



(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2025-0004394
(43) 공개일자 2025년01월07일

(51) 국제특허분류(Int. Cl.)
H04N 19/119 (2014.01) H04N 19/13 (2014.01)
H04N 19/176 (2014.01) H04N 19/70 (2014.01)
(52) CPC특허분류
H04N 19/119 (2015.01)
H04N 19/13 (2015.01)
(21) 출원번호 10-2024-7042623(분할)
(22) 출원일자(국제) 2017년09월25일
심사청구일자 2024년12월23일
(62) 원출원 특허 10-2023-7016274
원출원일자(국제) 2017년09월25일
심사청구일자 2023년05월12일
(85) 번역문제출일자 2024년12월23일
(86) 국제출원번호 PCT/EP2017/074224
(87) 국제공개번호 WO 2018/065250
국제공개일자 2018년04월12일
(30) 우선권주장
16306308.4 2016년10월05일
유럽특허청(EPO)(EP)

(71) 출원인
인터디지털 매디슨 페턴트 홀딩스 에스에이에스
프랑스 75017 빠리 뒤편 폴로벨 몰 3
(72) 발명자
레레아넥, 파브리스
프랑스 35576 쉐송 쉐비네 쉐에스 17616 아브뉴
데 상 블랑 975 떼끄니폴로르 내
로베르트, 안토이네
프랑스 35576 쉐송 쉐비네 쉐에스 17616 아브뉴
데 상 블랑 975 떼끄니폴로르 내
포이리에르, 탕이
프랑스 35576 쉐송 쉐비네 쉐에스 17616 아브뉴
데 상 블랑 975 떼끄니폴로르 내
(74) 대리인
양영준, 이민호, 백만기

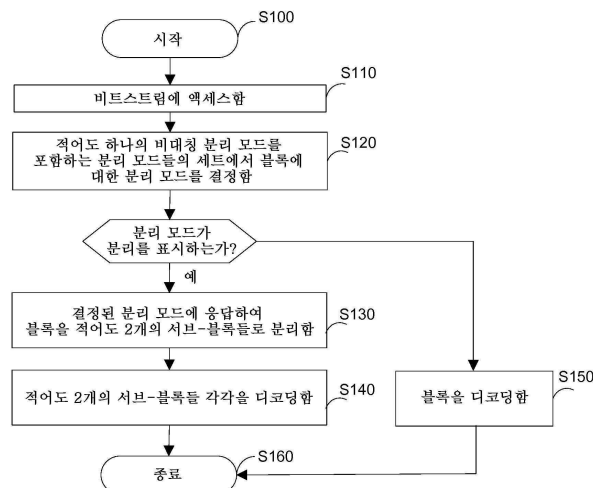
전체 청구항 수 : 총 20 항

(54) 발명의 명칭 인코딩 및 디코딩 방법들 및 대응하는 디바이스들

(57) 요약

이미지의 블록을 디코딩하기 위한 디코딩 방법이 개시되며: - 분리 모드들의 세트에서 상기 블록에 대한 분리 모드를 결정하는 단계; - 상기 분리 모드에 응답하여 상기 블록을 적어도 2개의 서브-블록들로 분리하는 단계; - 적어도 2개의 서브-블록들 각각을 디코딩하는 단계를 포함하고; 상기 분리 모드들의 세트는 상기 블록을 적어도 2개의 서브-블록들로 비대칭적으로 분리하기 위한 적어도 하나의 분리 모드를 포함한다.

대표도 - 도14



(52) CPC특허분류

H04N 19/176 (2015.01)

H04N 19/70 (2015.01)

명세서

청구범위

청구항 1

하나 이상의 프로세서들을 포함하는 디코딩 디바이스로서,

상기 하나 이상의 프로세서들은

복수의 선택스 요소들 중 하나의 선택스 요소를 디코딩하기 위한 컨텍스트를 결정하고 - 상기 복수의 선택스 요소들은 분할 모드들의 세트에서 대상 블록의 분할 모드를 나타내기 위해 비디오 데이터에서 특정되고, 상기 분할 모드들의 세트는 상기 대상 블록을 적어도 2개의 서브 블록으로 비대칭적으로 분할하기 위한 적어도 하나의 분할 모드를 포함하며, 상기 컨텍스트는 상기 대상 블록의 속성을 상기 대상 블록에 이웃하는 적어도 하나의 블록의 동일한 속성과 비교하는 함수임 -; 및

상기 결정된 컨텍스트에 응답하여 상기 선택스 요소를 컨텍스트-기반 산술 디코딩 하도록

구성되는, 디코딩 디바이스.

청구항 2

제1항에 있어서,

상기 대상 블록의 속성을 상기 대상 블록에 이웃하는 적어도 하나의 블록의 동일한 속성과 비교하는 것은, 상기 블록의 이웃 블록의 프론티어가 상기 대상 블록을 비대칭적으로 분할하여 생성된 프론티어와 정렬되는지를 결정하는 것을 포함하는, 디코딩 디바이스.

청구항 3

제1항에 있어서,

상기 대상 블록이 수평으로 분할되는 경우, 상기 대상 블록의 속성을 상기 대상 블록에 이웃하는 적어도 하나의 블록의 동일한 속성과 비교하는 것은, 상기 대상 블록의 좌측에 위치한 이웃 블록의 프론티어가 상기 대상 블록을 수평으로 비대칭적으로 분할하여 생성된 프론티어와 정렬되는지를 결정하는 것을 포함하는, 디코딩 디바이스.

청구항 4

제1항에 있어서,

상기 대상 블록이 수직으로 분할되는 경우, 상기 대상 블록의 속성을 상기 대상 블록에 이웃하는 적어도 하나의 블록의 동일한 속성과 비교하는 것은, 상기 대상 블록 상부에 위치한 이웃 블록의 프론티어가 상기 대상 블록을 수직으로 비대칭적으로 분할하여 생성되는 프론티어와 정렬되는지를 결정하는 것을 포함하는, 디코딩 디바이스.

청구항 5

제1항에 있어서,

상기 대상 블록의 속성을 상기 대상 블록에 이웃하는 적어도 하나의 블록의 동일한 속성과 비교하는 것은, 상기 대상 블록의 이웃 블록의 크기를 상기 대상 블록의 크기와 비교하는 것을 포함하며, 상기 크기는 폭 또는 높이인, 디코딩 디바이스.

청구항 6

제1항에 있어서,

상기 대상 블록의 속성을 상기 대상 블록에 이웃하는 적어도 하나의 블록의 동일한 속성과 비교하는 것은, 상기 대상 블록 크기의 1/4보다 작은 크기를 갖는 이웃 블록의 서브 블록들의 수를 결정하는 것을 포함하며, 상기 크

기는 폭 또는 높이인, 디코딩 디바이스.

청구항 7

제1항에 있어서,

상기 대상 블록의 속성을 상기 대상 블록에 이웃하는 적어도 하나의 블록의 동일한 속성과 비교하는 것은, 상기 대상 블록의 좌측에 위치한 이웃 블록에서 상기 대상 블록 높이의 1/4보다 작은 높이를 갖는 서브 블록들의 수를 결정하는 것을 포함하고, 상기 결정된 콘텍스트에 응답하여 상기 선택스 요소를 콘텍스트 기반 산술 디코딩하는 것은, 상기 수가 0과 같은 경우 제1 콘텍스트를 획득하고, 그렇지 않은 경우 제2 콘텍스트를 획득하는 것을 포함하는, 디코딩 디바이스.

청구항 8

제1항에 있어서,

상기 대상 블록의 속성을 상기 대상 블록에 이웃하는 적어도 하나의 블록의 동일한 속성과 비교하는 것은, 상기 대상 블록의 상부에 위치한 이웃 블록에서 상기 대상 블록의 폭의 1/4보다 작은 폭을 갖는 서브 블록의 수를 결정하는 것을 포함하고, 상기 결정된 콘텍스트에 응답하여 상기 선택스 요소를 콘텍스트-기반 산술 디코딩하는 것은, 상기 수가 0과 같은 경우 제1 콘텍스트를 획득하고 그렇지 않은 경우 제2 콘텍스트를 획득하는 것을 포함하는, 디코딩 디바이스.

청구항 9

하나 이상의 프로세서들을 포함하는 인코딩 디바이스로서,

상기 하나 이상의 프로세서들은

비디오 데이터에서 복수의 선택스 요소들 중 하나의 선택스 요소를 인코딩하기 위한 콘텍스트를 결정하고 - 상기 복수의 선택스 요소들은 분할 모드들의 세트에서 대상 블록의 분할 모드를 나타내기 위해 비디오 데이터에서 특정되고, 상기 분할 모드들의 세트는 상기 대상 블록을 적어도 2개의 서브 블록으로 비대칭적으로 분할하기 위한 적어도 하나의 분할 모드를 포함하며, 상기 콘텍스트는 상기 대상 블록의 속성을 상기 대상 블록에 이웃하는 적어도 하나의 블록의 동일한 속성과 비교하는 함수임 -; 및

상기 결정된 콘텍스트에 응답하여 상기 선택스 요소를 콘텍스트-기반 산술 인코딩 하도록

구성되는, 인코딩 디바이스.

청구항 10

제9항에 있어서,

상기 대상 블록의 속성을 상기 대상 블록에 이웃하는 적어도 하나의 블록의 동일한 속성과 비교하는 것은,

상기 블록의 이웃 블록의 프론티어가 상기 대상 블록을 비대칭적으로 분할하여 생성된 프론티어와 정렬되는지를 결정하는 것;

상기 대상 블록의 좌측에 위치한 이웃 블록의 프론티어가 상기 대상 블록을 수평으로 비대칭적으로 분할하여 생성된 프론티어와 정렬되는지를 결정하는 것;

상기 대상 블록 상부에 위치한 이웃 블록의 프론티어가 상기 대상 블록을 수직으로 비대칭적으로 분할하여 생성되는 프론티어와 정렬되는지를 결정하는 것;

상기 대상 블록의 이웃 블록의 크기를 상기 대상 블록의 크기와 비교하는 것 - 상기 크기는 폭 또는 높이임 -;

상기 대상 블록 크기의 1/4보다 작은 크기를 갖는 이웃 블록의 서브 블록들의 수를 결정하는 것;

상기 대상 블록의 좌측에 위치한 이웃 블록에서 상기 대상 블록 높이의 1/4보다 작은 높이를 갖는 서브 블록들의 수를 결정하는 것; 또는

상기 대상 블록의 상부에 위치한 이웃 블록에서 상기 대상 블록의 폭의 1/4보다 작은 폭을 갖는 서브 블록의 수를 결정하는 것

을 포함하는, 인코딩 디바이스.

청구항 11

디코딩 방법으로서,

복수의 선택스 요소들 중 하나의 선택스 요소를 디코딩하기 위한 컨텍스트를 결정하는 단계 - 상기 복수의 선택스 요소들은 분할 모드들의 세트에서 대상 블록의 분할 모드를 나타내기 위해 비디오 데이터에서 특정되고, 상기 분할 모드들의 세트는 상기 대상 블록을 적어도 2개의 서브 블록으로 비대칭적으로 분할하기 위한 적어도 하나의 분할 모드를 포함하며, 상기 컨텍스트는 상기 대상 블록의 속성을 상기 대상 블록에 이웃하는 적어도 하나의 블록의 동일한 속성과 비교하는 함수임 -; 및

상기 결정된 컨텍스트에 응답하여 상기 선택스 요소를 컨텍스트-기반 산술 디코딩하는 단계를 포함하는 디코딩 방법.

청구항 12

제11항에 있어서,

상기 대상 블록의 속성을 상기 대상 블록에 이웃하는 적어도 하나의 블록의 동일한 속성과 비교하는 것은, 상기 블록의 이웃 블록의 프론티어가 상기 대상 블록을 비대칭적으로 분할하여 생성된 프론티어와 정렬되는지를 결정하는 것을 포함하는, 디코딩 방법.

청구항 13

제11항에 있어서,

상기 대상 블록이 수평으로 분할되는 경우, 상기 대상 블록의 속성을 상기 대상 블록에 이웃하는 적어도 하나의 블록의 동일한 속성과 비교하는 것은, 상기 대상 블록의 좌측에 위치한 이웃 블록의 프론티어가 상기 대상 블록을 수평으로 비대칭적으로 분할하여 생성된 프론티어와 정렬되는지를 결정하는 것을 포함하는, 디코딩 방법.

청구항 14

제11항에 있어서,

상기 대상 블록이 수직으로 분할되는 경우, 상기 대상 블록의 속성을 상기 대상 블록에 이웃하는 적어도 하나의 블록의 동일한 속성과 비교하는 것은, 상기 대상 블록 상부에 위치한 이웃 블록의 프론티어가 상기 대상 블록을 수직으로 비대칭적으로 분할하여 생성되는 프론티어와 정렬되는지를 결정하는 것을 포함하는, 디코딩 방법.

청구항 15

제11항에 있어서,

상기 대상 블록의 속성을 상기 대상 블록에 이웃하는 적어도 하나의 블록의 동일한 속성과 비교하는 것은, 상기 대상 블록의 이웃 블록의 크기를 상기 대상 블록의 크기와 비교하는 것을 포함하며, 상기 크기는 폭 또는 높이인, 디코딩 방법.

청구항 16

제11항에 있어서,

상기 대상 블록의 속성을 상기 대상 블록에 이웃하는 적어도 하나의 블록의 동일한 속성과 비교하는 것은, 상기 대상 블록 크기의 1/4보다 작은 크기를 갖는 이웃 블록의 서브 블록들의 수를 결정하는 것을 포함하며, 상기 크기는 폭 또는 높이인, 디코딩 방법.

청구항 17

제11항에 있어서,

상기 대상 블록의 속성을 상기 대상 블록에 이웃하는 적어도 하나의 블록의 동일한 속성과 비교하는 것은, 상기 대상 블록의 좌측에 위치한 이웃 블록에서 상기 대상 블록 높이의 1/4보다 작은 높이를 갖는 서브 블록들의 수

를 결정하는 것을 포함하고, 상기 결정된 콘텍스트에 응답하여 상기 선택스 요소를 콘텍스트-기반 산술 디코딩하는 단계는, 상기 수가 0과 같은 경우 제1 콘텍스트를 획득하고, 그렇지 않은 경우 제2 콘텍스트를 획득하는 단계를 포함하는, 디코딩 방법.

청구항 18

제11항에 있어서,

상기 대상 블록의 속성을 상기 대상 블록에 이웃하는 적어도 하나의 블록의 동일한 속성과 비교하는 것은, 상기 대상 블록의 상부에 위치한 이웃 블록에서 상기 대상 블록의 폭의 1/4보다 작은 폭을 갖는 서브 블록의 수를 결정하는 것을 포함하고, 상기 결정된 콘텍스트에 응답하여 상기 선택스 요소를 콘텍스트-기반 산술 디코딩하는 단계는, 상기 수가 0과 같은 경우 제1 콘텍스트를 획득하고 그렇지 않은 경우 제2 콘텍스트를 획득하는 단계를 포함하는, 디코딩 방법.

청구항 19

인코딩 방법으로서,

비디오 데이터에서 복수의 선택스 요소들 중 하나의 선택스 요소를 인코딩하기 위한 콘텍스트를 결정하는 단계 - 상기 복수의 선택스 요소들은 분할 모드들의 세트에서 대상 블록의 분할 모드를 나타내기 위해 비디오 데이터에서 특정되고, 상기 분할 모드들의 세트는 상기 대상 블록을 적어도 2개의 서브 블록으로 비대칭적으로 분할하기 위한 적어도 하나의 분할 모드를 포함하며, 상기 콘텍스트는 상기 대상 블록의 속성을 상기 대상 블록에 이웃하는 적어도 하나의 블록의 동일한 속성과 비교하는 함수임 -; 및

상기 결정된 콘텍스트에 응답하여 상기 선택스 요소를 콘텍스트-기반 산술 인코딩 하는 단계를 포함하는, 인코딩 방법.

청구항 20

제19항에 있어서,

상기 대상 블록의 속성을 상기 대상 블록에 이웃하는 적어도 하나의 블록의 동일한 속성과 비교하는 것은,

상기 블록의 이웃 블록의 프론티어가 상기 대상 블록을 비대칭적으로 분할하여 생성된 프론티어와 정렬되는지를 결정하는 것;

상기 대상 블록의 좌측에 위치한 이웃 블록의 프론티어가 상기 대상 블록을 수평으로 비대칭적으로 분할하여 생성된 프론티어와 정렬되는지를 결정하는 것;

상기 대상 블록 상부에 위치한 이웃 블록의 프론티어가 상기 대상 블록을 수직으로 비대칭적으로 분할하여 생성되는 프론티어와 정렬되는지를 결정하는 것;

상기 대상 블록의 이웃 블록의 크기를 상기 대상 블록의 크기와 비교하는 것 - 상기 크기는 폭 또는 높이임 -;

상기 대상 블록 크기의 1/4보다 작은 크기를 갖는 이웃 블록의 서브 블록들의 수를 결정하는 것;

상기 대상 블록의 좌측에 위치한 이웃 블록에서 상기 대상 블록 높이의 1/4보다 작은 높이를 갖는 서브 블록들의 수를 결정하는 것; 또는

상기 대상 블록의 상부에 위치한 이웃 블록에서 상기 대상 블록의 폭의 1/4보다 작은 폭을 갖는 서브 블록의 수를 결정하는 것

을 포함하는, 인코딩 방법.

발명의 설명

기술 분야

이하, 픽처를 인코딩하기 위한 방법 및 디바이스가 개시된다. 대응하는 디코딩 방법 및 디코딩 디바이스가 추가로 개시된다.

[0001]

배경 기술

- [0002] 높은 압축 효율을 달성하기 위해, 비디오 코딩 방식들은 통상적으로 비디오 콘텐츠에서 공간적 및 시간적 리던던시를 레버리지하기 위해 예측 및 변환을 이용한다. 일반적으로, 인트라 또는 인터 프레임 상관을 활용하기 위해 인트라 또는 인터 예측이 사용되며, 잔차들로 종종 표시되는 원래의 이미지와 예측된 이미지 사이의 차이들은 변환, 양자화 및 엔트로피 코딩된다. 비디오를 재구성하기 위해, 압축된 데이터는 예측, 변환, 양자화 및 엔트로피 코딩에 대응하는 역 프로세스들에 의해 디코딩된다.
- [0003] HEVC 코딩("ITU-T H.265 Telecommunication standardization sector of ITU (10/2014), series H: audiovisual and multimedia systems, infrastructure of audiovisual services - coding of moving video, High efficiency video coding, Recommendation ITU-T H.265")에서, 픽처는 통상적으로 64x64, 128x128 또는 256x256인 구성가능한 크기를 갖는 정사각형 형상의 코딩 트리 유닛들(CTU)로 파티셔닝된다. CTU는 코딩 유닛들(CU)로의 퀴드-트리 파티셔닝의 루트이다. 각각의 CU에 대해, CU가 인트라 예측을 사용하여 코딩되는지 또는 인터 예측을 사용하여 코딩되는지 여부를 표시하는 예측 모드가 시그널링된다. 코딩 유닛은 하나 이상의 예측 유닛들(PU)로 파티셔닝되고, 변환 유닛들(TU들)로의 퀴드-트리(변환 트리로 공지됨) 파티셔닝의 루트를 형성한다. PU는 정사각형 또는 직사각형 형상을 가질 수 있는 한편, TU는 정사각형 형상을 갖는다. CU의 PU들로의 비대칭적 세분화가 또한 인터 예측에서 가능한데, 즉, CU가 크기 $N \times N$ 을 가지면, PU는 도 1에 예시된 바와 같이 크기 $N/4 \times N$, $3N/4 \times M$, $N \times N/4$, $N \times 3N/4$ 를 가질 수 있다. 각각의 PU에는 일부 예측 정보, 예를 들어, 모션 정보, 공간 인트라 예측 등이 할당된다.
- [0004] 퀴드-트리 플러스 바이너리-트리(QTBT) 코딩 툴("Algorithm Description of Joint Exploration Test Model 3", Document JVET-C1001_v3, Joint Video Exploration Team of ISO/IEC JTC1/SC29/WG11, 3rd meeting, 26th May-1st June 2015, Geneva, CH)은, HEVC 표준의 CU/PU/TU 배열보다 더 유연한 CTU 표현을 제공하는 새로운 비디오 코딩 툴이다. 퀴드-트리 플러스 바이너리-트리(QTBT) 코딩 툴은, 코딩 유닛들이 퀴드-트리 방식 및 바이너리-트리 방식 둘 모두로 분리될 수 있는 코딩 트리를 정의한다. 코딩 트리 유닛의 이러한 코딩 트리 표현은 도 2에 예시되며, 여기서 실선들은 퀴드-트리 파티셔닝을 표시하고 점선들은 CU의 바이너리 파티셔닝을 표시한다.
- [0005] CTU의 코딩 유닛들로의 분리는 인코더 측에서, 예를 들어, 최소 레이트 왜곡 비용으로 CTU의 QTBT 표현을 결정하는 것으로 구성된 레이트 왜곡 최적화 절차를 통해 판정된다. QTBT 표현에서, CU는 정사각형 또는 직사각형 형상을 갖는다. 코딩 유닛의 크기는 항상 2의 거듭제곱이고 통상적으로 4 내지 128에 이른다. CTU의 QTBT 분해는 2개의 스테이지들을 포함하여: CTU는 먼저 퀴드-트리 방식으로 4개의 CU들로 분리되고, 그 다음, 각각의 퀴드-트리 잎은 도 3에 예시된 바와 같이 바이너리 방식으로 2개의 CU들 또는 퀴드-트리 방식으로 4개의 CU들로 추가로 분할된다. 도 3에서, 실선들은 퀴드-트리 분리를 표현하고, 점선들은 퀴드-트리 잎들에 공간적으로 네장된 바이너리 분리를 표현한다.
- [0006] QTBT 표현에 있어서, CU는 PU들 또는 TU들로 추가로 파티셔닝되지 않는다. 즉, CTU의 파티셔닝이 판정되면, 각각의 CU는 단일 예측 유닛 및 단일 변환 유닛으로 고려된다. 그러나, 이러한 QTBT 표현은 오직 도 4에 예시된 바와 같이 CU의 대칭적 분리만을 허용한다. 도 4는 QTBT에 의해 허용되는 4개의 분리 모드들을 도시한다. 모드 NO_SPLIT는, CU가 추가로 분리되지 않음을 표시하고, 모드 QT_SPLIT는 CU가 퀴드-트리 방식에 따라 4개의 사분면들로 분리됨을 표시하고, 모드 HOR은 CU가 동일한 크기의 2개의 CU들로 수평으로 분리됨을 표시하고, VER은 CU가 동일한 크기의 2개의 CU들로 수직으로 분리됨을 표시한다.

발명의 내용

- [0007] 이미지의 블록을 디코딩하기 위한 디코딩 방법으로서,
- [0008] - 분리 모드들의 세트에서 블록에 대한 분리 모드를 결정하는 단계;
- [0009] - 분리 모드에 응답하여 블록을 적어도 2개의 서브-블록들로 분리하는 단계;
- [0010] - 적어도 2개의 서브-블록들 각각을 디코딩하는 단계를 포함하고;
- [0011] 분리 모드들의 세트는 블록을 적어도 2개의 서브-블록들로 비대칭적으로 분리하기 위한 적어도 하나의 분리 모드를 포함한다.
- [0012] 이미지의 블록을 디코딩하도록 구성되는 디코딩 디바이스가 개시되며:

- [0013] - 분리 모드들의 세트에서 블록에 대한 분리 모드를 결정하기 위한 수단;
- [0014] - 분리 모드에 응답하여 블록을 적어도 2개의 서브-블록들로 분리하기 위한 수단;
- [0015] - 적어도 2개의 서브-블록들 각각을 디코딩하기 위한 수단을 포함하고;
- [0016] 분리 모드들의 세트는 블록을 적어도 2개의 서브-블록들로 비대칭적으로 분리하기 위한 적어도 하나의 분리 모드를 포함한다.
- [0017] 적어도 하나의 스트림에 액세스하도록 구성된 통신 인터페이스 및 적어도 하나의 프로세서를 포함하는 디코딩 디바이스가 개시되며, 적어도 하나의 프로세서는,
- [0018] - 분리 모드들의 세트에서 이미지의 블록에 대한 분리 모드를 결정하고;
- [0019] - 분리 모드에 응답하여 블록을 적어도 2개의 서브-블록들로 분리하고;
- [0020] - 액세스된 스트림으로부터 적어도 2개의 서브-블록들 각각을 디코딩하도록 구성되고;
- [0021] 분리 모드들의 세트는 블록을 적어도 2개의 서브-블록들로 비대칭적으로 분리하기 위한 적어도 하나의 분리 모드를 포함한다.
- [0022] 하기 실시예들은 앞서 개시된 디코딩 방법 및 디코딩 디바이스들에 적용된다.
- [0023] 유리하게는, 분리 모드를 결정하는 단계는,
- [0024] - 블록이 수직으로 분리되는지 또는 수평으로 분리되는지 여부를 특정하는 제1 선택스 요소(btSplitOrientation)를 디코딩하는 단계;
- [0025] - 블록이 대칭적으로 분리되는지 또는 비대칭적으로 분리되는지 여부를 특정하는 제2 선택스 요소(horAsymmetricSplitFlag, verAsymmetricSplitFlag)를 디코딩하는 단계; 및
- [0026] - 블록이 비대칭적으로 분리된다고 제2 선택스 요소가 특정하는 경우, 분리 모드 선택스 요소(vertical/horizontal_asymmetric_type)를 디코딩하는 단계를 포함한다.
- [0027] 특정 특성에 따르면, 제2 선택스 요소, 즉, horAsymmetricSplitFlag 또는 verAsymmetricSplitFlag는 단일 비트로부터 디코딩된다.
- [0028] 변형에서, 제2 선택스 요소(horAsymmetricSplitFlag 또는 verAsymmetricSplitFlag)는, 디코딩 순서에서 현재 블록에 선행하는 블록들을 디코딩할 때 제2 선택스 요소에 의해 취해진 값들에 의존하는 단일 콘텍스트를 사용하여 콘텍스트-기반 산술 코딩에 의해 디코딩된다.
- [0029] 변형에서, 제2 선택스 요소 horAsymmetricSplitFlag는, 현재 블록 높이의 1/4보다 낮거나 그와 동일한 높이를 갖는 좌측 이웃 블록에 위치한 서브-블록들의 수에 의존하는 2개의 콘텍스트들을 사용하여 콘텍스트-기반 산술 코딩에 의해 디코딩된다. 동일한 방식으로, 제2 선택스 요소 verAsymmetricSplitFlag는, 현재 블록 폭의 1/4보다 낮거나 그와 동일한 폭을 갖는 최상부 이웃 블록에 위치한 서브-블록들의 수에 의존하는 2개의 콘텍스트들을 사용하여 콘텍스트-기반 산술 코딩에 의해 디코딩된다.
- [0030] 추가적인 실시예에 따르면, 선행 실시예들 중 하나에서 제안된 콘텍스트들은 현재 블록과 연관된 쿼드-트리 플러스 바이너리 트리 깊이 값의 함수로서 또는 블록의 폭 및 높이의 함수로서 인덱싱된다.
- [0031] 특정 특성에 따르면, 분리 모드 선택스 요소(vertical_asymmetric_type/horizontal_asymmetric_type)는 단일 비트로부터 디코딩된다.
- [0032] 변형에서, 분리 모드 선택스 요소(vertical_asymmetric_type 또는 horizontal_asymmetric_type)는, 디코딩 순서에서 현재 블록에 선행하는 블록들을 디코딩할 때 분리 모드 선택스 요소에 의해 취해진 값들에 의존하는 단일 콘텍스트를 사용하여 콘텍스트-기반 산술 코딩에 의해 디코딩된다.
- [0033] 변형에서, 분리 모드 선택스 요소 horizontal_asymmetric_type은, 현재 블록에 대한 수평 비대칭 분리의 선택을 통해 획득된 현재 블록의 수평 프론티어(frontier)와 공간적으로 일치하는 프론티어의 현재 블록의 좌측에 위치한 블록에서의 존재에 의존하는 3개의 콘텍스트들을 사용하여 콘텍스트-기반 산술 코딩에 의해 디코딩된다.
- [0034] 동일한 방식으로, 분리 모드 선택스 요소 vertical_asymmetric_type은, 현재 블록에 대한 수직 비대칭 분리의 선택을 통해 획득된 현재 블록의 수직 프론티어와 공간적으로 일치하는 프론티어의 현재 블록의 위에 위치한 블

록에서의 존재에 의존하는 3개의 콘텍스트들을 사용하여 콘텍스트-기반 산술 코딩에 의해 디코딩된다.

- [0035] 추가적인 실시예에 따르면, 선행 실시예들 중 하나에서 제안된 콘텍스트들은 현재 블록과 연관된 퀴드-트리 플러스 바이너리 트리 깊이 값의 함수로서 또는 블록의 폭 및 높이의 함수로서 인덱싱된다.
- [0036] 변형에서, 분리 모드를 결정하는 단계는,
- [0037] 블록이 디코딩 순서에서 다음 블록과 병합되는 것을 표시하는 선택스 요소(merge_with_next_CU_flag)를 디코딩하는 단계를 포함한다.
- [0038] 특정 실시예에서, 블록은 루마 샘플들의 블록, 크로마 샘플들의 블록 또는 루마 샘플들의 블록 및 크로마 샘플들의 2개의 블록들로 형성된 코딩 유닛 중 하나이다.
- [0039] 특정 실시예에서, 분리 모드에 응답하여 블록을 적어도 2개의 서브-블록들로 분리하는 단계는 블록을 높이 $h/4$ 의 하나의 서브-블록 및 높이 $3h/4$ 의 하나의 서브-블록으로 수평으로 분리하는 단계를 포함하고, 여기서 h 는 블록의 높이이다.
- [0040] 특정 실시예에서, 분리 모드에 응답하여 블록을 적어도 2개의 서브-블록들로 분리하는 단계는 블록을 폭 $w/4$ 의 하나의 서브-블록 및 폭 $3w/4$ 의 하나의 서브-블록으로 수직으로 분리하는 단계를 포함하고, 여기서 w 는 블록의 폭이다.
- [0041] 이미지의 블록을 인코딩하기 위한 코딩 방법이 또한 개시되며:
- [0042] - 분리 모드들의 세트에서 블록에 대한 분리 모드를 결정하는 단계;
- [0043] - 분리 모드에 응답하여 블록을 적어도 2개의 서브-블록들로 분리하는 단계;
- [0044] - 적어도 2개의 서브-블록들 각각을 인코딩하는 단계를 포함하고;
- [0045] 분리 모드들의 세트는 블록을 적어도 2개의 서브-블록들로 비대칭적으로 분리하기 위한 적어도 하나의 분리 모드를 포함한다.
- [0046] 이미지의 블록을 인코딩하도록 구성된 코딩 디바이스가 또한 개시되며:
- [0047] - 분리 모드들의 세트에서 블록에 대한 분리 모드를 결정하기 위한 수단;
- [0048] - 분리 모드에 응답하여 블록을 적어도 2개의 서브-블록들로 분리하기 위한 수단;
- [0049] - 적어도 2개의 서브-블록들 각각을 인코딩하기 위한 수단을 포함하고;
- [0050] 분리 모드들의 세트는 블록을 적어도 2개의 서브-블록들로 비대칭적으로 분리하기 위한 적어도 하나의 분리 모드를 포함한다.
- [0051] 현재 블록으로 지칭되는 적어도 이미지의 블록에 액세스하도록 구성된 통신 인터페이스 및 적어도 하나의 프로세서를 포함하는 코딩 디바이스가 개시된다. 적어도 하나의 프로세서는,
- [0052] - 분리 모드들의 세트에서 블록에 대한 분리 모드를 결정하고;
- [0053] - 분리 모드에 응답하여 블록을 적어도 2개의 서브-블록들로 분리하고;
- [0054] - 적어도 2개의 서브-블록들 각각을 인코딩하도록 구성되고;
- [0055] 분리 모드들의 세트는 블록을 적어도 2개의 서브-블록들로 비대칭적으로 분리하기 위한 적어도 하나의 분리 모드를 포함한다.
- [0056] 하기 실시예들은 앞서 개시된 코딩 방법 및 코딩 디바이스들에 적용된다.
- [0057] 유리하게는, 분리 모드를 결정하는 단계는,
- [0058] - 블록이 수직으로 분리되는지 또는 수평으로 분리되는지 여부를 특정하는 제1 선택스 요소(btSplitOrientation)를 결정 및 인코딩하는 단계;
- [0059] - 블록이 대칭적으로 분리되는지 또는 비대칭적으로 분리되는지 여부를 특정하는 제2 선택스 요소(horAsymmetricSplitFlag, verAsymmetricSplitFlag)를 결정 및 인코딩하는 단계; 및
- [0060] - 블록이 비대칭적으로 분리된다고 제2 선택스 요소가 특정하는 경우, 분리 모드 선택스 요소

(vertical_asymmetric_type /horizontal_asymmetric_type)를 결정 및 인코딩하는 단계를 포함한다.

- [0061] 특정 특성에 따르면, 제2 선택 요소, 즉, horAsymmetricSplitFlag 또는 verAsymmetricSplitFlag는 단일 비트로 인코딩된다.
- [0062] 변형에서, 제2 선택 요소는, 인코딩 순서에서 현재 블록에 선행하는 블록들을 인코딩할 때 제2 선택 요소에 의해 취해진 값들에 의존하는 단일 컨텍스트를 사용하여 컨텍스트-기반 산술 코딩에 의해 인코딩된다.
- [0063] 변형에서, 제2 선택 요소 horAsymmetricSplitFlag는, 현재 블록 높이의 1/4보다 낮거나 그와 동일한 높이를 갖는 좌측 이웃 블록에 위치한 서브-블록들의 수에 의존하는 2개의 컨텍스트들을 사용하여 컨텍스트-기반 산술 코딩에 의해 인코딩된다. 동일한 방식으로, 제2 선택 요소 verAsymmetricSplitFlag는, 현재 블록 폭의 1/4보다 낮거나 그와 동일한 폭을 갖는 최상부 이웃 블록에 위치한 서브-블록들의 수에 의존하는 2개의 컨텍스트들을 사용하여 컨텍스트-기반 산술 코딩에 의해 인코딩된다.
- [0064] 추가적인 실시예에 따르면, 선행 실시예들 중 하나에서 제안된 컨텍스트들은 현재 블록과 연관된 쿼드-트리 플러스 바이너리 트리 깊이 값의 함수로서 또는 블록의 폭 및 높이의 함수로서 인덱싱된다.
- [0065] 특정 특성에 따르면, 분리 모드 선택 요소(vertical_asymmetric_type/horizontal_asymmetric_type)는 단일 비트로 인코딩된다.
- [0066] 변형에서, 분리 모드 선택 요소는, 인코딩 순서에서 현재 블록에 선행하는 블록들을 인코딩할 때 분리 모드 선택 요소에 의해 취해진 값들에 의존하는 단일 컨텍스트를 사용하여 컨텍스트-기반 산술 코딩에 의해 인코딩된다.
- [0067] 변형에서, 분리 모드 선택 요소 horizontal_asymmetric_type은, 현재 블록에 대한 수평 비대칭 분리의 선택을 통해 획득된 현재 블록의 수평 프론티어(frontier)와 공간적으로 일치하는 프론티어의 현재 블록의 좌측에 위치한 블록에서의 존재에 의존하는 3개의 컨텍스트들을 사용하여 컨텍스트-기반 산술 코딩에 의해 인코딩된다.
- [0068] 동일한 방식으로, 분리 모드 선택 요소 vertical_asymmetric_type은, 현재 블록에 대한 수직 비대칭 분리의 선택을 통해 획득된 현재 블록의 수직 프론티어와 공간적으로 일치하는 프론티어의 현재 블록의 위에 위치한 블록에서의 존재에 의존하는 3개의 컨텍스트들을 사용하여 컨텍스트-기반 산술 코딩에 의해 인코딩된다.
- [0069] 추가적인 실시예에 따르면, 선행 실시예들 중 하나에서 제안된 컨텍스트들은 현재 블록과 연관된 쿼드-트리 플러스 바이너리 트리 깊이 값의 함수로서 또는 블록의 폭 및 높이의 함수로서 인덱싱된다.
- [0070] 다른 실시예에서, 분리 모드를 결정하는 단계는,
- [0071] 블록이 인코딩 순서에서 다음 블록과 병합되는 것을 표시하는 선택 요소(merge_with_next_CU_flag)를 결정 및 인코딩하는 단계를 포함한다.
- [0072] 예시적인 실시예로서, 블록은 루마 샘플들의 블록, 크로마 샘플들의 블록 또는 루마 샘플들의 블록 및 크로마 샘플들의 2개의 블록들로 형성된 코딩 유닛 중 하나이다.
- [0073] 일 실시예에서, 분리 모드에 응답하여 블록을 적어도 2개의 서브-블록들로 분리하는 단계는 블록을 높이 $h/4$ 의 하나의 서브-블록 및 높이 $3h/4$ 의 하나의 서브-블록으로 수평으로 분리하는 단계를 포함하고, 여기서 h 는 블록의 높이이다.
- [0074] 일 실시예에서, 분리 모드에 응답하여 블록을 적어도 2개의 서브-블록들로 분리하는 단계는 블록을 폭 $w/4$ 의 하나의 서브-블록 및 폭 $3w/4$ 의 하나의 서브-블록으로 수직으로 분리하는 단계를 포함하고, 여기서 w 는 블록의 폭이다.
- [0075] 블록을 표현하는 코딩된 데이터를 포함하는 스트림이 또한 개시되고, 코딩된 데이터는 분리 모드들의 세트에서 블록에 대한 분리 모드를 결정하고, 분리 모드에 응답하여 블록을 적어도 2개의 서브-블록들로 분리하고; 적어도 2개의 서브-블록들 각각을 코딩된 데이터로 인코딩함으로써 획득되고, 분리 모드들의 세트는 블록을 적어도 2개의 서브-블록들로 비대칭적으로 분리하기 위해 적어도 하나의 분리 모드를 포함한다.
- [0076] 스트림을 저장하는 비일시적 프로세서 판독가능 매체가 개시되며, 스트림은 블록을 표현하는 코딩된 데이터를 포함하고, 코딩된 데이터는 분리 모드들의 세트에서 블록에 대한 분리 모드를 결정하고, 분리 모드에 응답하여 블록을 적어도 2개의 서브-블록들로 분리하고; 적어도 2개의 서브-블록들 각각을 코딩된 데이터로 인코딩함으로써 획득되고, 분리 모드들의 세트는 블록을 적어도 2개의 서브-블록들로 비대칭적으로 분리하기 위해 적어도 하

나의 분리 모드를 포함한다.

- [0077] 블록을 표현하는 코딩된 데이터를 송신하는 단계를 포함하는 송신 방법이 개시되며, 코딩된 데이터는 분리 모드들의 세트에서 블록에 대한 분리 모드를 결정하고, 분리 모드에 응답하여 블록을 적어도 2개의 서브-블록들로 분리하고; 적어도 2개의 서브-블록들 각각을 코딩된 데이터로 인코딩함으로써 획득되고, 분리 모드들의 세트는 블록을 적어도 2개의 서브-블록들로 비대칭적으로 분리하기 위해 적어도 하나의 분리 모드를 포함한다.
- [0078] 유리하게는, 코딩된 데이터는,
- [0079] - 블록이 수직으로 분리되는지 또는 수평으로 분리되는지 여부를 특징하는 제1 선택스 요소(btSplitOrientation);
- [0080] - 블록이 대칭적으로 분리되는지 또는 비대칭적으로 분리되는지 여부를 특징하는 제2 선택스 요소(horAsymmetricSplitFlag, verAsymmetricSplitFlag); 및
- [0081] - 블록이 비대칭적으로 분리된다고 제2 선택스 요소가 특징하는 경우, 분리 모드 선택스 요소(vertical/horizontal_asymmetric_type)를 포함한다.
- [0082] 변형에서, 코딩된 데이터는 블록이 인코딩 순서에서 다음 블록과 병합되는 것을 표시하는 선택스 요소(merge_with_next_CU_flag)를 포함한다.

도면의 간단한 설명

- [0083] 도 1은 종래 기술에 따라 코딩 유닛의 예측 유닛들로의 비대칭적 세분화를 예시한다.
- 도 2 및 도 3은 최신 기술에 따른 코딩 트리 유닛의 코딩 유닛으로의 분리를 예시한다.
- 도 4는 쿼드-트리 플러스 바이너리-트리 코딩 틀에 의해 정의된 바와 같은 코딩 유닛의 분리 모드들을 예시한다.
- 도 5는 비제한적인 실시예에 따라 이미지 블록을 비트스트림에서 인코딩하도록 구성된 송신기의 예시적인 아키텍처를 표현한다.
- 도 6은 도 7 및/또는 도 11의 인코딩 방법을 실행하도록 적응된 예시적인 비디오 인코더를 예시한다.
- 도 7은 특정한 비제한적 실시예에 따라 이미지의 블록을 비트스트림에서 인코딩하기 위한 방법의 흐름도를 표현한다.
- 도 8은 본 원리들에 따라 비대칭적 분리 모드들의 예들을 도시한다.
- 도 9는 본 원리들에 따라 CTU의 CU들로의 파티셔닝의 예를 도시한다.
- 도 10은 인코딩 선택스의 그래픽 표현이다.
- 도 11은 특정한 비제한적 실시예에 따라 현재 블록에 대한 바이너리 분리 정보를 인코딩하기 위한 방법의 흐름도를 표현한다.
- 도 12는 비제한적인 실시예에 따라 디코딩된 이미지 블록을 획득하기 위해 비트스트림으로부터 이미지 블록을 디코딩하도록 구성된 수신기의 예시적인 아키텍처를 표현한다.
- 도 13은 도 14 및/또는 도 15의 디코딩 방법을 실행하도록 적응된 예시적인 비디오 디코더의 블록도를 예시한다.
- 도 14는 특정한 비제한적 실시예에 따라 비트스트림으로부터 이미지의 블록을 디코딩하기 위한 방법의 흐름도를 표현한다.
- 도 15는 특정한 비제한적 실시예에 따른 바이너리 분리의 경우 현재 블록의 분리 모드를 디코딩하기 위한 방법의 흐름도를 표현한다.
- 도 16 및 도 17은 본 원리들에 따른 비대칭적 분리 모드들을 표현하는 선택스 요소들의 엔트로피 코딩 프로세스를 예시한다.
- 도 18은 인코딩 선택스의 변형을 예시한다.

발명을 실시하기 위한 구체적인 내용

- [0084] 도면들 및 설명들은, 명료성을 위해, 통상적인 인코딩 및/또는 디코딩 디바이스들에서 발견되는 많은 다른 요소들을 제거하면서, 본 원리들의 명확한 이해와 관련된 요소들을 예시하기 위해 단순화되었음을 이해해야 한다. 제1 및 제2라는 용어들이 본 명세서에서 다양한 요소들을 설명하기 위해 사용될 수 있지만, 이러한 요소들은 이러한 용어들에 의해 제한되어서는 안됨을 이해할 것이다. 이러한 용어들은 하나의 요소를 다른 요소로부터 구별하기 위해서만 사용된다. 다양한 방법들이 앞서 설명되었고, 방법들 각각은 설명된 방법을 달성하기 위한 하나 이상의 단계들 또는 동작들을 포함한다. 방법의 적절한 동작을 위해 단계들 또는 동작들의 특정 순서가 요구되지 않으면, 특정 단계들 및/또는 동작들의 순서 및/또는 사용은 수정되거나 결합될 수 있다.
- [0085] 픽처는 모노크롬(monochrome) 포맷에서 루마 샘플들의 어레이 또는 4:2:0, 4:2:2, 및 4:4:4 컬러 포맷에서 루마 샘플들의 어레이 및 크로마 샘플들의 2개의 대응하는 어레이들이다. 일반적으로, "블록"은 샘플 어레이에서 특정 영역(예를 들어, 루마 Y)을 다루며, "유닛"은 모든 인코딩된 컬러 성분들(루마 Y 및 가능하게는 크로마 Cb 및 크로마 Cr)의 코로케이티드(collocated) 블록을 포함한다. 그러나, 용어 "블록"은 본 명세서에서 블록(예를 들어, CB) 또는 유닛(예를 들어, CU)을 지칭하기 위해 사용된다.
- [0086] 하기 섹션들에서, 단어 "재구성된" 및 "디코딩된"은 상호교환가능하게 사용될 수 있다. 필수적은 아니지만 통상적으로, "재구성된"은 인코더 측에서 사용되는 한편, "디코딩된"은 디코더 측에서 사용된다.
- [0087] 인코딩 장치
- [0088] 도 5는 비제한적인 실시예에 따라 이미지 블록을 비트스트림에서 인코딩하도록 구성된 송신기(200)의 예시적인 아키텍처를 표현한다.
- [0089] 송신기(200)는 내부 메모리(2030)(예를 들어, RAM, ROM 및/또는 EPROM)와 함께 예를 들어 CPU, GPU 및/또는 DSP(Digital Signal Processor의 영문 두문자어)를 포함할 수 있는 하나 이상의 프로세서(들)(2000)를 포함한다. 송신기(200)는, 출력 정보를 디스플레이하고/하거나 사용자가 커맨드들 및/또는 데이터(예를 들어, 스트림)를 입력할 수 있게 하도록 각각 적응된 하나 이상의 통신 인터페이스(들)(2010)(예를 들어, 키보드, 마우스, 터치패드, 웹캠); 및 송신기(200)의 외부에 있을 수 있는 전원(2020)을 포함할 수 있다. 송신기(200)는 또한 하나 이상의 네트워크 인터페이스(들)(미도시)를 포함할 수 있다. 인코더(2040)는 코딩 기능들을 수행하기 위해 디바이스에 포함될 수 있는 모듈을 표현한다. 추가적으로, 인코더 모듈(2040)은 송신기(200)의 개별 요소로서 구현될 수 있거나 또는 본 기술분야의 통상의 기술자들에게 공지된 바와 같이 하드웨어 및 소프트웨어의 조합으로서 프로세서(들)(2000) 내에 통합될 수 있다.
- [0090] 이미지 블록은 소스로부터 획득될 수 있다. 상이한 실시예들에 따르면, 소스는 다음과 같을 수 있지만, 이에 제한되는 것은 아니다:
- [0091] 로컬 메모리, 예를 들어, 비디오 메모리, RAM, 플래시 메모리, 하드 디스크;
- [0092] 저장 인터페이스, 예를 들어, 대용량 저장소, ROM, 광 디스크 또는 자기 지원부와의 인터페이스;
- [0093] 통신 인터페이스, 예를 들어, 유선 인터페이스(예를 들어, 버스 인터페이스, 광역 네트워크 인터페이스, 로컬 영역 네트워크 인터페이스) 또는 무선 인터페이스(예를 들어, IEEE 802.11 인터페이스 또는 블루투스 인터페이스); 및
- [0094] 이미지 캡처 회로(예를 들어, CCD(즉, Charge-Coupled Device) 또는 CMOS(즉, Complementary Metal-Oxide-Semiconductor)와 같은 센서).
- [0095] 상이한 실시예들에 따르면, 비트스트림은 목적지로 전송될 수 있다. 예로서, 비트스트림은 원격 또는 로컬 메모리, 예를 들어, 비디오 메모리 또는 RAM, 하드 디스크에 저장된다. 변형으로, 비트스트림은 저장 인터페이스, 예를 들어, 대용량 저장소, ROM, 플래시 메모리, 광 디스크 또는 자기 지원부와의 인터페이스에 전송되고 그리고/또는 통신 인터페이스, 예를 들어, 포인트 투 포인트 링크, 통신 버스, 포인트 투 멀티포인트 링크 또는 브로드캐스트 네트워크로의 인터페이스를 통해 송신된다.
- [0096] 예시적이고 비제한적인 실시예에 따르면, 송신기(200)는 메모리(2030)에 저장된 컴퓨터 프로그램을 더 포함한다. 컴퓨터 프로그램은, 송신기(200), 특히 프로세서(2000)에 의해 실행되는 경우, 송신기(200)로 하여금 도 7 및/또는 도 11을 참조하여 설명된 인코딩 방법을 실행하게 하는 명령어들을 포함한다. 변형에 따르면, 컴퓨터 프로그램은 송신기(200)의 외부에서, 비일시적 디지털 데이터 지원부 상에, 예를 들어, HDD, CD-ROM,

DVD, 판독-전용 및/또는 DVD 드라이브 및/또는 DVD 판독/기록 드라이브와 같은 외부 저장 매체 상에 저장되며, 이들 모두는 본 기술분야에 공지되어 있다. 따라서, 송신기(200)는 컴퓨터 프로그램을 판독하기 위한 메커니즘을 포함한다. 추가로, 송신기(200)는 하나 이상의 범용 직렬 버스(USB)-타입의 저장 디바이스들(예를 들어, "메모리 스틱들")에, 대응하는 USB 포트들(미도시)을 통해 액세스할 수 있다.

- [0097] 예시적이고 비제한적인 실시예들에 따르면, 송신기(200)는 다음과 같을 수 있지만, 이에 제한되는 것은 아니다:
- [0098] 모바일 디바이스;
- [0099] 통신 디바이스;
- [0100] 게임 디바이스;
- [0101] 태블릿(또는 태블릿 컴퓨터);
- [0102] 랩탑;
- [0103] 스틸 이미지 카메라;
- [0104] 비디오 카메라;
- [0105] 인코딩 칩 또는 인코딩 디바이스/장치;
- [0106] 스틸 이미지 서버; 및
- [0107] 비디오 서버(예를 들어, 브로드캐스트 서버, 비디오-온-디맨드 서버 또는 웹 서버).
- [0108] 도 6은 도 7 및/또는 도 11의 인코딩 방법을 실행하도록 적응된 예시적인 비디오 인코더(400)를 예시한다. 인코더(400)는 송신기(200)의 예이거나 또는 이러한 송신기(200)의 일부일 수 있다. 하나 이상의 픽처들을 갖는 비디오 시퀀스를 인코딩하기 위해, 픽처는 구성가능한 크기를 갖는 정사각형 형상의 코딩 트리 유닛들(CTU)로 파티셔닝될 수 있다. 코딩 트리 유닛들의 연속적인 세트는 슬라이스로 그룹화될 수 있다. CTU는 코딩 유닛들로의 QTBT 파티셔닝의 루트이다. 예시적인 인코더(400)에서, 픽처는 아래에서 설명되는 바와 같이 인코더 모듈들에 의해 인코딩된다. 각각의 블록은 인트라 또는 인터 모드를 사용하여 인코딩된다. 블록이 인트라 모드에서 인코딩되는 경우, 인코더(400)는 인트라 예측을 수행한다(모듈(460)). 인터 모드에서, 모션 추정(모듈(475)) 및 보상(모듈(470))이 수행된다. 인코더는, 블록을 인코딩하기 위해 인트라 모드 또는 인터 모드 중 어느 것을 사용할지를 판정하고(모듈(405)), 예측 모드 플래그에 의해 인트라/인터 결정을 표시한다. 잔차들은 예측된 샘플 블록(또한 예측자로 공지됨)을 원래의 이미지 블록으로부터 감산함으로써(모듈(410)) 계산된다.
- [0109] 일례로서, 인트라 모드의 블록들은 재구성된 이웃 샘플들로부터 예측된다. 인터 예측은 기준 픽처 버퍼(480)에 저장된 기준 블록을 모션-보상함으로써 수행된다.
- [0110] 잔차들은 변환되고(모듈(425)) 양자화된다(모듈(430)). 양자화된 변환 계수들 뿐만 아니라 모션 벡터들 및 다른 선택스 요소들은 비트스트림을 출력하도록 엔트로피 코딩된다(모듈(445)). 인코더는 또한 변환을 스킵하고, 비-변환된 잔차 신호에 직접 양자화를 적용할 수 있다. 인코더는 또한 변환 및 양자화 둘 모두를 우회할 수 있는데, 즉, 잔차는 변환 또는 양자화 프로세스의 적용 없이 직접 코딩된다. 직접 PCM 코딩에서, 어떠한 예측도 적용되지 않고, 블록 샘플들은 비트스트림으로 직접 코딩된다.
- [0111] 인코더는 디코딩 루프를 포함하고, 추가적인 예측들을 위한 기준을 제공하기 위해 인코딩된 블록을 디코딩한다. 잔차들을 디코딩하기 위해, 양자화된 변환 계수들은 역양자화되고(모듈(440)) 역변환된다(모듈(450)). 이미지 블록은 디코딩된 잔차들 및 예측된 샘플 블록을 결합함으로써(모듈(455)) 재구성된다. 루프 내 필터(465)는, 예를 들어, 코딩 아티팩트들을 감소시키기 위한 디블로킹/SAO(Sample Adaptive Offset) 필터링을 수행하기 위해, 재구성된 픽처에 적용될 수 있다. 필터링된 이미지는 기준 픽처 버퍼(480)에 저장된다.
- [0112] 인코딩 방법들
- [0113] 도 7은 특정한 비제한적 실시예에 따라 이미지의 블록을 비트스트림에서 인코딩하기 위한 방법의 흐름도를 표현한다.
- [0114] 방법은 단계(S200)에서 시작한다. 단계(S210)에서, 예를 들어, 인코더(400)와 같은 송신기가 이미지의 블록에 액세스한다. 블록은 크기 $w \times h$ 이고, 여기서 w 는 이의 폭(픽셀 단위)이고 h 는 이의 높이(픽셀 단위)이다. 블록은 루마 샘플들의 코딩 블록, 크로마 샘플들의 코딩 블록 또는 루마 샘플들의 코딩 블록 및 크로마 샘플들의 2

개의 코딩 블록들로 형성된 코딩 유닛 중 하나이다. 단계(S220)에서, 송신기는 분리 모드들의 세트에서 액세스된 블록에 대한 분리 모드를 결정한다. 특정 실시예에서, 결정된 분리 모드는 최소 레이트-왜곡 비용을 제공하는 분리 모드들의 세트 중 하나이다. 본 원리들에 따르면, 분리 모드들의 세트는 적어도 하나의 비대칭적 분리 모드를 포함한다. 도 8은 이러한 비대칭적 분리 모드들의 예들을 도시한다. 제1 실시예에서, 적어도 하나의 비대칭적 분리 모드는 블록을 높이 $h/4$ 의 하나의 서브-블록 및 높이 $3h/4$ 의 하나의 서브-블록으로 수평으로 분리하는 것을 표시한다. 높이 $h/4$ 의 서브-블록은 높이 $3h/4$ 의 서브-블록의 위(모드 HOR_UP) 또는 아래(모드 HOR_DOWN)에 있다. 다른 실시예에서, 적어도 하나의 비대칭적 분리 모드는 블록을 폭 $w/4$ 의 하나의 서브-블록 및 폭 $3w/4$ 의 하나의 서브-블록으로 수직으로 분리하는 것을 표시한다. 폭 $w/4$ 의 서브-블록은 폭 $3w/4$ 의 서브-블록의 좌측(모드 VER_LEFT) 또는 우측(모드 VER_RIGHT)에 있다. 특정 실시예에서, 분리 모드들의 세트는 도 8에 도시된 모든 분리 모드들을 포함한다. 변형 실시예에서, 적어도 하나의 비대칭 분리 모드는, 2의 거듭제곱이 아닌 크기를 갖는, 예를 들어, 폭 w 또는 높이 h 에서 이들의 크기가 $3 \cdot 2^n$ 픽셀들과 동일하고 n 은 정수인 서브-블록들로 수평으로 또는 수직으로 블록을 분리하는 것을 표시한다.

[0115] 변형에서, 분리 모드들의 세트는 적어도 하나의 비대칭적 분리 모드에 추가로, QTBT에 의해 정의되고 도 4에 도시된 분리 모드를 포함한다. 블록이 추가로 분리되지 않음을 분리 모드가 표시하는 경우(예를 들어, 분리 모드 = NO_SPLIT), 블록은 단계(S250)에서 인코딩된다.

[0116] 그렇지 않으면, 단계(S230)에서, 송신기는 결정된 분리 모드에 응답하여 액세스된 블록을 적어도 2개의 서브-블록들로 분리한다.

[0117] 단계(S240)에서, 송신기는 적어도 2개의 서브-블록들을 비트스트림에서 인코딩한다. 적어도 2개의 서브-블록들을 인코딩하는 단계는 적어도 2개의 서브-블록들 각각에 대해 단계들(S220 내지 S250)을 회귀적으로 적용하는 단계를 포함할 수 있다. 예를 들어, 분리 모드 NO_SPLIT가 단계(S220)에서 결정되기 때문에, 회귀적 분리는 블록이 추가로 분리되지 않을 때 종료된다. 이러한 경우, 블록은 하기 단계들을 적용함으로써 단계(S250)에서 인코딩된다:

[0118] - 예측된 블록을 획득하기 위해 상기 블록의 샘플들에 대해 동일한 예측을 적용하는 단계;

[0119] - 상기 예측된 블록 및 상기 블록으로부터 잔차 블록을 결정하는 단계;

[0120] - 변환된 블록을 획득하기 위해 상기 잔차 블록의 샘플들에 대해 동일한 변환을 적용하는 단계;

[0121] - 상기 변환된 블록을 비트스트림에서 인코딩, 예를 들어, 양자화 및 엔트로피 코딩하는 단계.

[0122] 블록이 분리되는 방법을 표시하는 분리 정보가 또한 비트스트림에서 인코딩된다.

[0123] 방법은 단계(S260)에서 종료된다.

[0124] 유리하게, $3 \cdot 2^n$ 과 동일한 폭 또는 높이를 갖는 코딩 유닛은 본 원리들에 따라 인코더에 의해 선택될 수 있다. 이러한 경우, 3의 배수인 크기를 갖는 직사각형 블록들의 인트라 예측 및/또는 인터 예측이 수행될 수 있다. 추가적으로, 폭 또는 높이에서 크기 $3 \cdot 2^n$ 을 갖는 2D 변환 및 후속 변환 계수 엔트로피 코딩 프로세스가 또한 수행될 수 있다.

[0125] 이러한 추가적인 코딩 유닛 토폴로지들은 원래의 신호에 포함된 구조들 및 불연속들에 공간적으로 매칭하는 코딩 구조들을 가질 때 도움이 된다. 본 원리들에 따른 CTU의 CU들로의 파티셔닝의 예가 도 9에 표현된다. 이 도면에서, 쿼드-트리 파티셔닝이 실선들로 표현되고, 바이너리 파티셔닝(대칭적 및 비대칭적)이 파선들에 의해 표현된다. 이러한 파티셔닝(특히 회색 CU들)은 HEVC 또는 QTBT에서 정의된 바와 같이 파티셔닝으로 획득될 수 없다. 바이너리 분리의 경우 비대칭적 분리 모드를 추가하는 것은 원래의 신호에 포함된 구조들 및 불연속들에 더 양호하게 매칭함으로써 코딩 효율을 개선한다.

[0126] 바이너리 분리 정보를 비트스트림에서 인코딩하기 위한 예시적인 선택스가 표 1 및 표 2에 정의된다. 선택스는 정보의 다양한 항목들이 코딩되는 방법을 정의한다. QTBT에 정의된 바이너리 분리 모드들을 인코딩하기 위해 사용된 선택스 요소들 `btSplitFlag` 및 `btSplitOrientation`은 불변으로 남는다. 제1 선택스 요소 `btSplitFlag`는, 블록(CU 또는 CB)이 분리되는지 여부를 표시하고, 분리되면, 제2 선택스 요소 `btSplitOrientation`은 블록이 수평으로 분리되는지 또는 수직으로 분리되는지 여부를 표시한다. QTBT 선택스는 도 8의 새로운 분리 모드들을 표시하기 위해 새로운 선택스 요소들을 추가함으로써 수정된다. 표 1 및 표 2는 본 원리들에 따른 선택스의 예들이고, QTBT 선택스에 대한 수정들은 이탤릭체로 표시된다. 도 10은 표 1 및 표 2에 정의된 바와 같이 선택스

의 그래픽 표현이다. 추가된 선택 요소들은 도 10의 파선들로 표시된다. QTBT로부터의 선택 요소들은 도 10에 실선들로 표시된다.

[0127] 새로운 선택 요소들은, 바이너리 분리가 비대칭인지 여부를 표시하고, 비대칭이면, 비대칭 바이너리 분리 모드가 사용된다. 더 정확하게, horAsymmetricFlag 및 verAsymmetricFlag 선택 요소들은, 바이너리 분리된 블록에 대해, 각각 수평 분리의 경우 및 수직 분리의 경우, 분리 모드가 비대칭인지 여부를 표시한다. *horizontal*_{asymmetric}_{type} 선택 요소는, 수평으로 및 비대칭적으로 분리된 블록에 대해, HOR_{UP} 또는 HOR_{DOWN} 비대칭 바이너리 분리 모드에 따라 분리되었는지 여부를 표시한다. *vertical*_{asymmetric}_{type} 선택 요소는, 수직으로 및 비대칭적으로 분리된 블록에 대해, VER_{LEFT} 또는 VER_{RIGHT} 비대칭 바이너리 분리 모드에 따라 분리되었는지 여부를 표시한다.

[0128] 특정 실시예에서, 선택 요소들 horAsymmetricFlag 및 *horizontal*_{asymmetric}_{type} (각각 verAsymmetricFlag 및 *vertical*_{asymmetric}_{type})은 비대칭 수평 분리(각각 비대칭 수직 분리)가 허용되지 않는 경우 존재하지 않는다.

표 1

coding_binary_tree(x0, y0, width, height, cqtDepth) {	디스크립터
if(btSplitAllowed(x0,y0,width,height){	
bt_split_mode(x0,y0,width,height,cqtDepth)	
}	
if(btSplitFlag) {	
if(btSplitMode==HOR)	
{	
x1 = x0	
y1 = y0 + (height >> 1)	
sub_width_1 = sub_width_0 = width;	
sub_height_1 = sub_height_0 = (height >> 1)	
}	
if(btSplitMode==VER)	
{	
x1 = x0 + (width >> 1)	
y1 = y0	
sub_width_1 = sub_width_0 = (width >> 1)	
sub_height_1 = sub_height_0 = height	
}	
if(btSplitMode==HOR_UP)	
{	
x1 = x0	
y1 = y0 + (height >> 2)	
sub_width_1 = sub_width_0 = width	
sub_height_0 = (height >> 2)	
sub_height_1 = ((height *3) >> 2)	
}	
if(btSplitMode==HOR_DOWN)	
{	
x1 = x0	
y1 = y0 + ((height*3) >> 2)	
sub_width_1 = sub_width_0 = width	
sub_height_0 = ((height *3) >> 2)	
sub_height_1 = (height >> 2)	
}	
if(btSplitMode==VER_LEFT)	
{	
x1 = x0 + (width >> 2)	
y1 = y0	
sub_width_0 = width >> 2	
sub_width_1 = (width *3) >> 2	
sub_height_1 = sub_height_0 = height	
}	
if(btSplitMode==VER_RIGHT)	
{	
x1 = x0 + (width*3) >> 2	

[0129]

y1 = y0	
sub_width_0 = (width*3) >> 2	
sub_width_1 = width >> 2	
sub_height_1 = sub_height_0 = height	
}	
coding_binary_tree(x0, y0, sub_width, sub_height, cqtDepth)	
if(x1 < pic_width_in_luma_samples && y1 < pic_height_in_luma_samples)	
coding_binary_tree(x1, y1, sub_width, sub_height, cqtDepth)	
}	
} else	
coding_unit(x0, y0, width, height)	
}	

[0130]

표 2

bt_split_mode(x0,y0,width,height,cqtDepth){	디스크립터
if(btSplitAllowed(x0,y0,width,height){	
btSplitFlag	ae(v)
if (btSplitFlag == true){	
if(horizontalSplitAllowed && verticalSplitAllowed){	
btSplitOrientation	ae(v)
if(btSplitOrientation==HOR && horizontal_asymmetric_allowed){	
horAsymmetricSplitFlag	ae(v)
if(AsymmetricSplitFlag==true){	
horizontal_asymmetric_type	ae(v)
}	
}	
if(btSplitOrientation==VER && vertical_asymmetric_allowed){	
verAsymmetricSplitFlag	ae(v)
if(AsymmetricSplitFlag==true){	
vertical_asymmetric_type	ae(v)
}	
}	
}	
}	
}	

[0131]

[0132] 도 11은 표 1 및 표 2에 정의된 수정된 선택스를 사용하여 특정한 비제한적 실시예에 따라 현재 블록에 대한 바이너리 분리 정보를 인코딩하기 위한 방법의 흐름도를 표현한다.

[0133] 도 11에서는 오직 현재 블록의 바이너리 분리 모드 인코딩과 관련된 단계들만이 표현된다. 방법은 단계(S200)에서 시작한다. 단계(S2010)에서, 인코더(400)와 같은 송신기(200)가 현재 블록에 액세스한다.

[0134] 단계(S2020)에서, 송신기(200)는 현재 블록에 대해 바이너리 분리가 허용되는지 여부를 체크한다. 이는 통상적으로 현재 블록 폭(w) 또는 현재 블록 높이(h)가 제1 임계치(minBTSize)보다 높은지 여부를 체크하는 단계로 구성된다. 폭 또는 높이에서 통상적인 최소 직사각형 블록 크기는 minBTSize=4이다. 그렇지 않다면, 블록은 분리되지 않고, 분리와 관련된 어떠한 다른 선택스 요소도 코딩되지 않는다.

[0135] 그렇지 않으면(바이너리 분리가 허용되면), 현재 블록이 분리되는지 여부를 표시하기 위해 제1 플래그(btSplitFlag)가 단계(S2030)에서 인코딩된다. 예시적인 btSplitFlag는 현재 블록이 분리되는 것을 표시하기 위해 값 1 또는 참을 취하고, 현재 블록이 분리되지 않는 것을 표시하기 위해 값 거짓 또는 0을 취할 수 있다. 단계(S2040)에서, 송신기(200)는 현재 블록이 분리되는 것을 제1 플래그가 표시하는지 여부를 체크한다.

[0136] 단계(S2040)에서, 현재 블록이 분리되지 않는다고 제1 플래그가 표시하는 경우(예를 들어, btSplitFlag = 거짓 이기 때문에), 분리와 관련된 어떠한 다른 선택스 요소도 코딩되지 않는다.

[0137] 그렇지 않고, 단계(S2040)에서 현재 블록이 분리된다고 제1 플래그가 표시하는 경우(예를 들어, btSplitFlag = 참이기 때문에), 송신기(200)는 단계(S2050)에서 현재 블록에 대해 수평, 수직, 비대칭 수평 및 비대칭 수직 분리 모드들이 허용되는지 여부를 결정한다.

[0138] 일례로, 수직 분리 모드들은, 하기 조건이 참($w > \text{minBTSize} \ \&\& \ w \% 3 \neq 0$)이면 허용되고, 여기서 %는 모듈로 연산자이다. $w \% 3$ 은 w를 3으로 나눈 후의 나머지를 리턴한다. 통상적으로, minBTSize는 4와 동일할 수 있고, 수직 분리 모드들은 w가 8, 16 또는 32와 동일하면 허용된다.

[0139] 이러한 경우, vertical_allowed 파라미터는 값 1 또는 참으로 설정될 수 있다. 일례로, 수평 분리 모드들은, 하기 조건이 참($h > \text{minBTSize} \ \&\& \ h \% 3 \neq 0$)이면 허용된다. 이러한 경우, horizontal_allowed 파라미터는 값 1 또는 참으로 설정될 수 있다. 일례로, 수직 비대칭 분리 모드들은, 하기 조건이 참($\text{vertical_allowed} \ \&\& \ w > 8$)이면 허용된다. 일례로, 수평 비대칭 분리 모드들은, 하기 조건이 참($\text{horizontal_allowed} \ \&\& \ h > 8$)이면 허용된다. 따라서, 비대칭 수평 또는 수직 분리로부터 얻어지는 통상적으로 허용된 블록 폭 또는 높이는 $16/4=4$, $16*3/4=12$, $32/4=8$, $32*3/4=24$ 이다.

- [0140] 이러한 실시예에서, 블록이 이미 비대칭 방식으로 분리되면, 블록이 3의 배수인 크기를 갖는 방향에서 블록의 추가적인 분리를 허용하지 않는다. 또한, 블록 크기에 대한 일부 제한은 블록의 비대칭 분리를 허용하도록 주어진다. 통상적으로, 블록 크기는 8보다 커야 한다. 이러한 경우, 3의 배수인 최소 블록 크기는 12이다.
- [0141] 특정 실시예에서, 분리할 블록이 2의 배수인 크기를 갖도록 주어지면, 일 방향에서 $3 \cdot 2^n, n \geq 1$ 과 동일한 크기를 갖는 블록의 바이너리 분리가 그 방향을 따라 허용된다. 이는 블록의 분리가 3의 블록 크기를 초래할 수 있음을 의미한다. 예를 들어, 블록 12x4는 크기 6x4를 각각 갖는 2개의 서브-블록으로 분리될 수 있다.
- [0142] 다른 특정 실시예에서, 블록의 비대칭 분리는 그 크기가 8보다 크고 2의 거듭제곱인 경우 허용된다. 예를 들어, 이는 크기 8x4를 갖는 블록의, 크기들 6x4 및 2x4를 갖는 2개의 서브 코딩 유닛들로의 비대칭 분리를 허용한다. 이는 특히, QTBT 톨이 블록 크기들을 2까지 낮추도록 허용하는 크로마 성분에서 가능할 수 있다. 이러한 경우, 조건 $\text{vertical}_{\text{allowed}} \ \&\& \ w > 8$ 은 조건 $\text{vertical}_{\text{allowed}} \ \&\& \ w \geq 8$ 로 교체된다. 유사하게, 이러한 경우, 조건 $\text{horizontal}_{\text{allowed}} \ \&\& \ w > 8$ 은 조건 $\text{horizontal}_{\text{allowed}} \ \&\& \ w \geq 8$ 로 교체된다.
- [0143] 단계(S2060)에서, 송신기(200)는 수평 및 수직 분리 모드들 둘 모두가 허용되는지 여부를 체크한다. 허용되면, 바이너리 분리 배향(수평 또는 수직)을 표시하기 위해 단계(S2070)에서 선택스 요소(btSplitOrientation)가 코딩된다. 수평 또는 수직 분리가 허용되지 않으면, 이러한 선택스 요소(btSplitOrientation)는 코딩되지 않는다. 이러한 후자의 경우, 바이너리 분리 배향은 다음과 같이 디코더 측에서 추론된다: 수평 분리가 허용되지 않으면, btSplitOrientation은 VER과 동일한 것으로 추론된다. 그렇지 않고 수직 분리가 허용되지 않으면, btSplitOrientation은 HOR과 동일한 것으로 추론된다.
- [0144] 그 다음, 송신기(200)는 단계(S2080)에서(수평 분리의 경우) 또는 단계(S2090)에서(수직 분리의 경우), 비대칭 분리 모드가 이 배향을 따라 허용되는지 여부를 체크한다. 허용되지 않으면, 분리와 관련된 어떠한 추가적인 선택스 요소도 코딩되지 않는다. 이러한 후자의 경우, 바이너리 분리 모드는 btSplitOrientation 값에 따라 디코더 측에서 HOR 또는 VER로 추론된다.
- [0145] 그렇지 않으면, 현재 블록에 대해 비대칭 분리가 사용되는지 여부를 표시하기 위해 (단계(S2082 또는 S2092)에서) 선택스 요소(horAsymmetricSplitFlag 또는 verAsymmetricSplitFlag)가 인코딩된다.
- [0146] 일례로, horAsymmetricSplitFlag(각각 verAsymmetricSplitFlag)는 현재 블록이 비대칭적으로 분리되는 것을 표시하기 위해 값 1 또는 참을 취하고, 현재 블록이 비대칭적으로 분리되지 않는 것을 표시하기 위해 값 거짓 또는 0을 취할 수 있다. 단계(S2084)(각각 S2094)에서, 송신기(200)는 현재 블록이 비대칭적으로 수평으로(각각 수직으로) 분리되는지 여부를 체크한다.
- [0147] 비대칭 분리가 사용되지 않으면(예를 들어, 단계(S2084)에서 horAsymmetricSplitFlag가 거짓이거나 단계(S2094)에서 verAsymmetricSplitFlag가 거짓), 분리와 관련된 어떠한 추가적인 선택스 요소도 코딩되지 않는다. 비대칭 분리가 사용되면(단계(S2084)에서 horAsymmetricSplitFlag가 참이거나 단계(S2094)에서 verAsymmetricSplitFlag가 참), 현재 블록의 비대칭 분리 모드를 표시하기 위해 (단계(S2086 또는 S2096)에서) 다른 선택스 요소(horizontal_asymmetric_type 또는 vertical_asymmetric_type)가 인코딩된다.
- [0148] horizontal_asymmetric_type은 값들 HOR_UP 또는 HOR_DOWN을 취할 수 있고 vertical_asymmetric_type은 값들 VER_LEFT 또는 VER_RIGHT를 취할 수 있다.
- [0149] 방법은 단계(S2100)에서 종료된다.
- [0150] 디코딩 장치
- [0151] 도 12는 비제한적인 실시예에 따라 디코딩된 이미지 블록을 획득하기 위해 비트스트림으로부터 이미지 블록을 디코딩하도록 구성된 수신기(100)의 예시적인 아키텍처를 표현한다.
- [0152] 수신기(100)는 내부 메모리(1030)(예를 들어, RAM, ROM 및/또는 EPROM)와 함께 예를 들어 CPU, GPU 및/또는 DSP(Digital Signal Processor의 영문 두문자어)를 포함할 수 있는 하나 이상의 프로세서(들)(1000)를 포함한다. 수신기(100)는, 출력 정보를 디스플레이하고/하거나 사용자가 커맨드들 및/또는 데이터(예를 들어, 디코딩된 이미지 블록)를 입력할 수 있게 하도록 각각 적응된 하나 이상의 통신 인터페이스(들)(1010)(예를 들어, 키보드, 마우스, 터치패드, 웹캠); 및 수신기(100)의 외부에 있을 수 있는 전원(1020)을 포함할 수 있다. 수신기(100)는 또한 하나 이상의 네트워크 인터페이스(들)(미도시)를 포함할 수 있다. 디코더(1040)는 디코딩 기능들을 수행하기 위해 디바이스에 포함될 수 있는 모듈을 표현한다. 추가적으로, 디코더 모듈(1040)은 수신기(10

0)의 개별 요소로서 구현될 수 있거나 또는 본 기술분야의 통상의 기술자들에게 공지된 바와 같이 하드웨어 및 소프트웨어의 조합으로서 프로세서(들)(1000) 내에 통합될 수 있다.

- [0153] 비트스트림은 소스로부터 획득될 수 있다. 상이한 실시예들에 따르면, 소스는 다음과 같을 수 있지만, 이에 제한되는 것은 아니다:
- [0154] 로컬 메모리, 예를 들어, 비디오 메모리, RAM, 플래시 메모리, 하드 디스크;
- [0155] 저장 인터페이스, 예를 들어, 대용량 저장소, ROM, 광 디스크 또는 자기 지원부와의 인터페이스;
- [0156] 통신 인터페이스, 예를 들어, 유선 인터페이스(예를 들어, 버스 인터페이스, 광역 네트워크 인터페이스, 로컬 영역 네트워크 인터페이스) 또는 무선 인터페이스(예를 들어, IEEE 802.11 인터페이스 또는 블루투스 인터페이스); 및
- [0157] 픽처 캡처 회로(예를 들어, CCD(즉, Charge-Coupled Device) 또는 CMOS(즉, Complementary Metal-Oxide-Semiconductor)와 같은 센서).
- [0158] 상이한 실시예들에 따르면, 디코딩된 이미지 블록은 목적지, 예를 들어, 디스플레이 디스플레이 디바이스에 전송될 수 있다. 예로서, 디코딩된 이미지 블록은 원격 또는 로컬 메모리, 예를 들어, 비디오 메모리 또는 RAM, 하드 디스크에 저장된다. 변형으로, 디코딩된 이미지 블록은 저장 인터페이스, 예를 들어, 대용량 저장소, ROM, 플래시 메모리, 광 디스크 또는 자기 지원부와의 인터페이스에 전송되고 그리고/또는 통신 인터페이스, 예를 들어, 포인트 투 포인트 링크, 통신 버스, 포인트 투 멀티포인트 링크 또는 브로드캐스트 네트워크로의 인터페이스를 통해 송신된다.
- [0159] 특정한 비제한적인 실시예에 따르면, 수신기(100)는 메모리(1030)에 저장된 컴퓨터 프로그램을 더 포함한다. 컴퓨터 프로그램은, 수신기(100), 특히 프로세서(1000)에 의해 실행되는 경우, 수신기로 하여금 도 14 및/또는 도 15를 참조하여 설명된 디코딩 방법을 실행하게 하는 명령어들을 포함한다. 변형에 따르면, 컴퓨터 프로그램은 수신기(100)의 외부에서, 비일시적 디지털 데이터 지원부 상에, 예를 들어, HDD, CD-ROM, DVD, 판독-전용 및/또는 DVD 드라이브 및/또는 DVD 판독/기록 드라이브와 같은 외부 저장 매체 상에 저장되며, 이들 모두는 본 기술분야에 공지되어 있다. 따라서, 수신기(100)는 컴퓨터 프로그램을 판독하기 위한 메커니즘을 포함한다. 추가로, 수신기(100)는 하나 이상의 범용 직렬 버스(USB)-타입의 저장 디바이스들(예를 들어, "메모리 스틱들")에, 대응하는 USB 포트들(미도시)을 통해 액세스할 수 있다.
- [0160] 예시적이고 비제한적인 실시예들에 따르면, 수신기(100)는 다음과 같을 수 있지만, 이에 제한되는 것은 아니다:
- [0161] 모바일 디바이스;
- [0162] 통신 디바이스;
- [0163] 게임 디바이스;
- [0164] 셋탑 박스;
- [0165] TV 세트;
- [0166] 태블릿(또는 태블릿 컴퓨터);
- [0167] 랩탑;
- [0168] 비디오 플레이어, 예를 들어, 블루-레이 플레이어, DVD 플레이어;
- [0169] 디스플레이 및
- [0170] 디코딩 칩 또는 디코딩 디바이스/장치;
- [0171] 도 13은 도 14 및/또는 도 15의 디코딩 방법을 실행하도록 적응된 예시적인 비디오 디코더(300)의 블록도를 예시한다. 비디오 디코더(300)는 수신기(100)의 예이거나 또는 이러한 수신기(100)의 일부일 수 있다. 예시적인 디코더(300)에서, 비트스트림은 아래에서 설명되는 바와 같이 디코더 모듈들에 의해 디코딩된다. 비디오 디코더(300)는 일반적으로 도 6에 설명된 바와 같이 비디오 인코더(400)에 의해 수행되는 인코딩 패스에 역인 디코딩 패스를 수행한다.
- [0172] 특히, 디코더의 입력은 비디오 인코더(400)에 의해 생성될 수 있는 비디오 비트스트림을 포함한다. 비트스트림

은, 변환 계수들, 모션 벡터들 및 다른 코딩된 정보를 획득하기 위해 먼저 엔트로피 디코딩된다(모듈(330)). 잔차들을 디코딩하기 위해, 변환 계수들은 역양자화되고(모듈(340)) 역변환된다(모듈(350)). 그 다음, 디코딩된/재구성된 이미지 블록을 획득하기 위해, 디코딩된 잔차들은 예측된 샘플 블록(또한 예측자로 공지됨)과 결합된다(모듈(355)). 예측된 샘플 블록은 인트라 예측(모듈(360)) 또는 모션-보상된 예측(즉, 인터 예측)(모듈(375))으로부터 획득될 수 있다(모듈(370)). 루프 내 필터(모듈(365))는 재구성된 이미지에 대해 적용될 수 있다. 루프 내 필터는 디블로킹 필터 및 SAO 필터를 포함할 수 있다. 필터링된 이미지는 기준 픽처 버퍼(380)에 저장된다.

[0173] 디코딩 방법들

[0174] 도 14는 특정한 비제한적 실시예에 따라 비트스트림으로부터 이미지의 블록을 디코딩하기 위한 방법의 흐름도를 표현한다. 블록은 크기 $w \times h$ 이고, 여기서 w 는 이의 픽셀 단위의 폭이고 h 는 이의 픽셀 단위의 높이이다. 블록은 루마 샘플들의 블록, 크로마 샘플들의 블록 또는 루마 샘플들의 블록 및 크로마 샘플들의 2개의 블록들로 형성된 코딩 유닛 중 하나이다.

[0175] 방법은 단계(S100)에서 시작한다. 단계(S110)에서, 디코더(300)와 같은 수신기(100)는 디코딩될 블록을 표현하는 비트스트림 또는 이의 일부에 액세스한다.

[0176] 단계(S120)에서, 수신기는 분리 모드들의 세트에서 블록에 대한 분리 모드를 결정한다. 분리 모드는 예를 들어, 비트스트림으로부터 디코딩된다. 본 원리들에 따르면, 분리 모드들의 세트는 적어도 하나의 비대칭적 분리 모드를 포함한다. 도 8은 이러한 비대칭적 분리 모드들의 예를 도시한다. 제1 실시예에서, 적어도 하나의 비대칭적 분리 모드는 블록을 높이 $h/4$ 의 하나의 서브-블록 및 높이 $3h/4$ 의 하나의 서브-블록으로 수평으로 분리하는 것을 표시한다. 높이 $h/4$ 의 서브-블록은 높이 $3h/4$ 의 서브-블록의 위(모드 HOR_UP) 또는 아래(모드 HOR_DOWN)에 있다. 다른 실시예에서, 적어도 하나의 비대칭적 분리 모드는 블록을 폭 $w/4$ 의 하나의 서브-블록 및 폭 $3w/4$ 의 하나의 서브-블록으로 수직으로 분리하는 것을 표시한다. 폭 $w/4$ 의 서브-블록은 폭 $3w/4$ 의 서브-블록의 좌측(모드 VER_LEFT) 또는 우측(모드 VER_RIGHT)에 있다. 특정 실시예에서, 분리 모드들의 세트는 도 8에 도시된 모든 분리 모드들을 포함한다. 변형 실시예에서, 적어도 하나의 비대칭 분리 모드는, 2의 거듭제곱이 아닌 크기를 갖는, 예를 들어, 폭 또는 높이에서 이들의 크기가 $3 \cdot 2^n$ 과 동일하고 n 은 정수인 서브-블록들로 수평으로 또는 수직으로 블록을 분리하는 것을 표시한다.

[0177] 변형에서, 분리 모드들의 세트는 적어도 하나의 비대칭적 분리 모드에 추가로, QTBT에 의해 정의되고 도 4에 도시된 분리 모드들을 포함한다. 블록이 추가로 분리되지 않음을 분리 모드가 표시하는 경우(NO_SPLIT), 블록은 단계(S150)에서 디코딩된다.

[0178] 그렇지 않으면, 단계(S130)에서, 수신기는 결정된 분리 모드에 응답하여 액세스된 블록을 적어도 2개의 서브-블록들로 분리한다.

[0179] 단계(S140)에서, 송신기는 적어도 2개의 서브-블록들 각각을 디코딩한다. 적어도 2개의 서브-블록들을 디코딩하는 단계는 적어도 2개의 서브-블록들 각각에 대해 단계들(S120 내지 S140)을 회귀적으로 적용하는 단계를 포함할 수 있다. 예를 들어, 분리 모드 NO_SPLIT가 단계(S120)에서 결정되기 때문에, 회귀적 분리는 블록이 추가로 분리되지 않을 때 종료된다. 이러한 경우, 블록은 하기 단계들을 적용함으로써 단계(S150)에서 디코딩된다:

[0180] - 상기 서브-블록에 대한 변환된 블록을 디코딩, 예를 들어, 엔트로피 디코딩 및 역양자화하는 단계;

[0181] - 잔차 블록을 획득하기 위해 상기 디코딩된 변환된 블록의 샘플들에 대해 동일한 변환(인코더 측에서 사용된 것의 역)을 적용하는 단계;

[0182] - 예측된 블록을 획득하기 위해 상기 서브-블록의 샘플들에 대해 동일한 예측을 적용하는 단계;

[0183] - 상기 예측된 블록 및 상기 잔차 블록으로부터 상기 서브-블록을 재구성하는 단계.

[0184] 방법은 단계(S160)에서 종료된다.

[0185] 블록이 분리되는 방법을 표시하는 분리 정보가 또한 비트스트림으로부터 디코딩된다.

[0186] 도 15는 표 1 및 표 2에 정의된 수정된 선택스를 사용하여 특정한 비제한적 실시예에 따른 바이너리 분리의 경우 현재 블록의 분리 모드를 디코딩하기 위한 방법의 흐름도를 표현한다.

[0187] 도 15에서는 오직 현재 블록의 바이너리 분리 모드 디코딩과 관련된 단계들만이 표현된다. 방법은 단계(S100

0)에서 시작한다. 단계(S1010)에서, 디코더(300)와 같은 수신기(100)는 디코딩될 현재 블록을 표현하는 비트스트림 또는 이의 일부에 액세스한다.

- [0188] 단계(S1020)에서, 수신기(100)는 디코딩될 현재 블록에 대해 바이너리 분리가 허용되는지 여부를 체크한다. 이는 통상적으로 현재 블록 폭(w) 또는 현재 블록 높이(h)가 제1 임계치(minBTSize)보다 높은지 여부를 체크하는 단계로 구성된다. 폭 또는 높이에서 통상적인 최소 직사각형 블록 크기는 minBTSize=4이다. 바이너리 분리가 허용되지 않으면, 블록은 분리되지 않고(단계(S1045)에서 btSplitMode는 NO_SPLIT로 설정됨) 분리와 관련된 어떠한 다른 선택스 요소도 디코딩되지 않는다.
- [0189] 그렇지 않으면(바이너리 분리가 허용되면), 현재 블록이 분리되는지 여부를 표시하는 단계(S1030)에서 비트스트림으로부터 제1 플래그(btSplitFlag)가 디코딩된다.
- [0190] 단계(S1040)에서, 수신기(100)는 현재 블록이 분리되는지 여부를 체크한다. 현재 블록이 분리되지 않으면(예를 들어, 단계(S1040)에서 btSplitFlag가 거짓), 분리와 관련된 어떠한 다른 선택스 요소도 디코딩되지 않는다. 단계(S1045)에서 btSplitMode는 NO_SPLIT로 설정된다.
- [0191] 현재 블록이 분리되는 경우(단계(S1040)에서 btSplitFlag가 참), 수신기(100)는 단계(S1050)에서 현재 블록에 대해 수평, 수직, 비대칭 수평 및 비대칭 수직 분리 모드들이 허용되는지 여부를 결정한다. 단계(S1050)는 도 11의 단계(S2050)와 동일하다.
- [0192] 단계(S1060)에서, 수신기(100)는 수평 및 수직 분리 모드들 둘 모두가 허용되는지 여부를 체크한다. 허용되면, 바이너리 분리 배향(수평 또는 수직)을 표시하기 위해 단계(S1070)에서 선택스 요소(btSplitOrientation)가 디코딩된다. 수평 또는 수직 분리가 허용되지 않으면, 이러한 선택스 요소(btSplitOrientation)는 디코딩되지 않는다. 이러한 후자의 경우, 바이너리 분리 배향은 단계(S1065)에서 추론된다. 수평 분리가 허용되지 않으면, btSplitOrientation은 VER과 동일한 것으로 추론된다. 그렇지 않고 수직 분리가 허용되지 않으면, btSplitOrientation은 HOR과 동일한 것으로 추론된다.
- [0193] 그 다음, 수신기(100)는 단계(S1080)에서(수평 분리의 경우) 또는 단계(S1090)에서(수직 분리의 경우), 비대칭 분리 모드가 이 배향을 따라 허용되는지 여부를 체크한다. 허용되지 않으면, 분리와 관련된 어떠한 추가적인 선택스 요소도 디코딩되지 않는다. 이러한 후자의 경우, 바이너리 분리 모드는 btSplitOrientation 값에 따라 HOR 또는 VER로 추론된다(단계(S1085)에서 btSplitMode는 btSplitOrientation로 설정된다).
- [0194] 그렇지 않으면, 현재 블록에 대해 비대칭 분리가 사용되는지 여부를 표시하는 선택스 요소(horAsymmetricSplitFlag 또는 verAsymmetricSplitFlag)가 (단계(S1082 또는 S1092)에서) 디코딩된다. 단계(S1084)(각각 S1094)에서, 수신기(100)는 현재 블록이 비대칭적으로 수평으로(각각 수직으로) 분리되는지 여부를 체크한다.
- [0195] 비대칭 분리가 사용되지 않으면(horAsymmetricSplitFlag 또는 verAsymmetricSplitFlag가 참이 아님), 분리와 관련된 어떠한 추가적인 선택스 요소도 디코딩되지 않는다. 비대칭 분리가 사용되면(horAsymmetricSplitFlag 또는 verAsymmetricSplitFlag가 참), 현재 블록에 대한 비대칭 분리 모드를 표시하는 선택스 요소(horizontal_asymmetric_type 또는 vertical_asymmetric_type)가 (단계(S1086 또는 S1096)에서) 디코딩된다.
- [0196] horizontal_asymmetric_type은 값들 HOR_UP 또는 HOR_DOWN을 취할 수 있고 vertical_asymmetric_type은 값들 VER_LEFT 또는 VER_RIGHT를 취할 수 있다.
- [0197] horizontal_asymmetric_type이 값 HOR_UP를 취하면, 바이너리 분리 모드(btSplitMode)는 HOR_UP와 동일하게 설정된다.
- [0198] horizontal_asymmetric_type이 값 HOR_DOWN를 취하면, 바이너리 분리 모드는 HOR_DOWN과 동일하게 설정된다.
- [0199] vertical_asymmetric_type이 값 VER_LEFT를 취하면, 바이너리 분리 모드는 VER_LEFT와 동일하게 설정된다.
- [0200] horizontal_asymmetric_type이 값 VER_LEFT를 취하면, 바이너리 분리 모드는 VER_LEFT와 동일하게 설정된다.
- [0201] 방법은 단계(S1100)에서 종료된다.
- [0202] 선택스 요소들의 엔트로피 코딩 및 디코딩
- [0203] 새로운 선택스 요소들 verAsymmetricSplitFlag 및 horAsymmetricSplitFlag가 상이한 방식으로 인코딩될 수 있다. 제1 실시예에서, verAsymmetricSplitFlag(각각 horAsymmetricSplitFlag)는 단일 비트로 비트-스트림에서

코딩된다. 이는, 오직 대칭적 및 비대칭적 분리 모드들이 동일한 확률을 갖는 경우, 즉, verAsymmetricSplitFlag(각각 horAsymmetricSplitFlag)가 참과 동일하도록 0.5의 확률을 갖는 경우에만 최적이다. 그러한 경우, 이러한 선택 요소들에 의해 각각 운반되는 정보의 평균량은 1 비트와 동일하고, 1 비트에 의한 고정-길이 코딩이 최적이다.

[0204] 가장 실용적인 경우들에서, 대칭적 및 비대칭적 분리 모드들은 동일 확률이 아니기 쉽다. 따라서, 다른 실시예에 따르면, 선택 요소들 verAsymmetricSplitFlag 및 horAsymmetricSplitFlag를 인코딩하기 위해 콘텍스트-기반 산술 코딩이 사용된다. 콘텍스트-기반 산술 코딩은 콘텍스트 모델링 및 산술 코딩을 포함한다. 특정 실시예들에서, 이는 콘텍스트 이전에 선택 요소들의 이진화를 모델링하는 것을 포함할 수 있다. 이진화 및 산술 코딩은 과거의 인코딩 방식들(예를 들어, H.264 및 HEVC)로부터 널리 공지되어 있다. 콘텍스트 모델링은 일부 콘텍스트에 기초하여 각각의 선택 요소의 확률을 추정하고, 인코더 및 디코더 측 상에서 대칭적으로 적용된다. 엔트로피 코딩에 대해 개시된 하기 실시예들은 또한 엔트로피 디코딩에 적용된다. 따라서 새로운 선택 요소들 horAsymmetricFlag 및 verAsymmetricFlag에 대해 새로운 콘텍스트들이 정의된다.

[0205] 이러한 실시예에서, 각각의 선택 요소 horAsymmetricFlag 및 verAsymmetricFlag와 각각 연관된 2개의 간단한 콘텍스트들이 사용된다. 이러한 콘텍스트들 각각은 관련된 선택 요소 자체에 의해 과거에(즉, 코딩 순서에서 현재 블록에 선행하는 블록들을 인코딩할 때) 취해진 값들에만 의존한다. 따라서, 고려되는 선택 요소에 의해 표현된 랜덤 변수의 1차 엔트로피에 가까운 정보의 평균량에 걸쳐 이러한 비대칭 분리 모드 정보를 인코딩할 수 있다. 따라서, 이러한 선택 요소들은, 이러한 플래그들의 통계적 분포들이 균일하지 않은 경우, 평균적으로 1 미만의 비트에 걸쳐 인코딩된다. 이러한 콘텍스트-기반 코딩의 변형에서, horAsymmetricFlag 및 verAsymmetricFlag에 대해 사용되는 콘텍스트들은 동일하다. 실제로, 이러한 2개의 선택 요소들의 코딩은 동일한 콘텍스트를 공유할 수 있다. horAsymmetricFlag가 참 또는 거짓이어서 현재 블록이 수평 배향에서 바이너리 분리된 것을 아는 확률들이, verAsymmetricFlag가 참 또는 거짓이어서 현재 블록이 수직 배향에서 바이너리 분리된 것을 아는 확률과 가깝다고 가정하면 효율적인 것이다.

[0206] 제3 실시예에서, 이러한 선택 요소들 verAsymmetricSplitFlag 및 horAsymmetricSplitFlag는 선행 이웃 블록들에서 이미 판정된 분리 구성들과 관련된 하나 이상의 콘텍스트를 사용하여 엔트로피 코딩된다. 현재 블록에 대해 비대칭적 분리가 사용되는 경우, 2개의 서브-블록들 중 하나의 서브-블록은 폭 또는 높이에서 현재 블록 크기의 1/4과 동일한 크기를 갖는다. 이는, 현재 블록의 원래의 신호가 일부 공간적 활동 또는 불연속을 포함하는 것, 및 코딩 효율의 관점에서 현재 블록을 분리하는 것이 가치 있음을 의미한다. 이러한 높은 공간적 활동 또는 불연속은 또한, 코딩 순서에서 현재 블록에 선행하는 이웃 블록들에서 존재하기 쉽다.

[0207] 콘텍스트들은, 고려되는 분리 배향을 따라 현재 블록 크기의 1/4보다 작거나 그와 동일한 크기를 갖는 서브-블록들의 수의 함수로서 비대칭 분리 선택 요소들을 인코딩하기 위해 정의된다. 이는 수평 및 수직 비대칭 분리 콘텍스트들에 대해 각각 하기의 형태를 취한다:

[0208]
$$nbQuarterHeightSubCU = \left\lceil \frac{height}{4} \right\rceil$$
 보다 작거나 그와 동일한 높이를 갖는 서브블록의 수로 결정하며, 여기서 $height$ = 현재 블록의 높이;

[0209] horAsymmetricSplitCtxt=(nbQuarterHeightSubCU>0?1:0)로 결정함 (1)

[0210] 수식 (1)에 따르면, horAsymmetricSplitFlag 선택 요소를 코딩하기 위해 2개의 상이한 콘텍스트들이 사용된다. 값 nbQuarterHeightSubCU가 엄격하게 제로보다 크면, horAsymmetricSplitFlag에 대응하는 바이너리 심볼을 인코딩하기 위해 제1 콘텍스트(인덱스 0을 가짐)가 사용된다. 반대의 경우에, 값 nbQuarterHeightSubCU가 제로와 동일하면, horAsymmetricSplitFlag에 대응하는 바이너리 심볼을 인코딩하기 위해 제2 콘텍스트(인덱스 1을 가짐)가 사용된다.

[0211] 좌측 이웃 블록은 현재 블록의 높이보다 크거나 그와 동일한 높이를 갖는 블록에 대응하고, 이의 우측 경계는 현재 블록의 좌측 경계의 x-좌표 마이너스 1과 동일한 x-좌표를 갖는다. 좌측 이웃 블록은 높이에서 하나의 또는 몇몇 서브-블록들로 분리될 수 있다.

[0212]
$$nbQuarterWidthSubCU = \left\lceil \frac{width}{4} \right\rceil$$
 보다 작거나 그와 동일한 폭을 갖는 서브블록의 수로 결정하며, 여기서 $width$ = 현재 블록의 폭;

- [0213] $verAsymmetricSplitCtxt=(nbQuarterWidthSubC>0?1:0)$ 로 결정함 (2)
- [0214] 수식 (2)에 따르면, $verAsymmetricSplitFlag$ 신택스 요소를 코딩하기 위해 2개의 상이한 컨텍스트들이 사용된다. 값 $nbQuarterWidthSubCU$ 가 엄격하게 제로보다 크면, $verAsymmetricSplitFlag$ 에 대응하는 바이너리 심볼을 인코딩하기 위해 제1 컨텍스트(인덱스 0을 가짐)가 사용된다. 반대의 경우에, 값 $nbQuarterWidthSubCU$ 가 제로와 동일하면, $verAsymmetricSplitFlag$ 에 대응하는 바이너리 심볼을 인코딩하기 위해 제2 컨텍스트(인덱스 1을 가짐)가 사용된다.
- [0215] 상단 이웃 블록은 현재 블록의 폭보다 크거나 그와 동일한 폭을 갖는 블록에 대응하고, 이의 바닥 경계는 현재 블록의 상단 경계의 y-좌표 마이너스 1과 동일한 y-좌표를 갖는다. 상단 이웃 블록은 폭에서 하나의 또는 몇몇 서브-블록들로 분리될 수 있다.
- [0216] 다른 실시예에 따르면, 비대칭 분리 플래그들 $verAsymmetricSplitFlag$ 및 $horAsymmetricSplitFlag$ 를 코딩하기 위해 사용되는 컨텍스트는 각각 폭 및 높이에서 현재 블록의 크기의 절반보다 큰 크기를 갖는 이웃 블록의 존재의 함수로서 결정된다. 이는 하기 형태를 취한다. 먼저, 요소 $horAsymmetricSplitFlag$ 의 코딩을 코딩하기 위해 사용되는 컨텍스트는 다음과 같이 결정된다:
- [0217] 현재 CU의 상단-좌측 코너의 좌측에 위치한 블록이 현재 블록의 높이의 절반보다 높은 높이를 갖는지 여부를 결정함.
- [0218] 현재 CU의 바닥-좌측 코너의 좌측에 위치한 블록이 현재 블록의 높이의 절반보다 큰 높이를 갖는지 여부를 결정함.
- [0219] 상기 테스트들 중 하나가 긍정이면, $horAsymmetricSplitCtxt$ 에 값 0이 부여된다.
- [0220] 그렇지 않으면, $horAsymmetricSplitCtxt$ 에 값 1이 부여된다.
- [0221] 유사하게, 요소 $verAsymmetricSplitFlag$ 의 코딩을 코딩하기 위해 사용되는 컨텍스트는 다음과 같이 결정된다:
- [0222] 현재 CU의 상단-좌측 코너의 상단에 위치한 블록이 현재 블록의 폭의 절반보다 높은 폭을 갖는지 여부를 결정함.
- [0223] 현재 CU의 상단-우측 코너의 상단에 위치한 블록이 현재 블록의 폭의 절반보다 큰 폭을 갖는지 여부를 결정함.
- [0224] 상기 테스트들 중 하나가 긍정이면, $verAsymmetricSplitCtxt$ 에 값 0이 부여된다.
- [0225] 그렇지 않으면, $verAsymmetricSplitCtxt$ 에 값 1이 부여된다.
- [0226] 추가적인 실시예에 따르면, 슬라이스 타입(인트라 또는 인터)에 따라 $horAsymmetricSplitFlag$ 및 $verAsymmetricSplitFlag$ 를 코딩/디코딩하기 위해 일부 별개의 컨텍스트들이 사용된다. 인터 슬라이스에서, 상기 컨텍스트 결정 방법이 사용된다. 인트라 슬라이스에서, $horAsymmetricSplitFlag$ 및 $verAsymmetricSplitFlag$ 에 대해 각각 단일 컨텍스트가 사용되어, 이러한 플래그들을 코딩/디코딩하는 경우 이용 가능한 컨텍스트 블록 크기 정보와 무관하게, 이러한 2개의 플래그들의 통계적 거동을 캡처한다.
- [0227] 이러한 실시예의 이점은 특히 인터 슬라이스들에서, 코딩된 플래그들과 이웃 블록 정보 사이의 통계적 의존도를 더 양호하게 모델링하는 것이며, 이는 이러한 플래그들의 코딩과 연관된 레이트 비용을 감소시킨다.
- [0228] 추가적인 실시예에 따르면, 선행 실시예들 중 하나에서 제안된 컨텍스트는 현재 블록과 연관된 쿼드-트리 플러스 바이너리 트리 깊이 값의 함수로서 인덱싱된다. 따라서, CTU의 쿼드-트리 플러스 바이너리-트리 표현에서 각각의 깊이 레벨에 대해 별개의 세트의 컨텍스트들이 사용된다. 유리하게, 코딩/디코딩되고 있는 신택스 요소들의 통계적 거동을 모델링하기 위해 더 정밀한 컨텍스트가 사용된다.
- [0229] 제3 실시예에서, 신택스 요소들 $horAsymmetricSplitFlag$ 및 $verAsymmetricSplitFlag$ 의 컨텍스트 적응형 코딩을 수행하기 위해 2개의 상이한 컨텍스트들의 세트가 사용된다. 따라서, 컨텍스트들의 세트들은 다음과 같이 표기된다:
- [0230] - $horAsymmetricSplitFlagCtx[0..1]$
- [0231] - $verAsymmetricSplitFlagCtx[0..1]$
- [0232] 본 실시예에서, 컨텍스트들은 쿼드-트리 플러스 바이너리 트리 CTU 표현에서 프로세싱되는 블록의 깊이에 의해

인택싱된다. 따라서, 콘텍스트들의 2개의 상기 세트들은 깊이 레벨에 기초하여 추가로 인택싱된다:

- [0233] - horAsymmetricSplitFlagCtx[0...maxDepth][0..1]
- [0234] - verAsymmetricSplitFlagCtx[0...maxDepth][0..1]
- [0235] 여기서 maxDepth는 블록의 최대 깊이를 표현한다. 여기서 깊이는, 현재 블록의 쿼드-트리 및 바이너리-트리 깊이들의 합을 의미한다.
- [0236] 추가적인 실시예에 따르면, 선행 실시예들 중 하나에서 제안된 콘텍스트는 블록의 형상의 함수, 즉, 폭 및 높이로서 인택싱된다. 이는, 가능한 코딩 크기의 각각의 튜플(w, h)에 대해 콘텍스트들의 별개의 세트가 사용됨을 의미한다. 이러한 실시예의 이점은 코딩/디코딩되고 있는 선택스 요소들의 통계적 거동을 모델링하기 위한 추가적으로 개선된 콘텍스트이다. 유리하게, 코딩/디코딩되고 있는 선택스 요소들의 통계적 거동을 모델링하기 위해 더 정밀한 콘텍스트가 사용된다.
- [0237] 본 실시예에서, 콘텍스트들은 폭의 log2에 의해 및 블록의 높이의 log2에 의해 인택싱된다.
- [0238] - horAsymmetricSplitFlagCtx[0...maxLog2CUHeight][0..1]
- [0239] - verAsymmetricSplitFlagCtx[0...maxLog2CUWidth][0..1]
- [0240] 여기서 maxLog2CUWidth 및 maxLog2CUHeight는 2의 거듭제곱과 동일한 최대 가능 블록 폭 및 블록 높이의 log2를 표현한다.
- [0241] 콘텍스트들은 2의 거듭제곱과 동일한 블록 크기들과 연관되는데, 이는 비대칭적 분리가 오직 2의 거듭제곱과 동일한 블록 크기에 적용될 수 있기 때문이다.
- [0242] 새로운 선택스 요소들 horizontal_asymmetric_type 또는 vertical_asymmetric_type은 상이한 방식으로 인코딩될 수 있다. 제1 실시예에서, horizontal_asymmetric_type(각각 vertical_asymmetric_type)은 단일 비트로 비트-스트림에서 코딩된다.
- [0243] 이는, 오직 horizontal_asymmetric_type 또는 vertical_asymmetric_type 선택스 요소들이 동일한 확률을 가질 때에만 최적이다.
- [0244] 다른 실시예에 따르면, 선택스 요소들 horizontal_asymmetric_type 또는 vertical_asymmetric_type을 인코딩하기 위해 콘텍스트-기반 산술 코딩이 사용된다. 이러한 실시예에서, 각각의 선택스 요소 horizontal_asymmetric_type 또는 vertical_asymmetric_type과 각각 연관된 2개의 간단한 콘텍스트들이 사용된다. 이러한 콘텍스트들 각각은 관련된 선택스 요소 자체에 의해 과거에(즉, 코딩 순서에서 선행하는 블록들을 인코딩할 때) 취해진 값들에만 의존한다. 이러한 콘텍스트-기반 코딩의 변형에서, horizontal_asymmetric_type 및 vertical_asymmetric_type에 대해 사용되는 콘텍스트들은 동일할 수 있다. 따라서, 이러한 2개의 선택스 요소들의 코딩은 동일한 콘텍스트를 공유한다. horizontal_asymmetric_type이 참 또는 거짓이어서 현재 블록이 수평 배향에서 비대칭적으로 바이너리 분리된 것을 아는 확률들이, vertical_asymmetric_type이 참 또는 거짓이어서 현재 블록이 수직 배향에서 비대칭적으로 바이너리 분리된 것을 아는 확률과 가깝다고 가정하면 효율적일 것이다.
- [0245] 제3 실시예에서, 이러한 선택스 요소들 horizontal_asymmetric_type 또는 vertical_asymmetric_type은 도 16 및 도 17에 예시된 바와 같이 선행 이웃 블록들에서 이미 판정된 분리 구성들과 관련된 일부 콘텍스트 정보의 함수로서 엔트로피 코딩된다. 수평 비대칭 분리에서, 현재 블록의 좌측 상의 이전 블록에 분리 프론티어(도 16에서 F1)가 존재하고, 수평 비대칭 분리의 선택을 통해 현재 블록에 나타날 수평 프론티어와 공간적으로 일치하면, 현재 블록에서 이러한 비대칭 분리의 사용 확률은 증가된다.
- [0246] 이러한 원리는 도 16에 예시되어 있다. 도면의 좌측에, 후보 HOR_UP 분리 모드가 현재 블록에서 평가되고 있는 한편, 좌측의 이웃 블록은 분리되지 않았다. 그 경우 현재 블록에 대한 HOR_UP 분리 모드의 사용 확률은 도 16의 우측에 도시된 경우에서보다 낮다. 실제로, 이러한 후자의 경우, HOR_UP 분리 모드의 선택에 의해 현재 블록에서 전파될 수평 프론티어를 포함하는 분리 구성으로 이웃 블록이 인코딩되었다. 현재 블록에서 HOR_UP 분리 모드를 선택하는 것은 도 16의 좌측에 도시된 이전 경우에서보다 이러한 경우에서 발생하기 더 쉽다.
- [0247] 이러한 원리는 또한 수직 비대칭 분리의 경우 도 17에 예시되어 있다. VER_RIGHT 분리 모드는 도 17의 좌측의 경우에서보다 도 17의 우측에 도시된 경우에서 현재 블록을 인코딩하기 위해 사용될 더 높은 확률을 갖는데, 이

는, 현재 블록 위에 위치한 이웃 블록으로부터의 수직 프론티어 F2를 현재 블록으로 전파하기 때문이다.

[0248] 이러한 원리에 기초하여, 선택스 요소 $horizontal_{asymmetrictype}$ 을 코딩 및 디코딩하기 위해 하기 콘텍스트들이 사용된다:

[0249] HOR_UP 비대칭 타입으로부터 얻어질 프론티어와 정렬되는 좌측 이웃 블록에서의 프론티어가 존재하는지 여부를 결정함. 결과들에 따라, $hor_{upalignedboundary}$ 는 참 또는 거짓으로 설정된다.

[0250] HOR_DOWN 비대칭 타입으로부터 얻어질 프론티어와 정렬되는 좌측 이웃 블록에서의 프론티어가 존재하는지 여부를 결정함. 결과들에 따라, $hor_{downalignedboundary}$ 는 참 또는 거짓으로 설정된다.

[0251] 콘텍스트 인덱스를 다음과 같이 값들 (0,1,2) 중에서 결정함:

$$hor_{asymmetrictypecontext} = hor_{upalignedboundary} ? 1 : (hor_{downalignedboundary} : 2 ? 0) \quad (3)$$

[0253] 수식 (3)에 따르면, $horizontal_{asymmetrictype}$ 선택스 요소를 코딩하기 위해 3개의 상이한 콘텍스트들이 사용된다. 값 $hor_{upalignedboundary}$ 가 참과 동일하면, $horizontal_{asymmetrictype}$ 에 대응하는 바이너리 심볼을 인코딩하기 위해 제1 콘텍스트(인덱스 1을 가짐)가 사용된다. 반대의 경우에, 값 $hor_{downalignedboundary}$ 가 참과 동일하면, $horizontal_{asymmetrictype}$ 에 대응하는 바이너리 심볼을 인코딩하기 위해 제2 콘텍스트(인덱스 2를 가짐)가 사용된다. 마지막으로, 값들 $hor_{upalignedboundary}$ 및 $hor_{downalignedboundary}$ 둘 모두가 거짓과 동일하면, $horizontal_{asymmetrictype}$ 에 대응하는 바이너리 심볼을 인코딩하기 위해 제3 콘텍스트(인덱스 0을 가짐)가 사용된다. 선택스 요소 $vertical_{asymmetrictype}$ 을 코딩 및 디코딩하기 위해 하기 콘텍스트들이 사용된다:

[0254] VER_LEFT 비대칭 타입으로부터 얻어질 프론티어와 정렬되는 상부 이웃 블록에서의 프론티어가 존재하는지 여부를 결정함. 결과들에 따라, $ver_{leftalignedboundary}$ 는 참 또는 거짓으로 설정된다.

[0255] VER_RIGHT 비대칭 타입으로부터 얻어질 프론티어와 정렬되는 상부 이웃 블록에서의 프론티어가 존재하는지 여부를 결정함. 결과들에 따라, $ver_{rightalignedboundary}$ 는 참 또는 거짓으로 설정된다.

[0256] 콘텍스트 인덱스를 다음과 같이 값들 (0,1,2) 중에서 컴퓨팅함:

$$ver_{asymmetrictypecontext} = ver_{leftalignedboundary} ? 1 : (ver_{rightalignedboundary} : 2 ? 0) \quad (4)$$

[0258] 수식 (3)에 따르면, $vertical_{asymmetrictype}$ 선택스 요소를 코딩하기 위해 3개의 상이한 콘텍스트들이 사용된다. 값 $ver_{leftalignedboundary}$ 가 참과 동일하면, $vertical_{asymmetrictype}$ 에 대응하는 바이너리 심볼을 인코딩하기 위해 제1 콘텍스트(인덱스 1을 가짐)가 사용된다. 반대의 경우에, 값 $ver_{rightalignedboundary}$ 가 참과 동일하면, $vertical_{asymmetrictype}$ 에 대응하는 바이너리 심볼을 인코딩하기 위해 제2 콘텍스트(인덱스 2를 가짐)가 사용된다. 마지막으로, 값들 $ver_{leftalignedboundary}$ 및 $ver_{rightalignedboundary}$ 둘 모두가 거짓과 동일하면, $vertical_{asymmetrictype}$ 에 대응하는 바이너리 심볼을 인코딩하기 위해 제3 콘텍스트(인덱스 0을 가짐)가 사용된다.

- [0259] 다른 실시예에 따르면, 선택 요소들 `horizontalAsymmetricType` 또는 `verticalAsymmetricType`의 코딩은 각각 다음과 같이 결정되는 컨텍스트를 이용한다. 먼저, 플래그 `horizontalAsymmetricType`을 코딩하기 위해 사용되는 컨텍스트 `horAsymmetricTypeContext`는 다음과 같이 결정된다:
- [0260] - 현재 블록의 상단-좌측 코너의 좌측에 위치한 블록의 높이를 획득함. 이는 값 `topLeftHeight`를 부여한다.
 - [0261] - 현재 블록의 바닥-좌측 코너의 좌측에 위치한 블록의 높이를 획득함. 이는 값 `bottomLeftHeight`를 부여한다.
 - [0262] - `topLeftHeight`가 `bottomLeftHeight`와 동일하면 `horAsymmetricTypeContext`에는 값 0이 부여된다.
 - [0263] - 그렇지 않고 `topLeftHeight < bottomLeftHeight`이면 `horAsymmetricTypeContext`에는 값 1이 부여된다.
 - [0264] - 그렇지 않으면(`topLeftHeight > bottomLeftHeight`인 경우) `horAsymmetricTypeContext`에는 값 2가 부여된다.
- [0265] 따라서 플래그 `horizontalAsymmetricType`을 코딩/디코딩하기 위해 3개의 컨텍스트들이 사용된다.
- [0266] 둘째로, 플래그 `verticalAsymmetricType`을 코딩하기 위해 사용되는 컨텍스트 `verAsymmetricTypeContext`는 다음과 같이 결정된다:
- [0267] - 현재 블록의 상단-좌측 코너의 상단에 위치한 블록의 폭을 획득함. 이는 값 `topLeftWidth`를 부여한다.
 - [0268] - 현재 블록의 상단-우측 코너의 상단에 위치한 블록의 폭을 획득함. 이는 값 `topRightWidth`를 부여한다.
 - [0269] - `topLeftWidth`가 `topRightWidth`와 동일하면 `verticalAsymmetricTypeContext`에는 값 0이 부여된다.
 - [0270] - 그렇지 않고 `topLeftWidth > topRightWidth`이면 `verAsymmetricTypeContext`에는 값 1이 부여된다.
 - [0271] - 그렇지 않으면(`topLeftWidth < topRightWidth`인 경우) `verAsymmetricTypeContext`에는 값 2가 부여된다.
- [0272] 따라서 플래그 `verticalAsymmetricType`을 코딩/디코딩하기 위해 3개의 컨텍스트들이 사용된다.
- [0273] 추가적인 실시예에 따르면, 슬라이스 타입(인트라 또는 인터)에 따라 `horizontalAsymmetricType` 및 `verticalAsymmetricType`을 코딩/디코딩하기 위해 일부 별개의 컨텍스트들이 사용된다. 인터 슬라이스에서, 상기 컨텍스트 결정 방법이 사용된다. 인트라 슬라이스에서, 플래그들 `horizontalAsymmetricType` 및 `verticalAsymmetricType`에 대해 각각 단일 컨텍스트가 사용되어, 이러한 플래그들을 코딩/디코딩하는 경우 이용가능한 컨텍스트 블록 크기 정보와 무관하게, 이러한 2개의 플래그들의 통계적 거동을 캡처한다.
- [0274] 이러한 실시예의 이점은 특히 인터 슬라이스들에서, 코딩된 플래그들과 이웃 블록 정보 사이의 통계적 의존도들을 더 양호하게 모델링하는 것이며, 이는 이러한 플래그들의 코딩과 연관된 레이트 비용을 감소시킨다.
- [0275] 추가적인 실시예에 따르면, 선행 실시예들 중 하나에서 제안된 컨텍스트는 현재 블록과 연관된 쿼드-트리 플러스 바이너리 트리 깊이 값의 함수로서 인덱싱된다. 이는, CTU의 쿼드-트리 플러스 바이너리-트리 표현에서 각각의 깊이 레벨에 대해 별개의 세트의 컨텍스트들이 사용됨을 의미한다. `verAsymmetricSplitFlag` 및 `horAsymmetricSplitFlag` 선택 요소들에 대해 정의된 것과 동일한 원리들이 적용된다. 이러한 실시예의 이점은 코딩/디코딩되고 있는 선택 요소들의 통계적 거동을 모델링하기 위해 사용되는 더 정밀한 컨텍스트이다.
- [0276] 추가적인 실시예에 따르면, 선행 실시예들 중 하나에서 제안된 컨텍스트는 코딩 유닛의 형상의 함수, 즉, 폭 및 높이로서 인덱싱된다. 이는, 가능한 코딩 크기의 각각의 튜플(w, h)에 대해 컨텍스트들의 별개의 세트가 사용됨을 의미한다. `verAsymmetricSplitFlag` 및 `horAsymmetricSplitFlag` 선택 요소들에 대해 정의된 것과 동일한 원리들이 적용된다. 이러한 실시예의 이점은 코딩/디코딩되고 있는 선택 요소들의 통계적 거동을 모델링

하기 위한 추가적으로 개선된 콘텍스트이다.

[0277]

표 1 및 표 2의 신택스의 변형

[0278]

표 1 및 표 2에 정의된 신택스의 변형은 표 3에 도시된다. 이는,

[0279]

- 현재 블록에 대해, 부(parent) 블록의 비대칭 서브-블록을 형성하기 위해 이웃 블록과 공간적으로 병합될 수 있는지 여부를 식별하는 단계를 포함한다. 부 블록은 현재 블록의 부 블록(사례 1) 또는 현재 블록의 부의 부 블록(사례 2)일 수 있다. 이러한 단계는 인코더 측 및 디코더 측에서 동일한 방식으로 수행된다.

[0280]

사례 1의 예는 도 18의 상단에 예시되어 있다. 이러한 도면에서, 회색으로 채워진 상단 서브-블록은 현재 블록의 부의 비대칭 분할을 형성하기 위해 다음 서브-블록과 병합될 수 있다.

[0281]

사례 2의 예는 도 18의 바닥에 도시되어 있다. 현재 블록은 회색으로 채워진 블록이다. 이러한 도면에서, 도 18에서 CU'로 지칭되는 블록의 비대칭 분할을 형성하기 위해 현재 블록은 다음 서브-블록(도 18에 nextSubCU로 명명됨)과 병합될 수 있다.

[0282]

- 크기 $3 \cdot 2^n$ 을 갖는 비대칭 블록을 형성하기 위해, 현재 블록에 대해, 프로세싱 순서에서 다음 블록과 병합되는지 여부를 표시하는 merge_with_next_CU로 지칭되는 신택스 요소를 코딩함.

[0283]

초기 QTBT 코딩 신택스에 대한 최소 변화를 갖는 이러한 실시예를 지원하기 위해, QTBT 신택스는 바이너리 트리들의 코딩 및 바이너리 분리 모드에 대해 불변이다.

표 3

coding_binary_tree(x0, y0, width, height, cqtDepth) {	디스크립터
if(btSplitAllowed(x0,y0,width,height){	
bt_split_mode(x0,y0,width,height,cqtDepth)	
}	
if(btSplitFlag) {	
if(btSplitMode==HOR)	
{	
x1 = x0	
y1 = y0 + (height >> 1)	
sub_width = width;	
sub_height = (height >> 1)	
}	
else	
{	
x1 = x0 + (width >> 1)	
y1 = y0	
sub_width = (width >> 1)	
sub_height = height	
}	
coding_binary_tree(x0, y0, sub_width, sub_height, cqtDepth)	
if(x1 < pic_width_in_luma_samples && y1 < pic_height_in_luma_samples)	
coding_binary_tree(x1, y1, sub_width, sub_height, cqtDepth)	
}	
} else{	
merge_with_next_CU_flag	ae(v)
if(!merge_with_next_CU_flag && !merge_with_previous_CU){	
coding_unit(x0, y0, width, height)	
}	
if(merge_with_previous_CU){	
if(horizontal_merge){	
coding_unit(x0-(width >> 1), y0, width*3/2, height)	
}	
if(vertical_merge){	
coding_unit(x0, y0 - (height>>1), width, height*3/2)	
}	
}	
}	
}	

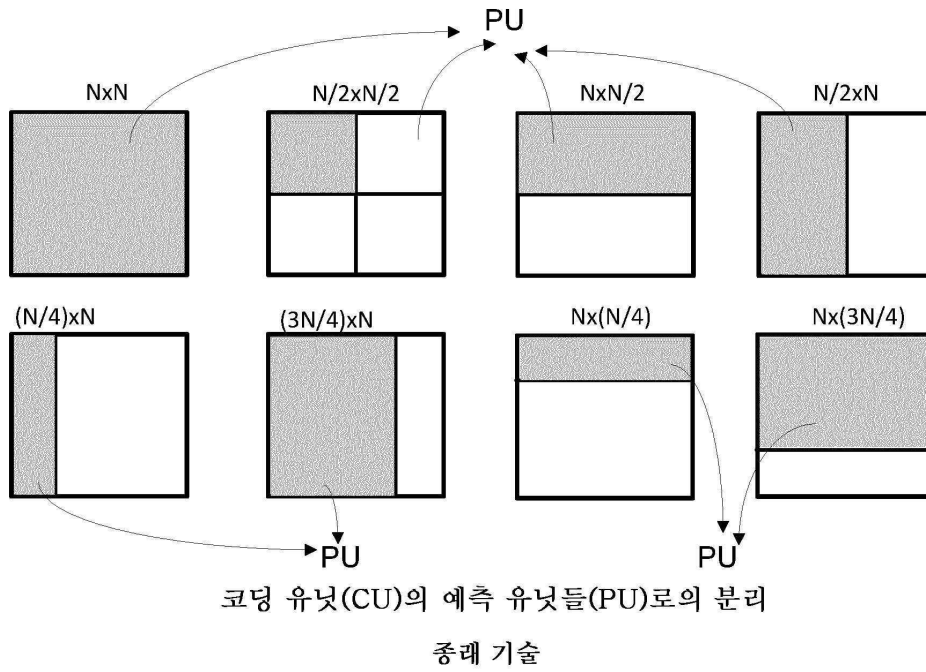
[0284]

- [0285] 블록이 분리되지 않는 경우, "merge_with_next_CU_flag"로 지칭되는 새로운 선택스 요소가 블록 코딩의 앞에 도입된다. 이러한 플래그는, 관련된 블록이 비트-스트림의 이러한 스테이지에서 아직 인코딩되지 않았지만 스케닝 순서에서 다음 블록과 병합됨을 표시한다. 이는 현재 바이너리 분리 배향에 따라 $\left(w * \frac{3}{2}, h\right)$ 또는 $\left(w, h * \frac{3}{2}\right)$ 와 동일한 크기를 갖는 비대칭 블록을 형성하는 것을 가능하게 한다.
- [0286] 이러한 실시예의 이점은, 비대칭 코딩 유닛들을 지원하는데 필요한 QTBT 선택스 수정의 양을 강력하게 감소시킨다는 점이다.
- [0287] 도 18에 대해 개시된 이러한 실시예는 또한 디코더 측에 적용된다. 이러한 경우, 비대칭 블록들을 복원하기 위해 선택스 요소 merge_with_next_CU_flag가 디코딩된다.
- [0288] 본원에서 설명되는 구현들은, 예를 들어, 방법 또는 프로세스, 장치, 소프트웨어 프로그램, 데이터 스트림 또는 신호에서 구현될 수 있다. 오직 단일 형태의 구현의 콘텍스트에서 논의되는(예를 들어, 오직 방법 또는 디바이스로서만 논의되는) 경우에도, 논의되는 특징들의 구현은 또한 다른 형태들(예를 들어, 프로그램)로 구현될 수 있다. 장치는 예를 들어, 적절한 하드웨어, 소프트웨어 및 펌웨어로 구현될 수 있다. 예를 들어, 방법들은, 예를 들어, 컴퓨터, 마이크로프로세서, 집적 회로 또는 프로그래밍가능 로직 디바이스를 포함하는 일반적인 프로세싱 디바이스들을 지칭하는, 예를 들어, 프로세서와 같은 장치로 구현될 수 있다. 프로세서들은 또한, 예를 들어, 컴퓨터들, 셀 폰들, 휴대용/개인 휴대 정보 단말("PDA들"), 및 최종 사용자들 사이에서 정보의 통신을 용이하게 하는 다른 디바이스들과 같은 통신 디바이스들을 포함한다.
- [0289] 본원에 설명된 다양한 프로세스들 및 특징들의 구현들은 특히, 예를 들어, 장비 또는 애플리케이션과 같은 다양한 상이한 장비 또는 애플리케이션들로 구현될 수 있다. 이러한 장비의 예들은 인코더, 디코더, 디코더로부터의 출력을 프로세싱하는 포스트-프로세서, 인코더에 입력을 제공하는 프리-프로세서, 비디오 코더, 비디오 디코더, 비디오 코덱, 웹 서버, 셋톱 박스, 랩톱, 개인용 컴퓨터, 셀 폰, PDA 및 다른 통신 디바이스들을 포함한다. 명확해야 할 바와 같이, 장치는 모바일일 수 있고, 심지어 모바일 차량에 설치될 수 있다.
- [0290] 추가적으로, 방법들은 프로세서에 의해 수행되는 명령어들에 의해 구현될 수 있고, 이러한 명령어들(및/또는 구현에 의해 생성된 데이터 값들)은 예를 들어, 집적 회로, 소프트웨어 캐리어 또는 다른 저장 디바이스, 예를 들어, 하드 디스크, 콤팩트 디스켓("CD"), 광 디스크(예를 들어, 종종 디지털 다기능 디스크 또는 디지털 비디오 디스크로 지칭되는 DVD), 랜덤 액세스 메모리("RAM") 또는 판독-전용 메모리("ROM")와 같은 프로세서 판독가능 매체 상에 저장될 수 있다. 명령어들은 프로세서 판독가능 매체 상에 유형으로 구현된 애플리케이션 프로그램을 형성할 수 있다. 명령어들은 예를 들어 하드웨어, 펌웨어, 소프트웨어 또는 이들의 조합일 수 있다. 명령어들은 예를 들어 운영 시스템, 별개의 애플리케이션 또는 이 둘의 조합에서 발견될 수 있다. 따라서, 프로세서는 프로세스를 수행하도록 구성된 디바이스, 및 프로세스를 수행하기 위한 명령어들을 갖는 프로세서-판독가능 매체(예를 들어, 저장 디바이스)를 포함하는 디바이스 둘 모두로서 특성화될 수 있다. 추가로, 프로세서-판독가능 매체는 명령어들에 추가하여 또는 명령어들 대신에, 구현에 의해 생성된 데이터 값들을 저장할 수 있다.
- [0291] 본 기술분야의 통상의 기술자에게 자명할 바와 같이, 구현들은, 예를 들어, 저장 또는 송신될 수 있는 정보를 반송하도록 포맷된 다양한 신호들을 생성할 수 있다. 정보는, 예를 들어, 방법을 수행하기 위한 명령어들, 또는 설명된 구현들 중 하나에 의해 생성되는 데이터를 포함할 수 있다. 예를 들어, 신호는 설명된 실시예의 선택스를 기록 또는 판독하기 위한 규칙들을 데이터로서 운반하거나, 설명된 실시예에 의해 기록된 실제 선택스 값들을 데이터로서 운반하도록 포맷될 수 있다. 이러한 신호는, 예를 들어, (예를 들어, 스펙트럼의 무선 주파수 부분을 사용하는) 전자기 파로서 또는 기저대역 신호로서 포맷될 수 있다. 포맷은, 예를 들어, 데이터 스트림을 인코딩하는 것 및 인코딩된 데이터 스트림과 캐리어를 변조하는 것을 포함할 수 있다. 신호가 반송하는 정보는, 예를 들어, 아날로그 또는 디지털 정보일 수 있다. 신호는, 공지된 바와 같이, 다양한 상이한 유선 또는 무선 링크들을 통해 송신될 수 있다. 신호는 프로세서-판독가능 매체 상에 저장될 수 있다.
- [0292] 다수의 구현들이 설명되었다. 그럼에도 불구하고, 다양한 수정들이 행해질 수 있음을 이해할 것이다. 예를 들어, 상이한 구현들의 요소들은 다른 구현들을 생성하기 위해 결합, 보충, 수정 또는 제거될 수 있다. 추가적으로, 통상의 기술자는, 다른 구조체들 및 프로세스들이 개시된 것들을 대체할 수 있고 결과적인 구현들이 적어도 실질적으로 동일한 방식(들)으로 적어도 실질적으로 동일한 기능(들)을 수행하여, 개시된 구현들과 적어도 실질

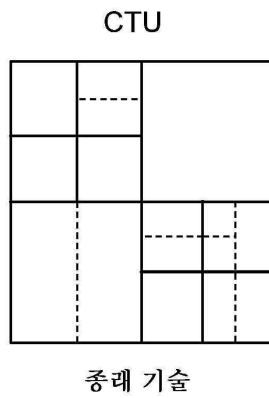
적으로 동일한 결과(들)를 달성할 것임을 이해할 것이다. 따라서, 이러한 및 다른 구현들이 본 출원에 의해 고려된다.

도면

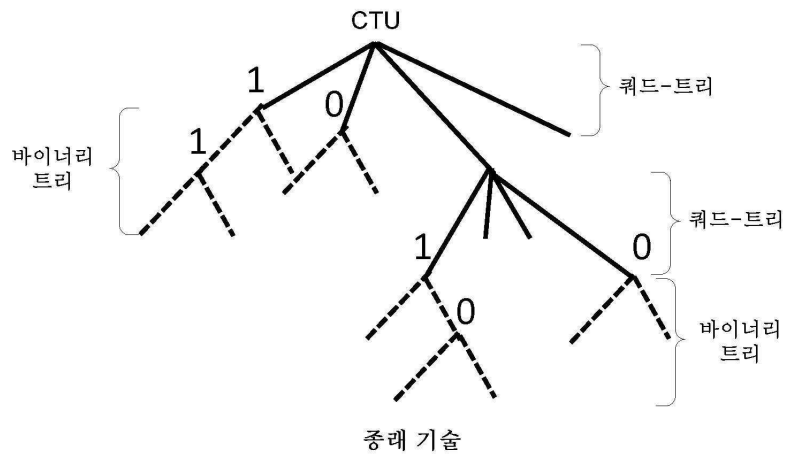
도면1



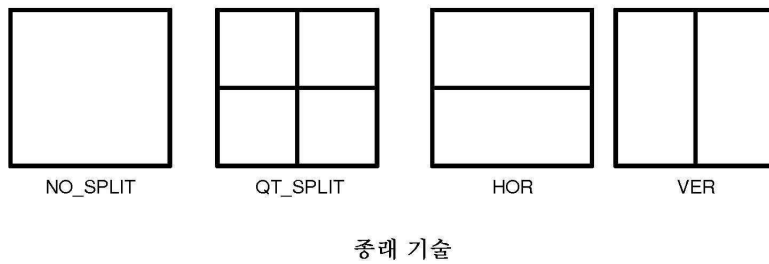
도면2



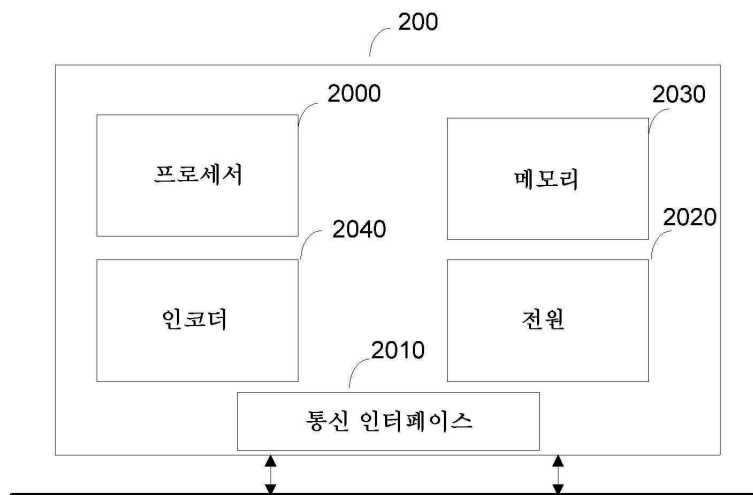
도면3



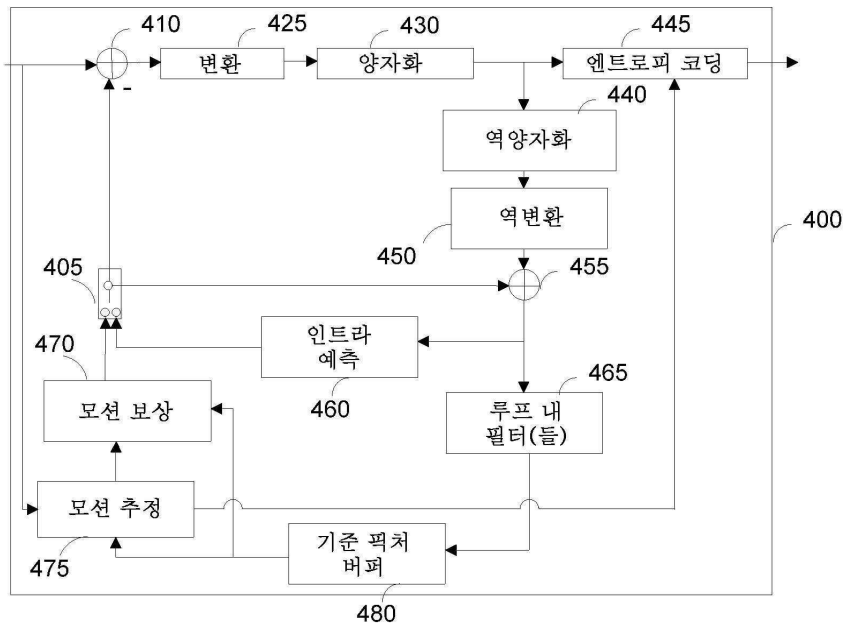
도면4



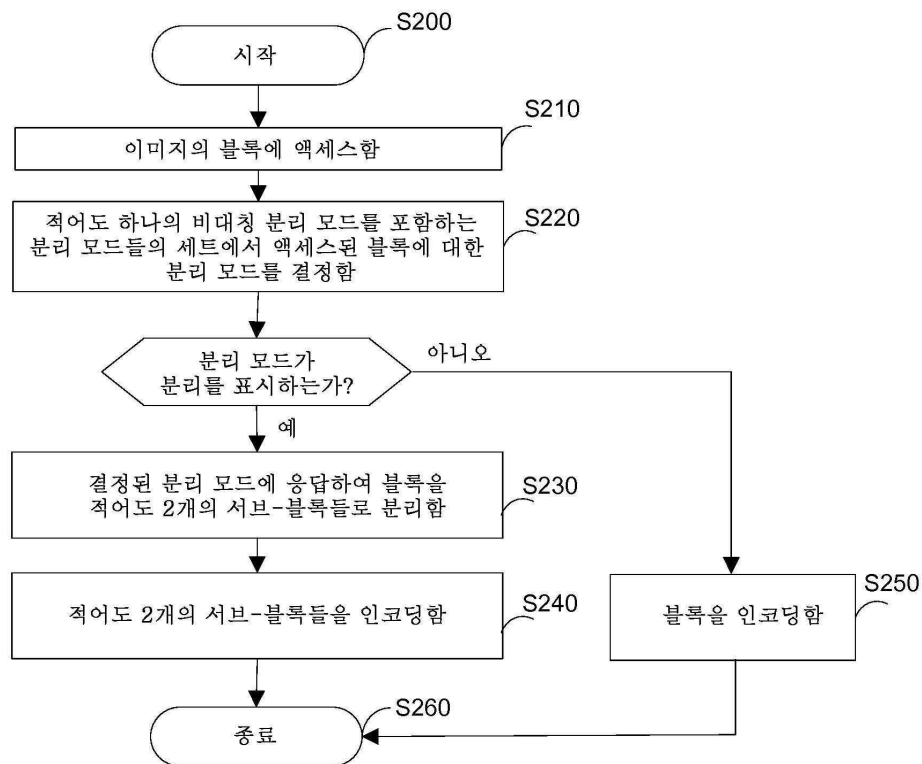
도면5



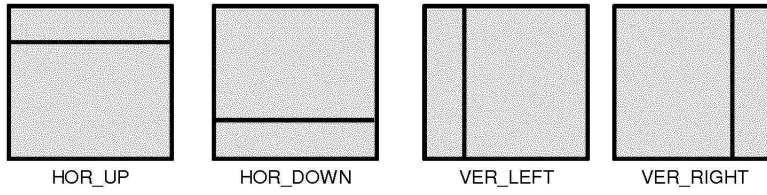
도면6



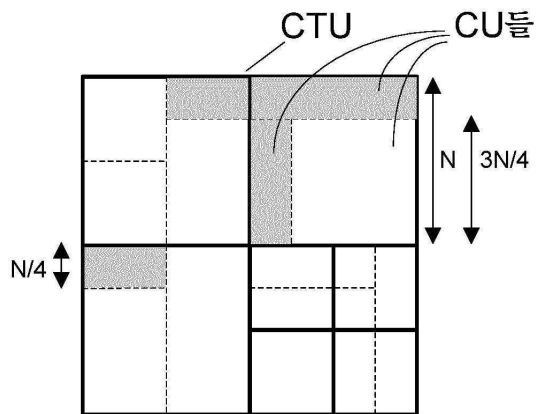
도면7



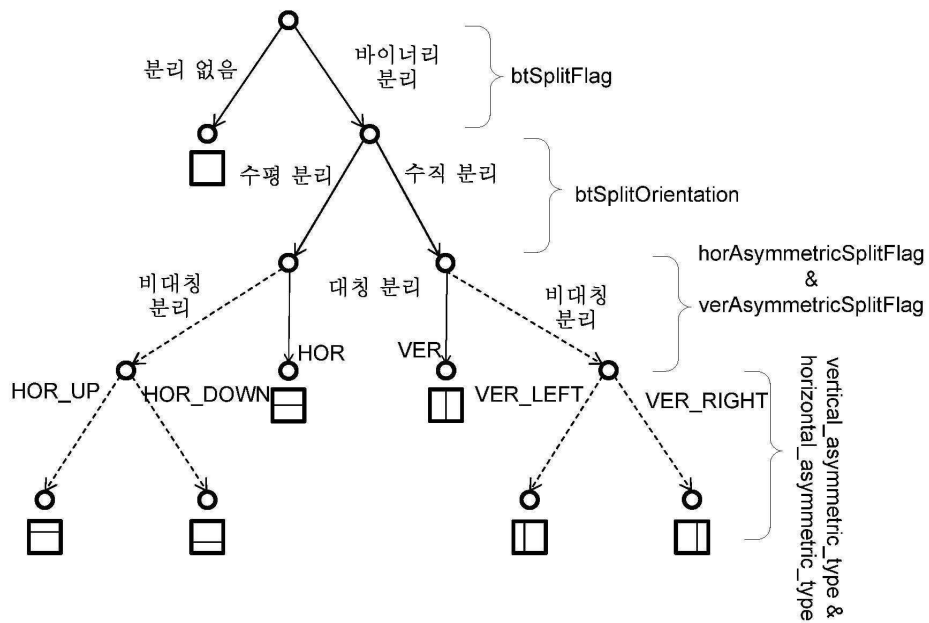
도면8



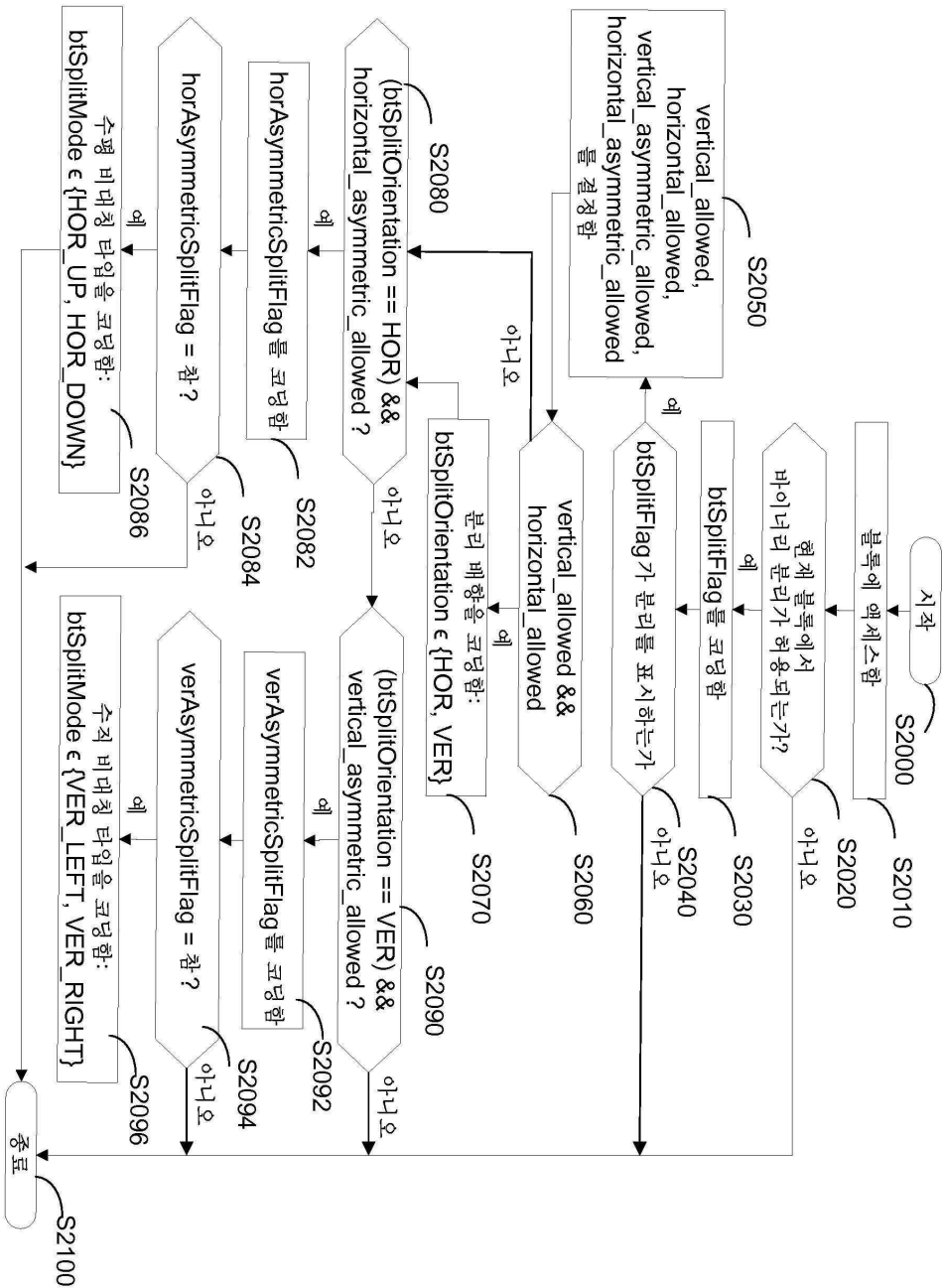
도면9



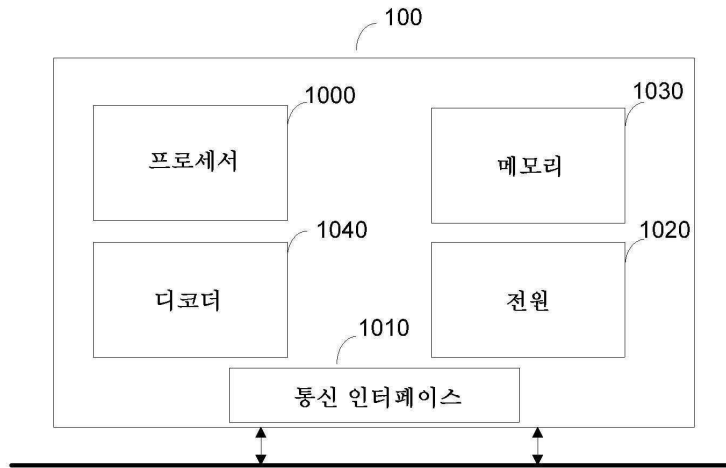
도면10



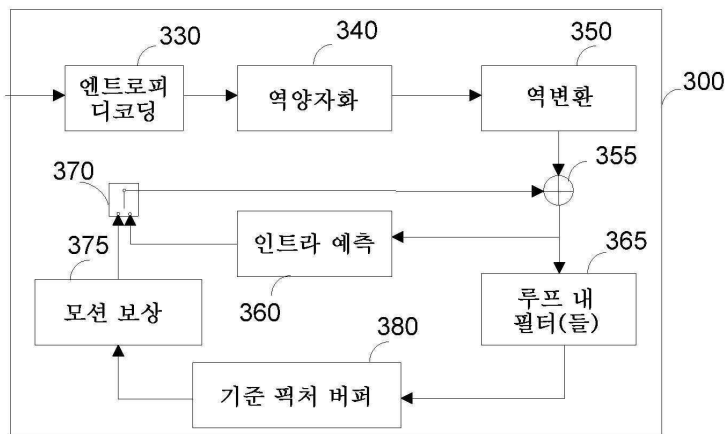
도면11



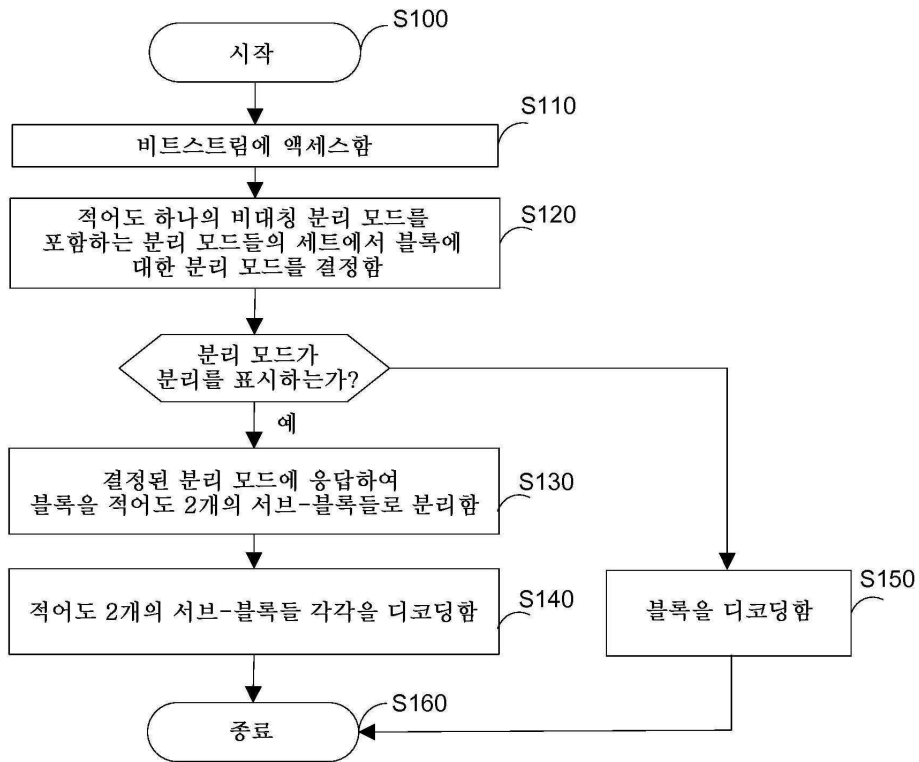
도면12



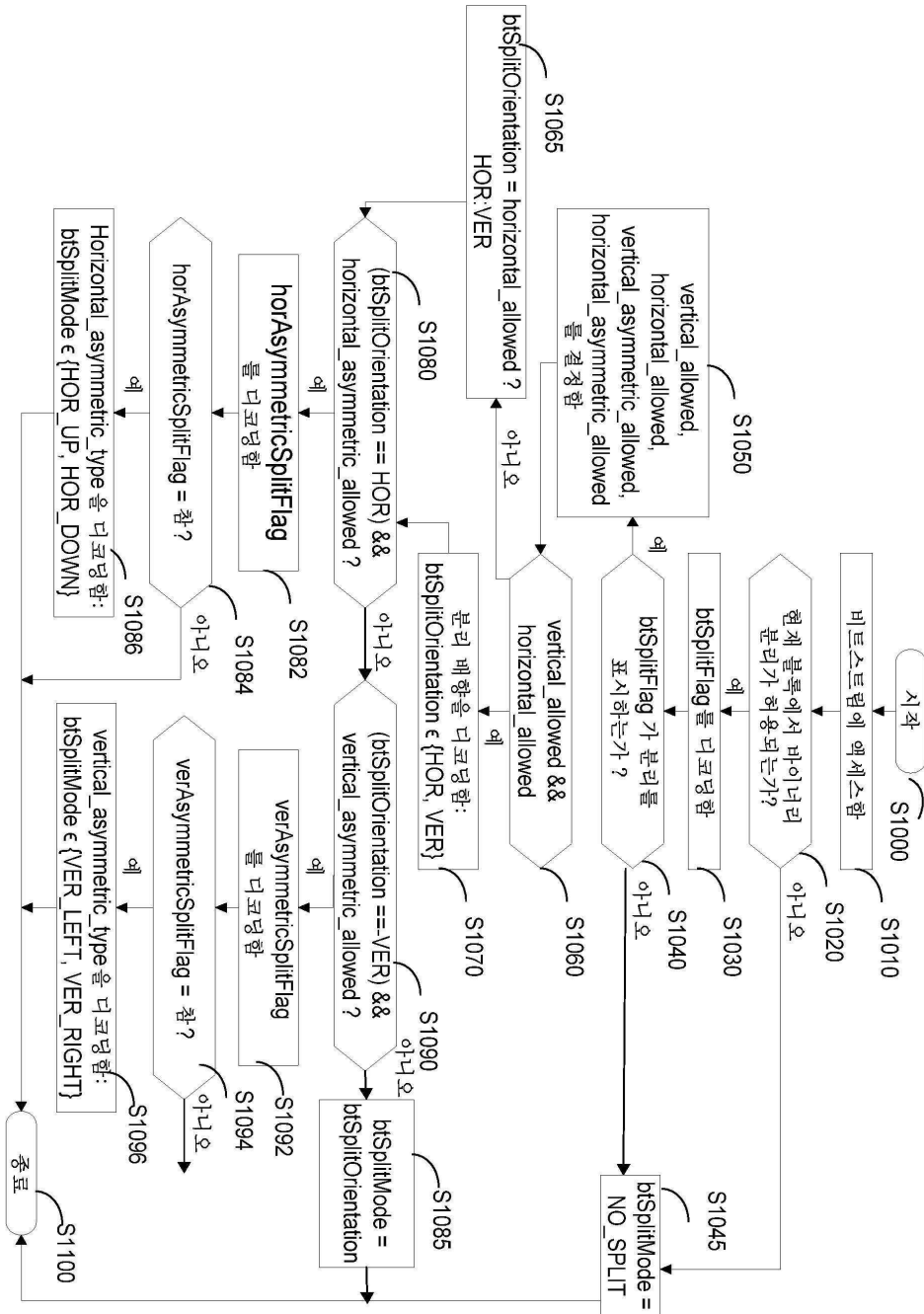
도면13



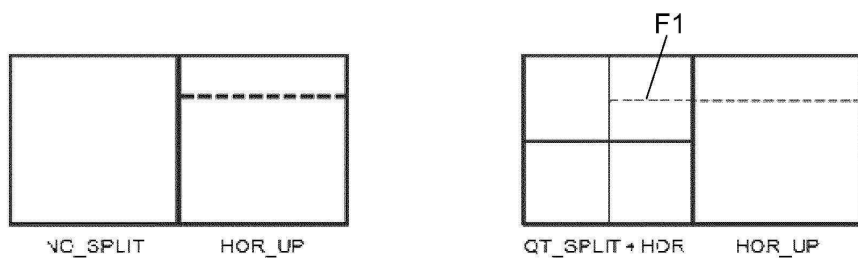
도면14



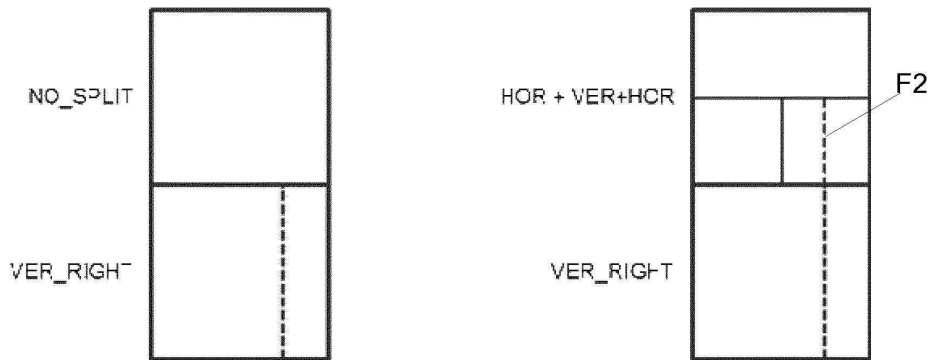
도면15



도면16



도면17



도면18

