



(19) 대한민국특허청(KR)

(12) 등록특허공보(B1)

(45) 공고일자 2025년06월26일

(11) 등록번호 10-2825177

(24) 등록일자 2025년06월20일

(51) 국제특허분류(Int. Cl.)

H04N 19/70 (2014.01) G06F 18/00 (2023.01)
G06V 10/24 (2022.01) H04N 19/117 (2014.01)
H04N 19/119 (2014.01) H04N 19/169 (2014.01)
H04N 19/174 (2014.01) H04N 19/52 (2014.01)
H04N 19/82 (2014.01)

(52) CPC특허분류

H04N 19/70 (2015.01)
G06V 10/62 (2023.08)

(21) 출원번호 10-2021-7032477

(22) 출원일자(국제) 2020년03월11일

심사청구일자 2021년10월08일

(85) 번역문제출일자 2021년10월08일

(65) 공개번호 10-2021-0135309

(43) 공개일자 2021년11월12일

(86) 국제출원번호 PCT/US2020/022080

(87) 국제공개번호 WO 2020/185883

국제공개일자 2020년09월17일

(30) 우선권주장

62/816,751 2019년03월11일 미국(US)

62/826,659 2019년03월29일 미국(US)

(56) 선행기술조사문헌

Byeongdoo Choi, et al., AHG8/AHG12: On
sub-picture partitioning support with layers,
JVET-00334, 2019.06.27.

(뒷면에 계속)

전체 청구항 수 : 총 10 항

심사관 : 이남숙

(54) 발명의 명칭 인코더, 디코더, 및 대응하는 방법들

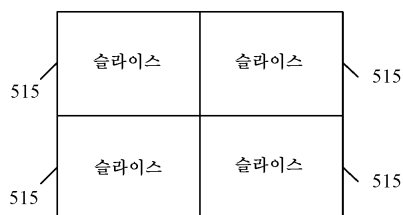
(57) 요약

비디오 코딩 메커니즘이 개시된다. 메커니즘은 슬라이스를 포함하는 서브-픽처를 포함하는 비트스트림을 수신하는 것을 포함한다. 슬라이스 어드레스가 획득된다. 슬라이스 어드레스는 슬라이스가 직사각형 슬라이스일 때 슬라이스의 서브-픽처 레벨 슬라이스 인덱스를 포함한다. 서브-픽처의 슬라이스는 슬라이스 어드레스에 기초하여 디코딩된다. 슬라이스는 디코딩된 비디오 시퀀스의 일부로서 디스플레이를 위해 전달된다.

대표도 - 도5b

510

서브-픽처



(52) CPC특허분류

G06V 20/49 (2022.01)
H04N 19/117 (2015.01)
H04N 19/119 (2015.01)
H04N 19/169 (2015.01)
H04N 19/174 (2015.01)
H04N 19/52 (2015.01)
H04N 19/82 (2015.01)

(56) 선행기술조사문헌

Miska M. Hannuksela, et al., AHG12: On grouping of tiles, JVET-M0261, 2019.01.02.*
 Benjamin Bross, et al., Versatile Video Coding (Draft 6), JVET-02001-vE, 2019.07.31.
 Minhua Zhou, AHG4: Enable parallel decoding with tiles, JCTVC-I0118, 2012.04.27.
 Lulin Chen, et al., AHG17/AHG12: On associating slices with a subpicture, JVET-P0144-v1, 2019.09.24.*

*는 심사관에 의하여 인용된 문헌

명세서

청구범위

청구항 1

디코더에서 구현되는 방법으로서,

타일 그룹을 포함하는 서브-픽처의 코딩된 데이터를 포함하는 비트스트림을 수신하는 단계;

타일 그룹 어드레스를 획득하는 단계 - 상기 타일 그룹 어드레스는 상기 타일 그룹이 직사각형 타일 그룹일 때 상기 타일 그룹의 서브-픽처 레벨 타일 그룹 인덱스를 포함함 - ; 및

상기 타일 그룹 어드레스에 기초하여 상기 서브-픽처의 타일 그룹을 디코딩하는 단계를 포함하는 방법.

청구항 2

제1항에 있어서,

rect_tile group_flag가 1일 때 상기 타일 그룹은 직사각형 타일 그룹인 방법.

청구항 3

인코더에서 구현되는 방법으로서,

서브-픽처를 타일 그룹으로 파티셔닝하는 단계;

타일 그룹 어드레스를 결정하는 단계 - 상기 타일 그룹 어드레스는 상기 타일 그룹이 직사각형 타일 그룹일 때 상기 타일 그룹의 서브-픽처 레벨 타일 그룹 인덱스를 포함함 - ;

상기 타일 그룹 어드레스를 비트스트림으로 인코딩하는 단계; 및

상기 서브-픽처의 타일 그룹을 상기 비트스트림으로 인코딩하는 단계를 포함하는 방법.

청구항 4

제3항에 있어서,

rect_tile group_flag가 1일 때 상기 타일 그룹은 직사각형 타일 그룹인 방법.

청구항 5

비디오 코딩 디바이스로서,

프로세서, 상기 프로세서에 결합된 수신기, 상기 프로세서에 결합된 메모리, 및 상기 프로세서에 결합된 송신기를 포함하고, 상기 프로세서, 수신기, 메모리, 및 송신기는 제1항 또는 제3항 중 어느 한 항의 방법을 수행하도록 구성되는 비디오 코딩 디바이스.

청구항 6

비디오 코딩 디바이스에 의한 사용을 위한 컴퓨터 프로그램 제품을 포함하는 비일시적 컴퓨터 판독가능 매체로서,

상기 컴퓨터 프로그램 제품은, 프로세서에 의해 실행될 때 상기 비디오 코딩 디바이스로 하여금 제1항 또는 제3항 중 어느 한 항의 방법을 수행하게 야기하도록 하는, 상기 비일시적 컴퓨터 판독가능 매체 상에 저장된 컴퓨터 실행가능 명령어들을 포함하는 비일시적 컴퓨터 판독가능 매체.

청구항 7

디코더로서,

타일 그룹을 포함하는 서브-픽처의 코딩된 데이터를 포함하는 비트스트림을 수신하도록 구성된 수신 유닛;

타일 그룹 어드레스를 획득하도록 구성된 획득 유닛 - 상기 타일 그룹 어드레스는 상기 타일 그룹이 직사각형 타일 그룹일 때 상기 타일 그룹의 서브-픽처 레벨 타일 그룹 인덱스를 포함함 - ; 및

상기 타일 그룹 어드레스에 기초하여 상기 서브-픽처의 타일 그룹을 디코딩하도록 구성된 디코딩 유닛을 포함하는 디코더.

청구항 8

인코더로서,

서브-픽처를 타일 그룹으로 파티셔닝하도록 구성된 파티셔닝 유닛;

타일 그룹 어드레스를 결정하도록 구성된 결정 유닛 - 상기 타일 그룹 어드레스는 상기 타일 그룹이 직사각형 타일 그룹일 때 상기 타일 그룹의 서브-픽처 레벨 타일 그룹 인덱스를 포함함 - ;

상기 타일 그룹 어드레스를 비트스트림으로 인코딩하고;

상기 서브-픽처의 타일 그룹을 상기 비트스트림으로 인코딩하도록 구성된 인코딩 유닛; 및

디코더를 향한 통신을 위해 상기 비트스트림을 저장하도록 구성된 저장 수단을 포함하는 인코더.

청구항 9

비일시적 컴퓨터 판독가능 매체 상에 저장된 컴퓨터 프로그램으로서,

컴퓨터 또는 프로세서 상에서 실행될 때 제1항 또는 제3항 중 어느 한 항에 따른 방법을 수행하기 위한 프로그램 코드를 포함하는, 비일시적 컴퓨터 판독가능 매체 상에 저장된 컴퓨터 프로그램.

청구항 10

코딩된 비디오 비트스트림을 포함하는 비일시적 저장 매체로서,

상기 코딩된 비디오 비트스트림은 타일 그룹을 포함하는 서브-픽처의 코딩된 데이터, 및 타일 그룹 어드레스를 포함하며, 상기 타일 그룹 어드레스는 상기 타일 그룹이 직사각형 타일 그룹일 때 상기 타일 그룹의 서브-픽처 레벨 타일 그룹 인덱스를 포함하는 비일시적 저장 매체.

청구항 11

삭제

청구항 12

삭제

청구항 13

삭제

청구항 14

삭제

청구항 15

삭제

청구항 16

삭제

청구항 17

삭제

청구항 18

삭제

청구항 19

삭제

청구항 20

삭제

청구항 21

삭제

청구항 22

삭제

발명의 설명

기술 분야

[0001] 삭제

[0002] 본 개시내용은 일반적으로 비디오 코딩에 관한 것이고, 구체적으로 비디오 코딩에서 픽처들의 서브-픽처들을 코딩하는 것에 관한 것이다.

배경 기술

[0003] 심지어 비교적 짧은 비디오를 묘사하기 위해 필요한 비디오 데이터의 양은 상당할 수 있으며, 이는 데이터가 대역폭 용량이 제한된 통신 네트워크에 걸쳐 스트리밍되거나 또는 달리 통신되어야 할 때 어려움들을 초래할 수 있다. 따라서, 비디오 데이터는 현대의 통신 네트워크들에 걸쳐 통신되기 전에 일반적으로 압축된다. 메모리 자원들이 제한될 수 있기 때문에 비디오가 저장 디바이스 상에 저장될 때 비디오의 크기가 또한 쟁점일 수 있다. 비디오 압축 디바이스들은 소스에서 소프트웨어 및/또는 하드웨어를 종종 사용하여 송신 또는 저장 전에 비디오 데이터를 코딩하고, 그렇게 함으로써 디지털 비디오 이미지를 표현하기 위해 필요한 데이터의 수량을 감소시킨다. 그 다음에 압축된 데이터는 비디오 데이터를 디코딩하는 비디오 압축해제 디바이스에 의해 목적지에서 수신된다. 제한된 네트워크 자원들과 더 높은 비디오 품질에 대한 수요가 계속 증가함에 따라, 이미지 품질을 거의 또는 전혀 희생하지 않고 압축 비율을 개선하는 개선된 압축 및 압축해제 기법들이 바람직하다.

발명의 내용

과제의 해결 수단

[0004] 일 실시예에서, 본 개시내용은 디코더에서 구현되는 방법을 포함하며, 이 방법은: 디코더의 수신기에 의해, 슬라이스를 포함하는 서브-픽처를 포함하는 비트스트림을 수신하는 단계; 프로세서에 의해, 슬라이스가 직사각형 슬라이스일 때 상기 슬라이스의 서브-픽처 레벨 슬라이스 인덱스를 포함하는 슬라이스 어드레스를 획득하는 단계; 및 프로세서에 의해, 슬라이스 어드레스에 기초하여 서브-픽처의 슬라이스를 디코딩하는 단계를 포함한다. 픽처는 인코딩을 위해 복수의 슬라이스로 파티셔닝될 수 있다. 일부 비디오 코딩 시스템들에서, 슬라이스들은 픽처에 대한 그들의 위치에 기초하여 어드레스된다. 또 다른 비디오 코딩 시스템들은 서브-픽처들의 개념을 이용한다. 서브-픽처는 코딩 관점에서 다른 서브-픽처들과 상이하게 취급될 수 있다. 예를 들어, 서브-픽처는 다른 서브-픽처들과 독립적으로 추출되고 디스플레이될 수 있다. 이러한 경우에, 픽처 위치에 기초하여 생성되는 슬라이스 어드레스들은, 상당한 수의 예상된 슬라이스 어드레스들이 생략됨에 따라 적절히 동작하는 것을 중단할 수 있다. 일부 비디오 코딩 시스템들은 서브-픽처 추출을 지원하기 위해 슬라이스 어드레스들을 변경하기 위한 요청시에 슬라이스 헤더들을 동적으로 재기입함으로써 이 문제를 다룬다. 이러한 프로세스는 자원 집약적

일 수 있는데, 그 이유는 이 프로세스가 사용자가 서브-픽처를 보도록 요청할 때마다 일어날 수 있기 때문이다. 본 개시내용은 슬라이스를 포함하는 서브-픽처와 관련하여 어드레싱되는 슬라이스들을 포함한다. 예를 들어, 슬라이스 헤더는 서브-픽처 식별자(ID) 및 서브-픽처에 포함된 각각의 슬라이스의 어드레스를 포함할 수 있다. 또한, SPS(sequence parameter set)는 서브-픽처 ID에 의해 참조될 수 있는 서브-픽처의 치수들을 포함할 수 있다. 따라서, 슬라이스 헤더는 서브-픽처의 개별 추출이 요청될 때 재기입될 필요가 없다. 슬라이스 헤더 및 SPS는 디스플레이를 위해 서브-픽처에 슬라이스들을 배치하는 것을 지원하기에 충분한 정보를 포함한다. 이와 같이, 본 예는 코딩 효율을 증가시키고 및/또는 슬라이스 헤더의 중복 재기입을 피하므로, 인코더 및/또는 디코더에서 프로세서, 메모리, 및/또는 네트워크 자원의 이용을 감소시킨다.

[0005] 선택적으로, 이전 양태들 중 어느 것에서, 상기 양태의 다른 구현은, 프로세서에 의해, 슬라이스를 포함하는 상기 서브-픽처의 서브-픽처 식별자(ID)를 획득하는 단계; 프로세서에 의해, 슬라이스 어드레스 및 서브-픽처 ID에 기초하여 서브-픽처 내에 슬라이스를 배치하는 단계를 추가로 포함하는 것을 제공한다.

[0006] 선택적으로, 이전 양태들 중 어느 것에서, 상기 양태의 다른 구현은, 비트스트림이 슬라이스와 관련된 슬라이스 헤더를 추가로 포함하고, 슬라이스 어드레스 및 서브-픽처 ID가 슬라이스 헤더로부터 획득되는 것을 제공한다.

[0007] 선택적으로, 이전 양태들 중 어느 것에서, 상기 양태의 다른 구현은, rect_slice_flag가 1일 때 슬라이스가 직사각형 슬라이스인 것을 제공한다.

[0008] 선택적으로, 이전 양태들 중 어느 것에서, 상기 양태의 다른 구현은, 슬라이스 어드레스가 slice_address 변수로부터 획득되고, 서브-픽처 ID가 slice_subpic_id 변수로부터 획득되는 것을 제공한다.

[0009] 선택적으로, 이전 양태들 중 어느 것에서, 상기 양태의 다른 구현은, 비트스트림이 시퀀스 파라미터 세트(SPS)를 추가로 포함하고, SPS는 서브-픽처의 설명을 포함하고, 서브-픽처 ID는 서브-픽처의 설명을 참조하는 것을 제공한다.

[0010] 선택적으로, 이전 양태들 중 어느 것에서, 상기 양태의 다른 구현은, 서브-픽처의 설명이 코딩 트리 블록(CTB) 단위의 서브-픽처의 폭, CTB 단위의 서브-픽처의 높이, 및 서브-픽처의 좌측 상단 코딩 트리 유닛(CTU)의 수직 위치 및 서브-픽처의 좌측 상단 CTU의 수평 위치로서 특정된 서브-픽처의 오프셋을 포함하는 것을 제공한다.

[0011] 일 실시예에서, 본 개시내용은 인코더에서 구현되는 방법을 포함하며, 이 방법은: 인코더의 프로세서에 의해, 서브-픽처를 슬라이스로 파티셔닝하는 단계; 프로세서에 의해, 슬라이스가 직사각형 슬라이스일 때 슬라이스의 서브-픽처 레벨 슬라이스 인덱스를 포함하는 슬라이스 어드레스를 결정하는 단계; 프로세서에 의해, 슬라이스 어드레스를 비트스트림으로 인코딩하는 단계; 프로세서에 의해, 서브-픽처의 슬라이스를 비트스트림으로 인코딩하는 단계; 및 프로세서에 결합된 메모리에 의해, 디코더를 향한 통신을 위해 비트스트림을 저장하는 단계를 포함한다. 픽처는 인코딩을 위해 복수의 슬라이스로 파티셔닝될 수 있다. 일부 비디오 코딩 시스템들에서, 슬라이스들은 픽처에 대한 그들의 위치에 기초하여 어드레싱된다. 또 다른 비디오 코딩 시스템들은 서브-픽처들의 개념을 이용한다. 서브-픽처는 코딩 관점에서 다른 서브-픽처들과 상이하게 취급될 수 있다. 예를 들어, 서브-픽처는 다른 서브-픽처들과 독립적으로 추출되고 디스플레이될 수 있다. 이러한 경우에, 픽처 위치에 기초하여 생성되는 슬라이스 어드레스들은, 상당한 수의 예상된 슬라이스 어드레스들이 생략됨에 따라 적절히 동작하는 것을 중단할 수 있다. 일부 비디오 코딩 시스템들은 서브-픽처 추출을 지원하기 위해 슬라이스 어드레스들을 변경하기 위한 요청시에 슬라이스 헤더들을 동적으로 재기입함으로써 이 문제를 다룬다. 이러한 프로세스는 자원 집약적일 수 있는데, 그 이유는 이 프로세스가 사용자가 서브-픽처를 보도록 요청할 때마다 일어날 수 있기 때문이다. 본 개시내용은 슬라이스를 포함하는 서브-픽처와 관련하여 어드레싱되는 슬라이스들을 포함한다. 예를 들어, 슬라이스 헤더는 서브-픽처 식별자(ID) 및 서브-픽처에 포함된 각각의 슬라이스의 어드레스를 포함할 수 있다. 또한, SPS(sequence parameter set)는 서브-픽처 ID에 의해 참조될 수 있는 서브-픽처의 치수들을 포함할 수 있다. 따라서, 슬라이스 헤더는 서브-픽처의 개별 추출이 요청될 때 재기입될 필요가 없다. 슬라이스 헤더 및 SPS는 디스플레이를 위해 서브-픽처에 슬라이스들을 배치하는 것을 지원하기에 충분한 정보를 포함한다. 이와 같이, 본 예는 코딩 효율을 증가시키고 및/또는 슬라이스 헤더의 중복 재기입을 피하므로, 인코더 및/또는 디코더에서 프로세서, 메모리, 및/또는 네트워크 자원의 이용을 감소시킨다.

[0012] 선택적으로, 이전 양태들 중 어느 것에서, 상기 양태의 다른 구현은, 프로세서에 의해, 슬라이스를 포함하는 서브-픽처의 서브-픽처 ID를 결정하는 단계; 및 프로세서에 의해, 서브-픽처 ID를 비트스트림으로 인코딩하는 단계를 추가로 포함하는 것을 제공한다.

[0013] 선택적으로, 이전 양태들 중 어느 것에서, 상기 양태의 다른 구현은, 프로세서에 의해, 슬라이스와 연관된 슬라이스

이스 헤더를 비트스트림으로 인코딩하는 단계를 추가로 포함하고, 슬라이스 어드레스 및 서브-픽처 ID는 슬라이스 헤더로 인코딩되는 것을 제공한다.

[0014] 선택적으로, 이전 양태들 중 어느 것에서, 상기 양태의 다른 구현은, rect_slice_flag가 1일 때 슬라이스가 직사각형 슬라이스인 것을 제공한다.

[0015] 선택적으로, 이전 양태들 중 어느 것에서, 상기 양태의 다른 구현은, 슬라이스 어드레스가 slice_address 변수로 인코딩되고, 서브-픽처 ID가 slice_subpic_id 변수로 인코딩되는 것을 제공한다.

[0016] 선택적으로, 이전 양태들 중 어느 것에서, 상기 양태의 다른 구현은, 프로세서에 의해, SPS를 비트스트림으로 인코딩하는 단계를 추가로 포함하고, SPS는 서브-픽처의 설명을 포함하고, 서브-픽처 ID는 서브-픽처의 설명을 참조하는 것을 제공한다.

[0017] 선택적으로, 이전 양태들 중 어느 것에서, 상기 양태의 다른 구현은, 서브-픽처의 설명이 CTB 단위의 서브-픽처의 폭, CTB 단위의 서브-픽처의 높이, 및 서브-픽처의 좌측 상단 CTU의 수직 위치 및 서브-픽처의 좌측 상단 CTU의 수평 위치로서 특정된 서브-픽처의 오프셋을 포함하는 것을 제공한다.

[0018] 일 실시예에서, 본 개시내용은 비디오 코딩 디바이스를 포함하고 이는: 프로세서, 프로세서에 결합된 수신기, 프로세서에 결합된 메모리, 및 프로세서에 결합된 송신기를 포함하며, 프로세서, 수신기, 메모리, 및 송신기는 이전 양태들 중 임의의 것의 방법을 수행하도록 구성된다.

[0019] 일 실시예에서, 본 개시내용은 비디오 코딩 디바이스에 의한 사용을 위한 컴퓨터 프로그램 제품을 포함하는 비일시적 컴퓨터 판독가능 매체를 포함하고, 컴퓨터 프로그램 제품은, 프로세서에 의해 실행될 때 비디오 코딩 디바이스로 하여금 이전 양태들 중 임의의 것의 방법을 수행하게 야기하도록 하는, 비일시적 컴퓨터 판독가능 매체 상에 저장된 컴퓨터 실행가능 명령어들을 포함한다.

[0020] 일 실시예에서, 본 개시내용은 디코더를 포함하며, 이는: 슬라이스를 포함하는 서브-픽처를 포함하는 비트스트림을 수신하는 수신 수단; 슬라이스가 직사각형 슬라이스일 때 슬라이스의 서브-픽처 레벨 슬라이스 인덱스를 포함하는 슬라이스 어드레스를 획득하는 획득 수단; 슬라이스 어드레스에 기초하여 서브-픽처의 슬라이스를 디코딩하는 디코딩 수단; 및 디코딩된 비디오 시퀀스의 부분으로서 디스플레이를 위해 슬라이스를 전달하는 전달 수단을 포함한다.

[0021] 선택적으로, 이전 양태들 중 임의의 것에서, 상기 양태의 다른 구현은, 디코더가 이전 양태들 중 임의의 것의 방법을 수행하도록 추가로 구성되는 것을 제공한다.

[0022] 일 실시예에서, 본 개시내용은 인코더를 포함하며, 이는: 서브-픽처를 슬라이스로 분할하는 분할 수단; 슬라이스가 직사각형 슬라이스일 때 슬라이스의 서브-픽처 레벨 슬라이스 인덱스를 포함하는 슬라이스 어드레스를 결정하는 결정 수단; 슬라이스 어드레스를 비트스트림으로 인코딩하고; 서브-픽처의 슬라이스를 비트스트림으로 인코딩하는 인코딩 수단; 및 디코더를 향한 통신을 위해 비트스트림을 저장하는 저장 수단을 포함한다.

[0023] 선택적으로, 이전 양태들 중 어느 것에서, 상기 양태의 다른 구현은, 인코더가 이전 양태들 중 임의의 것의 방법을 수행하도록 추가로 구성되는 것을 제공한다.

[0024] 선택적으로, 이전 양태들 중 어느 것에서, 상기 양태의 다른 구현은, 인코더가 이전 양태들 중 임의의 것의 방법을 수행하도록 추가로 구성되는 것을 제공한다.

[0025] 명료성을 목적으로, 전술한 실시예들 중 어느 하나는 본 개시내용의 범위 내에서 새로운 실시예를 창출하기 위해 다른 전술한 실시예들 중 하나 이상과 조합될 수 있다.

[0026] 이들 및 다른 특징들은 첨부된 도면 및 청구항과 관련하여 취해진 다음의 상세한 설명으로부터 더욱 명확하게 이해될 것이다.

도면의 간단한 설명

[0027] 본 개시내용의 보다 완전한 이해를 위해, 이제, 첨부 도면들 및 상세한 설명과 관련하여 취해진 다음의 간략한 설명을 참조하며, 유사한 참조 번호들은 유사한 부분들을 표현한다.

도 1은 비디오 신호를 코딩하는 예시적인 방법의 흐름도이다.

도 2는 비디오 코딩을 위한 예시적인 코딩 및 디코딩(코덱) 시스템의 개략도이다.

- 도 3은 예시적인 비디오 인코더를 예시하는 개략도이다.
- 도 4는 예시적인 비디오 디코더를 예시하는 개략도이다.
- 도 5a는 서브-픽처들로 파티셔닝된 예시적인 픽처를 예시하는 개략도이다.
- 도 5b는 슬라이스들로 파티셔닝된 예시적인 서브-픽처를 예시하는 개략도이다.
- 도 5c는 타일들로 파티셔닝된 예시적인 슬라이스를 예시하는 개략도이다.
- 도 5d는 CTU들(coding tree units)로 파티셔닝된 예시적인 슬라이스를 예시하는 개략도이다.
- 도 6은 단방향 인터-예측의 예를 예시하는 개략도이다.
- 도 7은 양방향 인터-예측의 예를 예시하는 개략도이다.
- 도 8은 인접한 코딩된 블록들로부터의 후보 모션 벡터들에 기초하여 현재 블록을 코딩하는 예를 예시하는 개략도이다.
- 도 9는 모션 벡터들의 후보 리스트를 결정하기 위한 예시적인 패턴을 예시하는 개략도이다.
- 도 10은 예시적인 인-루프 필터를 예시하는 블록도이다.
- 도 11은 픽처의 서브-픽처의 디코딩을 지원하기 위한 코딩 도구 파라미터들을 포함하는 예시적인 비트스트림을 예시하는 개략도이다.
- 도 12는 예시적인 비디오 코딩 디바이스의 개략도이다.
- 도 13은 서브-픽처에 대한 슬라이스를 어드레싱함으로써 비디오 시퀀스를 비트스트림으로 인코딩하는 예시적인 방법의 흐름도이다.
- 도 14는 슬라이스가 서브-픽처에 대해 어드레싱될 때 비트스트림으로부터 비디오 시퀀스를 디코딩하는 예시적인 방법의 흐름도이다.
- 도 15는 슬라이스가 서브-픽처에 대해 어드레싱될 때 비트스트림 내의 이미지들의 비디오 시퀀스를 코딩하기 위한 예시적인 시스템의 개략도이다.

발명을 실시하기 위한 구체적인 내용

- [0028] 하나 이상의 실시예의 예시적인 구현이 아래에 제공되지만, 개시된 시스템들 및/또는 방법들은, 현재 공지되어 있는 존재하는 간에, 임의의 수의 기법들을 이용하여 구현될 수 있다는 점을 처음부터 이해해야 한다. 본 개시 내용은 본 명세서에서 예시되고 설명되는 예시적인 설계들 및 구현들을 포함하여, 아래에 예시되는 예시적인 구현들, 도면들 및 기법들로 결코 제한되어서는 안 되고, 첨부된 청구항들의 범위와 함께 그들의 등가물들의 전체 범위 내에서 수정될 수 있다.
- [0029] 이하의 두문자어들, ALF(Adaptive Loop Filter), CTB(Coding Tree Block), CTU(Coding Tree Unit), CU(Coding Unit), CVS(Coded Video Sequence), JVET(Joint Video Experts Team), MCTS(Motion-Constrained Tile Set), MTU(Maximum Transfer Unit), NAL(Network Abstraction Layer), POC(Picture Order Count), RBSP(Raw Byte Sequence Payload), SAO(Sample Adaptive Offset), SPS(Sequence Parameter Set), TMVP(Temporal Motion Vector Prediction), VVC(Versatile Video Coding), 및 WD(Working Draft)가 본 명세서에서 사용된다.
- [0030] 데이터 손실을 최소로 하여 비디오 파일들의 크기를 감소시키기 위해 많은 비디오 압축 기법들이 이용될 수 있다. 예를 들어, 비디오 압축 기법들은 비디오 시퀀스들에서 데이터 중복성을 감소시키거나 제거하기 위해 공간(예를 들어, 인트라-픽처) 예측 및/또는 시간(예를 들어, 인터-픽처) 예측을 수행하는 것을 포함할 수 있다. 블록-기반 비디오 코딩을 위해, 비디오 슬라이스(예를 들어, 비디오 픽처 또는 비디오 픽처의 일부)가 비디오 블록들로 파티셔닝될 수 있고, 이는 트리블록들, 코딩 트리 블록(CTB)들, 코딩 트리 유닛(CTU)들, 코딩 유닛(CU)들, 및/또는 코딩 노드들이라고도 지칭될 수 있다. 픽처의 인트라-코딩된(I) 슬라이스 내의 비디오 블록들은 동일한 픽처 내의 이웃 블록들 내의 참조 샘플들에 관하여 공간 예측을 이용하여 코딩된다. 픽처의 인터-코딩된 단방향 예측(P) 또는 양방향 예측(B) 슬라이스 내의 비디오 블록들은 동일한 픽처 내의 이웃 블록들 내의 참조 샘플들에 관하여 공간 예측 또는 다른 참조 픽처들 내의 참조 샘플들에 관하여 시간 예측을 이용하여 코딩될 수 있다. 픽처들은 프레임들 및/또는 이미지들이라고 지칭될 수 있고, 참조 픽처들은 참조 프레임들 및

/또는 참조 이미지들이라고 지칭될 수 있다. 공간 또는 시간 예측은 이미지 블록을 나타내는 예측 블록을 야기한다. 잔차 데이터는 원래의 이미지 블록과 예측 블록 사이의 픽셀 차이들을 나타낸다. 따라서, 인트라-코딩된 블록은 코딩된 블록과 예측 블록 사이의 차이를 나타내는 잔차 데이터 및 예측 블록을 형성하는 참조 샘플들의 블록을 가리키는 모션 벡터에 따라 인코딩된다. 인트라-코딩된 블록은 인트라-코딩 모드 및 잔차 데이터에 따라 인코딩된다. 추가 압축을 위해, 잔차 데이터는 픽셀 도메인으로부터 변환 도메인으로 변환될 수 있다. 이들은 양자화될 수 있는 잔차 변환 계수들을 야기한다. 양자화된 변환 계수들은 초기에 2차원 어레이로 배열될 수 있다. 양자화된 변환 계수들은 변환 계수들의 1차원 벡터를 생성하기 위해 스캐닝될 수 있다. 훨씬 더 많은 압축을 달성하기 위해 엔트로피 코딩이 적용될 수 있다. 그러한 비디오 압축 기법들은 아래에 더 상세히 논의된다.

[0031] 인코딩된 비디오가 정확하게 디코딩될 수 있도록 보장하기 위해, 비디오는 대응하는 비디오 코딩 표준들에 따라 인코딩되고 디코딩된다. 비디오 코딩 표준들은 ITU(International Telecommunication Union) 표준화 섹터(Standardization Sector)(ITU-T) H.261, ISO/IEC(International Organization for Standardization/International Electrotechnical Commission) MPEG(Motion Picture Experts Group)-1 Part 2, ITU-T H.262 또는 ISO/IEC MPEG-2 Part 2, ITU-T H.263, ISO/IEC MPEG-4 Part 2, ITU-T H.264 또는 ISO/IEC MPEG-4 Part 10으로도 알려진 AVC(Advanced Video Coding), 및 ITU-T H.265 또는 MPEG-H Part 2로도 알려진 HEVC(High Efficiency Video Coding)를 포함한다. AVC는 SVC(Scalable Video Coding), MVC(Multiview Video Coding) 및 MVC+D(Multiview Video Coding plus Depth), 및 3차원(3D) AVC(3D-AVC)와 같은 확장들을 포함한다. HEVC는 SHVC(Scalable HEVC), MV-HEVC(Multiview HEVC), 및 3D HEVC(3D-HEVC)와 같은 확장들을 포함한다. ITU-T 및 ISO/IEC의 JVET(joint video experts team)는 VVC(Versatile Video Coding)라고 지칭되는 비디오 코딩 표준을 개발하기 시작하였다. VVC는 알고리즘 설명, VVC WD의 인코더-측 설명, 및 참조 소프트웨어를 제공하는 JVET-M1001-v6을 포함하는 WD(Working Draft)에 포함된다.

[0032] 비디오 이미지를 코딩하기 위해, 이미지는 먼저 파티셔닝되고, 파티션들은 비트스트림 내로 코딩된다. 다양한 픽처 파티셔닝 스킴들이 이용가능하다. 예를 들어, 이미지는 정규 슬라이스들, 종속 슬라이스들, 타일들로, 및/또는 WPP(Wavefront Parallel Processing)에 따라 파티셔닝될 수 있다. 단순화를 위해, HEVC는 비디오 코딩을 위해 슬라이스를 CTB들의 그룹들로 파티셔닝할 때 정규 슬라이스들, 종속 슬라이스들, 타일들, WPP, 및 이들의 조합들만이 사용될 수 있도록 인코더들을 제한한다. 그러한 파티셔닝은 MTU(Maximum Transfer Unit) 크기 매칭, 병렬 처리, 및 감소된 중단간 지연을 지원하기 위해 적용될 수 있다. MTU는 단일 패킷에서 송신될 수 있는 데이터의 최대량을 나타낸다. 패킷 페이로드가 MTU를 초과하면, 해당 페이로드는 단편화(fragmentation)로 불리는 프로세스를 통해 2개의 패킷으로 분할된다.

[0033] 단순히 슬라이스라고도 지칭되는 정규 슬라이스는, 루프 필터링 연산들로 인한 일부 상호의존성에도 불구하고, 동일한 픽처 내의 다른 정규 슬라이스들과 독립적으로 재구성될 수 있는 이미지의 파티셔닝된 부분이다. 각각의 정규 슬라이스는 송신을 위해 그 자신의 NAL(Network Abstraction Layer) 유닛에 캡슐화된다. 또한, 인-픽처 예측(in-picture prediction)(인트라 샘플 예측, 모션 정보 예측, 코딩 모드 예측) 및 슬라이스 경계들에 걸친 엔트로피 코딩 종속성은 독립적인 재구성을 지원하기 위해 디스에이블될 수 있다. 그러한 독립적인 재구성은 병렬화를 지원한다. 예를 들어, 정규 슬라이스 기반 병렬화는 최소의 프로세서 간 또는 코어 간 통신을 이용한다. 그러나, 각각의 정규 슬라이스는 독립적이므로, 각각의 슬라이스는 별개의 슬라이스 헤더와 연관된다. 정규 슬라이스들의 사용은 각각의 슬라이스에 대한 슬라이스 헤더의 비트 비용 및 슬라이스 경계들에 걸친 예측의 부재로 인해 상당한 코딩 오버헤드를 초래할 수 있다. 또한, 정규 슬라이스들은 MTU 크기 요구들에 대한 매칭을 지원하기 위해 이용될 수 있다. 구체적으로, 정규 슬라이스가 별개의 NAL 유닛 내에 캡슐화되고 독립적으로 코딩될 수 있으므로, 각각의 정규 슬라이스는 슬라이스를 다수의 패킷으로 분할하는 것을 피하기 위해 MTU 스킴들에서의 MTU보다 작아야 한다. 그에 따라, 병렬화의 목표와 MTU 크기 매칭의 목표는 픽처 내의 슬라이스 레이아웃에 대해 모순되는 요구들을 둘 수 있다.

[0034] 종속 슬라이스들은 정규 슬라이스들과 유사하지만, 단축된 슬라이스 헤더들을 갖고 인-픽처 예측을 깨지 않고 이미지 트리블록 경계들의 파티셔닝을 허용한다. 따라서, 종속 슬라이스들은 정규 슬라이스가 다수의 NAL 유닛으로 단편화되는 것을 허용하고, 이는 전체 정규 슬라이스의 인코딩이 완료되기 전에 정규 슬라이스의 일부가 송신되는 것을 허용함으로써 감소된 중단간 지연을 제공한다.

[0035] 타일은 타일들의 열들 및 행들을 생성하는 수평 및 수직 경계들에 의해 생성된 이미지의 파티셔닝된 부분이다. 타일들은 래스터 스캔 순서로(우측에서 좌측으로 및 상단에서 하단으로) 코딩될 수 있다. CTB들의 스캔 순서는 타일 내에서 로컬이다. 따라서, 첫 번째 타일 내의 CTB들은, 다음 타일 내의 CTB들로 진행하기 전에, 래스터

스캔 순서로 코딩된다. 정규 슬라이스들과 유사하게, 타일들은 인-픽처 예측 종속성들뿐만 아니라 엔트로피 디코딩 종속성들을 갖는다. 그러나, 타일들은 개별 NAL 유닛들 내에 포함되지 않을 수 있고, 따라서 타일들은 MTU 크기 매칭에 사용되지 않을 수 있다. 각각의 타일은 하나의 프로세서/코어에 의해 처리될 수 있고, 이웃 타일들을 디코딩하는 처리 유닛들 사이의 인-픽처 예측을 위해 이용되는 프로세서 간/코어 간 통신은 (인접한 타일들이 동일한 슬라이스에 있을 때) 공유 슬라이스 헤더를 전달하는 것, 및 재구성된 샘플들 및 메타데이터의 루프 필터링 관련 공유를 수행하는 것으로 제한될 수 있다. 하나보다 많은 타일이 슬라이스에 포함될 때, 슬라이스 내의 첫 번째 엔트리 포인트 오프셋 이외의 각각의 타일에 대한 엔트리 포인트 바이트 오프셋은 슬라이스 헤더에서 시그널링될 수 있다. 각각의 슬라이스 및 타일에 대해, 다음 조건들 중 적어도 하나가 충족되어야 한다: 1) 슬라이스 내의 모든 코딩된 트리블록들이 동일한 타일에 속한다; 그리고 2) 타일 내의 모든 코딩된 트리블록들이 동일한 슬라이스에 속한다.

[0036] WPP에서, 이미지는 CTB들의 단일 행들로 파티셔닝된다. 엔트로피 디코딩 및 예측 메커니즘들은 다른 행들에서의 CTB들로부터의 데이터를 사용할 수 있다. 병렬 처리는 CTB 행들의 병렬 디코딩을 통해 가능하게 된다. 예를 들어, 현재 행은 이전 행과 병렬로 디코딩될 수 있다. 그러나, 현재 행의 디코딩은 이전 행들의 디코딩 프로세스로부터 2개의 CTB만큼 지연된다. 이 지연은 현재 행 내의 현재 CTB 위의 그리고 우측의 CTB 및 위의 CTB와 관련된 데이터가 현재 CTB가 코딩되기 전에 이용가능한 것을 보장한다. 이 접근법은 그래픽으로 표현될 때 파면(wavefront)으로서 나타난다. 이러한 시차를 둔 시작은 이미지가 CTB 행들을 포함하는 만큼의 프로세서들/코어들까지 병렬화를 허용한다. 픽처 내의 이웃 트리블록 행들 간의 인-픽처 예측이 허용되기 때문에, 인-픽처 예측을 가능하게 하는 프로세서 간/코어 간 통신이 상당할 수 있다. WPP 파티셔닝은 NAL 유닛 크기를 고려한다. 따라서, WPP는 MTU 크기 매칭을 지원하지 않는다. 그러나, 정규 슬라이스들은, 원하는 대로 MTU 크기 매칭을 구현하기 위해, 특정 코딩 오버헤드로, WPP와 함께 사용될 수 있다.

[0037] 타일들은 모션 제약된 타일 세트(motion constrained tile set)들을 또한 포함할 수 있다. 모션 제약된 타일 세트(MCTS)는 관련된 모션 벡터들이 MCTS 내부의 풀-샘플(full-sample) 위치들 및 보간을 위해 MCTS 내부의 풀-샘플 위치들만을 요구하는 분수-샘플(fractional-sample) 위치들을 가리키도록 제한되도록 설계된 타일 세트이다. 또한, MCTS 외부의 블록들로부터 도출된 시간 모션 벡터 예측을 위한 모션 벡터 후보들의 사용이 허용되지 않는다. 이러한 방식으로, 각각의 MCTS는 MCTS에 포함되지 않은 타일들의 존재 없이 독립적으로 디코딩될 수 있다. 비트스트림 내의 MCTS들의 존재를 나타내고 MCTS들을 시그널링하기 위해 시간 MCTS들 SEI(supplemental enhancement information) 메시지들이 사용될 수 있다. MCTS들 SEI 메시지는 MCTS 세트에 대한 적합 비트스트림(conforming bitstream)을 생성하기 위해 MCTS 서브-비트스트림 추출(SEI 메시지의 시맨틱스의 일부로서 특정됨)에 사용될 수 있는 보충 정보를 제공한다. 정보는 다수의 추출 정보 세트를 포함하고, 각각은 다수의 MCTS 세트를 정의하고 MCTS 서브-비트스트림 추출 프로세스 동안 사용될 대체 비디오 파라미터 세트(video parameter set, VPS)들, 시퀀스 파라미터 세트(sequence parameter set, SPS)들, 및 픽처 파라미터 세트(picture parameter set, PPS)들의 RBSP(raw bytes sequence payload) 바이트들을 포함한다. MCTS 서브-비트스트림 추출 프로세스에 따라 서브-비트스트림을 추출할 때, 파라미터 세트들(VPS들, SPS들, 및 PPS들)이 재가입 또는 대체될 수 있고, 슬라이스 헤더들은 슬라이스 어드레스 관련 구문 요소들(first_slice_segment_in_pic_flag 및 slice_segment_address를 포함함) 중 하나 또는 전부가 추출된 서브-비트스트림에서 상이한 값들을 이용할 수 있기 때문에 업데이트될 수 있다.

[0038] 픽처들은 또한 하나 이상의 서브-픽처로 파티셔닝될 수 있다. 픽처를 서브-픽처로 파티셔닝하는 것은 픽처의 상이한 부분들이 코딩 관점과 상이하게 취급되게 할 수 있다. 예를 들어, 서브-픽처는 다른 서브-픽처들을 추출하지 않고 추출 및 디스플레이될 수 있다. 다른 예로서, 상이한 서브-픽처들이 상이한 해상도들로 디스플레이되거나, (예를 들어, 원격 회의 애플리케이션들에서) 서로에 대해 재배치되거나, 서브-픽처들이 공통 픽처로부터의 데이터를 총괄하여 포함하더라도 별개의 픽처들로서 다른 방식으로 코딩될 수 있다.

[0039] 서브-픽처들의 예시적인 구현은 다음과 같다. 픽처는 하나 이상의 서브-픽처로 파티셔닝될 수 있다. 서브-픽처는 0인 어드레스를 갖는 슬라이스/타일 그룹에서 시작하는 슬라이스들/타일 그룹들의 직사각형 또는 정사각형 세트이다. 각각의 서브-픽처는 상이한 PPS를 지칭할 수 있으며, 따라서 각각의 서브-픽처는 상이한 파티셔닝 메커니즘을 이용할 수 있다. 서브-픽처들은 디코딩 프로세스에서 픽처들과 같이 취급될 수 있다. 현재 서브-픽처를 디코딩하기 위해 사용되는 현재 참조 픽처는 디코딩된 픽처 버퍼 내의 참조 픽처들로부터 현재 서브-픽처와 동위치의 영역을 추출함으로써 생성될 수 있다. 추출된 영역은 디코딩된 서브-픽처일 수 있고, 따라서 인터-예측은 동일한 크기의 서브-픽처들과 픽처 내의 동일한 위치 사이에서 발생할 수 있다. 타일 그룹은 서브-픽처의 타일 래스터 스캔 내의 타일들의 시퀀스일 수 있다. 픽처 내의 서브-픽처의 위치를 결정하기 위해 이하

가 도출될 수 있다. 각각의 서브-픽처는 픽처 경계들 내의 서브-픽처에 맞기에 충분히 큰 픽처 내의 CTU 래스터 스캔 순서에서 다음 비점유 위치에 포함될 수 있다.

[0040] 다양한 비디오 코딩 시스템에 의해 이용되는 서브-픽처 스킴들은 코딩 효율 및/또는 기능성을 감소시키는 다양한 문제점을 포함한다. 본 개시내용은 이러한 문제들에 대한 다양한 해결책들을 포함한다. 제1 예시적인 문제에서, 인터-예측은 여러 인터-예측 모드들 중 하나에 따라 수행될 수 있다. 특정 인터-예측 모드들은 인코더 및 디코더 둘 다에서 모션 벡터 예측자들의 후보 리스트들을 생성한다. 이것은 인코더가 전체 모션 벡터를 시그널링하는 대신에 후보 리스트로부터의 인덱스를 시그널링함으로써 모션 벡터를 시그널링하는 것을 허용한다. 또한, 일부 시스템들은 독립적인 추출을 위해 서브-픽처들을 인코딩한다. 이것은 현재 서브-픽처가 다른 서브-픽처들로부터의 정보를 디코딩하지 않고 디코딩 및 디스플레이되는 것을 허용한다. 이것은 모션 벡터가 가리킨 데이터가 디코딩되지 않을 수 있고 따라서 이용가능하지 않을 수 있기 때문에 서브-픽처의 외부를 가리키는 모션 벡터가 이용될 때 에러들을 야기할 수 있다.

[0041] 따라서, 제1 예에서는, 서브-픽처가 픽처로서 취급되어야 한다는 것을 나타내는 플래그가 본 명세서에 개시된다. 이 플래그는 서브-픽처의 별개의 추출을 지원하도록 설정된다. 플래그가 설정될 때, 동위치의 블록으로부터 획득된 모션 벡터 예측자들은 서브-픽처 내부를 가리키는 모션 벡터들만을 포함한다. 서브-픽처의 외부를 가리키는 임의의 모션 벡터 예측자들은 제외된다. 이것은 서브-픽처의 외부를 가리키는 모션 벡터들이 선택되지 않고 연관된 에러들이 회피되는 것을 보장한다. 동위치의 블록은 현재 픽처와 상이한 픽처로부터의 블록이다. 보간 필터들과 같은 다른 프로세스들이 그러한 모션 벡터 예측자들에 대한 에러들을 방지할 수 있으므로, 현재 픽처 내의 블록들(동위치가 아닌 블록들)로부터의 모션 벡터 예측자들은 서브-픽처의 외부를 가리킬 수 있다. 그에 따라, 본 예는 서브-픽처 추출을 수행할 때 에러들을 방지함으로써 비디오 인코더/디코더(코덱)에 추가적인 기능성을 제공한다.

[0042] 제2 예에서는, 서브-픽처가 픽처로서 취급되어야 한다는 것을 나타내는 플래그가 본 명세서에 개시된다. 현재 서브-픽처가 픽처처럼 취급될 때, 현재 서브-픽처는 다른 서브-픽처들을 참조하지 않고 추출되어야 한다. 구체적으로, 본 예는 보간 필터들을 적용할 때 적용되는 클리핑 함수를 이용한다. 이 클리핑 함수는 보간 필터가 개별 추출을 지원하기 위해 서브-픽처들 간의 분리를 유지하기 위해 인접한 서브-픽처들로부터의 데이터에 의존하지 않도록 보장한다. 이와 같이, 클리핑 함수는 플래그가 설정되고 모션 벡터가 현재 서브-픽처의 외부를 가리킬 때 적용된다. 그 후, 보간 필터가 클리핑 함수의 결과들에 적용된다. 따라서, 본 예는 서브-픽처 추출을 수행할 때 에러를 방지함으로써 비디오 코덱에 추가적인 기능성을 제공한다. 이와 같이, 제1 예 및 제2 예는 제1 예시적인 문제를 다룬다.

[0043] 제2 예시적인 문제에서, 비디오 코딩 시스템들은 픽처들을 서브-픽처들, 슬라이스들, 타일들, 및/또는 코딩 트리 유닛들로 파티셔닝하고나서, 이들은 블록들로 파티셔닝된다. 그 후, 그런 블록들은 디코더를 향한 송신을 위해 인코딩된다. 이러한 블록들을 디코딩하는 것은 다양한 타입들의 잡음을 포함하는 디코딩된 이미지를 야기할 수 있다. 이러한 문제들을 정정하기 위해, 비디오 코딩 시스템들은 블록 경계들에 걸쳐 다양한 필터들을 적용할 수 있다. 이러한 필터들은 블로킹, 양자화 잡음, 및 다른 바람직하지 않은 코딩 아티팩트들을 제거할 수 있다. 전술한 바와 같이, 일부 시스템들은 독립적인 추출을 위해 서브-픽처들을 인코딩한다. 이것은 현재 서브-픽처가 다른 서브-픽처들로부터의 정보를 디코딩하지 않고 디코딩 및 디스플레이되는 것을 허용한다. 그러한 시스템들에서, 서브-픽처는 인코딩을 위한 블록들로 파티셔닝될 수 있다. 따라서, 서브-픽처 에지를 따르는 블록 경계들은 서브-픽처 경계들과 정렬될 수 있다. 일부 경우들에서, 블록 경계들은 또한 타일 경계들과 정렬될 수 있다. 필터들은 그러한 블록 경계들에 걸쳐 적용될 수 있으며, 따라서 서브-픽처 경계들 및/또는 타일 경계들에 걸쳐 적용될 수 있다. 이것은 현재 서브-픽처가 독립적으로 추출될 때 에러를 유발할 수 있는데, 그 이유는 인접한 서브-픽처로부터의 데이터가 이용 가능하지 않을 때 필터링 프로세스가 예상치 못한 방식으로 동작할 수 있기 때문이다.

[0044] 제3 예에서는, 서브-픽처 레벨에서 필터링을 제어하는 플래그가 본 명세서에 개시된다. 플래그가 서브-픽처에 대해 설정될 때, 필터들은 서브-픽처 경계에 걸쳐 적용될 수 있다. 플래그가 설정되지 않을 때, 필터들은 서브-픽처 경계에 걸쳐 적용되지 않는다. 이러한 방식으로, 필터들은 개별 추출을 위해 인코딩되는 서브-픽처들에 대해 턴오프될 수 있거나 또는 그룹으로서 디스플레이를 위해 인코딩되는 서브-픽처들에 대해 턴온될 수 있다. 이와 같이, 본 예는 서브-픽처 추출을 수행할 때 필터 관련 에러들을 방지함으로써 비디오 코덱에 추가적인 기능성을 제공한다.

[0045] 제4 예에서는, 타일 레벨에서 필터링을 제어하도록 설정될 수 있는 플래그가 본 명세서에 개시된다. 플래그가

타일에 대해 설정될 때, 필터들은 타일 경계에 걸쳐 적용될 수 있다. 플래그가 설정되지 않을 때, 필터들은 타일 경계에 걸쳐 적용되지 않는다. 이러한 방식으로, 필터들은 타일 경계들에서 사용하기 위해(예를 들어, 타일의 내부 부분들을 계속 필터링하면서) 턴 오프 또는 온될 수 있다. 따라서, 본 예는 타일 경계들에 걸쳐 선택적인 필터링을 지원함으로써 비디오 코덱에 추가적인 기능성을 제공한다. 이와 같이, 제3 예 및 제4 예는 제2 예시적인 문제를 다룬다.

[0046] 제3 예시적인 문제에서, 비디오 코딩 시스템들은 픽처를 서브-픽처들로 파티셔닝할 수 있다. 이것은 비디오를 코딩할 때 상이한 서브-픽처들이 상이하게 취급될 수 있게 한다. 예를 들어, 서브-픽처들은 개별적으로 추출되고 디스플레이되고, 애플리케이션 레벨 변경들 등에 기초하여 독립적으로 크기 조정될 수 있다. 일부 경우들에서, 서브-픽처들은 픽처를 타일들로 파티셔닝하고 타일들을 서브-픽처들에 할당함으로써 생성될 수 있다. 일부 비디오 코딩 시스템들은 서브-픽처에 포함된 타일들의 관점에서 서브-픽처 경계들을 기술한다. 그러나, 일부 픽처들에서는 타일링 스킵들이 이용되지 않을 수 있다. 따라서, 그러한 경계 설명들은 서브-픽처들의 사용을 타일들을 이용하는 픽처들로 제한할 수 있다.

[0047] 제5 예에서는, CTB들 및/또는 CTU들의 관점에서 서브-픽처 경계들을 시그널링하기 위한 메커니즘이 본 명세서에 개시된다. 구체적으로, 서브-픽처의 폭 및 높이는 CTB의 단위로 시그널링될 수 있다. 또한, 서브-픽처의 좌측 상단 CTU의 위치는 CTB들에서 측정된 픽처의 좌측 상단 CTU로부터의 오프셋으로서 시그널링될 수 있다. CTU 및 CTB 크기는 미리 결정된 값으로 설정될 수 있다. 따라서, CTB들 및 CTU들의 관점에서 서브-픽처 치수 및 위치를 시그널링하는 것은 디코더가 디스플레이를 위해 서브-픽처를 배치하기에 충분한 정보를 제공한다. 이것은 타일들이 이용되지 않을 때에도 서브-픽처들이 이용되게 한다. 또한, 이 시그널링 메커니즘 둘 다 복잡성을 회피하고 비교적 적은 비트를 사용하여 코딩될 수 있다. 이와 같이, 본 예는 서브-픽처들이 타일들과 독립적으로 이용되게 함으로써 비디오 코덱에 추가적인 기능성을 제공한다. 또한, 본 예는 코딩 효율을 증가시키고, 따라서 인코더 및/또는 디코더에서 프로세서, 메모리, 및/또는 네트워크 자원들의 사용을 감소시킨다. 이와 같이, 제5 예는 제3 예시적인 문제를 다룬다.

[0048] 제4 예시적인 문제에서는, 픽처가 인코딩을 위해 복수의 슬라이스로 파티셔닝될 수 있다. 일부 비디오 코딩 시스템들에서는, 슬라이스들이 픽처에 대한 그들의 위치에 기초하여 어드레싱된다. 또 다른 비디오 코딩 시스템들은 서브-픽처들의 개념을 이용한다. 전술한 바와 같이, 서브-픽처는 코딩 관점에서 다른 서브-픽처들과 다르게 취급될 수 있다. 예를 들어, 서브-픽처는 다른 서브-픽처들과 독립적으로 추출되고 디스플레이될 수 있다. 이러한 경우에, 픽처 위치에 기초하여 생성되는 슬라이스 어드레스들은, 상당한 수의 예상된 슬라이스 어드레스들이 생략됨에 따라 적절히 동작하는 것을 중단할 수 있다. 일부 비디오 코딩 시스템들은 서브-픽처 추출을 지원하기 위해 슬라이스 어드레스들을 변경하기 위한 요청시에 슬라이스 헤더들을 동적으로 재기입함으로써 이 문제를 다룬다. 이러한 프로세스는 자원 집약적일 수 있는데, 그 이유는 이 프로세스가 사용자가 서브-픽처를 보도록 요청할 때마다 일어날 수 있기 때문이다.

[0049] 제6 예에서는, 슬라이스를 포함하는 서브-픽처와 관련하여 어드레싱되는 슬라이스들이 본 명세서에 개시된다. 예를 들어, 슬라이스 헤더는 서브-픽처 식별자(ID) 및 서브-픽처에 포함된 각각의 슬라이스의 어드레스를 포함할 수 있다. 또한, SPS(sequence parameter set)는 서브-픽처 ID에 의해 참조될 수 있는 서브-픽처의 치수들을 포함할 수 있다. 따라서, 슬라이스 헤더는 서브-픽처의 개별 추출이 요청될 때 재기입될 필요가 없다. 슬라이스 헤더 및 SPS는 디스플레이를 위해 서브-픽처에 슬라이스들을 배치하는 것을 지원하기에 충분한 정보를 포함한다. 이와 같이, 본 예는 코딩 효율을 증가시키고 및/또는 슬라이스 헤더의 중복 재기입을 피하므로, 인코더 및/또는 디코더에서 프로세서, 메모리, 및/또는 네트워크 자원의 이용을 감소시킨다. 따라서, 제6 예는 제4 예시적인 문제를 다룬다.

[0050] 도 1은 비디오 신호를 코딩하는 예시적인 동작 방법(100)의 흐름도이다. 구체적으로, 비디오 신호는 인코더에서 인코딩된다. 인코딩 프로세스는 비디오 파일 크기를 감소시키기 위해 다양한 메커니즘을 이용하여 비디오 신호를 압축한다. 더 작은 파일 크기는 압축된 비디오 파일이 사용자를 향해 송신되는 것을 허용하면서, 연관된 대역폭 오버헤드를 감소시킨다. 디코더는 그 후 최종 사용자에게 디스플레이하기 위한 원래의 비디오 신호를 재구성하기 위해 압축된 비디오 파일을 디코딩한다. 디코딩 프로세스는 일반적으로 인코딩 프로세스를 미러링하여 디코더가 비디오 신호를 일관되게 재구성하는 것을 허용한다.

[0051] 단계 101에서, 비디오 신호는 인코더에 입력된다. 예를 들어, 비디오 신호는 메모리에 저장된 압축되지 않은 비디오 파일일 수 있다. 다른 예로서, 비디오 파일은 비디오 카메라와 같은 비디오 캡처 디바이스에 의해 캡처되고, 비디오의 라이브 스트리밍을 지원하도록 인코딩될 수 있다. 비디오 파일은 오디오 컴포넌트 및 비디오

컴포넌트 둘 다를 포함할 수 있다. 비디오 컴포넌트는 시퀀스에서 볼 때, 모션의 시각적 느낌을 제공하는 일련의 이미지 프레임들을 포함한다. 프레임들은 본 명세서에서 루마 성분들(또는 루마 샘플들)이라고 지칭되는 광, 및 크로마 성분들(또는 색 샘플들)이라고 지칭되는 색의 관점에서 표현되는 픽셀들을 포함한다. 일부 예들에서, 프레임들은 3차원 보기를 지원하기 위한 깊이 값들을 또한 포함할 수 있다.

[0052] 단계 103에서, 비디오는 블록들로 파티셔닝된다. 파티셔닝은 각각의 프레임 내의 픽셀들을 압축을 위해 정사각형 및/또는 직사각형 블록들로 세분하는 것을 포함한다. 예를 들어, HEVC(High Efficiency Video Coding)(H.265 및 MPEG-H Part 2로도 알려짐)에서, 프레임은 미리 정의된 크기(예를 들어, 64 픽셀 x 64 픽셀)의 블록들인 코딩 트리 유닛(CTU)들로 먼저 분할될 수 있다. CTU들은 루마 및 크로마 샘플들 둘 다를 포함한다. 추가 인코딩을 지원하는 구성들이 달성될 때까지 CTU들을 블록들로 분할하고 그 후 블록들을 재귀적으로 세분하기 위해 코딩 트리들이 이용될 수 있다. 예를 들어, 프레임의 루마 성분들은 개별 블록들이 비교적 균일한 광 값들을 포함할 때까지 세분될 수 있다. 또한, 프레임의 크로마 성분들은 개별 블록들이 비교적 균일한 색 값들을 포함할 때까지 세분될 수 있다. 따라서, 파티셔닝 메커니즘들은 비디오 프레임들의 콘텐츠에 따라 달라진다.

[0053] 단계 105에서, 단계 103에서 파티셔닝된 이미지 블록들을 압축하기 위해 다양한 압축 메커니즘들이 이용된다. 예를 들어, 인터-예측 및/또는 인트라-예측이 이용될 수 있다. 인터-예측은 공통 장면 내의 객체들이 연속적인 프레임들에 나타나는 경향이 있다는 사실을 이용하도록 설계된다. 따라서, 참조 프레임 내의 객체를 묘사하는 블록은 인접 프레임들에서 반복적으로 기술될 필요가 없다. 구체적으로, 테이블과 같은 객체는 다수의 프레임에 걸쳐 일정한 위치에 머무를 수 있다. 따라서, 테이블은 한 번 기술되고 인접 프레임들은 참조 프레임을 다시 참조할 수 있다. 다수의 프레임에 걸쳐 객체들을 매칭시키기 위해 패턴 매칭 메커니즘들이 이용될 수 있다. 또한, 예를 들어, 객체 이동 또는 카메라 이동으로 인해 다수의 프레임에 걸쳐 이동하는 객체들이 표현될 수 있다. 특정한 예로서, 비디오는 다수의 프레임에 걸쳐 스크린을 가로질러 이동하는 자동차를 보여줄 수 있다. 모션 벡터들은 그러한 이동을 기술하기 위해 이용될 수 있다. 모션 벡터는 프레임 내의 객체의 좌표들로부터 참조 프레임 내의 객체의 좌표들로의 오프셋을 제공하는 2차원 벡터이다. 그와 같이, 인터-예측은 참조 프레임 내의 대응하는 블록으로부터의 오프셋을 나타내는 모션 벡터들의 세트로서 현재 프레임 내의 이미지 블록을 인코딩할 수 있다.

[0054] 인트라-예측은 공통 프레임 내의 블록들을 인코딩한다. 인트라-예측은 루마 및 크로마 성분들이 프레임에서 클러스터링하는 경향이 있다는 사실을 이용한다. 예를 들어, 나무의 일부에서 녹색의 부분(patch)은 유사한 녹색의 부분들에 인접하여 배치되는 경향이 있다. 인트라-예측은 다수의 방향 예측 모드(예를 들어, HEVC에서 33), 평면 모드, 및 직류(DC) 모드를 이용한다. 방향 모드들은 현재 블록이 대응하는 방향으로 이웃 블록의 샘플들과 유사/동일하다는 것을 나타낸다. 평면 모드는 행/열(예를 들어, 평면)을 따른 일련의 블록들이 행의 예지들에서의 이웃 블록들에 기초하여 보간될 수 있다는 것을 나타낸다. 평면 모드는, 사실상, 값들을 변형함에 있어서 비교적 일정한 기울기를 이용하여 행/열에 걸쳐 광/색의 매끄러운 전이를 나타낸다. DC 모드는 경계 평활화를 위해 이용되고 블록이 방향 예측 모드들의 각도 방향들과 연관된 모든 이웃 블록들의 샘플들과 연관된 평균 값과 유사/동일하다는 것을 나타낸다. 따라서, 인트라-예측 블록들은 실제 값들 대신에 다양한 관계 예측 모드 값들로서 이미지 블록들을 표현할 수 있다. 또한, 인터-예측 블록들은 실제 값들 대신에 모션 벡터 값들로서 이미지 블록들을 표현할 수 있다. 어느 경우든, 예측 블록들은 일부 경우들에서 이미지 블록들을 정확하게 표현하지 않을 수 있다. 임의의 차이들이 잔차 블록들에 저장된다. 파일을 추가로 압축하기 위해 잔차 블록들에 변환들이 적용될 수 있다.

[0055] 단계 107에서, 다양한 필터링 기법들이 적용될 수 있다. HEVC에서, 필터들은 인-루프 필터링 스킴(in-loop filtering scheme)에 따라 적용된다. 위에서 논의된 블록 기반 예측은 디코더에서 농담이 고르지 않은 이미지들(blocky images)의 생성을 야기할 수 있다. 또한, 블록 기반 예측 스킴은 블록을 인코딩한 다음 인코딩된 블록을 참조 블록으로서 나중의 사용을 위해 재구성할 수 있다. 인-루프 필터링 스킴은 블록들/프레임들에 잡음 억제 필터들, 디블로킹 필터들, 적응 루프 필터들, 및 샘플 적응 오프셋(SAO) 필터들을 반복적으로 적용한다. 이들 필터는 인코딩된 파일이 정확하게 재구성될 수 있도록 그러한 블로킹 아티팩트들을 완화시킨다. 또한, 이들 필터는 재구성된 참조 블록들에서 아티팩트들을 완화시켜 아티팩트들이 재구성된 참조 블록들에 기초하여 인코딩되는 후속 블록들에서 추가적인 아티팩트들을 생성할 가능성이 더 적다.

[0056] 비디오 신호가 파티셔닝되고, 압축되고, 필터링되면, 결과 데이터는 단계 109에서 비트스트림에 인코딩된다. 비트스트림은 위에서 논의된 데이터뿐만 아니라 디코더에서 적절한 비디오 신호 재구성을 지원하기 위해 요구되는 임의의 시그널링 데이터를 포함한다. 예를 들어, 그러한 데이터는 파티션 데이터, 예측 데이터, 잔차 블록

들, 및 디코더에 코딩 명령어들을 제공하는 다양한 플러그들을 포함할 수 있다. 비트스트림은 요청 시에 디코더를 향한 송신을 위해 메모리에 저장될 수 있다. 비트스트림은 또한 복수의 디코더를 향해 브로드캐스트 및/또는 멀티캐스트될 수 있다. 비트스트림의 생성은 반복적 프로세스이다. 따라서, 단계들 101, 103, 105, 107 및 109는 많은 프레임들 및 블록들에 걸쳐 연속적으로 및/또는 동시에 발생할 수 있다. 도 1에 도시된 순서는 논의의 명료성 및 편의를 위해 제시되고, 비디오 코딩 프로세스를 특징한 순서로 제한하도록 의도되지 않는다.

[0057] 디코더는 단계 111에서 비트스트림을 수신하고 디코딩 프로세스를 시작한다. 구체적으로, 디코더는 엔트로피 디코딩 스킴을 이용하여 비트스트림을 대응하는 구문 및 비디오 데이터로 변환한다. 디코더는 단계 111에서 비트스트림으로부터의 구문 데이터를 이용하여 프레임들에 대한 파티션들을 결정한다. 단계 103에서 파티셔닝은 블록 파티셔닝의 결과들과 매칭해야 한다. 이제부터 단계 111에서 이용되는 바와 같은 엔트로피 인코딩/디코딩이 설명된다. 인코더는 압축 프로세스 동안, 입력 이미지(들) 내의 값들의 공간적 배치에 기초하여 여러 가능한 선택들로부터 블록 파티셔닝 스킴들을 선택하는 것과 같은, 많은 선택들을 행한다. 정확한 선택들을 시그널링하는 것은 많은 수의 빈들을 이용할 수 있다. 본 명세서에서 사용되는, 빈은 변수로서 취급되는 이진 값(예를 들어, 컨텍스트에 따라 달라질 수 있는 비트 값)이다. 엔트로피 코딩은 인코더가 특정한 경우에 대해 명확하게 실행가능하지 않은 임의의 옵션들을 버려, 허용가능한 옵션들(예를 들어, 후보들)의 세트를 남겨두는 것을 허용한다. 그 후 각각의 허용가능한 옵션에는 코드 워드가 할당된다. 코드 워드들의 길이는 허용가능한 옵션들의 수에 기초한다(예를 들어, 2개의 옵션에 대한 하나의 빈, 3개 내지 4개의 옵션에 대한 2개의 빈 등). 그 후 인코더는 선택된 옵션에 대한 코드 워드를 인코딩한다. 이 스킴은 코드 워드들의 크기를 감소시키는데 그 이유는 코드 워드들이 잠재적으로 큰 세트의 모든 가능한 옵션들로부터의 선택을 고유하게 나타내는 것과는 대조적으로 허용가능한 옵션들의 작은 서브-세트로부터의 선택을 고유하게 나타내기 위해 원하는 만큼 크기 때문이다. 디코더는 그 후 인코더와 유사한 방식으로 허용가능한 옵션들의 세트를 결정함으로써 선택을 디코딩한다. 허용가능한 옵션들의 세트를 결정함으로써, 디코더는 코드 워드를 판독하고 인코더에 의해 행해진 선택을 결정할 수 있다.

[0058] 단계 113에서, 디코더는 블록 디코딩을 수행한다. 구체적으로, 디코더는 잔차 블록들을 생성하기 위해 역변환들을 이용한다. 그 후 디코더는 잔차 블록들 및 대응하는 예측 블록들을 이용하여 파티셔닝에 따라 이미지 블록들을 재구성한다. 예측 블록들은 단계 105에서 인코더에서 생성된 인트라-예측 블록들 및 인터-예측 블록들 둘 다를 포함할 수 있다. 재구성된 이미지 블록들은 그 후 단계 111에서 결정된 파티셔닝 데이터에 따라 재구성된 비디오 신호의 프레임들 내에 배치된다. 단계 113에 대한 구문은 또한 위에서 논의된 바와 같이 엔트로피 코딩을 통해 비트스트림에서 시그널링될 수 있다.

[0059] 단계 115에서, 인코더에서의 단계 107과 유사한 방식으로 재구성된 비디오 신호의 프레임들에 대해 필터링이 수행된다. 예를 들어, 블로킹 아티팩트들을 제거하기 위해 프레임들에 잡음 억제 필터들, 디블로킹 필터들, 적응 루프 필터들, 및 SAO 필터들이 적용될 수 있다. 프레임들이 필터링되면, 최종 사용자에게 의한 보기를 위해 단계 117에서 비디오 신호가 디스플레이에 출력될 수 있다.

[0060] 도 2는 비디오 코딩을 위한 예시적인 코딩 및 디코딩(코덱) 시스템(200)의 개략도이다. 구체적으로, 코덱 시스템(200)은 동작 방법(100)의 구현을 지원하기 위한 기능성을 제공한다. 코덱 시스템(200)은 인코더 및 디코더 둘 다에서 이용되는 컴포넌트들을 묘사하기 위해 일반화되어 있다. 코덱 시스템(200)은 동작 방법(100)에서의 단계들 101 및 103과 관련하여 논의된 바와 같이 비디오 신호를 수신하여 파티셔닝하고, 이는 파티셔닝된 비디오 신호(201)를 야기한다. 코덱 시스템(200)은 그 후 방법(100)에서의 단계들 105, 107, 및 109와 관련하여 논의된 바와 같이 인코더로서의 역할을 할 때 파티셔닝된 비디오 신호(201)를 코딩된 비트스트림으로 압축한다. 디코더로서의 역할을 할 때, 코덱 시스템(200)은 동작 방법(100)에서의 단계들 111, 113, 115, 및 117과 관련하여 논의된 바와 같이 비트스트림으로부터 출력 비디오 신호를 생성한다. 코덱 시스템(200)은 일반 코더 제어 컴포넌트(211), 변환 스케일링 및 양자화 컴포넌트(213), 인트라-픽처 추정 컴포넌트(215), 인트라-픽처 예측 컴포넌트(217), 모션 보상 컴포넌트(219), 모션 추정 컴포넌트(221), 스케일링 및 역변환 컴포넌트(229), 필터 제어 분석 컴포넌트(227), 인-루프 필터들 컴포넌트(225), 디코딩된 픽처 버퍼 컴포넌트(223), 및 헤더 포매팅 및 CABAC(context adaptive binary arithmetic coding) 컴포넌트(231)를 포함한다. 그러한 컴포넌트들은 도시된 바와 같이 결합된다. 도 2에서, 검은 선들은 인코딩/디코딩될 데이터의 이동을 나타내는 반면 파선들은 다른 컴포넌트들의 동작을 제어하는 제어 데이터의 이동을 나타낸다. 코덱 시스템(200)의 컴포넌트들은 모두 인코더에 존재할 수 있다. 디코더는 코덱 시스템(200)의 컴포넌트들의 서브세트를 포함할 수 있다. 예를 들어, 디코더는 인트라-픽처 예측 컴포넌트(217), 모션 보상 컴포넌트(219), 스케일링 및 역변환 컴포넌트(229), 인-루프

필터들 컴포넌트(225), 및 디코딩된 픽처 버퍼 컴포넌트(223)를 포함할 수 있다. 이제부터 이들 컴포넌트가 설명된다.

[0061] 파티셔닝된 비디오 신호(201)는 코딩 트리에 의해 픽셀 블록들로 파티셔닝된 캡처된 비디오 시퀀스이다. 코딩 트리는 다양한 분할 모드들을 이용하여 픽셀 블록을 더 작은 픽셀 블록들로 세분화한다. 그 후 이들 블록은 더 작은 블록들로 추가로 세분될 수 있다. 블록들은 코딩 트리 상의 노드들이라고 지칭될 수 있다. 더 큰 부모 노드들은 더 작은 자식 노드들로 분할된다. 노드가 세분되는 횟수는 노드/코딩 트리의 깊이라고 지칭된다. 분할된 블록들은 일부 경우들에서 코딩 유닛(CU)들에 포함될 수 있다. 예를 들어, CU는 CU에 대한 대응하는 구문 명령어들과 함께 루마 블록, 적색 차이 크로마(Cr) 블록(들), 및 청색 차이 크로마(Cb) 블록(들)을 포함하는 CTU의 서브-부분일 수 있다. 분할 모드들은 노드를 이용되는 분할 모드들에 따라 변화하는 형상들의 2개, 3개, 또는 4개의 자식 노드로 각각 파티셔닝하기 위해 이용되는 이진 트리(binary tree, BT), 트리플 트리(triple tree, TT), 및 쿼드 트리(quad tree, QT)를 포함할 수 있다. 파티셔닝된 비디오 신호(201)는 압축을 위해 일반 코더 제어 컴포넌트(211), 변환 스케일링 및 양자화 컴포넌트(213), 인트라-픽처 추정 컴포넌트(215), 필터 제어 분석 컴포넌트(227), 및 모션 추정 컴포넌트(221)로 전달된다.

[0062] 일반 코더 제어 컴포넌트(211)는 애플리케이션 제약들에 따라 비트스트림 내로의 비디오 시퀀스의 이미지들의 코딩에 관련된 결정들을 하도록 구성된다. 예를 들어, 일반 코더 제어 컴포넌트(211)는 비트레이트/비트스트림 크기 대 재구성 품질의 최적화를 관리한다. 그러한 결정들은 저장 공간/대역폭 이용가능성 및 이미지 해상도 요청들에 기초하여 행해질 수 있다. 일반 코더 제어 컴포넌트(211)는 또한 버퍼 언더런 및 오버런 문제들을 완화시키기 위해 송신 속도를 고려하여 버퍼 이용을 관리한다. 이들 문제를 관리하기 위해, 일반 코더 제어 컴포넌트(211)는 다른 컴포넌트들에 의한 파티셔닝, 예측, 및 필터링을 관리한다. 예를 들어, 일반 코더 제어 컴포넌트(211)는 동적으로 압축 복잡도를 증가시켜 해상도를 증가시키고 대역폭 사용을 증가시키거나 압축 복잡도를 감소시켜 해상도 및 대역폭 사용을 감소시킬 수 있다. 따라서, 일반 코더 제어 컴포넌트(211)는 코덱 시스템(200)의 다른 컴포넌트들을 제어하여 비디오 신호 재구성 품질과 비트 레이트 관심사들의 균형을 잡는다. 일반 코더 제어 컴포넌트(211)는, 다른 컴포넌트들의 동작을 제어하는, 제어 데이터를 생성한다. 제어 데이터는 또한 디코더에서 디코딩하기 위한 파라미터들을 시그널링하기 위해 비트스트림에 인코딩되기 위해 헤더 포매팅 및 CABAC 컴포넌트(231)로 전달된다.

[0063] 파티셔닝된 비디오 신호(201)는 또한 인트라-예측을 위해 모션 추정 컴포넌트(221) 및 모션 보상 컴포넌트(219)에 전송된다. 파티셔닝된 비디오 신호(201)의 프레임 또는 슬라이스는 다수의 비디오 블록으로 분할될 수 있다. 모션 추정 컴포넌트(221) 및 모션 보상 컴포넌트(219)는 시간 예측을 제공하기 위해 하나 이상의 참조 프레임 내의 하나 이상의 블록에 대한 수신된 비디오 블록의 인트라-예측 코딩을 수행한다. 코덱 시스템(200)은, 예를 들어, 비디오 데이터의 각각의 블록에 대해 적절한 코딩 모드를 선택하기 위해 다수의 코딩 패스를 수행할 수 있다.

[0064] 모션 추정 컴포넌트(221) 및 모션 보상 컴포넌트(219)는 고도로 통합될 수 있지만, 개념적 목적을 위해 개별적으로 예시되어 있다. 모션 추정 컴포넌트(221)에 의해 수행되는 모션 추정은, 비디오 블록들에 대한 모션을 추정하는, 모션 벡터들을 생성하는 프로세스이다. 모션 벡터는, 예를 들어, 예측 블록에 대한 코딩된 객체의 변위를 나타낼 수 있다. 예측 블록은, 픽셀 차이의 관점에서, 코딩된 블록과 가깝게 매칭하는 것으로 밝혀지는 블록이다. 예측 블록은 참조 블록이라고도 지칭될 수 있다. 그러한 픽셀 차이는 SAD(sum of absolute difference), SSD(sum of square difference), 또는 다른 차이 메트릭들에 의해 결정될 수 있다. HEVC는 CTU, 코딩 트리 블록(CTB) 들 및 CU들을 포함하는 여러 코딩된 객체들을 이용한다. 예를 들어, CTU는 CTB들로 분할될 수 있고, 이는 그 후 CU들에 포함시키기 위한 CB들로 분할될 수 있다. CU는 예측 데이터를 포함하는 예측 유닛(PU) 및/또는 CU에 대한 변환된 잔차 데이터를 포함하는 변환 유닛(TU)으로서 인코딩될 수 있다. 모션 추정 컴포넌트(221)는 레이트 왜곡 최적화 프로세스의 일부로서 레이트-왜곡 분석을 이용하여 모션 벡터들, PU들 및 TU들을 생성한다. 예를 들어, 모션 추정 컴포넌트(221)는 현재 블록/프레임에 대한 다수의 참조 블록들, 다수의 모션 벡터들 등을 결정할 수 있고, 최상의 레이트-왜곡 특성들을 갖는 참조 블록들, 모션 벡터들 등을 선택할 수 있다. 최상의 레이트-왜곡 특성들은 비디오 재구성의 품질(예를 들어, 압축에 의한 데이터 손실의 양)과 코딩 효율(예를 들어, 인코딩의 크기) 둘 다의 균형을 잡는다.

[0065] 일부 예들에서, 코덱 시스템(200)은 디코딩된 픽처 버퍼 컴포넌트(223)에 저장된 참조 픽처들의 정수 미만의(sub-integer) 픽셀 위치에 대한 값들을 계산할 수 있다. 예를 들어, 비디오 코덱 시스템(200)은 참조 픽처의 1/4 픽셀 위치들, 1/8 픽셀 위치들, 또는 다른 분수 픽셀 위치들의 값들을 보관할 수 있다. 따라서, 모션 추정 컴포넌트(221)는 전체 픽셀 위치들 및 분수 픽셀 위치들에 대해 모션 검색을 수행하고 분수 픽셀 정밀도를 갖는

모션 벡터를 출력할 수 있다. 모션 추정 컴포넌트(221)는 PU의 위치를 참조 픽처의 예측 블록의 위치와 비교함으로써 인터-코딩된 슬라이스에서 비디오 블록의 PU에 대한 모션 벡터를 계산한다. 모션 추정 컴포넌트(221)는 계산된 모션 벡터를 모션 데이터로서 인코딩을 위해 헤더 포매팅 및 CABAC 컴포넌트(231)에 그리고 모션을 모션 보상 컴포넌트(219)에 출력한다.

[0066] 모션 보상 컴포넌트(219)에 의해 수행되는 모션 보상은 모션 추정 컴포넌트(221)에 의해 결정된 모션 벡터에 기초하여 예측 블록을 페치하거나 생성하는 것을 수반할 수 있다. 다시, 일부 예들에서, 모션 추정 컴포넌트(221) 및 모션 보상 컴포넌트(219)는 기능적으로 통합될 수 있다. 현재 비디오 블록의 PU에 대한 모션 벡터를 수신하면, 모션 보상 컴포넌트(219)는 모션 벡터가 가리키는 예측 블록을 찾을 수 있다. 그 후 코딩되고 있는 현재 비디오 블록의 픽셀 값들로부터 예측 블록의 픽셀 값들을 감산하여, 픽셀 차이 값들을 형성함으로써 잔차 비디오 블록이 형성된다. 일반적으로, 모션 추정 컴포넌트(221)는 루마 성분들에 대해 모션 추정을 수행하고, 모션 보상 컴포넌트(219)는 크로마 성분들 및 루마 성분들 둘 다에 대해 루마 성분들에 기초하여 계산된 모션 벡터들을 사용한다. 예측 블록 및 잔차 블록은 변환 스케일링 및 양자화 컴포넌트(213)로 전달된다.

[0067] 파티셔닝된 비디오 신호(201)는 또한 인트라-픽처 추정 컴포넌트(215) 및 인트라-픽처 예측 컴포넌트(217)로 전송된다. 모션 추정 컴포넌트(221) 및 모션 보상 컴포넌트(219)와 마찬가지로, 인트라-픽처 추정 컴포넌트(215) 및 인트라-픽처 예측 컴포넌트(217)는 고도로 통합될 수 있지만, 개념적 목적을 위해 개별적으로 예시되어 있다. 인트라-픽처 추정 컴포넌트(215) 및 인트라-픽처 예측 컴포넌트(217)는, 위에서 설명된 바와 같이, 프레임들 간에 모션 추정 컴포넌트(221) 및 모션 보상 컴포넌트(219)에 의해 수행되는 인터-예측에 대한 대안으로서, 현재 프레임 내의 블록들에 대해 현재 블록을 인트라-예측한다. 특히, 인트라-픽처 추정 컴포넌트(215)는 현재 블록을 인코딩하기 위해 사용할 인트라-예측 모드를 결정한다. 일부 예들에서, 인트라-픽처 추정 컴포넌트(215)는 다수의 테스트된 인트라-예측 모드로부터 현재 블록을 인코딩할 적절한 인트라-예측 모드를 선택한다. 선택된 인트라-예측 모드들은 그 후 인코딩을 위해 헤더 포매팅 및 CABAC 컴포넌트(231)로 전달된다.

[0068] 예를 들어, 인트라-픽처 추정 컴포넌트(215)는 다양한 테스트된 인트라-예측 모드들에 대한 레이트-왜곡 분석을 이용하여 레이트-왜곡 값들을 계산하고, 테스트된 모드들 중에서 최상의 레이트-왜곡 특성들을 갖는 인트라-예측 모드를 선택한다. 레이트-왜곡 분석은 일반적으로 인코딩된 블록과 인코딩된 블록을 생성하기 위해 인코딩된 원래의 인코딩되지 않은 블록 사이의 왜곡(또는 에러)의 양뿐만 아니라, 인코딩된 블록을 생성하기 위해 사용되는 비트레이트(예를 들어, 비트 수)를 결정한다. 인트라-픽처 추정 컴포넌트(215)는 어느 인트라-예측 모드가 블록에 대한 최상의 레이트-왜곡 값을 나타내는지를 결정하기 위해 다양한 인코딩된 블록들에 대한 레이트들 및 왜곡들로부터의 비율들을 계산한다. 또한, 인트라-픽처 추정 컴포넌트(215)는 레이트-왜곡 최적화(RDO)에 기초하여 깊이 모델링 모드(depth modeling mode, DMM)를 이용하여 깊이 맵의 깊이 블록들을 코딩하도록 구성될 수 있다.

[0069] 인트라-픽처 예측 컴포넌트(217)는 인코더 상에서 구현될 때 인트라-픽처 추정 컴포넌트(215)에 의해 결정된 선택된 인트라-예측 모드들에 기초하여 예측 블록으로부터 잔차 블록을 생성하거나 디코더 상에서 구현될 때 비트 스트림으로부터 잔차 블록을 판독할 수 있다. 잔차 블록은, 행렬로서 표현되는, 예측 블록과 원래의 블록 사이의 값들의 차이를 포함한다. 잔차 블록은 그 후 변환 스케일링 및 양자화 컴포넌트(213)로 전달된다. 인트라-픽처 추정 컴포넌트(215) 및 인트라-픽처 예측 컴포넌트(217)는 루마 및 크로마 성분들 둘 다에 대해 동작할 수 있다.

[0070] 변환 스케일링 및 양자화 컴포넌트(213)는 잔차 블록을 추가로 압축하도록 구성된다. 변환 스케일링 및 양자화 컴포넌트(213)는 DCT(discrete cosine transform), DST(discrete sine transform), 또는 개념적으로 유사한 변환과 같은 변환을 잔차 블록에 적용하여, 잔차 변환 계수 값들을 포함하는 비디오 블록을 생성한다. 웨이블릿 변환들, 정수 변환들, 부대역 변환들 또는 다른 타입의 변환들이 또한 사용될 수 있다. 변환은 잔차 정보를 픽셀 값 도메인으로부터 주파수 도메인과 같은 변환 도메인으로 변환할 수 있다. 변환 스케일링 및 양자화 컴포넌트(213)는 또한, 예를 들어, 주파수에 기초하여 변환된 잔차 정보를 스케일링하도록 구성된다. 그러한 스케일링은 상이한 주파수 정보가 상이한 입도들로 양자화되도록 스케일 인자를 잔차 정보에 적용하는 것을 수반하고, 이는 재구성된 비디오의 최종 시각적 품질에 영향을 미칠 수 있다. 변환 스케일링 및 양자화 컴포넌트(213)는 또한 비트 레이트를 추가로 감소시키기 위해 변환 계수들을 양자화하도록 구성된다. 양자화 프로세스는 계수들의 일부 또는 전부와 연관된 비트 깊이를 감소시킬 수 있다. 양자화의 정도는 양자화 파라미터를 조정함으로써 수정될 수 있다. 일부 예들에서, 변환 스케일링 및 양자화 컴포넌트(213)는 그 후 양자화된 변환 계수들을 포함하는 행렬의 스캔을 수행할 수 있다. 양자화된 변환 계수들은 비트스트림에 인코딩되기 위해 해

더 포매팅 및 CABAC 컴포넌트(231)로 전달된다.

- [0071] 스케일링 및 역변환 컴포넌트(229)는 모션 추정을 지원하기 위해 변환 스케일링 및 양자화 컴포넌트(213)의 역 동작을 적용한다. 스케일링 및 역변환 컴포넌트(229)는, 예를 들어, 다른 현재 블록에 대한 예측 블록이 될 수 있는 참조 블록으로서 나중의 사용을 위해, 픽셀 도메인에서의 잔차 블록을 재구성하기 위해 역 스케일링, 변환, 및/또는 양자화를 적용한다. 모션 추정 컴포넌트(221) 및/또는 모션 보상 컴포넌트(219)는 나중의 블록/프레임의 모션 추정에서의 사용을 위해 대응하는 예측 블록에 잔차 블록을 다시 추가함으로써 참조 블록을 계산할 수 있다. 스케일링, 양자화, 및 변환 동안 생성된 아티팩트들을 완화시키기 위해 재구성된 참조 블록들에 필터들이 적용된다. 그러한 아티팩트들은 후속 블록들이 예측될 때 부정확한 예측을 야기할 수 있다(그리고 추가적인 아티팩트들을 생성할 수 있다).
- [0072] 필터 제어 분석 컴포넌트(227) 및 인-루프 필터들 컴포넌트(225)는 필터들을 잔차 블록들 및/또는 재구성된 이미지 블록들에 적용한다. 예를 들어, 스케일링 및 역변환 컴포넌트(229)로부터의 변환된 잔차 블록은 인트라-픽처 예측 컴포넌트(217) 및/또는 모션 보상 컴포넌트(219)로부터의 대응하는 예측 블록과 조합되어 원래의 이미지 블록을 재구성할 수 있다. 필터들은 그 후 재구성된 이미지 블록에 적용될 수 있다. 일부 예들에서, 필터들은 대신에 잔차 블록들에 적용될 수 있다. 도 2의 다른 컴포넌트들과 마찬가지로, 필터 제어 분석 컴포넌트(227) 및 인-루프 필터들 컴포넌트(225)는 고도로 통합되고 함께 구현될 수 있지만, 개념적 목적을 위해 개별적으로 묘사되어 있다. 재구성된 참조 블록들에 적용되는 필터들은 특정한 공간 영역들에 적용되고, 그러한 필터들이 어떻게 적용되는지를 조정하기 위한 다수의 파라미터를 포함한다. 필터 제어 분석 컴포넌트(227)는 재구성된 참조 블록들을 분석하여 그러한 필터들이 어디에 적용되어야 하는지를 결정하고 대응하는 파라미터들을 설정한다. 그러한 데이터는 인코딩을 위한 필터 제어 데이터로서 헤더 포매팅 및 CABAC 컴포넌트(231)로 전달된다. 인-루프 필터들 컴포넌트(225)는 필터 제어 데이터에 기초하여 그러한 필터들을 적용한다. 필터들은 더블 로킹 필터, 잡음 억제 필터, SAO 필터, 및 적응 루프 필터를 포함할 수 있다. 그러한 필터들은, 예에 따라, 공간/픽셀 도메인에서(예를 들어, 재구성된 픽셀 블록에 대해) 또는 주파수 도메인에서 적용될 수 있다.
- [0073] 인코더로서 동작할 때, 필터링된 재구성된 이미지 블록, 잔차 블록, 및/또는 예측 블록은 위에서 논의된 바와 같이 모션 추정에서 나중의 사용을 위해 디코딩된 픽처 버퍼 컴포넌트(223)에 저장된다. 디코더로서 동작할 때, 디코딩된 픽처 버퍼 컴포넌트(223)는 재구성되고 필터링된 블록들을 저장하고 출력 비디오 신호의 일부로서 디스플레이를 향해 전달한다. 디코딩된 픽처 버퍼 컴포넌트(223)는 예측 블록들, 잔차 블록들, 및/또는 재구성된 이미지 블록을 저장할 수 있는 임의의 메모리 디바이스일 수 있다.
- [0074] 헤더 포매팅 및 CABAC 컴포넌트(231)는 코덱 시스템(200)의 다양한 컴포넌트들로부터 데이터를 수신하고 그러한 데이터를 디코더를 향한 송신을 위한 코딩된 비트스트림 내로 인코딩한다. 구체적으로, 헤더 포매팅 및 CABAC 컴포넌트(231)는 일반 제어 데이터 및 필터 제어 데이터와 같은 제어 데이터를 인코딩하기 위한 다양한 헤더들을 생성한다. 또한, 인트라-예측 및 모션 데이터를 포함하는 예측 데이터뿐만 아니라, 양자화된 변환 계수 데이터의 형식의 잔차 데이터가 모두 비트스트림에 인코딩된다. 최종 비트스트림은 원래의 파티셔닝된 비디오 신호(201)를 재구성하기 위해 디코더에 의해 요구되는 모든 정보를 포함한다. 그러한 정보는 인트라-예측 모드 인덱스 테이블들(코드워드 맵핑 테이블들이라고도 지칭됨), 다양한 블록들에 대한 인코딩 컨텍스트들의 정의들, 가장 가능성 있는 인트라-예측 모드들의 지시들, 파티션 정보의 지시 등을 또한 포함할 수 있다. 그러한 데이터는 엔트로피 코딩을 이용하여 인코딩될 수 있다. 예를 들어, 정보는 CAVLC(context adaptive variable length coding), CABAC, SBAC(syntax-based context-adaptive binary arithmetic coding), PIPE(probability interval partitioning entropy) 코딩, 또는 다른 엔트로피 코딩 기법을 이용하여 인코딩될 수 있다. 엔트로피 코딩에 이어서, 코딩된 비트스트림은 다른 디바이스(예를 들어, 비디오 디코더)로 송신되거나 나중의 송신 또는 검색을 위해 보관될 수 있다.
- [0075] 도 3은 예시적인 비디오 인코더(300)를 예시하는 블록도이다. 비디오 인코더(300)는 코덱 시스템(200)의 인코딩 기능들을 구현하고/하거나 동작 방법(100)의 단계들 101, 103, 105, 107, 및/또는 109를 구현하기 위해 이용될 수 있다. 인코더(300)는 입력 비디오 신호를 파티셔닝하여, 파티셔닝된 비디오 신호(301)를 야기하고, 이는 파티셔닝된 비디오 신호(201)와 실질적으로 유사하다. 파티셔닝된 비디오 신호(301)는 그 후 압축되고 인코더(300)의 컴포넌트들에 의해 비트스트림 내로 인코딩된다.
- [0076] 구체적으로, 파티셔닝된 비디오 신호(301)는 인트라-예측을 위해 인트라-픽처 예측 컴포넌트(317)로 전달된다. 인트라-픽처 예측 컴포넌트(317)는 인트라-픽처 추정 컴포넌트(215) 및 인트라-픽처 예측 컴포넌트(217)와 실질적으로 유사할 수 있다. 파티셔닝된 비디오 신호(301)는 또한 디코딩된 픽처 버퍼 컴포넌트(323) 내의 참조 블

록들에 기초하여 인터-예측을 위해 모션 보상 컴포넌트(321)로 전달된다. 모션 보상 컴포넌트(321)는 모션 추정 컴포넌트(221) 및 모션 보상 컴포넌트(219)와 실질적으로 유사할 수 있다. 인트라-픽처 예측 컴포넌트(317) 및 모션 보상 컴포넌트(321)로부터의 예측 블록들 및 잔차 블록들은 잔차 블록들의 변환 및 양자화를 위해 변환 및 양자화 컴포넌트(313)로 전달된다. 변환 및 양자화 컴포넌트(313)는 변환 스케일링 및 양자화 컴포넌트(213)와 실질적으로 유사할 수 있다. 변환되고 양자화된 잔차 블록들 및 대응하는 예측 블록들은 (연관된 제어 데이터와 함께) 비트스트림 내로 코딩을 위해 엔트로피 코딩 컴포넌트(331)로 전달된다. 엔트로피 코딩 컴포넌트(331)는 헤더 포매팅 및 CABAC 컴포넌트(231)와 실질적으로 유사할 수 있다.

[0077] 변환되고 양자화된 잔차 블록들 및/또는 대응하는 예측 블록들은 또한 모션 보상 컴포넌트(321)에 의한 사용을 위한 참조 블록들로의 재구성을 위해 변환 및 양자화 컴포넌트(313)로부터 역변환 및 양자화 컴포넌트(329)로 전달된다. 역변환 및 양자화 컴포넌트(329)는 스케일링 및 역변환 컴포넌트(229)와 실질적으로 유사할 수 있다. 인-루프 필터들 컴포넌트(325) 내의 인-루프 필터들은 또한, 예에 따라, 잔차 블록들 및/또는 재구성된 참조 블록들에 적용된다. 인-루프 필터들 컴포넌트(325)는 필터 제어 분석 컴포넌트(227) 및 인-루프 필터들 컴포넌트(225)와 실질적으로 유사할 수 있다. 인-루프 필터들 컴포넌트(325)는 인-루프 필터들 컴포넌트(225)에 관하여 논의된 바와 같은 다수의 필터를 포함할 수 있다. 필터링된 블록들은 그 후 모션 보상 컴포넌트(321)에 의한 참조 블록들로서 사용을 위해 디코딩된 픽처 버퍼 컴포넌트(323)에 저장된다. 디코딩된 픽처 버퍼 컴포넌트(323)는 디코딩된 픽처 버퍼 컴포넌트(223)와 실질적으로 유사할 수 있다.

[0078] 도 4는 예시적인 비디오 디코더(400)를 예시하는 블록도이다. 비디오 디코더(400)는 코덱 시스템(200)의 디코딩 기능들을 구현하고/하거나 동작 방법(100)의 단계들 111, 113, 115, 및/또는 117을 구현하기 위해 이용될 수 있다. 디코더(400)는 예를 들어 인코더(300)로부터 비트스트림을 수신하고, 최종 사용자에게 디스플레이하기 위한 비트스트림에 기초하여 재구성된 출력 비디오 신호를 생성한다.

[0079] 비트스트림은 엔트로피 디코딩 컴포넌트(433)에 의해 수신된다. 엔트로피 디코딩 컴포넌트(433)는 CAVLC, CABAC, SBAC, PIPE 코딩, 또는 다른 엔트로피 코딩 기법들과 같은 엔트로피 디코딩 스킴을 구현하도록 구성된다. 예를 들어, 엔트로피 디코딩 컴포넌트(433)는 헤더 정보를 이용하여 비트스트림 내에 코드워드들로서 인코딩된 추가적인 데이터를 해석하기 위한 컨텍스트를 제공할 수 있다. 디코딩된 정보는 잔차 블록들로부터의 일반 제어 데이터, 필터 제어 데이터, 파티션 정보, 모션 데이터, 예측 데이터, 및 양자화된 변환 계수들과 같은 비디오 신호를 디코딩하기 위한 임의의 원하는 정보를 포함한다. 양자화된 변환 계수들은 잔차 블록들로의 재구성을 위해 역변환 및 양자화 컴포넌트(429)로 전달된다. 역변환 및 양자화 컴포넌트(429)는 역변환 및 양자화 컴포넌트(329)와 유사할 수 있다.

[0080] 재구성된 잔차 블록들 및/또는 예측 블록들은 인트라-예측 동작들에 기초하여 이미지 블록들로의 재구성을 위해 인트라-픽처 예측 컴포넌트(417)로 전달된다. 인트라-픽처 예측 컴포넌트(417)는 인트라-픽처 추정 컴포넌트(215) 및 인트라-픽처 예측 컴포넌트(217)와 유사할 수 있다. 구체적으로, 인트라-픽처 예측 컴포넌트(417)는 예측 모드들을 이용하여 프레임 내의 참조 블록을 찾고 잔차 블록을 결과에 적용하여 인트라-예측된 이미지 블록을 재구성한다. 재구성된 인트라-예측된 이미지 블록들 및/또는 잔차 블록들 및 대응하는 인터-예측 데이터는 인-루프 필터들 컴포넌트(425)를 통해 디코딩된 픽처 버퍼 컴포넌트(423)에 전달되는데, 이들은 각각, 인-루프 필터들 컴포넌트(225) 및 디코딩된 픽처 버퍼 컴포넌트(223)와 실질적으로 유사할 수 있다. 인-루프 필터들 컴포넌트(425)는 재구성된 이미지 블록들, 잔차 블록들 및/또는 예측 블록들을 필터링하고, 그러한 정보는 디코딩된 픽처 버퍼 컴포넌트(423)에 저장된다. 디코딩된 픽처 버퍼 컴포넌트(423)로부터의 재구성된 이미지 블록들은 인터-예측을 위해 모션 보상 컴포넌트(421)로 전달된다. 모션 보상 컴포넌트(421)는 모션 추정 컴포넌트(221) 및/또는 모션 보상 컴포넌트(219)와 실질적으로 유사할 수 있다. 구체적으로, 모션 보상 컴포넌트(421)는 참조 블록으로부터 모션 벡터들을 이용하여 예측 블록을 생성하고 잔차 블록을 결과에 적용하여 이미지 블록을 재구성한다. 결과적인 재구성된 블록은 또한 인-루프 필터들 컴포넌트(425)를 통해 디코딩된 픽처 버퍼 컴포넌트(423)로 전달될 수 있다. 디코딩된 픽처 버퍼 컴포넌트(423)는 계속해서 추가적인 재구성된 이미지 블록들을 저장하고, 이들은 파티션 정보를 통해 프레임들로 재구성될 수 있다. 그러한 프레임들은 또한 시퀀스에 배치될 수 있다. 시퀀스는 재구성된 출력 비디오 신호로서 디스플레이를 향해 출력된다.

[0081] 도 5a는 서브-픽처들(510)로 파티셔닝된 예시적인 픽처(500)를 예시하는 개략도이다. 예를 들어, 픽처(500)는 코덱 시스템(200) 및/또는 인코더(300)에 의한 인코딩을 위해 파티셔닝될 수 있고, 코덱 시스템(200) 및/또는 디코더(400)에 의한 디코딩을 위해 파티셔닝될 수 있다. 다른 예로서, 픽처(500)는 단계 111에서 디코더에 의한 사용을 위해 방법(100)의 단계 103에서 인코더에 의해 파티셔닝될 수 있다.

- [0082] 픽처(500)는 특정된 시간 위치에서 비디오 시퀀스의 완전한 시각적 부분을 묘사하는 이미지이다. 픽처(500)는 또한 이미지 및/또는 프레임으로 지칭될 수 있다. 픽처(500)는 POC(picture order count)에 의해 특정될 수 있다. POC는 비디오 시퀀스에서 픽처(500)의 출력/디스플레이 순서를 나타내는 인덱스이다. 픽처(500)는 서브-픽처들(510)로 파티셔닝될 수 있다. 서브-픽처(510)는 픽처(500) 내의 하나 이상의 슬라이스/타일 그룹의 직사각형 또는 정사각형 영역이다. 서브-픽처들(510)은 선택적이며, 따라서 일부 비디오 시퀀스들은 서브-픽처들(510)을 포함하는 반면, 다른 것들은 그렇지 않다. 4개의 서브-픽처(510)가 도시되어 있지만, 픽처(500)는 임의의 수의 서브-픽처들(510)로 파티셔닝될 수 있다. 서브-픽처(510)의 파티셔닝은 전체 CVS(coded video sequence)에 걸쳐 일관될 수 있다.
- [0083] 서브-픽처들(510)은 픽처(500)의 상이한 영역들이 상이하게 취급되게 하기 위해 이용될 수 있다. 예를 들어, 특정된 서브-픽처(510)는 독립적으로 추출되어 디코더로 송신될 수 있다. 구체적인 예로서, 가상 현실(VR) 헤드셋을 이용하는 사용자는 픽처(500)의 서브-셋트를 볼 수 있고, 이는 픽처(500)에 묘사된 바와 같은 공간에 물리적으로 존재한다는 인상을 사용자에게 제공할 수 있다. 이러한 경우에, 사용자에게 디스플레이될 가능성이 있는 서브-픽처들(510)만을 스트리밍하는 것은 코딩 효율을 증가시킬 수 있다. 다른 예로서, 상이한 서브-픽처들(510)은 특정 애플리케이션들에서 상이하게 취급될 수 있다. 구체적인 예로서, 원격 회의 애플리케이션은 현재 말하고 있지 않은 사용자들보다 더 두드러진 위치에서 더 높은 해상도로 액티브 스피커(active speaker)를 디스플레이할 수 있다. 상이한 서브-픽처들(510) 내에 상이한 사용자들을 배치하는 것은 이 기능성을 지원하기 위해 디스플레이된 이미지의 실시간 재구성을 지원한다.
- [0084] 각각의 서브-픽처(510)는 전체 CVS에 대해 일관될 수 있는 고유 서브-픽처 ID에 의해 식별될 수 있다. 예를 들어, 픽처(500)의 좌측 상단에 있는 서브-픽처(510)는 0의 서브-픽처 ID를 가질 수 있다. 그러한 경우에, 시퀀스 내의 임의의 픽처(500)의 좌측 상단 서브-픽처(510)는 0의 서브-픽처 ID에 의해 참조될 수 있다. 또한, 각각의 서브-픽처(510)는 전체 CVS에 대해 일관될 수 있는 정의된 구성을 포함할 수 있다. 예를 들어, 서브-픽처(510)는 높이, 폭 및/또는 오프셋을 포함할 수 있다. 높이 및 폭은 서브-픽처(510)의 크기를 기술하고 오프셋은 서브-픽처(510)의 위치를 기술한다. 예를 들어, 행 내의 모든 서브-픽처들(510)의 폭들의 합은 픽처(500)의 폭이다. 또한, 열 내의 모든 서브-픽처들(510)의 높이들의 합은 픽처(500)의 높이이다. 또한, 오프셋은 픽처(500)의 좌측 상단 코너에 대한 서브-픽처(510)의 좌측 상단 코너의 위치를 나타낸다. 서브-픽처(510)의 높이, 폭 및 오프셋은 픽처(500)에 대응하는 서브-픽처(510)를 배치하기에 충분한 정보를 제공한다. 서브-픽처들(510)의 파티셔닝은 전체 CVS에 걸쳐 일관될 수 있기 때문에, 서브-픽처들에 관련된 파라미터들은 시퀀스 파라미터 세트(PPS)에 포함될 수 있다.
- [0085] 도 5b는 슬라이스들(515)로 파티셔닝된 예시적인 서브-픽처(510)를 예시하는 개략도이다. 도시된 바와 같이, 픽처(500)의 서브-픽처(510)는 하나 이상의 슬라이스(515)를 포함할 수 있다. 슬라이스(515)는 단일의 네트워크 추상화 계층(NAL) 유닛에 배타적으로 포함되는 픽처의 타일 내의 정수 개의 완전한 타일들 또는 정수 개의 연속적인 완전한 CTU 행들이다. 4개의 슬라이스(515)가 도시되어 있지만, 서브-픽처(510)는 임의의 수의 슬라이스(515)를 포함할 수 있다. 슬라이스들(515)은 특정된 POC의 픽처(500)에 특정한 시각적 데이터를 포함한다. 따라서, 슬라이스들(515)에 관련된 파라미터들은 픽처 파라미터 세트(PPS) 및/또는 슬라이스 헤더에 포함될 수 있다.
- [0086] 도 5c는 타일들(517)로 파티셔닝된 예시적인 슬라이스(515)를 예시하는 개략도이다. 도시된 바와 같이, 픽처(500)의 슬라이스(515)는 하나 이상의 타일(517)을 포함할 수 있다. 타일들(517)은 픽처(500)를 직사각형들의 행들 및 열들로 파티셔닝함으로써 생성될 수 있다. 따라서, 타일(517)은 픽처 내의 특정 타일 열 및 특정 타일 행 내의 CTU들의 직사각형 또는 정사각형 영역이다. 타일링은 선택적이며, 따라서 일부 비디오 시퀀스들은 타일들(517)을 포함하는 반면, 다른 것들은 그렇지 않다. 4개의 타일(517)이 도시되어 있지만, 슬라이스(515)는 임의의 수의 타일(517)을 포함할 수 있다. 타일들(517)은 특정된 POC의 픽처(500)의 슬라이스(515)에 특정한 시각적 데이터를 포함할 수 있다. 일부 경우들에서, 슬라이스들(515)은 또한 타일들(517)에 포함될 수 있다. 따라서, 타일들(517)과 관련된 파라미터들은 PPS 및/또는 슬라이스 헤더에 포함될 수 있다.
- [0087] 도 5d는 CTU들(519)로 파티셔닝된 예시적인 슬라이스(515)를 예시하는 개략도이다. 도시된 바와 같이, 픽처(500)의 슬라이스(515)(또는 슬라이스(515)의 타일(517))는 하나 이상의 CTU(519)를 포함할 수 있다. CTU(519)는 인코딩/디코딩되는 코딩 블록들을 생성하기 위해 코딩 트리에 의해 세분되는 픽처(500)의 영역이다. CTU(519)는 단색 픽처(500)에 대한 루마 샘플들 또는 컬러 픽처(500)에 대한 루마 및 크로마 샘플들의 조합을 포함할 수 있다. 코딩 트리에 의해 파티셔닝될 수 있는 루마 샘플들 또는 크로마 샘플들의 그룹화는 코딩 트리 블록(CTB)(518)으로 지칭된다. 이와 같이, CTU(519)는 3개의 샘플 어레이를 갖는 픽처(500)의 루마 샘플들의

CTB(518) 및 크로마 샘플들의 2개의 대응하는 CTB(518)를 포함하거나, 또는 샘플들을 코딩하기 위해 사용되는 3개의 개별 컬러 평면 및 구문 구조를 사용하여 코딩되는 픽처 또는 단색 픽처의 샘플들의 CTB(518)를 포함한다.

[0088] 전술한 바와 같이, 픽처(500)는 서브-픽처들(510), 슬라이스들(515), 타일들(517), CTU들(519), 및/또는 CTB들(518)로 파티셔닝될 수 있으며, 이들은 그 후 블록들로 파티셔닝된다. 그 후, 그런 블록들은 디코더를 향한 송신을 위해 인코딩된다. 이러한 블록들을 디코딩하는 것은 다양한 타입들의 잡음을 포함하는 디코딩된 이미지를 야기할 수 있다. 이러한 문제들을 정정하기 위해, 비디오 코딩 시스템들은 블록 경계들에 걸쳐 다양한 필터들을 적용할 수 있다. 이러한 필터들은 블로킹, 양자화 잡음, 및 다른 바람직하지 않은 코딩 아티팩트들을 제거할 수 있다. 전술한 바와 같이, 서브-픽처들(510)은 독립적인 추출을 수행할 때 이용될 수 있다. 이 경우, 현재 서브-픽처(510)는 다른 서브-픽처들(510)로부터의 정보를 디코딩하지 않고 디코딩 및 디스플레이될 수 있다. 이와 같이, 서브-픽처(510) 에지를 따르는 블록 경계들은 서브-픽처 경계들과 정렬될 수 있다. 일부 경우들에서, 블록 경계들은 또한 타일 경계들과 정렬될 수 있다. 필터들은 그러한 블록 경계들에 걸쳐 적용될 수 있으며, 따라서 서브-픽처 경계들 및/또는 타일 경계들에 걸쳐 적용될 수 있다. 이것은 현재 서브-픽처(510)가 독립적으로 추출될 때 에러를 유발할 수 있는데, 그 이유는 인접한 서브-픽처(510)로부터의 데이터가 이용 가능하지 않을 때 필터링 프로세스가 예상치 못한 방식으로 동작할 수 있기 때문이다.

[0089] 이들 문제를 다루기 위해, 서브-픽처(510) 레벨에서 필터링을 제어하는 플래그가 이용될 수 있다. 예를 들어, 플래그는 `loop_filter_across_subpic_enabled_flag`로서 표시될 수 있다. 플래그가 서브-픽처(510)에 대해 설정될 때, 필터들은 대응하는 서브-픽처 경계에 걸쳐 적용될 수 있다. 플래그가 설정되지 않을 때, 필터들은 대응하는 서브-픽처 경계에 걸쳐 적용되지 않는다. 이러한 방식으로, 필터들은 개별 추출을 위해 인코딩되는 서브-픽처들(510)에 대해 턴오프될 수 있거나 또는 그룹으로서 디스플레이를 위해 인코딩되는 서브-픽처들(510)에 대해 턴온될 수 있다. 타일(517) 레벨에서 필터링을 제어하기 위해 다른 플래그가 설정될 수 있다. 플래그는 `loop_filter_across_tiles_enabled_flag`로서 표시될 수 있다. 플래그가 타일(517)에 대해 설정될 때, 필터들이 타일 경계에 걸쳐 적용될 수 있다. 플래그가 설정되지 않을 때, 필터들은 타일 경계에 걸쳐 적용되지 않는다. 이러한 방식으로, 필터들은 타일 경계들에서 사용하기 위해(예를 들어, 타일의 내부 부분들을 계속 필터링하면서) 턴 오프 또는 온될 수 있다.

[0090] 또한, 전술한 바와 같이, 타일링은 선택적이다. 그러나, 일부 비디오 코딩 시스템들은 서브-픽처(510)에 포함된 타일들(517)의 관점에서 서브-픽처 경계들을 기술한다. 그러한 시스템들에서, 타일들(517)의 관점에서의 서브-픽처 경계 설명들은 서브-픽처들(510)의 사용을 타일들(517)을 이용하는 픽처들(500)로 제한한다. 서브-픽처들(510)의 적용 가능성을 넓히기 위해, 서브-픽처들(510)은 경계들의 관점에서, CTB들(518) 및/또는 CTU들(519)의 관점에서 설명될 수 있다. 구체적으로, 서브-픽처(510)의 폭 및 높이는 CTB들(518)의 단위로 시그널링될 수 있다. 또한, 서브-픽처(510)의 좌측 상단 CTU(519)의 위치는 CTB들(518)에서 측정된 픽처(500)의 좌측 상단 CTU(519)로부터의 오프셋으로서 시그널링될 수 있다. CTU(519) 및 CTB(518) 크기는 미리 결정된 값으로 설정될 수 있다. 따라서, CTB들(518) 및 CTU들(519)의 관점에서 서브-픽처 치수들 및 위치를 시그널링하는 것은 디코더가 디스플레이를 위해 서브-픽처(510)를 배치하기에 충분한 정보를 제공한다. 이것은 타일들(517)이 이용되지 않을 때에도 서브-픽처들(510)이 이용되게 한다. 본 명세서에서 사용되는 바와 같이, 필터는 필터가 경계의 양 측 상의 샘플들에 적용될 때 서브-픽처(510) 또는 타일(517) 경계에 걸쳐 적용된다.

[0091] 또한, 일부 비디오 코딩 시스템들은 픽처(500)에 대한 그들의 위치에 기초하여 슬라이스들(515)을 어드레싱한다. 이것은 서브-픽처들(510)이 독립적인 추출 및 디스플레이를 위해 코딩될 때 문제를 생성한다. 이러한 경우에, 생략된 서브-픽처들(510)과 연관된 슬라이스들(515) 및 대응하는 어드레스들도 생략된다. 슬라이스들(515)의 어드레스들의 생략은 디코더가 슬라이스들(515)을 적절히 배치하는 것을 방지할 수 있다. 일부 비디오 코딩 시스템들은 슬라이스(515)와 연관된 슬라이스 헤더들 내의 어드레스를 동적으로 재기입함으로써 이 문제를 다룬다. 사용자가 임의의 서브-픽처를 요청할 수 있기 때문에, 그러한 재기입은 사용자가 비디오를 요청할 때마다 발생하며, 이는 극도로 자원 집약적이다. 이 문제를 극복하기 위해, 슬라이스들(515)은 서브-픽처들(510)이 이용될 때 슬라이스(515)를 포함하는 서브-픽처(510)에 대해 어드레싱된다. 예를 들어, 슬라이스(515)는 슬라이스(515)를 포함하는 서브-픽처(510)에 특정한 인덱스 또는 다른 값에 의해 식별될 수 있다. 슬라이스 어드레스는 슬라이스(515)와 연관된 슬라이스 헤더로 코딩될 수 있다. 슬라이스(515)를 포함하는 서브-픽처(510)의 서브-픽처 ID도 슬라이스 헤더에 인코딩될 수 있다. 또한, 서브-픽처(510)의 치수들/구성은 서브-픽처 ID와 함께 SPS로 코딩될 수 있다. 이와 같이, 디코더는 서브-픽처 ID에 기초하여 SPS로부터 서브-픽처(510) 구성을 획득하고, 전체 픽처(500)를 참조하지 않고 슬라이스(515)를 서브-픽처(510) 내에 배치할 수 있다. 이와 같이, 슬라이스 헤더 재기입은 서브-픽처(510)가 추출될 때 생략될 수 있으며, 이는 인코더, 디코

더 및/또는 대응하는 슬라이서에서의 자원 사용을 상당히 감소시킨다.

- [0092] 픽처(500)가 CTB들(518) 및/또는 CTU들(519)로 파티셔닝되면, CTB들(518) 및/또는 CTU들(519)은 코딩 블록들로 더 분할될 수 있다. 코딩 블록들은 그 후 인트라-예측 및/또는 인터-예측에 따라 코딩될 수 있다. 본 개시내용은 또한 인터-예측 메커니즘들에 관련된 개선들을 포함한다. 인터-예측은 단방향 인터-예측 및/또는 양방향 인터-예측에 따라 동작할 수 있는 여러 상이한 모드들에서 수행될 수 있다.
- [0093] 도 6은 예를 들어, 블록 압축 단계 105, 블록 디코딩 단계 113, 모션 추정 컴포넌트(221), 모션 보상 컴포넌트(219), 모션 보상 컴포넌트(321), 및/또는 모션 보상 컴포넌트(421)에서 모션 벡터들(MV들)을 결정하기 위해 수행되는 바와 같은 단방향 인터-예측(600)의 예를 예시하는 개략도이다. 예를 들어, 단방향 인터-예측(600)은 픽처(500)와 같은 픽처를 파티셔닝할 때 생성되는 인코딩된 및/또는 디코딩 블록들에 대한 모션 벡터들을 결정하기 위해 이용될 수 있다.
- [0094] 단방향 인터-예측(600)은 현재 프레임(610) 내의 현재 블록(611)을 예측하기 위해 참조 블록(631)을 갖는 참조 프레임(630)을 이용한다. 참조 프레임(630)은 (예를 들어, 후속 참조 프레임으로서) 도시된 바와 같이 현재 프레임(610) 이후에 시간적으로 배치될 수 있지만, 일부 예들에서는 (예를 들어, 선행 참조 프레임으로서) 현재 프레임(610) 이전에 시간적으로 배치될 수도 있다. 현재 프레임(610)은 특정 시간에 인코딩/디코딩되는 예시적인 프레임/픽처이다. 현재 프레임(610)은 참조 프레임(630)의 참조 블록(631) 내의 객체와 매칭하는 현재 블록(611) 내의 객체를 포함한다. 참조 프레임(630)은 현재 프레임(610)을 인코딩하기 위한 참조로서 이용되는 프레임이고, 참조 블록(631)은 현재 프레임(610)의 현재 블록(611)에 또한 포함된 객체를 포함하는 참조 프레임(630) 내의 블록이다.
- [0095] 현재 블록(611)은 코딩 프로세스에서 특정된 포인트에서 인코딩/디코딩되고 있는 임의의 코딩 유닛이다. 현재 블록(611)은 전체 파티셔닝된 블록일 수 있거나, 또는 아핀 인터-예측 모드를 이용할 때의 서브-블록일 수 있다. 현재 프레임(610)은 일부 시간 거리(TD)(633)만큼 참조 프레임(630)으로부터 분리된다. TD(633)는 비디오 시퀀스에서의 현재 프레임(610)과 참조 프레임(630) 사이의 시간량을 나타내고, 프레임들의 단위로 측정될 수 있다. 현재 블록(611)에 대한 예측 정보는 프레임들 사이의 방향 및 시간 거리를 나타내는 참조 인덱스에 의해 참조 프레임(630) 및/또는 참조 블록(631)을 참조할 수 있다. TD(633)에 의해 표현되는 기간에 걸쳐, 현재 블록(611) 내의 객체는 현재 프레임(610)에서의 위치로부터 참조 프레임(630)에서의 다른 위치(예를 들어, 참조 블록(631)의 위치)로 이동한다. 예를 들어, 객체는 시간에 따른 객체의 이동 방향인 모션 궤적(613)을 따라 이동할 수 있다. 모션 벡터(635)는 TD(633)에 걸친 모션 궤적(613)을 따른 객체의 이동의 방향 및 크기를 설명한다. 따라서, 인코딩된 모션 벡터(635), 참조 블록(631), 및 현재 블록(611)과 참조 블록(631) 사이의 차이를 포함하는 잔차는 현재 블록(611)을 재구성하고 현재 프레임(610)에 현재 블록(611)을 위치시키기에 충분한 정보를 제공한다.
- [0096] 도 7은 예를 들어, 블록 압축 단계 105, 블록 디코딩 단계 113, 모션 추정 컴포넌트(221), 모션 보상 컴포넌트(219), 모션 보상 컴포넌트(321) 및/또는 모션 보상 컴포넌트(421)에서 MV들을 결정하기 위해 수행되는 양방향 인터-예측(700)의 예를 예시하는 개략도이다. 예를 들어, 양방향 인터-예측(700)은 픽처(500)와 같은 픽처를 파티셔닝할 때 생성되는 인코딩된 및/또는 디코딩 블록들에 대한 모션 벡터들을 결정하기 위해 이용될 수 있다.
- [0097] 양방향 인터-예측(700)은 단방향 인터-예측(600)과 유사하지만, 현재 프레임(710) 내의 현재 블록(711)을 예측하기 위해 한 쌍의 참조 프레임을 이용한다. 따라서, 현재 프레임(710) 및 현재 블록(711)은 각각 현재 프레임(610) 및 현재 블록(611)과 실질적으로 유사하다. 현재 프레임(710)은 비디오 시퀀스에서 현재 프레임(710) 이전에 발생하는 선행 참조 프레임(720)과 비디오 시퀀스에서 현재 프레임(710) 이후에 발생하는 후속 참조 프레임(730) 사이에 시간적으로 위치된다. 선행 참조 프레임(720) 및 후속 참조 프레임(730)은 그 외에는 참조 프레임(630)과 실질적으로 유사하다.
- [0098] 현재 블록(711)은 선행 참조 프레임(720) 내의 선행 참조 블록(721) 및 후속 참조 프레임(730) 내의 후속 참조 블록(731)에 매칭된다. 이러한 매칭은, 비디오 시퀀스의 코스에 걸쳐, 객체가 모션 궤적(713)을 따라 그리고 현재 블록(711)을 통해 이전 참조 블록(721)에서의 위치로부터 후속 참조 블록(731)에서의 위치로 이동한다는 것을 나타낸다. 현재 프레임(710)은 일부 선행 시간 거리(TD0)(723)만큼 선행 참조 프레임(720)으로부터 분리되고, 일부 후속 시간 거리(TD1)(733)만큼 후속 참조 프레임(730)으로부터 분리된다. TD0(723)은 비디오 시퀀스에서의 선행 참조 프레임(720)과 현재 프레임(710) 사이의 시간량을 프레임들의 단위로 나타낸다. TD1(733)은 비디오 시퀀스에서의 현재 프레임(710)과 후속 참조 프레임(730) 사이의 시간량을 프레임의 단위로 나타낸다. 따라서, 객체는 TD0(723)에 의해 나타난 기간에 걸쳐 모션 궤적(713)을 따라 이전 참조 블록(721)으

로부터 현재 블록(711)으로 이동한다. 객체는 또한 TD1(733)에 의해 나타난 기간에 걸쳐 모션 궤적(713)을 따라 현재 블록(711)으로부터 후속 참조 블록(731)으로 이동한다. 현재 블록(711)에 대한 예측 정보는 선행 참조 프레임(720) 및/또는 선행 참조 블록(721) 및 후속 참조 프레임(730) 및/또는 후속 참조 블록(731)을 프레임들 사이의 방향 및 시간 거리를 나타내는 한 쌍의 참조 인덱스에 의해 참조할 수 있다.

[0099] 선행 모션 벡터(MV0)(725)는 TD0(723)에 걸친(예를 들어, 선행 참조 프레임(720)과 현재 프레임(710) 사이의) 모션 궤적(713)을 따르는 객체의 이동의 방향 및 크기를 기술한다. 후속 모션 벡터(MV1)(735)는 TD1(733)에 걸친(예를 들어, 현재 프레임(710)과 후속 참조 프레임(730) 사이의) 모션 궤적(713)을 따르는 객체의 이동의 방향 및 크기를 기술한다. 이와 같이, 양방향 인터-예측(700)에서, 현재 블록(711)은 선행 참조 블록(721) 및/또는 후속 참조 블록(731), MV0(725), 및 MV1(735)을 이용함으로써 코딩되고 재구성될 수 있다.

[0100] 병합 모드 및 AMVP(advanced motion vector prediction) 모드 둘 다에서, 후보 리스트 결정 패턴에 의해 정의된 순서로 후보 모션 벡터들을 후보 리스트에 추가하는 것에 의해 후보 리스트가 생성된다. 이러한 후보 모션 벡터들은 단방향 인터-예측(600), 양방향 인터-예측(700), 또는 이들의 조합들에 따른 모션 벡터들을 포함할 수 있다. 구체적으로, 이웃 블록들이 인코딩될 때 이러한 블록들에 대해 모션 벡터들이 생성된다. 이러한 모션 벡터들은 현재 블록에 대한 후보 리스트에 추가되고, 현재 블록에 대한 모션 벡터는 후보 리스트로부터 선택된다. 그 후, 모션 벡터는 후보 리스트 내의 선택된 모션 벡터의 인덱스로서 시그널링될 수 있다. 디코더는 인코더와 동일한 프로세스를 사용하여 후보 리스트를 구성할 수 있고, 시그널링된 인덱스에 기초하여 후보 리스트로부터 선택된 모션 벡터를 결정할 수 있다. 따라서, 후보 모션 벡터들은, 이러한 이웃 블록들이 인코딩될 때 어느 접근법이 사용되는지에 따라, 단방향 인터-예측(600) 및/또는 양방향 인터-예측(700)에 따라 생성되는 모션 벡터들을 포함한다.

[0101] 도 8은 인접한 코딩된 블록들(802)로부터의 후보 모션 벡터들에 기초하여 현재 블록(801)을 코딩하는 예(800)를 예시하는 개략도이다. 방법(100)을 운영하는 및/또는 코덱 시스템(200)의 기능성을 이용하는 인코더(300) 및/또는 디코더(400)는 인접 블록들(802)을 이용하여 후보 리스트를 생성할 수 있다. 이러한 후보 리스트는 단방향 인터-예측(600) 및/또는 양방향 인터-예측(700)에 따라 인터-예측에 이용될 수 있다. 그 후, 후보 리스트는 현재 블록(801)을 인코딩/디코딩하는데 이용될 수 있고, 이는 픽처(500)와 같은 픽처를 파티셔닝함으로써 생성될 수 있다.

[0102] 현재 블록(801)은, 예에 따라, 특정된 시간에, 인코더에서 인코딩되거나 디코더에서 디코딩되는 블록이다. 코딩된 블록들(802)은 특정된 시간에 이미 인코딩된 블록들이다. 따라서, 코딩된 블록들(802)은 후보 리스트를 생성할 때 잠재적으로 사용가능하다. 현재 블록(801) 및 코딩된 블록들(802)은 공통 프레임에 포함될 수 있고/있거나 시간적으로 인접한 프레임에 포함될 수 있다. 코딩된 블록들(802)이 현재 블록(801)과 공통 프레임에 포함될 때, 코딩된 블록들(802)은 현재 블록(801)의 경계에 바로 인접한(예를 들어, 접하는) 경계를 포함한다. 코딩된 블록(802)이 시간적으로 인접한 프레임에 포함될 때, 코딩된 블록(802)은 현재 프레임에서의 현재 블록(801)의 위치와 동일한 시간적으로 인접한 프레임에서의 위치에 위치된다. 후보 리스트는 코딩된 블록들(802)로부터의 모션 벡터들을 후보 모션 벡터들로서 추가함으로써 생성될 수 있다. 그 후 현재 블록(801)은 후보 리스트로부터 후보 모션 벡터를 선택하고 선택된 후보 모션 벡터의 인덱스를 시그널링함으로써 코딩될 수 있다.

[0103] 도 9는 모션 벡터들의 후보 리스트를 결정하기 위한 예시적인 패턴(900)을 예시하는 개략도이다. 구체적으로, 방법(100)을 운영하고/하거나 코덱 시스템(200)의 기능성을 이용하는 인코더(300) 및/또는 디코더(400)는 픽처(500)로부터 파티셔닝된 현재 블록(801)을 인코딩하기 위한 후보 리스트(911)를 생성할 때 사용하기 위해 후보 리스트 결정 패턴(900)을 이용할 수 있다. 결과적인 후보 리스트(911)는 단방향 인터-예측(600) 및/또는 양방향 인터-예측(700)에 따른 인터-예측에 이용될 수 있는, 병합 후보 리스트 또는 AMVP 후보 리스트일 수 있다.

[0104] 현재 블록(901)을 인코딩할 때, 후보 리스트 결정 패턴(900)은 유효 후보 모션 벡터들에 대해 현재 블록(901)과 동일한 픽처/프레임에서 A0, A1, B0, B1, 및/또는 B2로 표시된 위치들(905)을 검색한다. 후보 리스트 결정 패턴(900)은 또한 유효 후보 모션 벡터들에 대해 동위치의 블록(909)을 검색할 수 있다. 동위치의 블록(909)은 현재 블록(901)과 동일한 위치의 블록이지만, 시간적으로 인접한 픽처/프레임에 포함된다. 그 후, 후보 모션 벡터들은 미리 결정된 체킹 순서로 후보 리스트(911)에 위치될 수 있다. 따라서, 후보 리스트(911)는 인덱싱된 후보 모션 벡터들의 절차적으로 생성된 리스트이다.

[0105] 후보 리스트(911)는 현재 블록(901)에 대한 인터-예측을 수행하기 위해 모션 벡터를 선택하는데 이용될 수 있다. 예를 들어, 인코더는 후보 리스트(911)로부터 후보 모션 벡터들이 가리키는 참조 블록들의 샘플들을 획득

득할 수 있다. 그 후, 인코더는 현재 블록(901)과 가장 가깝게 매칭되는 참조 블록을 가리키는 후보 모션 벡터를 선택할 수 있다. 그 후, 선택된 후보 모션 벡터의 인덱스는 현재 블록(901)을 표현하도록 인코딩될 수 있다. 일부 경우들에서, 후보 모션 벡터(들)는 부분 참조 샘플들(915)을 포함하는 참조 블록을 가리킨다. 이 경우에, 보간 필터(913)가 모션 벡터 선택을 지원하기 위해 완전한 참조 샘플(915)을 재구성하는 데 이용될 수 있다. 보간 필터(913)는 신호를 업샘플링할 수 있는 필터이다. 구체적으로는, 보간 필터(913)는 부분/저품질 신호를 입력으로서 수용하고 보다 완전한/고품질 신호의 근사치를 결정할 수 있는 필터이다. 이와 같이, 보간 필터(913)는 특정 경우들에서 현재 블록(901)에 대한 참조 블록을 선택하는 데 그리고 따라서 현재 블록(901)을 인코딩할 모션 벡터를 선택하기 위해 사용하기 위한 참조 샘플들(915)의 완전한 세트를 획득하기 위해 이용될 수 있다.

[0106] 후보 리스트를 이용하여 인터-예측에 기초하여 블록을 코딩하기 위한 전술한 메커니즘들은 서브-픽처들(510)과 같은 서브-픽처들이 이용될 때 소정의 에러를 유발할 수 있다. 구체적으로는, 현재 블록(901)이 현재 서브-픽처에 포함되어 있지만 모션 벡터가 인접한 서브-픽처에 적어도 부분적으로 위치하는 참조 블록을 가리킬 때 문제가 발생할 수 있다. 그러한 경우에, 현재 서브-픽처는 인접한 서브-픽처 없이 제시를 위해 추출될 수 있다. 이것이 발생할 때, 인접한 서브-픽처 내의 참조 블록의 부분들은 디코더로 송신되지 않을 수 있고, 따라서 참조 블록은 현재 블록(901)을 디코딩하는 데 이용 가능하지 않을 수 있다. 이것이 발생할 때, 디코더는 현재 블록(901)을 디코딩하기에 충분한 데이터에 액세스하지 못한다.

[0107] 본 개시내용은 이 문제를 다루기 위한 메커니즘들을 제공한다. 일 예에서, 현재 서브-픽처가 픽처로서 취급되어야 함을 나타내는 플래그가 이용된다. 이 플래그는 서브-픽처의 개별 추출을 지원하도록 설정될 수 있다. 구체적으로, 플래그가 설정될 때, 현재 서브-픽처는 다른 서브-픽처들 내의 데이터를 참조하지 않고 인코딩되어야 한다. 이 경우에, 현재 서브-픽처는 현재 서브-픽처가 다른 서브-픽처들과 별도로 코딩되고 개별 픽처로서 디스플레이될 수 있다는 점에서 픽처처럼 취급된다. 따라서, 이 플래그는 `subpic_treated_as_pic_flag[i]`로서 표시될 수 있으며, 여기서 *i*는 현재 서브-픽처의 인덱스이다. 플래그가 설정될 때, 동위치의 블록(909)으로부터 획득된 모션 벡터 후보들(모션 벡터 예측자들이라고도 함)은 현재 서브-픽처 내부를 가리키는 모션 벡터만을 포함한다. 현재 서브-픽처의 외부로 가리키는 임의의 모션 벡터 예측자들은 후보 리스트(911)로부터 배제된다. 이것은 현재 서브-픽처의 외부로 가리키는 모션 벡터들이 선택되지 않고 연관된 에러들이 회피되는 것을 보장한다. 이 예는 구체적으로 동위치의 블록(909)으로부터의 모션 벡터들에 적용된다. 동일한 픽처/프레임 내의 검색 위치들(905)로부터의 모션 벡터들은 후술하는 바와 같이 개별 메커니즘들에 의해 결정될 수 있다.

[0108] 현재 서브-픽처가 픽처로서 취급될 때(예를 들어, `subpic_treated_as_pic_flag[i]`가 설정될 때) 검색 위치들(905)을 어드레싱하기 위해 다른 예가 이용될 수 있다. 현재 서브-픽처가 픽처처럼 취급될 때, 현재 서브-픽처는 다른 서브-픽처들을 참조하지 않고 추출되어야 한다. 예시적인 메커니즘은 보간 필터들(913)에 관한 것이다. 보간 필터(913)는 하나의 위치에서의 샘플들에 적용되어 다른 위치에서의 관련 샘플들을 보간(예를 들어, 예측)할 수 있다. 본 예에서, 보간 필터들(913)이 현재 서브-픽처로부터의 참조 샘플들(915)에만 기초하여 이러한 참조 샘플들(915)을 보간할 수 있는 한, 검색 위치들(905)에 있는 코딩된 블록으로부터의 모션 벡터들은 현재 서브-픽처 외부의 참조 샘플들(915)을 가리킬 수 있다. 따라서, 본 예는 보간 필터들(913)을 동일한 픽처로부터의 검색 위치들(905)로부터의 모션 벡터 후보들에 적용할 때 적용되는 클리핑 함수를 이용한다. 이 클리핑 함수는 인접한 서브-픽처들로부터 데이터를 클리핑하고, 따라서 모션 벡터 후보들이 가리키는 참조 샘플들(915)을 결정할 때 보간 필터(913)에 대한 입력으로서 이러한 데이터를 제거한다. 이 접근법은 서브-픽처가 픽처로서 취급될 때 개별 추출 및 디코딩을 지원하기 위해 인코딩 동안 서브-픽처들 사이의 분리를 유지한다. 클리핑 함수는 루마 샘플 이중선형 보간 프로세스, 루마 샘플 8 탭 보간 필터링 프로세스, 및/또는 크로마 샘플 보간 프로세스에 적용될 수 있다.

[0109] 도 10은 예시적인 인-루프 필터(1000)를 예시하는 블록도이다. 인-루프 필터(1000)는 인-루프 필터(225, 325, 및/또는 425)를 구현하는 데 이용될 수 있다. 또한, 인-루프 필터(1000)는 방법(100)을 수행할 때 인코더 및 디코더에서 적용될 수 있다. 또한, 인-루프 필터(1000)는 패턴(900)에 따라 생성된 후보 리스트에 기초하여 단방향 인터-예측(600) 및/또는 양방향 인터-예측(700)에 따라 코딩될 수 있는, 픽처(500)로부터 파티셔닝된 현재 블록(801)을 필터링하기 위해 적용될 수 있다. 인-루프 필터(1000)는 디블록킹 필터(1043), SAO 필터(1045), 및 적응 루프 필터(ALF)(1047)를 포함한다. 인-루프 필터(1000)의 필터들은 (예를 들어, 참조 블록들로서 사용하기 전에) 인코더에서 그리고 디스플레이 전에 디코더에서 재구성된 이미지 블록들에 순차적으로 적용된다.

[0110] 디블록킹 필터(1043)는 블록 기반 인터 및 인트라 예측에 의해 생성된 블록 형상의 에지들을 제거하도록 구성된다. 디블록킹 필터(1043)는 파티션 경계들에서 발생하는 크로마 및/또는 루마 값들의 불연속들에 대해 이미지

부분(예를 들어, 이미지 슬라이스)을 스캐닝한다. 그 후, 디블록킹 필터(1043)는 평활화 함수를 블록 경계들에 적용하여 그러한 불연속들을 제거한다. 디블록킹 필터(1043)의 강도는 블록 경계에 인접한 영역에서 발생하는 공간 활동(예컨대, 루마/크로마 성분의 변동)에 따라 변할 수 있다.

[0111] SAO 필터(1045)는 인코딩 프로세스에 의해 야기되는 샘플 왜곡에 관련된 아티팩트들을 제거하도록 구성된다. 인코더에서의 SAO 필터(1045)는 재구성된 이미지의 디블로킹된 샘플들을 상대적인 디블로킹 에지 형상 및/또는 방향에 기초하여 여러 카테고리들로 분류한다. 그 후, 오프셋이 결정되고 카테고리에 기초하여 샘플들에 추가된다. 그 후, 오프셋들은 비트스트림으로 인코딩되고 디코더에서 SAO 필터(1045)에 의해 이용된다. SAO 필터(1045)는 밴딩(banding) 아티팩트들(매끄러운 전이들 대신에 값들의 대역들) 및 링잉(ringing) 아티팩트들(날카로운 에지들 근처의 의사 신호들)을 제거한다.

[0112] ALF(1047)는, 인코더에서, 재구성된 이미지를 원래의 이미지와 비교하도록 구성된다. ALF(1047)는, 예를 들어, 위너(Wiener) 기반 적응 필터를 통해, 재구성된 이미지와 원래의 이미지 사이의 차이를 기술하는 계수를 결정한다. 이러한 계수들은 비트스트림으로 인코딩되고 디코더에서 ALF(1047)에 의해 이용되어 재구성된 이미지와 원래의 이미지 사이의 차이를 제거한다.

[0113] 인-루프 필터(1000)에 의해 필터링된 이미지 데이터는 디코딩된 픽처 버퍼(223, 323, 및/또는 423)와 실질적으로 유사한 픽처 버퍼(1023)에 출력된다. 전술한 바와 같이, 디블록킹 필터(1043), SAO 필터(1045), 및/또는 ALF(1047)는 각각 `loop_filter_across_subpic_enabled` 플래그 및/또는 `loop_filter_across_tiles_enabled_flag`와 같은 플래그들에 의해 서브-픽처 경계들 및/또는 타일 경계들에서 턴오프될 수 있다.

[0114] 도 11은 픽처의 서브-픽처의 디코딩을 지원하기 위한 코딩 도구 파라미터들을 포함하는 예시적인 비트스트림(1100)을 예시하는 개략도이다. 예를 들어, 비트스트림(1100)은 코덱 시스템(200) 및/또는 디코더(400)에 의한 디코딩을 위해 코덱 시스템(200) 및/또는 인코더(300)에 의해 생성될 수 있다. 다른 예로서, 비트스트림(1100)은 단계 111에서 디코더에 의한 사용을 위해 방법(100)의 단계 109에서 인코더에 의해 생성될 수 있다. 또한, 비트스트림(1100)은 인코딩된 픽처(500), 대응하는 서브-픽처들(510), 및/또는 현재 블록들(801 및/또는 901)과 같은 연관된 코딩된 블록들을 포함할 수 있으며, 이들은 패턴(900)에 따라 생성된 후보 리스트에 기초하여 단방향 인터-예측(600) 및/또는 양방향 인터-예측(700)에 따라 코딩될 수 있다. 비트스트림(1100)은 또한 인-루프 필터(1000)를 구성하기 위한 파라미터들을 포함할 수 있다.

[0115] 비트스트림(1100)은 시퀀스 파라미터 세트(SPS)(1110), 복수의 픽처 파라미터 세트(PPS)(1111), 복수의 슬라이스 헤더(1115), 및 이미지 데이터(1120)를 포함한다. SPS(1110)는 비트스트림(1100)에 포함된 비디오 시퀀스 내의 모든 픽처들에 공통인 시퀀스 데이터를 포함한다. 그러한 데이터는 픽처 크기 조정, 비트 깊이, 코딩 도구 파라미터들, 비트 레이트 제한들 등을 포함할 수 있다. PPS(1111)는 각각의 픽처에 적용되는 파라미터들을 포함한다. 따라서, 비디오 시퀀스에서의 각각의 픽처는 PPS(1111)를 참조할 수 있다. 각각의 픽처가 PPS(1111)를 참조하지만, 일부 예들에서 단일 PPS(1111)가 다수의 픽처에 대한 데이터를 포함할 수 있다는 점에 유의해야 한다. 예를 들어, 다수의 유사한 픽처들이 유사한 파라미터들에 따라 코딩될 수 있다. 그러한 경우에, 단일 PPS(1111)는 그러한 유사한 픽처들에 대한 데이터를 포함할 수 있다. PPS(1111)는 대응하는 픽처들, 양자화 파라미터들, 오프셋들 등에서 슬라이스들에 대해 이용가능한 코딩 도구들을 나타낼 수 있다. 슬라이스 헤더(1115)는 픽처 내의 각각의 슬라이스에 특정한 파라미터들을 포함한다. 따라서, 비디오 시퀀스에서 슬라이스마다 하나의 슬라이스 헤더(1115)가 있을 수 있다. 슬라이스 헤더(1115)는 슬라이스 타입 정보, POC(picture order count)들, 참조 픽처 리스트들, 예측 가중치들, 타일 엔트리 포인트들, 디블로킹 파라미터들 등을 포함할 수 있다. 슬라이스 헤더(1115)는 일부 컨텍스트에서 타일 그룹 헤더라고도 지칭될 수 있다는 점에 유의해야 한다.

[0116] 이미지 데이터(1120)는 인터-예측 및/또는 인트라-예측에 따라 인코딩된 비디오 데이터뿐만 아니라 대응하는 변환된 및 양자화된 잔차 데이터를 포함한다. 예를 들어, 비디오 시퀀스는 이미지 데이터로서 코딩된 복수의 픽처를 포함한다. 픽처는 비디오 시퀀스의 단일 프레임이고, 따라서 비디오 시퀀스를 디스플레이할 때 일반적으로 단일 유닛으로서 디스플레이된다. 그러나, 서브-픽처들은 가상 현실, 픽처 인 픽처 등과 같은 특정 기술들을 구현하기 위해 디스플레이될 수 있다. 픽처들은 각각 PPS(1111)를 참조한다. 픽처들은 전술한 바와 같이 서브-픽처들, 타일들 및/또는 슬라이스들로 분할된다. 일부 시스템들에서, 슬라이스들은 타일들을 포함하는 타일 그룹들로서 지칭된다. 타일들의 슬라이스들 및/또는 타일 그룹들은 슬라이스 헤더(1115)를 참조한다. 슬라이스들은 CTU들 및/또는 CTB들로 추가로 분할된다. CTU들/CTB들은 코딩 트리들에 기초하여 코딩 블록들로 추가로

분할된다. 그 후 코딩 블록들은 예측 메커니즘들에 따라 인코딩/디코딩될 수 있다.

[0117] 비트스트림(1100) 내의 파라미터 세트들은 본 명세서에 설명된 예들을 구현하기 위해 이용될 수 있는 다양한 데이터를 포함한다. 제1 예시적인 구현을 지원하기 위해, 비트스트림(1100)의 SPS(1110)는 특정된 서브-픽처와 관련된 픽처로서 취급되는 서브-픽처(sub-pic) 플래그(1131)를 포함한다. 일부 예들에서, 픽처로서 취급되는 서브-픽처 플래그(1131)는 `subpic_treated_as_pic_flag[i]`로서 표시되고, 여기서 *i*는 플래그와 관련된 서브-픽처의 인덱스이다. 예를 들어, 픽처로서 취급되는 서브-픽처 플래그(1131)는 (이미지 데이터(1120) 내의) 코딩된 비디오 시퀀스 내의 각각의 코딩된 픽처의 *i*번째 서브-픽처가 인-루프 필터링 연산들을 제외한 디코딩 프로세스에서 픽처로서 취급된다는 것을 특정하기 위해 1로 설정될 수 있다. 픽처로서 취급되는 서브-픽처 플래그(1131)는 현재 픽처 내의 현재 서브-픽처가 인터-예측에 따라 코딩되었을 때 이용될 수 있다. 픽처로서 취급되는 서브-픽처 플래그(1131)가 현재 서브-픽처가 픽처로서 취급된다는 것을 나타내도록 설정될 때, 현재 블록에 대한 후보 모션 벡터들의 후보 리스트는 동위치의 블록에 포함되고 현재 서브-픽처의 외부를 가리키는 후보 리스트로부터 동위치의 모션 벡터들을 배제함으로써 결정될 수 있다. 이것은 현재 서브-픽처의 외부를 가리키는 모션 벡터들이 선택되지 않고 현재 서브-픽처가 다른 서브-픽처들과 별도로 추출될 때 연관된 에러가 회피되도록 보장한다.

[0118] 일부 예들에서, 현재 블록에 대한 모션 벡터들의 후보 리스트는 시간 루마 모션 벡터 예측에 따라 결정된다. 예를 들어, 현재 블록이 루마 샘플들의 루마 블록이고, 현재 블록에 대한 선택된 현재 모션 벡터가 참조 블록 내의 참조 루마 샘플들을 가리키는 시간 루마 모션 벡터이고, 현재 블록이 참조 루마 샘플들에 기초하여 코딩될 때, 시간 루마 모션 벡터 예측이 이용될 수 있다. 이러한 경우에, 시간 루마 모션 벡터 예측은 다음에 따라 수행된다:

[0119] `xColBr = xCb + cbWidth;`

[0120] `yColBr = yCb + cbHeight;`

[0121] `rightBoundaryPos = subpic_treated_as_pic_flag[ SubPicIdx ] ?`

[0122] `SubPicRightBoundaryPos : pic_width_in_luma_samples - 1;` 및

[0123] `botBoundaryPos = subpic_treated_as_pic_flag[ SubPicIdx ] ?`

[0124] `SubPicBotBoundaryPos : pic_height_in_luma_samples - 1,`

[0125] 여기서 `xColBr` 및 `yColBr`는 동위치의 블록의 위치를 특정하고, `xCb` 및 `yCb`는 현재 픽처의 좌측 상단 샘플에 대한 현재 블록의 좌측 상단 샘플을 특정하고, `cbWidth`는 현재 블록의 폭이고, `cbHeight`는 현재 블록의 높이이고, `SubPicRightBoundaryPos`는 서브-픽처의 우측 경계의 위치이고, `SubPicBotBoundaryPos`는 서브-픽처의 하단 경계의 위치이고, `pic_width_in_luma_samples`는 루마 샘플들에서 측정된 현재 픽처의 폭이고, `pic_height_in_luma_samples`는 루마 샘플들에서 측정된 현재 픽처의 높이이고, `botBoundaryPos`는 서브-픽처의 하단 경계의 산출된 위치이고, `rightBoundaryPos`는 서브-픽처의 우측 경계의 산출된 위치이고, `SubPicIdx`는 서브-픽처의 인덱스이고, 동위치의 모션 벡터들은 `yCb >> CtbLog2SizeY`가 `yColBr >> CtbLog2SizeY`와 동일하지 않을 때 제외되고, `CtbLog2SizeY`는 코딩 트리 블록의 크기를 나타낸다.

[0126] 픽처로서 취급되는 서브-픽처 플래그(1131)는 또한 제2 예시적인 구현을 위해 이용될 수 있다. 제1 예제와 같이, 픽처로서 취급되는 서브-픽처 플래그(1131)는 현재 픽처 내의 현재 서브-픽처가 인터-예측에 따라 코딩되었을 때 이용될 수 있다. 이 예에서, 모션 벡터는 (예를 들어, 후보 리스트로부터) 서브-픽처의 현재 블록에 대해 결정될 수 있다. 픽처로서 취급되는 서브-픽처 플래그(1131)가 설정될 때, 클리핑 함수가 참조 블록 내의 샘플 위치들에 적용될 수 있다. 샘플 위치는 루마 값 및/또는 한 쌍의 크로마 값을 포함하는 단일 샘플을 포함할 수 있는 픽처 내의 위치이다. 그 후, 모션 벡터가 현재 서브-픽처의 외부를 가리킬 때 보간 필터가 적용될 수 있다. 이 클리핑 함수는 보간 필터가 개별 추출을 지원하기 위해 서브-픽처들 간의 분리를 유지하기 위해 인접한 서브-픽처들로부터의 데이터에 의존하지 않도록 보장한다.

[0127] 클리핑 함수는 루마 샘플 이중선형 보간 프로세스에서 적용될 수 있다. 루마 샘플 이중선형 보간 프로세스는 전체 샘플 유닛들(`xIntL`, `yIntL`) 내의 루마 위치를 포함하는 입력들을 수신할 수 있다. 루마 샘플 이중선형 보간 프로세스는 예측된 루마 샘플 값(`predSampleLXL`)을 출력한다. 클리핑 함수는 다음과 같이 샘플 위치들에 적용된다. `subpic_treated_as_pic_flag[SubPicIdx]`가 1일 때, 다음이 적용된다:

[0128] $xInti = Clip3(SubPicLeftBoundaryPos, SubPicRightBoundaryPos, xIntL+i)$, 및

[0129] $yInti = Clip3(SubPicTopBoundaryPos, SubPicBotBoundaryPos, yIntL+i)$,

[0130] 여기서 `subpic_treated_as_pic_flag`는 서브-픽처가 픽처로서 취급된다는 것을 나타내기 위해 설정된 플래그이고, `SubPicIdx`는 서브-픽처의 인덱스이고, $xInti$ 및 $yInti$ 는 인덱스 i 에서의 클리핑된 샘플 위치이고, `SubPicRightBoundaryPos`는 서브-픽처의 우측 경계의 위치이고, `SubPicLeftBoundaryPos`는 서브-픽처의 좌측 경계의 위치이고, `SubPicTopBoundaryPos`는 서브-픽처의 상단 경계의 위치이고, `SubPicBotBoundaryPos`는 서브-픽처의 하단 경계의 위치이고, `Clip3`은 다음에 따른 클리핑 함수이다:

$$Clip3(x, y, z) = \begin{cases} x & ; \quad z < x \\ y & ; \quad z > y \\ z & ; \quad \text{그 이외} \end{cases}$$

[0132] 여기서, x , y 및 z 는 수치 입력 값들이다.

[0133] 클리핑 함수는 또한 루마 샘플 8 탭 보간 필터링 프로세스에서 적용될 수 있다. 루마 샘플 8 탭 보간 필터링 프로세스는 전체 샘플 유닛들($xIntL$, $yIntL$)에 루마 위치를 포함하는 입력들을 수신한다. 루마 샘플 이중선형 보간 프로세스는 예측된 루마 샘플 값(`predSampleLXL`)을 출력한다. 클리핑 함수는 다음과 같이 샘플 위치들에 적용된다. `subpic_treated_as_pic_flag[SubPicIdx]`가 1일 때, 다음이 적용된다:

[0134] $xInti = Clip3(SubPicLeftBoundaryPos, SubPicRightBoundaryPos, xIntL+i-3)$, 및

[0135] $yInti = Clip3(SubPicTopBoundaryPos, SubPicBotBoundaryPos, yIntL+i-3)$,

[0136] 여기서 `subpic_treated_as_pic_flag`는 서브-픽처가 픽처로서 취급된다는 것을 나타내기 위해 설정된 플래그이고, `SubPicIdx`는 서브-픽처의 인덱스이고, $xInti$ 및 $yInti$ 는 인덱스 i 에서의 클리핑된 샘플 위치이고, `SubPicRightBoundaryPos`는 서브-픽처의 우측 경계의 위치이고, `SubPicLeftBoundaryPos`는 서브-픽처의 좌측 경계의 위치이고, `SubPicTopBoundaryPos`는 서브-픽처의 상단 경계의 위치이고, `SubPicBotBoundaryPos`는 서브-픽처의 하단 경계의 위치이고, `Clip3`은 상술한 바와 같다.

[0137] 클리핑 함수는 또한 크로마 샘플 보간 프로세스에서 적용될 수 있다. 크로마 샘플 보간 프로세스는 전체 샘플 유닛들($xIntC$, $yIntC$)에 크로마 위치를 포함하는 입력들을 수신한다. 크로마 샘플 보간 프로세스는 예측된 크로마 샘플 값(`predSampleLXC`)을 출력한다. 클리핑 함수는 다음과 같이 샘플 위치들에 적용된다. `subpic_treated_as_pic_flag[SubPicIdx]`가 1일 때, 다음이 적용된다:

[0138] $xInti = Clip3(SubPicLeftBoundaryPos/SubWidthC, SubPicRightBoundaryPos/SubWidthC, xIntC+i)$, 및

[0139] $yInti = Clip3(SubPicTopBoundaryPos/SubHeightC, SubPicBotBoundaryPos/SubHeightC, yIntC+i)$,

[0140] 여기서 `subpic_treated_as_pic_flag`는 서브-픽처가 픽처로서 취급된다는 것을 나타내기 위해 설정된 플래그이고, `SubPicIdx`는 서브-픽처의 인덱스이고, $xInti$ 및 $yInti$ 는 인덱스 i 에서의 클리핑된 샘플 위치이고, `SubPicRightBoundaryPos`는 서브-픽처의 우측 경계의 위치이고, `SubPicLeftBoundaryPos`는 서브-픽처의 좌측 경계의 위치이고, `SubPicTopBoundaryPos`는 서브-픽처의 상단 경계의 위치이고, `SubPicBotBoundaryPos`는 서브-픽처의 하단 경계의 위치이고, `SubWidthC` 및 `SubHeightC`는 루마 샘플과 크로마 샘플 간의 수평 및 수직 샘플링 레이트 비율을 나타내고, `Clip3`은 전술한 바와 같다.

[0141] SPS(1110) 내의 서브-픽처에 걸친 루프 필터 인에이블 플래그(1132)가 제3 예시적인 구현을 위해 이용될 수 있다. 서브-픽처에 걸친 루프 필터 인에이블 플래그(1132)는 필터링이 특정된 서브-픽처들의 경계들에 걸쳐 이용되는지를 제어하도록 설정될 수 있다. 예를 들어, 서브-픽처에 걸친 루프 필터 인에이블 플래그(1132)는 `loop_filter_across_subpic_enabled_flag`로서 표시될 수 있다. 서브-픽처에 걸친 루프 필터 인에이블 플래그(1132)는 인-루프 필터링 연산들이 서브픽처의 경계들에 걸쳐 수행될 수 있음을 특정할 때 1로 설정되거나, 인-루프 필터링 연산들이 서브픽처의 경계들에 걸쳐 수행되지 않음을 특정할 때 0으로 설정될 수 있다. 따라서, 필터링 연산들은 서브-픽처에 걸친 루프 필터 인에이블 플래그(1132)의 값에 기초하여 서브-픽처 경계에 걸쳐 수행되거나 수행되지 않을 수 있다. 필터링 연산들은 디블록킹 필터(1043), ALF(1047), 및/또는 SAO 필터(1045)를 적용하는 것을 포함할 수 있다. 이러한 방식으로, 필터들은 개별 추출을 위해 인코딩되는 서브-픽처들에 대해 턴오프될 수 있거나 또는 그룹으로서 디스플레이를 위해 인코딩되는 서브-픽처들에 대해 턴온될 수 있다.

- [0142] PPS(1111) 내의 타일들에 걸친 루프 필터 인에이블 플래그(1134)가 제4 예시적인 구현을 위해 이용될 수 있다. 타일들에 걸친 루프 필터 인에이블 플래그(1134)는 필터링이 특정된 타일들의 경계들에 걸쳐 이용되는지를 제어하도록 설정될 수 있다. 예를 들어, 타일들에 걸친 루프 필터 인에이블 플래그(1134)는 `loop_filter_across_tiles_enabled_flag`로서 표시될 수 있다. 타일들에 걸친 루프 필터 인에이블 플래그(1134)는 인-루프 필터링 연산들이 타일의 경계들에 걸쳐 수행될 수 있음을 특정할 때 1로 설정되거나, 또는 인-루프 필터링 연산들이 타일의 경계들에 걸쳐 수행되지 않음을 특정할 때 0으로 설정될 수 있다. 따라서, 필터링 연산들은 타일들에 걸친 루프 필터 인에이블 플래그(1134)의 값에 기초하여 특정된 타일 경계들에 걸쳐 수행되거나 수행되지 않을 수 있다. 필터링 연산들은 더블록킹 필터(1043), ALF(1047), 및/또는 SAO 필터(1045)를 적용하는 것을 포함할 수 있다.
- [0143] SPS(1110) 내의 서브-픽처 데이터(1133)는 제5 예시적인 구현을 위해 이용될 수 있다. 서브-픽처 데이터(1133)는 이미지 데이터(1120) 내의 각각의 서브-픽처에 대한 폭, 높이, 및 오프셋을 포함할 수 있다. 예를 들어, 각각의 서브-픽처의 폭 및 높이는 서브-픽처 데이터(1133)에 CTB 단위로 기술될 수 있다. 일부 예들에서, 서브-픽처의 폭 및 높이는 각각 `subpic_width_minus1` 및 `subpic_height_minus1`로서 서브-픽처 데이터(1133)에 저장된다. 또한, 각각의 서브-픽처의 오프셋은 서브-픽처 데이터(1133)에 CTU 단위로 기술될 수 있다. 예를 들어, 각각의 서브-픽처의 오프셋은 서브-픽처의 좌측 상단 CTU의 수직 위치 및 수평 위치로서 특정될 수 있다. 구체적으로, 서브-픽처의 오프셋은 픽처의 좌측 상단 CTU와 서브-픽처의 좌측 상단 CTU 간의 차이로서 특정될 수 있다. 일부 예들에서, 서브-픽처의 좌측 상단 CTU의 수직 위치 및 수평 위치는 서브-픽처 데이터(1133)에 각각 `subpic_ctu_top_left_y` 및 `subpic_ctu_top_left_x`로서 저장된다. 이 예시적인 구현은 서브-픽처 데이터(1133) 내의 서브-픽처들을 타일들의 관점 대신에 CTB들/CTU들의 관점에서 설명한다. 이것은 타일들이 대응하는 픽처/서브-픽처에서 이용되지 않을 때에도 서브-픽처들이 이용되게 한다.
- [0144] SPS(1110) 내의 서브-픽처 데이터(1133), 슬라이스 헤더(1115) 내의 슬라이스 어드레스(1136), 및 슬라이스 헤더(1115) 내의 슬라이스 서브-픽처 ID(1135)는 제6 예시적인 구현을 위해 이용될 수 있다. 서브-픽처 데이터(1133)는 제5 예시적인 구현에서 설명된 바와 같이 구현될 수 있다. 슬라이스 어드레스(1136)는 슬라이스 헤더(1115)와 연관된(예를 들어, 이미지 데이터(1120) 내의) 슬라이스의 서브-픽처 레벨 슬라이스 인덱스를 포함할 수 있다. 예를 들어, 슬라이스는 픽처 내의 슬라이스의 위치에 기초하는 대신에 서브-픽처 내의 슬라이스의 위치에 기초하여 인덱싱된다. 슬라이스 어드레스(1136)는 `slice_address` 변수에 저장될 수 있다. 슬라이스 서브-픽처 ID(1135)는 슬라이스 헤더(1115)와 연관된 슬라이스를 포함하는 서브-픽처의 ID를 포함한다. 구체적으로, 슬라이스 서브-픽처 ID(1135)는 서브-픽처 데이터(1133) 내의 대응하는 서브-픽처의 설명(예를 들어, 폭, 높이, 및 오프셋)을 참조할 수 있다. 슬라이스 서브-픽처 ID(1135)는 `slice_subpic_id` 변수에 저장될 수 있다. 따라서, 슬라이스 어드레스(1136)는 서브-픽처 데이터(1133)에 기술된 바와 같이 슬라이스 서브-픽처 ID(1135)에 의해 표시되는 서브-픽처 내의 슬라이스의 위치에 기초한 인덱스로서 시그널링된다. 이러한 방식으로, 서브-픽처 내의 슬라이스의 위치는 서브-픽처가 개별적으로 추출되고 다른 서브-픽처들이 비트스트림(1100)으로부터 생략될 때에도 결정될 수 있다. 이것은 이러한 어드레싱 스킴이 각각의 서브-픽처의 어드레스들을 다른 서브-픽처들로부터 분리하기 때문이다. 따라서, 슬라이스 헤더(1115)는 슬라이스가 픽처 내의 슬라이스의 위치에 기초하여 어드레싱되는 어드레싱 스킴에서 요구되는 바와 같이 서브-픽처가 추출될 때 재가입될 필요가 없다. 이 접근법은 슬라이스가 (래스터 스캔 슬라이스와는 대조적으로) 직사각형/정사각형 슬라이스일 때 이용될 수 있다는 점에 유의해야 한다. 예를 들어, PPS(1111) 내의 `rect_slice_flag`는 슬라이스가 직사각형/정사각형 슬라이스임을 나타내기 위해 1로 설정될 수 있다.
- [0145] 일부 비디오 코딩 시스템들에서 사용되는 서브-픽처들의 예시적인 구현은 다음과 같다. CVS에 존재할 수 있는 서브-픽처들과 관련된 정보는 SPS에서 시그널링될 수 있다. 이러한 시그널링은 다음의 정보를 포함할 수 있다. CVS의 각각의 픽처 내에 존재하는 서브-픽처들의 수는 SPS 내에 포함될 수 있다. SPS 또는 CVS의 맥락에서, 모든 AU(access unit)에 대한 동위치의 서브-픽처는 총괄하여 서브-픽처 시퀀스라고 지칭될 수 있다. 각각의 서브-픽처의 특성들과 관련된 정보를 추가로 특정하기 위한 루프도 SPS 내에 포함될 수 있다. 이러한 정보는 서브-픽처 식별, 서브-픽처의 위치(예를 들어, 서브-픽처의 좌측 상단 코너 루마 샘플과 픽처의 좌측 상단 코너 루마 샘플 사이의 오프셋 거리), 및 서브-픽처의 크기를 포함할 수 있다. 또한, SPS는 또한 각각의 서브-픽처가 모션-제한된 서브-픽처인지를 시그널링하는 데 이용될 수 있고, 여기서 모션-제한된 서브-픽처는 MCTS를 포함하는 서브-픽처이다. 각각의 서브-픽처에 대한 프로파일, 티어(tier) 및 레벨 정보는 그러한 정보가 달리 도출될 수 없는 한 비트스트림에 포함될 수 있다. 이러한 정보는 전체 픽처를 포함하는 오리지널 비트스트림으로부터 서브-픽처를 추출하는 것으로부터 생성된 추출된 비트스트림에 대한 프로파일, 티어, 및 레벨 정보에 이용될 수

있다. 각각의 서브-픽처의 프로파일 및 티어는 원본의 프로파일 및 티어와 동일하도록 도출될 수 있다. 각각의 서브-픽처에 대한 레벨은 명시적으로 시그널링될 수 있다. 이러한 시그널링은 전술한 루프에 존재할 수 있다. 시퀀스-레벨 가상 참조 디코더(HRD) 파라미터들은 각각의 서브-픽처(또는 동등하게, 각각의 서브-픽처 시퀀스)에 대한 SPS의 VUI(video usability information) 부분에서 시그널링될 수 있다.

[0146] 픽처가 2개 이상의 서브-픽처로 파티셔닝되지 않을 때, 서브-픽처 ID를 제외한 서브-픽처의 특성들(예를 들어, 위치, 크기 등)은 비트스트림에서 시그널링되지 않을 수 있다. CVS 내의 픽처들 내의 서브-픽처가 추출될 때, 새로운 비트스트림 내의 각각의 액세스 유닛은 서브-픽처들을 포함하지 않을 수 있는데, 그 이유는 새로운 비트스트림 내의 각각의 AU 내의 결과적인 이미지 데이터가 다수의 서브-픽처로 파티셔닝되지 않기 때문이다. 따라서, 위치 및 크기와 같은 서브-픽처 특성들은 SPS로부터 생략될 수 있는데, 그 이유는 그러한 정보가 픽처 특성들로부터 도출될 수 있기 때문이다. 그러나, 서브-픽처 식별은 이 ID가 추출된 서브-픽처 내에 포함된 비디오 코딩 계층(VCL) NAL 유닛들/타일 그룹들에 의해 참조될 수 있으므로 여전히 시그널링된다. 서브-픽처 ID를 변경하는 것은 자원 사용을 줄이기 위해 서브-픽처를 추출할 때 회피되어야 한다.

[0147] 픽처 내의 서브-픽처의 위치(x 오프셋 및 y 오프셋)는 루마 샘플들의 단위로 시그널링될 수 있고, 서브-픽처의 좌측 상단 코너 루마 샘플과 픽처의 좌측 상단 코너 루마 샘플 사이의 거리를 나타낼 수 있다. 다른 예에서, 픽처 내의 서브-픽처의 위치는 최소 코딩 루마 블록 크기(MinCbSizeY)의 단위로 시그널링될 수 있고, 서브-픽처의 좌측 상단 코너 루마 샘플과 픽처의 좌측 상단 코너 루마 샘플 사이의 거리를 나타낼 수 있다. 다른 예에서, 서브-픽처 위치 오프셋들의 유닛은 파라미터 세트 내의 구문 요소에 의해 명시적으로 나타낼 수 있으며, 유닛은 CtbSizeY, MinCbSizeY, 루마 샘플 또는 다른 값들일 수 있다. 코텍은 서브-픽처의 우측 경계가 픽처의 우측 경계와 일치하지 않을 때, 서브-픽처의 폭이 루마 CTU 크기(CtbSizeY)의 정수배일 것을 요구할 수 있다. 마찬가지로, 코텍은 서브-픽처의 하단 경계가 픽처의 하단 경계와 일치하지 않을 때, 서브-픽처의 높이가 CtbSizeY의 정수배일 것을 추가로 요구할 수 있다. 코텍은 또한 서브-픽처의 폭이 루마 CTU 크기의 정수배가 아닐 때, 서브-픽처가 픽처 내의 최우측 위치에 위치하는 것을 요구할 수 있다. 마찬가지로, 코텍은 또한 서브-픽처의 높이가 루마 CTU 크기의 정수배가 아닐 때 서브-픽처가 픽처 내의 최하단 위치에 배치될 것을 요구할 수 있다. 서브-픽처의 폭이 루마 CTU 크기의 단위로 시그널링되고 서브-픽처의 폭이 루마 CTU 크기의 정수배가 아닐 때, 루마 샘플들에서의 실제 폭은 서브-픽처의 오프셋 위치, 루마 CTU 크기에서의 서브-픽처의 폭, 및 루마 샘플들에서의 픽처의 폭에 기초하여 도출될 수 있다. 마찬가지로, 서브-픽처의 높이가 루마 CTU 크기의 단위로 시그널링되고, 서브-픽처의 높이가 루마 CTU 크기의 정수배가 아닐 때, 루마 샘플들에서의 실제 높이는 서브-픽처의 오프셋 위치, 루마 CTU 크기에서의 서브-픽처의 높이 및 루마 샘플들에서의 픽처의 높이에 기초하여 도출될 수 있다.

[0148] 임의의 서브-픽처에 대해, 서브-픽처 ID는 서브-픽처 인덱스와 상이할 수 있다. 서브-픽처 인덱스는 SPS 내의 서브-픽처들의 루프에서 시그널링되는 서브-픽처의 인덱스일 수 있다. 대안적으로, 서브-픽처 인덱스는 픽처에 대해 서브-픽처 래스터 스캔 순서로 할당된 인덱스일 수 있다. 각각의 서브-픽처의 서브-픽처 ID의 값이 그의 서브-픽처 인덱스와 동일할 때, 서브-픽처 ID는 시그널링되거나 도출될 수 있다. 각각의 서브-픽처의 서브-픽처 ID가 그의 서브-픽처 인덱스와 상이할 때, 서브-픽처 ID가 명시적으로 시그널링된다. 서브-픽처 ID들을 시그널링하기 위한 비트들의 수는 서브-픽처 특성들을 포함하는 동일한 파라미터 세트에서(예를 들어, SPS에서) 시그널링될 수 있다. 서브-픽처 ID에 대한 일부 값들은 특정 목적들을 위해 예약될 수 있다. 이러한 값 예약은 다음과 같을 수 있다. 타일 그룹/슬라이스 헤더들이 어느 서브-픽처가 타일 그룹을 포함하는지를 특정하는 서브-픽처 ID를 포함할 때, 값 0은 예약될 수 있고, 에플레이션 방지 코드를 생성하는 것을 회피하기 위해 타일 그룹/슬라이스 헤더의 시작에서 처음 몇 개의 비트가 모두 0이 아닌 것을 보장하기 위해 서브-픽처들에 대해 사용되지 않을 수 있다. 픽처의 서브-픽처들이 오버랩 및 갭(gap) 없이 픽처의 전체 영역을 커버하지 않을 때, 값(예를 들어, 값 1)은 서브-픽처들 중 임의의 것의 일부가 아닌 타일 그룹들에 대해 예약될 수 있다. 대안적으로, 나머지 영역의 서브-픽처 ID는 명시적으로 시그널링될 수 있다. 서브-픽처 ID를 시그널링하기 위한 비트들의 수는 다음과 같이 제약될 수 있다. 값 범위는 서브-픽처 ID의 예약된 값들을 포함하는, 픽처 내의 모든 서브-픽처를 고유하게 식별하기에 충분해야 한다. 예를 들어, 서브-픽처 ID에 대한 최소 비트 수는 $\text{Ceil}(\log_2(\text{픽처 내의 서브-픽처들의 수} + \text{예약된 서브-픽처 ID의 수}))$ 의 값일 수 있다.

[0149] 루프 내의 서브-픽처들의 결합은 갭 없이 그리고 오버랩 없이 전체 픽처를 커버하도록 요구될 수 있다. 이 제약이 적용될 때, 서브-픽처가 모션 제약된 서브-픽처인지를 특정하기 위해 각각의 서브-픽처마다 플래그가 존재하며, 이는 서브-픽처가 추출될 수 있다는 것을 의미한다. 대안적으로, 서브-픽처들의 결합은 전체 픽처를 커버하지 않을 수 있다. 그러나, 픽처의 서브-픽처들 사이에 오버랩이 없을 수 있다.

- [0150] 서브-픽처 ID들은 서브-픽처 추출 프로세스를 지원하기 위해 NAL 유닛 헤더 직후에 존재할 수 있으며, 따라서 추출기는 나머지 NAL 유닛 비트들을 이해할 필요가 없다. VCL NAL 유닛들의 경우, 서브-픽처 ID는 타일 그룹 헤더들의 제1 비트들에 존재할 수 있다. 비-VCL NAL 유닛들의 경우, 다음이 적용될 수 있다. 서브-픽처 ID는 SPS에 대한 NAL 유닛 헤더 직후에 존재할 필요가 없을 수 있다. PPS와 관련하여, 동일한 픽처의 모든 타일 그룹이 동일한 PPS를 참조하도록 제약될 때, 서브-픽처 ID가 NAL 유닛 헤더 직후에 존재할 필요가 없다. 한편, 동일한 픽처의 타일 그룹들이 상이한 PPS들을 참조하도록 허용되는 경우, 서브-픽처 ID는 PPS의 제1 비트들에 (예를 들어, PPS NAL 유닛 헤더 직후에) 존재할 수 있다. 이 경우, 하나의 픽처의 2개의 상이한 타일 그룹은 동일한 PPS를 공유하도록 허용되지 않는다. 대안적으로, 동일한 픽처의 타일 그룹들이 상이한 PPS들을 참조하도록 허용되고, 동일한 픽처의 상이한 타일 그룹들이 또한 동일한 PPS를 공유하도록 허용될 때, 서브-픽처 ID는 PPS 구문에 존재하지 않는다. 대안적으로, 동일한 픽처의 타일 그룹들이 상이한 PPS들을 참조하도록 허용되고 동일한 픽처의 상이한 타일 그룹들이 또한 동일한 PPS를 공유하도록 허용될 때, 서브-픽처 ID들의 리스트가 PPS 구문에 존재한다. 리스트는 PPS가 적용되는 서브-픽처들을 나타낸다. 다른 비-VCL NAL 유닛들의 경우, 비-VCL 유닛이 픽처 레벨(예를 들어, 액세스 유닛 디리미터(delimiter), 시퀀스의 끝, 비트스트림의 끝 등) 또는 그 이상에 적용되면, 서브-픽처 ID가 그의 NAL 유닛 헤더 직후에 존재할 필요가 없다. 그렇지 않으면, 서브-픽처 ID는 NAL 유닛 헤더 직후에 존재할 수 있다.
- [0151] 개별 서브-픽처들 내의 타일 파티셔닝은 PPS에서 시그널링될 수 있지만, 동일한 픽처 내의 타일 그룹들은 상이한 PPS들을 참조하도록 허용된다. 이 경우, 타일들은 픽처에 걸쳐서가 아니라 각각의 서브-픽처 내에서 그룹화된다. 따라서, 이러한 경우의 타일 그룹화 개념은 서브-픽처를 타일들로 파티셔닝하는 것을 포함한다. 대안적으로, 서브-픽처 파라미터 세트(SPPS)는 개별 서브-픽처들 내의 타일 파티셔닝을 설명하기 위해 이용될 수 있다. SPPS는 SPS ID를 참조하는 구문 요소를 이용하여 SPS를 참조한다. SPPS는 서브-픽처 ID를 포함할 수 있다. 서브-픽처 추출을 위해, 서브-픽처 ID를 참조하는 구문 요소는 SPPS 내의 제1 구문 요소이다. SPPS는 열들의 수, 행들의 수, 균일한 타일 간격 등을 나타내는 타일 구조를 포함한다. SPPS는 루프 필터가 연관된 서브-픽처 경계들에 걸쳐 인에이블되는지의 여부를 나타내는 플래그를 포함할 수 있다. 대안적으로, 각각의 서브-픽처에 대한 서브-픽처 특성들은 SPS 대신에 SPPS에서 시그널링될 수 있다. 개별 서브-픽처들 내의 타일 파티셔닝은 PPS에서 시그널링될 수 있지만, 동일한 픽처 내의 타일 그룹들은 상이한 PPS들을 참조하도록 허용된다. 일단 활성화되면, SPPS는 디코딩 순서로 연속적인 AU들의 시퀀스에 대해 지속될 수 있지만, CVS의 시작이 아닌 AU에서 비활성화/활성화될 수 있다. 다수의 SPPS는 다수의 서브-픽처를 갖는 단일-계층 비트스트림의 디코딩 프로세스 동안 임의의 순간에 활성화될 수 있으며, SPPS는 AU의 상이한 서브-픽처들에 의해 공유될 수 있다. 대안적으로, SPPS 및 PPS는 하나의 파라미터 세트로 병합될 수 있다. 이것이 발생하는 경우, 동일한 서브-픽처에 포함되는 모든 타일 그룹은 SPPS와 PPS 사이의 병합으로부터 야기되는 동일한 파라미터 세트를 참조하도록 제약될 수 있다.
- [0152] 서브-픽처 ID를 시그널링하기 위해 사용되는 비트 수는 NAL 유닛 헤더에서 시그널링될 수 있다. 이러한 정보는, 존재할 때, 서브-픽처 추출 프로세스가 NAL 유닛의 페이로드(예를 들어, NAL 유닛 헤더 직후의 처음 몇 비트)의 시작에 대한 서브-픽처 ID 값을 파악하는 데 도움을 준다. 이러한 시그널링의 경우, NAL 유닛 헤더의 길이를 증가시키는 것을 피하기 위해 NAL 유닛 헤더 내의 예약된 비트들 중 일부가 사용될 수 있다. 그러한 시그널링을 위한 비트 수는 sub-picture-ID-bit-len의 값을 커버해야 한다. 예를 들어, VVCs NAL 유닛 헤더 내의 7개의 예약된 비트 중 4 비트가 이 목적을 위해 사용될 수 있다.
- [0153] 서브-픽처를 디코딩할 때, 수직 CTB 위치(xCtb) 및 수평 CTB 위치(yCtb)로 표시되는, 각각의 코딩 트리 블록의 위치는 서브-픽처 내의 루마 샘플 위치 대신에 픽처 내의 실제 루마 샘플 위치로 조정된다. 이러한 방식으로, 각각의 참조 픽처로부터 동위치의 서브-픽처의 추출은, 모든 것이 서브-픽처 대신에 픽처에 위치하는 것처럼 디코딩되기 때문에 회피될 수 있다. 코딩 트리 블록의 위치를 조정하기 위해, 변수 SubpictureXOffset 및 SubpictureYOffset는 서브-픽처 위치(subpic_x_offset 및 subpic_y_offset)에 기초하여 도출된다. 변수들의 값들은 서브-픽처 내의 각각의 코딩 트리 블록의 루마 샘플 위치 x 및 y 좌표들의 값들에 각각 가산된다. 서브-픽처 추출 프로세스는 다음과 같이 정의될 수 있다. 프로세스에 대한 입력은 추출될 타겟 서브-픽처를 포함한다. 이것은 서브-픽처 ID 또는 서브-픽처 위치의 형태로 입력될 수 있다. 입력이 서브-픽처의 위치일 때, 연관된 서브-픽처 ID는 SPS 내의 서브-픽처 정보를 파싱함으로써 해석될 수 있다. 비-VCL NAL 유닛들의 경우, 다음이 적용된다. 픽처 크기 및 레벨과 관련된 SPS 내의 구문 요소들은 서브-픽처의 크기 및 레벨 정보로 업데이트된다. 다음의 비-VCL NAL 유닛들은 추출에 의해 변경되지 않는다: PPS, AUD(access unit delimiter), EOS(end of sequence), EOB(end of bitstream), 및 픽처 레벨 또는 그 이상에 적용가능한 임의의 다른 비-VCL NAL 유닛

들. 서브-픽처 ID가 타겟 서브-픽처 ID와 동일하지 않은 나머지 비VCL NAL 유닛들이 제거된다. 서브-픽처 ID가 타겟 서브-픽처 ID와 동일하지 않은 VCL NAL 유닛들도 제거된다.

[0154] 서브-픽처 네스팅 SEI 메시지는 서브-픽처들의 세트에 대한 AU-레벨 또는 서브-픽처-레벨 SEI 메시지들의 네스팅을 위해 사용될 수 있다. 서브-픽처 네스팅 SEI 메시지에 운반되는 데이터는 버퍼링 기간, 픽처 타이밍, 및 비-HRD SEI 메시지들을 포함할 수 있다. 이 SEI 메시지의 구문 및 시맨틱스는 다음과 같을 수 있다. OMAF(omnidirectional media format) 환경들과 같은 시스템 운용의 경우, 뷰포트를 커버하는 서브-픽처 시퀀스들의 세트가 OMAF 플레이어에 의해 요청되고 디코딩될 수 있다. 따라서, 시퀀스-레벨 SEI 메시지는 직사각형 또는 정사각형 픽처 영역을 총괄하여 포함하는 서브-픽처 시퀀스들의 세트의 정보를 운반할 수 있다. 정보는 시스템들에 의해 사용될 수 있고, 정보는 서브-픽처 시퀀스들의 세트의 비트레이트뿐만 아니라 최소 디코딩 능력을 나타낸다. 정보는 서브-픽처 시퀀스들의 세트만을 포함하는 비트스트림의 레벨, 비트스트림의 비트 레이트, 및 선택적으로 서브-픽처 시퀀스들의 세트에 대해 특정된 서브-비트스트림 추출 프로세스를 포함한다.

[0155] 전술한 구현은 몇 가지 문제를 포함한다. 픽처의 폭 및 높이 및/또는 서브-픽처들의 폭들/높이들/오프셋들의 시그널링은 효율적이지 않다. 이러한 정보를 시그널링하기 위해 더 많은 비트가 저장될 수 있다. 서브-픽처 크기 및 위치 정보가 SPS에서 시그널링될 때, PPS는 타일 구성을 포함한다. 또한, PPS는 동일한 픽처의 다수의 서브-픽처에 의해 공유될 수 있다. 따라서, num_tile_columns_minus1 및 num_tile_rows_minus1에 대한 값 범위들은 더 명확하게 특정되어야 한다. 또한, 서브-픽처가 모션 제약되는지를 나타내는 플래그의 시맨틱스는 명확하게 특정되지 않는다. 레벨은 각각의 서브-픽처 시퀀스에 대해 강제적으로 시그널링된다. 그러나, 서브-픽처 시퀀스가 독립적으로 디코딩될 수 없을 때, 서브-픽처의 레벨을 시그널링하는 것은 유용하지 않다. 또한, 일부 애플리케이션들에서, 일부 서브-픽처 시퀀스들은 적어도 하나의 다른 서브-픽처 시퀀스와 함께 디코딩되고 렌더링되어야 한다. 따라서, 그러한 서브-픽처 시퀀스들 중 단일 서브-픽처 시퀀스에 대한 레벨을 시그널링하는 것은 유용하지 않을 수 있다. 또한, 각각의 서브-픽처에 대한 레벨 값을 결정하는 것은 인코더에 부담을 줄 수 있다.

[0156] 독립적으로 디코딩가능한 서브-픽처 시퀀스들의 도입으로, 픽처들의 특정 영역들의 독립적인 추출 및 디코딩을 요구하는 시나리오들은 타일 그룹들에 기초하여 작동하지 않을 수 있다. 따라서, 타일 그룹 ID들의 명시적 시그널링은 유용하지 않을 수 있다. 또한, PPS 구문 요소들 pps_seq_parameter_set_id 및 loop_filter_across_tiles_enabled_flag 각각의 값은 코딩된 픽처의 타일 그룹 헤더들에 의해 참조되는 모든 PPS에서 동일해야 한다. 이것은 활성 SPS가 CVS 내에서 변하지 않아야 하고, loop_filter_across_tiles_enabled_flag의 값이 타일들에 기초한 병렬 처리를 위한 픽처 내의 모든 타일에 대해 동일해야 하기 때문이다. 픽처 내에서 직사각형 및 래스터-스캔 타일 그룹들의 혼합을 허용할지는 명확하게 특정되어야 한다. 상이한 픽처들의 일부인 서브-픽처들을 허용하고 상이한 타일 그룹 모드들을 사용하기 위해 CVS에서 동일한 서브-픽처 ID를 이용할지도 특정되어야 한다. 시간 루마 모션 벡터 예측을 위한 도출 프로세스는 시간 모션 벡터 예측(TMVP)에서 서브-픽처 경계들을 픽처 경계들로서 취급하는 것을 가능하지 않게 할 수 있다. 또한, 루마 샘플 이중선형 보간 프로세스, 루마 샘플 8-탭 보간 필터링 프로세스, 및 크로마 샘플 보간 프로세스는 모션 보상에서 서브-픽처 경계들을 픽처 경계들로서 취급하도록 구성되지 않을 수 있다. 또한, 서브-픽처 경계들에서의 더블록킹, SAO 및 ALF 필터링 연산들의 제어를 위한 메커니즘도 특정되어야 한다.

[0157] 독립적으로 디코딩가능한 서브-픽처 시퀀스들의 도입으로, loop_filter_across_tile_groups_enabled_flag는 덜 유용할 수 있다. 이는, loop_filter_across_tile_groups_enabled_flag를 0으로 설정함으로써 병렬 처리 목적을 위한 인-루프 필터링 연산들을 턴 오프하는 것도 충족될 수 있기 때문이다. 또한, 픽처들의 특정 영역들의 독립적인 추출 및 디코딩을 가능하게 하기 위한 인-루프 필터링 연산들을 턴 오프하는 것도 loop_filter_across_sub_pic_enabled_flag를 0으로 설정함으로써 충족될 수 있다. 따라서, loop_filter_across_tile_groups_enabled_flag에 기초하여 타일 그룹 경계들에 걸쳐 인-루프 필터링 연산들을 턴 오프하기 위한 프로세스들을 추가적으로 특정하는 것은 디코더 및 낭비 비트들에 불필요하게 부담을 줄 것이다. 또한, 위에서 특정된 디코딩 프로세스는 타일 경계들에 걸쳐 ALF 필터링 연산들을 턴 오프하는 것을 가능하지 않게 할 수 있다.

[0158] 이와 같이, 본 개시내용은 서브-픽처 기반 비디오 코딩을 지원하기 위한 설계들을 포함한다. 서브-픽처는 픽처와 동일한 디코딩 프로세스를 이용하여 독립적으로 디코딩되거나 디코딩되지 않을 수 있는 픽처 내의 직사각형 또는 정사각형 영역이다. 기술의 설명은 VVC(Versatile Video Coding) 표준에 기초한다. 그러나, 기술들은 다른 비디오 코덱 사양들에도 적용될 수 있다.

- [0159] 일부 예들에서, 크기 단위는 픽처 폭 및 높이 구문 요소들 및 서브-픽처 폭/높이/offset_x/offset_y 구문 요소들의 리스트에 대해 시그널링된다. 모든 구문 요소는 xxx_minus1의 형태로 시그널링된다. 예를 들어, 크기 단위가 64개의 루마 샘플일 때, 99의 폭 값은 6400개의 루마 샘플의 픽처 폭을 특정한다. 동일한 예가 이러한 구문 요소들 중 다른 것에 적용된다. 다른 예에서, 다음 중 하나 이상이 적용된다. 크기 단위는 xxx_minus1의 형태로 픽처 폭 및 높이 구문 요소들에 대해 시그널링될 수 있다. 서브-픽처 폭/높이/offset_x/offset_y 구문 요소들의 리스트에 대해 시그널링된 그러한 크기 단위는 xxx_minus1의 형태일 수 있다. 다른 예에서, 다음 중 하나 이상이 적용된다. 픽처 폭 및 서브-픽처 폭/offset_x 구문 요소들의 리스트에 대한 크기 단위는 xxx_minus1의 형태로 시그널링될 수 있다. 픽처 높이 및 서브-픽처 높이/offset_y 구문 요소들의 리스트에 대한 크기 단위는 xxx_minus1의 형태로 시그널링될 수 있다. 다른 예에서, 다음 중 하나 이상이 적용된다. xxx_minus1 형태의 픽처 폭 및 높이 구문 요소들은 최소 코딩 유닛들의 단위로 시그널링될 수 있다. xxx_minus1 형태의 서브-픽처 폭/높이/offset_x/offset_y 구문 요소들은 CTU들 또는 CTB들의 단위로 시그널링될 수 있다. 우측 픽처 경계에서의 각각의 서브-픽처에 대한 서브-픽처 폭이 도출될 수 있다. 하단 픽처 경계에서의 각각의 서브-픽처에 대한 서브-픽처 높이가 도출될 수 있다. 서브-픽처 폭/높이/offset_x/offset_y의 다른 값들은 모두 비트스트림에서 시그널링될 수 있다. 다른 예들에서, 서브-픽처들의 폭들 및 높이들뿐만 아니라 픽처 내의 그들의 위치들을 시그널링하기 위한 모드가 서브-픽처들이 균일한 크기를 갖는 경우들에 추가될 수 있다. 서브-픽처들은 동일한 서브-픽처 행들 및 서브-픽처 열들을 포함할 때 균일한 크기를 갖는다. 이 모드에서, 서브-픽처 행들의 수, 서브-픽처 열들의 수, 각각의 서브-픽처 열의 폭, 및 각각의 서브-픽처 행의 높이가 모두 시그널링될 수 있다.
- [0160] 다른 예에서, 서브-픽처 폭 및 높이의 시그널링은 PPS에 포함되지 않을 수 있다. num_tile_columns_minus1 및 num_tile_rows_minus1은 0 내지 1의 범위에서, 정수 값(1024 포함)이어야 한다. 다른 예에서, PPS를 참조하는 서브-픽처들이 하나보다 많은 타일을 가질 때, 존재 플래그에 의해 건너뛰어지는 2개의 구문 요소가 PPS에서 시그널링될 수 있다. 이들 구문 요소는 서브-픽처 폭 및 높이를 CTB 단위로 시그널링하는 데 이용되고 PPS를 참조하는 모든 서브-픽처의 크기를 특정한다.
- [0161] 다른 예에서, 개별 서브-픽처들을 설명하는 추가 정보도 시그널링될 수 있다. sub_pic_treated_as_pic_flag[i]와 같은 플래그는 서브-픽처 시퀀스의 서브-픽처들이 인-루프 필터링 연산들 이외의 목적들을 위해 디코딩 프로세스에서 픽처들로서 취급되는지를 나타내기 위해 각각의 서브-픽처 시퀀스마다 시그널링될 수 있다. 서브-픽처 시퀀스가 따르는 레벨은 sub_pic_treated_as_pic_flag[i]가 1일 때만 시그널링될 수 있다. 서브-픽처 시퀀스는 동일한 서브-픽처 ID를 갖는 서브-픽처들의 CVS이다. sub_pic_treated_as_pic_flag[i]가 1일 때, 서브-픽처 시퀀스의 레벨도 시그널링될 수 있다. 이것은 모든 서브-픽처 시퀀스에 대한 플래그에 의해 또는 각각의 서브-픽처 시퀀스마다 하나의 플래그에 의해 제어될 수 있다. 다른 예에서, 서브-비트스트림 추출은 VCL NAL 유닛들을 변경하지 않고 인에이블될 수 있다. 이것은 PPS로부터 명시적 타일 그룹 ID 시그널링을 제거함으로써 달성될 수 있다. tile_group_address의 시맨틱스는 rect_tile_group_flag가 직사각형 타일 그룹들을 나타내는 것과 동일할 때 특정된다. tile_group_address는 서브-픽처 내의 타일 그룹들 중에서 타일 그룹의 타일 그룹 인덱스를 포함할 수 있다.
- [0162] 다른 예에서, PPS 구문 요소들 pps_seq_parameter_set_id 및 loop_filter_across_tiles_enabled_flag 각각의 값은 코딩된 픽처의 타일 그룹 헤더들에 의해 참조되는 모든 PPS에서 동일할 것이다. 다른 PPS 구문 요소들은 코딩된 픽처의 타일 그룹 헤더들을 참조하는 상이한 PPS들에 대해 상이할 수 있다. single_tile_in_pic_flag의 값은 코딩된 픽처의 타일 그룹 헤더들을 참조하는 상이한 PPS들에 대해 상이할 수 있다. 이러한 방식으로, CVS 내의 일부 픽처들은 하나의 타일만을 가질 수 있는 반면, CVS 내의 일부 다른 픽처들은 다수의 타일을 가질 수 있다. 이것은 또한 픽처의 일부 서브-픽처들(예를 들어, 매우 큰 것들)이 다수의 타일을 갖는 것을 허용하는 반면, 동일한 픽처의 다른 서브-픽처들(예를 들어, 매우 작은 것들)은 하나의 타일만을 갖는다.
- [0163] 다른 예에서, 픽처는 직사각형/정사각형 및 래스터-스캔 타일 그룹들의 혼합을 포함할 수 있다. 따라서, 픽처의 일부 서브-픽처들은 직사각형/정사각형 타일 그룹 모드를 이용하는 반면, 다른 서브-픽처들은 래스터-스캔 타일 그룹 모드를 이용한다. 이러한 유연성은 비트스트림 병합 시나리오들에 유리하다. 대안적으로, 제약은 픽처의 모든 서브-픽처가 동일한 타일 그룹 모드를 사용할 것을 요구할 수 있다. CVS에서 동일한 서브-픽처 ID를 갖는 상이한 픽처들로부터의 서브-픽처들은 상이한 타일 그룹 모드들을 사용하지 않을 수 있다. CVS에서 동일한 서브-픽처 ID를 갖는 상이한 픽처들로부터의 서브-픽처들은 상이한 타일 그룹 모드들을 사용할 수 있다.
- [0164] 다른 예에서, 서브-픽처에 대한 sub_pic_treated_as_pic_flag[i]가 1일 때, 서브-픽처에 대한 시간 모션 벡터 예측을 위한 동위치의 모션 벡터들은 서브-픽처의 경계들 내로부터 오도록 제한된다. 따라서, 서브-픽처에 대

한 시간 모션 벡터 예측은 마치 서브-픽처 경계들이 픽처 경계들인 것처럼 취급된다. 또한, 클리핑 동작들은 서브-픽처 경계들을 `sub_pic_treated_as_pic_flag[i]`가 1인 서브-픽처들에 대한 모션 보상에서의 픽처 경계들로서 취급하는 것을 가능하게 하기 위해, 루마 샘플 이중선형 보간 프로세스, 루마 샘플 8-탭 보간 필터링 프로세스, 및 크로마 샘플 보간 프로세스의 일부로서 특정된다.

[0165] 다른 예에서, 각각의 서브-픽처는 `loop_filter_across_sub_pic_enabled_flag`와 같은 시그널링된 플래그와 연관된다. 플래그는 서브-픽처의 경계들에서의 인-루프 필터링 연산들의 제어 및 대응하는 디코딩 프로세스들에서의 필터링 연산들의 제어를 위해 이용된다. 디블록킹 필터 프로세스는 `loop_filter_across_sub_pic_enabled_flag`가 0인 서브-픽처의 경계들과 일치하는 코딩 서브-블록 예지 및 변환 블록 예지에 적용되지 않을 수 있다. 대안적으로, 디블록킹 필터 프로세스는 `loop_filter_across_sub_pic_enabled_flag`가 0인 서브-픽처들의 상단 또는 좌측 경계들과 일치하는 코딩 서브-블록 예지들 및 변환 블록 예지들에 적용되지 않는다. 대안적으로, 디블록킹 필터 프로세스는 `sub_pic_treated_as_pic_flag[i]`가 1 또는 0인 서브-픽처들의 경계들과 일치하는 코딩 서브-블록 예지 및 변환 블록 예지에 적용되지 않는다. 대안적으로, 디블록킹 필터 프로세스는 서브-픽처들의 상부 또는 좌측 경계들과 일치하는 코딩 서브-블록 예지들 및 변환 블록 예지들에 적용되지 않는다. 서브-픽처에 대한 `loop_filter_across_sub_pic_enabled_flag`가 0일 때 서브-픽처의 경계들에 걸쳐 SAO 필터링 연산을 턴오프하기 위해 클리핑 동작이 특정될 수 있다. 서브-픽처에 대해 `loop_filter_across_sub_pic_enabled_flag`가 0일 때 서브-픽처의 경계들에 걸쳐 ALF 필터링 연산을 턴오프하기 위해 클리핑 동작이 특정될 수 있다. `loop_filter_across_tile_groups_enabled_flag`는 또한 PPS로부터 제거될 수 있다. 따라서, `loop_filter_across_tiles_enabled_flag`가 0일 때, 서브-픽처 경계들이 아닌 타일 그룹 경계들에 걸친 인-루프 필터링 연산들은 턴 오프되지 않는다. 루프 필터 동작들은 디블록킹, SAO, 및 ALF를 포함할 수 있다. 다른 예에서, 타일에 대한 `loop_filter_across_tiles_enabled_flag`가 0일 때 타일의 경계들에 걸쳐 ALF 필터링 연산을 턴 오프하기 위해 클리핑 동작이 특정된다.

[0166] 전술한 예들 중 하나 이상은 다음과 같이 구현될 수 있다. 서브-픽처는 픽처 내의 하나 이상의 타일 그룹 또는 슬라이스의 직사각형 또는 정사각형 영역으로서 정의될 수 있다. 처리 요소들의 다음의 분할들은 공간적 또는 컴포넌트별 파티셔닝을 형성할 수 있다: 각각의 픽처의 컴포넌트들로의 분할, 각각의 컴포넌트의 CTB들로의 분할, 각각의 픽처의 서브-픽처들로의 분할, 각각의 서브-픽처의 서브-픽처 내의 타일 열들로의 분할, 각각의 서브-픽처의 서브-픽처 내의 타일 행들로의 분할, 서브-픽처 내의 각각의 타일 열의 타일들로의 분할, 서브-픽처 내의 각각의 타일 행의 타일들로의 분할, 및 각각의 서브-픽처의 타일 그룹들로의 분할.

[0167] 서브-픽처 내의 CTB 래스터 및 타일 스캐닝 프로세스를 위한 프로세스는 다음과 같을 수 있다. CTB들의 단위로 i 번째 타일 열의 폭을 특정하는 0에서 `num_tile_columns_minus1`까지(언급된 것을 포함)의 범위에 있는 i 에 대한 리스트 `ColWidth[i]`는 다음과 같이 도출될 수 있다:

```
[0168] if( uniform_tile_spacing_flag )
[0169] for( i = 0; i <= num_tile_columns_minus1; i++ )
[0170]     ColWidth[ i ] = ( ( i + 1 ) * SubPicWidthInCtbsY ) /
[0171] ( num_tile_columns_minus1 + 1 ) -
[0172] ( i * SubPicWidthInCtbsY ) / ( num_tile_columns_minus1 + 1 )
[0173] else {
[0174] ColWidth[ num_tile_columns_minus1 ] = SubPicWidthInCtbsY      (6-1)
[0175] for( i = 0; i < num_tile_columns_minus1; i++ ) {
[0176]     ColWidth[ i ] = tile_column_width_minus1[ i ] + 1
[0177]     ColWidth[ num_tile_columns_minus1 ] -= ColWidth[ i ]
[0178] }
[0179] }
```

[0180] CTB들의 단위로 j 번째 타일 행의 높이를 특정하는 0에서 `num_tile_rows_minus1`까지(언급된 것을 포함)의 범위에 있는 j 에 대한 리스트 `RowHeight[j]`는 다음과 같이 도출된다:

```

[0181] if( uniform_tile_spacing_flag )
[0182] for( j = 0; j <= num_tile_rows_minus1; j++ )
[0183]     RowHeight[ j ] = ( ( j + 1 ) * SubPicHeightInCtbsY ) / (
[0184] num_tile_rows_minus1 + 1 ) -
[0185]         ( j * SubPicHeightInCtbsY ) / ( num_tile_rows_minus1 + 1 )
[0186] else {
[0187] RowHeight[ num_tile_rows_minus1 ] = SubPicHeightInCtbsY           (6-2)
[0188] for( j = 0; j < num_tile_rows_minus1; j++ ) {
[0189]     RowHeight[ j ] = tile_row_height_minus1[ j ] + 1
[0190]     RowHeight[ num_tile_rows_minus1 ] -= RowHeight[ j ]
[0191] }
[0192] }
[0193] CTB들의 단위로 i번째 타일 열 경계의 위치를 특정하는 0에서 num_tile_columns_minus1+1까지(언급된 것을
포함)의 범위에 있는 i에 대한 리스트 ColBd[i]는 다음과 같이 도출된다:
[0194] for( ColBd[0]=0, i=0; i<=num_tile_columns_minus1; i++)
[0195] ColBd[i+1]=ColBd [i]+ColWidth[i](6-3)
[0196] CTB들의 단위로 j번째 타일 행 경계의 위치를 특정하는 0에서 num_tile_rows_minus1+1까지(언급된 것을 포함)의
범위에 있는 j에 대한 리스트 RowBd[j]는 다음과 같이 도출된다:
[0197] for( RowBd[0]=0, j=0; j<=num_tile_rows_minus1; j++)
[0198] RowBd[j+1]=RowBd[j]+RowHeight[j](6-4)
[0199] 서브-픽처의 CTB 래스터 스캔 내의 CTB 어드레스로부터 서브-픽처의 타일 스캔 내의 CTB 어드레스로의 변환을
특정하는, 0에서 SubPicSizeInCtbsY-1까지(언급된 것을 포함)의 범위에 있는 ctbAddrRs에 대한 리스트
CtbAddrRsToTs[ctbAddrRs]는 다음과 같이 도출된다:
[0200] for( ctbAddrRs = 0; ctbAddrRs < SubPicSizeInCtbsY; ctbAddrRs++ ) {
[0201]     tbX = ctbAddrRs % SubPicWidthInCtbsY
[0202]     tbY = ctbAddrRs / SubPicWidthInCtbsY
[0203]     for( i = 0; i <= num_tile_columns_minus1; i++ )
[0204]         if( tbX >= ColBd[ i ] )
[0205]             tileX = i
[0206]     for( j = 0; j <= num_tile_rows_minus1; j++ )           (6-5)
[0207]         if( tbY >= RowBd[ j ] )
[0208]             tileY = j
[0209]     CtbAddrRsToTs[ ctbAddrRs ] = 0
[0210]     for( i = 0; i < tileX; i++ )
[0211]         CtbAddrRsToTs[ ctbAddrRs ] += RowHeight[ tileY ] * ColWidth[i]
[0212]         for( j = 0; j < tileY; j++ )
[0213]             CtbAddrRsToTs[ ctbAddrRs ] += SubPicWidthInCtbsY * RowHeight[ j ]

```

[0214] CtbAddrRsToTs[ctbAddrRs] += (tbY - RowBd[tileY]

[0215]) * ColWidth[tileX] + tbX - ColBd[tileX]

[0216] }

[0217] 타일 스캔에서의 CTB 어드레스로부터 서브-픽처의 CTB 래스터 스캔에서의 CTB 어드레스로의 변환을 특정하는, 0에서 SubPicSizeInCtbsY-1까지(언급된 것을 포함)의 범위에 있는 ctbAddrTs에 대한 리스트 CtbAddrTsToRs[ctbAddrTs]는 다음과 같이 도출된다:

[0218] for(ctbAddrRs=0; ctbAddrRs <SubPicSizeInCtbsY; ctbAddrRs++)(6-6)

[0219] CtbAddrTsToRs[CtbAddrRsToTs[ctbAddrRs]]=ctbAddrRs

[0220] 서브-픽처의 타일 스캔에서의 CTB 어드레스로부터 타일 ID로의 변환을 특정하는, 0에서 SubPicSizeInCtbsY-1까지(언급된 것을 포함)의 범위에 있는 ctbAddrTs에 대한 리스트 TileId[ctbAddrTs]는 다음과 같이 도출된다:

[0221] for(j = 0, tileIdx = 0; j <= num_tile_rows_minus1; j++)

[0222] for(i = 0; i <= num_tile_columns_minus1; i++, tileIdx++)

[0223] for(y = RowBd[j]; y < RowBd[j + 1]; y++) (6-7)

[0224] for(x = ColBd[i]; x < ColBd[i + 1]; x++)

[0225] TileId[CtbAddrRsToTs[y * SubPicWidthInCtbsY+ x]] = tileIdx

[0226] 타일 인덱스로부터 타일 내의 CTU들의 수로의 변환을 특정하는, 0에서 NumTilesInSubPic-1까지(언급된 것을 포함)의 범위에 있는 tileIdx에 대한 리스트 NumCtusInTile[tileIdx]는 다음과 같이 도출된다:

[0227] for(j = 0, tileIdx = 0; j <= num_tile_rows_minus1; j++)

[0228] for(i = 0; i <= num_tile_columns_minus1; i++, tileIdx++) (6-8)

[0229] NumCtusInTile[tileIdx] = ColWidth[i] * RowHeight[j]

[0230] 타일 내의 제1 CTB의 타일 스캔에서 타일 ID로부터 CTB 어드레스로의 변환을 특정하는, 0에서 NumTilesInSubPic-1까지(언급된 것을 포함)의 범위에 있는 tileIdx에 대한 리스트 FirstCtbAddrTs[tileIdx]는 다음과 같이 도출된다:

[0231] for(ctbAddrTs = 0, tileIdx = 0, tileStartFlag = 1; ctbAddrTs < SubPicSizeInCtbsY; ctbAddrTs++) {

[0232] if(tileStartFlag) {

[0233] FirstCtbAddrTs[tileIdx] = ctbAddrTs (6-9)

[0234] tileStartFlag = 0

[0235] }

[0236] tileEndFlag = ctbAddrTs == SubPicSizeInCtbsY - 1 || TileId[ctbAddrTs + 1]

[0237] != TileId[ctbAddrTs]

[0238] if(tileEndFlag) {

[0239] tileIdx++

[0240] tileStartFlag = 1

[0241] }

[0242] }

[0243] 루마 샘플들의 단위로 i번째 타일 열의 폭을 특정하는 ColumnWidthInLumaSamples[i]의 값들은 0에서 num_tile_columns_minus1까지(언급된 것을 포함)의 범위에 있는 i에 대해 ColWidth[i] << CtbLog2SizeY로 설정된다. 루마 샘플들의 단위로 j번째 타일 행의 높이를 특정하는 RowHeightInLumaSamples[j]의 값들은 0에서

num_tile_rows_minus1까지(언급된 것을 포함)의 범위에 있는 j에 대해 RowHeight[j] << CtbLog2SizeY로 설정된다.

[0244] 예시적인 시퀀스 파라미터 세트 RBSP 구문은 다음과 같다.

seq_parameter_set_rbsp() {	디스크립터
sps_max_sub_layers_minus1	u(3)
sps_reserved_zero_5bits	u(5)
profile_tier_level(sps_max_sub_layers_minus1)	

[0245]

sps_seq_parameter_set_id	ue(v)
chroma_format_idc	ue(v)
if(chroma_format_idc == 3)	
separate_colour_plane_flag	u(1)
pic_width_in_luma_samples	ue(v)
pic_height_in_luma_samples	ue(v)
bit_depth_luma_minus8	ue(v)
bit_depth_chroma_minus8	ue(v)
num_sub_pics_minus1	ue(v)
sub_pic_id_len_minus1	ue(v)
if(num_sub_pics_minus1 > 0)	
sub_pic_level_present_flag	u(1)
for (i = 0; i <= num_sub_pics_minus1; i++) {	
sub_pic_id[i]	u(v)
if(num_subpics_minus1 > 0) {	
sub_pic_treated_as_pic_flag[i]	u(1)
if(sub_pic_treated_as_pic_flag[i] && sub_pic_level_present_flag)	
sub_pic_level_idc[i]	u(8)
sub_pic_x_offset[i]	ue(v)
sub_pic_y_offset[i]	ue(v)
sub_pic_width_in_luma_samples[i]	ue(v)
sub_pic_height_in_luma_samples[i]	ue(v)
}	
}	
log2_max_pic_order_cnt_lsb_minus4	ue(v)
...	

[0246]

[0247] 예시적인 픽처 파라미터 세트 RBSP 구문은 다음과 같다.

pic_parameter_set_rbsp() {	디스크립터
pps_pic_parameter_set_id	ue(v)
pps_seq_parameter_set_id	ue(v)
loop_filter_across_sub_pic_enabled_flag	u(1)
single_tile_in_sub_pic_flag	u(1)
if(!single_tile_in_sub_pic_flag) {	
num_tile_columns_minus1	ue(v)
num_tile_rows_minus1	ue(v)
uniform_tile_spacing_flag	u(1)
if(!uniform_tile_spacing_flag) {	
for(i = 0; i < num_tile_columns_minus1; i++)	
tile_column_width_minus1[i]	ue(v)
for(i = 0; i < num_tile_rows_minus1; i++)	
tile_row_height_minus1[i]	ue(v)
}	
single_tile_per_tile_group_flag	u(1)
if(!single_tile_per_tile_group_flag)	
rect_tile_group_flag	u(1)
if(rect_tile_group_flag && !single_tile_per_tile_group_flag) {	
num_tile_groups_in_sub_pic_minus1	ue(v)
for(i = 0; i <= num_tile_groups_in_sub_pic_minus1; i++) {	
if(i > 0)	
top_left_tile_idx[i]	u(v)
bottom_right_tile_idx[i]	u(v)
}	
}	
loop_filter_across_tiles_enabled_flag	u(1)
}	
for(i = 0; i < 2; i++)	
num_ref_idx_default_active_minus1[i]	ue(v)

[0248]

[0249] 예시적인 일반적인 타일 그룹 헤더 구문은 다음과 같다.

tile_group_header() {	디스크립터
tile_group_pic_parameter_set_id	ue(v)
tile_group_sub_pic_id	u(v)
if(NumTilesInSubPic > 1 !(rect_tile_group_flag && num_tile_group_in_sub_pic_minus1 == 0))	
tile_group_address	u(v)

[0250]

[0251] 예시적인 코딩 트리 유닛 구문은 다음과 같다.

coding_tree_unit() {	디스크립터
xCtb = ((CtbAddrInRs % SubPicWidthInCtbsY) << CtbLog2SizeY) + sub_pic_x_offset[SubPicIdx[tile_group_subpic_id]]	
yCtb = ((CtbAddrInRs / SubPicWidthInCtbsY) << CtbLog2SizeY) + sub_pic_y_offset[SubPicIdx[tile_group_subpic_id]]	
...	

[0252]

[0253] 예시적인 시퀀스 파라미터 세트 RBSP 시맨틱스는 다음과 같다.

[0254] bit_depth_chroma_minus8은 크로마 어레이들의 샘플들의 비트 깊이 BitDepthC를 특정하고, 크로마 양자화 파라미터 범위 오프셋 QpBdOffsetC의 값은 다음과 같다:

[0255] $\text{BitDepthC} = 8 + \text{bit_depth_chroma_minus8} \quad (7-4)$

[0256] $\text{QpBdOffsetC} = 6 * \text{bit_depth_chroma_minus8} \quad (7-5)$

[0257] bit_depth_chroma_minus8은 0에서 8까지(언급된 것을 포함)의 범위에 있어야 한다.

[0258] num_sub_pics_minus1 + 1은 CVS 내의 각각의 코딩된 픽처 내의 서브-픽처들의 수를 특정한다. num_sub_pics_minus1의 값은 0에서 1024까지(언급된 것을 포함)의 범위에 있어야 한다. sub_pic_id_len_minus1 + 1은 SPS 내의 구문 요소 sub_pic_id[i] 및 타일 그룹 헤더들 내의 구문 요소 tile_group_sub_pic_id를 표현하기 위해 사용되는 비트들의 수를 특정한다. sub_pic_id_len_minus1의 값은 $\text{Ceil}(\log_2(\text{num_sub_pic_minus1}+1))-1$ 에서 9까지(언급된 것을 포함)의 범위에 있어야 한다. sub_pic_level_present_flag는 구문 요소 sub_pic_level_idc[i]가 존재할 수 있음을 특정하기 위해 1로 설정된다. sub_pic_level_present_flag는 구문 요소 sub_pic_level_idc[i]가 존재하지 않음을 특정하기 위해 0으로 설정된다. sub_pic_id[i]는 CVS 내의 각각의 코딩된 픽처의 i번째 서브-픽처의 서브-픽처 ID를 특정한다. sub_pic_id[i]의 길이는 sub_pic_id_len_minus1+1 비트이다.

[0259] sub_pic_treated_as_pic_flag[i]는 1로 설정되어, CVS 내의 각각의 코딩된 픽처의 i번째 서브-픽처가 인-루프 필터링 연산들을 제외한 디코딩 프로세스에서의 픽처로서 취급되는 것을 특정한다. sub_pic_treated_as_pic_flag[i]는 0으로 설정되어, CVS 내의 각각의 코딩된 픽처의 i번째 서브-픽처가 인-루프 필터링 연산들을 제외한 디코딩 프로세스에서의 픽처로서 취급되지 않음을 특정한다. sub_pic_level_idc[i]는 i번째 서브-픽처 시퀀스가 따르는 레벨을 나타내며, 여기서 i번째 서브-픽처 시퀀스는 CVS 내의 sub_pic_id[i]와 동일한 서브-픽처 ID를 갖는 서브-픽처들의 VCL NAL 유닛들 및 그들의 연관된 비VCL NAL 유닛들로만 구성된다. sub_pic_x_offset[i]는 CVS 내의 각각의 픽처의 좌측 상단 코너 루마 샘플에 대한 i번째 서브-픽처의 좌측 상단 코너 루마 샘플의 수평 오프셋을 루마 샘플 단위로 특정한다. 존재하지 않을 때, sub_pic_x_offset[i]의 값은 0인 것으로 추론된다. sub_pic_y_offset[i]는 CVS 내의 각각의 픽처의 좌측 상단 코너 루마 샘플에 대한 i번째 서브-픽처의 좌측 상단 코너 루마 샘플의 수직 오프셋을 루마 샘플 단위로 특정한다. 존재하지 않을 때, sub_pic_y_offset[i]의 값은 0인 것으로 추론된다. sub_pic_width_in_luma_samples[i]는 CVS 내의 각각의 픽처의 i번째 서브-픽처의 폭을 루마 샘플 단위로 특정한다. sub_pic_x_offset[i]와 sub_pic_width_in_luma_samples[i]의 합이 pic_width_in_luma_samples보다 작을 때, sub_pic_width_in_luma_samples[i]의 값은 CtbSizeY의 정수배일 것이다. 존재하지 않을 때, sub_pic_width_in_luma_samples[i]의 값은 pic_width_in_luma_samples와 동일한 것으로 추론된다. sub_pic_height_in_luma_samples[i]는 CVS 내의 각각의 픽처에 대한 i번째 서브-픽처의 높이를 루마 샘플 단위로 특정한다. sub_pic_y_offset[i]와 sub_pic_height_in_luma_samples[i]의 합이 pic_height_in_luma_samples보다 작을 때, sub_pic_height_in_luma_samples[i]의 값은 CtbSizeY의 정수배일 것이다. 존재하지 않을 때, sub_pic_height_in_luma_samples[i]의 값은 pic_height_in_luma_samples와 동일한 것으로 추론된다.

[0260] 비트스트림 적합성을 위해, 다음의 제약들이 적용된다. i 및 j의 임의의 정수 값들에 대해, i가 j와 동일할 때, sub_pic_id[i] 및 sub_pic_id[j]의 값들은 동일하지 않을 것이다. 임의의 2개의 서브-픽처 subpicA 및 subpicB에 대해, subpicA의 서브-픽처 ID가 subpicB의 서브-픽처 ID보다 작을 때, subPicA의 임의의 코딩된 타

일 그룹 NAL 유닛은 디코딩 순서로 subPicB의 임의의 코딩된 타일 그룹 NAL 유닛을 계승할 것이다. 서브-픽처들의 형상들은 각각의 서브-픽처가 디코딩될 때 그의 전체 좌측 경계 및 픽처 경계로 구성되거나 이전에 디코딩된 서브-픽처(들)의 경계들로 구성되는 전체 상단 경계를 갖도록 하는 것일 것이다.

[0261] sub_pic_id[i]와 동일한 spId 값들에 대한 리스트 SubPicIdx[spId]- i는 서브-픽처 ID로부터 서브-픽처 인덱스로의 변환을 특징하는, 0에서 num_sub_pics_minus1까지(언급된 것을 포함)의 범위에 있음 -는 다음과 같이 도출된다:

[0262] for(i = 0; i <= num_sub_pics_minus1; i++)

[0263] SubPicIdx[sub_pic_id[i]] = i (7-5)

[0264] log2_max_pic_order_cnt_lsb_minus4는 픽처 순서 카운트에 대한 디코딩 프로세스에서 사용되는 변수 MaxPicOrderCntLsb의 값을 다음과 같이 특정한다:

[0265] MaxPicOrderCntLsb = 2(log2_max_pic_order_cnt_lsb_minus4 + 4) (7-5)

[0266] log2_max_pic_order_cnt_lsb_minus4의 값은 0에서 12까지(언급된 것을 포함)의 범위에 있어야 한다.

[0267] 예시적인 픽처 파라미터 세트 RBSP 시맨틱스는 다음과 같다.

[0268] 존재할 때, PPS 구문 요소들 pps_seq_parameter_set_id 및 loop_filter_across_tiles_enabled_flag 각각의 값은 코딩된 픽처의 타일 그룹 헤더들에 의해 참조되는 모든 PPS들에서 동일할 것이다. pps_pic_parameter_set_id는 다른 구문 요소들에 의한 참조를 위해 PPS를 식별한다. pps_pic_parameter_set_id의 값은 0에서 63까지(언급된 것을 포함)의 범위에 있어야 한다. pps_seq_parameter_set_id는, 활성 SPS에 대한 sps_seq_parameter_set_id의 값을 특정한다. pps_seq_parameter_set_id의 값은 0에서 15까지(언급된 것을 포함)의 범위에 있어야 한다. loop_filter_across_sub_pic_enabled_flag는 PPS를 참조하는 서브-픽처의 경계들에 걸쳐 인-루프 필터링 연산들이 수행될 수 있음을 특징하기 위해 1로 설정된다. loop_filter_across_sub_pic_enabled_flag는 PPS를 참조하는 서브-픽처의 경계들에 걸쳐 인-루프 필터링 연산들이 수행되지 않음을 특징하기 위해 0으로 설정된다.

[0269] single_tile_in_sub_pic_flag는 PPS를 참조하는 각각의 서브-픽처 내에 하나의 타일만이 있음을 특징하기 위해 1로 설정된다. single_tile_in_sub_pic_flag는 PPS를 참조하는 각각의 서브-픽처에 2개 이상의 타일이 있음을 특징하기 위해 0으로 설정된다. num_tile_columns_minus1+1은 서브-픽처를 파티셔닝하는 타일 열들의 수를 특정한다. num_tile_columns_minus1은 0에서 1024까지(언급된 것을 포함)의 범위에 있어야 한다. 존재하지 않을 때, num_tile_columns_minus1의 값은 0과 동등한 것으로 추론된다. num_tile_rows_minus1 + 1은 서브-픽처를 파티셔닝하는 타일 행들의 수를 특정한다. num_tile_rows_minus1은 0에서 1024까지(언급된 것을 포함)의 범위에 있어야 한다. 존재하지 않을 때, num_tile_rows_minus1의 값은 0과 동등한 것으로 추론된다. 변수 NumTilesInSubPic는 (num_tile_columns_minus1 + 1) * (num_tile_rows_minus1 + 1)으로 설정된다. single_tile_in_sub_pic_flag가 0일 때, NumTilesInSubPic는 1보다 클 것이다.

[0270] uniform_tile_spacing_flag는 타일 열 경계들 및 마찬가지로 타일 행 경계들이 서브-픽처에 걸쳐 균일하게 분포되는 것을 특징하기 위해 1로 설정된다. uniform_tile_spacing_flag는 타일 열 경계들 및 마찬가지로 타일 행 경계들이 서브-픽처에 걸쳐 균일하게 분포되지 않지만 구문 요소들 tile_column_width_minus1[i] 및 tile_row_height_minus1[i]를 이용하여 명시적으로 시그널링됨을 특징하기 위해 0으로 설정된다. 존재하지 않을 때, uniform_tile_spacing_flag의 값은 1과 동등한 것으로 추론된다. tile_column_width_minus1[i] + 1은 CTB들의 단위로 i번째 타일 열의 폭을 특정한다. tile_row_height_minus1[i] + 1은 CTB들의 단위로 i번째 타일 행의 높이를 특정한다. single_tile_per_tile_group는 이 PPS를 참조하는 각각의 타일 그룹이 하나의 타일을 포함하는 것을 특징하기 위해 1로 설정된다. single_tile_per_tile_group는 이 PPS를 참조하는 타일 그룹이 하나보다 많은 타일을 포함할 수 있음을 특징하기 위해 0으로 설정된다.

[0271] rect_tile_group_flag는 서브-픽처의 각각의 타일 그룹 내의 타일들이 래스터 스캔 순서로 되어 있고 타일 그룹 정보가 PPS에서 시그널링되지 않는다는 것을 특징하기 위해 0으로 설정된다. rect_tile_group_flag는 각각의 타일 그룹 내의 타일들이 서브-픽처의 직사각형 또는 정사각형 영역을 커버하고 타일 그룹 정보가 PPS에서 시그널링되는 것을 특징하기 위해 1로 설정된다. single_tile_per_tile_group_flag가 1로 설정될 때, rect_tile_group_flag는 1과 동일한 것으로 추론된다. num_tile_groups_in_sub_pic_minus1 + 1은 PPS를 참조하는 각각의 서브-픽처 내의 타일 그룹들의 수를 특정한다. num_tile_groups_in_sub_pic_minus1의 값은 0에서

NumTilesInSubPic-1까지(언급된 것을 포함)의 범위에 있어야 한다. 존재하지 않고 single_tile_per_tile_group_flag가 1일 때, num_tile_groups_in_sub_pic_minus1의 값은 NumTilesInSubPic-1과 동일한 것으로 추론된다.

[0272] top_left_tile_idx[i]는 서브-픽처의 i번째 타일 그룹의 좌측 상단 코너에 위치하는 타일의 타일 인덱스를 특정한다. top_left_tile_idx[i]의 값은 임의의 i가 j와 동일하지 않은 경우 top_left_tile_idx[j]의 값과 동일하지 않을 것이다. 존재하지 않을 때, top_left_tile_idx[i]의 값은 i와 동일한 것으로 추론된다. top_left_tile_idx[i] 구문 요소의 길이는 $\text{Ceil}(\text{Log2}(\text{NumTilesInSubPic}))$ 비트들이다. bottom_right_tile_idx[i]는 서브-픽처의 i번째 타일 그룹의 우측 하단 코너에 위치한 타일의 타일 인덱스를 특정한다. single_tile_per_tile_group_flag가 1로 설정될 때, bottom_right_tile_idx[i]는 top_left_tile_idx[i]와 동일한 것으로 추론된다. bottom_right_tile_idx[i] 구문 요소의 길이는 $\text{Ceil}(\text{Log2}(\text{NumTilesInSubPic}))$ 비트들이다.

[0273] 임의의 특정 타일이 하나의 타일 그룹에만 포함되어야 하는 것은 비트스트림 적합성의 요건이다. 서브-픽처의 i번째 타일 그룹 내의 타일들의 수를 특정하는 변수 NumTilesInTileGroup[i], 및 관련 변수들은 다음과 같이 도출된다:

[0274] $\text{deltaTileIdx} = \text{bottom_right_tile_idx}[i] - \text{top_left_tile_idx}[i]$

[0275] $\text{NumTileRowsInTileGroupMinus1}[i] = \text{deltaTileIdx} /$

[0276] $(\text{num_tile_columns_minus1} + 1)$

[0277] $\text{NumTileColumnsInTileGroupMinus1}[i] = \text{deltaTileIdx} \%$

[0278] $(\text{num_tile_columns_minus1} + 1)$ (7-33)

[0279] $\text{NumTilesInTileGroup}[i] = (\text{NumTileRowsInTileGroupMinus1}[i] + 1) \times$

[0280] $(\text{NumTileColumnsInTileGroupMinus1}[i] + 1)$

[0281] loop_filter_across_tiles_enabled_flag는 인-루프 필터링 연산들이 PPS를 참조하는 서브-픽처들에서의 타일 경계들에 걸쳐 수행될 수 있는 것을 특정하기 위해 1로 설정된다. loop_filter_across_tiles_enabled_flag는 인-루프 필터링 연산들이 PPS를 참조하는 서브-픽처들에서의 타일 경계들에 걸쳐 수행되지 않는 것을 특정하기 위해 0으로 설정된다. 인-루프 필터링 연산들은 더블로킹 필터, 샘플 적응 오프셋 필터, 및 적응 루프 필터 동작들을 포함한다. 존재하지 않을 때, loop_filter_across_tiles_enabled_flag의 값은 1과 동일한 것으로 추론된다. num_ref_idx_default_active_minus1[i] + 1은, i가 0일 때, num_ref_idx_active_override_flag를 갖는 P 또는 B 타일 그룹들에 대한 변수 NumRefIdxActive[0]의 추론된 값을 0으로 특정하고, i가 1일 때, num_ref_idx_active_override_flag를 갖는 B 타일 그룹들에 대한 NumRefIdxActive[1]의 추론된 값을 0으로 특정한다. num_ref_idx_default_active_minus1[i]의 값은 0에서 14까지(언급된 것을 포함)의 범위에 있어야 한다.

[0282] 예시적인 일반적인 타일 그룹 헤더 시맨틱스는 다음과 같다. 존재할 때, 타일 그룹 헤더 구문 요소들 tile_group_pic_order_cnt_lsb 및 tile_group_temporal_mvp_enabled_flag 각각의 값은 코딩된 픽처의 모든 타일 그룹 헤더들에서 동일할 것이다. 존재할 때, tile_group_pic_parameter_set_id의 값은 코딩된 서브-픽처의 모든 타일 그룹 헤더들에서 동일할 것이다. tile_group_pic_parameter_set_id는 사용 중인 PPS에 대한 pps_pic_parameter_set_id의 값을 특정한다. tile_group_pic_parameter_set_id의 값은 0에서 63까지(언급된 것을 포함)의 범위에 있어야 한다. 현재 픽처의 TemporalId의 값이 현재 픽처의 타일 그룹에 의해 참조되는 각각의 PPS의 TemporalId의 값 이상이어야 한다는 것이 비트스트림 적합성의 요건이다. tile_group_sub_pic_id는 타일 그룹이 속하는 서브-픽처를 식별한다. tile_group_sub_pic_id의 길이는 sub_pic_id_len_minus1+1 비트이다. tile_group_sub_pic_id의 값은 코딩된 서브-픽처의 모든 타일 그룹 헤더들에 대해 동일할 것이다.

[0283] 변수 SubPicWidthInCtbsY, SubPicHeightInCtbsY, 및 SubPicSizeInCtbsY는 다음과 같이 도출된다:

```

i = SubPicIdx[ tile_group_subpic_id ]
SubPicWidthInCtbsY =
    Ceil( sub_pic_width_in_luma_samples[ i ] ÷ CtbSizeY )      (7-34)
SubPicHeightInCtbsY = Ceil( sub_pic_height_in_luma_samples[ i ] ÷ CtbSizeY )
SubPicSizeInCtbsY = SubPicWidthInCtbsY * SubPicHeightInCtbsY

```

[0284]

[0285] 다음의 변수들은 CTB 래스터 및 타일 스캐닝 변환 프로세스를 호출함으로써 도출된다: CTB들의 단위로 i번째 타일 열의 폭을 특징하는 0에서 num_tile_columns_minus1까지(언급된 것을 포함)의 범위에 있는 i에 대한 리스트 ColWidth[i]; CTB들의 단위로 j번째 타일 행의 높이를 특징하는 0에서 num_tile_rows_minus1까지(언급된 것을 포함)의 범위에 있는 j에 대한 리스트 RowHeight[j]; CTB들의 단위로 i번째 타일 열 경계의 위치를 특징하는 0에서 num_tile_columns_minus1 + 1까지(언급된 것을 포함)의 범위에 있는 i에 대한 리스트 ColBd[i]; CTB들의 단위로 j번째 타일 행 경계의 위치를 특징하는 0에서 num_tile_rows_minus1 + 1까지(언급된 것을 포함)의 범위에 있는 j에 대한 리스트 RowBd[j]; 서브-픽처의 CTB 래스터 스캔에서의 CTB 어드레스로부터 서브-픽처의 타일 스캔에서의 CTB 어드레스로의 변환을 특징하는 0에서 SubPicSizeInCtbsY - 1까지(언급된 것을 포함)의 범위에 있는 ctbAddrRs에 대한 리스트 CtbAddrRsToTs[ctbAddrRs]; 서브-픽처의 타일 스캔에서의 CTB 어드레스로부터 서브-픽처의 CTB 래스터 스캔에서의 CTB 어드레스로의 변환을 특징하는 0에서 SubPicSizeInCtbsY - 1까지(언급된 것을 포함)의 범위에 있는 ctbAddrTs에 대한 리스트 CtbAddrTsToRs[ctbAddrTs]; 서브-픽처의 타일 스캔에서의 CTB 어드레스로부터 타일 ID로의 변환을 특징하는 0에서 SubPicSizeInCtbsY - 1까지(언급된 것을 포함)의 범위에 있는 ctbAddrTs에 대한 리스트 TileId[ctbAddrTs]; 타일 인덱스로부터 타일 내의 CTU들의 수로의 변환을 특징하는 0에서 NumTilesInSubPic - 1까지(언급된 것을 포함)의 범위에 있는 tileIdx에 대한 리스트 NumCtusInTile[tileIdx]; 타일 내의 제1 CTB의 타일 스캔에서 타일 ID로부터 CTB 어드레스로의 변환을 특징하는 0에서 NumTilesInSubPic - 1까지(언급된 것을 포함)의 범위에 있는 tileIdx에 대한 리스트 FirstCtbAddrTs[tileIdx]; 루마 샘플들의 단위로 i번째 타일 열의 폭을 특징하는 0에서 num_tile_columns_minus1까지(언급된 것을 포함)의 범위에 있는 i에 대한 리스트 ColumnWidthInLumaSamples[i]; 및 루마 샘플들의 단위로 j번째 타일 행의 높이를 특징하는 0에서 num_tile_rows_minus1까지(언급된 것을 포함)의 범위에 있는 j에 대한 리스트 RowHeightInLumaSamples[j].

[0286] 0에서 num_tile_columns_minus1까지(언급된 것을 포함)의 범위에 있는 i에 대한 ColumnWidthInLumaSamples[i], 및 0에서 num_tile_rows_minus1까지(언급된 것을 포함)의 범위에 있는 j에 대한 RowHeightInLumaSamples[j]의 값들은 모두 0보다 클 것이다. 변수들 SubPicLeftBoundaryPos, SubPicTopBoundaryPos, SubPicRightBoundaryPos 및 SubPicBotBoundaryPos는 다음과 같이 도출된다:

```

[0287] i = SubPicIdx[ tile_group_subpic_id ]
[0288] if( sub_pic_treated_as_pic_flag[ i ] ) {
[0289]     SubPicLeftBoundaryPos = sub_pic_x_offset[ i ]
[0290]     SubPicRightBoundaryPos =
[0291]         SubPicLeftBoundaryPos + sub_pic_width_in_luma_samples[ i ] - 1
[0292]     SubPicTopBoundaryPos = sub_pic_y_offset[ i ]
[0293]     SubPicBotBoundaryPos =
[0294]         SubPicTopBoundaryPos + sub_pic_height_in_luma_samples[ i ]-1 (7-34)
[0295] }

```

[0296] 각각의 타일에 대해, 인덱스 i=0이다. 현재 서브-픽처에서의 NumTilesInSubPic-1, 변수들 TileLeftBoundaryPos[i], TileTopBoundaryPos[i], TileRightBoundaryPos[i], 및 TileBotBoundaryPos[i]는 다음과 같이 도출된다:

```

[0297] tileColIdx = i % ( num_tile_columns_minus1 + 1 )

```

[0298] $\text{tileRowIdx} = i / (\text{num_tile_columns_minus1} + 1)$

[0299] $\text{TileLeftBoundaryPos}[i] = \text{SubPicLeftBoundaryPos} + (\text{ColBd}[\text{tileColIdx}] \ll \text{CtbLog2SizeY})$

[0300] $\text{TileRightBoundaryPos}[i] = \text{SubPicLeftBoundaryPos} +$

[0301] $((\text{ColBd}[\text{tileColIdx}] + \text{ColWidth}[\text{tileColIdx}]) \ll \text{CtbLog2SizeY}) - 1$

[0302] $\text{if}(\text{TileRightBoundaryPos}[i] > \text{pic_width_in_luma_samples} - 1) \text{ (7-41)}$

[0303] $\text{TileRightBoundaryPos}[i] = \text{pic_width_in_luma_samples} - 1$

[0304] $\text{TileTopBoundaryPos}[i] = \text{SubPicTopBoundaryPos} + (\text{RowBd}[\text{tileRowIdx}]$

[0305] $\ll \text{CtbLog2SizeY})$

[0306] $\text{TileBotBoundaryPos}[i] = \text{SubPicTopBoundaryPos} +$

[0307] $((\text{RowBd}[\text{tileRowIdx}] + \text{RowHeight}[\text{tileRowIdx}]) \ll \text{CtbLog2SizeY}) - 1$

[0308] $\text{if}(\text{TileBotBoundaryPos}[i] > \text{pic_height_in_luma_samples} - 1)$

[0309] $\text{TileBotBoundaryPos}[i] = \text{pic_height_in_luma_samples} - 1$

[0310] $\text{tile_group_address}$ 는 타일 그룹 내의 제1 타일의 타일 어드레스를 특정한다. 존재하지 않을 때, $\text{tile_group_address}$ 의 값은 0과 동등한 것으로 추론된다. $\text{rect_tile_group_flag}$ 가 0이면, 다음이 적용된다: 타일 어드레스는 타일 ID이고; $\text{tile_group_address}$ 의 길이는 $\text{Ceil}(\text{Log2}(\text{NumTilesInSubPic}))$ 비트이고; $\text{tile_group_address}$ 의 값은 0에서 $\text{NumTilesInSubPic} - 1$ 까지(언급된 것을 포함)의 범위에 있어야 한다. 그렇지 않으면($\text{rect_tile_group_flag}$ 가 1이면), 다음이 적용된다: 타일 어드레스는 서브-픽처 내의 타일 그룹들 중 타일 그룹의 타일 그룹 인덱스이고; $\text{tile_group_address}$ 의 길이는 $\text{Ceil}(\text{Log2}(\text{num_tile_groups_in_sub_pic_minus1} + 1))$ 비트이고; $\text{tile_group_address}$ 의 값은 0에서 $\text{num_tile_groups_in_sub_pic_minus1}$ 까지(언급된 것을 포함)의 범위에 있어야 한다.

[0311] 다음의 제약들이 적용되는 것은 비트스트림 적합성의 요건이다. $\text{tile_group_address}$ 의 값은 동일한 코딩된 서브-픽처의 임의의 다른 코딩된 타일 그룹 NAL 유닛의 $\text{tile_group_address}$ 의 값과 동일하지 않을 것이다. 서브-픽처의 타일 그룹들은 그들의 $\text{tile_group_address}$ 값들의 오름차순으로 있어야 한다. 서브-픽처의 타일 그룹들의 형상들은 각각의 타일이 디코딩될 때 그의 전체 좌측 경계 및 서브-픽처 경계로 구성되거나 이전에 디코딩된 타일(들)의 경계들로 구성되는 전체 상단 경계를 갖도록 하는 것일 것이다.

[0312] $\text{num_tiles_in_tile_group_minus1}$ 은, 존재할 때, 타일 그룹 내의 타일들의 수 - 1을 특정한다. $\text{num_tiles_in_tile_group_minus1}$ 의 값은 0에서 $\text{NumTilesInSubPic} - 1$ 까지(언급된 것을 포함)의 범위에 있어야 한다. 존재하지 않을 때, $\text{num_tiles_in_tile_group_minus1}$ 의 값은 0과 동등한 것으로 추론된다. 현재 타일 그룹 내의 타일들의 수를 특정하는 변수 $\text{NumTilesInCurrTileGroup}$, 및 현재 타일 그룹 내의 i 번째 타일의 타일 인덱스를 특정하는 $\text{TgTileIdx}[i]$ 는 다음과 같이 도출된다:

[0313] $\text{if}(\text{rect_tile_group_flag}) \{$

[0314] $\text{NumTilesInCurrTileGroup} = \text{NumTilesInTileGroup}[\text{tile_group_address}]$

[0315] $\text{tileIdx} = \text{top_left_tile_idx}[\text{tile_group_address}]$

[0316] $\text{for}(j = 0, \text{tIdx} = 0; j <$

[0317] $\text{NumTileRowsInTileGroupMinus1}[\text{tile_group_address}] + 1; \text{ (7-35)}$

[0318] $j++, \text{tileIdx} += \text{num_tile_columns_minus1} + 1)$

[0319] $\text{for}(i = 0, \text{currTileIdx} = \text{tileIdx}; i <$

[0320] $\text{NumTileColumnsInTileGroupMinus1}[\text{tile_group_address}] + 1;$

[0321] $i++, \text{currTileIdx}++, \text{tIdx}++)$

[0322] $\text{TgTileIdx}[\text{tIdx}] = \text{currTileIdx}$

[0323] } else {

[0324] NumTilesInCurrTileGroup = num_tiles_in_tile_group_minus1 + 1

[0325] TgTileIdx[0] = tile_group_address

[0326] for(i = 1; i < NumTilesInCurrTileGroup; i++)

[0327] TgTileIdx[i] = TgTileIdx[i - 1] + 1

[0328] }

[0329] tile_group_type는 타일 그룹의 코딩 타입을 특정한다.

[0330] 시간 루마 모션 벡터 예측을 위한 예시적인 도출 프로세스는 다음과 같다. 변수들 mvLXCol 및 availableFlagLXCol은 다음과 같이 도출된다: tile_group_temporal_mvp_enabled_flag가 0이면, mvLXCol의 양쪽 성분들은 0으로 설정되고 availableFlagLXCol은 0으로 설정된다. 그렇지 않으면 (tile_group_temporal_mvp_enabled_flag가 1이면), 다음의 순서화된 단계들이 적용된다: 우측 하단 동위치의 모션 벡터, 및 하단 및 우측 경계 샘플 위치들은 다음과 같이 도출된다:

[0331] xColBr = xCb + cbWidth (8-414)

[0332] yColBr = yCb + cbHeight (8-415)

[0333] rightBoundaryPos = sub_pic_treated_as_pic_flag[SubPicIdx[tile_group_subpic_id]] ?

[0334] SubPicRightBoundaryPos : pic_width_in_luma_samples - 1 (8-415)

[0335] botBoundaryPos = sub_pic_treated_as_pic_flag[SubPicIdx[tile_group_subpic_id]] ?

[0336] SubPicBotBoundaryPos : pic_height_in_luma_samples - 1 (8-415)

[0337] yCb >> CtbLog2SizeY가 yColBr >> CtbLog2SizeY와 동일하고, yColBr이 botBoundaryPos 이하이고, xColBr이 rightBoundaryPos 이하인 경우, 다음과 같이 적용된다: 변수 colCb는 ColPic에 의해 특정된 동위치의 픽처 내에서 ((xColBr >> 3) << 3, (yColBr >> 3) << 3)에 의해 주어진 수정된 위치를 커버하는 루마 코딩 블록을 특정한다. 루마 위치 (xColCb, yColCb)는 ColPic에 의해 특정된 동위치의 픽처의 좌측 상단 루마 샘플에 대해 colCb에 의해 특정된 동위치의 루마 코딩 블록의 좌측 상단 샘플과 동등하게 설정된다. 동위치의 모션 벡터들에 대한 도출 프로세스는 currCb, colCb, (xColCb, yColCb), refIdxLX, 및 0으로 설정된 sbFlag를 입력들로 하여 호출되고, 출력은 mvLXCol 및 availableFlagLXCol에 할당된다. 그렇지 않으면, mvLXCol의 양쪽 성분들은 0으로 설정되고 availableFlagLXCol은 0으로 설정된다.

[0338] 예시적인 루마 샘플 이중선형 보간 프로세스는 다음과 같다. 전체-샘플 유닛들(xInti, yInti)에서의 루마 위치들은 i=0..1에 대해 다음과 같이 도출된다. sub_pic_treated_as_pic_flag[SubPicIdx[tile_group_subpic_id]]가 1인 경우, 다음이 적용된다:

[0339] xInti = Clip3(SubPicLeftBoundaryPos, SubPicRightBoundaryPos, xIntL + i) (8-458)

[0340] yInti = Clip3(SubPicTopBoundaryPos, SubPicBotBoundaryPos, yIntL + i) (8-458)

[0341] 그렇지 않으면(sub_pic_treated_as_pic_flag[SubPicIdx[tile_group_subpic_id]]가 0인 경우), 다음이 적용된다:

[0342] xInti = sps_ref_wraparound_enabled_flag ?

[0343]

ClipH((sps_ref_wraparound_offset_minus1 + 1) * MinCbSizeY, picW, (xIntL + i)) : (8-459)

[0344] Clip3(0, picW - 1, xIntL + i)

[0345] yInti = Clip3(0, picH - 1, yIntL + i) (8-460)

[0346] 예시적인 루마 샘플 8-탭 보간 필터링 프로세스는 다음과 같다. 전체-샘플 유닛들(xInti, yInti)에서의 루마 위치들은 i=0..7에 대해 다음과 같이 도출된다.

sub_pic_treated_as_pic_flag[SubPicIdx[tile_group_subpic_id]]가 1이면, 다음이 적용된다:

[0347] $xInt_i = Clip3(SubPicLeftBoundaryPos, SubPicRightBoundaryPos, xIntL + i - 3)$
(8-830)

[0348] $yInt_i = Clip3(SubPicTopBoundaryPos, SubPicBotBoundaryPos, yIntL + i - 3)$
(8-830)

[0349] 그렇지 않으면(sub_pic_treated_as_pic_flag[SubPicIdx[tile_group_subpic_id]]가 0인 경우), 다음이 적용된다:

[0350] $xInt_i = sps_ref_wraparound_enabled_flag ?$

[0351] $ClipH((sps_ref_wraparound_offset_minus1 + 1) * MinCbSizeY, picW, xIntL + i - 3) :$
(8-831)

[0352] $Clip3(0, picW - 1, xIntL + i - 3)$

[0353] $yInt_i = Clip3(0, picH - 1, yIntL + i - 3)$ (8-832)

[0354] 예시적인 크로마 샘플 보간 프로세스는 다음과 같다. 변수 xOffset은 $(sps_ref_wraparound_offset_minus1 + 1) * MinCbSizeY / SubWidthC$ 로 설정된다. 전체-샘플 유닛들(xInt_i, yInt_i)에서의 크로마 위치들은 i=0..3에 대해 다음과 같이 도출된다:

[0355] sub_pic_treated_as_pic_flag[SubPicIdx[tile_group_subpic_id]]가 1이면, 다음이 적용된다:

[0356] $xInt_i = Clip3(SubPicLeftBoundaryPos / SubWidthC, SubPicRightBoundaryPos / SubWidthC, xIntL + i)$
(8-844)

[0357] $yInt_i = Clip3(SubPicTopBoundaryPos / SubHeightC, SubPicBotBoundaryPos / SubHeightC, yIntL + i)$
(8-844)

[0358] 그렇지 않으면(sub_pic_treated_as_pic_flag[SubPicIdx[tile_group_subpic_id]]가 0인 경우), 다음이 적용된다:

[0359] $xInt_i = sps_ref_wraparound_enabled_flag ? ClipH(xOffset, picWC, xIntC + i - 1) :$
(8-845)

[0360] $Clip3(0, picWC - 1, xIntC + i - 1)$

[0361] $yInt_i = Clip3(0, picHC - 1, yIntC + i - 1)$ (8-846)

[0362] 예시적인 디블록킹 필터 프로세스는 다음과 같다. 디블록킹 필터 프로세스는 다음의 타입의 에지들을 제외하고, 픽처의 모든 코딩 서브-블록 에지들 및 변환 블록 에지들에 적용된다: 픽처의 경계에 있는 에지들; loop_filter_across_sub_pic_enabled_flag가 0인 서브-픽처의 경계와 일치하는 에지들; loop_filter_across_tiles_enabled_flag가 0인 타일들의 경계들과 일치하는 에지들; 1과 동일한 tile_group_deblocking_filter_disabled_flag를 갖는 타일 그룹들의 상부 또는 좌측 경계들과 일치하거나 타일 그룹들 내에 있는 에지들; 고려된 성분의 8x8 샘플 그리드 경계들에 대응하지 않는 에지들; 에지의 양측이 인터 예측을 사용하는 크로마 성분들 내의 에지들; 연관된 변환 유닛의 에지들이 아닌 크로마 변환 블록들의 에지들; 및 ISP_NO_SPLIT와 동일하지 않은 IntraSubPartitionsSplit 값을 갖는 코딩 유닛의 루마 변환 블록들에 걸친 에지들.

[0363] 한 방향에 대한 예시적인 디블록킹 필터 프로세스는 다음과 같다. 코딩 블록 폭 log2CbW, 코딩 블록 높이 log2CbH 및 코딩 블록의 좌측 상단 샘플의 위치(xCb, yCb)를 갖는 각각의 코딩 유닛에 대해, edgeType가 EDGE_VER과 동일하고 xCb % 8이 0일 때 또는 edgeType가 EDGE_HOR과 동일하고 yCb % 8이 0일 때, 에지들은 다음의 순서화된 단계들에 의해 필터링된다. 코딩 블록 폭 nCbW는 $1 < \log2CbW$ 로 설정되고 코딩 블록 높이 nCbH는 $1 < \log2CbH$ 로 설정된다. 변수 filterEdgeFlag는 다음과 같이 도출된다. edgeType가 EDGE_VER과 동일하고 다음의 조건들 중 하나 이상이 참이면, filterEdgeFlag는 0으로 설정된다. 현재 코딩 블록의 좌측 경계는 픽처의 좌측 경계이다. 현재 코딩 블록의 좌측 경계는 서브-픽처의 좌측 또는 우측 경계이고, loop_filter_across_sub_pic_enabled_flag는 0과 동일하다. 현재 코딩 블록의 좌측 경계는 타일의 좌측 경계

이고, loop_filter_across_tiles_enabled_flag는 0과 동일하다. 그렇지 않고 edgeType가 EDGE_HOR과 동일하고 다음의 조건들 중 하나 이상이 참이면, 변수 filterEdgeFlag는 0으로 설정된다. 현재 루마 코딩 블록의 상단 경계는 픽처의 상단 경계이다. 현재 코딩 블록의 상단 경계는 서브-픽처의 상단 또는 하단 경계이고, loop_filter_across_sub_pic_enabled_flag는 0과 동일하다. 현재 코딩 블록의 상단 경계는 타일의 상단 경계이고, loop_filter_across_tiles_enabled_flag는 0과 동일하다. 그렇지 않으면, filterEdgeFlag는 1로 설정된다.

[0364] 예시적인 CTB 수정 프로세스는 다음과 같다. recPicture[xSi][ySj]를 커버하는 코딩 블록을 포함하는 코딩 유닛의 pcm_loop_filter_disabled_flag, pcm_flag [xYi][yYj] 및 cu_transquant_bypass_flag의 값들에 따라, $i=0..nCtbSw-1$ 및 $j=0..nCtbSh-1$ 인 모든 샘플 위치들(xSi, ySj) 및 (xYi, yYj)에 대해, 이하가 적용된다. $k=0..1$ 인 모든 샘플 위치들(xSik', ySjk') 및 (xYik', yYjk')에 대한 다음의 조건들 중 하나 이상이 참이면, edgeIdx는 0으로 설정된다. 위치(xSik', ySjk')에 있는 샘플은 픽처 경계 밖에 있다. 위치(xSik', ySjk')에 있는 샘플은 상이한 서브-픽처에 속하고, 샘플 recPicture[xSi][ySj]가 속하는 타일 그룹 내의 loop_filter_across_sub_pic_enabled_flag는 0과 동일하다. loop_filter_across_tiles_enabled_flag는 0과 동일하고 위치(xSik', ySjk')에 있는 샘플은 상이한 타일에 속한다.

[0365] 루마 샘플들에 대한 예시적인 코딩 트리 블록 필터링 프로세스는 다음과 같다. 필터링된 재구성된 루마 샘플들 alfPictureL[x][y]의 도출을 위해, 현재 루마 코딩 트리 블록 recPictureL[x][y] 내의 각각의 재구성된 루마 샘플은 다음과 같이 필터링되며, 단, $x, y=0..CtbSizeY-1$ 이다. 루마 샘플들의 주어진 어레이 recPicture 내의 대응하는 루마 샘플들(x, y) 각각에 대한 위치들(hx, vy)은 다음과 같이 도출된다. 위치(hx, vy)에 있는 루마 샘플을 포함하는 타일 tileA에 대한 loop_filter_across_tiles_enabled_flag가 0과 동일한 경우, 변수 tileIdx를 tileA의 타일 인덱스라고 하고, 다음이 적용된다:

[0366]
$$hx = Clip3(TileLeftBoundaryPos[tileIdx], TileRightBoundaryPos[tileIdx], xCtb + x)$$
 (8-1140)

[0367]
$$vy = Clip3(TileTopBoundaryPos[tileIdx], TileBotBoundaryPos[tileIdx], yCtb + y)$$
 (8-1141)

[0368] 그렇지 않고, 위치(hx, vy)에 있는 루마 샘플을 포함하는 서브-픽처 내의 loop_filter_across_sub_pic_enabled_flag가 0인 경우, 다음이 적용된다:

[0369]
$$hx = Clip3(SubPicLeftBoundaryPos, SubPicRightBoundaryPos, xCtb + x)$$
 (8-1140)

[0370]
$$vy = Clip3(SubPicTopBoundaryPos, SubPicBotBoundaryPos, yCtb + y)$$
 (8-1141)

[0371] 그렇지 않으면, 다음이 적용된다:

[0372]
$$hx = Clip3(0, pic_width_in_luma_samples - 1, xCtb + x)$$
 (8-1140)

[0373]
$$vy = Clip3(0, pic_height_in_luma_samples - 1, yCtb + y)$$
 (8-1141)

[0374] 루마 샘플들에 대한 ALF 전치(transpose) 및 필터 인덱스에 대한 예시적인 도출 프로세스는 다음과 같다. 루마 샘플들의 주어진 어레이 recPicture 내의 대응하는 루마 샘플들(x, y) 각각에 대한 위치들(hx, vy)은 다음과 같이 도출된다. 위치(hx, vy)에 있는 루마 샘플을 포함하는 타일 tileA에 대한 loop_filter_across_tiles_enabled_flag가 0이면, tileIdx를 tileA의 타일 인덱스라고 하고, 다음이 적용된다:

[0375]
$$hx = Clip3(TileLeftBoundaryPos[tileIdx], TileRightBoundaryPos[tileIdx], x)$$
 (8-1140)

[0376]
$$vy = Clip3(TileTopBoundaryPos[tileIdx], TileBotBoundaryPos[tileIdx], y)$$
 (8-1141)

[0377] 그렇지 않고, 위치(hx, vy)에 있는 루마 샘플을 포함하는 서브-픽처에 대한 loop_filter_across_sub_pic_enabled_flag가 0인 경우, 다음이 적용된다:

[0378]
$$hx = Clip3(SubPicLeftBoundaryPos, SubPicRightBoundaryPos, x)$$
 (8-1140)

[0379] $vy = \text{Clip3}(\text{SubPicTopBoundaryPos}, \text{SubPicBotBoundaryPos}, y)$ (8-1141)

[0380] 그렇지 않으면, 다음이 적용된다:

[0381] $hx = \text{Clip3}(0, \text{pic_width_in_luma_samples} - 1, x)$ (8-1145)

[0382] $vy = \text{Clip3}(0, \text{pic_height_in_luma_samples} - 1, y)$ (8-1146)

[0383] 크로마 샘플들에 대한 예시적인 코딩 트리 블록 필터링 프로세스는 다음과 같다. 필터링된 재구성된 크로마 샘플들 $\text{alfPicture}[x][y]$ 의 도출을 위해, 현재 크로마 코딩 트리 블록 $\text{recPicture}[x][y]$ 내의 각각의 재구성된 크로마 샘플은 다음과 같이 필터링되며, 단, $x, y = 0 \dots \text{ctbSizeC} - 1$ 이다. 크로마 샘플들의 주어진 어레이 recPicture 내의 대응하는 크로마 샘플들(x, y) 각각에 대한 위치들(hx, vy)는 다음과 같이 도출된다. 위치(hx, vy)에 있는 크로마 샘플을 포함하는 타일 tileA 에 대한 $\text{loop_filter_across_tiles_enabled_flag}$ 가 0인 경우, tileIdx 를 tileA 의 타일 인덱스라고 하고, 다음이 적용된다:

[0384] $hx = \text{Clip3}(\text{TileLeftBoundaryPos}[\text{tileIdx}] / \text{SubWidthC},$

[0385] $\text{TileRightBoundaryPos}[\text{tileIdx}] / \text{SubWidthC}, \text{xCtb} + x)$ (8-1140)

[0386] $vy = \text{Clip3}(\text{TileTopBoundaryPos}[\text{tileIdx}] / \text{SubWidthC},$

[0387] $\text{TileBotBoundaryPos}[\text{tileIdx}] / \text{SubWidthC}, \text{yCtb} + y)$ (8-1141)

[0388] 그렇지 않고, 위치(hx, vy)에 있는 크로마 샘플을 포함하는 서브-픽처에 대한 $\text{loop_filter_across_sub_pic_enabled_flag}$ 가 0인 경우, 다음이 적용된다:

[0389] $hx = \text{Clip3}(\text{SubPicLeftBoundaryPos} / \text{SubWidthC},$

[0390] $\text{SubPicRightBoundaryPos} / \text{SubWidthC}, \text{xCtb} + x)$ (8-1140)

[0391] $vy = \text{Clip3}(\text{SubPicTopBoundaryPos} / \text{SubWidthC},$

[0392] $\text{SubPicBotBoundaryPos} / \text{SubWidthC}, \text{yCtb} + y)$ (8-1141)

[0393] 그렇지 않으면, 다음이 적용된다:

[0394] $hx = \text{Clip3}(0, \text{pic_width_in_luma_samples} / \text{SubWidthC} - 1, \text{xCtbC} + x)$ (8-1177)

[0395] $vy = \text{Clip3}(0, \text{pic_height_in_luma_samples} / \text{SubHeightC} - 1, \text{yCtbC} + y)$

[0396] (8-1178)

[0397] 변수 sum 은 다음과 같이 도출된다:

[0398] $\text{sum} = \text{AlfCoeffC}[0] * (\text{recPicture}[hx, vy + 2] + \text{recPicture}[hx, vy - 2]) +$

[0399] $\text{AlfCoeffC}[1] * (\text{recPicture}[hx + 1, vy + 1] + \text{recPicture}[hx - 1, vy - 1]) +$

[0400] $\text{AlfCoeffC}[2] * (\text{recPicture}[hx, vy + 1] + \text{recPicture}[hx, vy - 1]) +$ (8-1179)

[0401] $\text{AlfCoeffC}[3] * (\text{recPicture}[hx - 1, vy + 1] + \text{recPicture}[hx + 1, vy - 1]) +$

[0402] $\text{AlfCoeffC}[4] * (\text{recPicture}[hx + 2, vy] + \text{recPicture}[hx - 2, vy]) +$

[0403] $\text{AlfCoeffC}[5] * (\text{recPicture}[hx + 1, vy] + \text{recPicture}[hx - 1, vy]) +$

[0404] $\text{AlfCoeffC}[6] * \text{recPicture}[hx, vy]$

[0405] $\text{sum} = (\text{sum} + 64) \gg 7$ (8-1180)

[0406] 수정된 필터링된 재구성된 크로마 픽처 샘플 $\text{alfPicture}[\text{xCtbC} + x][\text{yCtbC} + y]$ 는 다음과 같이 도출된다:

[0407] $\text{alfPicture}[\text{xCtbC} + x][\text{yCtbC} + y] = \text{Clip3}(0, (1 \ll \text{BitDepthC}) - 1, \text{sum})$ (8-1181)

[0408] 도 12는 예시적인 비디오 코딩 디바이스(1200)의 개략도이다. 비디오 코딩 디바이스(1200)는 본 명세서에서 설

명된 바와 같은 개시된 예들/실시예들을 구현하기에 적합하다. 비디오 코딩 디바이스(1200)는 다운스트림 포트들(1220), 업스트림 포트들(1250), 및/또는 네트워크를 통해 업스트림 및/또는 다운스트림으로 데이터를 통신하기 위한 송신기들 및/또는 수신기들을 포함하는 송수신기 유닛들(Tx/Rx)(1210)을 포함한다. 비디오 코딩 디바이스(1200)는 데이터를 처리하기 위한 로직 유닛 및/또는 중앙 처리 유닛(CPU)을 포함하는 프로세서(1230) 및 데이터를 저장하기 위한 메모리(1232)를 또한 포함한다. 비디오 코딩 디바이스(1200)는 광학 또는 무선 통신 네트워크들을 통한 데이터의 통신을 위해 업스트림 포트들(1250) 및/또는 다운스트림 포트들(1220)에 결합되는 OE(optical-to-electrical) 컴포넌트들, EO(electrical-to-optical) 컴포넌트들, 및/또는 무선 통신 컴포넌트들을 또한 포함할 수 있다. 비디오 코딩 디바이스(1200)는 사용자에게 그리고 사용자로부터 데이터를 통신하기 위한 입력 및/또는 출력(I/O) 디바이스들(1260)을 또한 포함할 수 있다. I/O 디바이스들(1260)은 비디오 데이터를 디스플레이하기 위한 디스플레이, 오디오 데이터를 출력하기 위한 스피커들 등과 같은 출력 디바이스들을 포함할 수 있다. I/O 디바이스들(1260)은 키보드, 마우스, 트랙볼 등과 같은 입력 디바이스들 및/또는 그러한 출력 디바이스들과 상호작용하기 위한 대응하는 인터페이스들을 또한 포함할 수 있다.

[0409] 프로세서(1230)는 하드웨어 및 소프트웨어에 의해 구현된다. 프로세서(1230)는, 하나 이상의 CPU 칩, 코어(예를 들어, 멀티 코어 프로세서), FPGA(field-programmable gate array), ASIC(application specific integrated circuit) 및 DSP(digital signal processor)로서 구현될 수 있다. 프로세서(1230)는 다운스트림 포트들(1220), Tx/Rx(1210), 업스트림 포트들(1250), 및 메모리(1232)와 통신 상태에 있다. 프로세서(1230)는 코딩 모듈(1214)을 포함한다. 코딩 모듈(1214)은, 패턴(900)에 따라 생성된 후보 리스트에 기초하여 단방향 인터-예측(600) 및/또는 양방향 인터-예측(700)에 따라 코딩될 수 있는 인-루프 필터(1000), 비트스트림(1100), 픽처(500), 및/또는 현재 블록(801 및/또는 901)을 이용할 수 있는, 방법(100, 1300 및 1400) 등의, 본 명세서에서 설명된 개시된 실시예들을 구현한다. 코딩 모듈(1214)은 또한 본 명세서에서 설명된 임의의 다른 방법/메커니즘을 구현할 수 있다. 또한, 코딩 모듈(1214)은 코덱 시스템(200), 인코더(300), 및/또는 디코더(400)를 구현할 수 있다. 예를 들어, 코딩 모듈(1214)은 전술한 바와 같은 제1, 제2, 제3, 제4, 제5 및/또는 제6 예시적인 구현을 구현할 수 있다. 따라서, 코딩 모듈(1214)은 비디오 코딩 디바이스(1200)로 하여금 비디오 데이터를 코딩할 때 추가적인 기능성 및/또는 코딩 효율을 제공하게 한다. 이와 같이, 코딩 모듈(1214)은 비디오 코딩 디바이스(1200)의 기능성을 개선할 뿐만 아니라 비디오 코딩 기술분야에 특징적인 문제들을 다룬다. 또한, 코딩 모듈(1214)은 비디오 코딩 디바이스(1200)의 상이한 상태로의 변환을 달성한다. 대안적으로, 코딩 모듈(1214)은 메모리(1232)에 저장되고 프로세서(1230)에 의해 실행되는 명령어들로서(예를 들어, 비일시적 매체상에 저장된 컴퓨터 프로그램 제품으로서) 구현될 수 있다.

[0410] 메모리(1232)는 디스크들, 테이프 드라이브들, 솔리드-스테이트 드라이브들, ROM(read only memory), RAM(random access memory), 플래시 메모리, TCAM(ternary content-addressable memory), SRAM(static random-access memory) 등과 같은 하나 이상의 메모리 타입을 포함한다. 메모리(1232)는 오버-플로우 데이터 저장 디바이스로서, 그러한 프로그램들이 실행을 위해 선택될 때 프로그램들을 저장하고, 프로그램 실행 동안 판독되는 명령어들 및 데이터를 저장하기 위해 사용될 수 있다.

[0411] 도 13은 서브-픽처(510)와 같은 서브-픽처에 대해 슬라이스(515)와 같은 슬라이스를 어드레싱함으로써 비디오 시퀀스를 비트스트림(1100)과 같은 비트스트림으로 인코딩하는 예시적인 방법(1300)의 흐름도이다. 방법(1300)은, 인-루프 필터(1000)를 이용함으로써 단방향 인터-예측(600) 및/또는 양방향 인터-예측(700)에 따라 및/또는 패턴(900)에 따라 생성된 후보 리스트에 관해 현재 블록(801 및/또는 901)을 인코딩하기 위해 방법(100)을 수행할 때 코덱 시스템(200), 인코더(300), 및/또는 비디오 코딩 디바이스(1200) 등의 인코더에 의해 이용될 수 있다.

[0412] 방법(1300)은 인코더가 복수의 픽처를 포함하는 비디오 시퀀스를 수신하고, 예를 들어, 사용자 입력에 기초하여 해당 비디오 시퀀스를 비트스트림 내로 인코딩하기로 결정할 때 시작될 수 있다. 단계 1301에서, 인코더는 픽처를 서브-픽처로 그리고 서브-픽처를 슬라이스로 파티셔닝한다.

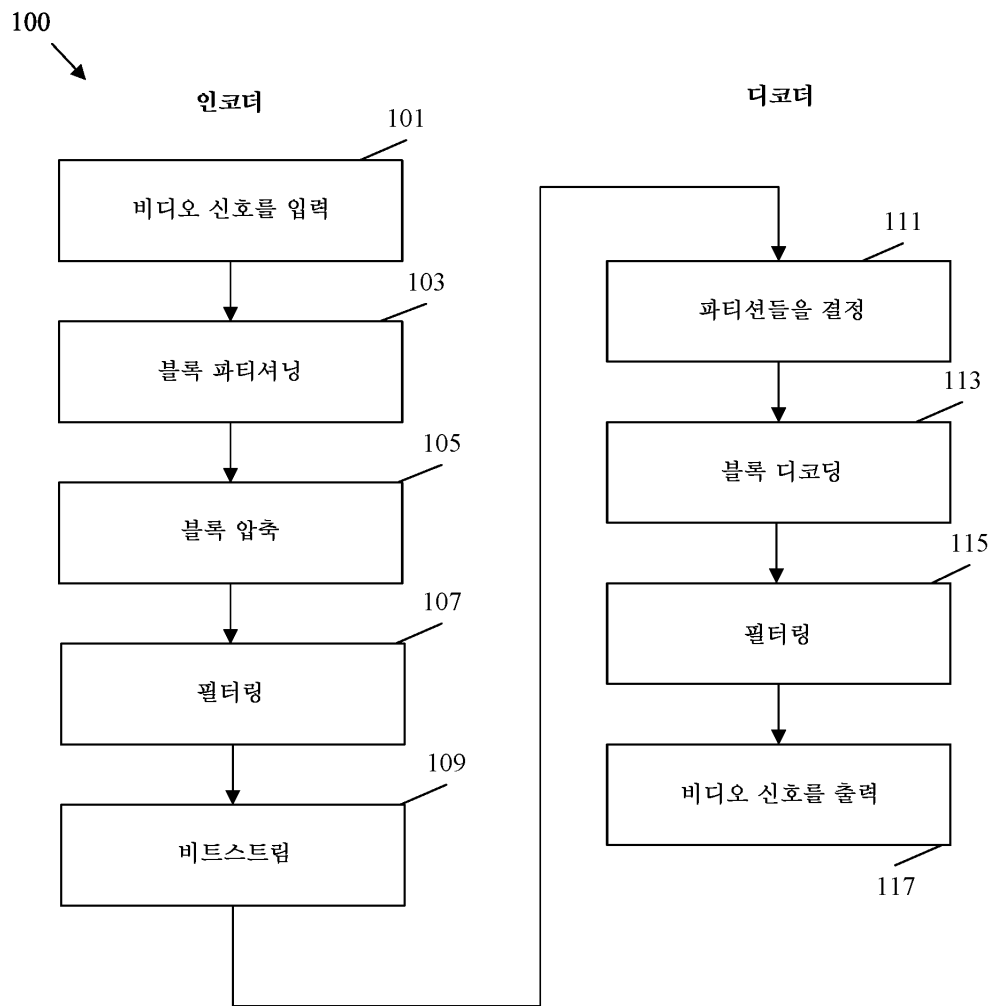
[0413] 단계 1303에서, 인코더는 슬라이스에 대한 슬라이스 어드레스를 결정한다. 구체적으로, 슬라이스 어드레스는 슬라이스가 직사각형 또는 정사각형 슬라이스일 때 슬라이스의 서브-픽처 레벨 슬라이스 인덱스를 포함한다. 서브-픽처 레벨 인덱스는 픽처 내의 슬라이스의 위치에 기초하는 대신에 서브-픽처 내의 슬라이스의 위치에 기초하는 인덱스이다. 또한, 인코더는 슬라이스를 포함하는 서브-픽처의 서브-픽처 ID를 결정한다. 일부 예들에서, rect_slice_flag가 1로 설정될 때 슬라이스는 직사각형 또는 정사각형 슬라이스일 수 있다. rect_slice_flag는 SPS 및/또는 PPS로 인코딩될 수 있다.

- [0414] 단계 1305에서, 인코더는 슬라이스 어드레스 및 서브-픽처 ID를 비트스트림으로 인코딩한다. 예를 들어, 인코더는 슬라이스 헤더를 비트스트림으로 인코딩할 수 있다. 슬라이스 헤더는 슬라이스와 연관되고, 슬라이스를 인코딩/디코딩하는데 이용되는 파라미터들을 포함한다. 예를 들어, 슬라이스는 슬라이스 헤더를 참조할 수 있다. 슬라이스 어드레스 및 서브-픽처 ID는 비트스트림 내의 슬라이스 헤더로 인코딩될 수 있다. 일부 예들에서, 슬라이스 어드레스는 slice_address 변수로 인코딩되고, 서브-픽처 ID는 slice_subpic_id 변수로 인코딩된다.
- [0415] 단계 1307에서, 인코더는 서브-픽처의 슬라이스를 비트스트림으로 인코딩한다. 인코더는 또한 비트스트림에서 전체 서브-픽처 및/또는 전체 픽처를 인코딩할 수 있다. 단계 1309에서, 인코더는 또한 SPS를 비트스트림으로 인코딩할 수 있다. 예를 들어, 인코더는 다양한 서브-픽처(들)의 설명을 SPS로 인코딩할 수 있다. 서브-픽처 ID는 SPS 내의 관련 서브-픽처의 설명을 참조할 수 있다. 일부 예들에서, 서브-픽처의 설명은 CTB 단위의 서브-픽처의 폭, CTB 단위의 서브-픽처의 높이, 및 서브-픽처의 좌측 상단 CTU의 수직 위치 및 서브-픽처의 좌측 상단 CTU의 수평 위치로서 특징되는 CTB 단위의 서브-픽처의 오프셋을 포함한다. 단계 1311에서, 인코더는 디코더를 향한 통신을 위해 비트스트림을 저장한다.
- [0416] 도 14는 슬라이스(515)와 같은 슬라이스가 서브-픽처(510)와 같은 서브-픽처에 대해 어드레싱될 때 비트스트림(1100)과 같은 비트스트림으로부터 비디오 시퀀스를 디코딩하는 예시적인 방법(1400)의 흐름도이다. 방법(1400)은, 인-루프 필터(1000)를 이용함으로써 단방향 인터-예측(600) 및/또는 양방향 인터-예측(700)에 따라 및/또는 패턴(900)에 따라 생성된 후보 리스트에 관해 현재 블록(801 및/또는 901)을 디코딩하는 방법(100)을 수행할 때 코덱 시스템(200), 디코더(400), 및/또는 비디오 코딩 디바이스(1200) 등의 디코더에 의해 이용될 수 있다.
- [0417] 방법(1400)은 디코더가 예를 들어, 방법(1300)의 결과로서 비디오 시퀀스를 나타내는 코딩된 데이터의 비트스트림을 수신하기 시작할 때 시작될 수 있다. 단계 1401에서, 디코더는 비트스트림을 수신한다. 비트스트림은 SPS, PPS, 슬라이스 헤더, 및 슬라이스를 포함하는 서브-픽처를 포함할 수 있다. 일부 예들에서, 비트스트림은 또한 복수의 서브-픽처를 포함하는 전체 픽처를 포함할 수 있다.
- [0418] 단계 1403에서, 디코더는 슬라이스 어드레스를 획득한다. 슬라이스 어드레스는 슬라이스가 직사각형 또는 정사각형 슬라이스일 때 슬라이스의 서브-픽처 레벨 슬라이스 인덱스를 포함한다. 서브-픽처 레벨 인덱스는 픽처 내의 슬라이스의 위치에 기초하는 대신에 서브-픽처 내의 슬라이스의 위치에 기초하는 인덱스이다. 서브-픽처 레벨 인덱스는 픽처 내의 슬라이스의 위치에 기초하는 대신에 서브-픽처 내의 슬라이스의 위치에 기초하는 인덱스이다. 또한, 디코더는 슬라이스를 포함하는 서브-픽처의 서브-픽처 ID를 획득할 수 있다. 예를 들어, 슬라이스 헤더는 슬라이스와 연관될 수 있고, 슬라이스를 인코딩/디코딩하는데 이용되는 파라미터들을 포함할 수 있다. 예를 들어, 슬라이스는 슬라이스 헤더를 참조할 수 있다. 슬라이스 어드레스 및 서브-픽처 ID는 비트스트림 내의 슬라이스 헤더로부터 획득될 수 있다. 일부 예들에서, 슬라이스 어드레스는 slice_address 변수로 인코딩되고, 서브-픽처 ID는 slice_subpic_id 변수로 인코딩된다. 일부 예들에서, rect_slice_flag가 1로 설정될 때 슬라이스는 직사각형 또는 정사각형 슬라이스일 수 있다. rect_slice_flag는 SPS 및/또는 PPS로부터 획득될 수 있다.
- [0419] 단계 1405에서, 디코더는 SPS로부터 서브-픽처의 설명을 획득할 수 있다. 예를 들어, 서브-픽처 ID는 관련된 서브-픽처를 참조할 수 있으며, 따라서 SPS 내의 서브-픽처의 설명을 참조할 수 있다. 일부 예들에서, 서브-픽처의 설명은 CTB 단위의 서브-픽처의 폭, CTB 단위의 서브-픽처의 높이, 및 서브-픽처의 좌측 상단 CTU의 수직 위치 및 서브-픽처의 좌측 상단 CTU의 수평 위치로서 특징되는 CTB들 내의 서브-픽처의 오프셋을 포함한다.
- [0420] 단계 1407에서, 디코더는 슬라이스 어드레스, 서브-픽처 ID, 및 SPS로부터의 서브-픽처의 설명에 기초하여 서브-픽처 내에 슬라이스를 배치할 수 있다. 단계 1409에서, 디코더는 슬라이스 어드레스에 기초하여 서브-픽처의 슬라이스를 디코딩한다. 그 후, 디코더는 디코딩된 비디오 시퀀스의 부분으로서 디스플레이를 위해 슬라이스를 전달할 수 있다.
- [0421] 도 15는 슬라이스(515)와 같은 슬라이스가 서브-픽처(510)와 같은 서브-픽처에 대해 어드레싱될 때 비트스트림(1100)과 같은 비트스트림 내의 이미지들의 비디오 시퀀스를 코딩하기 위한 예시적인 시스템(1500)의 개략도이다. 시스템(1500)은 코덱 시스템(200), 인코더(300), 디코더(400), 및/또는 비디오 코딩 디바이스(1200)와 같은 인코더 및 디코더에 의해 구현될 수 있다. 또한, 시스템(1500)은, 인-루프 필터(1000)를 이용함으로써 단방향 인터-예측(600) 및/또는 양방향 인터-예측(700)에 따라 및/또는 패턴(900)에 따라 생성된 후보 리스트에 관해 현재 블록(801 및/또는 901)을 코딩하기 위해 방법(100, 1300 및/또는 1400)을 구현할 때 이용될 수 있다.

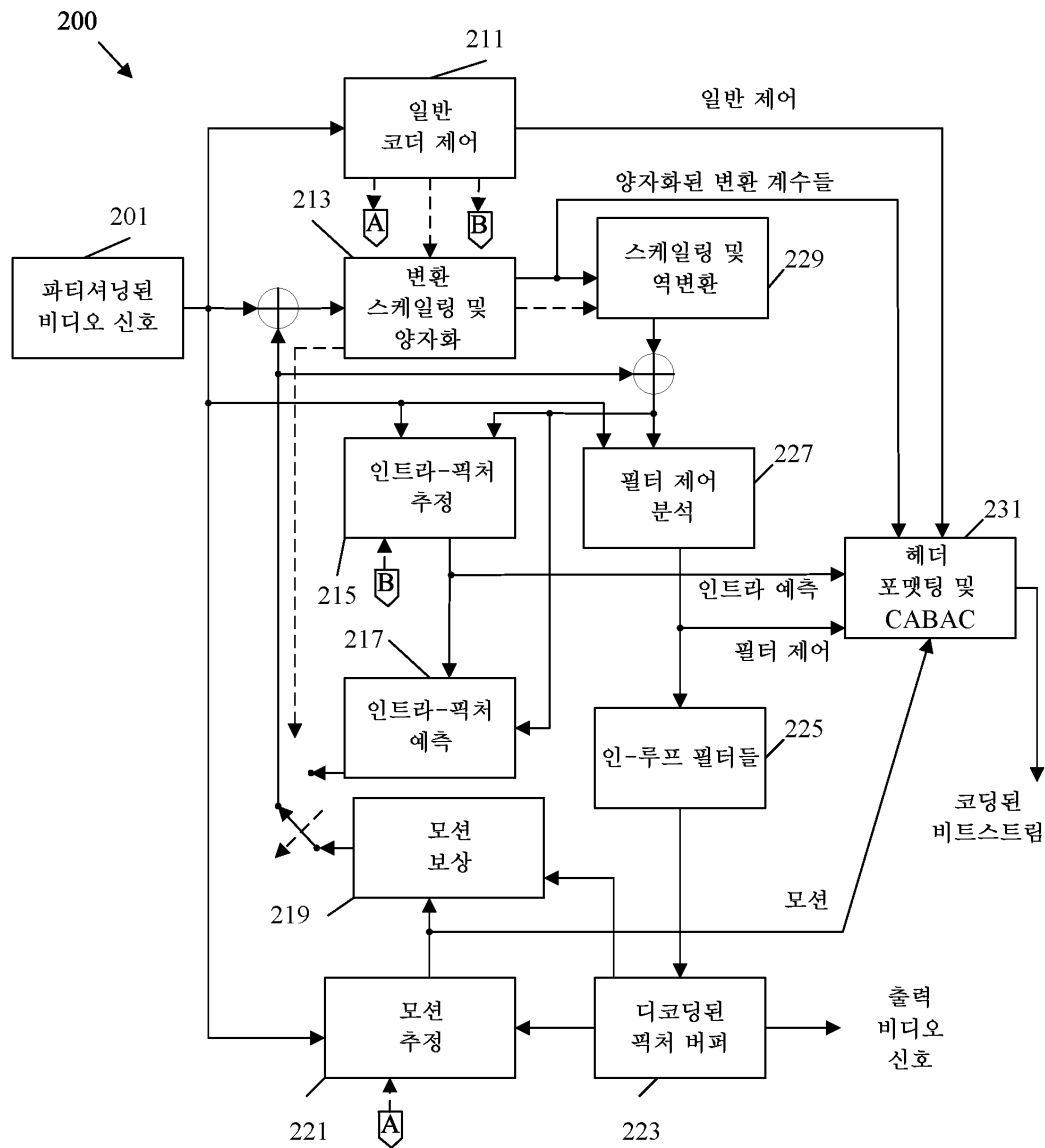
- [0422] 시스템(1500)은 비디오 인코더(1502)를 포함한다. 비디오 인코더(1502)는 서브-픽처를 슬라이스로 파티셔닝하는 파티셔닝 모듈(1505)을 포함한다. 비디오 인코더(1502)는 슬라이스가 직사각형 또는 정사각형 슬라이스일 때 슬라이스의 서브-픽처 레벨 슬라이스 인덱스를 포함하는 슬라이스 어드레스를 결정하는 결정 모듈(1506)을 추가로 포함한다. 비디오 인코더(1502)는 슬라이스 어드레스를 비트스트림으로 인코딩하기 위한 인코딩 모듈(1507)을 추가로 포함한다. 인코딩 모듈(1507)은 또한 서브-픽처의 슬라이스를 비트스트림으로 인코딩하기 위한 것이다. 비디오 인코더(1502)는 디코더를 향한 통신을 위해 비트스트림을 저장하기 위한 저장 모듈(1508)을 추가로 포함한다. 비디오 인코더(1502)는 비트스트림을 비디오 디코더(1510)를 향해 송신하기 위한 송신 모듈(1509)을 추가로 포함한다. 비디오 인코더(1502)는 방법(1300)의 단계들 중 임의의 단계를 수행하도록 추가로 구성될 수 있다.
- [0423] 시스템(1500)은 또한 비디오 디코더(1510)를 포함한다. 비디오 디코더(1510)는 슬라이스를 포함하는 서브-픽처를 포함하는 비트스트림을 수신하기 위한 수신 모듈(1511)을 포함한다. 비디오 디코더(1510)는 슬라이스가 직사각형 또는 정사각형 슬라이스일 때 슬라이스의 서브-픽처 레벨 슬라이스 인덱스를 포함하는 슬라이스 어드레스를 획득하기 위한 획득 모듈(1512)을 추가로 포함한다. 비디오 디코더(1510)는 슬라이스 어드레스에 기초하여 서브-픽처의 슬라이스를 디코딩하는 디코딩 모듈(1513)을 추가로 포함한다. 비디오 디코더(1510)는 디코딩된 비디오 시퀀스의 부분으로서 디스플레이를 위해 슬라이스를 전달하는 전달 모듈(1515)을 추가로 포함한다. 비디오 디코더(1510)는 방법(1400)의 단계들 중 임의의 것을 수행하도록 추가로 구성될 수 있다.
- [0424] 제1 컴포넌트와 제2 컴포넌트 사이에 라인, 트레이스, 또는 다른 매체를 제외하고는 개재하는 컴포넌트들이 없을 때 제1 컴포넌트는 제2 컴포넌트에 직접 결합된다. 제1 컴포넌트와 제2 컴포넌트 사이에 라인, 트레이스, 또는 다른 매체 이외의 개재하는 컴포넌트들이 있을 때 제1 컴포넌트는 제2 컴포넌트에 간접적으로 결합된다. "결합된(coupled)"이라는 용어 및 그 변형들은 직접적으로 결합되는 및 간접적으로 결합되는 둘 모두를 포함한다. "약(about)"이라는 용어의 사용은 달리 언급되지 않는 한 후속 수의 $\pm 10\%$ 를 포함하는 범위를 의미한다.
- [0425] 본 명세서에서 설명된 예시적인 방법들의 단계들은 반드시 설명된 순서로 수행될 필요는 없다는 것도 이해되어야 하고, 그러한 방법들의 단계들의 순서는 단지 예시적인 것으로 이해되어야 한다. 마찬가지로, 추가적인 단계들이 그러한 방법들에 포함될 수 있고, 본 개시내용의 다양한 실시예들과 일관된 방법들에서, 특정 단계들이 생략되거나 조합될 수 있다.
- [0426] 본 개시내용에서는 여러 실시예들이 제공되었지만, 개시된 시스템들 및 방법들은 본 개시내용의 사상 또는 범위를 벗어나지 않고 많은 다른 특정 형식들로 구현될 수 있다는 것을 이해할 수 있다. 본 예들은 제한적인 것이 아니라 예시적인 것으로 간주되어야 하고, 의도는 본 명세서에서 주어진 세부사항들로 제한되지 않는 것이다. 예를 들어, 다양한 요소들 또는 컴포넌트들이 다른 시스템에서 조합 또는 통합될 수 있거나 특정 특징들이 생략될 수 있거나, 또는 구현되지 않을 수 있다.
- [0427] 또한, 다양한 실시예들에서 개별 또는 별개로서 설명되고 예시된 기법들, 시스템들, 서브시스템들, 및 방법들은 본 개시내용의 범위를 벗어나지 않고 다른 시스템들, 컴포넌트들, 기법들, 또는 방법들과 조합 또는 통합될 수 있다. 변화들, 치환들 및 변경들의 다른 예들은 본 기술분야의 통상의 기술자에 의해 확인될 수 있고 본 명세서에서 개시된 사상 및 범위를 벗어나지 않고 이루어질 수 있다.

도면

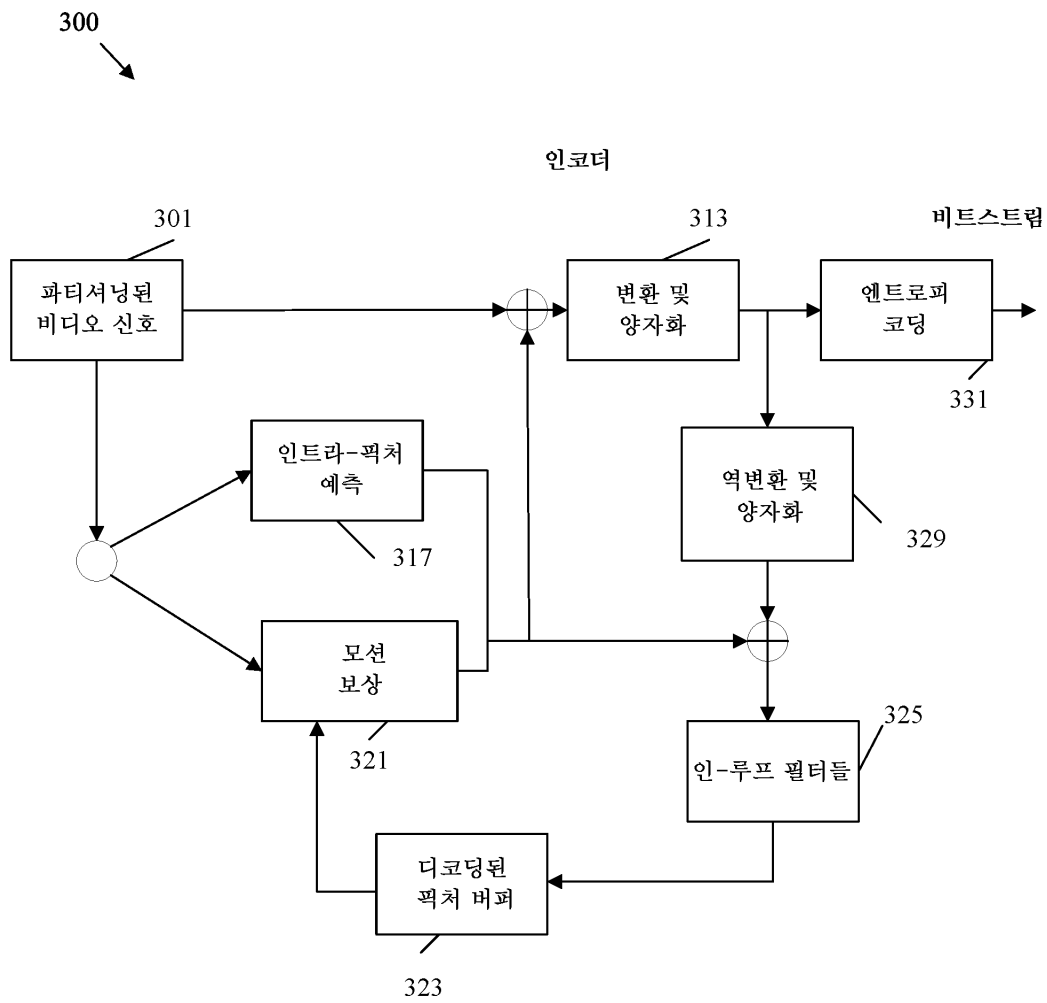
도면1



도면2

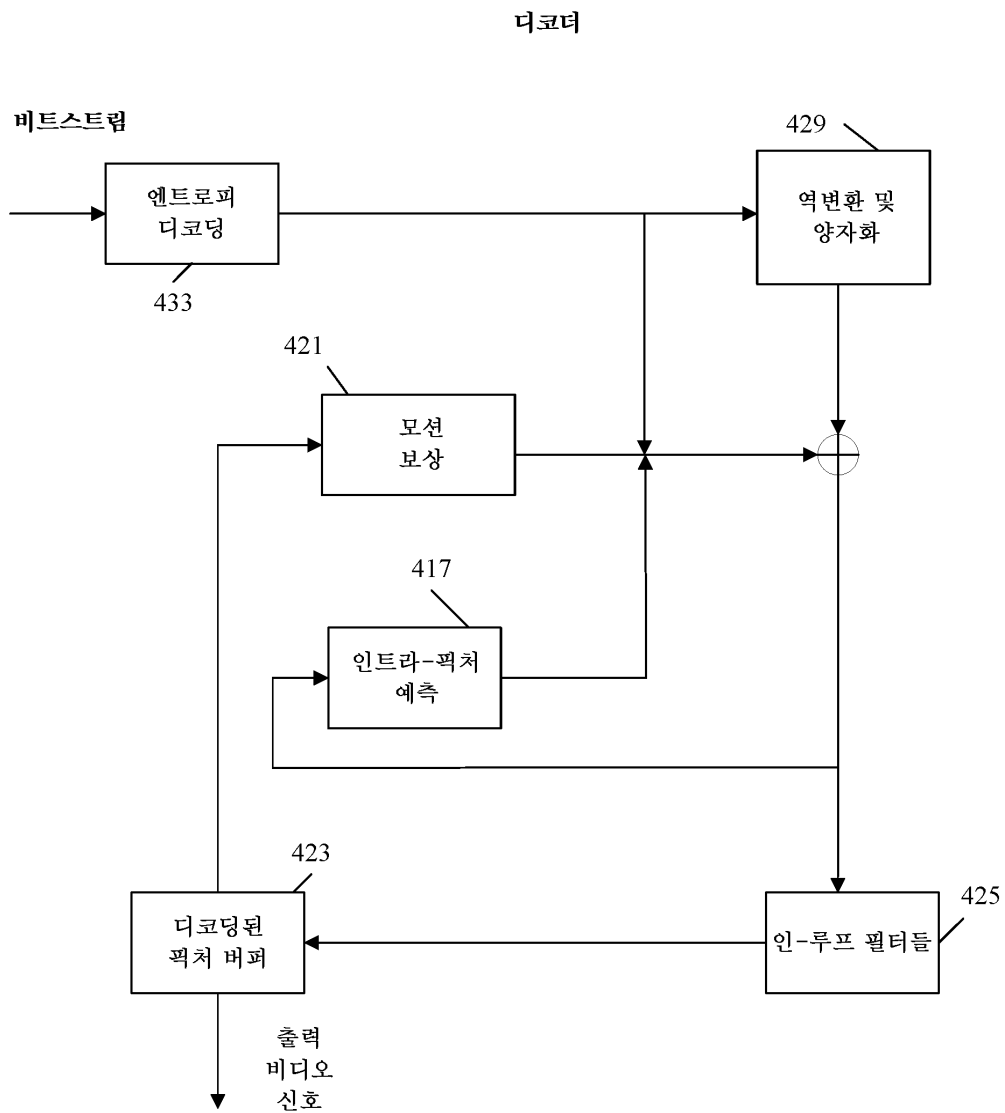


도면3

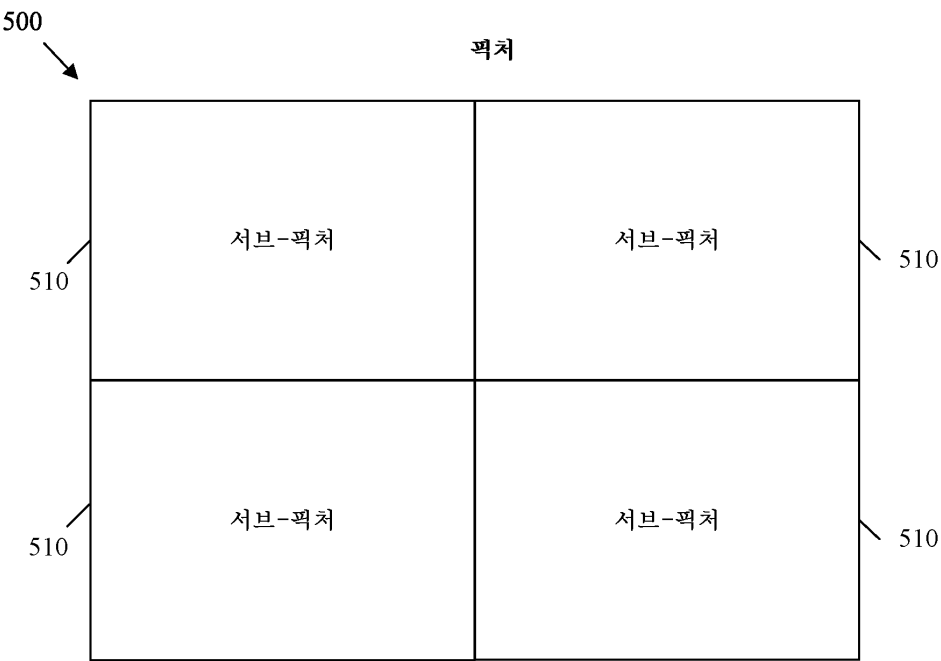


도면4

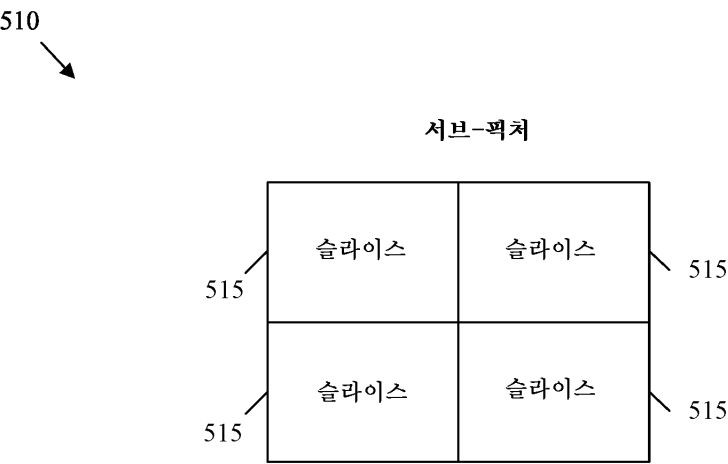
400



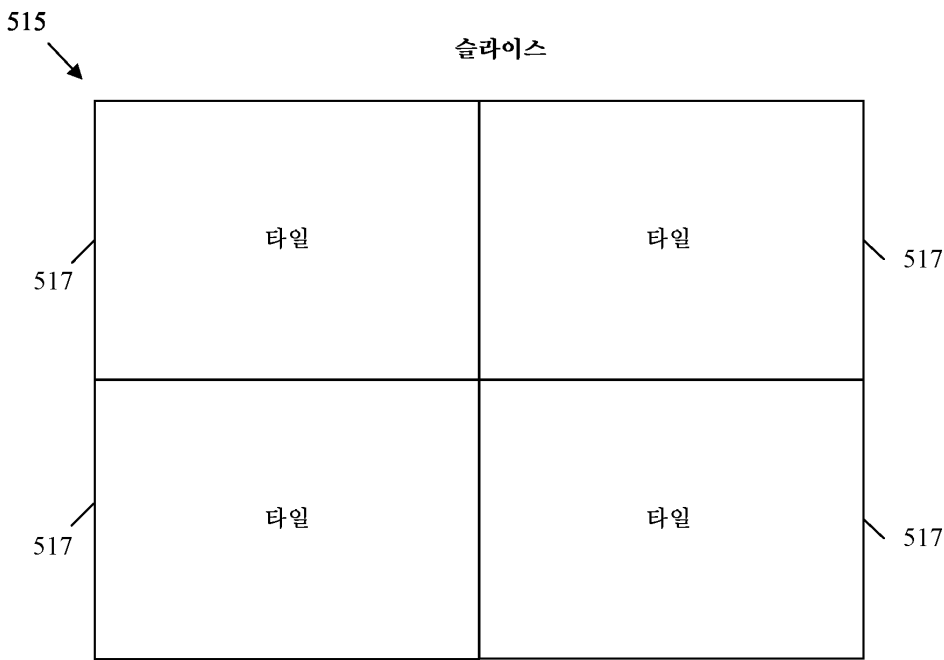
도면5a



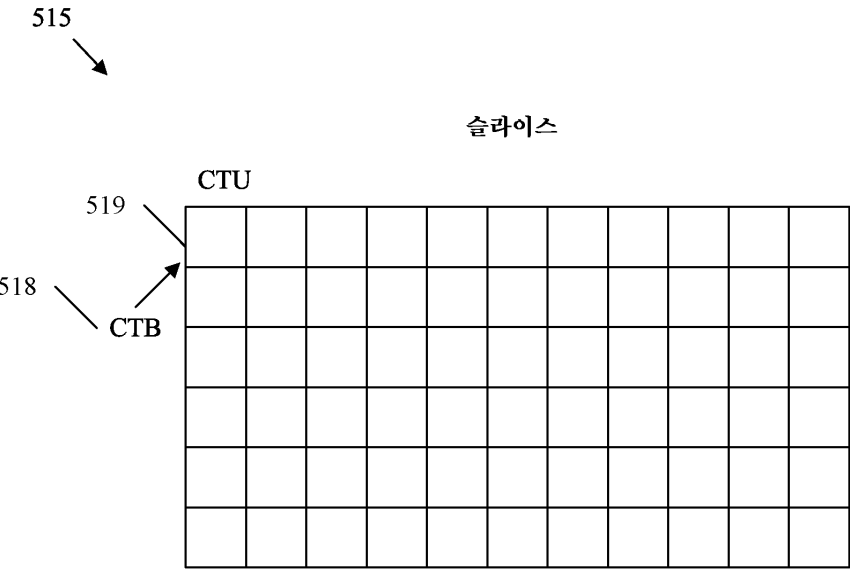
도면5b



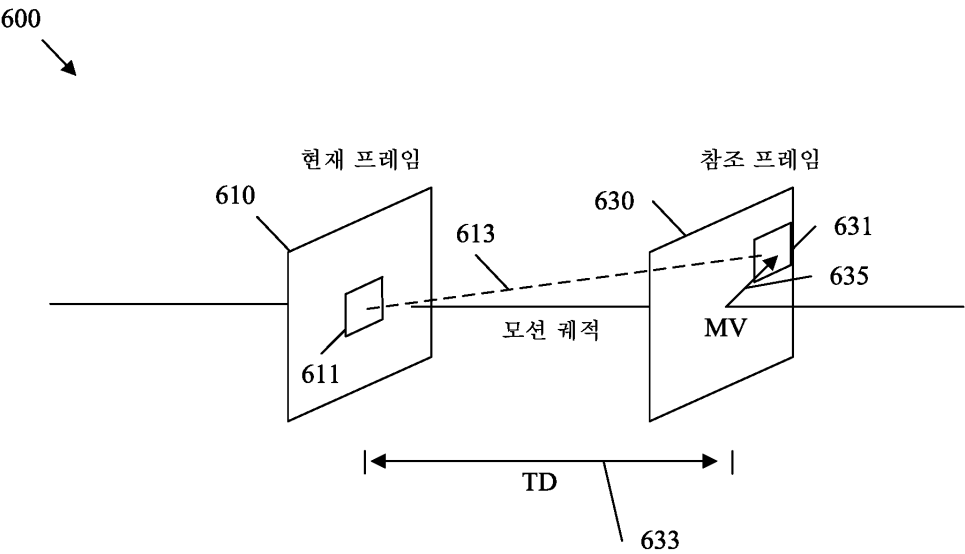
도면5c



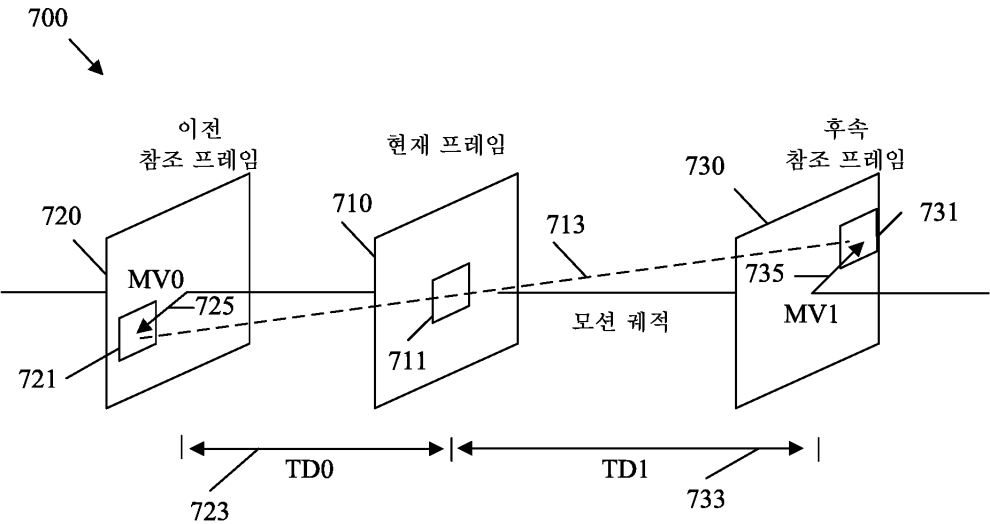
도면5d



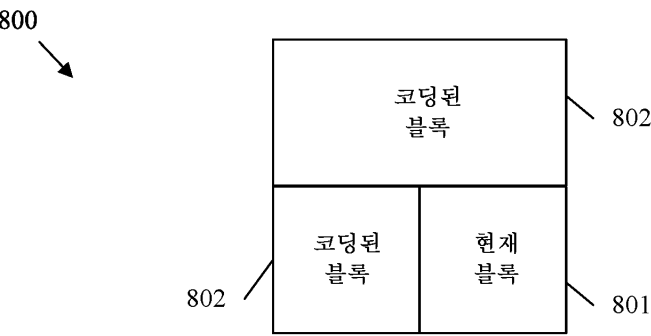
도면6



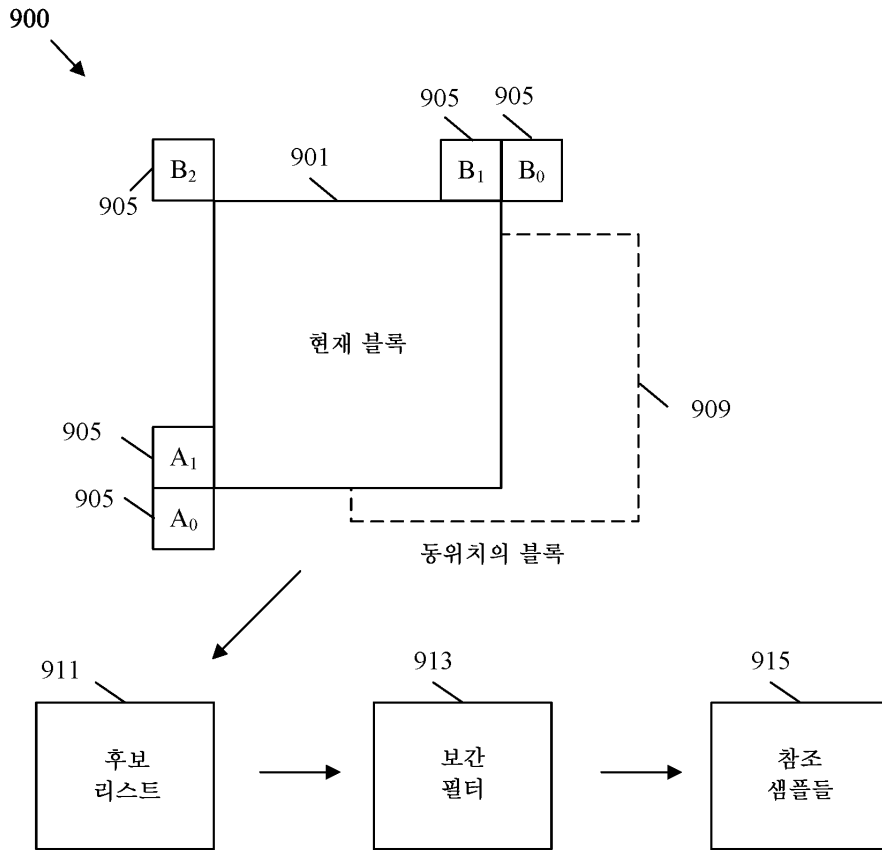
도면7



도면8

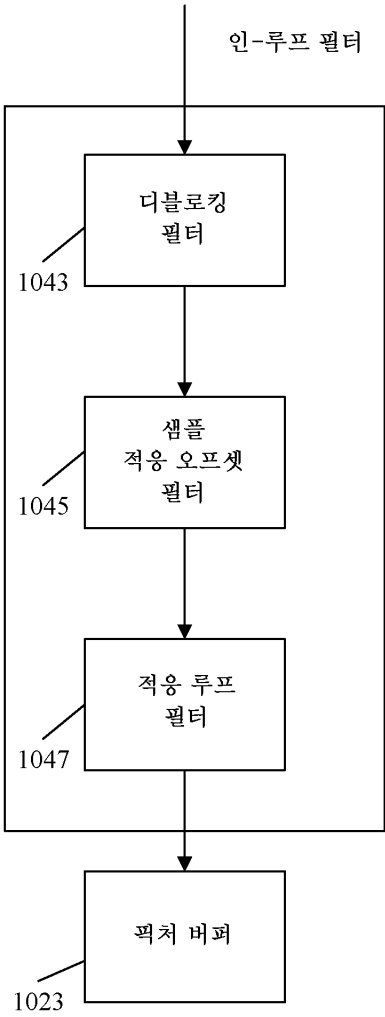


도면9

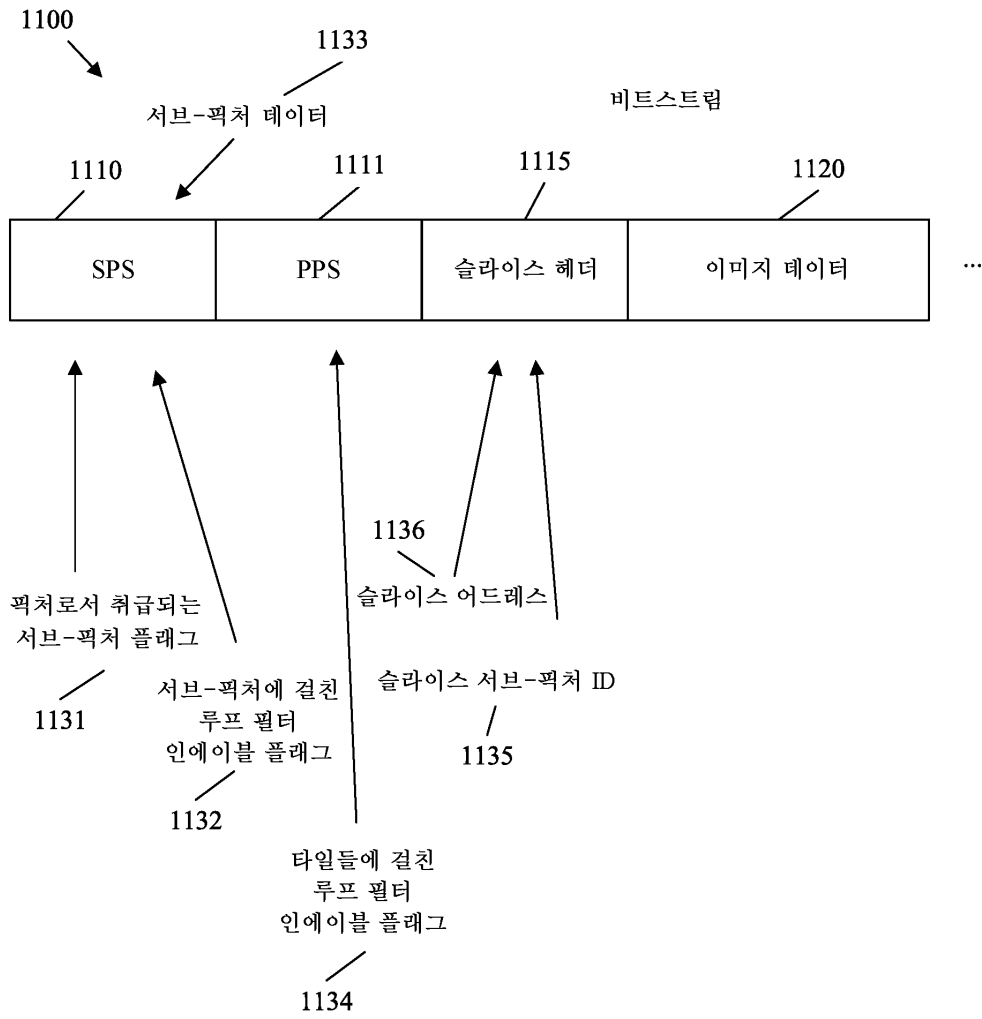


도면10

1000
↓

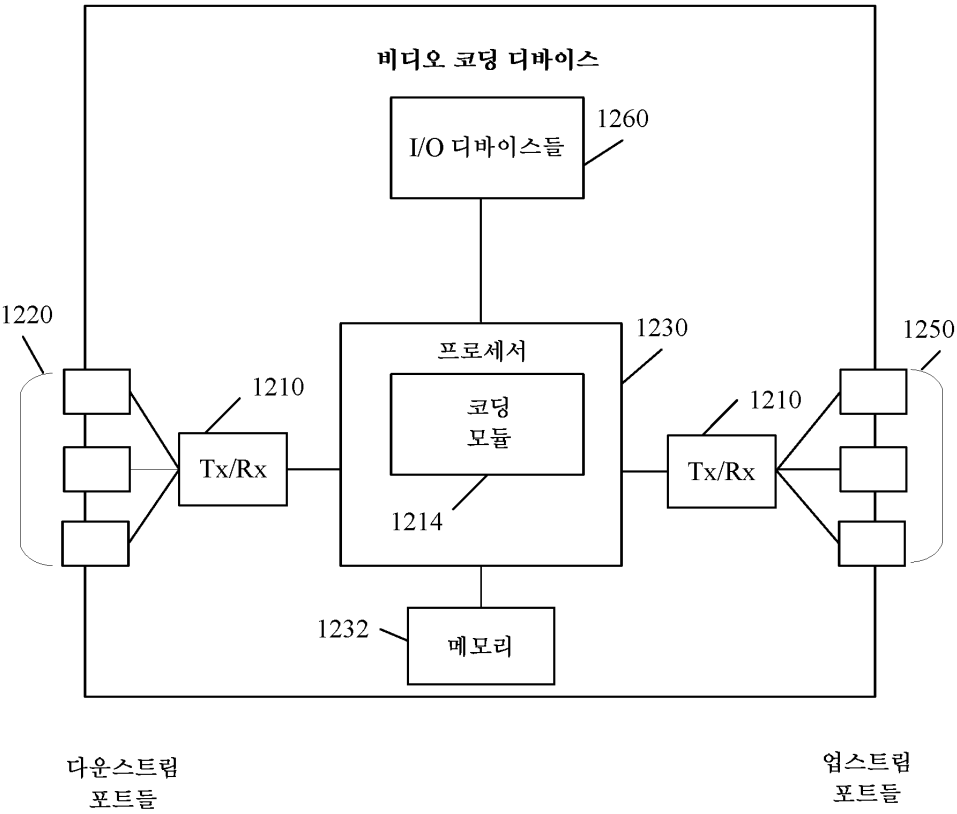


도면11



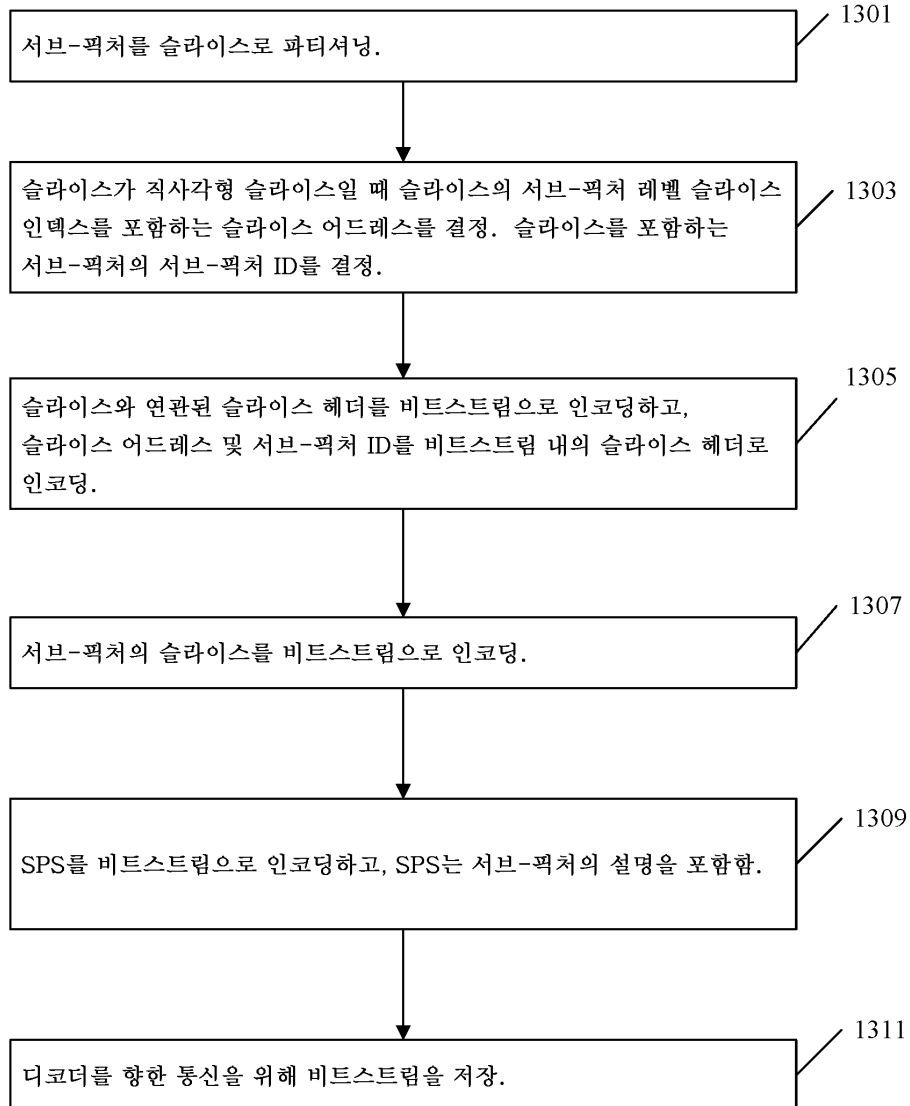
도면12

1200
↙



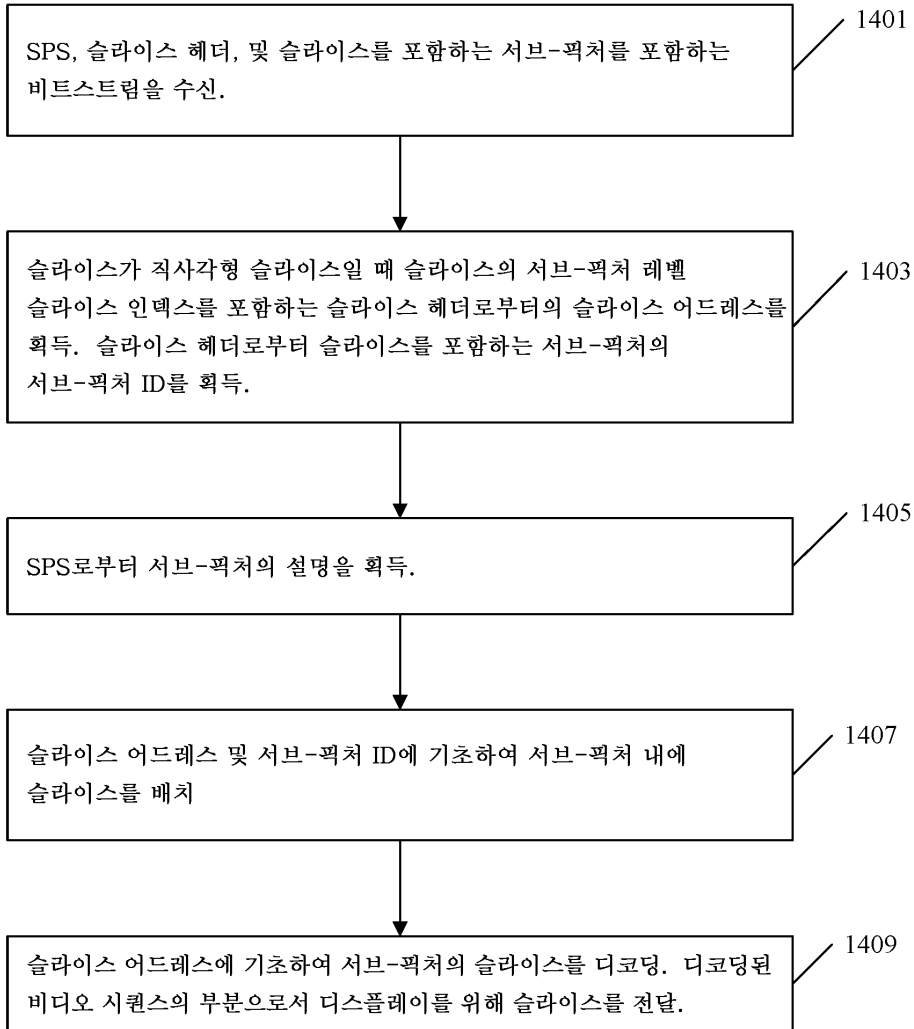
도면13

1300



도면14

1400



도면15

