(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2022-0157414
(43) 공개일자 2022년11월29일

(51) 국제특허분류(Int. Cl.)
H04N 19/70 (2014.01) *H04N 19/119* (2014.01)
H04N 19/167 (2014.01) *H04N 19/172* (2014.01)
H04N 19/174 (2014.01)
(52) CPC특허분류
H04N 19/70 (2015.01)
H04N 19/119 (2015.01)
(21) 출원번호 10-2022-7035379
(22) 출원일자(국제) 2021년03월17일
심사청구일자 2022년11월09일
(85) 번역문제출일자 2022년10월12일
(86) 국제출원번호 PCT/EP2021/056869
(87) 국제공개번호 WO 2021/185928
국제공개일자 2021년09월23일
(30) 우선권주장
2004099.4 2020년03월20일 영국(GB)

(71) 출원인
캐논 가부시끼가이샤
일본 도쿄도 오오따꾸 시모마루꼬 3쪼메 30방 2고
(72) 발명자
라로슈 기욤
프랑스 35250 생 토뱅 도비네 5 라 바스 로므래
우드라오고 나엘
프랑스 35330 발 다나스트 모르 드 브르타뉴 11 쿠투즈
온노 파트리스
프랑스 35700 렌느 5 뤼 메슬레
(74) 대리인
장수길, 이중희

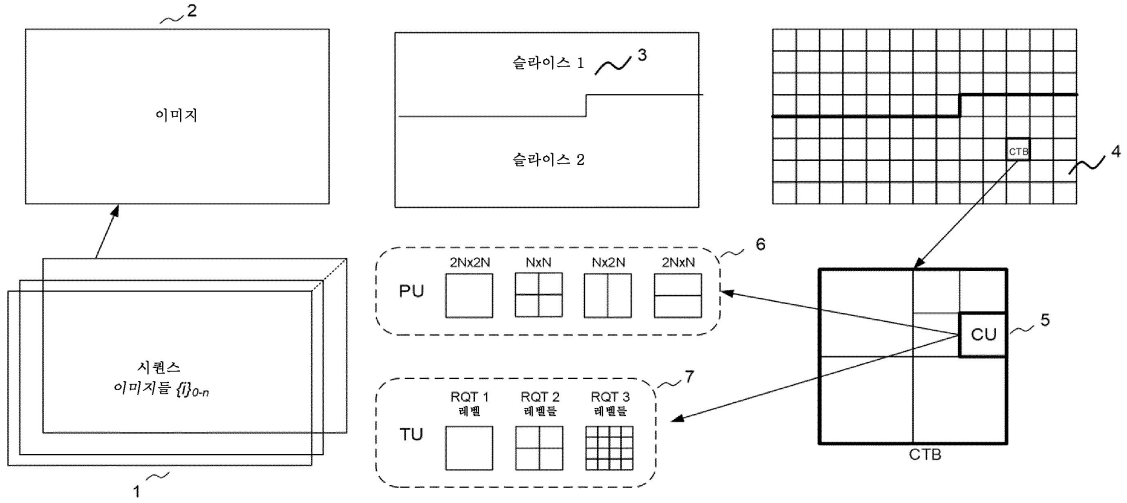
전체 청구항 수 : 총 21 항

(54) 발명의 명칭 **비디오 코딩 및 디코딩을 위한 고레벨 신택스**

(57) 요 약

비트스트림으로부터 비디오 데이터를 디코딩하는 방법이 제공되며, 비트스트림은 하나 이상의 슬라이스에 대응하는 비디오 데이터를 포함한다. 각각의 슬라이스는 하나 이상의 타일을 포함할 수 있다. 비트스트림은 하나 이상의 슬라이스를 디코딩할 때 이용될 신택스 요소들을 포함하는 픽처 헤더와, 이용될 신택스 요소들을 포함하는 슬라이스 헤더를 포함한다. 슬라이스를 디코딩하는 것은 신택스 요소들을 파싱하는 것을 포함한다. 슬라이스가 복수의 타일을 포함하는 경우, 픽처 헤더가 슬라이스 헤더에서 시그널링된다는 것을 나타내는 신택스 요소가 파싱된다면 슬라이스의 어드레스를 나타내는 신택스 요소의 파싱이 생략된다. 비트스트림은 상기 신택스 요소들을 이용하여 디코딩된다.

대 표 도



(52) CPC특허분류
H04N 19/167 (2015.01)
H04N 19/172 (2015.01)
H04N 19/174 (2015.01)

명 세 서

청구범위

청구항 1

비트스트림으로부터 비디오 데이터를 디코딩하는 방법으로서,

상기 비트스트림은 하나 이상의 슬라이스에 대응하는 비디오 데이터를 포함하고, 각각의 슬라이스는 하나 이상의 타일을 포함할 수 있고,

상기 비트스트림은, 하나 이상의 슬라이스를 디코딩할 때 이용될 신택스 요소들을 포함하는 픽처 헤더와, 슬라이스를 디코딩할 때 이용될 신택스 요소들을 포함하는 슬라이스 헤더를 포함하며, 상기 방법은,

상기 신택스 요소들을 파싱하는 단계;

픽처가 복수의 타일을 포함하는 경우, 픽처 헤더가 상기 슬라이스 헤더에서 시그널링된다는 것을 나타내는 신택스 요소가 파싱된다면 슬라이스의 어드레스를 나타내는 신택스 요소의 파싱을 생략하는 단계; 및

상기 신택스 요소들을 이용하여 상기 비트스트림을 디코딩하는 단계

를 포함하는, 방법.

청구항 2

제1항에 있어서,

상기 생략하는 단계는 래스터 스캔 슬라이스 모드가 상기 슬라이스를 디코딩하는데 이용될 때 수행될 것인, 방법.

청구항 3

제1항 또는 제2항에 있어서,

상기 생략하는 단계는 상기 슬라이스에서의 타일들의 수를 나타내는 신택스 요소의 파싱을 생략하는 단계를 더 포함하는, 방법.

청구항 4

비트스트림으로부터 비디오 데이터를 디코딩하는 방법으로서,

상기 비트스트림은 하나 이상의 슬라이스에 대응하는 비디오 데이터를 포함하고, 각각의 슬라이스는 하나 이상의 타일을 포함할 수 있고,

상기 비트스트림은, 하나 이상의 슬라이스를 디코딩할 때 이용될 신택스 요소들을 포함하는 픽처 헤더와, 슬라이스를 디코딩할 때 이용될 신택스 요소들을 포함하는 슬라이스 헤더를 포함하며, 상기 디코딩하는 것은,

하나 이상의 신택스 요소를 파싱하는 것;

픽처가 복수의 타일을 포함하는 경우, 상기 픽처 헤더가 상기 슬라이스 헤더에서 시그널링된다는 것을 나타내는 신택스 요소가 파싱된다면 상기 슬라이스에서의 타일들의 수를 나타내는 신택스 요소의 파싱을 생략하는 것; 및

상기 신택스 요소들을 이용하여 상기 비트스트림을 디코딩하는 것

을 포함하는, 방법.

청구항 5

제4항에 있어서,

상기 생략하는 것은 래스터 스캔 슬라이스 모드가 상기 슬라이스를 디코딩하는데 이용될 때 수행될 것인, 방법.

청구항 6

제4항 또는 제5항에 있어서,

상기 픽처에서의 타일들의 수를 나타내는 신택스 요소들을 파싱하는 것, 및 파싱된 신택스 요소들에 의해 표시된 상기 픽처에서의 타일들의 수에 기반하여 상기 슬라이스에서의 타일들의 수를 결정하는 것을 더 포함하는, 방법.

청구항 7

제4항 내지 제6항 중 어느 한 항에 있어서,

상기 생략하는 것은 슬라이스의 어드레스를 나타내는 신택스 요소의 파싱을 생략하는 것을 더 포함하는, 방법.

청구항 8

비디오 데이터를 비트스트림으로 인코딩하는 방법으로서,

상기 비트스트림은 하나 이상의 슬라이스에 대응하는 비디오 데이터를 포함하고, 각각의 슬라이스는 하나 이상의 타일을 포함할 수 있고,

상기 비트스트림은, 하나 이상의 슬라이스를 디코딩할 때 이용될 신택스 요소들을 포함하는 픽처 헤더와, 슬라이스를 인코딩할 때 이용될 신택스 요소들을 포함하는 슬라이스 헤더를 포함하며, 상기 인코딩하는 것은,

상기 비디오 데이터를 인코딩하기 위한 하나 이상의 신택스 요소를 결정하는 것;

픽처가 복수의 타일을 포함하는 경우, 픽처 헤더가 상기 슬라이스 헤더에서 시그널링된다는 것을 신택스 요소가 나타낸다면 슬라이스의 어드레스를 나타내는 신택스 요소의 인코딩을 생략하는 것; 및

상기 신택스 요소들을 이용하여 상기 비디오 데이터를 인코딩하는 것

을 포함하는, 방법.

청구항 9

제14항에 있어서,

상기 생략하는 것은 래스터 스캔 슬라이스 모드가 상기 슬라이스를 인코딩하는데 이용될 때 수행될 것인, 방법.

청구항 10

제14항 또는 제15항에 있어서,

상기 생략하는 것은 상기 슬라이스에서의 타일들의 수를 나타내는 신택스 요소의 인코딩을 생략하는 것을 더 포함하는, 방법.

청구항 11

비디오 데이터를 비트스트림으로 인코딩하는 방법으로서,

상기 비트스트림은 하나 이상의 슬라이스에 대응하는 비디오 데이터를 포함하고, 각각의 슬라이스는 하나 이상의 타일을 포함할 수 있고,

상기 비트스트림은, 하나 이상의 슬라이스를 디코딩할 때 이용될 신택스 요소들을 포함하는 픽처 헤더와, 슬라이스를 디코딩할 때 이용될 신택스 요소들을 포함하는 슬라이스 헤더를 포함하며, 상기 인코딩하는 것은,

상기 비디오 데이터를 인코딩하기 위한 하나 이상의 신택스 요소를 결정하는 것;

픽처가 복수의 타일을 포함하는 경우, 상기 픽처 헤더가 상기 슬라이스 헤더에서 시그널링된다는 것을 나타내는 신택스 요소가 인코딩을 위해 결정된다면 상기 슬라이스에서의 타일들의 수를 나타내는 신택스 요소의 인코딩을 생략하는 것; 및

상기 신택스 요소들을 이용하여 상기 비디오 데이터를 인코딩하는 것

을 포함하는, 방법.

청구항 12

제17항에 있어서,

상기 생략하는 것은 래스터 스캔 슬라이스 모드가 상기 슬라이스를 인코딩하는데 이용될 때 수행될 것인, 방법.

청구항 13

제17항 또는 제18항에 있어서,

상기 픽처에서의 타일들의 수를 나타내는 신택스 요소들을 인코딩하는 것을 더 포함하고, 상기 슬라이스에서의 타일들의 수는 파싱된 신택스 요소들에 의해 표시된 상기 픽처에서의 타일들의 수에 기반하는, 방법.

청구항 14

제17항 내지 제19항 중 어느 한 항에 있어서,

상기 생략하는 것은 슬라이스의 어드레스를 나타내는 신택스 요소의 인코딩을 생략하는 것을 더 포함하는, 방법.

청구항 15

비트스트림으로부터 비디오 데이터를 디코딩하는 방법으로서,

상기 비트스트림은 하나 이상의 슬라이스에 대응하는 비디오 데이터를 포함하고, 각각의 슬라이스는 하나 이상의 타일을 포함할 수 있고,

상기 비트스트림은, 하나 이상의 슬라이스를 디코딩할 때 이용될 신택스 요소들을 포함하는 픽처 헤더와, 슬라이스를 디코딩할 때 이용될 신택스 요소들을 포함하는 슬라이스 헤더를 포함하며,

상기 비트스트림은, 상기 비트스트림이 픽처가 복수의 타일을 포함한다는 것을 나타내는 값을 갖는 신택스 요소를 포함하고, 상기 비트스트림이 픽처 헤더가 상기 슬라이스 헤더에서 시그널링된다는 것을 나타내는 신택스 요소를 포함하는 경우, 상기 비트스트림이 또한 슬라이스의 어드레스를 나타내는 신택스 요소가 파싱되지 않을 것임을 나타내는 신택스 요소를 포함하도록 제약되고, 상기 방법은 상기 신택스 요소들을 이용하여 상기 비트스트림을 디코딩하는 단계를 포함하는, 방법.

청구항 16

비트스트림으로부터 비디오 데이터를 디코딩하는 방법으로서,

상기 비트스트림은 하나 이상의 슬라이스에 대응하는 비디오 데이터를 포함하고, 각각의 슬라이스는 하나 이상의 타일을 포함할 수 있고,

상기 비트스트림은, 하나 이상의 슬라이스를 디코딩할 때 이용될 신택스 요소들을 포함하는 픽처 헤더와, 슬라이스를 디코딩할 때 이용될 신택스 요소들을 포함하는 슬라이스 헤더를 포함하며,

상기 비트스트림은, 상기 비트스트림이 픽처가 복수의 타일을 포함한다는 것을 나타내는 값을 갖는 신택스 요소를 포함하고, 상기 비트스트림이 상기 픽처 헤더가 상기 슬라이스 헤더에서 시그널링된다는 것을 나타내는 신택스 요소를 포함하는 경우, 상기 비트스트림이 또한 상기 슬라이스에서의 타일들의 수를 나타내는 신택스 요소가 파싱되지 않을 것임을 나타내는 신택스 요소를 포함하도록 제약되고, 상기 방법은 상기 신택스 요소들을 이용하여 상기 비트스트림을 디코딩하는 단계를 포함하는, 방법.

청구항 17

비디오 데이터를 비트스트림으로 인코딩하는 방법으로서,

상기 비트스트림은 하나 이상의 슬라이스에 대응하는 비디오 데이터를 포함하고, 각각의 슬라이스는 하나 이상의 타일을 포함할 수 있고,

상기 비트스트림은, 하나 이상의 슬라이스를 디코딩할 때 이용될 신택스 요소들을 포함하는 픽처 헤더와, 슬라

이스를 인코딩할 때 이용될 신택스 요소들을 포함하는 슬라이스 헤더를 포함하며,

상기 비트스트림은, 상기 비트스트림이 픽처가 복수의 타일을 포함한다는 것을 나타내는 값을 갖는 신택스 요소를 포함하고, 상기 비트스트림이 픽처 헤더가 상기 슬라이스 헤더에서 시그널링된다는 것을 나타내는 신택스 요소를 포함하는 경우, 상기 비트스트림이 또한 슬라이스의 어드레스를 나타내는 신택스 요소가 파싱되지 않을 것임을 나타내는 신택스 요소를 포함하도록 제약되고, 상기 방법은 상기 신택스 요소들을 이용하여 상기 비디오 데이터를 인코딩하는 단계를 포함하는, 방법.

청구항 18

비디오 데이터를 비트스트림으로 인코딩하는 방법으로서,

상기 비트스트림은 하나 이상의 슬라이스에 대응하는 비디오 데이터를 포함하고, 각각의 슬라이스는 하나 이상의 타일을 포함할 수 있고,

상기 비트스트림은, 하나 이상의 슬라이스를 디코딩할 때 이용될 신택스 요소들을 포함하는 픽처 헤더와, 슬라이스를 디코딩할 때 이용될 신택스 요소들을 포함하는 슬라이스 헤더를 포함하며,

상기 비트스트림은, 상기 비트스트림이 픽처가 복수의 타일을 포함한다는 것을 나타내는 값을 갖는 신택스 요소를 포함하고, 상기 비트스트림이 상기 픽처 헤더가 상기 슬라이스 헤더에서 시그널링된다는 것을 나타내는, 인코딩을 위해 결정되는 신택스 요소를 포함하는 경우, 상기 비트스트림이 또한 상기 슬라이스에서의 타일들의 수를 나타내는 신택스 요소가 파싱되지 않을 것임을 나타내는 신택스 요소를 포함하도록 제약되고, 상기 방법은 상기 신택스 요소들을 이용하여 상기 비디오 데이터를 인코딩하는 단계를 포함하는, 방법.

청구항 19

비트스트림으로부터 비디오 데이터를 디코딩하기 위한 디코더로서,

제1항 내지 제7항, 제15항 및 제16항 중 어느 한 항의 방법을 수행하도록 구성되는, 디코더.

청구항 20

비디오 데이터를 비트스트림으로 인코딩하기 위한 인코더로서,

제8항 내지 제14항, 제17항 및 제18항 중 어느 한 항의 방법을 수행하도록 구성되는, 인코더.

청구항 21

컴퓨터 프로그램으로서,

실행 시에 제1항 내지 제18항 중 어느 한 항의 방법이 수행되게 하는, 컴퓨터 프로그램.

발명의 설명

기 술 분 야

[0001] 본 발명은 비디오 코딩 및 디코딩에 관한 것으로, 특히 비트스트림에서 이용되는 고레벨 신택스(high level syntax)에 관한 것이다.

배 경 기 술

[0002] 최근, MPEG 및 ITU-T 연구 그룹 16의 VCEG에 의해 형성된 협업 팀인 JVET(Joint Video Experts Team)는, 다목적 비디오 코딩(Versatile Video Coding)(VVC)이라고 하는 새로운 비디오 코딩 표준에 관한 작업을 개시하였다. VVC의 목표는, 기존의 HEVC 표준에 비해 압축 성능의 상당한 개선들(즉, 통상적으로 이전보다 2배)을 제공하고 2020년에 완료되는 것이다. 주요 타겟 애플리케이션들 및 서비스들은, 360도 및 높은 동적 범위(high-dynamic-range)(HDR) 비디오들을 포함하지만, 이것으로 제한되는 것은 아니다. 전체적으로, JVET는 독립된 테스트 실험실들에 의해 수행된 공식적 주관적인 테스트들을 이용하여 32개 조직으로부터의 응답들을 평가하였다. 일부 제안들은 HEVC를 이용하는 것과 비교할 때 통상적으로 40% 이상의 압축 효율 이득들을 보여주었다. 초고해상도(UHD) 비디오 테스트 자료에 대해 특정한 유효성이 보여졌다. 따라서, 최종 표준에 대해 타겟팅된 50%를 훨씬 넘는 압축 효율 이득들을 예상할 수 있다.

[0003] JVET 탐색 모델(JEM)은 모든 HEVC 툴들을 이용하고 다수의 새로운 툴들을 도입하였다. 이러한 변경들은 비트스트림의 구조, 특히 비트스트림의 전체 비트레이트에 영향을 미칠 수 있는 고레벨 신택스에 대한 변경을 필요로 하였다.

발명의 내용

[0004] 본 발명은 코딩 성능의 어떠한 저하도 없이 복잡도의 감소로 이어지는 고레벨 신택스 구조의 개선에 관한 것이다.

[0005] 본 발명에 따른 제1 양태에서, 비트스트림으로부터 비디오 데이터를 디코딩하는 방법이 제공되며, 비트스트림은 하나 이상의 슬라이스에 대응하는 비디오 데이터를 포함하고, 각각의 슬라이스는 하나 이상의 타일을 포함할 수 있고, 비트스트림은 하나 이상의 슬라이스를 디코딩할 때 이용될 신택스 요소들을 포함하는 픽처 헤더와, 슬라이스를 디코딩할 때 이용될 신택스 요소들을 포함하는 슬라이스 헤더를 포함하며, 이 방법은, 신택스 요소들을 파싱하는 단계; 슬라이스(또는 픽처)가 복수의 타일을 포함하는 경우, 픽처 헤더가 슬라이스 헤더에서 시그널링된다는 것을 나타내는 신택스 요소가 파싱된다면 슬라이스의 어드레스를 나타내는 신택스 요소의 파싱을 생략하는 단계; 및 상기 신택스 요소들을 이용하여 상기 비트스트림을 디코딩하는 단계를 포함한다. 본 발명에 따른 다른 양태에서, 비트스트림으로부터 비디오 데이터를 디코딩하는 방법이 제공되며, 비트스트림은 하나 이상의 슬라이스에 대응하는 비디오 데이터를 포함하고, 각각의 슬라이스는 하나 이상의 타일을 포함할 수 있고, 비트스트림은 하나 이상의 슬라이스를 디코딩할 때 이용될 신택스 요소들을 포함하는 픽처 헤더와, 슬라이스를 디코딩할 때 이용될 신택스 요소들을 포함하는 슬라이스 헤더를 포함하며, 이 방법은, 신택스 요소들을 파싱하는 단계; 슬라이스 또는 픽처가 복수의 타일을 포함하는 경우, 픽처 헤더가 슬라이스 헤더에서 시그널링된다는 것을 나타내는 신택스 요소가 파싱된다면 슬라이스의 어드레스를 나타내는 신택스 요소의 파싱을 생략하는 단계; 및 상기 신택스 요소들을 이용하여 상기 비트스트림을 디코딩하는 단계를 포함한다. 본 발명에 따른 다른 추가적인 양태에서, 비트스트림으로부터 비디오 데이터를 디코딩하는 방법이 제공되며, 비트스트림은 하나 이상의 슬라이스에 대응하는 비디오 데이터를 포함하고, 각각의 슬라이스는 하나 이상의 타일을 포함할 수 있고, 비트스트림은 하나 이상의 슬라이스를 디코딩할 때 이용될 신택스 요소들을 포함하는 픽처 헤더와, 슬라이스를 디코딩할 때 이용될 신택스 요소들을 포함하는 슬라이스 헤더를 포함하며, 비트스트림은, 비트스트림이 슬라이스 또는 픽처가 복수의 타일을 포함한다는 것을 나타내는 값을 갖는 신택스 요소를 포함하고, 비트스트림이 픽처 헤더가 슬라이스 헤더에서 시그널링된다는 것을 나타내는 신택스 요소를 포함하는 경우, 비트스트림이 또한 슬라이스의 어드레스를 나타내는 신택스 요소가 파싱되지 않을 것임을 나타내는 신택스 요소를 포함하도록 제약되고, 이 방법은 상기 신택스 요소들을 이용하여 상기 비트스트림을 디코딩하는 단계를 포함한다.

[0006] 따라서, 특히 낮은 지연 및 낮은 비트레이트 응용들에 대해, 픽처 헤더가 비트레이트를 감소시키는 슬라이스 헤더에 있을 때 슬라이스 어드레스는 파싱되지 않는다. 또한, 픽처가 슬라이스 헤더에서 시그널링될 때 파싱 복잡도가 감소될 수 있다.

[0007] 실시예에서, 생략하는 단계는 래스터 스캔 슬라이스 모드(raster-scan slice mode)가 슬라이스를 디코딩하는데 이용될 때(에만) 수행될 것이다. 이것은 파싱 복잡도를 감소시키지만, 여전히 일부 비트레이트 감소를 허용한다.

[0008] 생략하는 단계는 슬라이스에서의 타일들의 수를 나타내는 신택스 요소의 파싱을 생략하는 단계를 더 포함할 수 있다. 따라서, 비트레이트의 추가적인 감소가 달성될 수 있다.

[0009] 제2 양태에서, 비트스트림으로부터 비디오 데이터를 디코딩하는 방법이 제공되며, 비트스트림은 하나 이상의 슬라이스에 대응하는 비디오 데이터를 포함하고, 각각의 슬라이스는 하나 이상의 타일을 포함할 수 있고, 비트스트림은 하나 이상의 슬라이스를 디코딩할 때 이용될 신택스 요소들을 포함하는 픽처 헤더와, 슬라이스를 디코딩할 때 이용될 신택스 요소들을 포함하는 슬라이스 헤더를 포함하며, 디코딩하는 것은, 하나 이상의 신택스 요소를 파싱하는 것; 슬라이스(또는 픽처)가 복수의 타일을 포함하는 경우, 픽처 헤더가 슬라이스 헤더에서 시그널링된다는 것을 나타내는 신택스 요소가 파싱된다면 슬라이스에서의 타일들의 수를 나타내는 신택스 요소의 파싱을 생략하는 것; 및 상기 신택스 요소들을 이용하여 상기 비트스트림을 디코딩하는 것을 포함한다. 추가적인 양태에서, 비트스트림으로부터 비디오 데이터를 디코딩하는 방법이 제공되며, 비트스트림은 하나 이상의 슬라이스에 대응하는 비디오 데이터를 포함하고, 각각의 슬라이스는 하나 이상의 타일을 포함할 수 있고, 비트스트림은 하나 이상의 슬라이스를 디코딩할 때 이용될 신택스 요소들을 포함하는 픽처 헤더와, 슬라이스를 디코딩할 때 이용될 신택스 요소들을 포함하는 슬라이스 헤더를 포함하며, 디코딩하는 것은, 하나 이상의 신택스 요소를

파싱하는 것; 슬라이스 또는 픽처가 복수의 타일을 포함하는 경우, 픽처 헤더가 슬라이스 헤더에서 시그널링된다는 것을 나타내는 신택스 요소가 파싱된다면 슬라이스에서의 타일들의 수를 나타내는 신택스 요소의 파싱을 생략하는 것; 및 상기 신택스 요소들을 이용하여 상기 비트스트림을 디코딩하는 것을 포함한다. 본 발명의 다른 추가적인 양태에서, 비트스트림으로부터 비디오 데이터를 디코딩하는 방법이 제공되며, 비트스트림은 하나 이상의 슬라이스에 대응하는 비디오 데이터를 포함하고, 각각의 슬라이스는 하나 이상의 타일을 포함할 수 있고, 비트스트림은 하나 이상의 슬라이스를 디코딩할 때 이용될 신택스 요소들을 포함하는 픽처 헤더와, 슬라이스를 디코딩할 때 이용될 신택스 요소들을 포함하는 슬라이스 헤더를 포함하며, 비트스트림은, 비트스트림이 슬라이스 또는 픽처가 복수의 타일을 포함한다는 것을 나타내는 값을 갖는 신택스 요소를 포함하고, 비트스트림이 픽처 헤더가 슬라이스 헤더에서 시그널링된다는 것을 나타내는 신택스 요소를 포함하는 경우, 비트스트림이 또한 슬라이스에서의 타일들의 수를 나타내는 신택스 요소가 파싱되지 않을 것임을 나타내는 신택스 요소를 포함하도록 제약되고, 이 방법은 상기 신택스 요소들을 이용하여 상기 비트스트림을 디코딩하는 단계를 포함한다.

[0010] 따라서, 비트레이트가 감소될 수 있으며, 이는 타일들의 수가 전송될 필요가 없는 낮은 지연 및 낮은 비트레이트 응용들에 특히 유리하다.

[0011] 생략하는 것은 래스터 스캔 슬라이스 모드가 슬라이스를 디코딩하는데 이용될 때(에만) 수행될 수 있다. 이것은 파싱 복잡도를 감소시키지만, 여전히 일부 비트레이트 감소를 허용한다.

[0012] 이 방법은 픽처에서의 타일들의 수를 나타내는 신택스 요소들을 파싱하는 것 및 파싱된 신택스 요소들에 의해 표시된 픽처에서의 타일들의 수에 기반하여 슬라이스에서의 타일들의 수를 결정하는 것을 더 포함할 수 있다. 이것은 추가의 시그널링을 요구하지 않고 픽처 헤더가 슬라이스 헤더에서 시그널링되는 경우에 슬라이스에서의 타일들의 수가 용이하게 예측되는 것을 허용하기 때문에 유리하다.

[0013] 생략하는 것은 슬라이스의 어드레스를 나타내는 신택스 요소의 파싱을 생략하는 것을 더 포함할 수 있다. 따라서, 비트레이트가 추가로 감소될 수 있다.

[0014] 본 발명의 제3 양태에서, 비트스트림으로부터 비디오 데이터를 디코딩하는 방법이 제공되며, 비트스트림은 하나 이상의 슬라이스에 대응하는 비디오 데이터를 포함하고, 각각의 슬라이스는 하나 이상의 타일을 포함할 수 있고, 비트스트림은 하나 이상의 슬라이스를 디코딩할 때 이용될 신택스 요소들을 포함하는 픽처 헤더와, 슬라이스를 디코딩할 때 이용될 신택스 요소들을 포함하는 슬라이스 헤더를 포함하며, 디코딩하는 것은, 하나 이상의 신택스 요소를 파싱하는 것; 슬라이스(또는 픽처)가 복수의 타일을 포함하는 경우, 슬라이스에서의 타일들의 수가 픽처에서의 타일들의 수와 동일하다면 슬라이스 어드레스를 나타내는 신택스 요소의 파싱을 생략하는 것; 및 상기 신택스 요소들을 이용하여 상기 비트스트림을 디코딩하는 것을 포함한다. 이것은 슬라이스에서의 타일들의 수가 픽처에서의 타일들의 수와 동일한 경우, 현재 픽처가 하나의 슬라이스만을 포함하는 것이 확실하다는 통찰을 이용한다. 따라서, 슬라이스 어드레스를 생략함으로써, 비트레이트가 개선될 수 있고, 파싱 및/또는 인코딩에서의 복잡도가 감소될 수 있다.

[0015] 생략하는 것은 래스터 스캔 슬라이스 모드가 슬라이스를 디코딩하는데 이용될 때(에만) 수행될 수 있다. 따라서, 복잡도가 감소되면서도 여전히 일부 비트레이트 감소를 제공할 수 있다.

[0016] 디코딩하는 것은, 슬라이스에서, 슬라이스에서의 타일들의 수를 나타내는 신택스 요소를 파싱하는 것; 및 픽처 파라미터 세트에서, 픽처에서의 타일들의 수를 나타내는 신택스 요소들을 파싱하는 것을 더 포함할 수 있고, 슬라이스 어드레스를 나타내는 신택스 요소의 파싱을 생략하는 것은 파싱된 신택스 요소들에 기반한다.

[0017] 디코딩하는 것은, 슬라이스 어드레스를 시그널링하기 위한 하나 이상의 신택스 요소 이전에, 슬라이스에서의 타일들의 수를 나타내는 신택스 요소를 슬라이스에서 파싱하는 것을 더 포함할 수 있다.

[0018] 디코딩하는 것은, 슬라이스에서, 픽처 헤더가 슬라이스 헤더에서 시그널링되는지를 나타내는 신택스 요소를 파싱하는 것, 및 파싱된 신택스 요소가 픽처 헤더가 슬라이스 헤더에서 시그널링된다는 것을 나타내는 경우, 슬라이스에서의 타일들의 수가 픽처에서의 타일들의 수와 동일하다고 결정(추론)하는 것을 더 포함할 수 있다.

[0019] 제4 양태에서, 비트스트림으로부터 비디오 데이터를 디코딩하는 방법이 제공되며, 비트스트림은 하나 이상의 슬라이스에 대응하는 비디오 데이터를 포함하고, 각각의 슬라이스는 하나 이상의 타일을 포함할 수 있고, 비트스트림은 하나 이상의 슬라이스를 디코딩할 때 이용될 신택스 요소들을 포함하는 픽처 헤더와, 슬라이스를 디코딩할 때 이용될 신택스 요소들을 포함하는 슬라이스 헤더를 포함하며, 디코딩하는 것은, 하나 이상의 신택스 요소를 파싱하는 것; 신택스 요소가 래스터 스캔 디코딩 모드가 슬라이스에 대해 인에이블된다는 것을 나타내는 경우, 하나 이상의 신택스 요소로부터 슬라이스 어드레스 및 슬라이스에서의 타일들의 수 중 적어도 하나를 디코

딩하는 것 - 래스터 스캔 디코딩 모드가 슬라이스에 대해 인에이블되는 경우, 하나 이상의 신택스 요소로부터 슬라이스 어드레스 및 슬라이스에서의 타일들의 수 중 적어도 하나를 디코딩하는 것은 픽처에서의 타일들의 수에 의존하지 않음 -; 및 상기 신택스 요소들을 이용하여 상기 비트스트림을 디코딩하는 것을 포함한다. 따라서, 슬라이스 헤더의 파싱 복잡도가 감소될 수 있다.

[0020] 본 발명에 따른 제5 양태에서, 제1 및 제2 양태들을 포함하는 방법이 제공된다.

[0021] 본 발명에 따른 제6 양태에서, 제1 및 제2 및 제3 양태들을 포함하는 방법이 제공된다.

[0022] 본 발명의 제7 양태에 따르면, 비디오 데이터를 비트스트림으로 인코딩하는 방법이 제공되며, 비트스트림은 하나 이상의 슬라이스에 대응하는 비디오 데이터를 포함하고, 각각의 슬라이스는 하나 이상의 타일을 포함할 수 있고, 비트스트림은 하나 이상의 슬라이스를 디코딩할 때 이용될 신택스 요소들을 포함하는 픽처 헤더와, 슬라이스를 인코딩할 때 이용될 신택스 요소들을 포함하는 슬라이스 헤더를 포함하며, 인코딩하는 것은, 비디오 데이터를 인코딩하기 위한 하나 이상의 신택스 요소를 결정하는 것; 슬라이스(또는 픽처)가 복수의 타일을 포함하는 경우, 픽처 헤더가 슬라이스 헤더에서 시그널링된다는 것을 신택스 요소가 나타낸다면 슬라이스의 어드레스를 나타내는 신택스 요소의 인코딩을 생략하는 것; 및 상기 신택스 요소들을 이용하여 상기 비디오 데이터를 인코딩하는 것을 포함한다. 본 발명의 추가적인 양태에 따르면, 비디오 데이터를 비트스트림으로 인코딩하는 방법이 제공되며, 비트스트림은 하나 이상의 슬라이스에 대응하는 비디오 데이터를 포함하고, 각각의 슬라이스는 하나 이상의 타일을 포함할 수 있고, 비트스트림은 하나 이상의 슬라이스를 디코딩할 때 이용될 신택스 요소들을 포함하는 픽처 헤더와, 슬라이스를 인코딩할 때 이용될 신택스 요소들을 포함하는 슬라이스 헤더를 포함하며, 인코딩하는 것은, 비디오 데이터를 인코딩하기 위한 하나 이상의 신택스 요소를 결정하는 것; 슬라이스 또는 픽처가 복수의 타일을 포함하는 경우, 픽처 헤더가 슬라이스 헤더에서 시그널링된다는 것을 신택스 요소가 나타낸다면 슬라이스의 어드레스를 나타내는 신택스 요소의 인코딩을 생략하는 것; 및 상기 신택스 요소들을 이용하여 상기 비디오 데이터를 인코딩하는 것을 포함한다. 본 발명의 추가적인 보충 양태에 따르면, 비디오 데이터를 비트스트림으로 인코딩하는 방법이 제공되며, 비트스트림은 하나 이상의 슬라이스에 대응하는 비디오 데이터를 포함하고, 각각의 슬라이스는 하나 이상의 타일을 포함할 수 있고, 비트스트림은 하나 이상의 슬라이스를 디코딩할 때 이용될 신택스 요소들을 포함하는 픽처 헤더와, 슬라이스를 인코딩할 때 이용될 신택스 요소들을 포함하는 슬라이스 헤더를 포함하며, 비트스트림은, 비트스트림이 슬라이스 또는 픽처가 복수의 타일을 포함한다는 것을 나타내는 값을 갖는 신택스 요소를 포함하고, 비트스트림이 픽처 헤더가 슬라이스 헤더에서 시그널링된다는 것을 나타내는 신택스 요소를 포함하는 경우, 비트스트림이 또한 슬라이스의 어드레스를 나타내는 신택스 요소가 파싱되지 않을 것임을 나타내는 신택스 요소를 포함하도록 제약되고, 이 방법은 상기 신택스 요소들을 이용하여 상기 비디오 데이터를 인코딩하는 단계를 포함한다.

[0023] 하나 이상의 실시예에서, 생략하는 것은 래스터 스캔 슬라이스 모드가 슬라이스를 인코딩하는데 이용될 때(에만) 수행될 것이다.

[0024] 생략하는 것은 슬라이스에서의 타일들의 수를 나타내는 신택스 요소의 인코딩을 생략하는 것을 더 포함할 수 있다.

[0025] 본 발명의 제8 양태에 따르면, 비디오 데이터를 비트스트림으로 인코딩하는 방법이 제공되며, 비트스트림은 하나 이상의 슬라이스에 대응하는 비디오 데이터를 포함하고, 각각의 슬라이스는 하나 이상의 타일을 포함할 수 있고, 비트스트림은 하나 이상의 슬라이스를 디코딩할 때 이용될 신택스 요소들을 포함하는 픽처 헤더와, 슬라이스를 디코딩할 때 이용될 신택스 요소들을 포함하는 슬라이스 헤더를 포함하며, 인코딩하는 것은, 비디오 데이터를 인코딩하기 위한 하나 이상의 신택스 요소를 결정하는 것; 슬라이스가 복수의 타일을 포함하는 경우, 픽처 헤더가 슬라이스 헤더에서 시그널링된다는 것을 나타내는 신택스 요소가 인코딩을 위해 결정된다면 슬라이스에서의 타일들의 수를 나타내는 신택스 요소의 인코딩을 생략하는 것; 및 상기 신택스 요소들을 이용하여 상기 비디오 데이터를 인코딩하는 것을 포함한다. 본 발명의 더 추가적인 양태에 따르면, 비디오 데이터를 비트스트림으로 인코딩하는 방법이 제공되며, 비트스트림은 하나 이상의 슬라이스에 대응하는 비디오 데이터를 포함하고, 각각의 슬라이스는 하나 이상의 타일을 포함할 수 있고, 비트스트림은 하나 이상의 슬라이스를 디코딩할 때 이용될 신택스 요소들을 포함하는 픽처 헤더와, 슬라이스를 디코딩할 때 이용될 신택스 요소들을 포함하는 슬라이스 헤더를 포함하며, 인코딩하는 것은, 비디오 데이터를 인코딩하기 위한 하나 이상의 신택스 요소를 결정하는 것; 슬라이스 또는 픽처가 복수의 타일을 포함하는 경우, 픽처 헤더가 슬라이스 헤더에서 시그널링된다는 것을 나타내는 신택스 요소가 인코딩을 위해 결정된다면 슬라이스에서의 타일들의 수를 나타내는 신택스 요소의 인코딩을 생략하는 것; 및 상기 신택스 요소들을 이용하여 상기 비디오 데이터를 인코딩하는 것을 포함

한다. 본 발명의 추가적인 보충 양태에 따르면, 비디오 데이터를 비트스트림으로 인코딩하는 방법이 제공되며, 비트스트림은 하나 이상의 슬라이스에 대응하는 비디오 데이터를 포함하고, 각각의 슬라이스는 하나 이상의 타일을 포함할 수 있고, 비트스트림은 하나 이상의 슬라이스를 디코딩할 때 이용될 신택스 요소들을 포함하는 픽처 헤더와, 슬라이스를 디코딩할 때 이용될 신택스 요소들을 포함하는 슬라이스 헤더를 포함하며, 비트스트림은, 비트스트림이 슬라이스 또는 픽처가 복수의 타일을 포함한다는 것을 나타내는 값을 갖는 신택스 요소를 포함하고, 비트스트림이 픽처 헤더가 슬라이스 헤더에서 시그널링된다는 것을 나타내는, 인코딩을 위해 결정되는 신택스 요소를 포함하는 경우, 비트스트림이 또한 슬라이스에서의 타일들의 수를 나타내는 신택스 요소가 파싱되지 않을 것임을 나타내는 신택스 요소를 포함하도록 제약되고, 이 방법은 상기 신택스 요소들을 이용하여 상기 비디오 데이터를 인코딩하는 단계를 포함한다.

[0026] 실시예에서, 생략하는 것은 래스터 스캔 슬라이스 모드가 슬라이스를 인코딩하는데 이용될 때(에만) 수행될 것이다.

[0027] 인코딩하는 것은 픽처에서의 타일들의 수를 나타내는 신택스 요소들을 인코딩하는 것을 더 포함할 수 있고, 슬라이스에서의 타일들의 수는 파싱된 신택스 요소들에 의해 표시된 픽처에서의 타일들의 수에 기반한다.

[0028] 생략하는 것은 슬라이스의 어드레스를 나타내는 신택스 요소의 인코딩을 생략하는 것을 더 포함할 수 있다.

[0029] 본 발명의 제9 양태에 따르면, 비디오 데이터를 비트스트림으로 인코딩하는 방법이 제공되며, 비트스트림은 하나 이상의 슬라이스에 대응하는 비디오 데이터를 포함하고, 각각의 슬라이스는 하나 이상의 타일을 포함할 수 있고, 비트스트림은 하나 이상의 슬라이스를 디코딩할 때 이용될 신택스 요소들을 포함하는 픽처 헤더와, 슬라이스를 디코딩할 때 이용될 신택스 요소들을 포함하는 슬라이스 헤더를 포함하며, 인코딩하는 것은, 하나 이상의 신택스 요소를 결정하는 것; 슬라이스(또는 픽처)가 복수의 타일을 포함하는 경우, 슬라이스에서의 타일들의 수가 픽처에서의 타일들의 수와 동일하다면 슬라이스 어드레스를 나타내는 신택스 요소의 인코딩을 생략하는 것; 및 상기 신택스 요소들을 이용하여 상기 비디오 데이터를 인코딩하는 것을 포함한다.

[0030] 하나 이상의 실시예에서, 생략하는 것은 래스터 스캔 슬라이스 모드가 슬라이스를 인코딩하는데 이용될 때(에만) 수행될 것이다.

[0031] 인코딩하는 것은, 슬라이스에서, 슬라이스에서의 타일들의 수를 나타내는 신택스 요소를 인코딩하는 것; 및 픽처 파라미터 세트에서, 픽처에서의 타일들의 수를 나타내는 신택스 요소들을 인코딩하는 것을 더 포함할 수 있고, 슬라이스 어드레스를 나타내는 신택스 요소의 인코딩의 생략 여부는 인코딩된 신택스 요소들의 값에 기반한다.

[0032] 인코딩하는 것은, 슬라이스 어드레스를 시그널링하기 위한 하나 이상의 신택스 요소 이전에, 슬라이스에서의 타일들의 수를 나타내는 신택스 요소를 슬라이스에서 인코딩하는 것을 더 포함할 수 있다.

[0033] 인코딩하는 것은, 픽처 헤더가 슬라이스 헤더에서 시그널링되는지를 나타내는 신택스 요소를 슬라이스에서 인코딩하는 것, 및 인코딩될 신택스 요소가 픽처 헤더가 슬라이스 헤더에서 시그널링된다는 것을 나타내는 경우, 슬라이스에서의 타일들의 수가 픽처에서의 타일들의 수와 동일하다고 결정하는 것을 더 포함할 수 있다.

[0034] 본 발명의 제10 양태에 따르면, 비디오 데이터를 비트스트림으로 인코딩하는 방법이 제공되며, 비트스트림은 하나 이상의 슬라이스에 대응하는 비디오 데이터를 포함하고, 각각의 슬라이스는 하나 이상의 타일을 포함할 수 있고, 비트스트림은 하나 이상의 슬라이스를 디코딩할 때 이용될 신택스 요소들을 포함하는 픽처 헤더와, 슬라이스를 디코딩할 때 이용될 신택스 요소들을 포함하는 슬라이스 헤더를 포함하며, 인코딩하는 것은, 비디오 데이터를 인코딩하기 위한 하나 이상의 신택스 요소를 결정하는 것; 인코딩을 위해 결정된 신택스 요소가 슬라이스에 대해 래스터 스캔 디코딩 모드가 인에이블된다는 것을 나타내는 경우, 슬라이스 어드레스 및 슬라이스에서의 타일들의 수 중 적어도 하나를 나타내는 신택스 요소들을 인코딩하는 것 - 슬라이스에 대해 래스터 스캔 디코딩 모드가 인에이블되는 경우, 하나 이상의 신택스 요소로부터 슬라이스 어드레스 및 슬라이스에서의 타일들의 수 중 적어도 하나를 인코딩하는 것은 픽처에서의 타일들의 수에 의존하지 않음 -; 및 상기 신택스 요소들을 이용하여 상기 비트스트림을 인코딩하는 것을 포함한다.

[0035] 본 발명에 따른 제11 양태에서, 제7 및 제8 양태들을 포함하는 방법이 제공된다.

[0036] 본 발명에 따른 제12 양태에서, 제7 및 제8 및 제9 양태들을 포함하는 방법이 제공된다.

[0037] 본 발명의 제13 양태에 따르면, 비트스트림으로부터 비디오 데이터를 디코딩하기 위한 디코더가 제공되며, 디코

더는 제1 양태 내지 제6 양태 중 어느 한 양태의 방법을 수행하도록 구성된다.

[0038] 본 발명의 제14 양태에 따르면, 비디오 데이터를 비트스트림으로 인코딩하기 위한 인코더가 제공되며, 인코더는 제7 양태 내지 제12 양태 중 어느 한 양태의 방법을 수행하도록 구성된다.

[0039] 본 발명의 제15 양태에 따르면, 실행 시에 제1 양태 내지 제12 양태 중 어느 한 양태의 방법이 수행되게 하는 컴퓨터 프로그램이 제공된다. 프로그램은 자체적으로 제공되거나 캐리어 매체 상에서, 캐리어 매체에 의해 또는 캐리어 매체 내에서 운반될 수 있다. 캐리어 매체는 비일시적, 예를 들어 저장 매체, 특히 컴퓨터 판독가능한 저장 매체일 수 있다. 캐리어 매체는 또한 일시적, 예를 들어 신호 또는 다른 전송 매체일 수 있다. 신호는 인터넷을 포함하는 임의의 적절한 네트워크를 통해 전송될 수 있다. 본 발명의 추가 특징들은 독립 청구항 및 종속 청구항에 의해 특징지어진다.

[0040] 본 발명의 일 양태에서의 임의의 특징은 임의의 적절한 조합으로 본 발명의 다른 양태들에 적용될 수 있다. 특히, 방법 양태들은 장치 양태들에 적용될 수 있고, 그 반대도 가능하다.

[0041] 또한, 하드웨어로 구현되는 특징들은 소프트웨어로 구현될 수 있고, 그 반대도 가능하다. 본 명세서에서의 소프트웨어 및 하드웨어 특징들에 대한 임의의 언급은 이에 따라 해석되어야 한다.

[0042] 본 명세서에 설명된 임의의 장치 특징은 또한 방법 특징으로서 제공될 수 있고, 그 반대도 가능하다. 본 명세서에서 사용되는 바와 같이, 수단 플러스 기능 특징들(means plus function features)은 적절하게 프로그래밍된 프로세서 및 연관된 메모리와 같은 그 대응하는 구조의 면에서 대안적으로 표현될 수 있다.

[0043] 또한, 본 발명의 임의의 양태들에서 설명되고 정의되는 다양한 특징들의 특정 조합들이 독립적으로 구현 및/또는 공급 및/또는 이용될 수 있음을 이해해야 한다.

도면의 간단한 설명

[0044] 이제, 첨부 도면들을 예로서 참조할 것이다.

도 1은 HEVC 및 VVC에서 이용되는 코딩 구조를 설명하는데 이용하기 위한 도면이다.

도 2는 본 발명의 하나 이상의 실시예가 구현될 수 있는 데이터 통신 시스템을 개략적으로 나타내는 블록도이다.

도 3은 본 발명의 하나 이상의 실시예가 구현될 수 있는 처리 디바이스의 구성요소들을 나타내는 블록도이다.

도 4는 본 발명의 실시예들에 따른 인코딩 방법의 단계들을 나타내는 흐름도이다.

도 5는 본 발명의 실시예들에 따른 디코딩 방법의 단계들을 나타내는 흐름도이다.

도 6은 예시적인 코딩 시스템 VVC에서의 비트스트림의 구조를 도시한다.

도 7은 예시적인 코딩 시스템 VVC에서의 비트스트림의 다른 구조를 도시한다.

도 8은 루마 모델링 크로마 스케일링(LMCS)을 도시한다.

도 9는 LMCS의 서브-툴을 도시한다.

도 10은 현재의 VVC 드래프트 표준의 래스터 스캔 슬라이스 모드 및 직사각형 슬라이스 모드를 도시한다.

도 11은 본 발명의 실시예들에 따른, 인코더 또는 디코더 및 통신 네트워크를 포함하는 시스템을 도시하는 도면이다.

도 12는 본 발명의 하나 이상의 실시예의 구현을 위한 컴퓨팅 디바이스의 개략적인 블록도이다.

도 13은 네트워크 카메라 시스템을 도시하는 도면이다.

도 14는 스마트폰을 도시하는 도면이다.

발명을 실시하기 위한 구체적인 내용

[0045] 도 1은 고효율 비디오 코딩(HEVC) 비디오 표준에서 이용되는 코딩 구조에 관한 것이다. 비디오 시퀀스(1)는 일련의 디지털 이미지들 i로 구성된다. 각각의 이러한 디지털 이미지는 하나 이상의 행렬로 표현된다. 행렬 계수들은 픽셀들을 나타낸다.

[0046] 시퀀스의 이미지(2)는 슬라이스들(3)로 분할될 수 있다. 슬라이스는 일부 경우들에서 전체 이미지를 구성할 수 있다. 이들 슬라이스들은 비-중첩 코딩 트리 유닛들(CTU들)로 분할된다. 코딩 트리 유닛(CTU)은 고효율 비디오 코딩(HEVC) 비디오 표준의 기본 처리 유닛이고 개념적으로 구조상 여러 이전 비디오 표준들에서 이용되었던 매크로블록 유닛들에 대응한다. CTU는 때때로 최대 코딩 유닛(LCU)이라고도 한다. CTU는 루마 및 크로마 성분 부분들을 가지며, 이들 성분 부분들 각각은 코딩 트리 블록(CTB)이라고 불린다. 이들 상이한 컬러 성분들은 도 1에 도시되지 않는다.

[0047] CTU는 일반적으로 크기가 64 픽셀 x 64 픽셀이다. 각각의 CTU는 차례로 쿼드트리 분해(quadtree decomposition)를 이용하여 보다 작은 가변 크기의 코딩 유닛들(CU들)(5)로 반복적으로 분할될 수 있다.

[0048] 코딩 유닛들은 기본 코딩 요소들이고, 예측 유닛(PU) 및 변환 유닛(TU)이라고 불리는 2가지 종류의 서브유닛에 의해 구성된다. PU 또는 TU의 최대 크기는 CU 크기와 동일하다. 예측 유닛은 픽셀 값들의 예측을 위한 CU의 파티션에 대응한다. 4개의 정사각형 PU로의 파티션 및 2개의 직사각형 PU로의 2개의 상이한 파티션을 포함하는 CU의 PU들로의 다양한 상이한 파티션들이 606에 의해 도시된 바와 같이 가능하다. 변환 유닛은 DCT를 이용한 공간적 변환을 겪는 기본 유닛이다. CU는 쿼드트리 표현(607)에 기반하여 TU들로 파티셔닝될 수 있다.

[0049] 각각의 슬라이스는 하나의 네트워크 추상화 계층(Network Abstraction Layer)(NAL) 유닛에 임베딩된다. 또한, 비디오 시퀀스의 코딩 파라미터들은 파라미터 세트들이라 불리는 전용 NAL 유닛들에 저장된다. HEVC 및 H.264/AVC에서, 2가지 종류의 파라미터 세트 NAL 유닛들이 이용된다: 첫째, 전체 비디오 시퀀스 동안 변경되지 않는 모든 파라미터들을 수집하는 시퀀스 파라미터 세트(SPS) NAL 유닛이다. 전형적으로, 이것은 코딩 프로파일, 비디오 프레임들의 크기 및 다른 파라미터들을 처리한다. 둘째, 픽처 파라미터 세트(PPS) NAL 유닛은 시퀀스의 하나의 이미지(또는 프레임)로부터 다른 것으로 변경할 수 있는 파라미터들을 포함한다. HEVC는 또한 비트스트림의 전체 구조를 기술하는 파라미터들을 포함하는 비디오 파라미터 세트(VPS) NAL 유닛을 포함한다. VPS는 HEVC에서 정의된 새로운 유형의 파라미터 세트이고, 비트스트림의 계층들 모두에 적용된다. 계층은 복수의 시간적 서브계층들을 포함할 수 있고, 모든 버전 1 비트스트림들은 단일 계층으로 제한된다. HEVC는 확장성 및 멀티뷰를 위한 특정 계층화된 확장들을 가지며, 이들은 역호환 버전 1 베이스 계층과 함께 복수의 계층들을 가능하게 할 것이다.

[0050] 다목적 비디오 코딩(VVC)의 현재 정의에서, 픽처의 파티셔닝을 위한 3개의 고레벨 가능성들, 즉 서브픽처들, 슬라이스들 및 타일들이 있다. 그 각각은 그들 자신의 특성들 및 유용성을 갖는다. 서브픽처들로의 파티셔닝은 비디오의 영역들의 공간 추출 및/또는 병합을 위한 것이다. 슬라이스들로의 파티셔닝은 이전 표준들과 유사한 개념에 기반하며, 다른 응용들에 이용될 수 있더라도 비디오 전송을 위한 패킷화에 대응한다. 타일들로의 파티셔닝은, 픽처를 픽처의 (거의) 동일한 크기의 독립적 코딩 영역들로 분할하기 때문에, 개념적으로 인코더 병렬화 툴이다. 그러나, 이 툴은 다른 응용들에도 이용될 수 있다.

[0051] 픽처의 파티셔닝의 이러한 3개의 고레벨 이용가능한 방식들이 함께 이용될 수 있기 때문에, 이들의 이용을 위한 몇몇 모드들이 존재한다. VVC의 현재의 드래프트 사양들에서 정의된 바와 같이, 슬라이스들의 2개의 모드가 정의된다. 래스터 스캔 슬라이스 모드의 경우, 슬라이스는 픽처의 타일 래스터 스캔에서 완전한 타일들의 시퀀스를 포함한다. 현재의 VVC 사양에서의 이 모드가 도 10(a)에 예시되어 있다. 이 도면에 도시된 바와 같이, 픽처는 12개의 타일 및 3개의 래스터 스캔 슬라이스로 파티셔닝되는 18x12 루마 CTU들을 포함한다.

[0052] 제2의 직사각형 슬라이스 모드의 경우, 슬라이스는 픽처의 직사각형 영역을 집합적으로 형성하는 다수의 완전한 타일들을 포함한다. 현재의 VVC 사양에서의 이 모드가 도 10(b)에 예시되어 있다. 이 예에서, 24개의 타일 및 9개의 직사각형 슬라이스로 파티셔닝되는 18x12 루마 CTU들을 갖는 픽처가 도시되어 있다.

[0053] 도 2는 본 발명의 하나 이상의 실시예가 구현될 수 있는 데이터 통신 시스템을 도시한다. 데이터 통신 시스템은 데이터 통신 네트워크(200)를 통해 데이터 스트림의 데이터 패킷들을 수신 디바이스, 이 경우에는 클라이언트 단말기(202)에 전송하도록 동작가능한 전송 디바이스, 이 경우에는 서버(201)를 포함한다. 데이터 통신 네트워크(200)는 WAN(Wide Area Network) 또는 LAN(Local Area Network)일 수 있다. 이러한 네트워크는, 예를 들어, 무선 네트워크(Wifi/802.11a 또는 b 또는 g), 이더넷 네트워크, 인터넷 네트워크, 또는 수 개의 상이한 네트워크들로 구성된 혼합 네트워크일 수 있다. 본 발명의 특정 실시예에서, 데이터 통신 시스템은 서버(201)가 동일한 데이터 콘텐츠를 복수의 클라이언트들에 전송하는 디지털 텔레비전 방송 시스템일 수 있다.

[0054] 서버(201)에 의해 제공되는 데이터 스트림(204)은 비디오 및 오디오 데이터를 나타내는 멀티미디어 데이터로 구성될 수 있다. 오디오 및 비디오 데이터 스트림들은, 본 발명의 일부 실시예들에서, 각각 마이크로폰 및 카메

라를 이용하여 서버(201)에 의해 캡처될 수 있다. 일부 실시예들에서, 데이터 스트림들은 서버(201) 상에 저장되거나 서버(201)에 의해 다른 데이터 제공자로부터 수신되거나, 서버(201)에서 생성될 수 있다. 서버(201)에는 비디오 및 오디오 스트림들을 인코딩하기 위한 인코더가 제공되어, 특히 인코더에 대한 입력으로서 제시되는 데이터의 보다 콤팩트한 표현인 전송을 위한 압축된 비트스트림을 제공한다.

[0055] 전송된 데이터의 품질 대 전송된 데이터의 양에 대한 보다 나은 비율을 획득하기 위해, 비디오 데이터의 압축은, 예를 들어, HEVC 포맷 또는 H.264/AVC 포맷을 따를 수 있다.

[0056] 클라이언트(202)는 전송된 비트스트림을 수신하고, 디스플레이 디바이스 상에서 비디오 이미지들을 그리고 라우드 스피커에 의해 오디오 데이터를 재생하기 위해 재구성된 비트스트림을 디코딩한다.

[0057] 도 2의 예에서 스트리밍 시나리오가 고려되지만, 본 발명의 일부 실시예들에서, 인코더와 디코더 사이의 데이터 통신이, 예를 들어, 광학 디스크 등의 미디어 저장 디바이스를 이용하여 수행될 수 있다는 것을 이해할 것이다.

[0058] 본 발명의 하나 이상의 실시예에서, 비디오 이미지는 최종 이미지에서 필터링된 픽셀들을 제공하기 위해 이미지의 재구성된 픽셀들에 적용하기 위한 보상 오프셋들을 나타내는 데이터와 함께 전송된다.

[0059] 도 3은 본 발명의 적어도 하나의 실시예를 구현하도록 구성된 처리 디바이스(300)를 개략적으로 도시한다. 처리 디바이스(300)는 마이크로컴퓨터, 워크스테이션 또는 경량 휴대용 디바이스와 같은 디바이스일 수 있다. 디바이스(300)는,

[0060] - CPU로 표시되는, 마이크로프로세서와 같은 중앙 처리 유닛(311);

[0061] - 본 발명을 구현하기 위한 컴퓨터 프로그램들을 저장하기 위한, ROM으로 표시된 판독 전용 메모리(306);

[0062] - 본 발명의 실시예들의 방법의 실행가능한 코드뿐만 아니라, 본 발명의 실시예들에 따라 디지털 이미지들의 시퀀스를 인코딩하는 방법 및/또는 비트스트림을 디코딩하는 방법을 구현하는데 필요한 변수들 및 파라미터들을 기록하도록 적응되는 레지스터들을 저장하기 위한, RAM으로 표시된 랜덤 액세스 메모리(312); 및

[0063] - 처리될 디지털 데이터가 전송되거나 수신되는 통신 네트워크(303)에 접속된 통신 인터페이스(302)

[0064] 에 접속된 통신 버스(313)를 포함한다.

[0065] 선택적으로, 장치(300)는 또한 다음과 같은 구성요소들을 포함할 수 있다:

[0066] - 본 발명의 하나 이상의 실시예의 방법들을 구현하기 위한 컴퓨터 프로그램들 및 본 발명의 하나 이상의 실시예의 구현 동안 이용되거나 생성되는 데이터를 저장하기 위한, 하드 디스크와 같은 데이터 저장 수단(304);

[0067] - 디스크(306)로부터 데이터를 판독하거나 상기 디스크 상에 데이터를 기입하도록 적응되는, 디스크(306)를 위한 디스크 드라이브(305);

[0068] - 키보드(310) 또는 임의의 다른 포인팅 수단에 의해, 데이터를 표시하고/하거나 사용자와의 그래픽 인터페이스로서 역할을 하기 위한 스크린(309).

[0069] 장치(300)는, 예를 들어, 디지털 카메라(320) 또는 마이크로폰(308) 등의 다양한 주변기기들에 접속될 수 있고, 그 각각은 멀티미디어 데이터를 장치(300)에 공급하기 위해 입력/출력 카드(도시되지 않음)에 접속되어 있다.

[0070] 통신 버스는 장치(300)에 포함되거나 이에 접속된 다양한 요소들 사이의 통신 및 상호운용성을 제공한다. 버스의 표현은 제한적이지 않고, 특히 중앙 처리 유닛은 직접 또는 장치(300)의 다른 요소에 의해 장치(300)의 임의의 요소에 명령어들을 통신하도록 동작가능하다.

[0071] 디스크(306)는, 예를 들어, 콤팩트 디스크(CD-ROM)(재기입가능하거나 그렇지 않음), ZIP 디스크 또는 메모리 카드 등의 임의의 정보 매체로, 그리고, 일반적인 표현으로, 마이크로컴퓨터에 의해 또는 마이크로프로세서에 의해 판독될 수 있고, 장치 내에 통합되거나 통합되지 않을 수 있으며, 가능하게는 이동식이고 그 실행이 구현될 본 발명에 따라 디지털 이미지들의 시퀀스를 인코딩하는 방법 및/또는 비트스트림을 디코딩하는 방법을 가능하게 하는 하나 이상의 프로그램을 저장하도록 적응될 수 있는 정보 저장 수단으로 대체될 수 있다.

[0072] 실행가능한 코드는 판독 전용 메모리(306)에, 하드 디스크(304) 상에, 또는 예를 들어 전술한 바와 같은 디스크(306) 등의 이동식 디지털 매체 상에 저장될 수 있다. 변형예에 따르면, 프로그램들의 실행가능한 코드는 실행되기 전에 하드 디스크(304)와 같은 장치(300)의 저장 수단들 중 하나에 저장되기 위해 인터페이스(302)를 통해 통신 네트워크(303)에 의해 수신될 수 있다.

[0073] 중앙 처리 유닛(311)은 본 발명에 따른 프로그램 또는 프로그램들의 소프트웨어 코드의 명령어들 또는 부분들, 전술한 저장 수단들 중 하나에 저장된 명령어들의 실행을 제어 및 지시하도록 적응된다. 전원이 켜지면, 비휘발성 메모리, 예를 들어 하드 디스크(304) 또는 판독 전용 메모리(306)에 저장된 프로그램 또는 프로그램들은 랜덤 액세스 메모리(312)로 전송되고, 랜덤 액세스 메모리(312)는 그 후 프로그램 또는 프로그램들의 실행가능한 코드뿐만 아니라, 본 발명을 구현하는데 필요한 변수들 및 파라미터들을 저장하기 위한 레지스터들을 포함한다.

[0074] 이 실시예에서, 장치는 본 발명을 구현하기 위해 소프트웨어를 이용하는 프로그래밍가능한 장치이다. 그러나, 대안적으로, 본 발명은 하드웨어로(예를 들어, 주문형 집적 회로 또는 ASIC의 형태로) 구현될 수 있다.

[0075] 도 4는 본 발명의 적어도 하나의 실시예에 따른 인코더의 블록도를 도시한다. 인코더는 접속된 모듈들에 의해 표현되고, 각각의 모듈은, 예를 들어, 디바이스(300)의 CPU(311)에 의해 실행될 프로그래밍 명령어들의 형태로, 본 발명의 하나 이상의 실시예에 따라 이미지들의 시퀀스 중의 이미지를 인코딩하는 적어도 하나의 실시예를 구현하는 방법의 적어도 하나의 대응하는 단계를 구현하도록 적응된다.

[0076] 원래 시퀀스의 디지털 이미지들 *i*0 내지 *i*n(401)이 인코더(400)에 의해 입력으로서 수신된다. 각각의 디지털 이미지는 픽셀들로 알려진 샘플들의 세트로 표현된다.

[0077] 비트스트림(410)은 인코딩 프로세스의 구현 후에 인코더(400)에 의해 출력된다. 비트스트림(410)은 복수의 인코딩 유닛 또는 슬라이스를 포함하고, 각각의 슬라이스는 슬라이스를 인코딩하는데 이용되는 인코딩 파라미터들의 인코딩 값들을 전송하기 위한 슬라이스 헤더와, 인코딩된 비디오 데이터를 포함하는 슬라이스 바디를 포함한다.

[0078] 입력 디지털 이미지들 *i*0 내지 *i*n(401)은 모듈(402)에 의해 픽셀들의 블록들로 분할된다. 블록들은 이미지 부분들에 대응하고 가변 크기들을 가질 수 있다(예를 들어, 4x4, 8x8, 16x16, 32x32, 64x64, 128x128 픽셀들 및 몇몇 직사각형 블록 크기들이 또한 고려될 수 있다). 코딩 모드가 각각의 입력 블록에 대해 선택된다. 두 가지 부류의 코딩 모드들, 즉 공간적 예측 코딩에 기반한 코딩 모드들(인트라 예측), 및 시간적 예측에 기반한 코딩 모드들(인터 코딩, 병합, 스킵)이 제공된다. 가능한 코딩 모드들이 테스트된다.

[0079] 모듈(403)은, 인코딩될 주어진 블록이 인코딩될 상기 블록의 이웃의 픽셀들로부터 계산된 예측자에 의해 예측되는 인트라 예측 프로세스를 구현한다. 인트라 코딩이 선택되는 경우, 선택된 인트라 예측자의 표시 및 주어진 블록과 그 예측자 사이의 차이가 인코딩되어 잔차를 제공한다.

[0080] 시간적 예측은 모션 추정 모듈(404) 및 모션 보상 모듈(405)에 의해 구현된다. 먼저, 참조 이미지들(416)의 세트 중에서 참조 이미지가 선택되고, 인코딩될 주어진 블록에 가장 가까운 영역인, 참조 영역 또는 이미지 부분으로 또한 불리는 참조 이미지의 부분이 모션 추정 모듈(404)에 의해 선택된다. 모션 보상 모듈(405)은 그 후 선택된 영역을 이용하여 인코딩될 블록을 예측한다. 선택된 참조 영역과, 잔차 블록으로 또한 불리는 주어진 블록 사이의 차이는 모션 보상 모듈(405)에 의해 계산된다. 선택된 참조 영역은 모션 벡터에 의해 표시된다.

[0081] 따라서, 양 경우들(공간적 예측 및 시간적 예측)에서, 잔차는 원래 블록으로부터 예측을 감산함으로써 계산된다.

[0082] 모듈(403)에 의해 구현되는 인트라(INTRA) 예측에서, 예측 방향이 인코딩된다. 시간적 예측에서, 적어도 하나의 모션 벡터가 인코딩된다. 모듈들(404, 405, 416, 418, 417)에 의해 구현되는 인터 예측에서, 적어도 하나의 모션 벡터 또는 이러한 모션 벡터를 식별하기 위한 데이터가 시간적 예측을 위해 인코딩된다.

[0083] 인터 예측이 선택되면 모션 벡터 및 잔차 블록에 대한 정보가 인코딩된다. 모션이 균일하다고 가정하여, 비트레이트를 추가로 감소시키기 위해, 모션 벡터는 모션 벡터 예측자에 대한 차이에 의해 인코딩된다. 모션 정보 예측자들의 세트의 모션 벡터 예측자들은 모션 벡터 예측 및 코딩 모듈(417)에 의해 모션 벡터 필드(418)로부터 획득된다.

[0084] 인코더(400)는 레이트-왜곡 기준과 같은 인코딩 비용 기준을 적용함으로써 코딩 모드의 선택을 위한 선택 모듈(406)을 추가로 포함한다. 리던던시들을 추가로 감소시키기 위해, 변환 모듈(407)에 의해 변환(예를 들어, DCT)이 잔차 블록에 적용되고, 이어서 획득된 변환된 데이터는 양자화 모듈(408)에 의해 양자화되고 엔트로피 인코딩 모듈(409)에 의해 엔트로피 인코딩된다. 마지막으로, 인코딩되는 현재 블록의 인코딩된 잔차 블록이 비트스트림(410)에 삽입된다.

[0085] 인코더(400)는 또한 후속 이미지들의 모션 추정을 위한 참조 이미지를 생성하기 위해 인코딩된 이미지의 디코딩

을 수행한다. 이것은 비트스트림을 수신하는 인코더 및 디코더가 동일한 참조 프레임들을 가질 수 있게 한다. 역양자화 모듈(411)은 양자화된 데이터의 역양자화를 수행하고, 역변환 모듈(412)에 의한 역변환이 이어진다. 역인트라 예측 모듈(413)은 주어진 블록에 대해 어느 예측자를 이용할지를 결정하기 위해 예측 정보를 이용하고, 역모션 보상 모듈(414)은 실제로 모듈(412)에 의해 획득된 잔차를 참조 이미지들(416)의 세트로부터 획득된 참조 영역에 추가한다.

[0086] 이어서, 재구성된 픽셀 프레임을 필터링하기 위해 모듈(415)에 의해 후필터링이 적용된다. 본 발명의 실시예들에서, 보상 오프셋들이 재구성된 이미지의 재구성된 픽셀들의 픽셀 값들에 추가되는 SAO 루프 필터가 이용된다.

[0087] 도 5는 본 발명의 실시예에 따른, 인코더로부터 데이터를 수신하는데 이용될 수 있는 디코더(60)의 블록도를 나타낸 것이다. 디코더는 접속된 모듈들로 표현되며, 각각의 모듈은, 예를 들어, 디바이스(300)의 CPU(311)에 의해 실행될 프로그래밍 명령어들의 형태로, 디코더(60)에 의해 구현되는 방법의 대응하는 단계를 구현하도록 적응된다.

[0088] 디코더(60)는 인코딩 유닛들을 포함하는 비트스트림(61)을 수신하고, 각각의 인코딩 유닛은 인코딩 파라미터들에 관한 정보를 포함하는 헤더 및 인코딩된 비디오 데이터를 포함하는 바디로 구성된다. VVC에서의 비트스트림의 구조는 도 6을 참조하여 아래에 더 상세히 설명된다. 도 4와 관련하여 설명된 바와 같이, 인코딩된 비디오 데이터는 엔트로피 인코딩되고, 모션 벡터 예측자들의 인덱스들은 주어진 블록에 대해 미리 결정된 수의 비트들로 인코딩된다. 수신된 인코딩된 비디오 데이터는 모듈(62)에 의해 엔트로피 디코딩된다. 잔차 데이터는 그 후 모듈(63)에 의해 역양자화되고, 그 후 픽셀 값들을 획득하기 위해 모듈(64)에 의해 역변환이 적용된다.

[0089] 코딩 모드를 나타내는 모드 데이터가 또한 엔트로피 디코딩되고, 그 모드에 기반하여, 인트라 유형 디코딩 또는 인터 유형 디코딩이 인코딩된 이미지 데이터 블록들에 대해 수행된다.

[0090] 인트라 모드의 경우에, 비트스트림에 지정된 인트라 예측 모드에 기반하여 역인트라 예측 모듈(65)에 의해 인트라 예측자가 결정된다.

[0091] 모드가 인터(INTER)인 경우, 인코더에 의해 이용되는 참조 영역을 찾기 위해 비트스트림으로부터 모션 예측 정보가 추출된다. 모션 예측 정보는 참조 프레임 인덱스 및 모션 벡터 잔차로 구성된다. 모션 벡터 디코딩 모듈(70)에 의해 모션 벡터를 획득하기 위해 모션 벡터 예측자가 모션 벡터 잔차에 추가된다.

[0092] 모션 벡터 디코딩 모듈(70)은 모션 예측에 의해 인코딩된 각각의 현재 블록에 대해 모션 벡터 디코딩을 적용한다. 현재 블록에 대한 모션 벡터 예측자의 인덱스가 획득되었다면, 현재 블록과 연관된 모션 벡터의 실제 값이 디코딩되어 모듈(66)에 의해 역모션 보상을 적용하는데 이용될 수 있다. 디코딩된 모션 벡터에 의해 표시된 참조 이미지 부분은 역모션 보상(66)을 적용하기 위해 참조 이미지(68)로부터 추출된다. 모션 벡터 필드 데이터(71)는 후속 디코딩된 모션 벡터들의 역예측에 이용되기 위해 디코딩된 모션 벡터로 업데이트된다.

[0093] 마지막으로, 디코딩된 블록이 획득된다. 후필터링은 후필터링 모듈(67)에 의해 적용된다. 디코딩된 비디오 신호(69)가 최종적으로 디코더(60)에 의해 제공된다.

[0094] 도 6은 JVET-Q2001-vD에 기술된 바와 같은 예시적인 코딩 시스템(VVC)에서의 비트스트림의 구성을 예시한다.

[0095] VVC 코딩 시스템에 따른 비트스트림(61)은 신택스 요소들 및 코딩된 데이터의 순서화된 시퀀스로 구성된다. 신택스 요소들 및 코딩된 데이터는 네트워크 추상화 계층(NAL) 유닛들(601-608)에 배치된다. 상이한 NAL 유닛 유형들이 존재한다. 네트워크 추상화 계층은 비트스트림을 실시간 프로토콜/인터넷 프로토콜, ISO 베이스 미디어 파일 포맷 등을 나타내는 RTP/IP와 같은 상이한 프로토콜들로 캡슐화하는 능력을 제공한다. 네트워크 추상화 계층은 또한 패킷 손실 복원력을 위한 프레임워크를 제공한다.

[0096] NAL 유닛들은 비디오 코딩 계층(VCL) NAL 유닛들 및 비-VCL NAL 유닛들로 분할된다. VCL NAL 유닛들은 실제 인코딩된 비디오 데이터를 포함한다. 비-VCL NAL 유닛들은 추가 정보를 포함한다. 이 추가 정보는 인코딩된 비디오 데이터 또는 디코딩된 비디오 데이터의 유용성을 향상시킬 수 있는 보충 데이터의 디코딩에 필요한 파라미터들일 수 있다. NAL 유닛들(606)은 슬라이스들에 대응하고 비트스트림의 VCL NAL 유닛들을 구성한다.

[0097] 상이한 NAL 유닛들(601-605)은 상이한 파라미터 세트들에 대응하고, 이러한 NAL 유닛들은 비-VCL NAL 유닛들이다. 디코더 파라미터 세트(DPS) NAL 유닛(301)은 주어진 디코딩 프로세스에 대해 일정한 파라미터들을 포함한다. 비디오 파라미터 세트(VPS) NAL 유닛(602)은 전체 비디오, 및 이에 따른 전체 비트스트림에 대해 정의된 파라미터들을 포함한다. DPS NAL 유닛은 VPS에서의 파라미터들보다 더 정적인 파라미터들을 정의할 수 있다.

즉, DPS의 파라미터들은 VPS의 파라미터보다 덜 빈번하게 변한다.

[0098] 시퀀스 파라미터 세트(SPS) NAL 유닛(603)은 비디오 시퀀스에 대해 정의된 파라미터들을 포함한다. 특히, SPS NAL 유닛은 비디오 시퀀스들의 서브픽처 레이아웃 및 연관된 파라미터들을 정의할 수 있다. 각각의 서브픽처에 연관된 파라미터들은 서브픽처에 적용되는 코딩 제약들을 지정한다. 특히, 이는 서브픽처들 간의 시간적 예측이 동일한 서브픽처로부터 오는 데이터로 제한된다는 것을 나타내는 플래그를 포함한다. 다른 플래그는 서브픽처 경계들에 걸쳐 루프 필터들을 인에이블 또는 디스에이블할 수 있다.

[0099] 픽처 파라미터 세트(PPS) NAL 유닛(604)의 PPS는 픽처 또는 픽처들의 그룹에 대해 정의된 파라미터들을 포함한다. 적응 파라미터 세트(APS) NAL 유닛(605)은 슬라이스 레벨에서 이용되는 루프 필터들, 전형적으로 ALF(Adaptive Loop Filter) 또는 재정형기 모델(또는 크로마 스케일링과의 루마 매핑(luma mapping with chroma scaling)(LMCS) 모델) 또는 스케일링 행렬들에 대한 파라미터들을 포함한다.

[0100] VVC의 현재 버전에서 제안된 바와 같은 PPS의 신택스는 루마 샘플들 내의 픽처의 크기 그리고 또한 타일들 및 슬라이스들 내의 각각의 픽처의 파티셔닝을 지정하는 신택스 요소들을 포함한다.

[0101] PPS는 프레임 내의 슬라이스 위치를 결정하는 것을 가능하게 하는 신택스 요소들을 포함한다. 서브픽처가 프레임 내에 직사각형 영역을 형성하기 때문에, 파라미터 세트 NAL 유닛들로부터 서브픽처에 속하는 슬라이스들의 세트, 타일들의 부분들 또는 타일들을 결정하는 것이 가능하다. APS에서와 같이 PPS는 전송되는 동일한 PPS들의 양을 제한하기 위한 ID 메커니즘을 갖는다.

[0102] PPS와 픽처 헤더 사이의 주요 차이점은 그 전송이고, PPS는 일반적으로 각각의 픽처에 대해 체계적으로 전송되는 PH에 비해 픽처들의 그룹에 대해 전송된다. 따라서, PH에 비해 PPS는 여러 픽처에 대해 일정할 수 있는 파라미터들을 포함한다.

[0103] 비트스트림은 또한 SEI(Supplemental Enhancement Information) NAL 유닛들(도 6에 표현되지 않음)을 포함할 수 있다. 비트스트림에서의 이러한 파라미터 세트들의 발생의 주기성은 가변적이다. 전체 비트스트림에 대해 정의되는 VPS는 비트스트림에서 한 번만 발생할 수 있다. 반대로, 슬라이스에 대해 정의되는 APS는 각각의 픽처 내의 각각의 슬라이스에 대해 한 번 발생할 수 있다. 실제로, 상이한 슬라이스들은 동일한 APS에 의존할 수 있고, 따라서 일반적으로 각각의 픽처 내의 슬라이스들보다 더 적은 APS가 존재한다. 특히, APS는 픽처 헤더에서 정의된다. 그러나, ALF APS는 슬라이스 헤더에서 정밀화될 수 있다.

[0104] 액세스 유닛 구분자(Access Unit Delimiter)(AUD) NAL 유닛(607)은 2개의 액세스 유닛을 분리한다. 액세스 유닛은 동일한 디코딩 타임스탬프를 갖는 하나 이상의 코딩된 픽처를 포함할 수 있는 NAL 유닛들의 세트이다. 이 선택적 NAL 유닛은 현재 VVC 사양에서 하나의 신택스 요소, 즉 pic_type만을 포함하고, 이 신택스 요소는 AU에서의 코딩된 픽처들의 모든 슬라이스들에 대한 slice_type 값들을 나타낸다. pic_type가 0과 동일하게 설정되면, AU는 인트라 슬라이스만을 포함한다. 이것이 1과 동일하면, P 및 I 슬라이스들을 포함한다. 이것이 2와 동일하면, B, P 또는 인트라 슬라이스를 포함한다. 이 NAL 유닛은 pic-type의 하나의 신택스 요소만을 포함한다.

[0105] **표 1**

신택스 AUD

access_unit_delimiter_rbsp( ) {	**기술자**
pic_type	u(3)
rbsp_trailing_bits( )	
}	

[0106]

[0107] JVET-Q2001-vD에서, *pic_type*는 다음과 같이 정의된다:

[0108] "*pic_type*는 AU 구분자 NAL 유닛을 포함하는 AU에서의 코딩된 픽처들의 모든 슬라이스들에 대한 *slice_type* 값들이 *pic_type*의 주어진 값에 대해 표 2에 열거된 세트의 멤버들임을 나타낸다. *pic_type*의 값은 이 사양의 이 버전에 따르는 비트스트림들에서 0, 1 또는 2와 동일할 것이다. *pic_type*의 다른 값들은 ITU-T | ISO/IEC에 의해 장래의 이용을 위해 예비된다. 이 사양의 이 버전에 따르는 디코더들은 *pic_type*의 예비된 값들을 무시할 것이다".

[0109] rbsp_trailing_bits()는 바이트의 끝에 정렬되기 위해 비트들을 추가하는 함수이다. 따라서, 이 함수 후에, 파싱된 비트스트림의 양은 정수 바이트이다.

[0110] **표 2**

***pic_type*의 해석**

pic_type	AU에 존재할 수 있는 slice_type 값들
0	I
1	P, I
2	B, P, I

[0111]

[0112] PH NAL 유닛(608)은 하나의 코딩된 픽처의 슬라이스들의 세트에 공통인 파라미터들을 그룹화하는 픽처 헤더 NAL 유닛이다. 픽처는 픽처의 슬라이스들에 의해 이용되는 AFL 파라미터들, 재정형기 모델 및 스케일링 행렬들을 나타내기 위해 하나 이상의 APS를 참조할 수 있다.

[0113] VCL NAL 유닛들(606) 각각은 슬라이스를 포함한다. 슬라이스는 전체 픽처 또는 서브픽처, 단일 타일 또는 복수의 타일 또는 타일의 일부에 대응할 수 있다. 예를 들어, 도 3의 슬라이스는 몇 개의 타일들(620)을 포함한다. 슬라이스는 슬라이스 헤더(610)와, 코딩된 블록들(640)로서 인코딩된 코딩된 픽셀 데이터를 포함하는 원시 바이트 시퀀스 페이로드(RBSP)(611)로 구성된다.

[0114] VVC의 현재 버전에서 제안된 바와 같은 PPS의 신택스는 루마 샘플들 내의 픽처의 크기 그리고 또한 타일들 및 슬라이스들 내의 각각의 픽처의 파티셔닝을 지정하는 신택스 요소들을 포함한다.

[0115] PPS는 프레임 내의 슬라이스 위치를 결정하는 것을 가능하게 하는 신택스 요소들을 포함한다. 서브픽처가 프레임에서 직사각형 영역을 형성하기 때문에, 파라미터 세트 NAL 유닛들로부터 서브픽처에 속하는 슬라이스들의 세트, 타일들의 부분들 또는 타일들을 결정하는 것이 가능하다.

[0116] **NAL 유닛 슬라이스**

[0117] NAL 유닛 슬라이스 계층은 표 3에 나타낸 바와 같이 슬라이스 헤더와 슬라이스 데이터를 포함한다.

[0118] **표 3**

슬라이스 계층 신택스

slice_layer_rbsp( ) {	기술자
slice_header( )	
slice_data( )	
rbsp_slice_trailing_bits( )	
}	

[0119]

[0120] **APS**

[0121] 적응 파라미터 세트(APS) NAL 유닛(605)은 신택스 요소들을 보여주는 표 4에 정의된다.

[0122] 표 4에 나타낸 바와 같이, aps_params_type 신택스 요소에 의해 주어진 3개의 가능한 유형의 APS가 있다:

[0123] • ALF_AP: ALF 파라미터들에 대해

[0124] • LMCS 파라미터들에 대한 LMCS_APS

[0125] • 스케일링 리스트 상대 파라미터들에 대한 SCALING_APS

[0126] 표 4

적응 파라미터 세트 신택스

adaptation_parameter_set_rbsp( ) {	기술자
adaptation_parameter_set_id	u(5)
aps_params_type	u(3)
if(aps_params_type == ALF_APS)	
alf_data( )	
else if(aps_params_type == LMCS_APS)	
lmcs_data( )	
else if(aps_params_type == SCALING_APS)	
scaling_list_data( )	
aps_extension_flag	u(1)
if(aps_extension_flag)	
while(more_rbsp_data( ))	
aps_extension_data_flag	u(1)
rbsp_trailing_bits( )	
}	

[0127]

[0128] 이들 3가지 유형의 APS 파라미터는 이하에서 차례로 논의된다.

[0129] ALF APS

[0130] ALF 파라미터들은 적응적 루프 필터 데이터 신택스 요소들(표 5)에 기술되어 있다. 먼저, ALF 필터들이 루마 및/또는 크로마에 대해 전송되는지 여부 그리고 CC-ALF(Cross Component Adaptive Loop Filtering)가 Cb 성분 및 Cr 성분에 대해 인에이블되는지를 지정하는데 4개의 플래그가 전용된다. 루마 필터 플래그가 인에이블되는 경우, 클립 값들이 시그널링되는지를 알기 위해 다른 플래그(*alf_luma_clip_flag*)가 디코딩된다. 이어서, 시그널링된 필터들의 수가 alf_luma_num_filters_signalled_minus1 신택스 요소를 이용하여 디코딩된다. 필요한 경우, ALF 계수 델타 "*alf_luma_coeff_delta_idx*"를 나타내는 신택스 요소가 각각의 인에이블된 필터에 대해 디코딩된다. 이어서, 각각의 필터의 각각의 계수에 대한 절대값 및 부호가 디코딩된다.

[0131] *alf_luma_clip_flag*가 인에이블되면, 각각의 인에이블된 필터의 각각의 계수에 대한 클립 인덱스가 디코딩된다.

[0132] 동일한 방식으로, 필요한 경우 ALF 크로마 계수들이 디코딩된다.

[0133] CC-ALF가 Cr 또는 Cb에 대해 인에이블되면, 필터의 수(*alf_cc_cb_filters_signalled_minus1* 또는 *alf_cc_cr_filters_signalled_minus1*)가 디코딩되고, 관련 계수들(*alf_cc_cb_mapped_coeff_abs* 및 *alf_cc_cb_coeff_sign* 또는 각각 *alf_cc_cr_mapped_coeff_abs* 및 *alf_cc_cr_coeff_sign*)이 디코딩된다.

[0134] 표 5

적응적 루프 필터 데이터 신택스

alf_data( ) {	기술자
alf_luma_filter_signal_flag	u(1)
alf_chroma_filter_signal_flag	u(1)
alf_cc_cb_filter_signal_flag	u(1)
alf_cc_cr_filter_signal_flag	u(1)
if(alf_luma_filter_signal_flag) {	
alf_luma_clip_flag	u(1)
alf_luma_num_filters_signalled_minus1	ue(v)
if(alf_luma_num_filters_signalled_minus1 > 0)	
for(filtIdx = 0; filtIdx < NumAlfFilters; filtIdx++)	
alf_luma_coeff_delta_idx[filtIdx]	u(v)
for(sfIdx = 0; sfIdx <= alf_luma_num_filters_signalled_minus1; sfIdx++)	
for(j = 0; j < 12; j++) {	
alf_luma_coeff_abs[sfIdx][j]	ue(v)
if(alf_luma_coeff_abs[sfIdx][j])	
alf_luma_coeff_sign[sfIdx][j]	u(1)
}	
if(alf_luma_clip_flag)	
for(sfIdx = 0; sfIdx <= alf_luma_num_filters_signalled_minus1; sfIdx++)	
for(j = 0; j < 12; j++)	
alf_luma_clip_idx[sfIdx][j]	u(2)
}	
if(alf_chroma_filter_signal_flag) {	
alf_chroma_clip_flag	u(1)
alf_chroma_num_alt_filters_minus1	ue(v)
for(altIdx = 0; altIdx <= alf_chroma_num_alt_filters_minus1; altIdx++) {	
for(j = 0; j < 6; j++) {	
alf_chroma_coeff_abs[altIdx][j]	ue(v)
if(alf_chroma_coeff_abs[altIdx][j] > 0)	
alf_chroma_coeff_sign[altIdx][j]	u(1)
}	
if(alf_chroma_clip_flag)	
for(j = 0; j < 6; j++)	
alf_chroma_clip_idx[altIdx][j]	u(2)
}	
}	

[0135]

if(alf_cc_cb_filter_signal_flag) {	
alf_cc_cb_filters_signalled_minus1	ue(v)
for(k = 0; k < alf_cc_cb_filters_signalled_minus1 + 1; k++) {	
for(j = 0; j < 7; j++) {	
alf_cc_cb_mapped_coeff_abs[k][j]	u(3)
if(alf_cc_cb_mapped_coeff_abs[k][j])	
alf_cc_cb_coeff_sign[k][j]	u(1)
}	
}	
}	
if(alf_cc_cr_filter_signal_flag) {	
alf_cc_cr_filters_signalled_minus1	ue(v)
for(k = 0; k < alf_cc_cr_filters_signalled_minus1 + 1; k++) {	
for(j = 0; j < 7; j++) {	
alf_cc_cr_mapped_coeff_abs[k][j]	u(3)
if(alf_cc_cr_mapped_coeff_abs[k][j])	
alf_cc_cr_coeff_sign[k][j]	u(1)
}	
}	
}	
}	

[0136]

[0137] **루마 매핑 및 크로마 스케일링 둘 다에 대한 LMCS 신택스 요소들**

[0138] 아래의 표 6은 aps_params_type 파라미터가 1(LMCS_APS)로 설정될 때 적응 파라미터 세트(APS) 신택스 구조에서 코딩되는 모든 LMCS 신택스 요소들을 제공한다. 코딩된 비디오 시퀀스에서 최대 4개의 LMCS APS가 이용될 수 있지만, 주어진 픽처에 대해 단일 LMCS APS만이 이용될 수 있다.

[0139] 이러한 파라미터들은 루마에 대한 순방향 및 역방향 매핑 함수들 및 크로마에 대한 스케일링 함수를 구축하는데 이용된다.

[0140] **표 6**

크로마 스케일링과의 루마 매핑 데이터 신택스

lmcs_data () {	**기술자**
lmcs_min_bin_idx	ue(v)
lmcs_delta_max_bin_idx	ue(v)
lmcs_delta_cw_prec_minus1	ue(v)
for(i = lmcs_min_bin_idx; i <= LmcsMaxBinIdx; i++) {	
lmcs_delta_abs_cw[i]	u(v)
if(lmcs_delta_abs_cw[i]) > 0)	
lmcs_delta_sign_cw_flag[i]	u(1)
}	
lmcs_delta_abs_crs	u(3)
if(lmcs_delta_abs_crs) > 0)	
lmcs_delta_sign_crs_flag	u(1)
}	

[0141]

[0142] **스케일링 리스트 APS**

[0143] 스케일링 리스트는 정량화에 이용되는 양자화 행렬을 업데이트할 가능성을 제공한다. VVC에서, 이 스케일링 행렬은 스케일링 리스트 데이터 신택스 요소들(표 7의 스케일링 리스트 데이터 신택스)에 기술된 바와 같이 APS에서 시그널링된다. 제1 신택스 요소는 플래그 *scaling_matrix_for_lfnst_disabled_flag*에 기반하여 스케일링 행렬이 LFNST(Low Frequency Non-Separable Transform) 툴에 이용되는지를 지정한다. 제2 신택스 요소는 스케일링 리스트가 크로마 성분들(*scaling_list_chroma_present_flag*)에 대해 이용되는 경우에 지정된다. 그 다음, 스케일링 행렬을 구축하는데 필요한 신택스 요소들(*scaling_list_copy_mode_flag*, *scaling_list_pred_mode_flag*, *scaling_list_pred_id_delta*, *scaling_list_dc_coef*, *scaling_list_delta_coef*)이 디코딩된다.

[0144] **표 7**

스케일링 리스트 데이터 신택스

scaling_list_data( ) {	기술자
scaling_matrix_for_lfnst_disabled_flag	u(1)
scaling_list_chroma_present_flag	u(1)
for(id = 0; id < 28; id ++)	
matrixSize = (id < 2) ? 2 : ((id < 8) ? 4 : 8)	
if(scaling_list_chroma_present_flag || (id % 3 == 2) || (id == 27)) {	
scaling_list_copy_mode_flag[id]	u(1)
if(!scaling_list_copy_mode_flag[id])	
scaling_list_pred_mode_flag[id]	u(1)
if((scaling_list_copy_mode_flag[id] || scaling_list_pred_mode_flag[id]) && id != 0 && id != 2 && id != 8)	
scaling_list_pred_id_delta[id]	ue(v)
if(!scaling_list_copy_mode_flag[id]) {	
nextCoef = 0	
if(id > 13) {	
scaling_list_dc_coef[id − 14]	se(v)
nextCoef += scaling_list_dc_coef[id − 14]	
}	
for(i = 0; i < matrixSize * matrixSize; i++) {	
x = DiagScanOrder[3][3][i][0]	
y = DiagScanOrder[3][3][i][1]	
if(!(id > 25 && x >= 4 && y >= 4)) {	
scaling_list_delta_coef[id][i]	se(v)
nextCoef += scaling_list_delta_coef[id][i]	
}	
ScalingList[id][i] = nextCoef	
}	
}	
}	
}	
}	

[0145]

[0146] **픽처 헤더**

[0147] 픽처 헤더는 다른 슬라이스 데이터 전의 각각의 픽처의 시작 부분에서 전송된다. 이것은 그 표준의 이전 드래프트들에서의 이전 헤더들에 비해 매우 크다. 모든 이러한 파라미터들의 완전한 설명은 JVET-Q2001-vD에서 발견될 수 있다. 표 9는 현재 픽처 헤더 디코딩 신택스에서의 이러한 파라미터들을 보여준다.

[0148] 디코딩될 수 있는 관련된 신택스 요소들은 다음과 관련된다:

[0149] • 이 픽처, 참조 프레임의 이용 여부

[0150] • 픽처 유형

[0151] • 출력 프레임

[0152] • 픽처 수

[0153] • 필요한 경우 서브픽처 이용

[0154] • 필요한 경우 참조 픽처 리스트들

[0155] • 필요한 경우 컬러 평면

[0156] • 오버라이드 플래그가 인에이블되는 경우 파티셔닝 업데이트

[0157] • 필요한 경우 델타 QP 파라미터들

[0158] • 필요한 경우 모션 정보 파라미터들

[0159] • 필요한 경우 ALF 파라미터들

[0160] • 필요한 경우 SAO 파라미터들

[0161] • 필요한 경우 정량화 파라미터들

[0162] • 필요한 경우 LMCS 파라미터들

[0163] • 필요한 경우 스케일링 리스트 파라미터들

[0164] • 필요한 경우 픽처 헤더 확장

[0165] • 기타 등등.

[0166] **픽처 "유형"**

[0167] 제1 플래그는 현재 픽처가 재동기화 픽처(IRAP 또는 GDR)인지를 나타내는 *gdr_or_irap_pic_flag*이다. 이 플래그가 참이면, 현재 픽처가 IRAP 또는 GDR 픽처인지를 알기 위해 *gdr_pic_flag*가 디코딩된다.

[0168] 이어서, 인터 슬라이스가 허용됨을 식별하기 위해 *ph_inter_slice_allowed_flag*가 디코딩된다.

[0169] 이들이 허용될 때, 플래그 *ph_intra_slice_allowed_flag*는 인트라 슬라이스가 현재 픽처에 대해 허용되는지를 알기 위해 디코딩된다.

[0170] 그 후, *non_reference_picture_flag*, PPS ID를 나타내는 *ph_pic_parameter_set_id* 및 픽처 순서 카운트 *ph_pic_order_cnt_lsb*가 디코딩된다. 픽처 순서 카운트는 현재 픽처의 번호를 제공한다.

[0171] 픽처가 GDR 또는 IRAP 픽처인 경우, 플래그 *no_output_of_prior_pics_flag*가 디코딩된다.

[0172] 그리고, 픽처가 GDR인 경우, *recovery_poc_cnt*가 디코딩된다. 이어서, 필요한 경우에 *ph_poc_msb_present_flag* 및 *poc_msb_val*이 디코딩된다.

[0173] **ALF**

[0174] 현재 픽처에 대한 중요한 정보를 기술하는 이들 파라미터들 이후에, ALF가 SPS 레벨에서 인에이블되고 ALF가 픽처 헤더 레벨에서 인에이블되면 ALF APS id 신택스 요소들의 세트가 디코딩된다. ALF는 *sps_alf_enabled_flag* 플래그 덕분에 SPS 레벨에서 인에이블된다. 그리고 ALF 시그널링은 1과 동일한 *alf_info_in_ph_flag* 덕분에 픽처 헤더 레벨에서 인에이블되고, 그렇지 않으면(0과 동일한 alf_info_in_ph_flag) ALF는 슬라이스 레벨에서 시그널링된다.

[0175] alf_info_in_ph_flag는 다음과 같이 정의된다:

[0176] "*1과 동일한 alf_info_in_ph_flag는 ALF 정보가 PH 신택스 구조에 존재하고 PH 신택스 구조를 포함하지 않는 PPS를 참조하는 슬라이스 헤더들에 존재하지 않는다는 것을 지정한다. 0과 동일한 alf_info_in_ph_flag는 ALF 정보가 PH 신택스 구조에 존재하지 않고 PH 신택스 구조를 포함하지 않는 PPS를 참조하는 슬라이스 헤더들에 존재할 수 있다는 것을 지정한다*".

[0177] 먼저, *ph_alf_enabled_present_flag*가 디코딩되어 *ph_alf_enabled_flag*가 디코딩되어야 하는지 여부를 결정한다. *ph_alf_enabled_flag*가 인에이블되면, ALF는 현재 픽처의 모든 슬라이스들에 대해 인에이블된다.

[0178] ALF가 인에이블되면, 루마에 대한 ALF APS id의 양은 *pic_num_alf_aps_ids_luma* 신택스 요소를 이용하여 디코딩된다. 각각의 APS id에 대해, 루마에 대한 APS id 값 "*ph_alf_aps_id_luma*"가 디코딩된다.

[0179] 크로마의 경우, ALF가 Cr만에 대해 또는 Cb만에 대해, 크로마에 대해 인에이블되는지 여부를 결정하기 위해 신택스 요소 *ph_alf_chroma_idc*가 디코딩된다. 이것이 인에이블되면, 크로마에 대한 APS ID의 값은

ph_alf_aps_id_chroma 신택스 요소를 이용하여 디코딩된다.

[0180] 이런 식으로, CC-ALF 방법에 대한 APS ID는 Cb 및/또는 CR 성분들에 대해 필요하다면 디코딩된다.

[0181] **LMCS**

[0182] LMCS가 SPS 레벨에서 인에이블되었다면 LMCS APS ID 신택스 요소들의 세트가 디코딩된다. 먼저, *ph_lmcs_enabled_flag*가 디코딩되어 LMCS가 현재 픽처에 대해 인에이블되는지 여부를 결정한다. LMCS가 인에이블되면, ID 값 *ph_lmcs_aps_id*가 디코딩된다. 크로마의 경우, 크로마에 대한 방법을 인에이블 또는 디스에이블하기 위해 *ph_chroma_residual_scale_flag*만이 디코딩된다.

[0183] **스케일링 리스트**

[0184] 그 다음, 스케일링 리스트 APS ID의 세트는, 스케일링 리스트가 SPS 레벨에서 인에이블된다면 디코딩된다. *ph_scaling_list_present_flag*는 현재 픽처에 대해 스케일링 행렬이 인에이블되는지 여부를 결정하기 위해 디코딩된다. 그 다음, APS ID의 값 *ph_scaling_list_aps_id*가 디코딩된다.

[0185] **서브픽처**

[0186] 서브픽처 파라미터들은 이들이 SPS에서 인에이블될 때 그리고 서브픽처 id 시그널링이 디스에이블되면 인에이블된다. 이것은 또한 가상 경계들에 관한 일부 정보를 포함한다. 서브픽처 파라미터들에 대해, 8개의 신택스 요소가 정의된다:

[0187] • *ph_virtual_boundaries_present_flag*

[0188] • *ph_num_ver_virtual_boundaries*

[0189] • *ph_virtual_boundaries_pos_x[i]*

[0190] • *ph_num_hor_virtual_boundaries*

[0191] • *ph_virtual_boundaries_pos_y[i]*

[0192] **출력 플래그**

[0193] 이러한 서브픽처 파라미터들은 존재한다면 *pic_output_flag*에 선행한다.

[0194] **참조 픽처 리스트들**

[0195] 참조 픽처 리스트들이 (1과 동일한 rpl_info_in_ph_flag 덕분에) 픽처 헤더에서 시그널링되면, 참조 픽처 리스트들에 대한 파라미터들 *ref_pic_lists()*가 디코딩되고 다음의 신택스 요소들을 포함한다:

[0196] • *rpl_sps_flag[]*

[0197] • *rpl_idx[]*

[0198] • *poc_lsb_lt[][]*

[0199] • *delta_poc_msb_present_flag[][]*

[0200] • *delta_poc_msb_cycle_lt[][]*

[0201] **파티셔닝**

[0202] 파티셔닝 파라미터들의 세트는 필요한 경우 디코딩되며, 이하의 신택스 요소들을 포함한다:

[0203] • *partition_constraints_override_flag*

[0204] • *ph_log2_diff_min_qt_min_cb_intra_slice_luma*

[0205] • *ph_max_mtt_hierarchy_depth_intra_slice_luma*

[0206] • *ph_log2_diff_max_bt_min_qt_intra_slice_luma*

[0207] • *ph_log2_diff_max_tt_min_qt_intra_slice_luma*

[0208] • *ph_log2_diff_min_qt_min_cb_intra_slice_chroma*

[0209] • *ph_max_mtt_hierarchy_depth_intra_slice_chroma*

[0210] • *ph_log2_diff_max_bt_min_qt_intra_slice_chroma*

[0211] • *ph_log2_diff_max_tt_min_qt_intra_slice_chroma*

[0212] • *ph_log2_diff_min_qt_min_cb_inter_slice*

[0213] • *ph_max_mtt_hierarchy_depth_inter_slice*

[0214] • *ph_log2_diff_max_bt_min_qt_inter_slice*

[0215] • *ph_log2_diff_max_tt_min_qt_inter_slice*

[0216] **가중 예측**

[0217] 가중 예측 방법이 PPS 레벨에서 인에이블되는 경우 그리고 가중 예측 파라미터들이 픽처 헤더에서 시그널링되는 경우(*wp_info_in_ph_flag*가 1과 동일한 경우), 가중 예측 파라미터들 *pred_weight_table()*가 디코딩된다.

[0218] *pred_weight_table()*는 양방향 예측의 가중 예측이 인에이블될 때 리스트 L0 및 리스트 L1에 대한 가중 예측 파라미터들을 포함한다. 가중 예측 파라미터들이 픽처 헤더에서 전송될 때, 각각의 리스트에 대한 가중치들의 수는 표 8의 *pred_weight_table()* 신택스 표에 나타낸 바와 같이 명시적으로 전송된다.

[0219] 표 8

가중 예측 파라미터 신택스

pred_weight_table( ) {	기술자
luma_log2_weight_denom	ue(v)
if(ChromaArrayType != 0)	
delta_chroma_log2_weight_denom	se(v)
if(wp_info_in_ph_flag)	
num_l0_weights	ue(v)
for(i = 0; i < NumWeightsL0; i++)	
luma_weight_l0_flag[i]	u(1)
if(ChromaArrayType != 0)	
for(i = 0; i < NumWeightsL0; i++)	
chroma_weight_l0_flag[i]	u(1)
for(i = 0; i < NumWeightsL0; i++) {	
if(luma_weight_l0_flag[i]) {	
delta_luma_weight_l0[i]	se(v)
luma_offset_l0[i]	se(v)
}	
if(chroma_weight_l0_flag[i])	
for(j = 0; j < 2; j++) {	
delta_chroma_weight_l0[i][j]	se(v)
delta_chroma_offset_l0[i][j]	se(v)
}	
}	
if(pps_weighted_bipred_flag && wp_info_in_ph_flag)	
num_l1_weights	ue(v)
for(i = 0; i < NumWeightsL1; i++)	
luma_weight_l1_flag[i]	u(1)
if(ChromaArrayType != 0)	
for(i = 0; i < NumWeightsL1; i++)	
chroma_weight_l1_flag[i]	u(1)
for(i = 0; i < NumWeightsL1; i++) {	
if(luma_weight_l1_flag[i]) {	
delta_luma_weight_l1[i]	se(v)
luma_offset_l1[i]	se(v)
}	
if(chroma_weight_l1_flag[i])	
for(j = 0; j < 2; j++) {	
delta_chroma_weight_l1[i][j]	se(v)
delta_chroma_offset_l1[i][j]	se(v)
}	
}	
}	

[0220]

[0221] **델타 QP**

[0222] 픽처가 인트라일 때, *ph_cu_qp_delta_subdiv_intra_slice* 및 *ph_cu_chroma_qp_offset_subdiv_intra_slice*가 필요한 경우 디코딩된다. 그리고, 인터 슬라이스가 허용되는 경우, 필요하다면, *ph_cu_qp_delta_subdiv_inter_slice* 및 *ph_cu_chroma_qp_offset_subdiv_inter_slice*가 디코딩된다. 마지막으로, 픽처 헤더 확장 신택스 요소들이 필요한 경우에 디코딩된다.

[0223] 모든 파라미터들 *alf_info_in_ph_flag*, *rpl_info_in_ph_flag*, *qp_delta_info_in_ph_flag*, *sao_info_in_ph_flag*, *dbf_info_in_ph_flag*, *wp_info_in_ph_flag*가 PPS에서 시그널링된다.

[0224] 표 9

픽처 헤더 구조

picture_header_structure( ) {	기술자
gdr_or_irap_pic_flag	u(1)
if(gdr_or_irap_pic_flag)	
gdr_pic_flag	u(1)
ph_inter_slice_allowed_flag	u(1)
if(ph_inter_slice_allowed_flag)	
ph_intra_slice_allowed_flag	u(1)
non_reference_picture_flag	u(1)
ph_pic_parameter_set_id	ue(v)
ph_pic_order_cnt_lsb	u(v)
if(gdr_or_irap_pic_flag)	
no_output_of_prior_pics_flag	u(1)
if(gdr_pic_flag)	
recovery_poc_cnt	ue(v)
for(i = 0; i < NumExtraPhBits; i++)	
ph_extra_bit[i]	u(1)
if(sps_poc_msb_flag) {	
ph_poc_msb_present_flag	u(1)
if(ph_poc_msb_present_flag)	
poc_msb_val	u(v)
}	
if(sps_alf_enabled_flag && alf_info_in_ph_flag) {	
ph_alf_enabled_flag	u(1)
if(ph_alf_enabled_flag) {	
ph_num_alf_aps_ids_luma	u(3)
for(i = 0; i < ph_num_alf_aps_ids_luma; i++)	
ph_alf_aps_id_luma[i]	u(3)
if(ChromaArrayType != 0)	
ph_alf_chroma_idc	u(2)
if(ph_alf_chroma_idc > 0)	
ph_alf_aps_id_chroma	u(3)
if(sps_ccalf_enabled_flag) {	

[0225]

ph_cc_alf_cb_enabled_flag	u(1)
if(ph_cc_alf_cb_enabled_flag)	
ph_cc_alf_cb_aps_id	u(3)
ph_cc_alf_cr_enabled_flag	u(1)
if(ph_cc_alf_cr_enabled_flag)	
ph_cc_alf_cr_aps_id	u(3)
}	
}	
}	
if(sps_lmcs_enabled_flag) {	
ph_lmcs_enabled_flag	u(1)
if(ph_lmcs_enabled_flag) {	
ph_lmcs_aps_id	u(2)
if(ChromaArrayType != 0)	
ph_chroma_residual_scale_flag	u(1)
}	
}	
if(sps_scaling_list_enabled_flag) {	
ph_scaling_list_present_flag	u(1)
if(ph_scaling_list_present_flag)	
ph_scaling_list_aps_id	u(3)
}	
if(sps_virtual_boundaries_enabled_flag && !sps_virtual_boundaries_present_flag) {	
ph_virtual_boundaries_present_flag	u(1)
if(ph_virtual_boundaries_present_flag) {	
ph_num_ver_virtual_boundaries	u(2)
for(i = 0; i < ph_num_ver_virtual_boundaries; i++)	
ph_virtual_boundaries_pos_x[i]	u(13)
ph_num_hor_virtual_boundaries	u(2)
for(i = 0; i < ph_num_hor_virtual_boundaries; i++)	
ph_virtual_boundaries_pos_y[i]	u(13)
}	
}	
if(output_flag_present_flag)	
pic_output_flag	u(1)
if(rpl_info_in_ph_flag)	
ref_pic_lists( )	
if(partition_constraints_override_enabled_flag)	
partition_constraints_override_flag	u(1)
if(ph_intra_slice_allowed_flag) {	
if(partition_constraints_override_flag) {	
ph_log2_diff_min_qt_min_cb_intra_slice_luma	ue(v)
ph_max_mtt_hierarchy_depth_intra_slice_luma	ue(v)
if(ph_max_mtt_hierarchy_depth_intra_slice_luma != 0) {	
ph_log2_diff_max_bt_min_qt_intra_slice_luma	ue(v)
ph_log2_diff_max_tt_min_qt_intra_slice_luma	ue(v)

[0226]

}	
if(qtbtt_dual_tree_intra_flag) {	
ph_log2_diff_min_qt_min_cb_intra_slice_chroma	ue(v)
ph_max_mtt_hierarchy_depth_intra_slice_chroma	ue(v)
if(ph_max_mtt_hierarchy_depth_intra_slice_chroma != 0) {	
ph_log2_diff_max_bt_min_qt_intra_slice_chroma	ue(v)
ph_log2_diff_max_tt_min_qt_intra_slice_chroma	ue(v)
}	
}	
}	
if(cu_qp_delta_enabled_flag)	
ph_cu_qp_delta_subdiv_intra_slice	ue(v)
if(pps_cu_chroma_qp_offset_list_enabled_flag)	
ph_cu_chroma_qp_offset_subdiv_intra_slice	ue(v)
}	
if(ph_inter_slice_allowed_flag) {	
if(partition_constraints_override_flag) {	
ph_log2_diff_min_qt_min_cb_inter_slice	ue(v)
ph_max_mtt_hierarchy_depth_inter_slice	ue(v)
if(ph_max_mtt_hierarchy_depth_inter_slice != 0) {	
ph_log2_diff_max_bt_min_qt_inter_slice	ue(v)
ph_log2_diff_max_tt_min_qt_inter_slice	ue(v)
}	
}	
if(cu_qp_delta_enabled_flag)	
ph_cu_qp_delta_subdiv_inter_slice	ue(v)
if(pps_cu_chroma_qp_offset_list_enabled_flag)	
ph_cu_chroma_qp_offset_subdiv_inter_slice	ue(v)
if(sps_temporal_mvp_enabled_flag) {	
ph_temporal_mvp_enabled_flag	u(1)
if(ph_temporal_mvp_enabled_flag && rpl_info_in_ph_flag) {	
ph_collocated_from_l0_flag	u(1)
if((ph_collocated_from_l0_flag && num_ref_entries[0][RplsIdx[0]] > 1) || (!ph_collocated_from_l0_flag && num_ref_entries[1][RplsIdx[1]] > 1))	
ph_collocated_ref_idx	ue(v)
}	
}	
mvd_l1_zero_flag	u(1)
if(sps_fpel_mmvd_enabled_flag)	
ph_fpel_mmvd_enabled_flag	u(1)
if(sps_bdof_pic_present_flag)	
ph_disable_bdof_flag	u(1)
if(sps_dmvr_pic_present_flag)	
ph_disable_dmvr_flag	u(1)
if(sps_prof_pic_present_flag)	

[0227]

ph_disable_prof_flag	u(1)
if((pps_weighted_pred_flag || pps_weighted_bipred_flag) && wp_info_in_ph_flag)	
pred_weight_table( )	
}	
if(qp_delta_info_in_ph_flag)	
ph_qp_delta	se(v)
if(sps_joint_cbcr_enabled_flag)	
ph_joint_cbcr_sign_flag	u(1)
if(sps_sao_enabled_flag && sao_info_in_ph_flag) {	
ph_sao_luma_enabled_flag	u(1)
if(ChromaArrayType != 0)	
ph_sao_chroma_enabled_flag	u(1)
}	
if(sps_dep_quant_enabled_flag)	
ph_dep_quant_enabled_flag	u(1)
if(sps_sign_data_hiding_enabled_flag && !ph_dep_quant_enabled_flag)	
pic_sign_data_hiding_enabled_flag	u(1)
if(deblocking_filter_override_enabled_flag && dbf_info_in_ph_flag) {	
ph_deblocking_filter_override_flag	u(1)
if(ph_deblocking_filter_override_flag) {	
ph_deblocking_filter_disabled_flag	u(1)
if(!ph_deblocking_filter_disabled_flag) {	
ph_beta_offset_div2	se(v)
ph_tc_offset_div2	se(v)
ph_cb_beta_offset_div2	se(v)
ph_cb_tc_offset_div2	se(v)
ph_cr_beta_offset_div2	se(v)
ph_cr_tc_offset_div2	se(v)
}	
}	
}	
if(picture_header_extension_present_flag) {	
ph_extension_length	ue(v)
for(i = 0; i < ph_extension_length; i++)	
ph_extension_data_byte[i]	u(8)
}	
}	

[0228]

[0229] **슬라이스 헤더**

[0230] 슬라이스 헤더는 각각의 슬라이스의 시작 부분에서 전송된다. 슬라이스 헤더는 약 65개의 신택스 요소를 포함한다. 이것은 이전의 비디오 코딩 표준들에서의 이전 슬라이스 헤더에 비해 매우 크다. 모든 슬라이스 헤더 파라미터들의 완전한 설명은 JVET-Q2001-vD에서 발견될 수 있다. 표 10은 현재 슬라이스 헤더 디코딩 신택스에서의 이러한 파라미터들을 보여준다.

[0231] 표 10

부분적 슬라이스 헤더

slice_header( ) {	기술자
picture_header_in_slice_header_flag	u(1)
if(picture_header_in_slice_header_flag)	
picture_header_structure( )	
if(subpic_info_present_flag)	
slice_subpic_id	u(v)
if((rect_slice_flag && NumSlicesInSubpic[CurrSubpicIdx] > 1) || (!rect_slice_flag && NumTilesInPic > 1))	
slice_address	u(v)
for(i = 0; i < NumExtraShBits; i++)	
sh_extra_bit[i]	u(1)
if(!rect_slice_flag && NumTilesInPic > 1)	
num_tiles_in_slice_minus1	ue(v)
if(ph_inter_slice_allowed_flag)	
slice_type	ue(v)
if(sps_alf_enabled_flag && !alf_info_in_ph_flag) {	
slice_alf_enabled_flag	u(1)
if(slice_alf_enabled_flag) {	
slice_num_alf_aps_ids_luma	u(3)
for(i = 0; i < slice_num_alf_aps_ids_luma; i++)	
slice_alf_aps_id_luma[i]	u(3)
if(ChromaArrayType != 0)	
slice_alf_chroma_idc	u(2)
if(slice_alf_chroma_idc)	
slice_alf_aps_id_chroma	u(3)
if(sps_ccalf_enabled_flag) {	
slice_cc_alf_cb_enabled_flag	u(1)
if(slice_cc_alf_cb_enabled_flag)	
slice_cc_alf_cb_aps_id	u(3)
slice_cc_alf_cr_enabled_flag	u(1)
if(slice_cc_alf_cr_enabled_flag)	
slice_cc_alf_cr_aps_id	u(3)
}	
}	
}	
if(separate_colour_plane_flag == 1)	
colour_plane_id	u(2)
if(!rpl_info_in_ph_flag && ((nal_unit_type != IDR_W_RADL && nal_unit_type != IDR_N_LP) || sps_idr_rpl_present_flag))	
ref_pic_lists( )	
if((rpl_info_in_ph_flag || ((nal_unit_type != IDR_W_RADL && nal_unit_type != IDR_N_LP) || sps_idr_rpl_present_flag)) && ((slice_type != I && num_ref_entries[0][RplsIdx[0]] > 1) || (slice_type == B && num_ref_entries[1][RplsIdx[1]] > 1)) {	
num_ref_idx_active_override_flag	u(1)
if(num_ref_idx_active_override_flag)	

[0232]

for(i = 0; i < (slice_type == B ? 2: 1); i++)	
if(num_ref_entries[i][RplsIdx[i]] > 1)	
num_ref_idx_active_minus1[i]	ue(v)
}	
if(slice_type != I) {	
if(cabac_init_present_flag)	
cabac_init_flag	u(1)
if(ph_temporal_mvp_enabled_flag && !rpl_info_in_ph_flag) {	
if(slice_type == B)	
slice_collocated_from_l0_flag	u(1)
if((slice_collocated_from_l0_flag && NumRefIdxActive[0] > 1) || (! slice_collocated_from_l0_flag && NumRefIdxActive[1] > 1))	
slice_collocated_ref_idx	ue(v)
}	
if(!wp_info_in_ph_flag && ((pps_weighted_pred_flag && slice_type == P) || (pps_weighted_bipred_flag && slice_type == B)))	
pred_weight_table( )	
}	
if(!qp_delta_info_in_ph_flag)	
slice_qp_delta	se(v)
if(pps_slice_chroma_qp_offsets_present_flag) {	
slice_cb_qp_offset	se(v)
slice_cr_qp_offset	se(v)
if(sps_joint_cbcr_enabled_flag)	
slice_joint_cbcr_qp_offset	se(v)
}	
if(pps_cu_chroma_qp_offset_list_enabled_flag)	
cu_chroma_qp_offset_enabled_flag	u(1)
if(sps_sao_enabled_flag && !sao_info_in_ph_flag) {	
slice_sao_luma_flag	u(1)
if(ChromaArrayType != 0)	
slice_sao_chroma_flag	u(1)
}	
if(deblocking_filter_override_enabled_flag && !dbf_info_in_ph_flag)	
slice_deblocking_filter_override_flag	u(1)
if(slice_deblocking_filter_override_flag) {	
slice_deblocking_filter_disabled_flag	u(1)
if(!slice_deblocking_filter_disabled_flag) {	
slice_beta_offset_div2	se(v)
slice_tc_offset_div2	se(v)
slice_cb_beta_offset_div2	se(v)
slice_cb_tc_offset_div2	se(v)
slice_cr_beta_offset_div2	se(v)
slice_cr_tc_offset_div2	se(v)
}	
}	

[0233]

slice_ts_residual_coding_disabled_flag	u(1)
if(ph_lmcs_enabled_flag)	
slice_lmcs_enabled_flag	u(1)
if(ph_scaling_list_present_flag)	
slice_scaling_list_present_flag	u(1)
if(NumEntryPoints > 0) {	
offset_len_minus1	ue(v)
for(i = 0; i < NumEntryPoints; i++)	
entry_point_offset_minus1[i]	u(v)
}	
if(slice_header_extension_present_flag) {	
slice_header_extension_length	ue(v)
for(i = 0; i < slice_header_extension_length; i++)	
slice_header_extension_data_byte[i]	u(8)
}	
byte_alignment( )	
}	

[0234]

[0235] 먼저, *picture_header_in_slice_header_flag*가 디코딩되어 picture_header_structure()가 슬라이스 헤더에 존

재하는지를 안다.

[0236] 그 후, *slice_subpic_id*는 필요하다면 디코딩되어 현재 슬라이스의 서브픽처 id를 결정한다. 그 다음, *slice_address*가 디코딩되어 현재 슬라이스의 어드레스를 결정한다. 슬라이스 어드레스는 현재 슬라이스 모드가 직사각형 슬라이스 모드(*rect_slice_flag*가 1과 동일함)인 경우에 그리고 현재 서브픽처에서의 슬라이스들의 수가 1보다 큰 경우에 디코딩된다. 슬라이스 어드레스는 또한 현재 슬라이스 모드가 래스터 스캔 모드(*rect_slice_flag*가 0과 동일함)인 경우에 그리고 현재 픽처에서의 타일들의 수가 PPS에서 정의된 변수들에 기반하여 계산된 1보다 큰 경우에 디코딩될 수 있다.

[0237] *num_tiles_in_slice_minus1*은 그 후 현재 픽처에서의 타일들의 수가 1보다 크고 현재 슬라이스 모드가 직사각형 슬라이스 모드가 아닌 경우 디코딩된다. 현재의 VVC 드래프트 사양들에서, *num_tiles_in_slice_minus1*은 다음과 같이 정의된다:

[0238] "*num_tiles_in_slice_minus1* + 1은, 존재할 때, 슬라이스에서의 타일들의 수를 지정한다. num_tiles_in_slice_minus1의 값은 0 내지 NumTilesInPic - 1(경계 포함)의 범위에 있어야 한다".

[0239] 그 후, *slice_type*가 디코딩된다.

[0240] ALF가 SPS 레벨(*sps_alf_enabled_flag*)에서 인에이블되는 경우 그리고 ALF가 슬라이스 헤더에서 시그널링되는 경우(0과 동일한 *alf_info_in_ph_flag*), ALF 정보가 디코딩된다. 이것은 현재 슬라이스에 대해 ALF가 인에이블됨을 나타내는 플래그(*slice_alf_enabled_flag*)를 포함한다. 이것이 인에이블되면, 루마에 대한 APS ALF ID의 수(*slice_num_alf_aps_ids_luma*)가 디코딩되고, 그 후 APS ID(*slice_alf_aps_id_luma[i]*)가 디코딩된다. 그 후, *slice_alf_chroma_idc*는 ALF가 크로마 성분들에 대해 인에이블되는지 그리고 어느 크로마 성분이 인에이블되는지를 알기 위해 디코딩된다. 그 다음, 크로마에 대한 APS ID *slice_alf_aps_id_chroma*가 필요한 경우 디코딩된다. 동일한 방식으로, *slice_cc_alf_cb_enabled_flag*는 CC ALF 방법이 인에이블되는지를 알기 위해, 필요하다면, 디코딩된다. CC ALF가 인에이블되면, CR 및/또는 CB에 대한 관련 APS ID는 CC ALF가 CR 및/또는 CB에 대해 인에이블되면 디코딩된다.

[0241] 컬러 평면들이 독립적으로 전송되는 경우(*separate_colour_plane_flag*가 1과 동일한 경우), *colour_plane_id*가 디코딩된다.

[0242] 참조 픽처 리스트들이 픽처 헤더에서 전송되지 않을 때(0과 동일한 *rpl_info_in_ph_flag*) 그리고 Nal 유닛이 IDR이 아닐 때 또는 참조 픽처 리스트들이 IDR 픽처들에 대해 전송되면(1과 동일한 *sps_idr_rpl_present_flag*), 참조 픽처 리스트 파라미터들이 디코딩되며, 이들은 픽처 헤더에서의 것들과 유사하다.

[0243] 참조 픽처 리스트들이 픽처 헤더에서 전송되거나(1과 동일한 *rpl_info_in_ph_flag*) 또는 Nal 유닛이 IDR이 아닌 경우 또는 참조 픽처 리스트들이 IDR 픽처들에 대해 전송되는 경우(1과 동일한 *sps_idr_rpl_present_flag*) 그리고 적어도 하나의 리스트에 대한 참조의 수가 1보다 큰 경우, 오버라이드 플래그 *num_ref_idx_active_override_flag*가 디코딩된다. 이 플래그가 인에이블되는 경우, 각각의 리스트에 대한 참조 인덱스가 디코딩된다.

[0244] 슬라이스 유형이 인트라가 아니고 필요하다면, *cabac_init_flag*가 디코딩된다. 참조 픽처 리스트들이 슬라이스 헤더에서 전송되고 다른 조건들이 되면, *slice_collocated_from_l0_flag*와 *slice_collocated_ref_idx*가 디코딩된다. 이러한 데이터는 병치된 CABAC 코딩 및 모션 벡터와 관련된다.

[0245] 동일한 방식으로, 슬라이스 유형이 인트라가 아닐 때, 가중 예측 파라미터들 *pred_weight_table()*가 디코딩된다.

[0246] *slice_qp_delta*는 델타 QP 정보가 슬라이스 헤더에서 전송되는 경우(0과 동일한 *qp_delta_info_in_ph_flag*) 디코딩된다. 필요한 경우, 신택스 요소들 *slice_cb_qp_offset*, *slice_cr_qp_offset*, *slice_joint_cbcr_qp_offset*, *cu_chroma_qp_offset_enabled_flag*가 디코딩된다.

[0247] SAO 정보가 슬라이스 헤더에서 전송되는 경우(0과 동일한 *sao_info_in_ph_flag*) 그리고 이것이 SPS 레벨에서 인에이블되는 경우(*sps_sao_enabled_flag*), SAO에 대한 인에이블된 플래그들은 루마 및 크로마 둘 다에 대해 디코딩된다: *slice_sao_luma_flag*, *slice_sao_chroma_flag*.

[0248] 이어서, 디블로킹 필터 파라미터들은 슬라이스 헤더에서 시그널링되는 경우(0과 동일한 *dbf_info_in_ph_flag*)

디코딩된다.

[0249] 플래그 *slice_ts_residual_coding_disabled_flag*는 변환 스킵 잔차 코딩 방법이 현재 슬라이스에 대해 인에이블되는지를 알기 위해 체계적으로 디코딩된다.

[0250] LMCS가 픽처 헤더에서 인에이블되었다면(1과 동일한 *ph_lmcs_enabled_flag*), 플래그 *slice_lmcs_enabled_flag*가 디코딩된다.

[0251] 동일한 방식으로, 스케일링 리스트가 픽처 헤더에서 인에이블된 경우(*phpic_scaling_list_presentenabled_flag*가 1과 동일한 경우), 플래그 *slice_scaling_list_present_flag*가 디코딩된다.

[0252] 그 후, 필요하다면 다른 파라미터들이 디코딩된다.

[0253] **슬라이스 헤더에서의 픽처 헤더**

[0254] 특정한 시그널링 방식에서, 픽처 헤더(708)는 도 7에 도시된 바와 같이 슬라이스 헤더(710) 내에서 시그널링될 수 있다. 그 경우, 픽처 헤더(608)만을 포함하는 NAL 유닛은 없다. NAL 유닛들(701-707)은 도 6에서의 각각의 NAL 유닛들(601-607)에 대응한다. 유사하게, 코딩 타일들(720) 및 코딩 블록들(740)은 도 6의 블록들(620 및 640)에 대응한다. 따라서, 이들 유닛들 및 블록들의 설명은 본 명세서에서 반복되지 않을 것이다. 이것은 플래그 picture_header_in_slice_header_flag 덕분에 슬라이스 헤더에서 인에이블될 수 있다. 또한, 픽처 헤더가 슬라이스 헤더 내에서 시그널링될 때, 픽처는 하나의 슬라이스만을 포함할 것이다. 따라서, 픽처당 하나의 픽처 헤더만이 항상 존재한다. 더욱이, 플래그 picture_header_in_slice_header_flag는 CLVS(Coded Layer Video Sequence)의 모든 픽처들에 대해 동일한 값을 가질 것이다. 이것은 제1 IRAP를 포함하는 2개의 IRAP 사이의 모든 픽처들이 픽처당 하나의 슬라이스만을 갖는다는 것을 의미한다.

[0255] 플래그 picture_header_in_slice_header_flag는 다음과 같이 정의된다:

[0256] "*1과 동일한 picture_header_in_slice_header_flag는 PH 신택스 구조가 슬라이스 헤더에 존재함을 지정한다. 0과 동일한 picture_header_in_slice_header_flag는 PH 신택스 구조가 슬라이스 헤더에 존재하지 않음을 지정한다.*

[0257] *picture_header_in_slice_header_flag의 값이 CLVS에서의 모든 코딩된 슬라이스들에서 동일해야 한다는 것이 비트스트림 적합성의 요건이다.*

[0258] *picture_header_in_slice_header_flag가 코딩된 슬라이스에 대해 1과 동일할 때, PH_NUT와 동일한 nal_unit_type를 갖는 어떠한 VCL NAL 유닛도 CLVS에 존재하지 않아야 한다는 것이 비트스트림 적합성의 요건이다.*

[0259] *picture_header_in_slice_header_flag가 0과 동일할 때, 현재 픽처에서의 모든 코딩된 슬라이스들은 picture_header_in_slice_header_flag가 0과 동일할 것이고, 현재 PU는 PH NAL 유닛을 가질 것이다.*

[0260] *picture_header_structure()는 스터핑 비트들 rbsp_trailing_bits()를 제외한 picture_rbsp()의 신택스 요소들을 포함한다*".

[0261] **스트리밍 애플리케이션들**

[0262] 일부 스트리밍 애플리케이션들은 비트스트림의 특정 부분들만을 추출한다. 이러한 추출들은 (서브픽처로서) 공간적 또는 시간적(비디오 시퀀스의 하위 부분)일 수 있다. 그 후, 이러한 추출된 부분들은 다른 비트스트림들과 병합될 수 있다. 일부 다른 애플리케이션들은 일부 프레임들만을 추출함으로써 프레임 레이트를 감소시킨다. 일반적으로, 이러한 스트리밍 애플리케이션들의 주요 목적은 허용된 대역폭의 최대치를 이용하여 최종 사용자에게 최대 품질을 생성하는 것이다.

[0263] VVC에서, APS ID 넘버링은 프레임에 대한 새로운 APS id 번호가 시간적 계층구조에서 상위 레벨에 있는 프레임에 대해 이용될 수 없도록 프레임 레이트 감소를 위해 제한되었다. 그러나, 비트스트림의 부분들을 추출하는 스트리밍 애플리케이션들의 경우, APS ID는 (IRAP로서) 프레임이 APS ID의 넘버링을 리셋하지 않으므로 비트스트림의 하위 부분에 대해 어느 APS가 유지되어야 하는지를 결정하기 위해 추적될 필요가 있다.

[0264] **LMCS(크로마 스케일링과의 루마 매핑)**

[0265] 크로마 스케일링과의 루마 매핑(LMCS) 기술은 VVC와 같은 비디오 디코더에서 루프 필터들을 적용하기 전에 블록

에 적용되는 샘플 값 변환 방법이다.

[0266] LMCS는 2개의 서브-툴로 분할될 수 있다. 후술하는 바와 같이, 제1 서브-툴은 루마 블록에 적용되는 반면, 제2 서브-툴은 크로마 블록들에 적용된다:

[0267] 1) 제1 서브-툴은 적응적 구간 선형 모델들(adaptive piecewise linear models)에 기반한 루마 성분의 인-루프 매핑이다. 루마 성분의 인-루프 매핑은 압축 효율을 향상시키기 위해 동적 범위에 걸쳐 코드워드들을 재분배함으로써 입력 신호의 동적 범위를 조정한다. 루마 매핑은 "매핑된 도메인"으로의 순방향 매핑 함수 및 "입력 도메인"에서 돌아오는 대응하는 역방향 매핑 함수를 이용한다.

[0268] 2) 제2 서브-툴은 루마-의존적 크로마 잔차 스케일링이 적용되는 크로마 성분들과 관련된다. 크로마 잔차 스케일링은 루마 신호와 그 대응하는 크로마 신호들 사이의 상호작용을 보상하도록 설계된다. 크로마 잔차 스케일링은 현재 블록의 상단 및/또는 좌측 재구성된 이웃 루마 샘플들의 평균 값에 의존한다.

[0269] VVC와 같은 비디오 코더에서의 대부분의 다른 툴들과 같이, LMCS는 SPS 플래그를 이용하여 시퀀스 레벨에서 인에이블/디스에이블될 수 있다. 크로마 잔차 스케일링이 인에이블되는지 여부는 또한 슬라이스 레벨에서 시그널링된다. 루마 매핑이 인에이블되면, 루마-의존적 크로마 잔차 스케일링이 인에이블되는지 여부를 나타내기 위해 추가 플래그가 시그널링된다. 루마 매핑이 이용되지 않을 때, 루마-의존적 크로마 잔차 스케일링은 완전히 디스에이블된다. 또한, 루마-의존적 크로마 잔차 스케일링은 그 크기가 4 이하인 크로마 블록들에 대해 항상 디스에이블된다.

[0270] 도 8은 루마 매핑 서브-툴에 대해 전술한 바와 같은 LMCS의 원리를 도시한다. 도 8의 빗금친 블록들은 루마 신호의 순방향 및 역방향 매핑을 포함하는 새로운 LMCS 기능 블록들이다. LMCS를 이용할 때, 일부 디코딩 동작들이 "매핑된 도메인"에서 적용된다는 점에 유의하는 것이 중요하다. 이러한 동작들은 이 도 8에서 파선들의 블록들에 의해 나타내진다. 이들은 통상적으로 역양자화, 역변환, 루마 인트라 예측, 및 루마 잔차로 루마 예측을 추가하는 것으로 구성되는 재구성 단계에 대응한다. 반대로, 도 8의 실선 블록들은 디코딩 프로세스가 원래의(즉, 비-매핑된) 도메인에 적용되는 곳을 나타내고, 이것은 루프 필터링, 예컨대 디블로킹, ALF, 및 SAO, 모션 보상된 예측, 및 참조 픽처들(DPB)로서 디코딩된 픽처들의 저장을 포함한다.

[0271] 도 9는 도 8과 유사한 도면을 도시하지만, 이번에 이것은 LMCS 툴의 크로마 스케일링 서브-툴에 대한 것이다. 도 9의 빗금친 블록은 루마-의존적 크로마 스케일링 프로세스를 포함하는 새로운 LMCS 기능 블록이다. 그러나, 크로마에서는, 루마 경우에 비해 일부 중요한 차이들이 있다. 여기서, 파선들의 블록에 의해 표현된 역양자화 및 역변환만이 크로마 샘플들에 대해 "매핑된 도메인"에서 수행된다. 인트라 크로마 예측, 모션 보상, 루프 필터링의 모든 다른 단계들은 원래의 도메인에서 수행된다. 도 9에 도시된 바와 같이, 스케일링 프로세스만이 존재하고, 루마 매핑에 대한 것과 같은 순방향 및 역방향 처리가 없다.

[0272] **구간 선형 모델을 이용한 루마 매핑**

[0273] 루마 매핑 서브-툴은 구간 선형 모델을 이용하고 있다. 이는 구간 선형 모델이 입력 신호 동적 범위를 16개의 동일한 하위 범위로 분리하고, 각각의 하위 범위에 대해, 그 선형 매핑 파라미터들이 그 범위에 할당된 코드워드들의 수를 이용하여 표현된다는 것을 의미한다.

[0274] **루마 매핑에 대한 시맨틱스**

[0275] 신택스 요소 *lmcs_min_bin_idx*는 크로마 스케일링과의 루마 매핑(LMCS) 구성 프로세스에서 이용되는 최소 빈 인덱스를 지정한다. *lmcs_min_bin_idx*의 값은 0 내지 15(경계 포함)의 범위에 있어야 한다.

[0276] 신택스 요소 *lmcs_delta_max_bin_idx*는 크로마 스케일링 구성 프로세스를 갖는 루마 매핑에서 이용되는 15와 최대 빈 인덱스 *LmcsMaxBinIdx* 사이의 델타 값을 지정한다. *lmcs_delta_max_bin_idx*의 값은 0 내지 15(경계 포함)의 범위에 있어야 한다. *LmcsMaxBinIdx*의 값은 15 - *lmcs_delta_max_bin_idx*와 동일하게 설정된다. *LmcsMaxBinIdx*의 값은 *lmcs_min_bin_idx*보다 크거나 같아야 한다.

[0277] 신택스 요소 *lmcs_delta_cw_prec_minus1* + 1은 신택스 *lmcs_delta_abs_cw[i]*의 표현에 이용되는 비트 수를 지정한다.

[0278] 신택스 요소 *lmcs_delta_abs_cw[i]*는 i번째 빈에 대한 절대 델타 코드워드 값을 지정한다.

[0279] 신택스 요소 *lmcs_delta_sign_cw_flag[i]*는 변수 *lmcsDeltaCW[i]*의 부호를 지정한다. *lmcs_delta_sign_cw_flag[i]*가 존재하지 않을 때, 이는 0과 동일한 것으로 추론된다.

[0280] **루마 매핑을 위한 LMCS 중간 변수 계산**

[0281] 순방향 및 역방향 루마 매핑 프로세스들을 적용하기 위해, 일부 중간 변수들 및 데이터 어레이들이 필요하다.

[0282] 우선, 변수 OrgCW는 다음과 같이 도출된다:

[0283] OrgCW = (1 << BitDepth) / 16

[0284] 그 다음, i = lmcs_min_bin_idx .. LmcsMaxBinIdx인 변수 lmcsDeltaCW[i]는 다음과 같이 계산된다:

[0285] lmcsDeltaCW[i] = (1 - 2*lmcs_delta_sign_cw_flag[i])*lmcs_delta_abs_cw[i]

[0286] 새로운 변수 lmcsCW[i]는 다음과 같이 도출된다:

[0287] - i = 0 .. lmcs_min_bin_idx - 1에 대해, lmcsCW[i]는 0으로 설정된다.

[0288] - i = lmcs_min_bin_idx .. LmcsMaxBinIdx에 대해, 다음이 적용된다:

[0289] lmcsCW[i] = OrgCW + lmcsDeltaCW[i]

[0290] lmcsCW[i]의 값은 (OrgCW>>3) 내지 (OrgCW<<3 - 1)(경계 포함)의 범위에 있어야 한다.

[0291] - i = LmcsMaxBinIdx + 1 .. 15에 대해, lmcsCW[i]는 0과 동일하게 설정된다.

[0292] i = 0 .. 16인 변수 InputPivot[i]는 다음과 같이 도출된다:

[0293] InputPivot[i] = i*OrgCW

[0294] i = 0 .. 16인 변수 LmcsPivot[i], i = 0 .. 15인 변수들 ScaleCoeff[i] 및 InvScaleCoeff[i]는 다음과 같이 계산된다:

```
LmcsPivot[ 0 ] = 0;
for( i = 0; i <= 15; i++ ) {
        LmcsPivot[ i + 1 ] = LmcsPivot[ i ] + lmcsCW[ i ]
        ScaleCoeff[ i ] = ( lmcsCW[ i ] * (1 << 11 ) + ( 1 <<
( Log2( OrgCW ) - 1 ) ) ) >> ( Log2( OrgCW ) )
        if( lmcsCW[ i ] == 0 )
                InvScaleCoeff[ i ] = 0
        else
                InvScaleCoeff[ i ] = OrgCW * ( 1 << 11 ) / lmcsCW[ i ]
```

[0295]

[0296] **순방향 루마 매핑**

[0297] 도 8에 도시된 바와 같이, LMCS가 루마에 적용될 때, *predMapSamples[i][j]*라고 하는 루마 리매핑된 샘플은 예측 샘플 *predSamples[i][j]*로부터 획득된다.

[0298] *predMapSamples[i][j]*는 다음과 같이 계산된다:

[0299] 우선, 인덱스 idxY가 위치(i, j)에서 예측 샘플 *predSamples[i][j]*로부터 계산된다:

[0300] idxY = predSamples[i][j] >> Log2(OrgCW)

[0301] 이어서, 섹션 0의 중간 변수들 idxY, LmcsPivot[idxY] 및 InputPivot[idxY]를 이용하여 predMapSamples[i][j]가 다음과 같이 도출된다:

```
predMapSamples[ i ][ j ] = LmcsPivot[ idxY ]
 + ( ScaleCoeff[ idxY ] * ( predSamples[ i ][ j ] - InputPivot[ idxY ] ) + ( 1 << 10 ) )
 >> 11
```

[0302]

[0303] **루마 재구성 샘플들**

[0304] 재구성 프로세스는 예측된 루마 샘플 *predMapSample[i][j]* 및 잔차 루마 샘플들 *resiSamples[i][j]*로부터 획득된다.

[0305] 재구성된 루마 픽처 샘플 *recSamples[i][j]*는 다음과 같이 *resiSamples[i][j]*에 *predMapSample[i][j]*를 추가함으로써 간단히 획득된다:

[0306] $$recSamples[i][j] = Clip1(predMapSamples[i][j]+ resiSamples[i][j])$$

[0307] 이러한 위의 관계에서, 클립 1 함수는 재구성된 샘플이 0과 1 << BitDepth - 1 사이인 것을 확인하기 위한 클리핑 함수이다.

[0308] **역방향 루마 매핑**

[0309] 도 8에 따라 역방향 루마 매핑을 적용할 때, 처리되고 있는 현재 블록의 각각의 샘플 *recSample[i][j]*에 대해 다음의 동작들이 적용된다:

[0310] 먼저, 인덱스 idxY가 위치(i,j)에서 재구성 샘플 *recSamples[i][j]*로부터 계산된다:

[0311] $$idxY = recSamples[i][j] >> Log2(OrgCW)$$

[0312] 역방향 매핑된 루마 샘플 invLumaSample[i][j]는 다음에 기반하여 다음과 같이 도출된다:

[0313] $$invLumaSample[i][j] = InputPivot[idxYInv] + (InvScaleCoeff[idxYInv] * (recSample[i][j] - LmcsPivot[idxYInv]) + (1 << 10)) >> 11$$

[0314] 이어서, 클리핑 동작을 수행하여 최종 샘플을 얻는다:

[0315] $$finalSample[i][j] = Clip1(invLumaSample[i][j])$$

[0316] **크로마 스케일링**

[0317] **크로마 스케일링에 대한 LMCS 시맨틱스**

[0318] 표 6에서의 신택스 요소 *lmcs_delta_abs_crs*는 변수 *lmcsDeltaCrs*의 절대 코드워드 값을 지정한다. *lmcs_delta_abs_crs*의 값은 0 및 7(경계 포함)의 범위에 있어야 한다. 존재하지 않을 때, *lmcs_delta_abs_crs*는 0과 동일한 것으로 추론된다.

[0319] 신택스 요소 *lmcs_delta_sign_crs_flag*는 변수 *lmcsDeltaCrs*의 부호를 지정한다. 존재하지 않을 때, *lmcs_delta_sign_crs_flag*는 0과 동일한 것으로 추론된다.

[0320] **크로마 스케일링을 위한 LMCS 중간 변수 계산**

[0321] 크로마 스케일링 프로세스를 적용하기 위해, 일부 중간 변수들이 필요하다.

[0322] 변수 *lmcsDeltaCrs*는 다음과 같이 도출된다:

[0323] $$lmcsDeltaCrs = (1 - 2 * lmcs\_delta\_sign\_crs\_flag) * lmcs\_delta\_abs\_crs$$

[0324] i = 0 ... 15인 변수 *ChromaScaleCoeff[i]*는 다음과 같이 도출된다:

[0325]
```
if( lmcsCW[ i ] == 0 )
  ChromaScaleCoeff[ i ] = ( 1 << 11 )
else
  ChromaScaleCoeff[ i ] = OrgCW * ( 1 << 11 ) / ( lmcsCW[ i ] + lmcsDeltaCrs )
```

[0326] **크로마 스케일링 프로세스**

[0327] 제1 단계에서, 현재 대응하는 크로마 블록 주위의 재구성된 루마 샘플들의 평균 루마 값을 계산하기 위해 변수 *invAvgLuma*가 도출된다. 평균 루마는 대응하는 크로마 블록 주위의 좌측 및 상단 루마 블록으로부터 계산된다.

[0328] 샘플이 이용가능하지 않은 경우, 변수 *invAvgLuma*는 다음과 같이 설정된다:

[0329] $$invAvgLuma = 1 << (BitDepth - 1)$$

[0330] 섹션 0의 중간 어레이들 *LmcsPivot[ ]*에 기반하여, 변수 idxYInv는 다음과 같이 도출된다:

```
For ( idxYInv = lmcs_min_bin_idx; idxYInv <= LmcsMaxBinIdx; idxYInv++ ) {
        if(invAvgLuma < LmcsPivot [ idxYInv + 1 ] )        break
}
IdxYInv = Min( idxYInv, 15 )
```

[0331]

[0332] 변수 varScale는 다음과 같이 도출된다:

[0333] $$varScale = ChromaScaleCoeff[idxYInv]$$

[0334] 변환이 현재 크로마 블록에 적용될 때, 재구성된 크로마 픽처 샘플 어레이 *recSamples*는 다음과 같이 도출된다:

```
recSamples[i ][j ] = Clip1( predSamples[ i ][ j ] +
Sign( resiSamples[ i ][ j ] ) * ( ( Abs( resiSamples[ i ][ j ] ) * varScale + ( 1 <<
10 ) ) >> 11 ) )
```

[0335]

[0336] 현재 블록에 대해 변환이 적용되지 않았다면, 다음이 적용된다:

[0337] $$recSamples[i][j] = Clip1(predSamples[i][j])$$

[0338] **인코더 고려사항**

[0339] LMCS 인코더의 기본 원리는 그 동적 범위 세그먼트들이 평균 분산보다 더 낮은 코드워드들을 갖는 범위들에 더 많은 코드워드들을 먼저 할당하는 것이다. 이것의 대안적인 공식에서, LMCS의 주요 목표는 평균 분산보다 더 높은 코드워드들을 갖는 그 동적 범위 세그먼트들에 더 적은 코드워드들을 할당하는 것이다. 이러한 방식으로, 픽처의 매끄러운 영역들이 평균보다 더 많은 코드워드들로 코딩될 것이고, 그 반대도 가능하다.

[0340] APS에 저장되는 LMCS 툴들의 모든 파라미터들(표 6 참조)은 인코더 측에서 결정된다. LMCS 인코더 알고리즘은 로컬 루마 분산의 평가에 기반하고, 전술한 기본 원리에 따라 LMCS 파라미터들의 결정을 최적화한다. 그 후, 최적화는 주어진 블록의 최종 재구성된 샘플들에 대한 최상의 PSNR 메트릭들을 얻기 위해 수행된다.

[0341] **실시예들**

[0342] **필요하지 않을 때 슬라이스 어드레스 신택스 요소의 회피**

[0343] 일 실시예에서, 픽처 헤더가 슬라이스 헤더에서 시그널링될 때, 슬라이스 어드레스 신택스 요소(*slice_address*)는 타일들의 수가 1보다 크더라도 값 0과 동일한 것으로 추론된다. 표 11은 이 실시예를 나타낸다.

[0344] 이 실시예의 이점은, 특히 낮은 지연 및 낮은 비트레이트 응용들의 경우, 비트레이트를 감소시키는 슬라이스 헤더에 픽처 헤더가 있을 때 슬라이스 어드레스가 파싱되지 않고, 픽처가 슬라이스 헤더에서 시그널링될 때 일부 구현들의 경우 파싱 복잡도를 감소시킨다는 것이다.

[0345] 실시예에서, 이것은 래스터 스캔 슬라이스 모드(*rect_slice_flag*가 0과 동일함)에 대해서만 적용된다. 이것은 일부 구현들의 경우 파싱 복잡도를 감소시킨다.

[0346] 표 11

수정들을 보여주는 부분적 슬라이스 헤더

slice_header( ) {	기술자
picture_header_in_slice_header_flag	u(1)
if(picture_header_in_slice_header_flag)	
picture_header_structure( )	
…	
if((rect_slice_flag && NumSlicesInSubpic[CurrSubpicIdx] > 1) || ((!rect_slice_flag && NumTilesInPic > 1) && !picture_header_in_slice_header_flag))	
slice_address	u(v)
…	

[0347]

[0348] **필요하지 않을 때 슬라이스에서의 타일들의 수의 전송의 회피**

[0349] 일 실시예에서, 슬라이스에서의 타일들의 수는 픽처 헤더가 슬라이스 헤더에서 전송될 때 전송되지 않는다. 표 12는 플래그 *picture_header_in_slice_header_flag*가 1과 동일하게 설정될 때 *num_tiles_in_slice_minus1* 신택스 요소가 전송되지 않는 이 실시예를 나타낸다. 이 실시예의 이점은, 특히 낮은 지연 및 낮은 비트레이트 응용들의 경우, 타일들의 수가 전송될 필요가 없기 때문에, 비트레이트가 감소된다는 것이다.

[0350] 실시예에서, 이것은 래스터 스캔 슬라이스 모드(*rect_slice_flag*가 0과 동일함)에 대해서만 적용된다. 이것은 일부 구현들의 경우 파싱 복잡도를 감소시킨다.

[0351] 표 12

수정들을 보여주는 부분적 슬라이스 헤더

slice_header( ) {	기술자
picture_header_in_slice_header_flag	u(1)
if(picture_header_in_slice_header_flag)	
picture_header_structure( )	
…	
if((!rect_slice_flag && NumTilesInPic > 1) && !picture_header_in_slice_header_flag)	
num_tiles_in_slice_minus1	ue(v)
…	

[0352]

[0353] **PPS 값 NumTilesInPic(시맨틱스)에 의한 예측**

[0354] 하나의 추가적인 실시예에서, 현재 슬라이스에서의 타일들의 수는 픽처 헤더가 슬라이스 헤더에서 전송될 때 픽처에서의 타일들의 수와 동일한 것으로 추론된다. 이것은 신택스 요소 *num_tiles_in_slice_minus1*의 시맨틱스에 다음과 같은 문장을 추가함으로써 설정될 수 있다: "*존재하지 않을 때, 변수 num_tiles_in_slice_minus1은 NumTilesInPic-1과 동일하게 설정된다*".

[0355] 여기서, 변수 *NumTilesInPic*는 픽처에 대한 타일들의 최대 수를 제공한다. 이 변수는 PPS에서 전송된 신택스 요소들에 기반하여 계산된다.

[0356] **슬라이스 어드레스 이전에 타일들의 수를 설정하고, slice_address의 필요하지 않은 전송을 회피함**

[0357] 일 실시예에서, 슬라이스에서의 타일들의 수에 전용인 신택스 요소는 슬라이스 어드레스 이전에 전송되며, 그 값은 슬라이스 어드레스를 디코딩할 필요가 있는지를 아는데 이용된다. 보다 정확하게는, 슬라이스 어드레스를 디코딩할 필요가 있는지를 알기 위해 슬라이스에서의 타일들의 수가 픽처에서의 타일들의 수와 비교된다. 실제로, 슬라이스에서의 타일들의 수가 픽처에서의 타일들의 수와 동일하다면, 현재 픽처가 하나의 슬라이스만을 포함하는 것이 확실하다.

[0358] 실시예에서, 이것은 래스터 스캔 슬라이스 모드(*rect_slice_flag*가 0과 동일함)에 대해서만 적용된다. 이것은 일부 구현들의 경우 파싱 복잡도를 감소시킨다.

[0359] 표 13은 이 실시예를 나타낸다. 신택스 요소 *num_tiles_in_slice_minus1*의 값이 변수 *NumTilesInPic* - 1과 동일하면, 신택스 요소 *slice_address*가 디코딩되지 않는다. *um_tiles_in_slice_minus1*이 변수 *NumTilesInPic* - 1과 동일할 때, *slice_address*는 0과 동일한 것으로 추론된다.

[0360] 표 13

수정들을 보여주는 부분적 슬라이스 헤더

slice_header( ) {	기술자
picture_header_in_slice_header_flag	u(1)
if(picture_header_in_slice_header_flag)	
picture_header_structure( )	
…	
if(!rect_slice_flag && NumTilesInPic > 1)	
num_tiles_in_slice_minus1	**ue(v)**
if((rect_slice_flag && NumSlicesInSubpic[CurrSubpicIdx] > 1) || ((!rect_slice_flag && NumTilesInPic > 1) && num_tiles_in_slice_minus1 != NumTilesInPic-1)	
slice_address	u(v)
…	

[0361]

[0362] 이 실시예의 이점은 슬라이스 어드레스가 전송되지 않으므로 조건이 참과 동일하게 설정될 때의 비트레이트 감소 및 파싱 복잡도 감소이다.

[0363] 일 실시예에서, 픽처 헤더가 슬라이스 헤더에서 전송될 때, 현재 슬라이스에서의 타일들의 수를 나타내는 신택스 요소는 디코딩되지 않고, 슬라이스에서의 타일들의 수는 1과 동일한 것으로 추론된다. 그리고, 슬라이스에서의 타일들의 수가 픽처에서의 타일들의 수와 동일할 때, 슬라이스 어드레스는 0과 동일한 것으로 추론되고, 관련 신택스 요소는 디코딩되지 않는다. 표 14는 이 실시예를 나타낸다.

[0364] 이것은 이들 2개의 실시예의 조합에 의해 획득되는 비트레이트 감소를 증가시킨다.

[0365] 표 14

수정들을 보여주는 부분적 슬라이스 헤더

slice_header( ) {	기술자
picture_header_in_slice_header_flag	u(1)
if(picture_header_in_slice_header_flag)	
picture_header_structure( )	
…	
if((!rect_slice_flag && NumTilesInPic > 1) && !picture_header_in_slice_header_flag)	
num_tiles_in_slice_minus1	**ue(v)**
if((rect_slice_flag && NumSlicesInSubpic[CurrSubpicIdx] > 1) || ((!rect_slice_flag && NumTilesInPic > 1) && num_tiles_in_slice_minus1 != NumTilesInPic-1)	
slice_address	u(v)
…	

[0366]

[0367] **불필요한 조건들 numTileInPic > 1의 제거**

[0368] 일 실시예에서, 현재 픽처에서의 타일들의 수가 1보다 클 필요가 있다는 조건은, 신택스 요소들 *slice_address* 및/또는 현재 슬라이스에서의 타일들의 수가 디코딩되기 위해, 래스터 스캔 슬라이스 모드가 인에이블될 때 테스트될 필요가 없다. 구체적으로, 현재 픽처에서의 타일들의 수가 1과 동일할 때, *rect_slice_flag* 값은 1과 동일한 것으로 추론된다. 그 결과, 그 경우에 래스터 스캔 슬라이스 모드가 인에이블될 수 없다. 표 15는 이 실시예를 나타낸다.

[0369] 이 실시예는 슬라이스 헤더의 파싱 복잡도를 감소시킨다.

[0370] 표 15

수정들을 보여주는 부분적 슬라이스 헤더

slice_header( ) {	기술자
…	
if((rect_slice_flag && NumSlicesInSubpic[CurrSubpicIdx] > 1) || (!rect_slice_flag ~~&& NumTilesInPic > 1~~))	
slice_address	u(v)
for(i = 0; i < NumExtraShBits; i++)	
sh_extra_bit[i]	u(1)
if(!rect_slice_flag ~~&& NumTilesInPic > 1~~)	
num_tiles_in_slice_minus1	ue(v)
…	

[0371]

[0372] 일 실시예에서, 픽처 헤더가 슬라이스 헤더에서 전송될 때 그리고 래스터 스캔 슬라이스 모드가 인에이블될 때, 현재 슬라이스에서의 타일들의 수를 나타내는 신택스 요소는 디코딩되지 않고, 슬라이스에서의 타일들의 수는 1과 동일한 것으로 추론된다. 그리고, 슬라이스에서의 타일들의 수가 픽처에서의 타일들의 수와 동일할 때 그리고 래스터 스캔 슬라이스 모드가 인에이블될 때, 슬라이스 어드레스는 0과 동일한 것으로 추론되고, 관련 신택스 요소 *slice_address*는 디코딩되지 않는다. 표 16은 이 실시예를 나타낸다.

[0373] 그 이점들은 비트레이트 감소 및 파싱 복잡도 감소이다.

[0374] 표 16

수정들을 보여주는 부분적 슬라이스 헤더

slice_header( ) {	기술자
picture_header_in_slice_header_flag	u(1)
if(picture_header_in_slice_header_flag)	
picture_header_structure( )	
…	
if(!rect_slice_flag && !picture_header_in_slice_header_flag)	
num_tiles_in_slice_minus1	**ue(v)**
if((rect_slice_flag && NumSlicesInSubpic[CurrSubpicIdx] > 1) || (!rect_slice_flag && num_tiles_in_slice_minus1 != NumTilesInPic-1)	
slice_address	u(v)
…	

[0375]

[0376] 구현들

[0377] 도 11은 본 발명의 실시예들에 따른, 인코더(150) 또는 디코더(100) 및 통신 네트워크(199) 중 적어도 하나를 포함하는 시스템(191, 195)을 도시한다. 실시예에 따르면, 시스템(195)은 콘텐츠(예를 들어, 비디오/오디오 콘텐츠를 표시/출력 또는 스트리밍하기 위한 비디오 및 오디오 콘텐츠)를 처리하고 이를 사용자에게 제공하기 위한 것이며, 사용자는 예를 들어, 디코더(100)를 포함하는 사용자 단말기 또는 디코더(100)와 통신가능한 사용자 단말기의 사용자 인터페이스를 통해 디코더(100)에 액세스한다. 이러한 사용자 단말기는, (제공된/스트리밍된) 콘텐츠를 사용자에게 제공/표시할 수 있는 컴퓨터, 모바일 폰, 태블릿 또는 임의의 다른 유형의 디바이스일 수 있다. 시스템(195)은 통신 네트워크(199)를 통해 (예를 들어, 더 이른 비디오/오디오가 표시/출력되고 있는 동안 연속적인 스트림 또는 신호의 형태로) 비트스트림(101)을 획득/수신한다. 실시예에 따르면, 시스템(191)은 콘텐츠를 처리하고 처리된 콘텐츠, 예를 들어, 나중에 표시/출력/스트리밍을 위해 처리된 비디오 및 오디오 콘텐츠를 저장하기 위한 것이다. 시스템(191)은 인코더(150)에 의해 수신 및 처리(본 발명에 따른 디블로킹 필터에 의한 필터링을 포함함)되는 이미지들(151)의 원래의 시퀀스를 포함하는 콘텐츠를 획득/수신하고, 인코더(150)는 통신 네트워크(191)를 통해 디코더(100)에 통신될 비트스트림(101)을 생성한다. 이어서, 비트스트림(101)은, 다수의 방식으로 디코더(100)에 통신되고, 예를 들어, 인코더(150)에 의해 미리 생성되고, 사용자가 저장 장치로부터 콘텐츠(즉, 비트스트림 데이터)를 요청할 때까지 (예를 들어, 서버 또는 클라우드 저장소 상에서) 통신 네트워크(199) 내의 저장 장치에 데이터로서 저장될 수 있고, 요청 시에 데이터는 저장 장치로부터 디코더(100)에 통신/스트리밍된다. 시스템(191)은 또한, (예를 들어, 사용자 단말기 상에 표시될 사용자 인터페이스에 대한 데이터를 통신함으로써) 사용자에게, 저장 장치에 저장된 콘텐츠에 대한 콘텐츠 정보(예를 들어, 콘텐츠의 제목 및 콘텐츠를 식별, 선택 및 요청하기 위한 다른 메타/저장 위치 데이터)를 제공/스트리밍하고,

콘텐츠에 대한 사용자 요청을 수신 및 처리하여 요청된 콘텐츠가 저장 장치로부터 사용자 단말기에 전달/스트리밍될 수 있게 하기 위한 콘텐츠 제공 장치를 포함할 수 있다. 대안적으로, 인코더(150)는 비트스트림(101)을 생성하고, 사용자가 콘텐츠를 요청할 때 이를 디코더(100)에 직접 통신/스트리밍한다. 그 다음, 디코더(100)는 비트스트림(101)(또는 신호)을 수신하고, 본 발명에 따른 디블로킹 필터에 의한 필터링을 수행하여 비디오 신호(109) 및/또는 오디오 신호를 획득/생성하며, 이는 그 후 사용자에게 요청된 콘텐츠를 제공하기 위해 사용자 단말기에 의해 이용된다.

[0378] 본 명세서에 설명된 기능들 또는 본 발명에 따른 방법/프로세스의 임의의 단계는 하드웨어, 소프트웨어, 펌웨어, 또는 이들의 임의의 조합으로 구현될 수 있다. 소프트웨어로 구현되는 경우, 단계들/기능들은 하나 이상의 명령어 또는 코드 또는 프로그램, 또는 컴퓨터 판독가능한 매체로서 저장되거나 그를 통해 전송될 수 있고, PC("개인용 컴퓨터"), DSP("디지털 신호 프로세서"), 회로, 회로부, 프로세서 및 메모리, 범용 마이크로프로세서 또는 중앙 처리 유닛, 마이크로제어기, ASIC("주문형 집적 회로"), 필드 프로그래밍가능한 로직 어레이들(FPGA들), 또는 다른 등가의 집적 또는 이산 로직 회로일 수 있는 프로그래밍가능한 컴퓨팅 머신과 같은 하나 이상의 하드웨어 기반 처리 유닛에 의해 실행될 수 있다. 따라서, 본 명세서에서 사용되는 용어 "프로세서"는 전술한 구조 중 어느 하나 또는 본 명세서에 설명된 기술들의 구현에 적합한 임의의 다른 구조를 지칭할 수 있다.

[0379] 본 발명의 실시예들은 또한 무선 핸드셋, 집적 회로(IC) 또는 JC 세트(예를 들어, 칩셋)를 포함하는 매우 다양한 디바이스들 또는 장치들에 의해 실현될 수 있다. 다양한 구성요소들, 모듈들, 또는 유닛들은 이러한 실시예들을 수행하도록 구성된 디바이스들/장치들의 기능적 양태들을 예시하기 위해 본 명세서에서 설명되지만, 반드시 상이한 하드웨어 유닛들에 의한 실현을 요구하는 것은 아니다. 오히려, 다양한 모듈들/유닛들은 코덱 하드웨어 유닛에 결합되거나, 적절한 소프트웨어/펌웨어와 함께 하나 이상의 프로세서를 포함하는 상호운용적 하드웨어 유닛들의 집합에 의해 제공될 수 있다.

[0380] 본 발명의 실시예들은, 전술된 실시예들 중 하나 이상의 모듈들/유닛들/기능들을 수행하기 위해 저장 매체 상에 기록된 컴퓨터 실행가능한 명령어들(예를 들어, 하나 이상의 프로그램)을 판독 및 실행하고/하거나 전술된 실시예들 중 하나 이상의 기능들을 수행하기 위한 하나 이상의 처리 유닛 또는 회로를 포함하는 시스템 또는 장치의 컴퓨터에 의해, 그리고 예를 들어, 전술된 실시예들 중 하나 이상의 기능들을 수행하기 위해 저장 매체로부터 컴퓨터 실행가능한 명령어들을 판독 및 실행하고/하거나 전술된 실시예들 중 하나 이상의 기능들을 수행하기 위해 하나 이상의 처리 유닛 또는 회로를 제어함으로써 시스템 또는 장치의 컴퓨터에 의해 수행되는 방법에 의해 실현될 수 있다. 컴퓨터는 컴퓨터 실행가능한 명령어들을 판독 및 실행하기 위해 별개의 컴퓨터들 또는 별개의 처리 유닛들의 네트워크를 포함할 수 있다. 컴퓨터 실행가능한 명령어들은, 예를 들어, 네트워크 또는 유형의 저장 매체를 통해 통신 매체 등의 컴퓨터 판독가능한 매체로부터 컴퓨터에 제공될 수 있다. 통신 매체는 신호/비트스트림/캐리어파일 수 있다. 유형의 저장 매체는, 예를 들어, 하드 디스크, 랜덤 액세스 메모리(RAM), 판독 전용 메모리(ROM), 분산형 컴퓨팅 시스템 저장소, 광학 디스크(예를 들어, 콤팩트 디스크(CD), 디지털 다기능 디스크(DVD), 또는 블루-레이 디스크(BD™)), 플래시 메모리 디바이스, 메모리 카드 등 중 하나 이상을 포함할 수 있는 "비일시적 컴퓨터 판독가능한 저장 매체"이다. 단계들/기능들 중 적어도 일부는 또한, FPGA("Field-Programmable Gate Array") 또는 ASIC("Application-Specific Integrated Circuit") 등의 머신 또는 전용 구성요소에 의해 하드웨어로 구현될 수 있다.

[0381] 도 12는 본 발명의 하나 이상의 실시예의 구현을 위한 컴퓨팅 디바이스(2000)의 개략적인 블록도이다. 컴퓨팅 디바이스(2000)는 마이크로컴퓨터, 워크스테이션 또는 경량 휴대용 디바이스와 같은 디바이스일 수 있다. 컴퓨팅 디바이스(2000)는 통신 버스를 포함하고, 이 통신 버스는, 마이크로프로세서와 같은 중앙 처리 유닛(CPU)(2001); 본 발명의 실시예들의 방법의 실행가능한 코드뿐만 아니라, 본 발명의 실시예들에 따른 이미지의 적어도 일부를 인코딩 또는 디코딩하기 위한 방법을 구현하는데 필요한 변수들 및 파라미터들을 기록하도록 적응된 레지스터들을 저장하기 위한 랜덤 액세스 메모리(RAM)(2002) - 그 메모리 용량은, 예를 들어, 확장 포트에 접속된 선택적인 RAM에 의해 확장될 수 있음 -; 본 발명의 실시예들을 구현하기 위한 컴퓨터 프로그램들을 저장하기 위한 판독 전용 메모리(ROM)(2003); 처리될 디지털 데이터가 전송 또는 수신되는 통신 네트워크에 전형적으로 접속되는 네트워크 인터페이스(NET)(2004)에 접속된다. 네트워크 인터페이스(NET)(2004)는 단일 네트워크 인터페이스일 수 있거나, 상이한 네트워크 인터페이스들(예를 들어, 유선 및 무선 인터페이스들, 또는 상이한 종류들의 유선 또는 무선 인터페이스들)의 세트로 구성될 수 있다. 데이터 패킷들은 CPU(2001)에서 실행되는 소프트웨어 애플리케이션의 제어 하에 전송을 위해 네트워크 인터페이스에 기입되거나, 수신을 위해 네트워크 인터페이스로부터 판독되고; 사용자 인터페이스(UI)(2005)는 사용자로부터 입력들을 수신하거나 정보를 사용자

에게 표시하는데 이용될 수 있고; 하드 디스크(HD)(2006)는 대용량 저장 디바이스로서 제공될 수 있고; 입력/출력 모듈(IO)(2007)은 비디오 소스 또는 디스플레이와 같은 외부 디바이스들로부터/로 데이터를 수신/전송하는데 이용될 수 있다. 실행가능한 코드는 ROM(2003)에, HD(2006) 상에 또는 예를 들어, 디스크와 같은 이동식 디지털 매체 상에 저장될 수 있다. 변형예에 따르면, 프로그램들의 실행가능한 코드는, 실행되기 전에, 통신 디바이스(2000)의 저장 수단들 중 하나, 예컨대 HD(2006) 내에 저장되기 위해, NET(2004)를 통해, 통신 네트워크에 의해 수신될 수 있다. CPU(2001)는 본 발명의 실시예들에 따른 프로그램 또는 프로그램들의 소프트웨어 코드의 명령어들 또는 부분들의 실행을 제어 및 지시하도록 적응되고, 명령어들은 전술한 저장 수단들 중 하나에 저장된다. 전원이 켜진 후에, CPU(2001)는 그 명령어들이 예를 들어 프로그램 ROM(2003) 또는 HD(2006)로부터 로딩된 후에 소프트웨어 애플리케이션과 관련된 주 RAM 메모리(2002)로부터의 명령어들을 실행할 수 있다. 이러한 소프트웨어 애플리케이션은 CPU(2001)에 의해 실행될 때, 본 발명에 따른 방법의 단계들이 수행되게 한다.

[0382] 또한, 본 발명의 다른 실시예에 따르면, 컴퓨터, 모바일 폰(셀룰러 폰), 테이블 또는 사용자에게 콘텐츠를 제공/표시할 수 있는 임의의 다른 유형의 디바이스(예를 들어, 디스플레이 장치)와 같은 사용자 단말기에 전술한 실시예에 따른 디코더가 제공된다는 것을 이해한다. 또 다른 실시예에 따르면, 전술한 실시예에 따른 인코더는 인코더가 인코딩할 콘텐츠를 캡처 및 제공하는 카메라, 비디오 카메라 또는 네트워크 카메라(예를 들어, 폐회로 텔레비전 또는 비디오 감시 카메라)를 또한 포함하는 이미지 캡처 장치에 제공된다. 2개의 이러한 예가 도 13 및 도 14를 참조하여 아래에 제공된다.

[0383] 네트워크 카메라

[0384] 도 13은 네트워크 카메라(2102) 및 클라이언트 장치(2104)를 포함하는 네트워크 카메라 시스템(2100)을 도시하는 도면이다.

[0385] 네트워크 카메라(2102)는 이미징 유닛(2106), 인코딩 유닛(2108), 통신 유닛(2110) 및 제어 유닛(2112)을 포함한다.

[0386] 네트워크 카메라(2102)와 클라이언트 장치(2104)는 네트워크(200)를 통해 서로 통신할 수 있도록 상호 접속된다.

[0387] 이미징 유닛(2106)은 렌즈 및 이미지 센서(예를 들어, 전하 결합 디바이스(CCD) 또는 상보형 금속 산화물 반도체(CMOS))를 포함하고, 객체의 이미지를 캡처하고 이미지에 기반하여 이미지 데이터를 생성한다. 이 이미지는 정지 이미지 또는 비디오 이미지일 수 있다.

[0388] 인코딩 유닛(2108)은 전술한 상기 인코딩 방법들을 이용하여 이미지 데이터를 인코딩한다.

[0389] 네트워크 카메라(2102)의 통신 유닛(2110)은 인코딩 유닛(2108)에 의해 인코딩된, 인코딩된 이미지 데이터를 클라이언트 장치(2104)에 전송한다.

[0390] 또한, 통신 유닛(2110)은 클라이언트 장치(2104)로부터 명령들을 수신한다. 이러한 명령들은 인코딩 유닛(2108)의 인코딩을 위한 파라미터들을 설정하기 위한 명령들을 포함한다.

[0391] 제어 유닛(2112)은 통신 유닛(2110)에 의해 수신된 명령들에 따라 네트워크 카메라(2102) 내의 다른 유닛들을 제어한다.

[0392] 클라이언트 장치(2104)는 통신 유닛(2114), 디코딩 유닛(2116), 및 제어 유닛(2118)을 포함한다.

[0393] 클라이언트 장치(2104)의 통신 유닛(2114)은 명령들을 네트워크 카메라(2102)에 전송한다.

[0394] 또한, 클라이언트 장치(2104)의 통신 유닛(2114)은 네트워크 카메라(2102)로부터 인코딩된 이미지 데이터를 수신한다.

[0395] 디코딩 유닛(2116)은 전술한 상기 디코딩 방법들을 이용하여, 인코딩된 이미지 데이터를 디코딩한다.

[0396] 클라이언트 장치(2104)의 제어 유닛(2118)은 통신 유닛(2114)에 의해 수신된 사용자 동작 또는 명령들에 따라 클라이언트 장치(2104) 내의 다른 유닛들을 제어한다.

[0397] 클라이언트 장치(2104)의 제어 유닛(2118)은 디코딩 유닛(2116)에 의해 디코딩된 이미지를 표시하도록 디스플레이 장치(2120)를 제어한다.

[0398] 클라이언트 장치(2104)의 제어 유닛(2118)은 또한 인코딩 유닛(2108)의 인코딩을 위한 파라미터들을 포함하는, 네트워크 카메라(2102)에 대한 파라미터들의 값들을 지정하기 위해 GUI(Graphical User Interface)를 표시하도

록 디스플레이 장치(2120)를 제어한다.

[0399] 클라이언트 장치(2104)의 제어 유닛(2118)은 또한 디스플레이 장치(2120)에 의해 표시된 GUI에 입력되는 사용자 동작에 따라 클라이언트 장치(2104) 내의 다른 유닛들을 제어한다.

[0400] 클라이언트 장치(2104)의 제어 유닛(2119)은 디스플레이 장치(2120)에 의해 표시된 GUI에 입력되는 사용자 동작에 따라, 네트워크 카메라(2102)에 대한 파라미터들의 값들을 지정하는 명령들을 네트워크 카메라(2102)에 전송하도록 클라이언트 장치(2104)의 통신 유닛(2114)을 제어한다.

[0401] 스마트폰

[0402] 도 14는 스마트폰(2200)을 도시하는 도면이다.

[0403] 스마트폰(2200)은 통신 유닛(2202), 디코딩 유닛(2204), 제어 유닛(2206), 디스플레이 유닛(2208), 이미지 기록 디바이스(2210) 및 센서들(2212)을 포함한다.

[0404] 통신 유닛(2202)은 네트워크(200)를 통해, 인코딩된 이미지 데이터를 수신한다.

[0405] 디코딩 유닛(2204)은 통신 유닛(2202)에 의해 수신된 인코딩된 이미지 데이터를 디코딩한다.

[0406] 디코딩 유닛(2204)은 전술한 상기 디코딩 방법들을 이용하여, 인코딩된 이미지 데이터를 디코딩한다.

[0407] 제어 유닛(2206)은 통신 유닛(2202)에 의해 수신된 사용자 동작 또는 명령들에 따라 스마트폰(2200) 내의 다른 유닛들을 제어한다.

[0408] 예를 들어, 제어 유닛(2206)은 디코딩 유닛(2204)에 의해 디코딩된 이미지를 표시하도록 디스플레이 유닛(2208)을 제어한다.

[0409] 본 발명이 실시예들을 참조하여 설명되었지만, 본 발명은 개시된 실시예들로 제한되지 않는다는 것을 이해해야 한다. 본 기술분야의 통상의 기술자라면, 첨부된 청구항들에 정의된 바와 같은 본 발명의 범위로부터 벗어나지 않고 다양한 변경들 및 수정이 이루어질 수 있다는 것을 이해할 것이다. (임의의 첨부된 청구항들, 요약서 및 도면들을 포함한) 본 명세서에 개시된 모든 특징들, 및/또는 이와 같이 개시된 임의의 방법 또는 프로세스의 모든 단계들은, 이러한 특징들 및/또는 단계들 중 적어도 일부가 상호 배타적인 조합들을 제외하고는, 임의의 조합으로 결합될 수 있다. (임의의 첨부된 청구항들, 요약서 및 도면들을 포함한) 본 명세서에 개시된 각각의 특징은, 명시적으로 달리 언급되지 않는 한, 동일하거나, 동등하거나 유사한 목적을 제공하는 대안적인 특징들로 대체될 수 있다. 따라서, 명시적으로 달리 언급되지 않는 한, 개시된 각각의 특징은 일반적인 일련의 동등하거나 유사한 특징들의 일 예에 불과하다.

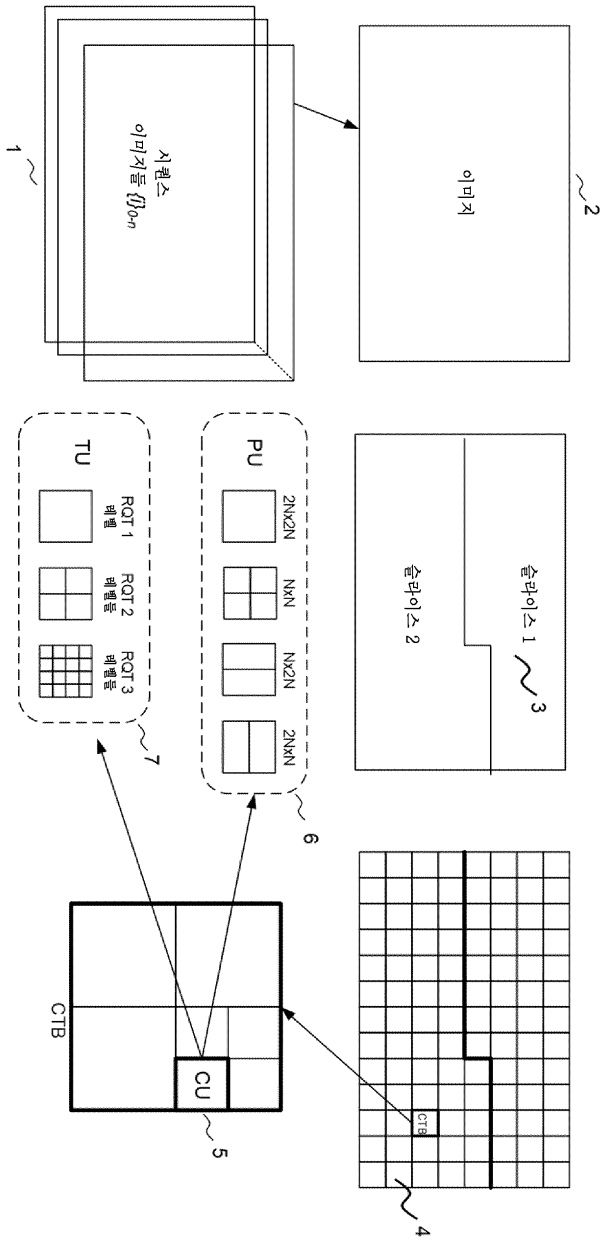
[0410] 또한, 예를 들어 디코딩 프로세스 동안, 비교, 결정, 평가, 선택, 실행, 수행 또는 고려를 실제로 수행하는 대신에, 표시된 또는 결정된/추론된 결과가 처리에서 이용될 수 있도록, 위에서 설명된 비교, 결정, 평가, 선택, 실행, 수행 또는 고려, 예를 들어, 인코딩 또는 필터링 프로세스 동안 이루어진 선택의 임의의 결과가 비트스트림에서의 데이터, 예를 들어 결과를 나타내는 플래그 또는 데이터에 표시되거나 그로부터 결정가능/추론가능할 수 있음이 이해된다.

[0411] 청구항들에서, 단어 "포함하는"은 다른 요소들 또는 단계들을 배제하지 않고, 단수형은 복수를 배제하지 않는다. 상이한 특징들이 서로 상이한 종속항들에서 인용된다는 단순한 사실은 이러한 특징들의 조합이 유리하게 이용될 수 없다는 것을 나타내지 않는다.

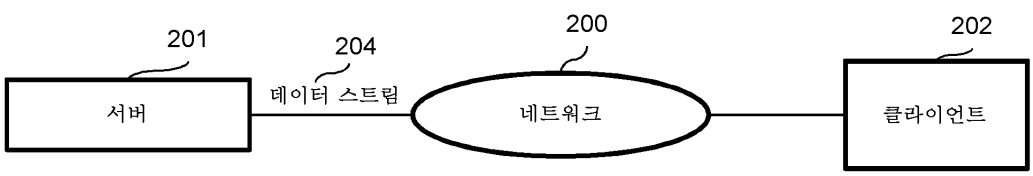
[0412] 청구항들에 나타나는 참조 번호들은 예시에 불과하고, 청구항들의 범위를 제한하는 효과를 갖지 않아야 한다.

도면

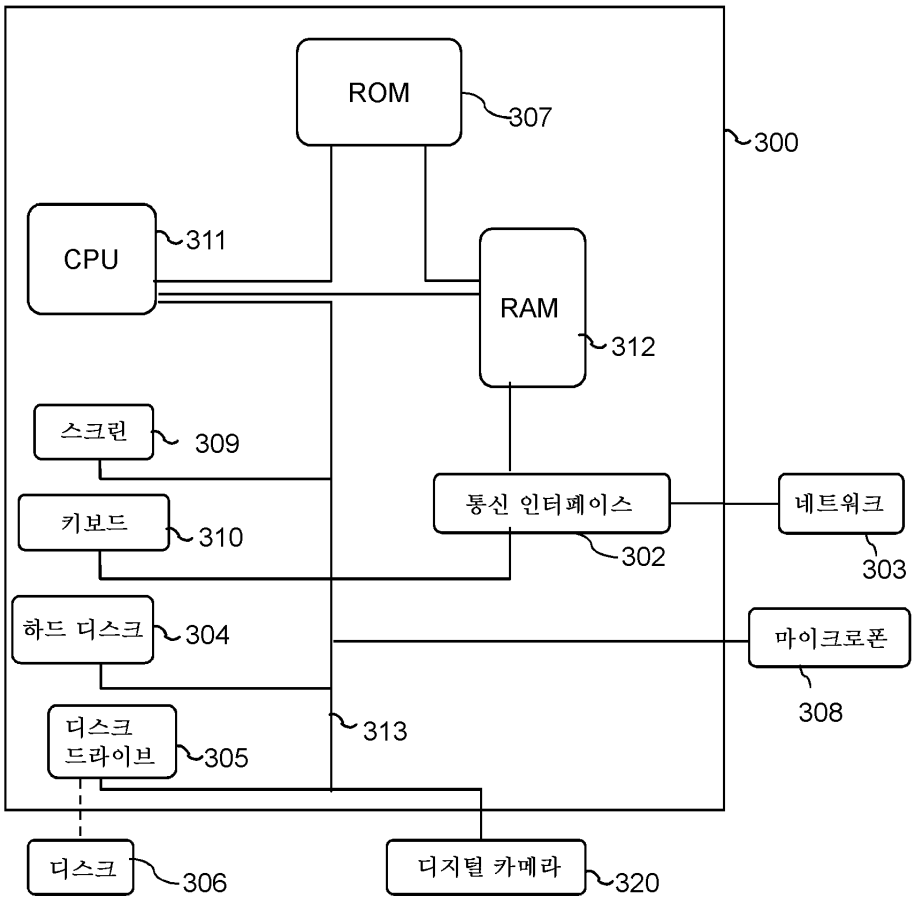
도면1


시퀀스
이미지들 {I}0-n
1
이미지
2
슬라이스 1
슬라이스 2
3
PU
2Nx2N
NxN
Nx2N
2NxN
6
TU
RQT 1 레벨
RQT 2 레벨들
RQT 3 레벨들
7
CTB
CU
5
CTB
4

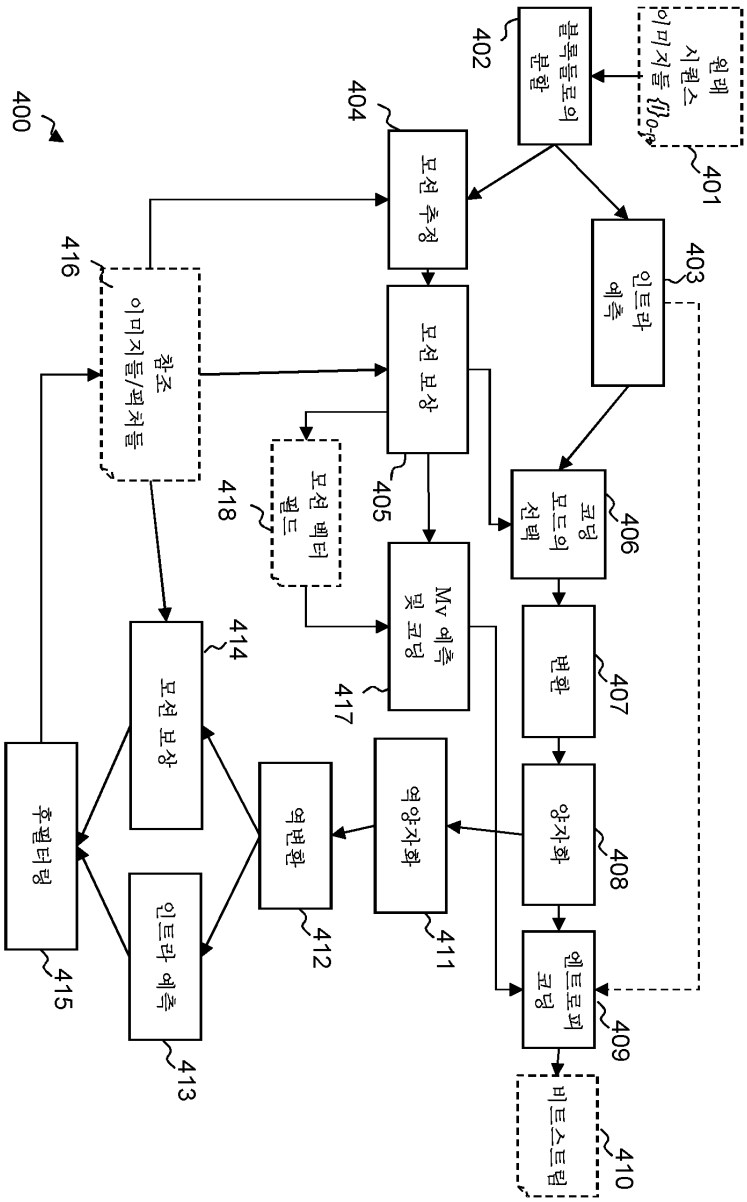
도면2


201
204
200
202
서버
데이터 스트림
네트워크
클라이언트

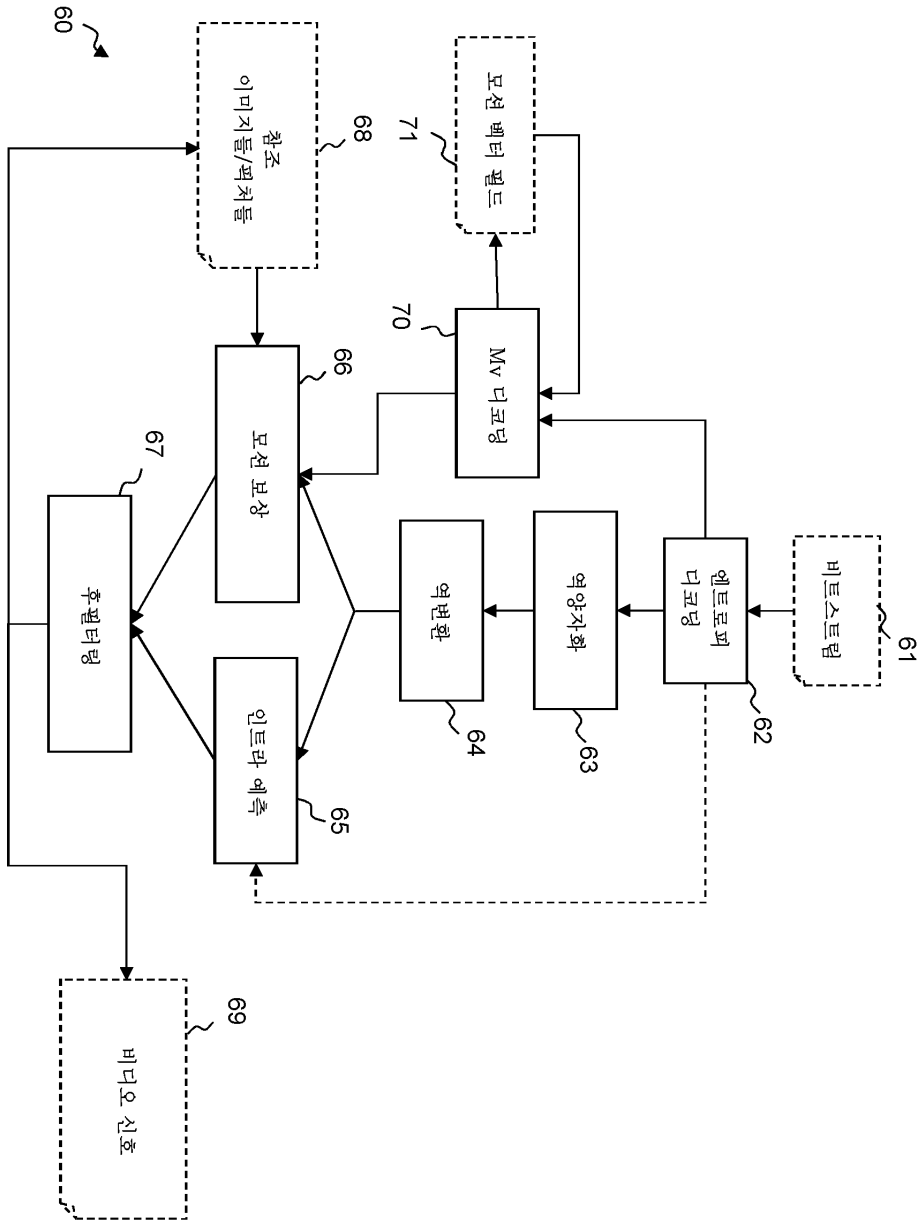
도면3


ROM
307
300
CPU
311
RAM
312
스크린
309
통신 인터페이스
네트워크
303
302
키보드
310
하드 디스크
304
마이크로폰
308
313
디스크
드라이브
305
디스크
306
디지털 카메라
320

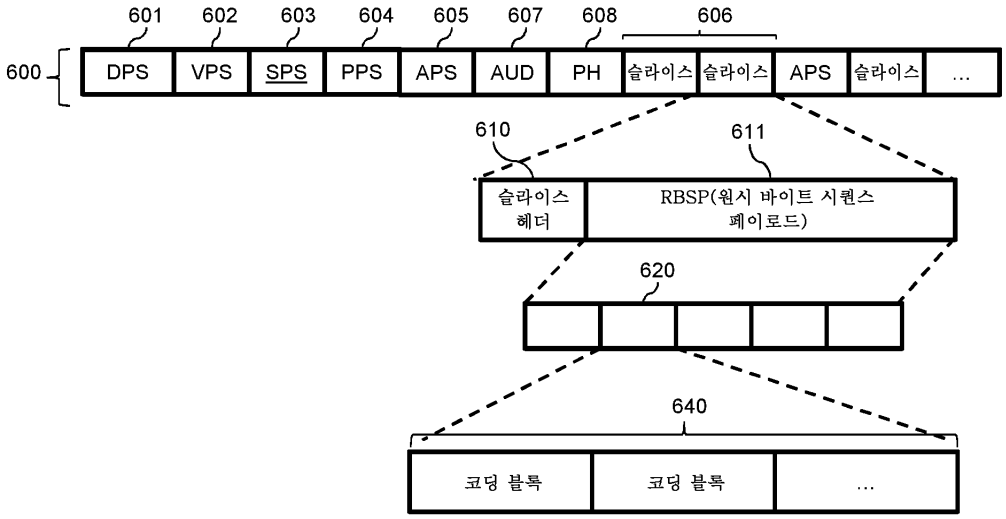
도면4


400
원래 시퀀스 이미지들 {I0,...,IN-1}
401
블록들로의 분할
402
인트라 예측
403
모션 추정
404
모션 보상
405
코딩 모드의 선택
406
변환
407
양자화
408
엔트로피 코딩
409
비트스트림
410
역양자화
411
역변환
412
인트라 예측
413
모션 보상
414
후필터링
415
참조 이미지들/픽처들
416
Mv 예측 및 코딩
417
모션 벡터 필드
418

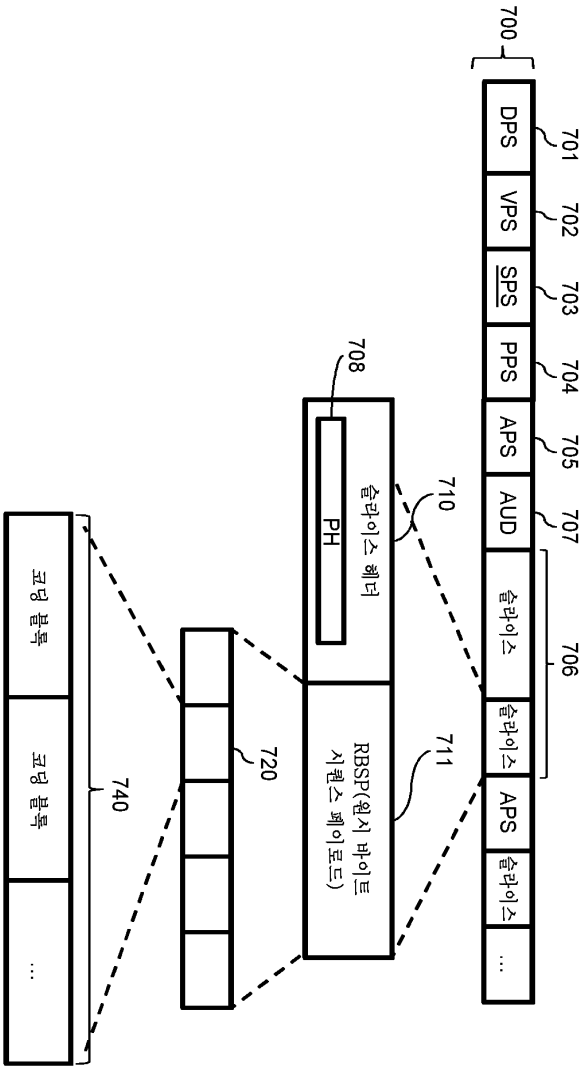
도면5


60
참조 이미지들/픽처들
68
모션 벡터 필드
71
Mv 디코딩
70
모션 보상
66
후필터링
67
역변환
64
역양자화
63
엔트로피 디코딩
62
비트스트림
61
인트라 예측
65
비디오 신호
69

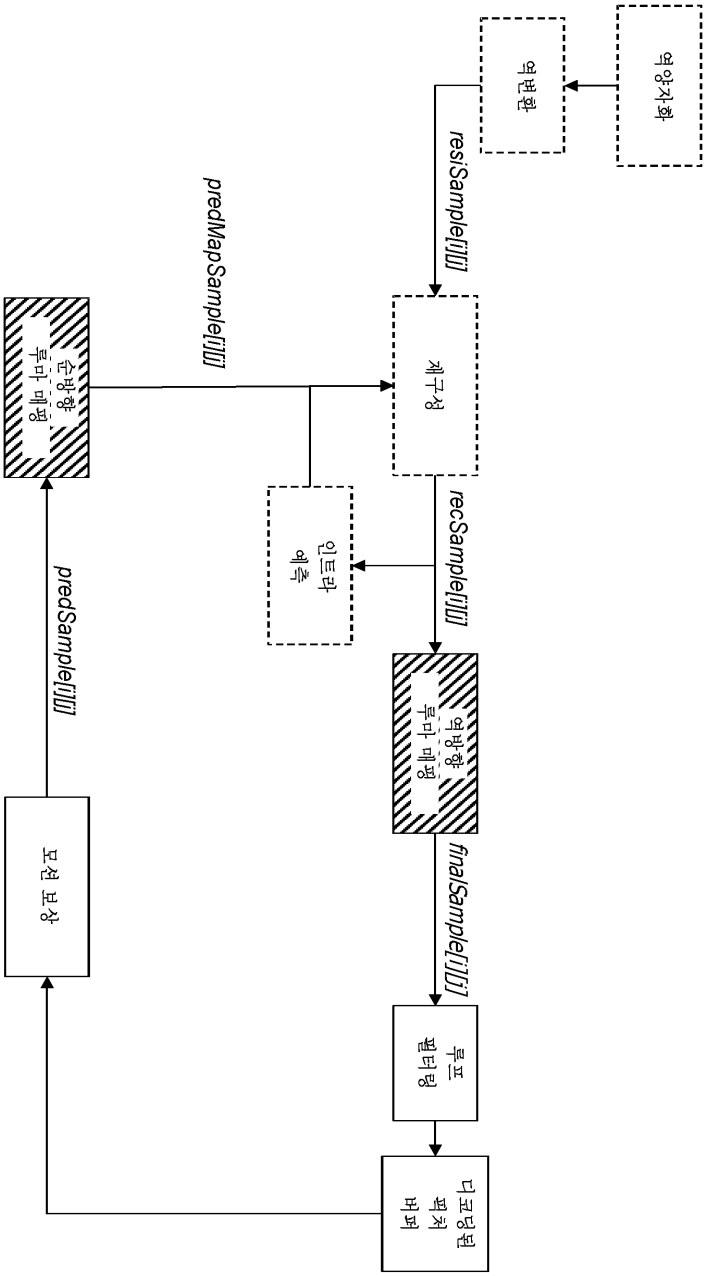
도면6


601
602
603
604
605
607
608
606
600
DPS
VPS
SPS
PPS
APS
AUD
PH
슬라이스
슬라이스
APS
슬라이스
...
610
611
슬라이스 헤더
RBSP(원시 바이트 시퀀스 페이로드)
620
640
코딩 블록
코딩 블록
...

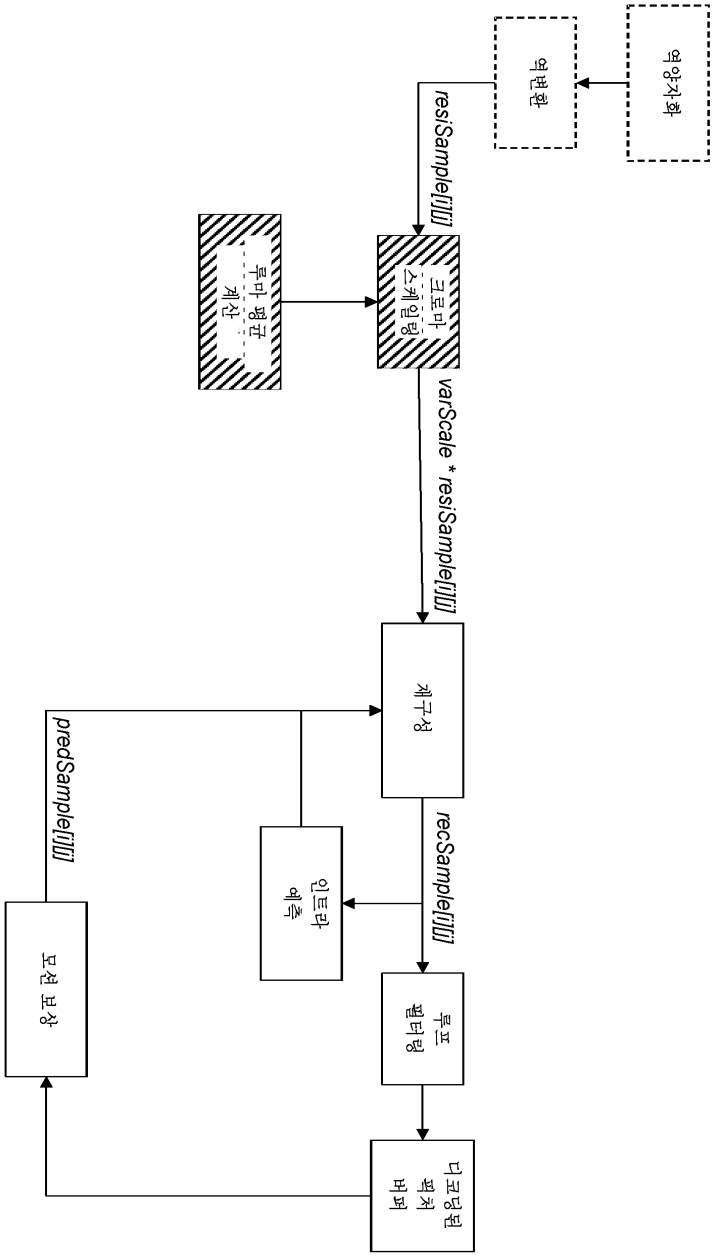
도면7


700
DPS
VPS
SPS
PPS
APS
AUD
슬라이스
슬라이스
APS
슬라이스
...
701
702
703
704
705
707
706
708
PH
슬라이스 헤더
710
711
RBSP(원시 바이트 시퀀스 페이로드)
720
740
코딩 블록
코딩 블록
...

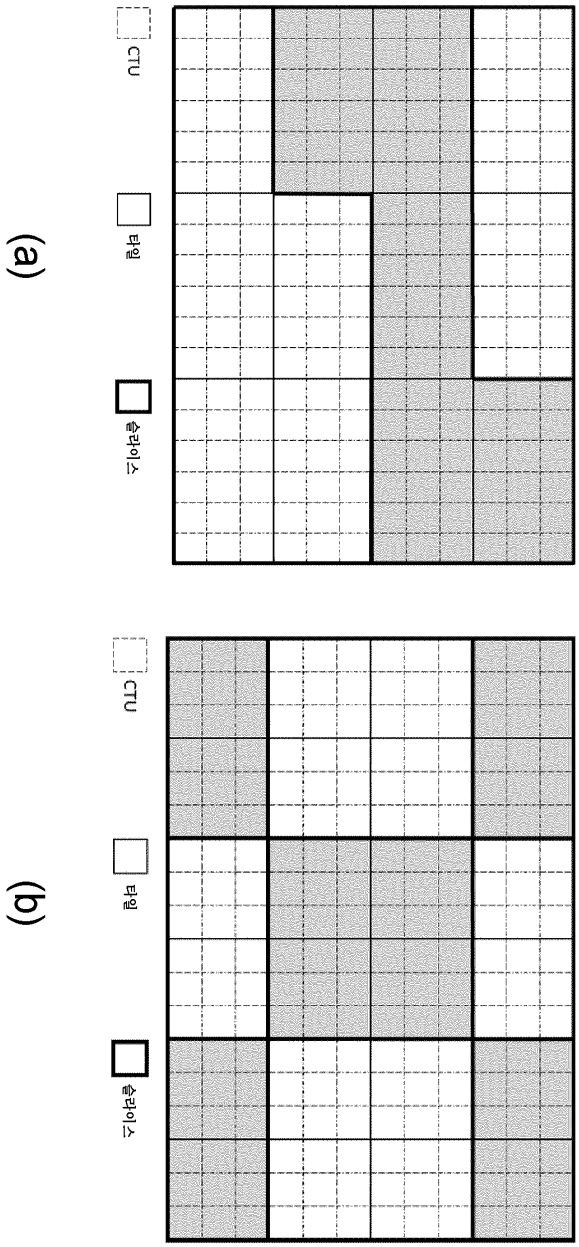
도면8


역양자화
역변환
resiSample[i][j][]
predMapSample[i][j][]
순방향 루마 매핑
재구성
인트라 예측
recSample[i][j][]
역방향 루마 매핑
finalSample[i][j][]
루프 필터링
디코딩된 픽처 버퍼
모션 보상
predSample[i][j][]

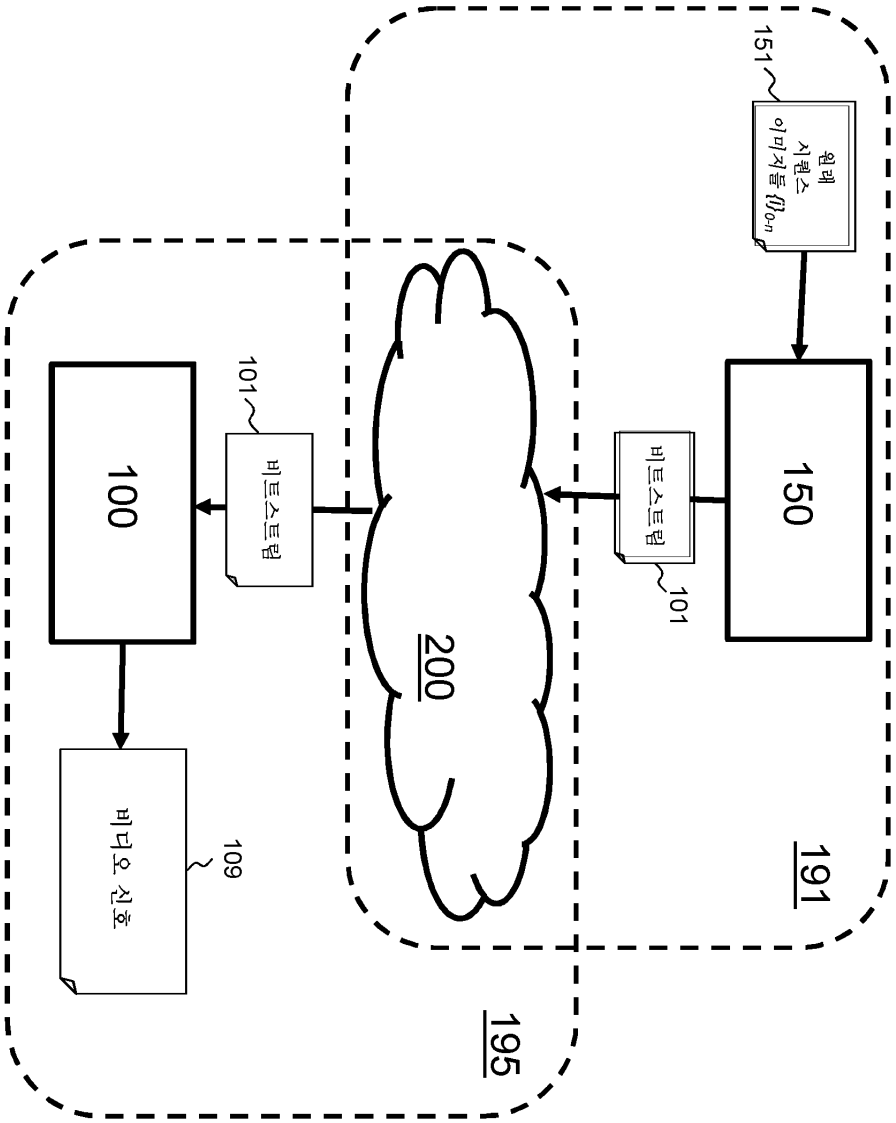
도면9


역양자화
역변환
resiSample[i][j]
루마 평균 계산
크로마 스케일링
varScale * resiSample[i][j]
재구성
recSample[i][j]
인트라 예측
루프 필터링
디코딩된 픽처 버퍼
모션 보상
predSample[i][j]

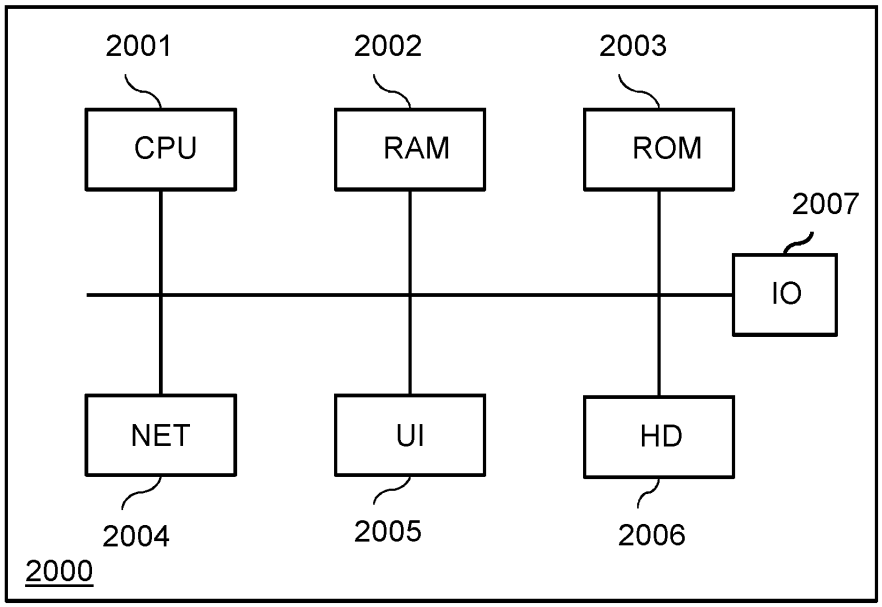
도면10



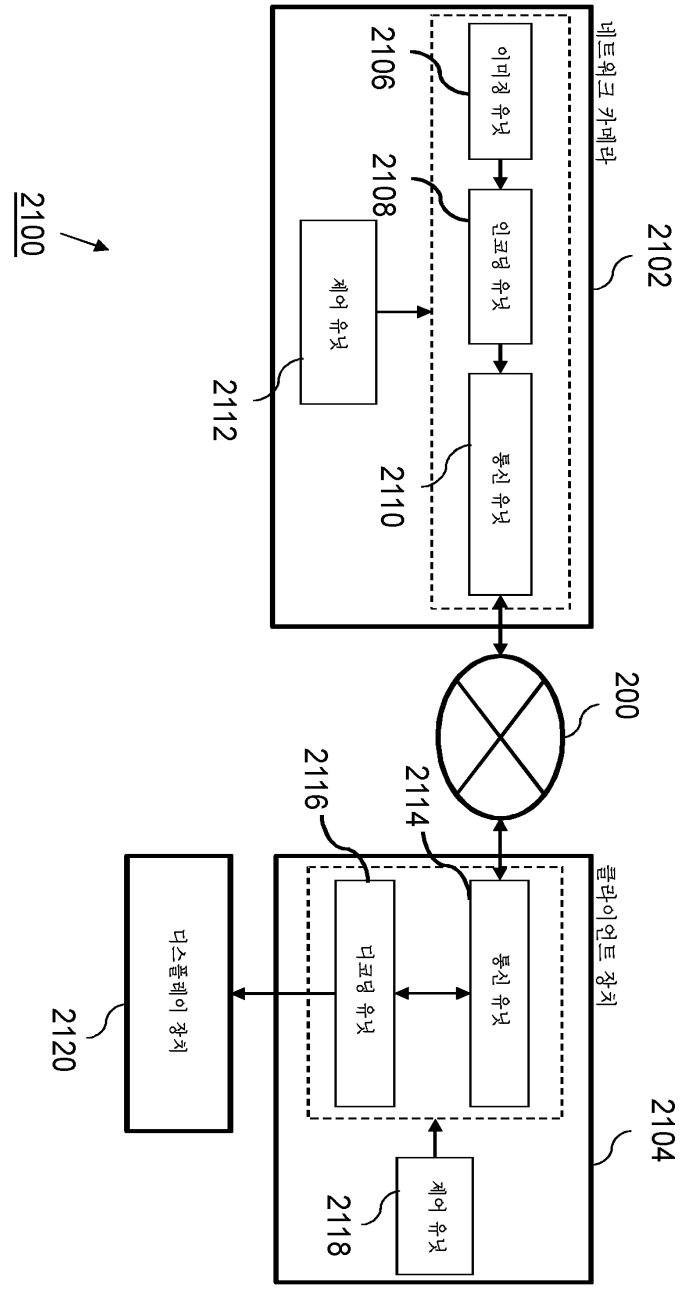
도면11


151
원래 시퀀스 이미지들 $\{I\}_{0:n}$
150
101
비트스트림
200
101
비트스트림
100
109
비디오 신호
191
195

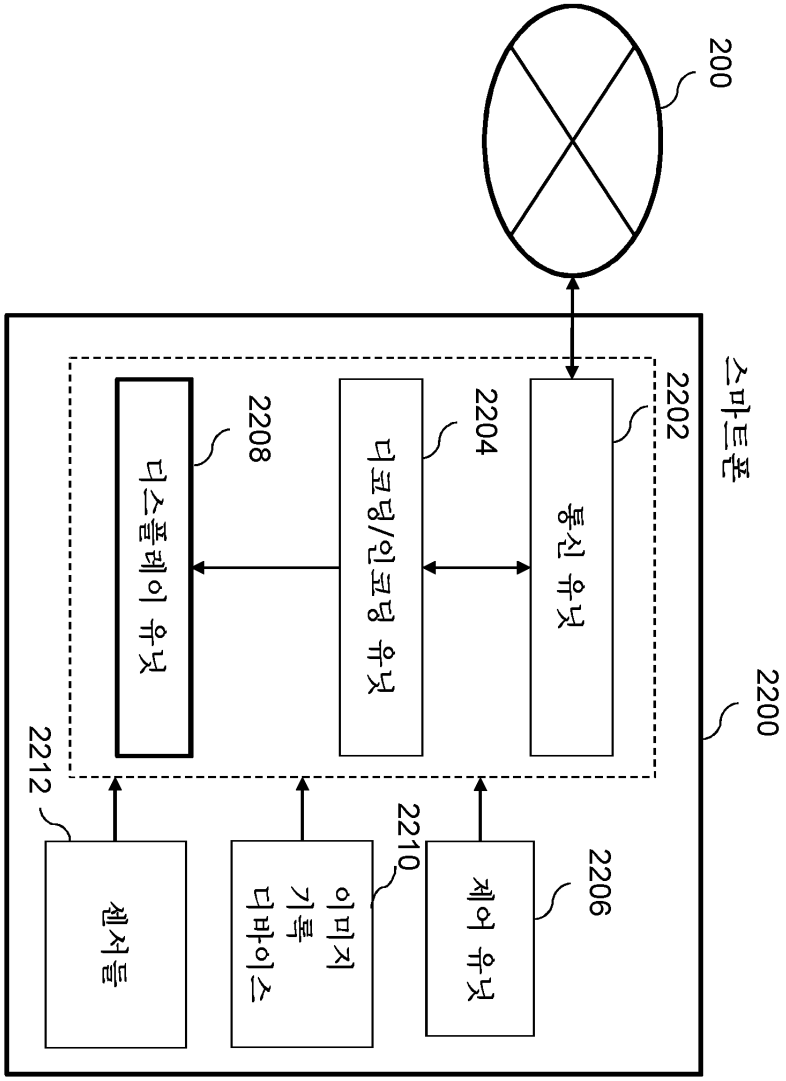
도면12


2001
2002
2003
CPU
RAM
ROM
2007
IO
NET
UI
HD
2004
2005
2006
2000

도면13


2100
네트워크 카메라
2102
이미징 유닛
2106
인코딩 유닛
2108
제어 유닛
2112
통신 유닛
2110
200
2116
2114
클라이언트 장치
2104
디코딩 유닛
통신 유닛
제어 유닛
2118
디스플레이 장치
2120

도면14


200
스마트폰
2200
2202
통신 유닛
2204
디코딩/인코딩 유닛
2208
디스플레이 유닛
2206
제어 유닛
2210
이미지 기록 디바이스
2212
센서들