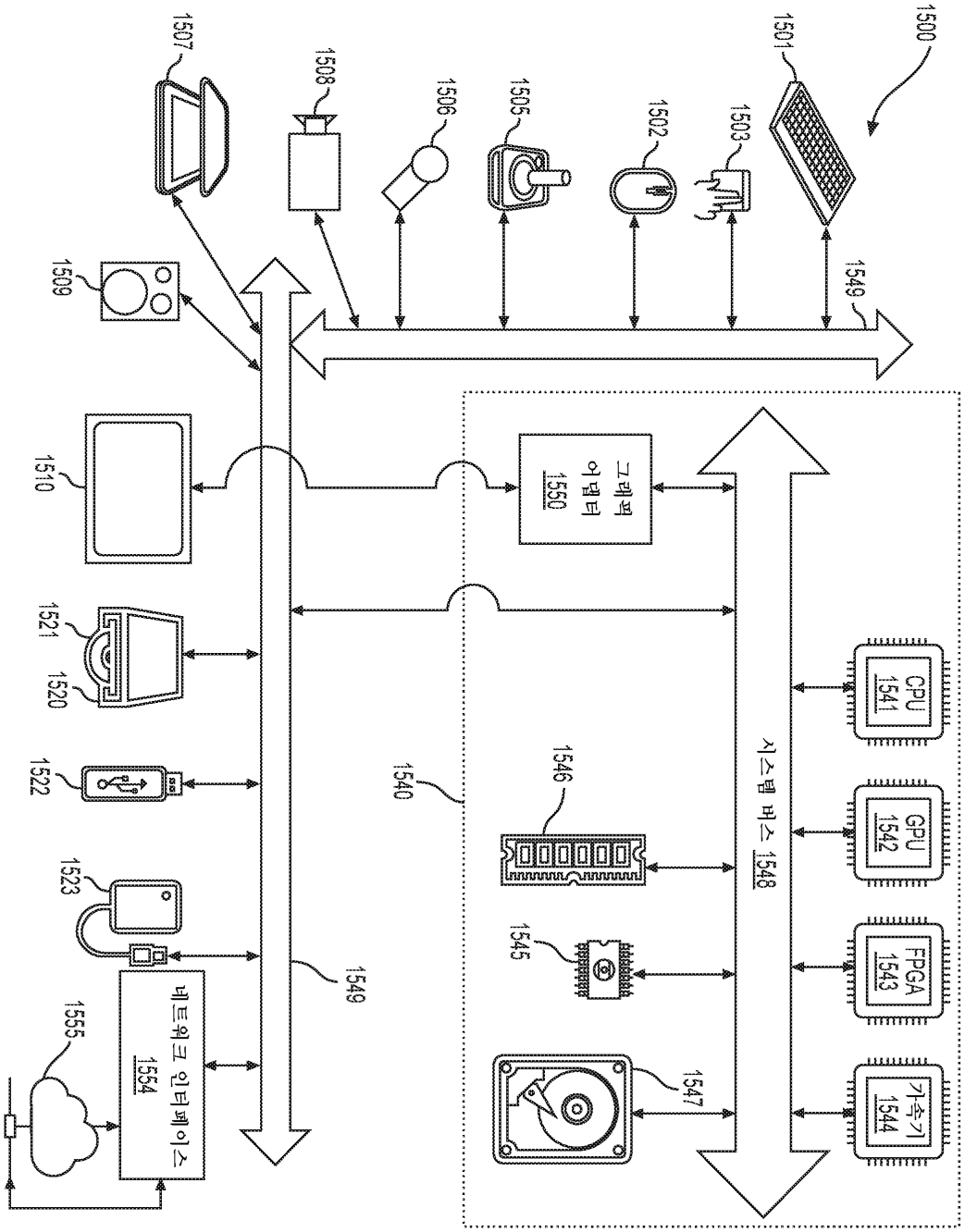
도면13f



도면14


1400
시작
S1401
코딩된 비디오 비트스트림으로부터 현재 블록의 예측 정보를 디코딩- 예측 정보는 인트라 블록 카피 모드를 지시함 -
S1410
현재 블록과 동일한 화상에서의 참조 블록을 가리키는 블록 벡터를 결정
S1420
참조 블록이 완전히 코딩 영역 내에 있는가?
S1430
예
아니오
코딩 영역 내에 있도록 참조 블록을 제한
S1440
재구성된 참조 블록의 참조 샘플들에 기초하여 현재 블록의 샘플들을 재구성
S1450
종료
S1499

도면15


1500
1501
1503
1502
1505
1506
1508
1507
1549
1509
1510
그래픽 어댑터 1550
1521
1520
1522
1523
1540
1546
1545
1547
시스템 버스 1548
CPU 1541
GPU 1542
FPGA 1543
가속기 1544
1549
네트워크 인터페이스 1554
1555