



(11) 공개번호 10-2025-0038837
(43) 공개일자 2025년03월19일

(51) 국제특허분류(Int. Cl.)
H04N 19/124 (2014.01) *H04N 19/132* (2014.01)
H04N 19/182 (2014.01) *H04N 19/423* (2014.01)
H04N 19/60 (2014.01) *H04N 19/70* (2014.01)
H04N 19/82 (2014.01)

(52) CPC특허분류
H04N 19/124 (2015.01)
H04N 19/132 (2015.01)

(21) 출원번호 10-2025-7007945(분할)

(22) 출원일자(국제) 2018년06월29일
 심사청구일자 없음

(62) 원출원 특허 10-2023-7031495
 원출원일자(국제) 2018년06월29일
 심사청구일자 2023년09월14일

(85) 번역문제출일자 2025년03월10일

(86) 국제출원번호 PCT/US2018/040287

(87) 국제공개번호 WO 2019/006300
 국제공개일자 2019년01월03일

(30) 우선권주장
 62/526,577 2017년06월29일 미국(US)
 (뒷면에 계속)

(71) 출원인
 돌비 레버러토리즈 라이선싱 코오폰레이션
 미합중국, 캘리포니아 94103, 샌프란시스코, 마켓 스트리트 1275

(72) 발명자
 루, 타오란
 미국 캘리포니아 94103 샌프란시스코 마켓 스트리트 1275 돌비 레버러토리즈, 아이엔씨 내.
 푸, 광준
 미국 캘리포니아 94103 샌프란시스코 마켓 스트리트 1275 돌비 레버러토리즈, 아이엔씨 내.
 (뒷면에 계속)

(74) 대리인
 특허법인 광장리앤코

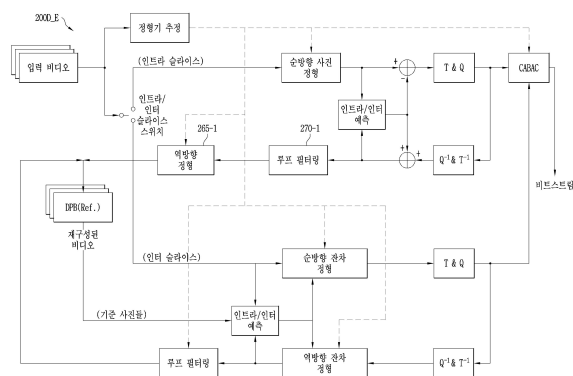
전체 청구항 수 : 총 3 항

(54) 발명의 명칭 통합된 이미지 정형 및 비디오 코딩

(57) 요약

제1 코드워드 표현의 이미지들의 시퀀스(sequence)가 제공될 때, 이미지들을 인코딩 및 디코딩하기 위한 차세대 비디오 코덱에 정형(reshaping)을 통합하기 위한 방법들, 처리들, 및 시스템들이 제시되는데, 여기서 정형은 이미지들의 부분으로 하여금, 제1 코드워드 표현을 사용하는 것보다 효율적인 압축을 허용하는 제2 코드워드 표현으로 코딩되도록 허용한다. 다음을 포함하는 다양한 구조들이 설명된다: 루프-밖 정형 구조, 루프-내 인트라 사진들 전용 정형 구조(in-loop-for intra pictures only reshaping architecture), 예측 잔차들용 루프-내 구조, 및 하이브리드 루프-내 정형 구조(hybrid in-loop reshaping architecture). 정형 매개변수들을 시그널링하기 위한 선택스(syntax) 방법, 및 정형에 관련하여 최적화되는 이미지-인코딩 방법들이 또한 제시된다.

대표도



(52) CPC특허분류

H04N 19/182 (2015.01)
H04N 19/423 (2015.01)
H04N 19/60 (2015.01)
H04N 19/70 (2015.01)
H04N 19/82 (2015.01)

(72) 발명자

인, 펙

미국 캘리포니아 94103 샌프란시스코 마켓 스트리트 1275 돌비 레버러토리즈, 아이엔씨 내.

첸, 타오

미국 캘리포니아 94103 샌프란시스코 마켓 스트리트 1275 돌비 레버러토리즈, 아이엔씨 내.

후삭, 월터 제이.

미국 캘리포니아 94103 샌프란시스코 마켓 스트리트 1275 돌비 레버러토리즈, 아이엔씨 내.

(30) 우선권주장

62/561,561	2017년09월21일	미국(US)
62/629,313	2018년02월12일	미국(US)
62/680,710	2018년06월05일	미국(US)
62/686,738	2018년06월19일	미국(US)

명세서

청구범위

청구항 1

제1 코드워드 표현(codeword representation)의 출력 이미지를 생성하기 위하여 코딩된 비트스트림을 디코딩하는 방법으로서, 상기 방법은:

제2 코드워드 표현으로 부분적으로 코딩된, 코딩된 이미지를 수신하는 단계;

상기 코딩된 이미지에 대한 정형 메타데이터를 수신하는 단계;

상기 정형 메타데이터에 기초하여, 화소들을 상기 제1 코드워드 표현에서 상기 제2 코드워드 표현으로 매핑하는 순방향 정형 함수를 생성하는 단계;

상기 정형 메타데이터에 기초하여 역방향 정형 함수를 생성하는 단계 - 상기 역방향 정형 함수는 화소들을 상기 제2 코드워드 표현에서 상기 제1 코드워드 표현으로 매핑함 - ; 및

출력 화소 영역들을 생성하기 위해 상기 순방향 정형 함수 및 상기 역방향 정형 함수에 기초하여 상기 코딩된 이미지의 코딩된 영역들을 디코딩하는 단계 - 상기 코딩된 영역을 디코딩하는 것은:

상기 제2 코드워드 표현에서 상기 코딩된 영역의 잔차 화소들을 디코딩하는 단계;

상기 제1 코드워드 표현에서 기준 화소 버퍼로부터 상기 코딩된 영역에 대한 예측 화소들을 액세스하는 단계;

상기 순방향 정형 함수를 상기 예측 화소들에 적용하여 상기 제2 코드워드 표현에서 정형된 예측 화소들을 생성하는 단계; 및

상기 정형된 예측 화소들과 상기 잔차 화소들을 가산하여 상기 제2 코드워드 표현에서 제1 재구성된 화소들을 생성하는 단계를 포함함 -

를 포함하는, 방법.

청구항 2

제1항에 있어서,

상기 역방향 정형 함수를 상기 제1 재구성된 화소들에 적용하여 상기 제1 코드워드 표현에서 출력 재구성된 화소들을 생성하는 단계; 및

상기 출력 재구성된 화소들을 상기 기준 화소 버퍼에 저장하는 단계

를 더 포함하는, 방법.

청구항 3

제1항에 있어서,

상기 정형 메타데이터는 구간적 선형 표현(a piecewise linear representation)에 기초하여 상기 순방향 정형 함수를 특성화(characterize) 하고, 상기 순방향 정형 함수의 구간적 선형 표현의 각 세그먼트에 대하여, 상기 정형 메타데이터는 델타 절대값 및 상기 델타 절대값의 부호를 포함하는, 방법.

발명의 설명

기술 분야

[0001]

관련 출원에 대한 상호-참조

[0002]

이 출원은 2018년 6월 19일자 출원된 미국 가특허출원 번호 제62/686,738호; 2018년 6월 5일자 출원된 미국 가특허출원 번호 제62/680,710호; 2018년 2월 12일자 출원된 미국 가특허출원 번호 제62/629,313호; 2017년 9월 21일자 출원된 미국 가특허출원 번호 제62/561,561호; 2017년 6월 29일자 출원된 미국 가특허출원 번호 제

62/526,577호의 우선권을 주장하며, 이들 각각은 그들 전체가 참조에 의해 여기에 통합된다.

[0003] 기술

[0004] 본 발명은 포괄적으로 이미지들 및 비디오 코딩에 관한 것이다. 보다 구체적으로, 본 발명의 실시예는 통합된 이미지 정형 및 비디오 코딩에 관한 것이다.

배경 기술

[0005] 2013년, 국제 표준화 기구(International Standardization Organization; ISO) 내의 MPEG 그룹은, 국제 전기 통신 연합(International Telecommunications Union; ITU)과 함께, HEVC(또한 H.265로도 알려짐) 비디오 코딩 표준의 초안을 발표했다. 보다 최근에는, 같은 그룹이 기존의 비디오 코딩 기술들에 비해 개선된 코딩 성능을 제공하는 차세대 코딩 표준의 개발을 지원하기 위한 증거를 위한 호출(call for evidence)를 발표했다.

[0006] 여기서 사용된 바와 같이, '비트 심도(bit depth)'란 용어는 이미지의 색 구성요소들 중 하나를 표현하는데 사용되는 화소들의 수를 나타낸다. 전형적으로, 이미지들은 색 구성요소당, 화소당 8-비트로 코딩되었지만(예를 들어, 화소당 24 비트); 현대의 구조들은 이제 10 비트, 12 비트 또는 그 이상과 같은 더욱 높은 비트 심도들을 지원할 수 있다.

[0007] 전형적인 이미지 파이프라인(image pipeline)에서, 캡처된 이미지들은, 선형 장면 빛을 비-선형 비디오 신호(예를 들어, 감마-코딩된 RGB 또는 YCbCr)로 변환시키는, 비-선형 광학-전기 변환 함수(opto-electronic transfer function; OETF)를 사용하여 양자화된다. 그 후, 수신기 상에서, 표시장치 상에 표시되기 전에, 신호는, 스크린 색 값들을 출력하도록 비디오 신호 값들을 변환하는 전기-광학 전달 함수(electro-optical transfer function; EOTF)로 처리된다. 이러한 비-선형 함수들은 ITU-R Rec. BT.709 및 BT.2020에 문서화된 전형적인 "감마" 곡선 및 SMPTE ST 2084 및 Rec. ITU-R BT.2100에 기술된 "인지 양자화(perceptual quantization; PQ)" 곡선을 포함한다.

[0008] 여기에 사용된 바와 같이, "순방향 정형(forward reshaping)"이란 용어는 디지털 이미지를 그것의 본래의 비트 심도 및 본래의 코드워드들 분포 또는 표현(예를 들어, 감마 또는 PQ 등)으로부터 같은 또는 상이한 비트 심도 및 상이한 코드워드들 분포 또는 표현의 이미지로, 샘플-대-샘플 또는 코드워드-대-코드워드 매핑하는 처리를 나타낸다. 정형은 고정된 비트 레이트(bit rate)에서 개선된 압축률 또는 개선된 이미지 품질을 허용한다. 예를 들어, 제한 없이, 10-비트 비디오 코딩 구조에서의 코딩 효율을 개선시키기 위해 정형이 10-비트 또는 12-비트 PQ-코딩된 HDR 비디오에 적용될 수 있다. 수신기에서는, 정형된 신호가 압축해제된 후, 수신기는 신호를 그것의 본래의 코드워드 분포로 복원하기 위해 "역방향 정형 함수(inverse reshaping function)"를 적용할 수 있다. 여기서 발명자들에 의해 인식될 바와 같이, 차세대 비디오 코딩 표준에 대한 개발이 시작되면서, 이미지들의 통합된 정형 및 코딩에 대한 개선된 기술들이 바래지고 있다. 이 발명의 방법들은 표준 다이내믹 범위(standard dynamic range; SDR) 및/또는 높은-다이내믹 범위(high-dynamic range; HDR) 내의 콘텐츠를 포함하는(그러나 이에 제한되는 것은 아님) 다양한 비디오 콘텐츠에 적용될 수 있다.

[0009] 이 섹션에 설명된 접근법들은 추구될 수 있었던 접근법들이지만, 반드시 이전에 구상되거나 추구되었던 접근법들은 아니다. 따라서, 달리 언급되지 않는 한, 이 섹션에 설명된 접근법들 중 어느 것도 단순히 그것이 이 섹션에 포함되어 있다는 것만으로 종래 기술로서의 자격이 주어졌다고 가정해서는 안된다. 마찬가지로, 달리 언급되지 않는 한, 하나 이상의 접근법들에 관련하여 확인된 쟁점들이 이 섹션에 기초하여 임의의 종래 기술에서 인식되었다고 가정해서는 안된다.

도면의 간단한 설명

[0010] 본 발명의 실시예는 예로서 나타내진 것으로, 본 발명을 제한하기 위한 것은 아니며, 첨부된 도면들에서 같은 도면 참조 부호들은 유사한 요소들을 의미한다.

도 1a는 비디오 전달 파이프라인에 대한 예시적인 처리를 나타낸다.

도 1b는 종래 기술에 따른 신호 정형을 사용하는 데이터 압축에 대한 예시적인 처리를 나타낸다.

도 2a는 본 발명의 실시예에 따른 규범적인 루프-밖 정형(out-of-loop reshaping)을 사용하는 인코더에 대한 예시적인 구조를 나타낸다.

도 2b는 본 발명의 실시예에 따른 규범적인 루프-밖 정형을 사용하는 디코더에 대한 예시적인 구조를 나타낸다.

도 2c는 본 발명의 실시예에 따른 규범적인 인트라-전용 루프-내 정형(intra-only in-loop reshaping)을 사용하는 인코더에 대한 예시적인 구조를 나타낸다.

도 2d는 본 발명의 실시예에 따른 규범적인 인트라-전용 루프-내 정형을 사용하는 디코더에 대한 예시적인 구조를 나타낸다.

도 2e는 본 발명의 실시예에 따른 예측 잔차들(prediction residuals)용 루프-내 정형을 사용하는 인코더에 대한 예시적인 구조를 나타낸다.

도 2f는 본 발명의 실시예에 따른 예측 잔차들용 루프-내 정형을 사용하는 디코더에 대한 예시적인 구조를 나타낸다.

도 2g는 본 발명의 실시예에 따른 하이브리드 루프-내 정형(hybrid in-loop reshaping)을 사용하는 인코더에 대한 예시적인 구조를 나타낸다.

도 2h는 본 발명의 실시예에 따른 하이브리드 루프-내 정형을 사용하는 디코더에 대한 예시적인 구조를 나타낸다.

도 3a는 본 발명의 실시예에 따른 루프-밖 정형 구조를 사용하여 비디오를 인코딩하기 위한 예시적인 처리를 나타낸다.

도 3b는 본 발명의 실시예에 따른 루프-밖 정형 구조를 사용하여 비디오를 디코딩하기 위한 예시적인 처리를 나타낸다.

도 3c는 본 발명의 실시예에 따른 루프-내 인트라-전용 정형 구조를 사용하여 비디오를 인코딩하기 위한 예시적인 처리를 나타낸다.

도 3d는 본 발명의 실시예에 따른 루프-내 인트라-전용 정형 구조를 사용하여 비디오를 디코딩하기 위한 예시적인 처리를 나타낸다.

도 3e는 본 발명의 실시예에 따른 예측 잔차들(prediction residuals)용 루프-내 정형 구조를 사용하여 비디오를 인코딩하기 위한 예시적인 처리를 나타낸다.

도 3f는 본 발명의 실시예에 따른 예측 잔차들용 루프-내 정형 구조를 사용하여 비디오를 디코딩하기 위한 예시적인 처리를 나타낸다.

도 4a는 본 발명의 실시예에 따른 3가지 정형-기반 구조들 중 임의의 하나 또는 그들의 조합을 사용하여 비디오를 인코딩하기 위한 예시적인 처리를 나타낸다.

도 4b는 본 발명의 실시예에 따른 3가지 정형-기반 구조들 중 임의의 하나 또는 그들의 조합을 사용하여 비디오를 디코딩하기 위한 예시적인 처리를 나타낸다.

도 5a 및 5b는 본 발명의 실시예에 따른 비디오 디코더 내에서의 정형 함수 재구성 처리를 나타낸다.

도 6a 및 6b는 본 발명의 실시예에 따른, 크로마 QP 오프셋 값들이 PQ- 및 HLG-코딩된 신호들에 대하여 루마 양자화 매개변수(quantization parameter; QP)에 따라 어떻게 변하는지의 예들을 나타낸다.

도 7은 본 발명의 실시예에 따른 정형 함수의 피벗-기반 표현(pivot-based representation)의 예를 나타낸다.

발명을 실시하기 위한 구체적인 내용

[0011] 이미지들을 압축하기 위한 규범적인 루프-밖 및 루프-내 통합된 신호 정형 및 코딩 기술들이 여기에 기술된다. 다음 기술에는, 설명을 위해, 본 발명의 완벽한 이해를 제공하기 위한 여러 구체적인 세부사항들이 설명되어 있다. 그러나, 본 발명이 이들 구체적인 세부사항들 없이도 실현될 수 있다는 것이 명확해질 것이다. 다른 경우에는, 본 발명을 불필요하게 가리거나, 모호하게 하거나, 애매하게 하는 것을 피하기 위해, 잘-알려진 구조들 및 장치들이 자세히 세부적으로 설명되지 않았다.

[0012] 개요

[0013] 여기에 기술된 예시적인 실시예들은 비디오용 통합된 신호 정형 및 코딩에 관한 것이다. 인코더에서, 처리기는 입력 비트 심도(N)로 표현된 제1 코드워드 표현의 입력 이미지 및 입력 코드워드 매핑(예를 들어, 감마, PQ 등)을 수신한다. 처리기는 제1 코드워드 표현보다 효율적인 압축을 허용하는 제2 코드워드 표현을 사용하여 입

력 이미지를 압축하기 위한 2개 이상의 후보 인코더 구조 중 (정형기가 인코더의 한 통합 부분(integral part)인) 한 인코더 구조를 선택하는데, 여기서 2개 이상의 후보 인코더 구조는 루프-밖 정형 구조, 루프-내 인트라 사진 전용 정형 구조(in-loop-for intra pictures only reshaping architecture), 또는 예측 잔차들용 루프-내 구조를 포함하고, 처리기는 선택된 인코더 구조에 따라 입력 이미지를 압축한다.

[0014] 또 다른 실시예에서, 제1 코드워드 표현의 출력 이미지들을 생성하기 위한 디코더는, 코딩된 이미지들 중 적어도 일부가 제2 코드워드 표현으로 압축된, 코딩된 비트스트림을 수신한다. 그것은 또한 연관된 정형 정보를 수신한다. 처리기는 입력 코딩된 비트스트림을 압축해제하기 위한 2개 이상의 후보 디코더 구조 중 상기 디코더 구조를 나타내는 시그널링(signaling)을 수신하는데, 여기서 2개 이상의 후보 디코더 구조는 루프-밖 정형 구조, 루프-내 인트라 사진 전용 정형 구조, 또는 예측 잔차들용 루프-내 구조를 포함하고, 그것은 수신된 정형 구조에 따라 출력 이미지를 생성하도록 코딩된 이미지를 압축해제한다.

[0015] 또 다른 실시예에서, 예측 잔차들용 루프-내 구조에 따라 이미지들을 압축하기 위한 인코더에서, 처리기는 제1 코드워드 표현의 입력 이미지에 액세스하고, 입력 이미지의 화소들을 제1 코드워드 표현에서 제2 코드워드 표현으로 매핑하는 순방향 정형 함수를 생성한다. 그것은 순방향 정형 함수에 기초하여, 화소들을 제2 코드워드 표현에서 제1 코드워드 표현의 화소들로 매핑하는 역방향 정형 함수를 생성한다. 그 후, 입력 이미지 내의 입력 화소 영역에 대하여: 그것은

[0016] 기준 프레임 버퍼 또는 미리 코딩된 공간적인 주변부들 내의 화소 데이터에 기초하여 적어도 하나의 예측 영역을 컴퓨팅(computing)하고;

[0017] 입력 화소 영역, 예측 영역, 및 순방향 정형 함수에 기초하여 정형된 잔차 영역을 생성하고;

[0018] 정형된 잔차 영역에 기초하여 코딩된(변형 및 양자화된) 잔차 영역을 생성하고;

[0019] 코딩된 잔차 영역에 기초하여 디코딩된(역방향 양자화 및 변형된) 잔차 영역을 생성하고;

[0020] 디코딩된 잔차 영역, 예측 영역, 순방향 정형 함수, 및 역방향 정형 함수에 기초하여 재구성된 화소 영역을 생성하고;

[0021] 재구성된 화소 영역에 기초하여 기준 프레임 버퍼 상에 저장될 기준 화소 영역을 생성한다.

[0022] 또다른 실시예에서는, 예측 잔차들용 루프-내 구조에 따라 제1 코드워드 표현의 출력 이미지들을 생성하기 위한 디코더에서, 처리기는 제2 코드워드 표현으로 부분적으로 코딩된, 코딩된 비트스트림을 수신한다. 그것은 또한 연관된 정형 정보를 수신한다. 처리기는 정형 정보에 기초하여 순방향 정형 함수 및 역방향 정형 함수를 생성하는데, 여기서, 순방향 정형 함수는 화소들을 제1 코드워드 표현에서 제2 코드워드 표현으로 매핑하고, 역방향 정형 함수는 화소들을 제2 코드워드 표현에서 제1 코드워드 표현으로 매핑한다. 코딩된 이미지의 영역에 대해, 처리기는:

[0023] 코딩된 이미지에 기초하여 디코딩된 정형된 잔차 영역을 생성하고;

[0024] 기준 화소 버퍼 또는 미리 디코딩된 공간적인 주변부들 내의 화소들에 기초하여 예측 영역을 생성하고;

[0025] 디코딩된 정형된 잔차 영역, 예측 영역, 순방향 정형 함수, 및 역방향 정형 함수에 기초하여 재구성된 화소 영역을 생성하고;

[0026] 재구성된 화소 영역에 기초하여 출력 화소 영역을 생성하고;

[0027] 기준 화소 버퍼에 출력 화소 영역을 저장한다.

[0028] 예시적인 비디오 전달 처리 파이프라인

[0029] 도 1a는 비디오 캡처(capture)에서 비디오 콘텐츠 표시까지의 다양한 단계들을 보여주는 종래의 비디오 전달 파이프라인(100)의 예시적인 처리를 나타낸다. 비디오 프레임들(102)의 시퀀스(sequence)가 이미지 생성 블록(105)을 사용하여 캡처 또는 생성된다. 비디오 프레임(102)은 비디오 데이터(107)를 제공하도록 (예를 들어, 디지털 카메라에 의해) 디지털적으로 캡처되거나 (예를 들어, 컴퓨터 애니메이션을 사용하여) 컴퓨터에 의해 생성될 수 있다. 대안적으로, 비디오 프레임들(102)은 필름 카메라에 의해 필름 상에 캡처될 수도 있다. 필름은 비디오 데이터(107)를 제공하도록 디지털 포맷으로 변환된다. 제작 단계(110)에서, 비디오 데이터(107)는 비디오 제작 스트림(112)을 제공하도록 편집된다.

[0030] 제작 스트림(112)의 비디오 데이터는 그 후 제작-후 편집을 위한 블록(115)에서 처리기에 제공된다. 이미지 폼

질을 향상시키거나 비디오 크리에이터(creator)의 창의적인 의도에 따라 이미지의 특정한 표현을 달성하기 위해, 제작-후 편집 블록(115)은 이미지의 특정한 영역의 색 또는 밝기를 조정 또는 수정하는 것을 포함할 수 있다. 이것은 종종 "색 조정(color timing)" 또는 "색 보정(color grading)"이라고 불린다. 배포할 최종 제작 버전(117)을 생성하기 위해, 다른 편집(예를 들어, 장면 선택 및 시퀀싱, 이미지 크로핑(image cropping), 컴퓨터-생성된 시각적인 특수 효과의 추가 등)이 블록(115)에서 수행될 수 있다. 제작-후 편집(115) 동안, 비디오 이미지들은 기준 표시장치(125) 상에 보여진다.

[0031] 제작-후(115)에 이어서, 최종 제작 비디오 데이터(117)는 텔레비전 세트들, 셋-톱 박스(set-top box)들, 영화 극장들 등과 같은 디코딩 및 재생 장치들에 다운로드를 전달하기 위해 인코딩 블록(120)에 전달될 수 있다. 몇몇 실시예에서, 코딩된 비트 스트림(122)을 생성하기 위해, 코딩 블록(120)은 ATSC, DVB, DVD, 블루-레이(Blu-Ray) 및 다른 전달 포맷들에 의해 정의된 것과 같은 오디오 및 비디오 인코더들을 포함할 수 있다. 수신기에서, 신호(117)와 동일한 또는 유사한 근사치를 표현하는 디코딩된 신호(132)를 생성하기 위해, 코딩된 비트 스트림(122)은 디코딩 유닛(130)에 의해 디코딩된다. 수신기는 기준 표시장치(125)와 완전히 다른 특징들을 가질 수 있는 목표 표시장치(140)에 부착될 수 있다. 그 경우, 표시-매핑 신호(137)를 생성함으로써 디코딩된 신호(132)의 다이내믹 영역을 목표 표시장치(140)의 특징들로 매핑하기 위해, 표시 관리 블록(135)이 사용될 수 있다.

[0032] 신호 정형

[0033] 도 1b는 참고 문헌 [1]의 종래 기술에 따른 신호 정형을 위한 예시적인 처리를 나타낸다. 입력 프레임들(117)이 주어지면, 순방향 정형 블록(150)은 입력 및 코딩 제약사항들을 분석하고, 입력 프레임들(117)을 재-양자화된 출력 프레임들(152)로 매핑하는 코드워드 매핑 함수들을 생성한다. 예를 들어, 입력(117)은 어떤 전자-광학 전달 함수(EOTF)(예를 들어, 감마)에 따라 인코딩될 수 있다. 몇몇 실시예에서, 정형 처리에 대한 정보는 메타데이터를 사용하여 (디코더들과 같은) 다운로드 장치들에 전달될 수 있다. 여기에 사용된 바와 같이, "메타데이터"란 용어는 코딩된 비트스트림의 부분으로써 전송되고, 디코더가 디코딩된 이미지를 렌더링(rendering)하도록 돕는 임의의 보조 정보에 관한 것이다. 이러한 메타데이터는 여기에 설명된 것들과 같은 색 공간 또는 영역 정보, 기준 표시 매개변수들, 및 보조 신호 매개변수들을 포함할 수 있지만, 이에 한정되는 것은 아니다.

[0034] 코딩(120) 및 디코딩(130)에 이어서, 디코딩된 프레임들(132)은 반대방향(backward)(또는 역방향(inverse)) 정형 함수(160)에 의해 처리될 수 있는데, 여기서 반대방향(또는 역방향) 정형 함수는 전송된 표시 관리 처리(135)와 같은 추가적인 다운로드 처리를 위해, 재-양자화된 프레임들(132)을 본래의 EOTF 도메인(예를 들어, 감마)으로 반대로 변환시킨다. 몇몇의 실시예에서, 반대방향 정형 함수(160)는 디코더(130) 내의 역-양자화기와, 예를 들어, AVC 또는 HEVC 비디오 디코더 내의 역-양자화기의 부분으로서, 통합될 수 있다.

[0035] 여기에 사용된 바와 같이, "정형기"란 용어는 디지털 이미지들을 코딩 및/또는 디코딩할 때 사용될 수 있는 순방향 또는 역방향 정형 함수를 나타낼 수 있다. 정형 함수들의 예는 참고 문헌 [1] 및 [2]에 설명되어 있다. 본 발명의 목적을 위하여, 당업자는 입력 비디오 신호의 특징들과, 인코딩 및 디코딩 구조들의 사용가능한 비트-심도에 따라 적합한 순방향 및 역방향 정형 함수들을 도출해낼 수 있다고 가정된다.

[0036] 참고 문헌 [1]에는, 높은 다이내믹 범위 비디오 코딩을 위한 루프-내 블록-기반 이미지 정형 방법이 제안되었다. 그 디자인은 코딩 루프 내에서의 블록-기반 정형을 허용하지만, 복잡도가 증가한다. 구체적으로, 그 디자인은 두 세트의 디코딩된-이미지 버퍼를 유지할 것을 필요로 하는데: 하나의 세트는 역방향-정형된(또는 미-정형된) 디코딩된 사진들을 위한 것으로, 이것은 정형되지 않은 예측 및 표시장치에의 출력 모두를 위해 사용될 수 있고, 또다른 세트는 순방향-정형된 디코딩된 사진들을 위한 것으로, 이것은 오직 정형된 예측을 위해 서만 사용된다. 순방향-정형된 디코딩된 사진이 그때 그때 컴퓨팅될 수 있지만, 복잡도가 특히 인터-예측(inter-prediction)(서브-화소 보간법을 갖는 움직임 보상)에 대해서 매우 높다. 일반적으로, 표시-사진-버퍼(display-picture-buffer; DPB) 관리가 복잡하고 매우 세심한 주의를 필요로 하므로, 발명자들에 의해 인식될 바와 같이, 비디오 코딩을 위한 간단한 방법이 요구된다.

[0037] 여기에 나타난 정형-기반 코덱 구조의 실시예들은 다음과 같이 나뉠 수 있다: 외장형 루프-밖 정형기를 갖는 구조, 루프-내 인트라 전용 정형기를 갖는 구조, 및 짧게 '루프-내 잔차 정형기'로도 불리는 예측 잔차들용 루프-내 정형기를 갖는 구조. 비디오 인코더 또는 디코더는 이들 구조들 중 임의의 하나 또는 그들의 조합을 지원할 수 있다. 이들 구조들 각각은 독립적으로 또는 임의의 다른 것과 조합하여 적용될 수 있다. 각각의 구조는 휘도 구성요소나, 크로마 구성요소나, 루마 및 하나 이상의 크로마 구성요소들의 조합에 적용될 수 있다.

- [0038] 이들 3가지 구조들 이외에, 추가적인 실시예들은, 정형에 관련된 메타데이터의 효율적인 시그널링 방법 및 정형이 적용될 때 코딩 효율을 개선시키기 위한 몇몇의 인코더-기반 최적화 도구들을 설명한다.
- [0039] 규범적인 루프-밖 정형기
- [0040] 도 2a 및 2b는 "규범적인" 루프-밖 정형기를 갖는 비디오 인코더(200A_E) 및 대응하는 비디오 디코더(200A_D)에 대한 구조들을 나타낸다. "규범적인"이란 용어는, 정형이 전-처리 단계로 고려되어, AVC, HEVC 등과 같은 코딩 표준의 규범적인 설명 밖에 있는 이전 디자인과 달리, 이 실시예에서, 순방향 및 역방향 정형이 규범적인 요구 사항들의 부분인 것을 의미한다. 표준에 따른 비트스트림 적합성(bitstream conformance)이 디코딩(130) 이후에 시험되는 도 1b의 구조와 달리, 도 2b에서는, 적합성이 역방향 정형 블록(265) 이후에 시험된다(예를 들어, 도 1b 내의 출력(162)에서).
- [0041] 인코더(200A_E)에는, 2개의 새로운 블록이 전형적인 블록-기반 인코더(예를 들어, HEVC)에 추가되는데; 순방향 정형 함수를 추정하기 위한 블록(205), 및 입력 비디오(117)의 하나 이상의 색 구성요소에 순방향 정형을 적용하는 순방향 사진 정형 블록(210)이다. 몇몇 실시예에서, 이들 2개의 동작(operation)은 단일의 이미지 정형 블록의 부분으로 수행될 수 있다. 디코더 내에서 역방향 정형 함수를 결정하는 것에 관련된 매개변수들(207)은 비디오 인코더(예를 들어, CABAC(220))의 무손실 인코더 블록에 전달될 수 있고, 이에 따라 그들은 코딩된 비트스트림(122)에 포함될 수 있다. 인트라 또는 인터-예측(225), 변형 및 양자화(T&Q), 역방향 변형 및 양자화(Q^{-1} & T^{-1}), 및 루프 필터링에 관련된 모든 동작은 DPB(215)에 저장된 정형된 사진들을 사용하여 수행된다.
- [0042] 디코더(200A_D)에는, 2개의 새로운 규범적인 블록들이 전형적인 블록-기반 디코더에 추가되는데; 인코딩된 정형 함수 매개변수들(207)에 기초하여 역방향 정형 함수를 재구성하기 위한 블록(250), 및 디코딩된 비디오 신호(162)를 생성하기 위해 디코딩된 데이터(262)에 역방향 정형 함수를 적용하기 위한 블록(265)이다. 몇몇의 실시예들에서, 블록(250 및 265)에 관련된 동작은 단일의 처리 블록에 조합될 수 있다.
- [0043] 도 3a는 본 발명의 실시예에 따라 루프-밖 정형 구조(200A_E)를 사용하여 비디오를 인코딩하기 위한 예시적인 처리(300A_E)를 나타낸다. 사용가능한 정형이 없으면(경로(305)), 인코딩은 종래 기술 인코더들에서 알려진 바와 같이 처리된다(예를 들어, HEVC). 정형이 사용가능하면(경로(310)), 인코더는 미리-결정된 (디폴트) 정형 함수를 적용할지(315) 또는 (예를 들어, 참고 문헌 [1]-[3]에 설명된 바와 같이) 사진 분석(320)에 기초하여 새로운 정형 함수를 적응적으로 결정할지(325)에 대한 선택권을 가질 수 있다. 순방향 정형(330)에 이어서, 나머지 인코딩은 전형적인 코딩 파이프라인(335)을 따른다. 적응형 정형(312)이 사용되면, 역방향 정형 함수에 관련된 메타데이터는 "정형기 인코딩" 단계(327)의 부분으로서 생성된다.
- [0044] 도 3b는 본 발명의 실시예에 따라 루프-밖 정형 구조(200A_D)를 사용하여 비디오를 디코딩하기 위한 예시적인 처리(300A_D)를 나타낸다. 사용가능한 정형이 없으면(경로(355)), 사진이 디코딩된(350) 후, 전형적인 디코딩 파이프라인에서처럼 출력 프레임들이 생성된다(390). 정형이 사용가능하면(경로(360)), 단계(370)에서, 디코더는 미리-결정된 (디폴트) 정형 함수를 적용할지(375) 또는 수신된 매개변수들(예를 들어, 207)에 기초하여 역방향 정형 함수(380)를 적응적으로 결정할지를 결정한다. 역방향 정형(385)에 이어서, 나머지 디코딩은 전형적인 디코딩 파이프라인을 따른다.
- [0045] 규범적인 루프-내 인트라-전용 정형기
- [0046] 도 2c는 본 발명의 실시예에 따라 규범적인 인트라-전용 루프-내 정형을 사용하는 인코더(200B_E)에 대한 예시적인 구조를 나타낸다. 그 디자인은 참고 문헌 [1]에 제안된 디자인과 매우 유사하지만; 복잡도를 줄이기 위해, 특히 그것이 DPB 메모리(215 및 260)의 사용에 관련된다는 점에서, 오직 인트라 사진들만이 이 구조를 사용하여 인코딩된다.
- [0047] 루프-밖 정형(200A_E)에 비해, 인코더(200B_E) 내에서의 주요 차이점은 DPB(215)가 정형된 사진들 대신 역방향-정형된 사진들을 저장한다는 점이다. 즉, 디코딩된 인트라 사진은 DPB에 저장되기 전에 (역방향 정형 유닛(265)에 의해) 역방향 정형될 필요가 있다. 이 접근법 뒤의 이론은, 인트라 사진들이 정형으로 코딩되었으면, 인터 사진들이 정형되지 않고 코딩되더라도, 인트라 사진들을 코딩하는 것의 개선된 성능이 인터 사진들의 코딩도 (암묵적으로) 개선시키도록 전파될 것이라 기대하는 것이다. 이 방식에서는, 인터 사진들에 대한 루프-내 정형의 복잡도를 처리하지 않으면서 정형의 이점을 취할 수 있다. 역방향 정형(265)이 내부 루프의 부분이므로, 그것은 루프-내 필터(270) 이전에 구현될 수 있다. 루프-내 필터 전에 역방향 정형을 추가하는 것의 이점은, 이 경우에, 루프-내 필터의 디자인이 순방향-정형된 사진들 대신 본래 사진들의 특징들에 기초하여 최적화될 수 있다는 것이

다.

[0048] 도 2d는 본 발명의 실시예에 따라 규범적인 인트라-전용 루프-내 정형을 사용하는 디코더(200B_D)에 대한 예시적인 구조를 나타낸다. 도 2d에 나타난 바와 같이, 역방향 정형 함수(250)를 결정하는 것 및 역방향 정형(265)을 적용하는 것은 이제 루프-내 필터링(270) 이전에 수행된다.

[0049] 도 3c는 본 발명의 실시예에 따라 루프-내 인트라-전용 정형 구조를 사용하여 비디오를 인코딩하기 위한 예시적인 처리(300B_E)를 나타낸다. 나타난 바와 같이, 도 3c의 동작 흐름은 도 3a의 동작 흐름과 많은 요소들을 공유한다. 이제, 기본적으로, 인트라-코딩에 정형이 적용되지 않는다. 인트라-코딩된 사진들에 대하여, 정형이 사용가능하면, 인코더는 디폴트 정형 곡선을 사용할지 또는 적응형 정형(312)을 적용할지에 대한 선택권을 또다시 갖게 된다. 사진이 정형되었으면, 역방향 정형(385)이 처리의 부분이 되고, 단계(327)에서 연관된 매개변수들이 인코딩된다. 대응하는 디코딩 처리(300B_D)가 도 3d에 나타나있다.

[0050] 도 3d에 나타난 바와 같이, 정형 관련 동작들은 단지 수신된 인트라 사진들에 대해서만 및 인트라 정형이 인코더에 적용됐을 때만 사용가능하다.

[0051] 예측 잔차들용 루프-내 정형기

[0052] 코딩에서, '잔차'란 용어는 샘플 또는 데이터 요소의 예측과 그것의 본래의 또는 디코딩된 값 간의 차이를 나타낸다. 예를 들어, *Orig_sample*이라고 언급되는 입력 비디오(117)로부터의 본래의 샘플이 주어지면, 인트라 또는 인트라 예측(225)은 *Pred_sample*이라고 언급되는 대응하는 예측 샘플(227)을 생성할 수 있다. 정형이 없으면, 미정형된 잔차(*Res_u*)가 다음과 같이 정의될 수 있다.

[0053] [수학식 1]

$$Res_u = Orig_sample - Pred_sample.$$

[0054]

[0055] 몇몇 실시예에서는, 정형을 잔차 도메인(residual domain)에 적용하는 것이 유익할 수 있다. 도 2e는 본 발명의 실시예에 따라 예측 잔차들용 루프-내 전형을 사용하는 인코더(200C_E)에 대한 예시적인 구조를 나타낸다. *Fwd()*는 순방향 정형 함수를 나타내고, *Inv()*는 대응하는 역방향 정형 함수를 나타내게 하자. 실시예에서, 정형된 잔차(232)는 다음과 같이 정의될 수 있다.

[0056] [수학식 2]

$$Res_r = Fwd(Orig_sample) - Fwd(Pred_sample)$$

[0057]

[0058] 따라서, 역방향 정형기(265)의 출력(267)에서, *Reco_sample*(267)로 나타난 재구성된 샘플은 다음과 같이 표현될 수 있는데

[0059] [수학식 3]

$$Reco_sample = Inv(Res_d + Fwd(Pred_sample))$$

[0060]

[0061] 여기서, *Res_d*는 200C_E 내의 루프-내 코딩 및 디코딩 이후의 잔차(234)(*Res_r*의 근사치)를 나타낸다.

[0062] 잔차들에 정형이 적용되지만, 실제 입력 비디오 화소들은 정형되지 않는다. 도 2f는 대응하는 디코더(200C_D)를 나타낸다. 도 2f에 나타난 바와 같이, 그리고 수학식 3에 기초하여, 디코더는 순방향 및 역방향 정형 함수 모두에 액세스를 필요로하고, 이들은 수신된 메타데이터(207) 및 "정형기 디코딩" 블록(250)을 사용하여 추출될 수 있다.

[0063] 실시예에서, 복잡도를 낮추기 위해, 수학식 2 및 3이 간단해질 수 있다. 예를 들어, 순방향 정형 함수가 구간적 선형 함수(piecewise linear function)에 의해 근사화될 수 있고, *Pred_sample*과 *Orig_sample* 간의 절대차가 비교적 작다고 가정하면, 수학식 2는 다음과 같이 근사화될 수 있는데

[0064] [수학식 4]

$$Res_r = a(Pred_sample) * (Orig_sample - Pred_sample).$$

[0065]

[0066] 여기서 *a(Pred_sample)*은 *Pred_sample*의 값에 기초한 스케일링 인자(scaling factor)를 나타낸다. 수학식 3 및

4로부터, 수학식 3은 다음과 같이 근사화될 수 있다.

[수학식 5]

$$Reco_sample = Pred_sample + (1/a(Pred_sample)) * Res_r,$$

따라서, 실시예에서, 디코더에는 구간적 선형 모델에 대한 스케일링 인자 $a(Pred_sample)$ 만 전달할 필요가 있다.

도 3e 및 도 3f는 예측 잔차들의 루프-내 정형을 사용하여 비디오를 인코딩(300C_E) 및 디코딩(300C_D)하기 위한 예시적인 처리 흐름을 나타낸다. 처리는 도 3a 및 3b에 설명된 것과 상당히 유사하므로, 자명하다.

표 1은 3가지 제안된 구조의 핵심 특징들을 요약한 것이다.

【표 1: 고려된 정형 구조들에 대한 핵심 특징들】

구조	루프-밖	루프-내 인트라 전용	루프-내 잔차
DPB 저장소	정형된 사진	인트라 모드: 역방향-정형된 사진내부 모드: 정형 없음	미-정형된 사진
인트라 예측이 수행되는 대상	정형된 사진	정형된 사진	미-정형된 사진
인타 예측(움직임 추정)이 수행되는 대상	정형된 사진	미-정형된 사진	미-정형된 사진
여분의 사진 버퍼가 필요한가	예(DPB 내의 정형된 사진 및 출력할 미-정형된 사진을 보유하기 위한 버퍼가 필요)	아니오(그때 그때 사진 샘플을 대체)	아니오(그때 그때 잔차 샘플을 대체)
적응형 정형 추정 위치/빈도	제한 없음(인트라 전용이거나, 장면-기반하거나, 구성가능할 수 있음)	인트라-사진 상에서만	제한 없음(인트라 전용이거나, 장면-기반하거나, 구성가능할 수 있음)
샘플 수정(정형)의 복잡도	모든 사진들을 처리함	인트라-사진만 처리함(최저 복잡도)	인트라 또는 인타인지에 상관없이 잔차를 처리함
루프 필터 상호작용	기준으로서 정형된 사진을 사용하여 최적화함	기준으로서 본래의 사진을 사용하여 최적화함	기준으로서 본래의 사진을 사용하여 최적화함
적응형, 블록/영역-수준 정형이 가능한가	아니오	아니오	예
기타 양상들	기준 프레임들이 상이한 정형 함수들을 가지면, 인타 예측 성능이 악화됨		(스스로 상이한 정형기를 가질 수 있는) 기준 프레임에 대해 잔차를 처리하기 위해, 인타 예측이 현재 사진에 대해 정형기를 사용할 수 있음
	디코더 측에서 단지 역방향 정형 함수만을 필요로 함	디코더 측에서 단지 역방향 정형 함수만을 필요로 함	디코더는 순방향 및 역방향 정형 함수 모두를 필요로 함

도 4a 및 4b는 3가지 제안된 구조의 조합을 사용하여 인코딩 및 디코딩하기 위한 예시적인 인코딩 및 디코딩 처리 흐름을 나타낸다. 도 4a에 나타난 바와 같이, 정형이 사용가능하지 않으면, 입력 비디오는 어떠한 정형도 사용하지 않고 (예를 들어, HEVC 등과 같은) 알려진 비디오 코딩 기술에 따라 인코딩된다. 이와 달리, 인코더는 목표 수신기의 능력 및/또는 입력 특징들에 따라, 3가지 주요하게 제안된 방법들 중 임의의 하나를 선택할 수 있다. 예를 들어, 실시예에서, 인코더는 장면 수준에서 이들 방법들 간에 스위칭할 수 있는데, 여기서, '장면'은 유사한 휘도 특징들을 갖는 연속된 프레임들의 시퀀스를 나타낸다. 또다른 실시예에서는, 높은 수준의 매개 변수들이 시퀀스 매개변수 세트(Sequence Parameter Set; SPS) 수준에서 정의된다.

- [0074] 도 4b에 나타낸 바와 같이, 정형 정보의 수신된 시그널링에 따라, 디코더는 인입하는 코딩된 비트스트림을 디코딩하기 위해 대응하는 디코딩 처리들 중 임의의 것을 호출할 수 있다.
- [0075] 하이브리드 루프-내 정형
- [0076] 도 2g는 하이브리드 루프-내 정형 구조를 사용하는 인코더에 대한 예시적인 구조(200D_E)를 나타낸다. 이 구조는 전술된 루프-내 인트라 전용 정형(200B_E) 및 루프-내 잔차(200C_E) 구조 모두로부터의 요소들을 조합한다. 이 구조 하에서, 인트라 슬라이스들에 대해서, 역방향 사진 정형(265-1)이 루프 필터링(270-1) 이후에 수행된다는 한가지 차이점 이외에, 인트라 슬라이스들은 루프-내 인트라 정형 인코딩 구조(예를 들어, 도 2c 내의 200B_E)에 따라 인코딩된다. 또다른 실시예에서는, 인트라 슬라이스에 대한 루프-내 필터링이 역방향 정형 이후에 수행될 수 있지만; 실험 결과는, 이러한 배치가 역방향 정형이 루프 필터링 이후에 수행되는 것보다 악화된 코딩 효율을 낼 수 있다는 것을 보여준다. 나머지 동작은 전술된 것과 같다.
- [0077] 전술된 바와 같이, 인트라 슬라이스들은 루프-내 잔차 인코딩 구조(예를 들어, 도 2e의 200C_E)에 따라 인코딩된다. 도 2g에 나타낸 바와 같이, 인트라/인트라 슬라이스 스위치는 인코딩되는 슬라이스 유형에 따라 2개의 구조 간에 스위칭을 허용한다.
- [0078] 도 2h는 하이브리드 루프-내 정형을 사용하는 디코더에 대한 예시적인 구조(200D_D)를 나타낸다. 또다시, 인트라 슬라이스들은 루프-내 인트라 정형 디코더 구조(예를 들어, 도 2d의 200B_D)에 따라 디코딩되는데, 여기서 또다시, 인트라 슬라이스들에 대해서, 루프 필터링(270-1)은 역방향 사진 정형(265-1)을 선행한다. 인트라 슬라이스들은 루프-내 잔차 디코딩 구조(예를 들어, 도 2f의 200C_D)에 따라 디코딩된다. 도 2h에 나타낸 바와 같이, 인트라/인트라 슬라이스 스위치는 인코딩된 비디오 사진의 슬라이스 유형에 따라 2가지 구조 간에 스위칭을 허용한다.
- [0079] 도 4a는 또한 도 2g에 나타낸 인코딩 처리(300D_E)를 호출함으로써 하이브리드 루프-내 정형 인코딩 방법을 포함하도록 용이하게 확장될 수 있다. 마찬가지로, 도 4b는 또한 도 2h에 나타낸 디코딩 처리(300D_D)를 호출함으로써 하이브리드 루프-내 정형 디코딩 방법을 포함하도록 용이하게 확장될 수 있다.
- [0080] 슬라이스 수준에서의 정형
- [0081] 본 발명의 실시예들은 다양한 슬라이스-수준 적응들을 허용한다. 예를 들어, 컴퓨팅을 줄이기 위해, 정형이 인트라 슬라이스에 대해서만 또는 인트라 슬라이스에 대해서만 사용가능할 수 있다. 또다른 실시예에서는, 정형이 시간 ID 값(예를 들어, HEVC의 변수 TemporalId(참고 문헌 [11]), 여기서 TemporalId = `nuh_temporal_id_plus1` - 1)에 기초하여 허용될 수 있다. 예를 들어, 현재의 슬라이스에 대한 TemporalId가 미리정의된 값 이하면, 현재의 슬라이스에 대한 `slice_resaper_enable_flag`는 1로 설정될 수 있고, 그렇지 않으면, `slice_resaper_enable_flag`는 0일 것이다. 각각의 슬라이스에 대해 `slice_resaper_enable_flag` 매개변수를 송신하는 것을 피하기 위해, SPS 수준에서 `sps_resaper_temporal_id` 매개변수를 지정할 수 있고, 이에 따라 그 값이 추론될 수 있다.
- [0082] 정형이 사용가능한 슬라이스들에 대해서, 디코더는 어떤 정형 모델이 사용되었는지 알 필요가 있다. 일 실시예에서, 그것은 SPS 수준에서 정의된 정형 모델을 항상 사용할 수 있다. 또다른 실시예에서, 그것은 슬라이스 헤더에 정의된 정형 모델을 항상 사용할 수 있다. 현재의 슬라이스 내에 어떠한 정형 모델도 정의되어 있지 않으면, 그것은 정형을 사용한 가장 최근에 디코딩된 슬라이스에서 사용된 정형 모델을 적용할 수 있다. 또다른 실시예에서, 정형이 인트라 슬라이스에 대해 사용되었는지 아닌지에 상관없이, 정형 모델은 항상 인트라 슬라이스 내에 특정될 수 있다. 이러한 구현에서, 매개변수 `slice_resaper_enable_flag` 및 `slice_resaper_model_present_flag`는 연결해제될 필요가 있다. 이러한 슬라이스 선택스(syntax)의 예가 표 5에 나타나있다.
- [0083] 정형 정보의 시그널링
- [0084] 순방향 및/또는 역방향 정형에 관한 정보는 상이한 정보 계층들, 예를 들어, 비디오 매개변수 세트(video parameter set; VPS), 시퀀스 매개변수 세트(SPS), 사진 매개변수 세트(picture parameter set; PPS), 슬라이스 헤더, 추가 정보(supplemental information; SEI), 또는 임의의 다른 높은-수준 선택스에 제시될 수 있다. 예로서, 제한 없이, 표 2는 정형이 사용가능한지에 대해, 정형이 적응형인지 아닌지에 대해, 3가지 구조 중 어떤 것이 사용 중인지에 대해 시그널링하기 위한, SPS 내의 높은-수준 선택스의 예를 제공한다.

【표 2: SPS 내의 정형 정보의 예】

SPS()	기술자
...	
<code>sps_resampler_enable_flag</code> /*1: 정형 온(on), 그 외에는, 오프(off) */	u(1)
if (sps_resampler_enable_flag) {	
<code>sps_resampler_adaptive_flag</code> /* 1: 적응형 정형 온, 그 외에는, 오프 */	u(1)
<code>sps_resampler_architecture</code> /* 예를 들어: 0: 루프-밖, 1:루프-내 인트라, 2: 루프-내 잔차 */	ue(v)
}	
...	

[0085]

[0086]

추가적인 정보가 또한 몇몇의 다른 계층에서, 즉, 슬라이스 헤더 내에서 전달될 수 있다. 정형 함수들은 룩-업 표들(look-up tables; LUT), 구간적 다항식들, 또는 다른 종류의 매개변수 모델들에 의해 기술될 수 있다. 정형 함수들을 전달하는데 사용되는 정형 모델의 유형은 추가적인 선택스 요소, 예를 들어, **reshaping_model_type** 플래그에 의해 시그널링될 수 있다. 예를 들어, 2가지 별개의 표현을 사용하는 시스템을 고려하자: **model_A**(예를 들어, **reshaping_model_type** = 0)는 구간적 다항식들의 세트로 정형 함수를 나타내는 반면(예를 들어, 참고 문헌 [4]를 참조), **model_B**(예를 들어, **reshaping_model_type** = 1)에서는, 정형 함수가 사진 휘도 특징들 및 시각적인 중요도에 기초하여 상이한 휘도 대역들에 코드워드들을 할당함으로써 적응적으로 도출된다(예를 들어, 참고 문헌 [3]을 참조). 표 3은 사용되는 적절한 정형 모델을 결정하도록 디코더를 돕기 위한, 사진의 슬라이스 헤더 내의 선택스 요소들의 예를 제공한다.

【표 3: 슬라이스 헤더 내에서의 정형 시그널링을 위한 예시적인 선택스】

slice_segment_header()	기술자
...	
if (sps_resampler_adaptive_flag) {	
reshaping_model_type	ue(v)
if (reshaping_model_type == model_A) {	
<code>reshaping_sliceheader_table_model_A()</code>	
}	
else if (reshaping_model_type == model_B) {	
<code>reshaping_sliceheader_table_model_B()</code>	
}	
else ...	
}	
...	

[0087]

[0088]

다음의 3개의 표는 시퀀스, 슬라이스, 또는 코딩 트리 유닛(Coding Tree Unit; CTU) 계층들에서 정형을 시그널링하기 위한 비트스트림 선택스의 대안적인 예를 기술한다.

【표 4: SPS 내의 정형 정보의 예】

SPS()	기술자
...	
<code>sps_resampler_enable_flag</code> /*1: 정형 온, 그 외에는, 오프*/	u(1)
if (sps_resampler_enable_flag) {	
<code>sps_resampler_signal_type</code> /* 0:SDR, 1:PQ, 2:HLG */	u(2)
<code>sps_resampler_ILF_opt</code> /* 그 도메인 내의 루프 필터: 2 비트 인터/인트라 */	u(2)
<code>sps_resampler_chromaAdj</code> /* 1: chromaDQP; 2: chroma scaling*/	u(2)
<code>sps_resampler_model_present_flag</code> /*1: 존재함*/	u(1)
if (sps_resampler_model_present_flag)	
<code>sps_resampler_model ()</code>	
}	

[0089]

【표 5 슬라이스 헤더 내의 정형 시그널링을 위한 예시적인 선택스】

slice_header()	기술자
...	
slice_reshaper_model_present_flag	u(1)
if (slice_reshaper_model_present_flag)	
slice_reshaper_model ()	
slice_reshaper_enable_flag	u(1)
if (slice_reshaper_enable_flag) {	
reshaper_CTU_control_flag /*1: 온, CTU 수준 온/오프 플래그*/	u(1)
}	
...	

[0090]

【표 6 CTU 내의 정형 시그널링을 위한 예시적인 선택스】

coding_tree_unit()	기술자
...	
if (reshape_CTU_control_flag) {	
reshaper_CTU_flag	ae(v)
}	
...	

[0091]

[0092] 표 4 내지 6에 대해서, 예시적인 시맨틱스(semantics)이 다음과 같이 나타내질 수 있다: sps_reshaper_enable_flag가 1인 것은, 정형기가 코딩된 비디오 시퀀스(coded video sequence; CVS)에 사용된다는 것을 특징한다. sps_reshaper_enabled_flag가 0인 것은, 정형기가 CVS에 사용되지 않았다는 것을 특징한다.

[0093] slice_reshaper_enable_flag가 1인 것은, 정형기가 현재의 슬라이스에 대해 사용가능하다는 것을 특징한다. slice_reshaper_enable_flag가 0인 것은, 정형기가 현재의 슬라이스에 대해 사용가능하지 않다는 것을 특징한다.

[0094] sps_reshaper_signal_type은 본래의 코드워드들 분포 또는 표현을 나타낸다. 예로서, 제한 없이, sps_reshaper_signal_type이 0인 것은, SDR (gamma)을 특징하고; sps_reshaper_signal_type이 1인 것은 PQ를 특징하고; sps_reshaper_signal_type이 2인 것은, HLG를 특징한다.

[0095] reshaper_CTU_control_flag가 1인 것은, 정형기가 각각의 CTU에 대해 적응되도록 허용된다는 것을 나타낸다. reshaper_CTU_control_flag가 0인 것은, 정형기가 각각의 CTU에 대해 적응되도록 허용되지 않다는 것을 나타낸다. reshaper_CTU_control_flag가 존재하지 않으면, 그 값은 0이라고 추론된다.

[0096] reshaper_CTU_flag가 1인 것은, 정형기가 현재의 CTU에 대해 사용된다는 것을 특징한다. reshaper_CTU_flag가 0인 것은, 정형기가 현재의 CTU에 대해 사용되지 않는다는 것을 특징한다. reshaper_CTU_flag가 존재하지 않으면, 그 값은 slice_reshaper_enabled_flag와 등가라고 추론될 것이다.

[0097] sps_reshaper_model_present_flag가 1인 것은, sps 내에 sps_reshaper_model()이 존재한다는 것을 나타낸다. sps_reshaper_model_present_flag가 0인 것은, SPS 내에 sps_reshaper_model()이 존재하지 않다는 것을 나타낸다.

[0098] slice_reshaper_model_present_flag가 1인 것은, 슬라이스 헤더 내에 slice_reshaper_model()이 존재한다는 것을 나타낸다. slice_reshaper_model_present_flag가 0인 것은, SPS 내에 slice_reshaper_model()이 존재하지 않다는 것을 나타낸다.

[0099] sps_reshaper_chromaAdj가 1인 것은, 크로마 QP 조정이 chromaDQP를 사용하여 행해진다는 것을 나타낸다. sps_reshaper_chromaAdj가 2인 것은, 크로마 QP 조정이 크로마 스케일링을 사용하여 행해진다는 것을 나타낸다.

[0100] sps_reshaper_ILF_opt는 루프-내 필터가 인트라 및 인터 슬라이스들에 대하여, 본래의 도메인 또는 정형된 도메인 내에서 적용되어야하는지를 나타낸다. 예를 들어, 2-비트 선택스를 사용하는데, 여기서 최하위 비트는 인트라

슬라이스에 대해 나타낸다:

sps_reshaper_ILF_opt	루프-내 필터 동작
0 0	인트라 및 인터 모두에 대하여 본래의 도메인 내
0 1	인터에 대하여 본래의 도메인 내, 인트라에 대하여 정형된 도메인 내
1 0	인터에 대하여 정형된 도메인 내, 인트라에 대하여 본래의 도메인 내
1 1	인트라 및 인터 모두에 대하여 정형된 도메인 내

[0101]

[0102]

몇몇 실시예에서, 이 매개변수는 슬라이스 수준에서 조정될 수 있다. 예를 들어, 실시예에서, slice_reshaper_enable_flag가 1로 설정되면 슬라이스는 slice_reshape_ILFOPT_flag를 포함할 수 있다. 또다른 실시예에서는, SPS에서, sps_reshaper_ILF_opt가 사용가능하면 sps_reshaper_ILF_Tid 매개변수가 포함될 수 있다. 현재의 슬라이스에 대한 TemporalID <= sps_reshaper_ILF_Tid이고 slice_reshaper_enable_flag가 1로 설정되면, 루프-내 필터가 정형 도메인에 적용된다. 그렇지 않으면, 그것은 미-정형된 도메인에 적용된다.

[0103]

표 4에서, 크로마 QP 조정은 SPS 수준에서 제어된다. 실시예에서, 크로마 QP 조정은 또한 슬라이스 수준에서도 제어될 수 있다. 예를 들어, 각각의 슬라이스에서, slice_reshaper_enable_flag가 1로 설정되면 선택스 요소 slice_reshape_chromaAdj_flag가 추가될 수 있다. 또다른 실시예에서는, SPS에서, sps_reshaper_chromaAdj가 사용가능하면, 선택스 요소 sps_reshaper_ChromaAdj_Tid가 추가될 수 있다. 현재의 슬라이스에 대한 TemporalID <= sps_reshaper_ChromaAdj_Tid이고 slice_reshaper_enable_flag가 1로 설정되면, 크로마 조정이 적용된다. 그렇지 않으면, 크로마 조정은 적용되지 않는다. 표 4B는 전술된 선택스를 사용하는 표 4의 예시적인 변형을 나타낸다.

【표 4B: 시간 ID를 사용하는 SPS 내의 정형 시그널링을 위한 예시적인 선택스】

SPS0	기술자
...	
sps_reshaper_enable_flag /*1: 정형 문, 그 외에는, 오프 */	u(1)
if (sps_reshaper_enable_flag) {	
sps_reshaper_signal_type /* 0:HDR, 1:PQ, 2:HLG */	u(2)
sps_reshaper_ILF_opt /* 그 도메인 내의 루프 필터: 2 비트 인터/인트라 */	u(2)
if (sps_reshaper_ILF_opt == 3)	
sps_reshaper_ILF_Tid	u(3)
sps_reshaper_chromaAdj /* 1: chromaDQP: 2: chromaScaling/	u(2)
if (sps_reshaper_chromaAdj)	
sps_reshaper_chromaAdj_Tid	u(3)
sps_reshaper_model_present_flag /*1: 존재함*/	u(1)
if (sps_reshaper_model_present_flag)	
sps_reshaper_model ()	
}	

[0104]

[0105]

sps_reshaper_ILF_Tid는 정형된 도메인 내에서 정형된 슬라이스에 대해 루프-내 필터가 적용되는 최고 TemporalID를 특징한다. sps_reshaper_chromaAdj_Tid는 정형된 슬라이스에 대해 크로마 조정이 적용되는 최고 TemporalID를 특징한다.

[0106]

또다른 실시예에서, 예를 들어, slice_reshape_model() 함수의 부분으로서, 정형-모델 ID, 예를 들어, reshape_model_id를 사용하여 정형 모델이 정의될 수 있다. 정형 모델은 SPS, PPS, 또는 슬라이스-헤더 수준에서 시그널링될 수 있다. SPS 또는 PPS에서 시그널링되면, reshape_model_id의 값은 또한 sps_seq_parameter_set_id 또는 pps_pic_parameter_set_id로부터 추론될 수 있다. slice_reshape_model()를 전달하지 않는(예를 들어, slice_reshaper_model_present_flag가 0임) 슬라이스들에 대해 reshape_model_id를 어떻게 사용할지에 대한 예가 아래 표 5B에 나타나있으며, 표 5의 변형이다.

【표 5B: reshape_model_id를 사용하는 슬라이스 헤더 내의 정형 시그널링을 위한 예시적인

신택스】

slice_header()	기술자
...	
slice_reshaper_model_present_flag	u(1)
if (slice_reshaper_model_present_flag)	
slice_reshaper_model ()	
else	
reshape_model_id	us(v)
slice_reshaper_enable_flag	u(1)
if (slice_reshaper_enable_flag) {	
reshaper_CTU_control_flag /*1: 온, CTU 수준 온/오프	u(1)
플래그 */	
}	
...	

[0107]

[0108]

예시적인 신택스에서, 매개변수 **reshape_model_id**는 사용중인 reshape_model에 대한 값을 특정한다. reshape_model_id의 값은 0 내지 15의 범위에 있을 것이다.

[0109]

제안된 신택스를 사용하는 예로서, 정형이 SPS 수준에서 사용되고, 슬라이스 수준에서는 특정한 정형이 사용되지 않고(정형은 모든 슬라이스들에 대해서 사용됨), CTU 적응은 인터 슬라이스에 대해서만 허용되는, PQ EOTF를 사용하여 코딩된 HDR 신호를 고려해보자. 그러면:

[0110]

sps_reshaper_signal_type = 1 (PQ);

[0111]

sps_reshaper_model_present_flag = 1;

[0112]

// 주의: 인터 슬라이스들에 대해 정형기를 사용가능하게 및 사용불가능하게 하도록 slice_reshaper_enable_flag를 조작할 수 있음.

slice_reshaper_enable_flag = 1;

if (CTUAdp)

{

 if (I_slice)

slice_reshaper_model_present_flag = 0;

reshaper_CTU_control_flag = 0;

 else

slice_reshaper_model_present_flag = 0;

[0113]

```

        reshaper_CTU_control_flag = 1;
    }
else
{
    slice_reshaper_model_present_flag = 0;
    reshaper_CTU_control_flag = 0;
}

```

[0114]

[0115] 또다른 예에서, 정형이 슬라이스 수준에서만 및 인트라 슬라이스들에 대해서만 적용되는 SDR 신호를 고려해보자. CTU 정형 적용은 인트라 슬라이스들에 대해서만 허용된다. 그러면:

```

sps_reshaper_signal_type = 0 (SDR);
sps_reshaper_model_present_flag = 0;
slice_reshaper_enable_flag = 1;
if (I_slice)
{
    slice_reshaper_model_present_flag = 1;
    reshaper_CTU_control_flag = 0;
}
else
{
    slice_reshaper_model_present_flag = 0;
    if (CTUAdp)
        reshape_CTU_control_flag = 1;
    else
        reshaper_CTU_control_flag = 0;
}

```

[0116]

[0117] CTU 수준에서, 실시예에서, CTU-수준 정형은 CTU의 휘도 특징들에 기초하여 사용가능해질 수 있다. 예를 들어, 각각의 CTU에 대해서, 평균 휘도가 컴퓨팅되고(예를 들어, CTU_avg_lum_value), 그것이 하나 이상의 임계와 비교되어, 그들 비교 결과에 기초하여 정형을 온 또는 오프로 전환할지가 결정된다. 예를 들어,

[0118] $CTU_avg_lum_value < THR1$ 이면, 또는

[0119] $CTU_avg_lum_value > THR2$ 이면, 또는

[0120] $THR3 < CTU_avg_lum_value < THR4$ 이면,

[0121] 이 CTU에 대해 $reshaper_CTU_Flag = 1$ 이다.

[0122] 실시예에서, 평균 휘도를 사용하는 대신, 최소, 최대, 또는 평균 휘도와 같은 CTU의 몇몇의 다른 휘도 특징, 변형(variance) 등이 사용될 수 있다. CTU의 크로마-기반 특징들이 적용되거나, 휘도 및 크로마 특징들 및 임계들이 조합될 수 있다.

[0123] (예를 들어, 도 3a, 3b, 및 3c의 단계들에 관련하여) 전송된 바와 같이, 실시예들은 디폴트 또는 고정 정형 함수, 또는 적응형 정형 모두를 지원할 수 있다. "디폴트 정형기"는 미리-정의된 정형 함수를 수행하기 위해 사용될 수 있어, 이에 따라 정형 곡선의 도출시에 각각의 사진 또는 장면을 분석하는 것에 대한 복잡도를 줄일 수

있다. 이 경우에는, 장면, 사진, 또는 슬라이스 수준에서 역방향 정형 함수를 시그널링할 필요가 없다. 디폴트 정형기는 어떠한 시그널링도 피하기 위해 디코더에 저장된, 고정된 매핑 곡선을 사용하여 구현될 수 있거나, 그것은 시퀀스 수준 매개변수 세트의 부분으로서 한번에 시그널링될 수 있다. 또다른 실시예에서, 이전에 디코딩된 적응형 정형 함수가 코딩 순서로 나중의 사진을 위해 재-사용될 수 있다. 또다른 실시예에서, 정형 곡선들은 이전에 디코딩된 것과는 상이한 방식으로 시그널링될 수 있다. (예를 들어, 역방향 정형을 수행하는데 *Inv()* 및 *Fwd()* 함수 모두가 필요한 루프-내 잔차 정형에 대한) 다른 실시예에서, *Inv()* 또는 *Fwd()* 함수 중 오직 하나만이 비트스트림에서 시그널링되거나, 대안적으로, 디코더 복잡도를 줄이기 위해, 둘 다를 시그널링할 수 있다. 표 7 및 8은 정형 정보를 시그널링하는 것에 대한 2가지 예를 제공한다.

[0124] 표 7에서, 정형 함수는 2차 다항식들의 세트로 전달된다. 그것은 탐색 시험 모델(Exploratory Test Model; ETM)의 간략화된 선택스이다(참고 문헌 [5]). 이전 변환을 또한 참고 문헌 [4]에서 찾을 수 있다.

【표 7: 정형 함수의 구간적 표현에 대한 예시적인 선택스(model_A)】

reshaping_sliceheader_table_model_A() {	기술자
reshape_input_luma_bit_depth_minus8	us(v)
coeff_log2_offset_minus2	us(v)
reshape_num_ranges_minus1	us(v)
reshape_equal_ranges_flag	u(1)
reshape_global_offset_val	u(v)
if(!reshape_equal_ranges_flag)	
for (i = 0; i < reshape_num_ranges_minus1+ 1; i++)	
reshape_range_val[i]	u(v)
reshape_continuity_flag	u(1)
for(i = 0; i < reshape_num_ranges_minus1 + 2; i++) {	
reshape_poly_coeff_order0_int[i]	us(v)
reshape_poly_coeff_order0_frac[i]	u(v)
}	
if(reshape_continuity_flag == 1) {	
reshape_poly_coeff_order1_int	se(v)
reshape_poly_coeff_order1_frac	u(v)
}	
}	

- [0125]
- [0126] **reshape_input_luma_bit_depth_minus8**는 정형 처리의 입력 루마 구성요소의 샘플 비트 심도를 특정한 다. **coeff_log2_offset_minus2**는 루마 구성요소에 대한 정형 관련 계수 계산을 위한 소수(fractional) 비트들의 수를 특정한다. **coeff_log2_offset_minus2**의 값은 0 내지 3의 범위에 있을 것이다.
- [0127] **reshape_num_ranges_minus1** 더하기 1은 구간적 정형 함수의 범위들의 수를 특정한다. 존재하지 않으면, **reshape_num_ranges_minus1**의 값은 0이라고 추론된다. **reshape_num_ranges_minus1**은 루마 구성요소에 대해 0 내지 7의 범위에 있을 것이다.
- [0128] **reshape_equal_ranges_flag**가 1인 것은, 구간적 정형 함수가 거의 같은 길이로 NumberRange 조각들(pieces)로 분할된다는 것을 특정한고, 각각의 범위의 길이는 명확히 시그널링되지는 않는다. **reshape_equal_ranges_flag**가 0인 것은, 각각의 범위의 길이가 명확히 시그널링된다는 것을 특정한다.
- [0129] **reshape_global_offset_val**는 0번째 범위의 시작점을 특정하는데 사용되는 오프셋 값을 도출해내는데 사용된다.
- [0130] **reshape_range_val[i]**은 루마 구성요소의 i번째 범위의 길이를 도출해내는데 사용된다.
- [0131] **reshape_continuity_flag**는 루마 구성요소에 대한 정형 함수의 연속성 속성을 특정한다. **reshape_continuity_flag**가 0이면, 0차 연속성(zeroth order continuity)이 연속한 피벗 점(pivot point)들 간의 구간적 선형 역방향 정형 함수들에 적용된다. **reshape_continuity_flag**가 1이면, 1차 매끄러움(first order smoothness)이 연속한 피벗 점들 간의 전체 2차 다항식 역방향 정형 함수들을 도출해내는데 사용된다.
- [0132] **reshape_poly_coeff_order0_int[i]**는 루마 구성요소에 대한 i-번째 조각 0차 다항식 계수의 정수 값을 특정한다.
- [0133] **reshape_poly_coeff_order0_frac[i]**는 루마 구성요소에 대한 i-번째 조각 0차 다항식 계수의 소수(fractional)

값을 특정한다.

[0134] **reshape_poly_coeff_order1_int**는 루마 구성요소에 대한 1차 다항식 계수의 정수 값을 특정한다.

[0135] **reshape_poly_coeff_order1_frac**는 루마 구성요소에 대한 1차 다항식 계수의 소수(fractional) 값을 특정한다.

[0136] 표 8은 전술된 model_B에 따른 대안적인 매개변수 표현의 예시적인 실시예를 기술한다(참고 문헌 [3]).

【표 8: 정형 함수의 매개변수 표현을 위한 예시적인 선택스(model_B)】

reshaping_sliceheader_table_model_B0 {	기술자
reshape_model_profile_type	us(v)
reshape_model_scale_idx	u(2)
reshape_model_min_bin_idx	u(5)
reshape_model_max_bin_idx	u(5)
reshape_model_num_band	u(4)
for (i = 0; i < reshape_model_num_band; i++){	
reshape_model_band_profile_delta [i]	u(1)
}	
}	

[0137]

[0138] 표 8에서, 실시예에서, 선택스 매개변수들은 다음과 같이 정의될 수 있다: **reshape_model_profile_type**은 정형기 구성 처리에 사용될 프로파일 유형을 특정한다.

[0139] **reshape_model_scale_idx**는 정형기 구성 처리에 사용될 스케일 인자(ScaleFactor로 나타냄)의 색인 값을 특정한다. ScaleFactor의 값은 개선된 전체 코딩 효율을 위한 정형 함수의 개선된 제어를 허용한다. 이 ScaleFactor를 사용하는 것에 대한 추가적인 세부사항들은 (예를 들어, 도 5a 및 도 5b에 나타난 바와 같이) 정형 함수 재구성 처리에 대한 설명에 관련하여 제공된다. 예로서, 제한 없이, reshape_model_scale_idx의 값은 0 내지 3의 범위에 있을 것이다. 실시예에서, 하기 표에 나타난 바와 같은 scale_idx와 ScaleFactor 간의 매핑 관계는 다음에 의해 제공된다:

$$\text{ScaleFactor} = 1.0 - 0.05 * \text{reshape_model_scale_idx}.$$

reshape_model_scale_idx	ScaleFactor
0	1.0
1	0.95
2	0.9
3	0.85

[0140]

[0141] 또다른 실시예에서, 보다 효율적인 고정 소수점(fixed-point) 구현에 대해서는,

ScaleFactor = $1 - 1/16 * \text{reshape_model_scale_idx}$ 이다.

reshape_model_scale_idx	ScaleFactor
0	1.0
1	0.9375
2	0.875
3	0.8125

[0143]

[0144] **reshape_model_min_bin_idx**는 정형기 구성 처리에 사용되는 최소 빈 색인(bin index)을 특정한다. **reshape_model_min_bin_idx**의 값은 0 내지 31의 범위에 있을 것이다.

[0145] **reshape_model_max_bin_idx**는 정형기 구성 처리에 사용되는 최대 빈 색인을 특정한다. **reshape_model_max_bin_idx**의 값은 0 내지 31의 범위에 있을 것이다.

[0146] **reshape_model_num_band**는 정형기 구성 처리에 사용되는 대역들의 수를 특정한다. **reshape_model_num_band**의 값은 0 내지 15의 범위에 있을 것이다.

[0147] **reshape_model_band_profile_delta[i]**는 정형기 구성 처리의 i-번째 대역의 프로파일을 조정하는데 사용되는 델타 값을 특정한다. **reshape_model_band_profile_delta[i]**의 값은 0 내지 1의 범위에 있을 것이다.

[0148] 참고 문헌 [3]에 비해, 표 8의 선택스는 "디폴트 프로파일 유형들"의 세트, 즉, 밝음(highlights), 중간-톤, 및 어두움을 정의함으로써 보다 효율적이다. 실시예에서, 각각의 유형은 미리-정의된 시각적인 대역 중요도 프로파일을 갖는다. 미리-정의된 대역들 및 대응하는 프로파일들은 디코더 내에서 고정된 값들로 구현되거나, 또한 (시퀀스 매개변수 세트와 같은) 높은-수준 선택스를 사용하여 시그널링될 수 있다. 인코더에서, 각각의 이미지는 우선 분석되고 프로파일링된 유형들 중 하나로 범주화된다. 프로파일 유형은 선택스 요소 "**reshape_model_profile_type**"에 의해 시그널링된다. 적응형 정형에서는, 전체 범위의 이미지 다이내믹을 캡처하기 위해, 디폴트 프로파일링이 각각의 휘도 대역 또는 휘도 대역의 하위세트(subset)에 대한 델타에 의해 더욱 조정된다. 델타 값들은 휘도 대역의 시각적인 중요도에 기초하여 도출되고, 선택스 요소 "**reshape_model_band_profile_delta**"에 의해 시그널링된다.

[0149] 일 실시예에서, 델타 값은 단지 0 또는 1의 값만을 취할 수 있다. 인코더에서, 시각적인 중요도는 "우세한 대역" 내의 대역 화소들의 퍼센테이지와 전체 이미지 내의 대역 화소들의 퍼센테이지를 비교함으로써 결정되는데, 여기서 우세한 대역은 국부적인 히스토그램(local histogram)을 사용하여 탐지될 수 있다. 작은 국부적인 블록 내에서 한 대역 내의 화소가 집중되면, 그 대역은 아마도 그 블록에서 시각적으로 중요한 것이다. 각각의 대역에 대한 델타 값을 얻기 위한 의미있는 비교를 형성하도록, 우세한 대역들에 대한 총 수가 합산되고, 정규화된다(normalized).

[0150] 디코더에서, 참고 문헌 [3]에 기술된 방법들에 기초하여 정형 LUT들을 도출해내기 위해 정형기 함수 재구성 처리가 호출되어야만 한다. 따라서, LUT를 컴퓨팅하기 위해 단지 구간적 다항식 함수들을 평가할 필요만 있는 단순한 구간적 근사화 모델에 비해 복잡도가 높다. 매개변수-모델 선택스를 사용하는 것의 이점은 그것이 정형기를 사용하는 것에 대한 비트레이트를 상당히 줄일 수 있다는 것이다. 예를 들어, 전형적인 시험 콘텐츠에 기초하여, 표 7에 나타난 모델은 정형기를 시그널링하는데 200-300 비트를 필요로 하지만, (표 8에서와 같은) 매개변수 모델은 약 40비트만을 사용한다.

[0151] 또다른 실시예에서, 표 9에 나타난 바와 같이, 순방향 정형 룩-업 표가 dQP 값들에 대한 매개변수 모델에 따라 도출될 수 있다. 예를 들어, 실시예에서,

$dQP = clip3(min, max, scale * X + offset)$ 인데,

여기서, min 및 max 는 dQP 의 경계를 나타내고, $scale$ 및 $offset$ 은 2개의 모델 매개변수이고, X 는 신호 휘도에 기초하여 도출된 매개변수(예를 들어, 화소의 휘도 값, 또는 블록들에 대하여, 블록 휘도의 메트릭, 예를 들어, 그것의 최소, 최대, 평균, 변형, 표준 편차 등)를 나타낸다. 예를 들어, 제한 없이,

$dQP = clip3(-3, 6, 0.015 * X - 7.5)$ 이다.

【표 9: 정형 함수의 매개변수 표현에 대한 예시적인 선택스(model C)】

sps_reshaper_model_C0 {	기술자
full_range_input_flag	u(1)
dQP_model_scale_int_prec	ue(v)
if (dQP_model_scale_int_prec > 0) {	
dQP_model_scale_int	u(v)
}	
dQP_model_scale_frac_prec_minus16	ue(v)
dQP_model_scale_frac	u(v)
if (dQPMModelScaleAbs) {	
dQP_model_scale_sign	u(1)
}	
dQP_model_offset_int_prec_minus3	ue(v)
dQP_model_offset_int	u(v)
dQP_model_offset_frac_prec_minus1	ue(v)
dQP_model_offset_frac	u(v)
if (dQPMModelOffsetAbs) {	
dQP_model_offset_sign	u(1)
}	
dQP_model_abs_prec_minus3	ue(v)
dQP_model_max_abs	u(v)
if (dQP_model_max_abs) {	
dQP_model_max_sign	u(1)
}	
dQP_model_min_abs	u(v)
if (dQP_model_min_abs) {	
dQP_model_min_sign	u(1)
}	
}	

실시예에서, 표 9의 매개변수는 다음과 같이 정의될 수 있다: **full_range_input_flag**는 입력 비디오 신호 범위를 특정한다. 0의 **full_range_input_flag**은 표준 다이내믹 범위 입력 비디오 신호에 대응한다. 1의 **full_range_input_flag**은 전체 범위 입력 비디오 신호에 대응한다. 존재하지 않으면, **full_range_input_flag**는 0이라고 추론된다.

주의: 여기에 사용된 바와 같이, "전체-범위 비디오"란 용어는 비디오 내의 유효한 코드워드가 "제한되지" 않는다는 것을 나타낸다. 예를 들어, 10-비트 전체-범위 비디오에 대해서, 유효한 코드워드는 0과 1023 사이인데, 여기서 0은 최저 휘도 수준에 매핑된다. 반대로, 10-비트 "표준 범위 비디오"에 대해서는, 유효한 코드워드들이 64와 940 사이이고, 64가 최저 휘도 수준에 매핑된다.

예를 들어, "전체 범위" 및 "표준 범위"의 계산은 다음과 같이 컴퓨팅될 수 있다:

[0 1] 내로 정규화된 휘도 값들(E_y')에 대해서, BD 비트(예를 들어, BD=10, 12 등)로 코딩:

전체 범위: $Y = clip3(0, (1 \ll BD) - 1, E_y' * ((1 \ll BD) - 1)))$

표준 범위: $Y = clip3(0, (1 \ll BD) - 1, round(1 \ll (BD - 8) * (219 * E_y' + 16)))$

이 선택스는 HEVC(H.265) 명세서(참고 문헌 [11])의 섹션 E.2.1에 기술된 바와 같은 HEVC VUI 매개변수들 내의 "video_full_range_flag" 선택스와 유사하다.

dQP_model_scale_int_prec는 **dQP_model_scale_int**의 표현을 위해 사용되는 비트들의 수를 특정한다. **dQP_model_scale_int_prec**가 0인 것은, **dQP_model_scale_int**가 시그널링되지 않고 0이라고 추론된다는 것을 나

타낸다.

[0164] **dQP_model_scale_int**는 dQP 모델 스케일(dQP model scale)의 정수 값을 특정한다.

[0165] **dQP_model_scale_frac_prec_minus16** 더하기 16은 dQP_model_scale_frac의 표현을 위해 사용되는 비트들의 수를 특정한다.

[0166] **dQP_model_scale_frac**는 dQP 모델 스케일의 소수(fractional) 값을 특정한다.

[0167] 변수 dQPModelScaleAbs는 다음과 같이 도출된다:

$$\text{dQPModelScaleAbs} = \text{dQP_model_scale_int} \ll (\text{dQP_model_scale_frac_prec_minus16} + 16) + \text{dQP_model_scale_frac}$$

[0169] **dQP_model_scale_sign**은 dQP 모델 스케일의 부호를 특정한다. dQPModelScaleAbs이 0이면, dQP_model_scale_sign은 시그널링되지 않고 0이라고 추론된다.

[0170] **dQP_model_offset_int_prec_minus3** 더하기 3은 dQP_model_offset_int의 표현을 위해 사용되는 비트들의 수를 특정한다.

[0171] **dQP_model_offset_int**는 dQP 모델 오프셋(dQP model offset)의 정수 값을 특정한다.

[0172] **dQP_model_offset_frac_prec_minus1** 더하기 1은 dQP_model_offset_frac의 표현을 위해 사용되는 비트들의 수를 특정한다.

[0173] **dQP_model_offset_frac**는 dQP 모델 오프셋의 소수(fractional) 값을 특정한다.

[0174] 변수 dQPModelOffsetAbs는 다음과 같이 도출된다:

$$\text{dQPModelOffsetAbs} = \text{dQP_model_offset_int} \ll (\text{dQP_model_offset_frac_prec_minus1} + 1) + \text{dQP_model_offset_frac}$$

[0176] **dQP_model_offset_sign**은 dQP 모델 오프셋의 부호를 특정한다. dQPModelOffsetAbs가 0이면, dQP_model_offset_sign은 시그널링되지 않고 0이라고 추론된다.

[0177] **dQP_model_abs_prec_minus3** 더하기 3은 dQP_model_max_abs 및 dQP_model_min_abs의 표현을 위해 사용되는 비트들의 수를 특정한다.

[0178] **dQP_model_max_abs**는 dQP 모델 최대(dQP model max)의 정수 값을 특정한다.

[0179] **dQP_model_max_sign**는 dQP 모델 최대의 부호를 특정한다. dQP_model_max_abs가 0이면, dQP_model_max_sign은 시그널링되지 않고 0이라고 추론된다.

[0180] **dQP_model_min_abs**는 dQP 모델 최소(dQP model min)의 정수 값을 특정한다.

[0181] **dQP_model_min_sign**은 dQP 모델 최소의 신호를 특정한다. dQP_model_min_abs가 0이면, dQP_model_min_sign은 시그널링되지 않고 0이라고 추론된다.

[0182] 모델 C에 대한 디코딩 처리

[0183] 표 9의 선택스 요소들이 주어지면, 정형 LUT가 다음과 같이 도출될 수 있다.

[0184] 변수 dQPModelScaleFP는 다음과 같이 도출된다:

$$\text{dQPModelScaleFP} = ((1 - 2 * \text{dQP_model_scale_sign}) * \text{dQPModelScaleAbs}) \ll (\text{dQP_model_offset_frac_prec_minus1} + 1).$$

[0186] 변수 dQPModelOffsetFP는 다음과 같이 도출된다:

$$\text{dQPModelOffsetFP} = ((1 - 2 * \text{dQP_model_offset_sign}) * \text{dQPModelOffsetAbs}) \ll (\text{dQP_model_scale_frac_prec_minus16} + 16).$$

변수 dQPModelShift는 다음과 같이 도출된다:

$$\text{dQPModelShift} = (\text{dQP_model_offset_frac_prec_minus1} + 1) + (\text{dQP_model_scale_frac_prec_minus16} + 16).$$

변수 dQPModelMaxFP는 다음과 같이 도출된다:

$$\text{dQPModelMaxFP} = ((1 - 2 * \text{dQP_model_max_sign}) * \text{dQP_model_max_abs}) \ll \text{dQPModelShift}.$$

변수 dQPModelMinFP는 다음과 같이 도출된다:

$$\text{dQPModelMinFP} = ((1 - 2 * \text{dQP_model_min_sign}) * \text{dQP_model_min_abs}) \ll \text{dQPModelShift}.$$

```
for Y=0: maxY // 예를 들어, 10-비트 비디오에 대해서, maxY=1023
{
    dQP[Y] = clip3( dQPModelMinFP, dQPModelMaxFP, dQPModelScaleFP*Y +
dQPModelOffsetFP);
    slope[Y] = exp2((dQP[Y]+3)/6); // exp2(x) = 2^(x) 인 곳에서의 고정 소수점 exp2 구현
    exp2(x) = 2^(x);
}
If ( full_range_input_flag == 0 ) // 입력이 표준 범위 비디오이면
    For Y out of standard range (i.e. Y = [0:63] and [940:1023]), set slope[Y] = 0;

CDF[0] = slope[0];
for Y=0: maxY-1
{
    CDF[Y+1] = CDF[Y] + slope[Y]; // CDF[Y] 는 slope[Y] 의 적분임
}

for Y=0: maxY
{
    FwdLUT[Y] = round(CDF[Y]*maxY/CDF[maxY]); //FwdLUT 를 얻기 위해 반올림 및 정규화
}
```

또다른 실시예에서, 표 10에 나타낸 바와 같이, 순방향 정형 함수는 루마 피봇 점들(In_Y) 및 그들의 대응하는 코드워드들(Out_Y)의 집합으로 표현될 수 있다. 코딩을 간단히 하기 위해, 입력 휘도 범위는 선형의 구간적 표현을 사용하여 시작 피봇 및 동일하게-이격된 후속한 피봇들의 시퀀스의 면에서 기술된다. 10-비트 입력 데이터에 대한 순방향 정형 함수를 나타내는 예가 도 7에 나타나있다.

【표 10: 정형 함수의 피벗-기반 표현에 대한 예시적인 선택스(model D)】

sps_resaper_model_D0 {	기술자
full_range_input_flag	u(1)
bin_pivot_start	u(v)
bin_cw_start	u(v)
log2_num_equal_bins_minus3	us(v)
equal_bin_pivot_delta	u(v)
bin_cw_in_first_equal_bin	u(v)
bin_cw_delta_abs_prec_minus4	us(v)
for(i = 0 ; i < NumEqualBins - 1 ; i++) {	
bin_cw_delta_abs[i]	u(v)
if (bin_cw_delta_abs[i]) {	
bin_cw_delta_sign[i]	u(1)
}	
}	
}	

[0196]

[0197]

실시예에서, 표 10의 매개변수들은 다음과 같이 정의될 수 있다: **full_range_input_flag**는 입력 비디오 신호 범위를 특정한다. 0의 **full_range_input_flag**는 표준 범위 입력 비디오 신호에 대응한다. 1의 **full_range_input_flag**는 전체 범위 입력 비디오 신호에 대응한다. 존재하지 않으면, **full_range_input_flag**는 0이라고 추론된다.

[0198]

bin_pivot_start는 제1의 동일-길이 빈(710)의 피벗 값을 특정한다. **full_range_input_flag**이 0이면, **bin_pivot_start**는 최소 표준 범위 입력 이상, 최대 표준 범위 입력 미만일 것이다. (예를 들어, 10-비트 SDR 입력에 대해서, **bin_pivot_start**(710)은 64와 940 사이일 것임).

[0199]

bin_cw_start는 **bin_pivot_start**(710)의 매핑된 값(715)을 특정한다(예를 들어, **bin_cw_start** = FwdLUT[**bin_pivot_start**]).

[0200]

log2_num_equal_bins_minus3 더하기 3은 시작 피벗(710)에 후속하는 동일-길이 빈들의 수를 특정한다. 변수 **NumEqualBins** 및 **NumTotalBins**는 다음에 의해 정의된다:

NumEqualBins = 1<<(log2_num_equal_bins_minus3+3)

if **full_range_input_flag** == 0

NumTotalBins = **NumEqualBins** + 4

else

NumTotalBins = **NumEqualBins** + 2

[0201]

[0202]

주의: 실험 결과는 대부분의 순방향 정형 함수들이 8개의 동일-길이 세그먼트들을 사용하여 표현될 수 있다는 것을 보여주지만; 복잡한 정형 함수들은 보다 많은 세그먼트들(예를 들어, 16개 이상)을 필요로 할 수 있다.

[0203]

equal_bin_pivot_delta는 동일-길이 빈들의 길이를 특정한다(예를 들어, 720-1, 720-N). **NumEqualBins** * **equal_bin_pivot_delta**은 유효 입력 범위 이하일 것이다. (예를 들어, **full_range_input_flag**가 0이면, 10-비트 입력들에 대해서, 유효 입력 범위는 940-64 = 876이고; **full_range_input_flag**가 1이면, 10-비트 입력들에 대해서, 유효 입력 범위는 0에서 1023까지이다.)

[0204]

bin_cw_in_first_equal_bin은 제1의 동일-길이 빈(720-1) 내의 매핑된 코드워드들(725)의 수를 특정한다.

[0205]

bin_cw_delta_abs_prec_minus4 더하기 4는 각각의 후속하는 동일한 빈에 대하여 **bin_cw_delta_abs[i]**의 표현을 위해 사용되는 비트들의 수를 특정한다.

[0206]

bin_cw_delta_abs[i]는 각각의 후속하는 동일-길이 빈에 대한 **bin_cw_delta_abs[i]**의 값을 특정한다. **bin_cw_delta[i]** (e.g., 735)는 이전의 동일-길이 빈(i-1) 내의 코드워드들(예를 들어, 730)에 대한 현재의 동일-길이 빈(i)(예를 들어, 720-N) 내의 코드워드들(예를 들어, 740)의 차이이다.

[0207]

bin_cw_delta_sign[i]는 **bin_cw_delta_abs[i]**의 부호를 특정한다. **bin_cw_delta_abs[i]**이 0이면, **bin_cw_delta_sign[i]**는 시그널링되지 않고 0이라고 추론된다. 변수 **bin_cw_delta[i]** = (1-

$2 * \text{bin_cw_delta_sign}[i] * \text{bin_cw_delta_abs}[i]$ 이다.

[0208] 모델 D에 대한 디코딩 처리

[0209] 표 10의 선택스 요소들이 주어지면, 10-비트 입력에 대하여, 정형 LUT가 다음과 같이 도출될 수 있다:

[0210] 상수들 정의:

[0211] $\text{minIN} = \text{minOUT} = 0;$

[0212] $\text{maxIN} = \text{maxOUT} = 2^{\text{BD}} - 1 = 10\text{-비트에 대해 } 1023 \quad // \text{BD} = \text{비트 심도}$

[0213] $\text{minStdIN} = 10\text{-비트에 대해 } 64$

[0214] $\text{maxStdIN} = 10\text{-비트에 대해 } 940$

[0215] 단계 1: $j = 0$ 에 대하여 피벗 값 $\text{In_Y}[j]$ 을 도출: NumTotalBins

$\text{In_Y}[0] = 0;$

$\text{In_Y}[\text{NumTotalBins}] = \text{maxIN};$

if (full_range_input_flag == 0)

{

$\text{In_Y}[1] = \text{minStdIN};$

$\text{In_Y}[2] = \text{bin_pivot_start};$

[0216]

for ($j = 3; \text{NumTotalBins} - 2$)

$\text{In_Y}[j] = \text{In_Y}[j - 1] + \text{equal_bin_pivot_delta};$

$\text{In_Y}[\text{NumTotalBins} - 1] = \text{maxStdIN};$

}

else

{

$\text{In_Y}[1] = \text{bin_pivot_start};$

for $j = 2; \text{NumTotalBins} - 1$

$\text{In_Y}[j] = \text{In_Y}[j - 1] + \text{equal_bin_pivot_delta};$

}

[0217]

[0218] 단계 2: $j = 0$ 에 대하여 매핑된 값 $\text{Out_Y}[j]$ 을 도출: NumTotalBins

```

Out_Y [ 0 ] = 0;
Out_Y [ NumTotalBins ] = maxOUT;

if ( full_range_input_flag == 0 )
{
    Out_Y [ 1 ] = 0;
    Out_Y [ 2 ] = bin_cw_start;
    Out_Y [ 3 ] = bin_cw_start + bin_cw_in_first_equal_bin;
    bin_cw [ 3 ] = bin_cw_in_first_equal_bin;
    for j = ( 4 : NumTotalBins - 2 )
        bin_cw [ j ] = bin_cw [ j - 1 ] + bin_cw_delta [ j - 4 ]; // bin_cw_delta[ i ] 은
idx 0 에서부터 시작
    for j = ( 4 : NumTotalBins - 2 )
        Out_Y [ j ] = Out_Y [ j - 1 ] + bin_cw [ j ];
    Out_Y [ NumTotalBins - 1 ] = maxOUT;
}
else
{
    Out_Y [ 1 ] = bin_cw_start;
    Out_Y [ 2 ] = bin_cw_start + bin_cw_in_first_equal_bin;
    bin_cw [ 2 ] = bin_cw_in_first_equal_bin;
    for j = ( 3 : NumTotalBins - 1 )

        bin_cw [ j ] = bin_cw [ j - 1 ] + bin_cw_delta [ j - 3 ]; // bin_cw_delta[ i ] 는
idx 0 에서부터 시작
    for j = 3: NumTotalBins - 1
        Out_Y [ j ] = Out_Y [ j - 1 ] + bin_cw [ j ];
}

```

[0219]

[0220]

단계 3: 모든 LUT 엔트리를 얻기 위한 선형 보간

```

Init: FwdLUT[ ]

for ( j = 0: NumTotalBins - 1 )
{
    InS  = In_Y [ j ];
    InE  = In_Y [ j + 1 ];
    OutS = Out_Y [ j ];
    OutE = Out_Y [ j + 1 ];

    for i = In_Y[ j ] : In_Y[ j + 1 ] - 1
    {
        FwdLUT [ i ] = OutS + round ( ( OutE - OutS ) * ( i - InS ) / ( InE - InS ) );
    }
}

FwdLUT [ In_Y [ NumTotalBins ] ] = Out_Y [ NumTotalBins ];

```

[0221]

[0222] 일반적으로, 정형은 각각의 슬라이스에 대해 온 또는 오프 스위칭될 수 있다. 예를 들어, 인트라 슬라이스들에 대해서만 정형을 사용가능하게 하고 인터 슬라이스들에 대해서는 정형을 사용불가능하게 할 수 있다. 또다른 예에서는, 최고 시간 수준을 갖는 인터-슬라이스들에 대해서는 정형을 사용불가능하게 할 수 있다. (주의: 예로서, 여기에 사용된 바와 같이, 시간 하위-계층들(temporal sub-layers)은 HEVC 내의 시간 하위-계층들의 정의에 매칭할 수 있다.) 정형기 모델을 정의할 때, 일례에서는, 단지 정형기 모델이 SPS 내에서 시그널링될 수 있지만, 또다른 예에서는, 슬라이스 정형기 모델이 인트라 슬라이스들 내에서 시그널링될 수 있다. 대안적으로, 정형기 모델이 SPS 내에서 시그널링되고, 슬라이스 정형기 모델로 하여금 모든 슬라이스들에 대해 SPS 정형기 모델을 갱신하도록 허용하거나, 슬라이스 정형기 모델로 하여금 인트라 슬라이스들에 대해 SPS 정형기 모델을 갱신하도록 허용할 수 있다. 인트라 슬라이스를 따르는 인터 슬라이스들에 대하여, SPS 정형기 모델 또는 인트라 슬라이스 정형기 모델이 적용될 수 있다.

[0223] 또다른 예로서, 도 5a 및 5b는 실시예에 따른 디코더 내에서의 정형 함수 재구성 처리를 나타낸다. 처리는, [0 5] 내의 시각적인 등급 범위를 가지면서, 여기에 및 참고 문헌 [3]에 기술된 방법들을 사용한다.

[0224] 도 5a에 나타난 바와 같이, 첫번째로(단계(510)), 디코더는 reshape_model_profile_type 변수를 추출하고, 각각의 빈에 대해서 적합한 초기 대역 프로파일을 설정한다(단계(515, 520, 및 525)). 예를 들어, 수도코드(pseudocode)에서:

```

if (reshape_model_profile_type == 0) R[bi] = Rbright[bi];
elseif (reshape_model_profile_type == 1) R[bi] = Rdark[bi];
else R[bi] = Rmid[bi].

```

[0225]

[0226] 단계(530)에서, 디코더는 수신된 reshape_model_band_profile_delta[b_i] 값들을 사용하여, 다음에서와 같이, 각각의 대역 프로파일을 조정한다.

```

for (i = 0: reshape_model_num_band-1)
    {R[bi] = R[bi] + reshape_model_band_profile_delta [bi]}.

```

[0227]

[0228] 단계(535)에서, 디코더는 조정된 값들을 다음에서와 같이 각각의 빈 프로파일에 전파한다.

```

if bin[j] belongs to band bi , Rbin[j] = R[bi]

```

[0229]

[0230] 단계(540)에서, 빈 프로파일은 다음에서와 같이 수정된다.

if (j > reshape_model_max_bin_idx) or (j < reshape_model_min_bin_idx)
{R_bin[j] = 0 }.

[0231]

[0232] 병렬로, 단계(545) 및 단계(550)에서, 디코더는 매개변수들을 추출하여 다음에서와 같이 스케일 인자 값 및 각각의 bin[j]에 대한 후보 코드워드들을 컴퓨팅할 수 있다.

ScaleFactor = 1.0 - 0.05* reshape_model_scale_idx
CW_dft[j] = 디폴트 정형을 사용할 때의 빈 내의 코드워드들
CW_PQ[j]=TotalCW/TotalNumBins.

[0233]

[0234] ScaleFactor 값을 컴퓨팅할 때, 고정 소수점 구현에 대하여, 스케일러 0.05를 사용하는 대신 1/16=0.0625을 사용할 수 있다.

[0235]

도 5b로 이어서, 단계(560)에서, 디코더는 빈 프로파일에 기초하여 각각의 빈에 대하여 다음에서와 같이 코드워드(codeword; CW) 사전-할당을 시작한다.

R_bin[j] == 0이면, CW[j] = 0
R_bin[j] == 1이면, CW[j] = CW_dft[j]/2;
R_bin[j] == 2이면, CW[j] = min(CW_PQ[j], CW_dft[j]);
R_bin[j] == 3이면, CW[j] = (CW_PQ[j] + CW_dft[j])/2;
R_bin[j] >=4이면, CW[j] = max(CW_PQ[j], CW_dft[j]);

[0236]

[0237] 단계(565)에서, 다음에서와 같이, 디코더는 총 사용된 코드워드들을 컴퓨팅하고 코드워드(CW) 할당을 개선/완료한다.

CW_{used} = Sum(CW[j]):
if CW_{used} > TotalCW, rebalance CW[j] = CW[j]/(CW_{used}/TotalCW);
else
{
 CW_remain = TotalCW - CW_{used} ;
 CW_remain is assigned to the bins with largest R_bin[j]);
}

[0238]

[0239] 마지막으로, 단계(567)에서, 디코더는 a) CW[j] 값들을 축적하여 순방향 정형 함수(예를 들어, FwdLUT)를 생성하고, b) 최종 FwdLUT(FFwdLUT)를 형성하기 위해 ScaleFactor 값과 FwdLUT 값들을 곱하고, c) FFwdLUT에 기초하여 역방향 정형 함수 InvLUT를 생성한다.

[0240]

고정 소수점 구현에서, ScaleFactor와 FFwdLUT를 컴퓨팅하는 것은 다음과 같이 표현될 수 있는데:

ScaleFactor = (1<< SF_PREC) - reshape_model_scale_idx
FFwdLUT = (FwdLUT * ScaleFactor + (1 << (FP_PREC + SF_PREC - 1))) >> (FP_PREC + SF_PREC),

[0241]

[0242] 여기서, SF_PREC 및 FP_PREC는 미리정의된 정확도-관련 변수들이고(예를 들어, SF_PREC=4, 및 FP_PREC=14), " $c = a \ll n$ "는 a를 n 비트만큼 이진 좌측 편이하는 연산(operation)을 나타내고(또는 $c = a * (2^n)$), " $c = a \gg n$ "

는 a 를 n 비트만큼 이진 우측 편이하는 연산을 나타낸다(또는 $c = a/(2^n)$).

[0243] 크로마 QP 파생(Chroma QP derivations)

[0244] 크로마-코딩 성능은 루마-코딩 성능과 밀접하게 관련된다. 예를 들어, AVC 및 HEVC에서, 루마 구성요소와 크로마 구성요소에 대한 양자화 매개변수들(QP) 간의 관계, 또는 휘도와 색차 간의 관계를 특정하는 표가 정의된다. 명세서들은 또한 루마와 크로마 간의 QP 관계를 정의하는 것의 추가적인 유연성을 위해 하나 이상의 크로마 QP 오프셋들을 사용하도록 허용한다. 정형이 사용되면, 루마 값이 수정되고, 이에 따라, 휘도와 색차 간의 관계도 또한 수정될 수 있다. 정형 하에서 코딩 효율을 유지하고 더욱 개선시키기 위해, 실시예에서는, 코딩 유닛(coding unit; CU) 수준에서, 크로마 QP 오프셋이 정형 곡선에 기초하여 도출된다. 이 동작은 디코더 및 인코더 모두에서 수행될 필요가 있다.

[0245] 여기에 사용된 바와 같이, "코딩 유닛"(CU)이란 용어는 (예를 들어, 매크로블록 등과 같은) 코딩된 블록을 나타낸다. 예를 들어, 제한 없이, HEVC에서, CU는 "루마 샘플들의 코딩 블록, 3개의 샘플 어레이를 갖는 사진의 크로마 샘플들의 2개의 대응하는 코딩 블록들, 또는 샘플들을 코딩하는데 사용되는 선택스 구조들 및 3개의 개별적인 색 평면들을 사용하여 코딩된 사진 또는 흑백 사진의 샘플들의 코딩 블록"으로 정의된다.

[0246] 실시예에서, 크로마 양자화 매개변수(QP)(chromaQP) 값은 다음과 같이 도출될 수 있다:

[0247] 1) 정형 곡선에 기초하여, 등가의 루마 dQP 매핑을 도출하는데,

dQPLUT:

for CW = 0: MAX_CW_VALUE-1

dQPLUT [CW] = $-6 * \log_2(\text{slope}[CW])$;

[0248]

[0249] 여기서, slope[CW]는 각각의 CW(코드워드) 점에서의 순방향 정형 곡선의 기울기를 나타내고, MAX_CW_VALUE는 주어진 비트-심도, 예를 들어, 10-비트 신호에 대한 최대 코드워드 값임,

[0250] MAX_CW_VALUE = $1024 (2^{10})$.

[0251] 그 후, 각각의 코딩 유닛(CU)에 대하여:

[0252] 2) AvgY로 나타낸, 코딩 유닛의 평균 휘도를 컴퓨팅함:

[0253] 3) 다음의 표 11에 나타낸 바와 같이, dQPLUT[], AvgY, 정형 구조, 역방향 정형 함수 Inv(), 및 슬라이스 유형에 기초하여, chromaDQP 값을 컴퓨팅함:

【표 11: 정형 구조들에 따른 예시적인 chromaDQP 값들】

정형 구조	인트라 슬라이스	인터 슬라이스
루프-밖	dQPLUT[Inv(AvgY)]	dQPLUT[Inv(AvgY)]
루프-내 인트라 전용 정형기	dQPLUT[Inv(AvgY)]	0
잔차들용 루프-내 정형기	dQPLUT[AvgY]	dQPLUT[AvgY]
하이브리드 루프-내 정형	dQPLUT[Inv(AvgY)]	dQPLUT[AvgY]

[0254]

[0255] 4) 다음과 같이 chromaQP를 컴퓨팅함:

[0256] chromaQP = QP_luma + chromaQPOffset + chromaDQP;

[0257] 여기서, chromaQPOffset은 크로마 QP 오프셋을 나타내고, QP_luma는 코딩 유닛에 대한 루마 QP를 나타냄. 크로마 QP 오프셋의 값은 각각의 크로마 구성요소(즉, Cb 및 Cr)에 대해 상이할 수 있고, 크로마 QP 오프셋 값들은 코딩된 비트스트림의 부분으로서 디코더에 전달된다는 것을 주의하자.

[0258] 실시예에서, dQPLUT[]는 미리-정의된 LUT로서 구현될 수 있다. 모든 코드워드가 N개의 bin들로 나뉘고(예를

들어, $N = 32$) 각각의 빈이 $M = \text{MAX_CW_VALUE}/N$ 개의 코드워드를 포함한다고(예를 들어, $M = 1024/32 = 32$) 가정하자. 각각의 빈에 새로운 코드워드들이 할당되면, 코드워드들의 수가 1 내지 $2 \cdot M$ 으로 제한될 수 있고, 따라서, $\text{dQPLUT}[1 \dots 2 \cdot M]$ 이 사전 컴퓨팅되고 그것이 LUT로서 저장될 수 있다. 이 접근법은 임의의 부동 소수점 컴퓨팅 또는 고정 소수점 컴퓨팅의 근사화를 피할 수 있다. 그것은 또한 인코딩/디코딩 시간을 절약할 수 있다. 각각의 빈에 대하여, 하나의 고정된 chromaQPoffset 이 이 빈 내의 모든 코드워드들에 대해 사용된다. DQP 값은 $\text{dQPLUT}[L]$ 과 같도록 설정되는데, 여기서 L 은 이 빈에 대한 코드워드들의 수이고, $1 \leq L \leq 2 \cdot M$ 이다.

[0259] dQPLUT 값들은 다음과 같이 사전 컴퓨팅될 수 있다:

```
for i=1:2*M
    slope[i] = i/M;
    dQPLUT[i] = -6*log2(slope[i]);
end
```

[0260]

[0261] $\text{round}()$, $\text{ceil}()$, $\text{floor}()$, 또는 그들의 혼합과 같은 상이한 양자화 방식들을 사용하여, $\text{dQPLUT}[x]$ 을 컴퓨팅할 때 정수 QP 값을 얻을 수 있다. 예를 들어, 임계 TH 를 설정하고, $Y < TH$ 이면, dQP 값을 양자화하기 위해 $\text{floor}()$ 을 사용하고, 그렇지 않고, $Y \geq TH$ 이면, dQP 값을 양자화하기 위해 $\text{ceil}()$ 을 사용할 수 있다. 이러한 양자화 방식들 및 대응하는 매개변수들의 사용은 코덱 내에 미리-정의되어 있을 수 있거나 적응(adaptation)을 위해 비트스트림 내에서 시그널링될 수 있다. 전술된 바와 같이 1개의 임계를 가지면서 양자화 방식들의 혼합을 허용하는 예시적인 선택스가 다음과 같이 나타나있다:

[0262]

[0263] $\text{quant_scheme_signal_table}()$ 함수는 적응 필요성에 따라, (예를 들어, 시퀀스 수준, 슬라이스 수준 등과 같은) 상이한 수준의 정형 선택스에서 정의될 수 있다.

[0264] 또다른 실시예에서, chromaDQP 값들은 각각의 코딩 유닛(또는 더욱 구체적으로는 변형 유닛) 내의 잔차 신호에 스케일링 인자를 적용함으로써 컴퓨팅될 수 있다. 이 스케일링 인자는 루마-종속 값일 수 있고, a) 수치적으로, 예를 들어, 순방향 정형 LUT의 1차 도함수(기울기)로서(예를 들어, 다음 섹션의 수학적 6을 참조) 또는 b)

[0265]

$$\text{Slope}(x) = 2^{\left(\frac{\text{dQP}(x)}{6}\right)}$$

로서 컴퓨팅될 수 있다.

[0266]

$\text{dQP}(x)$ 를 사용하여 $\text{Slope}(x)$ 를 컴퓨팅할 때, dQP 는 정수 양자화 없이 부동 소수점 정확도로 유지될 수 있다. 대안적으로, 여러 상이한 양자화 방식들을 사용하여 양자화된 정수 dQP 값들을 컴퓨팅할 수 있다. 몇몇 실시예들에서, 이러한 스케일링은 블록 수준 대신 화소 수준에서 수행될 수도 있는데, 여기서 각각의 크로마 잔차는 그 크로마 샘플의 병치된(co-located) 루마 정확도 값을 사용하여 도출된, 상이한 스케일 인자에 의해 스케일링될 수 있다. 따라서,

【표 12: 하이브리드 루프-내 정형 구조에 대한 스케일링을 사용한 예시적인 크로마 dQP 값들】

정형	인트라 슬라이스	인터 슬라이스
CU-기반 크로마 스케일링(같은 S_{cu} 스케일러가 CU 내의 모든 샘플들에 의해 공유됨)	$S_{cu} = \text{SlopeLUT}[\text{Inv}(\text{AvgY})]$ $C_{Res_scaled} = C_{Res} * S_{cu}$	$S_{cu} = \text{SlopeLUT}[\text{AvgY}]C_{Res_scaled} = C_{Res} * S_{cu}$
화소-기반 크로마 스케일링(각각의 샘플 내의 상이한 S_{px} 스케일러)	$S_{px} = \text{SlopeLUT}[\text{Inv}(\text{ColPredY})]$ $C_{Res_scaled} = C_{Res} * S_{px}$	$S_{px} = \text{SlopeLUT}[\text{ColPredY}]$ $C_{Res_scaled} = C_{Res} * S_{px}$

예를 들어, $CSCALE_FP_PREC = 16$ 이면

- 순방향 스케일링: 크로마 잔차가 생성된 후, 변형 및 양자화 전:

$$\begin{aligned}
 & - C_{Res} = C_{orig} - C_{pred} \\
 & - C_{Res_scaled} = C_{Res} * S + (1 \ll (CSCALE_FP_PREC - 1)) \gg \\
 & \quad CSCALE_FP_PREC
 \end{aligned}$$

- 역방향 스케일링: 크로마 역방향 양자화 및 역방향 변형 후, 그러나 재구성 전:

$$\begin{aligned}
 & - C_{Res_inv} = (C_{Res_scaled} \ll CSCALE_FP_PREC) / S \\
 & - C_{Reco} = C_{Pred} + C_{Res_inv};
 \end{aligned}$$

여기서 S 는 S_{cu} 또는 S_{px} 이다.

주의: 표 12에서, S_{cu} 를 컴퓨팅할 때, 블록의 평균 루마(AvgY)는 역방향 정형을 적용하기 전에 계산된다. 대안적으로, 평균 루마를 컴퓨팅하기 전에 역방향 정형이 적용될 수도 있다(예를 들어, $S_{cu} = \text{SlopeLUT}[\text{Avg}(\text{Inv}[Y])]$). 이 대안적인 컴퓨팅 순서는 표 11의 값들 컴퓨팅에도 또한 적용되는데; 즉, $\text{Inv}(\text{AvgY})$ 를 컴퓨팅하는 것은 $\text{Avg}(\text{Inv}[Y])$ 값들을 컴퓨팅하는 것으로 대체될 수 있다. 후자의 접근법이 보다 정확하다고 고려될 수 있지만, 컴퓨팅 복잡도가 증가한다.

정형에 관련된 인코더 최적화

이 섹션은, 정형이 (3개의 후보 구조 중 하나에서 기술된 바와 같은) 규범적인 디코딩 처리의 부분일 때, 정형 및 인코더 매개변수들을 함께 최적화함으로써, 인코더 내의 코딩 효율을 개선시키기 위한 다수의 기술들을 설명한다. 일반적으로, 인코더 최적화 및 정형은 그들 자신의 제한사항들을 갖는 상이한 위치들에서의 코딩 문제점과 씨름을 한다(tackle). 전형적인 이미징 및 코딩 시스템에서는, 2가지 유형의 양자화가 존재한다: a) 기저대역 신호 내에서의 샘플 양자화(예를 들어, 감마 또는 PQ 코딩) 및 변형-관련 양자화(압축의 부분). 정형은 그 사이에 위치한다. 사진-기반 정형은 일반적으로 사진 기반으로 갱신되고, 어떠한 공간적인 정보도 고려하지 않고 단지 그것의 휘도 수준에만 기초하여 샘플 값 매핑들을 허용한다. (HEVC와 같은) 블록-기반 코덱에서는, (예를 들어, 루마에 대한) 변형 양자화가 공간적인 블록 내에 적용되고 공간적으로 조정될 수 있으므로, 인코더 최적화 방법들은 상이한 휘도 값들을 갖는 샘플들을 포함하는 전체 블록에 대해 같은 매개변수 세트를 적용해야만 한다. 발명자에 의해 인식되고 여기에 기술된 바와 같이, 공동의 정형 및 인코더 최적화는 코딩 효율을 더욱 개선시킬 수 있다.

인터/인트라 모드 결정

전형적인 코딩에서, 인터/인트라-모드 결정들은 본래의 샘플들과 예측 샘플들 간에 왜곡 함수($dfunc()$)를 컴퓨팅하는 것에 기초한다. 이러한 함수들의 예는 제곱 오차들의 합(sum of square errors; SSE), 절대차들의 합(sum of absolute differences; SAD), 및 다른 것들을 포함한다. 실시예에서, 이러한 왜곡 매트릭들은 정형된 화소 값들을 사용하여 사용될 수 있다. 예를 들어, 정형이 적용되면, 본래의 $dfunc()$ 가 $Orig_sample(i)$ 및

$Pred_sample(i)$ 를 사용하면, $dfunc()$ 는 그들의 대응하는 정형된 값들, $Fwd(Orig_sample(i))$, 및 $Fwd(Pred_sample(i))$ 을 사용할 수 있다. 이 접근법은 보다 정확한 인터/인트라 모드 결정을 허용하고, 따라서 코딩 효율이 개선된다.

[0279] 정형을 갖는 $LumaDQP$

[0280] JCTVC HDR 공통 시험 조건들(common test conditions; CTC) 문서(참고 문헌 [6])에서, $lumaDQP$ 및 $chromaQPoffset$ 들은 HDR 코딩 효율을 개선시키기 위한, 루마 및 크로마 구성요소들에 대한 양자화(QP) 매개변수들을 수정하는데 사용되는 2개의 인코더 설정이다. 본 발명에서, 본래 제안을 더욱 개선시키기 위한 몇몇의 새로운 인코더 알고리즘이 제안된다. 각각의 $lumaDQP$ 적응 유닛(예를 들어, 64x64 CTU)에 대해서, (참고 문헌 [6]의 표 3에서와 같이) dQP 값이 유닛의 평균 입력 루마 값에 기초하여 컴퓨팅된다. 이 $lumaDQP$ 적응 유닛 내의 각각의 코딩 유닛에 대해 사용되는 최종 양자화 매개변수 QP는 이 dQP 를 감산함으로써 조정되어야 한다. dQP 매핑 표는 인코더 입력 구성에서 구성가능하다. 이 입력 구성은 dQP_{inp} 로 나타내진다.

[0281] 참고 문헌 [6] 및 [7]에 설명된 바와 같이, 기존의 코딩 방법들에서, 같은 $lumaDQP$ LUT dQP_{inp} 가 인트라 및 인터 사진들 모두에 대해 사용된다. 인트라-사진 및 인터-사진은 상이한 속성들 및 품질 특징들을 가질 수 있다. 본 발명에서는, 사진 코딩 유형에 기초하여 $lumaDQP$ 설정에 적응하는 것이 제안된다. 따라서, 2개의 dQP 매핑 표가 인코더 입력 구성에 구성가능해지고, 이들은 $dQP_{inpIntra}$ 및 $dQP_{inpInter}$ 로 나타내진다.

[0282] 전술된 바와 같이, 루프-내 인트라 정형 방법을 사용할 때는, 정형이 인터 사진들 상에서는 수행되지 않기 때문에, 인트라 사진에 대해 사용되는 것과 같은 정형기에 의해 인터 사진들이 정형될 때와 비슷한 품질을 달성하기 위해, 몇몇의 $lumaDQP$ 설정이 인터-코딩된 사진들에 적용되는 것이 중요하다. 일 실시예에서, 인터-사진들에 대한 $lumaDQP$ 설정은 인트라 사진들에 사용되는 정형 곡선의 특징들에 매칭해야 한다.

[0283] [수학식 6]

$$Slope(x) = Fwd'(x) = (Fwd(x+dx) - Fwd(x-dx)) / (2dx)$$

[0284]

[0285] 은 순방향 정형 함수의 제1 도함수를 나타내고, 그리고, 실시예에서, 자동으로 도출된 $dQP_{auto}(x)$ 값들은 다음과 같이 컴퓨팅될 수 있는데:

[0286] $Slope(x) = 0$ 이면, $dQP_{auto}(x) = 0$ 이고, 그렇지 않으면

[0287] [수학식 7]

$$dQP_{auto}(x) = 6 \log_2(Slope(x))$$

[0288]

[0289] 인데, 여기서 $dQP_{auto}(x)$ 는 적합한 범위, 예를 들어, $[-6, 6]$ 내로 클리핑(clipping)될 수 있다.

[0290] $lumaDQP$ 가 정형된 인트라 사진들에 대하여 사용가능해지면(즉, 외부 $dQP_{inpIntra}$ 가 설정되면), 인터-사진들에 대한 $lumaDQP$ 는 그것을 고려해야만 한다. 실시예에서, 최종 인터 dQP_{final} 은 정형기로부터 도출된 dQP_{auto} (수학식 7)와 인트라 사진들에 대한 $dQP_{inpIntra}$ 설정을 합함으로써 컴퓨팅될 수 있다. 또다른 실시예에서는, 인트라 품질 전파의 이점을 취하기 위해, 인터-사진들에 대한 dQP_{final} 은 dQP_{auto} 에 설정되거나, ($dQP_{inpInter}$ 를 설정함으로써) 단지 작은 증분에서 dQP_{auto} 에 가산될 수 있다.

[0291] 실시예에서, 정형이 사용가능하면, 루마 dQP 값들을 설정하는 것에 대한 다음의 일반적인 규칙들이 적용될 수 있다;

[0292] (1) 루마 dQP 매핑 표들이 (사진 코딩 유형에 기초하여) 인트라 및 인터 사진들에 대해 개별적으로 설정될 수 있음.

[0293] (2) 코딩 루프 내의 사진이 정형된 도메인 내에 있으면(예를 들어, 루프-내 인트라 정형 구조 내의 인트라 사진들 또는 루프-밖 정형 구조 내의 모든 사진), 입력 루마 대 델타 QP 매핑(dQP_{inp})은 또한 정형된 도메인(dQP_{rsp})으로 변환될 필요가 있다. 즉,

[0294] [수학식 8]

$$dQP_{rsp}(x) = dQP_{inp}[Inv(x)]$$

[0296] 임.

[0297] (3) 코딩 루프 내의 사진이 미-정형된 도메인 내에 있으면(예를 들어, 역방향-정형되거나 미정형된, 예를 들어, 루프-내 인트라 정형 구조 내의 인터 사진들 또는 루프-내 잔차 정형 구조 내의 모든 사진), 입력 루마 대 델타 QP 매핑은 변환될 필요없이 바로 사용될 수 있음.

[0298] (4) 자동적인 인터 델타QP 파생은 루프-내 인트라 정형 구조에 대해서만 유효하다. 이러한 경우 인터 사진들에 대해 사용되는 실제 델타 QP는 자동-도출된 것과 입력의 합산된 값이고:

[0299] [수학식 9]

$$dQP_{final}[x] = dQP_{inp}[x] + dQP_{auto}[x]$$

[0301] $dQP_{final}[x]$ 는 적합한 범위, 예를 들어, [-12 12]로 클리핑됨;

[0302] (5) 루마 대 dQP 매핑 표는 매 사진마다 또는 정형 LUT에 변화가 있을 때 갱신될 수 있다. 실제 dQP 적응은 (블록의 평균 루마 값으로부터, 이 블록의 양자화를 위한 대응하는 dQP 를 얻음) CU 수준에서 일어날 수 있다(인코더 구성가능).

[0303] 표 13은 3가지 제안된 구조의 각각에 대한 dQP 설정을 요약한다.

【표 13: dQP 설정】

구조	루프-밖	루프-내 인트라 전용	루프-내 잔차
인트라 사진dQP	$dQP_{final}(x) = dQP_{rsp}(x) = dQP_{intra}(Inv(x))$	$dQP_{final}(x) = dQP_{rsp}(x) = dQP_{intra}(Inv(x))$	$dQP_{final}(x) = dQP_{intra}(x)$
인터 사진dQP	$dQP_{final}(x) = dQP_{rsp}(x) = dQP_{intra}(Inv(x))$	$dQP_{final}(x) = dQP_{auto}(x) + dQP_{intra}(x)$	$dQP_{final}(x) = dQP_{intra}(x)$

[0304] [0305] 레이트 왜곡 최적화(Rate Distortion Optimization: RDO)

[0306] JEM6.0 소프트웨어에서(참고 문헌 [8]), RDO 화소-기반 가중 왜곡은 lumaDQP가 사용가능할 때 사용된다. 가중치 표는 휘도 값들에 기초하여 고정된다. 실시예에서, 가중치 표는 lumaDQP 설정에 기초하여 적응적으로 조정되어야 한다(이전 섹션에서 제안된 바와 같이 컴퓨팅됨). 제곱 오차의 합(SSE) 및 절대차들의 합(SAD)에 대한 2개의 가중치들이 다음과 같이 제안된다:

[0307] [수학식 10a]

$$weight_SSE[x] = 2^{\left(\frac{dQP[x]}{3}\right)}$$

[0309] [수학식 10b]

$$weight_SAD[x] = 2^{\left(\frac{dQP[x]}{6}\right)}$$

[0311] 수학식 10a 또는 수학식 10b에 의해 컴퓨팅된 가중치는, 입력 lumaDQP 및 순방향 정형 함수로부터 유도된 dQP 모두를 포함하는, 최종 dQP에 기초한 총 가중치이다. 예를 들어, 수학식 9에 기초하여, 수학식 10a는 다음과 같이 쓰일 수 있다,

$$\begin{aligned} weight_{SSE}[x] &= 2^{\left(\frac{dQP_{final}[x]}{3}\right)} = 2^{\left(\frac{dQP_{inp}[x] + dQP_{auto}[x]}{3}\right)} \\ &= 2^{\left(\frac{dQP_{inp}[x]}{3}\right)} * 2^{\left(\frac{dQP_{auto}[x]}{3}\right)}. \end{aligned}$$

총 가중치는 입력 lumaDQP에 의해 컴퓨팅된 가중치에 의해:

$$weight_{SSE}[x]_{inp} = 2^{\left(\frac{dQP_{inp}[x]}{3}\right)}$$

정형으로부터의 가중치에 의해:

$$weight_{SSE}[x]_{reshape} = 2^{\left(\frac{dQP_{auto}[x]}{3}\right)} = 2^{\left(\frac{6\log_2(Slope(x))}{3}\right)} = (Slope(x))^2$$

나눌 수 있다.

총 가중치는 우선 정형으로부터의 가중치를 컴퓨팅함으로써 총 dQP를 사용하여 컴퓨팅되면, 그것은 정수 dQP_{auto} 를 얻기 위한 클리핑 연산에 의해 정확도를 잃게 된다. 대신, 정형으로부터의 가중치를 계산하기 위해 기울기 함수를 바로 사용하면 가중치의 더욱 높은 정확도를 지킬 수 있기에 보다 선호될 수 있다.

입력 lumaDQP로부터 도출된 가중치를 W_{dqp} 로 나타내자. $f'(x)$ 를 순방향 정형 곡선의 제1 도함수(또는 기울기)라고 나타내자. 실시예에서, 총 가중치는 dQP 값들 및 정형 곡선의 모양 모두를 고려하므로, 총 가중치 값은 다음과 같이 표현될 수 있다:

[수학식 11]

$$weight_{total} = Clip3(0.0, 30.0, W_{dqp} * f'(x)^2)$$

유사한 접근법이 크로마 구성요소들에도 또한 적용될 수 있다. 예를 들어, 실시예에서, 크로마에 대해, $dQP[x]$ 가 표 13에 따라 정의될 수 있다.

다른 코딩 도구들과의 상호작용

정형이 사용가능하면, 이 섹션은 다른 코딩 도구들에서 필요한 제안된 변화들에 대한 몇몇의 예를 제공한다. 임의의 가능한 기존의 코딩 도구 또는 (차세대 비디오 코딩 표준에 포함될) 미래의 코딩 도구에 대한 상호작용들이 존재할 수 있다. 아래에 제공된 예들은 본 발명을 제한하는 것은 아니다. 일반적으로, 코딩 단계들 동안의 (정형된, 미-정형된, 역방향-정형된) 비디오 신호 도메인이 식별될 필요가 있고, 각각의 단계에서 비디오 신호를 처리하는 연산들이 정형 영향을 고려할 필요가 있다.

교차-구성요소 선형 모델 예측

교차-구성요소 선형 모델 예측(cross-component linear model prediction; CCLM)(참고 문헌 [8])에서, 예측 크로마 샘플들 $pred_c(i, j)$ 은 루마 재구성 신호 $rec_L'(i, j)$ 를 사용하여 도출될 수 있다:

[수학식 12]

$$pred_c(i, j) = \alpha \cdot rec_L'(i, j) + \beta$$

정형이 사용가능하면, 실시예에서, 루마 재구성된 신호가 정형된 도메인에 있는지(예를 들어, 루프-밖 정형기 또는 루프-내 인트라 정형기) 또는 미-정형된 도메인에 있는지(예를 들어, 루프-내 잔차 정형기)를 구별할 필요가 있을 수 있다. 일 실시예에서, 재구성 루마 신호는 어떠한 추가적인 시그널링 또는 연산없이 있는 그대로 암묵적으로 사용될 수 있다. 다른 실시예들에서, 재구성된 신호가 미-정형된 도메인에 있으면, 재구성 루마 신호를 미-정형된 도메인에 또한 있도록 다음에서와 같이 변환할 수 있다:

[0330] [수학식 13]

$$pred_c(i, j) = \alpha \cdot Inv(rec_L'(i, j)) + \beta$$

[0332] 다른 실시예들에서, RDO 처리에 의해 결정될 수 있는, 어느 도메인이 바람직한지(정형된 또는 미-정형된)를 시그널링하기 위해 비트스트림 선택스 요소가 추가되거나, 디코딩된 정보에 기초하여 결정이 도출될 수 있는데, 이에 따라, 명백한 시그널링에 의해 필요로되는 오버헤드를 피할 수 있다. 결정에 기초하여 재구성된 신호에 대응하는 연산이 수행될 수 있다.

[0333] 잔차 예측 도구를 갖는 정형기

[0334] HEVC 범위 확장 프로파일에서, 잔차 예측 도구가 포함된다. 크로마 잔차 신호는 다음과 같이 인코더 측에서 루마 잔차 신호로부터 예측되고:

[0335] [수학식 14]

$$*_\Delta r_c(x, y) = r_c(x, y) - (\alpha \times r'_L(x, y)) \gg 3,$$

[0338] 그것은 다음과 같이 디코더 측에서 보상되는데:

[0339] [수학식 15]

$$*_r'_c(x, y) = \Delta r'_c(x, y) + (\alpha \times r'_L(x, y)) \gg 3,$$

[0342] 여기서 r_c 는 위치(x,y)에서의 크로마 잔차 샘플을 나타내고, r'_L 는 루마 구성요소의 재구성된 잔차 샘플을 나타내고, Δr_c 는 인터-색 예측을 사용하는 예측 신호를 나타내고, $\Delta r'_c$ 는 Δr_c 를 코딩 및 디코딩한 후에 재구성된 신호를 나타내고, r'_c 는 재구성된 크로마 잔차를 나타낸다.

[0343] 정형이 사용가능하면, 크로마 잔차 예측을 위해 어느 루마 잔차를 사용할지를 고려해야할 필요가 있을 수 있다. 일 실시예에서, "잔차"는 있는 그대로 사용될 수 있다(정형기 구조에 기초하여 정형되거나 미-정형될 수 있음). 또다른 실시예에서는, 루마 잔차가 (미-정형된 도메인과 같은) 하나의 도메인에 있도록 강요되어 적절한 매핑들이 수행될 수 있다. 또다른 실시예에서는, 디코더에 의해 적합한 처리가 도출되거나, 전송된 바와 같이 명확하게 시그널링될 수 있다.

[0344] 적응형 클리핑을 갖는 정형기

[0345] 적응형 클리핑(참고 문헌 [8])은 콘텐츠 다이내믹들에 관련하여 본래의 데이터 범위를 시그널링하고, 클리핑이 일어나는 압축 작업흐름(예를 들어, 변형/양자화, 루프 필터링, 출력) 내의 각각의 단계에서, 고정된 클리핑 대신 적응형 클리핑을 행하기 위해 도입된 새로운 도구이다.

[0346] [수학식 16]

[0347] $T_{clip} = Clip_{BD}(T, bitdepth, C) = Clip3(min_c, max_c, T)$ 이라고 하고, 여기서 $x = Clip3(min, max, c)$ 는:

$$x = \begin{cases} min, c \leq min \text{ 일 때} \\ max, c \geq max \text{ 일 때} \\ c, \text{ 그 외} \end{cases}$$

[0348]

- [0349] 를 나타내고,
- [0350] ● C 는 구성요소 ID(전형적으로 Y, Cb, 또는 Cr)이고
- [0351] ● min_c 는 구성요소 ID C 에 대한 현재 슬라이스 내에서 사용되는 보다 낮은 클리핑 경계이고
- [0352] ● max_c 는 구성요소 ID C 에 대한 현재 슬라이스 내에서 사용되는 보다 높은 클리핑 경계이다.
- [0353] 정형이 사용가능하면, 실시예에서, 데이터 흐름이 현재 존재하고 있는 도메인을 알아내고 클리핑을 올바르게 수행할 필요가 있다. 예를 들어, 정형된 도메인 내에서 데이터를 클리핑하면, 본래의 클리핑 경계들은 정형된 도메인으로 변환될 필요가 있다:
- [0354] [수학식 17]
- [0355]
$$T_{clip} = Clip_{BD}(T, bitdepth, C) =$$
- [0356]
$$= Clip3(Fwd(min_c), Fwd(max_c), T) .$$
- [0357] 일반적으로, 각각의 클리핑 단계를 정형 구조에 관련하여 적절하게 처리할 필요가 있다.
- [0358] 정형기 및 루프 필터링
- [0359] HEVC 및 JEM 6.0 소프트웨어에서, ALF 및 SAO와 같은 루프 필터들은 재구성된 루마 샘플들 및 압축되지 않은 "본래의" 루마 샘플들을 사용하여 최적의 필터 매개변수들을 추정할 필요가 있다. 정형이 사용가능하면, 실시예에서, 필터 최적화가 수행되길 바래지는 도메인이 (명확하게 또는 암묵적으로) 특정될 수 있다. 일 실시예에서는, 정형된 도메인 상에서 필터 매개변수들이 추정될 수 있다(재구성이 정형된 도메인 내에 있을 때, 정형된 본래 대비(versus a reshaped original)). 다른 실시예에서는, 미-정형된 도메인 상에서 필터 매개변수들이 추정될 수 있다(재구성이 미-정형된 도메인 내에 있거나 역방향 정형될 때, 본래 대비(versus the original)).
- [0360] 예를 들어, 루프-내 정형 구조에 따라, 루프-내 필터 최적화(in-loop filter optimization; ILFOPT) 선택들 및 연산들이 표 14 및 15에 의해 기술될 수 있다.
- [0361] **【표 14: 루프-내 인트라 전용 정형 구조 및 루프-내 하이브리드 정형 내에서의 루프-내 필터링 최적화】**

	인트라-프레임	인터-프레임
선택 1: 정형된 도메인 내에서 루프 필터링(LF)을 추정 및 수행:	● 인코더 측: ○ LF 기준으로 이미 정형된 본래의 인트라 이미지를 사용 ○ LF 매개변수들을 추정하기 위해 (정형된 도메인 내의) 재구성된 인트라 이미지 및 LF 기준을 사용; LF 모듈 내 역방향 정형 모듈의 위치에 따라 2가지 경우(도 2c 내의 블록(265) 및 블록(270)): ■ 역방향 정형이 LF 전에 수행되면, 재구성된 인트라 이미지 상에 순방향 정형을 적용할 필요가 있음 ■ 역방향 정형이 LF 후에 수행되면, 재구성된 인트라 이미지를 바로 사용 ○ 정형된 도메인 내의 재구성된 인트라 이미지에 LF를 적용 ○ 전체 재구성된 인트라 이미지를 역방향 정형 ○ DPB 내에 저장 ● 디코더 측: ○ 정형된 도메인 내의 재구성된 인트라 이미지에 LF를 적용; LF 모듈 내 역방향 정형 모듈의 위치에 따라 2가지 경우(도 2d 내의 블록(265) 및 블록(270)): ■ 역방향 정형이 LF 전에 수행되면, LF를 적용하기 전에 재구성된 인트라 이미지를 순방향 정형할 필요가 있음 ■ 역방향 정형이 LF 후에 수행되면, 재구성된 인트라 이미지 상에 LF를 바로 적용 ○ 전체 재구성된 인트라 이미지를 역방향 정형 ○ DPB 내에 저장	● 인코더 측: ○ LF 기준으로서의 본래의 인터 이미지를 순방향 정형 ○ 재구성된 인터 이미지를 순방향 정형 ○ LF 매개변수들을 추정 ○ 정형된 도메인 내의 재구성된 인터 이미지에 LF를 적용 ○ 전체 재구성된 인터 이미지를 역방향 정형 ○ DPB 내에 저장 ● 디코더 측: ○ 재구성된 인터 이미지를 순방향 정형 ○ 정형된 도메인 내의 재구성된 인터 이미지에 LF를 적용 ○ 전체 재구성된 인터 이미지를 역방향 정형 ○ DPB 내에 저장

[0362]

선택 2: 미-정형된 도메인 내에서 LF를 추정 및 수행:	● 인코더 측: ○ 루프-내(in-loop) 정형이 본래의 이미지 버퍼 상에서 수행되면, LF 기준으로서의 본래의 이미지를 역방향 정형; 또는 LF 기준으로서 미-정형된 본래의 이미지를 가져옴 ○ 재구성된 인트라 이미지를 역방향 정형 ○ LF 매개변수들을 추정 ○ 전체 재구성된 이미지를 역방향-정형 ○ 재구성된 인트라 이미지에 LF를 적용 ○ DPB 내에 저장 ● 디코더 측: ○ 전체 재구성된 이미지를 역방향-정형 ○ 재구성된 인트라 이미지에 LF를 적용 ○ DPB 내에 저장	(정확히 전형적인 LF 작업 흐름) ● 인코더 측: ○ LF 매개변수들을 추정하기 위한 LF 기준으로서 재구성된 인터 이미지 및 본래의 이미지를 사용 ○ 재구성된 인터 이미지에 LF를 적용 ○ DPB 내에 저장 ● 디코더 측: ○ 재구성된 인터 이미지에 LF를 적용 ○ DPB 내에 저장
----------------------------------	---	--

[0363]

【표 15: 루프-내 잔차 정형 구조 내에서의 루프-내 필터링 최적화】

루프-내 잔차 정형	인트라-프레임 및 인터-프레임 (표 14의 인트라-프레임 연산들과 같음)
------------	---

[0364]

[0365]

여기서 대부분의 상세한 설명은 휘도 구성요소 상에서 수행되는 방법들을 언급하지만, 당업자는 유사한 방법들이 크로마 색 구성요소들 및 chromaQPoffset과 같은 크로마 관련 매개변수들에서 수행될 수 있다는 것을 인식할 것이다(예를 들어, 참고 문헌 [9]를 참조).

[0366] 루프-내 정형 및 관심 영역들(Regions of Interest; ROI)

[0367] 여기서 사용된 바와 같이, 이미지가 주어지면, '관심 영역'(ROI)이란 용어는 특별한 관심이 있다고 고려되는 이미지의 영역을 나타낸다. 이 섹션에서는, 오직 관심 영역에 대해서만 루프-내 정형을 지원하는 새로운 실시예들이 제시된다. 즉, 실시예에서, 정형은 오직 ROI 내에서만 적용되고 ROI 밖에서는 적용되지 않을 수 있다. 또다른 실시예에서, 관심 영역 내와 관심 영역 밖에서 상이한 정형 곡선들이 적용될 수 있다.

[0368] ROI의 사용은 비트 레이트와 이미지 품질의 균형을 맞추는 필요에 의해 동기 부여된다. 예를 들어, 일몰의 비디오 시퀀스를 고려하자. 이미지들의 상부-절반 상에는 비교적 균일한 색의 하늘 위에 태양이 있을 수 있다(따라서, 하늘 배경의 화소들은 매우 낮은 변화를 가질 수 있음). 반대로, 이미지의 하부-절반은 일렁이는 파도를 나타낼 수 있다. 관찰자의 관점에서, 상부가 하부보다 훨씬 더 중요하다고 고려될 수도 있다. 반면, 그들의 화소들 내의 보다 높은 변화로 인해, 일렁이는 파도는 압축하기 더욱 어렵고, 화소 당 더 많은 비트를 필요로 하지만; 파도 부분보다 태양-부분에 더욱 많은 비트가 할당되길 바라질 수 있다. 이 경우에, 상부 절반이 관심 영역을 의미할 수 있다.

[0369] ROI 설명

[0370] (예를 들어, AVC, HEVC 등과 같은) 요즈음 대부분의 코덱들은 블록 기반이다. 구현을 간단히 하기 위해, 영역은 블록 단위로 특정될 수 있다. 제한 없이, 예로서 HEVC를 사용하여, 영역은 다수의 코딩 유닛(Coding Unit; CU) 또는 코딩 트리 유닛(Coding Tree Unit; CTU)으로 정의될 수 있다. 하나의 ROI 또는 다수의 ROI들이 특정될 수 있다. 다수의 ROI들은 이산되거나 중첩될 수 있다. ROI는 사각형일 필요 없다. ROI에 대한 선택스는 슬라이스 수준, 사진 수준, 비디오 스트림 수준 등과 같은 임의의 관심 수준에서 제공될 수 있다. 실시예에서, ROI는 우선 시퀀스 매개변수 세트(SPS)에서 특정된다. 그 후, 슬라이스 헤더에서, ROI의 작은 변형들이 허용될 수 있다. 표 16은 하나의 ROI가 어디서 사각형 영역의 다수의 CTU들로 특정되는지의 선택스의 예를 나타낸다. 표 17은 슬라이스 수준의 수정된 ROI의 선택스를 기술한다.

【표 16: ROI에 대한 SPS 선택스】

SPS() {	기술자
...	
sps_resaper_enable_flag	u(1)
if (sps_resaper_enable_flag) {	
...	
sps_resaper_active_ROI_flag	u(1)
if (sps_resaper_active_ROI_flag) {	
resaper_active ROI in CTUsize left	ue(v)
resaper_active ROI in CTUsize right	ue(v)
resaper_active ROI in CTUsize top	ue(v)
resaper_active ROI in CTUsize bottom	ue(v)
}	
}	
...	
}	

[0371]

【표 17: ROI에 대한 슬라이스 헤더 선택스】

reshaping_sliceheader table() {	기술자
...	
if (sps_reshaper_active_ROI_flag) {	
reshape_model_ROI_modification_flag	u(1)
if (reshape_model_ROI_modification_flag) {	
reshaper_ROI_mod_offset_left	se(v)
reshaper_ROI_mod_offset_right	se(v)
reshaper_ROI_mod_offset_top	se(v)
reshaper_ROI_mod_offset_bottom	se(v)
}	
}	
...	
}	

[0372]

[0373]

sps_reshaper_active_ROI_flag가 1인 것은, ROI가 코딩된 비디오 시퀀스(coded video sequence; CVS) 내에 존재한다는 것을 특정한다. sps_reshaper_active_ROI_flag가 0인 것은, ROI가 CVS 내에 존재하지 않는다는 것을 특정한다.

[0374]

reshaper_active_ROI_in_CTUsize_left, reshaper_active_ROI_in_CTUsize_right, reshaper_active_ROI_in_CTUsize_top, 및 reshaper_active_ROI_in_CTUsize_bottom 각각은 사진 좌표로 특정된 사각형 영역에 관련하여, ROI 내의 사진들의 샘플을 특정한다. 좌표들은 좌측 및 상부에 대해서 offset*CTUsize이고, 우측 및 하부에 대해서 offset*CTUsize-1이다.

[0375]

reshape_model_ROI_modification_flag가 1인 것은, ROI가 현재의 슬라이스 내에서 수정되었다는 것을 특정한다. reshape_model_ROI_modification_flag가 0인 것은, ROI가 현재의 슬라이스 내에서 수정되지 않았다는 것을 특정한다.

[0376]

reshaper_ROI_mod_offset_left, reshaper_ROI_mod_offset_right, reshaper_ROI_mod_offset_top, 및 reshaper_ROI_mod_offset_bottom 각각은 reshaper_active_ROI_in_CTUsize_left, reshaper_active_ROI_in_CTUsize_right, reshaper_active_ROI_in_CTUsize_top, 및 reshaper_active_ROI_in_CTUsize_bottom으로부터의 좌/우/상/하의 오프셋 값을 특정한다.

[0377]

다수의 ROI들을 위하여, 단일 ROI에 대한 표 16 및 17의 예시적인 선택스는 각각의 ROI에 대한 색인(또는 ID)를 사용하여, HEVC에서 사용되는 방식과 유사하게, SEI 메시지를 사용하여 다수의 팬-스캔 사각형들을 정의하도록 확장될 수 있다(HEVC 명세서, 참고 문헌 [11], 섹션 D.2.4를 참조).

[0378]

루프-내 인트라-전용 정형에서의 ROI들의 처리

[0379]

인트라-전용 정형에 대해서, 사진의 ROI 부분이 우선 정형되고, 그 후 코딩이 적용된다. 정형이 ROI에만 적용되기 때문에, 사진의 ROI와 비-ROI 간의 경계가 보일 수 있다. 루프 필터(예를 들어, 도 2c 또는 도 2d의 270)가 경계를 거쳐갈 수 있기 때문에, 루프 필터 최적화(ILFOPT)를 위해 특별한 관심이 ROI들에 취해져야만 한다. 실시예에서, 전체 디코딩된 사진이 같은 도메인에 있는 곳에만 루프 필터가 적용되는 것이 제안된다. 즉, 전체 사진은 모두가 정형된 도메인에 있거나 모두가 미-정형된 도메인에 있다. 일 실시예에서, 디코더 측에서, 루프 필터링이 미-정형된 도메인 상에 적용되면, 우선 디코딩된 사진의 ROI 섹션에 역방향 정형이 적용되고, 그 후 루프 필터가 적용되어야 한다. 다음으로, 디코딩된 사진이 DPB에 저장된다. 또다른 실시예에서, 루프 필터가 정형된 도메인 상에 적용되면, 우선 디코딩된 사진의 비-ROI 부분에 정형이 적용되고, 그 후 루프 필터가 적용되고, 그 리고 나서 전체 사진이 역방향 정형되어야 한다. 다음으로, 디코딩된 사진이 DPB에 저장된다. 또한 또다른 실시예에서, 루프 필터링이 정형된 도메인 상에 적용되면, 우선 디코딩된 사진의 ROI 부분이 역방향-정형되고, 그 후 전체 사진이 정형되고, 그리고 나서 루프 필터가 적용되고, 그 후 전체 사진이 역방향 정형될 수 있다. 다음으로, 디코딩된 사진이 DPB에 저장된다. 3가지 접근법이 표 18에 요약되어 있다. 컴퓨팅 관점에서, 방법 "A"가 보다 간단하다. 실시예에서, ROI를 사용가능하게 하는 것은 역방향 정형 대 루프 필터링(LF)을 수행하는 순서를 특정하는데 사용될 수 있다. 예를 들어, ROI가 적극적으로 사용되면(예를 들어, SPS 선택스 플래그 = 참), LF (도 2c 및 도 2d 내의 블록(270))는 역방향 정형(도 2c 및 도 2d의 블록(265)) 후에 수행된다. ROI가 적극적으로 사용되지 않으면, LF는 역방향 정형 전에 수행된다.

【표 18: ROI들을 사용하는 루프-필터링(LF) 선택들】

방법 A	방법 B	방법 C
● 디코딩된 사진의 ROI를 역방향-정형 ● 전체 사진에 루프 필터링(LF)을 적용 ● DPB에 저장	● 디코딩된 사진의 비-ROI 부분에 정형을 적용 ● 전체 사진에 LF를 적용 ● 전체 사진을 역방향-정형 ● DPB에 저장	● 디코딩된 사진의 ROI를 역방향-정형 ● 전체 사진을 정형 ● 전체 사진에 LF를 적용 ● 전체 사진을 역방향-정형 ● DPB에 저장

루프-내 예측 잔차 정형에서의 ROI들의 처리

루프-내 (예측) 잔차 정형 구조(예를 들어, 도 2f의 200C_D 참조)에 대해서, 디코더에서, 수학적 3을 사용하여, 처리는 다음과 같이 표현될 수 있다:

If (currentCTU가 ROI에 속함)

$Reco_sample = Inv(Res_d + Fwd(Pred_sample))$, (수학적 3을 참조)

else

$Reco_sample = Res_d + Pred_sample$

end

ROI들 및 인코더 고려사항들

인코더에서, 각각의 CTU는 그것이 ROI에 속하는지 아닌지에 대해서 확인될 필요가 있다. 예를 들어, 루프-내 예측 잔차 정형에 대해서, 수학적 3에 기초한 간단한 확인이 수행될 수 있다:

If (currentCTU가 ROI에 속함)

루마에 대한 RDO에 가중된 왜곡을 적용. 가중은 수학적 10에 기초하여 도출됨.

else

루마에 대한 RDO에 비-가중된 왜곡을 적용

end

정형 동안 ROI들을 고려하는 예시적인 인코딩 작업흐름은 다음의 단계들을 포함할 수 있다:

- 인트라 사진에 대해:

- 본래 사진의 ROI 영역에 순방향 정형을 적용

- 인트라 프레임을 코딩

- 루프 필터(LF) 전에, 재구성된 사진의 ROI 영역에 역방향 정형을 적용

- 다음과 같이 미정형된 도메인에서 루프 필터링을 수행(예를 들어, 표 18의 방법 "C"를 참조), 이는 다음의 단계들을 포함함:

● (루프 필터 기준에 대해 전체 본래 사진을 정형시키기 위해) 본래의 사진의 비-ROI 영역에 순방향 정형을 적용

● 재구성된 사진의 전체 사진 영역에 순방향 정형을 적용

● 루프 필터 매개변수들을 도출하여 루프 필터링을 적용

● 재구성된 사진의 전체 사진 영역에 역방향 정형을 적용하고 DPB에 저장.

인코더 측에서, LF는 필터 매개변수 추정을 위한 미-압축된 기준 이미지를 가질 필요가 있으므로, 각각의 방법에 대한 LF 기준의 처리는 표 19에서와 같다:

【표 19: ROI들에 대한 LF 기준의 처리】

방법 A	방법 B	방법 C
● LF 기준을 위해 미-정형된 본래의 입력 사진을 사용;	● LF 기준을 위해 본래의 입력 사진 전체(비-ROI 및 ROI 부분 모두) 상에 정형을 적용;	● LF 기준을 위해 본래의 입력 사진 전체(비-ROI 및 ROI 부분 모두) 상에 정형을 적용;

- 인트라 사진에 대해:

- 인트라 프레임을 코딩할 때, ROI 내부의 각각의 CU에 대해서는, 예측 잔차 정형 및 루마에 대한 가중된 왜곡을 적용하고; ROI 외부의 각각의 CU에 대해서는, 어떠한 정형도 적용하지 않음

- 상기에서처럼(ROI가 사용되지 않음), 루프 필터링 최적화(선택 1)가 수행됨:

● 본래 사진의 전체 사진 영역을 순방향 정형

● 재구성된 사진의 전체 사진 영역을 순방향 정형

● 루프 필터 매개변수들을 도출하고 루프 필터링을 적용

● 재구성된 사진의 전체 사진 영역에 역방향 정형을 적용하고 DPB에 저장.

HLG-코딩된 콘텐츠의 정형

하이브리드로그-감마(HybridLog-Gamma) 또는 HLG란 용어는 높은-다이내믹 범위 신호들을 매핑하기 위한 Rec.BT.2100에 정의된 또다른 전달 함수를 나타낸다. HLG는 전형적인 감마 함수를 사용하여 코딩된 전형적인 표준 다이내믹 범위 신호와의 백워드 호환성(backward compatibility)을 유지하기 위해 개발되었다. PQ-코딩된 콘텐츠와 HLG-코딩된 콘텐츠 간에 코드워드 분포를 비교하면, PQ 매핑은 어두운 영역과 밝은 영역에 더 많은 코드워드들을 할당하는 경향이 있는 반면, 대부분의 HLG 콘텐츠 코드워드들은 중간 범위에 할당되는 것으로 보인다. 두 접근법은 HLG 루마 정형을 위해 사용될 수 있다. 일 실시예에서, 간단히 HLG 콘텐츠를 PQ 콘텐츠로 변환시킨 후, 전술된 모든 PQ-관련 정형 기술들을 적용할 수 있다. 예를 들어, 다음의 단계들이 적용될 수 있다:

1) HLG 휘도(예를 들어, Y)를 PQ 휘도로 매핑. 변형에 대한 함수 또는 LUT를 $HLG2PQLUT(Y)$ 로 나타냄

2) PQ 휘도 값들을 분석하여 PQ-기반 순방향 정형 함수 또는 LUT를 도출. 그것을 $PQAdpFLUT(Y)$ 로 나타냄

3) 2개의 함수들 또는 LUT들을 단일의 함수 또는 LUT로 합침: $HLGAdpFLUT[i] = PQAdpFLUT[HLG2PQLUT[i]]$.

HLG 코드워드 분포가 PQ 코드워드 분포와 상당히 상이하기 때문에, 이러한 접근법은 최적이지 않은 정형 결과를 낸다. 또다른 실시예에서는, HLG 정형 함수가 HLG 샘플들로부터 바로 도출된다. PQ 신호들을 위해 사용되는 것과 같은 프레임워크가 적용될 수 있지만, CW_Bins_Dft 표가 HLG 신호의 특징들을 반영하도록 바뀔 수 있다. 실시예에서, HLG 신호들에 대한 중간-톤 프로파일을 사용하여, 몇몇의 CW_Bins_Dft 표가 사용자-선택에 따라 디자인된다. 예를 들어, 밝음을 보존하는 것이 바람직할 때, 알파 = 1.4에 대하여,

$g_DftHLGCWBin0 = \{ 8, 14, 17, 19, 21, 23, 24, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 36, 37, 38, 39, 39, 40, 41, 41, 42, 43, 43, 44, 44, 30 \}$.

중간-톤(또는 중간-범위)을 보존하는 것이 바람직할 때:

$g_DftHLGCWBin1 = \{ 12, 16, 16, 20, 24, 28, 32, 32, 32, 32, 36, 36, 40, 44, 48, 52, 56, 52, 48, 44, 40, 36, 36, 32, 32, 32, 26, 26, 20, 16, 16, 12 \}$.

피부 톤을 보존하는 것이 바람직할 때:

$g_DftHLGCWBin2 = \{ 12, 16, 16, 24, 28, 32, 56, 64, 64, 64, 64, 56, 48, 40, 32, 32, 32, 32, 32, 28, 28, 24, 24, 20, 20, 20, 20, 20, 16, 16, 12 \}$;

비트스트림 선택스 관점에서, PQ와 HLG-기반 정형을 구별하기 위해, **sps_reshaper_signal_type**로 나타내지는 새로운 매개변수가 추가되는데, 여기서 **sps_reshaper_signal_type**란 값은 정형된 신호 유형을 나타낸다(예를 들어, 감마-기반 SDR 신호들에 대해서는 0, PQ-코딩된 신호들에 대해서는 1, HLG-코딩된 신호들에 대해서는 2).

전술된 모든 특징들(예를 들어, ROI, 루프 내 필터 최적화(ILFOPT), 및 ChromaDQAdjustment)을 갖는, PQ 및

HLG 모두에 대한 SPS 및 슬라이스 헤더 내에서의 HDR 정형에 대한 선택스 표들의 예들이 표 20 및 21에 나타나 있다.

【표 20: 정형을 위한 예시적인 SPS 선택스】

SPS()	기술자
...	
sps_reshaper_enable_flag /*1: 정형 온, 그 외에는, 오프 */	u(1)
if (sps_reshaper_enable_flag) {	
sps_reshaper_adaptive_flag /* 1: 적응형 정형 온, 그외에는, 오프 */	u(1)
sps_reshaper_signal_type /* 예를 들어: 0: SDR, 1:PQ, 2: HLG */	u(2)
sps_in_loop_filter_opt_flag /* ILFOPT 플래그*/	u(1)
sps_luma_based_chroma_qp_offset_flag /*chromaDQPAjustment 플래그*/	u(1)
sps_reshaper_active_ROI_flag	u(1)
if (sps_reshaper_active_ROI_flag) {	
reshaper_active_ROI_in_CTUsize_left	ue(v)
reshaper_active_ROI_in_CTUsize_right	ue(v)
reshaper_active_ROI_in_CTUsize_top	ue(v)
reshaper_active_ROI_in_CTUsize_bottom	ue(v)
}	
}	

[0427]

[0428]

sps_in_loop_filter_opt_flag가 1인 것은, 루프-내 필터 최적화가 코딩된 비디오 시퀀스(CVS) 내의 정형된 도메인에서 수행된다는 것을 특징한다. **sps_in_loop_filter_opt_flag**가 0인 것은, 루프-내 필터 최적화가 CVS 내의 미-정형된 도메인에서 수행된다는 것을 특징한다. **sps_luma_based_chroma_qp_offset_flag**가 1인 것은, (예를 들어, 표 11 또는 12에 따라) 루마 기반 크로마 QP 오프셋이 도출되어, 코딩된 비디오 시퀀스(CVS) 내의 각각의 CU의 크로마 코딩에 적용된다는 것을 특징한다. **sps_luma_based_chroma_qp_offset_flag**가 0인 것은, 루마 기반 크로마 QP 오프셋이 CVS에서 사용가능하지 않다는 것을 특징한다.

【표 21: 슬라이스 수준에서의 정형에 대한 예시적인 선택스】

reshaping_sliceheader_table_model() {	기술자
reshape_model_profile_type	ue(v)
reshape_model_scale_idx	u(2)
reshape_model_min_bin_idx	u(5)
reshape_model_max_bin_idx	u(5)
reshape_model_num_band	u(4)
for (i = 0; i < reshape_model_num_band; i++) {	
reshape_model_band_profile_delta [i]	u(1)
}	
if (sps_reshaper_active_ROI_flag) {	
reshape_model_ROI_modification_flag	u(1)
if (reshape_model_ROI_modification_flag) {	
reshaper_ROI_mod_offset_left	se(v)
reshaper_ROI_mod_offset_right	se(v)
reshaper_ROI_mod_offset_top	se(v)
reshaper_ROI_mod_offset_bottom	se(v)
}	
}	
}	

[0429]

[0430]

크로마 품질 개선

[0431]

HLG-기반 코딩의 지지자들은, 그것이 SDR 시그널링과의 보다 양호한 백워드 호환성을 제공한다고 주장한다. 따라서, 이론적으로, HLG-기반 신호들은 레거시(legacy) SDR 신호들과 같은 인코딩 설정들을 사용할 수 있다. 그러나 HDR 모드에서 HLG-코딩된 신호들을 보면, 몇몇의 색 아티팩트(artifact)들이, 특히 (흰색 및 회색과 같은) 무채색 영역에서 여전히 관찰될 수 있다. 실시예에서, 이러한 아티팩트들은 인코딩하는 동안 chromaQPoffset 값들을 조정함으로써 줄어들 수 있다. HLG 콘텐츠에 대해서, PQ 신호들을 코딩할 때 사용되는 것보다 덜 공격적인 chromaQP 조정을 적용하는 것이 제안된다. 예를 들어, 참고 문헌 [10]에서, 캡처 및 표현 색 프라이어머리들

(colour primaries)에 기초하여 인자 및 루마 QP에 기초한 Cb 및 Cr에 대한 QP 오프셋을 할당하기 위한 모델이 다음과 같이 기술되는데:

[0432] [수학식 18a]

$$QP_{offsetCb} = Clip3(-12, 0, Round(c_{cb} * (k * QP + l)))$$

[0434] [수학식 18b]

$$QP_{offsetCr} = Clip3(-12, 0, Round(c_{cr} * (k * QP + l)))$$

[0436] 여기서, 캡처 색 프라임리들이 표현 색 프라임리들과 같으면, $c_{cb} = 1$ 이고, 캡처 색 프라임리들은 P3D65 프라임리들과 같고 표현 색 프라임리들은 Rec. ITU-R BT.2020 프라임리들과 같으면, $c_{cb} = 1.04$ 이고, 캡처 색 프라임리들은 Rec. ITU-R BT.709 프라임리들과 같고 표현 색 프라임리들은 Rec. ITU-R BT.2020 프라임리들과 같으면, $c_{cb} = 1.14$ 이다. 마찬가지로, 캡처 색 프라임리들이 표현 색 프라임리들과 같으면, $c_{cr} = 1$ 이고, 캡처 색 프라임리들은 P3D65 프라임리들과 같고 표현 색 프라임리들은 Rec. ITU-R BT.2020 프라임리들과 같으면, $c_{cb} = 1.39$ 이고, 캡처 색 프라임리들은 Rec. ITU-R BT.709 프라임리들과 같고 표현 색 프라임리들은 Rec. ITU-R BT.2020 프라임리들과 같으면, $c_{cr}=1.78$ 이다. 마지막으로, $k = -0.46$ 및 $l = 0.26$ 이다.

[0437] 실시예에서, 덜 공격적인 chromaQPoffset 변화를 내는 매개변수들이 상이하다는 것 외에는 같은 모델을 사용하는 것이 제안된다. 예를 들어, 제한 없이, 실시예에서, 수학식 18a에서의 Cb에 대하여, $c_{cb} = 1$, $k = -0.2$, 및 $l = 7$ 이고, 수학식 18b에서의 Cr에 대하여, $c_{cr} = 1$, $k = -0.2$, 및 $l = 7$ 이다. 도 6a 및 도 6b는 PQ(Rec. 709) 및 HLG에 대하여 루마 양자화 매개변수(QP)에 따라 chromaQPoffset 값들이 어떻게 변하는지를 나타낸다. PQ-관련 값들은 HLG-관련 값들보다 극적으로 변한다. 도 6a는 Cb에 대응하고(수학식 18a), 반면 도 6b는 Cr에 대응한다(수학식 18b).

[0438] 참고 문헌들

[0439] 여기에 나열된 참고 문헌들 각각은 그 전체가 참조에 의해 통합된다.

[0440] [1] PCT 특허 출원 제PCT/US2016/025082호, *높은 다이내믹 범위 비디오 코딩 내에서의 루프-내 블럭-기반 이미지 정형(In-Loop Block-Based Image Reshaping in High Dynamic Range Video Coding)*, 쥐-엠. 수(G-M. Su)에 의해, 2016년 3월 30일자 출원되었고, 또한 제W02016/164235호로 공개됨.

[0441] [2] 디. 바이론(D. Baylon), 지. 구(Z. Gu), 에이. 루스라(A. Luthra), 케이. 미누(K. Minoo), 피. 인(P. Yin), 에프. 푸(F. Pu), 티. 루(T. Lu), 티. 첸(T. Chen), 더블유. 후삭(W. Husak), 와이. 허(Y. He), 엘. 케로프스키(L. Kerofsky), 와이. 예(Y. Ye), 비. 이(B. Yi), "HDR 및 WCG 비디오 코딩에 대한 증거를 위한 호출에의 응답:아리스, 돌비 및 인터디지탈(Response to Call for Evidence for HDR and WCG Video Coding: Arris, Dolby and InterDigital)", 문서 번호 m36264, 2015년 6월, 폴란드, 바르샤바.

[0442] [3] 미국 특허 출원 번호 제15/410,563호, *높은 코드워드 표현 이미지들을 위한 콘텐츠-적응형 정형(Content-Adaptive Reshaping for High Codeword representation Images)*, 티. 루(T. Lu) 등에 의해, 2017년 1월 19일자 출원됨.

[0443] [4] PCT 특허 출원 제PCT/US2016/042229호, *HDR 및 광범위한 색 영역 신호들에 대한 신호 정형 및 코딩(Signal Reshaping and Coding for HDR and Wide color Gamut Signals)*, 피. 인 등에 의해, 2016년 7월 14일자 출원되었고, 또한 제W02017/011636호로 공개됨.

[0444] [5] "HEVC의 HDR 확장을 위한 탐색 시험 모델(Exploratory Test Model for HDR extension of HEVC)", 케이. 미누 등, MPEG 발행 문서, JCTVC-W0092 (m37732), 2016년, 미국, 샌 디에고.

[0445] [6] 이. 프란체스코(E. François), 제이. 솔(J. Sole), 제이. 스트롬(J. Ström), 피. 인, "HDR/WCG 비디오 코딩 실험들을 위한 공통 시험 조건들(Common Test Conditions for HDR/WCG video coding experiments)",

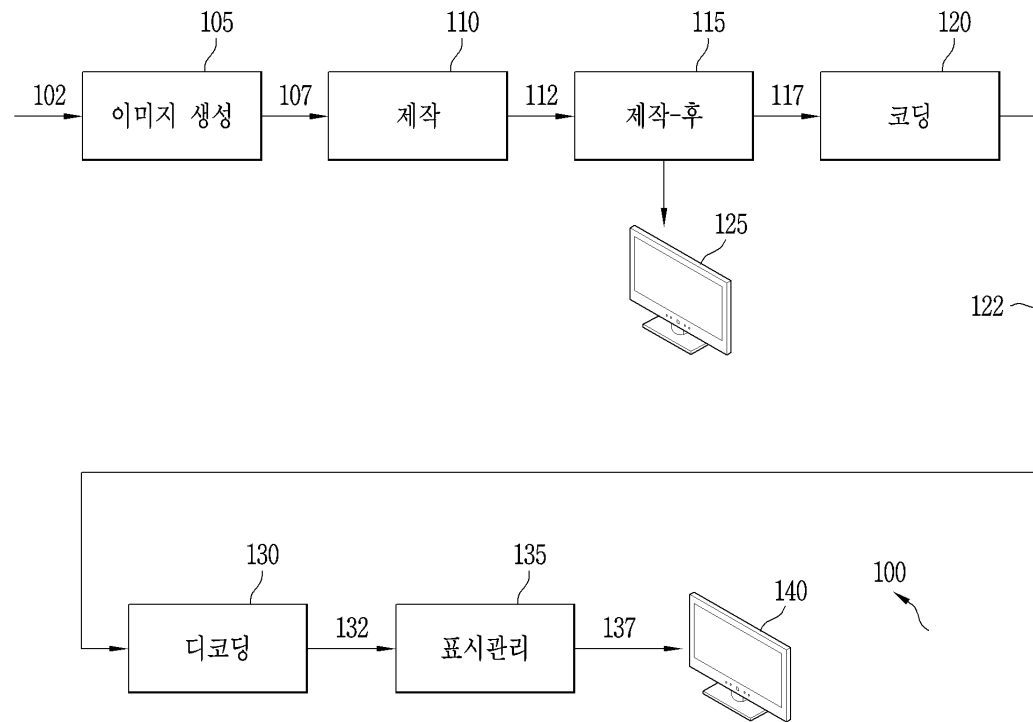
JCTVC 문서. Z1020, 제네바, 2017년 1월.

- [0446] [7] 에이. 세갈(A. Segal), 이. 프란체스코, 및 디. 루자노브스키(D. Rusanovsky), "HDR/WCG 비디오를 위한 JVET 공통 시험 조건들 및 평가 절차들(JVET common test conditions and evaluation procedures for HDR/WCG Video)", JVET-E1020, ITU-T 회의, 제네바, 2017년 1월.
- [0447] [8] JEM 6.0 소프트웨어: https://jvet.hhi.fraunhofer.de/svn/svn_HMJEMSoftware/tags/HM-16.6-JEM-6.0
- [0448] [9] 2016년 10월 11일자 출원된, 미국 가특허출원 번호 제62/406,483호, "*다수의 색 이미징 포맷들을 위한 비디오 코딩 내에서의 적응형 크로마 양자화(Adaptive Chroma quantization in Video Coding for Multiple color Imaging Formats)*", 티. 루 등, 또한 미국 특허 출원 번호 제15/728,939호로 제출되고, 미국 특허 출원 공개 번호 제US 2018/0103253호로도 공개됨.
- [0449] [10] 제이. 사무엘슨(J. Samuelsson) 등 (Eds), "PQ 전환 특징들을 갖는 HDR/WCG Y'CbCr 4:2:0 비디오를 위한 변환 및 코딩 실행(Conversion and coding practices for HDR/WCG Y'CbCr 4:2:0 Video with PQ transfer Characteristics)" JCTVC-Y1017, ITU-T/ISO 회의, 청두(Chengdu), 2016년 10월.
- [0450] [11] ITU-T H.265, "*고 효율 비디오 코딩(High efficiency video coding)*" ITU, 버전 4.0, (2016년 12월).
- [0451] 예시적인 컴퓨터 시스템 구현
- [0452] 본 발명의 실시예들은 컴퓨터 시스템, 전자 회로 및 구성요소들로 구성된 시스템들, 마이크로제어기와 같은 집적 회로(IC) 장치, 필드 프로그래밍가능 게이트 어레이(field programmable gate array; FPGA), 또는 또다른 구성가능하거나 프로그래밍가능한 논리 장치(programmable logic device; PLD), 이산적인 시간 또는 디지털 신호 처리기(digital signal processor; DSP), 응용 주문형 IC(application specific IC; ASIC), 및/또는 하나 이상의 이러한 시스템들, 장치들, 또는 구성요소들을 포함하는 장치로 구현될 수 있다. 컴퓨터 및/또는 IC는, 여기에 기술된 것들과 같은, 이미지들의 통합된 신호 정형 및 코딩에 관한 명령어들을 수행, 제어, 또는 실행할 수 있다. 컴퓨터 및/또는 IC는 여기에 설명된 신호 정형 및 코딩 처리들에 관한 다양한 매개변수들 또는 값들 중 임의의 것을 컴퓨팅할 수 있다. 이미지 및 비디오 실시예들은 하드웨어, 소프트웨어, 펌웨어, 및 그들의 다양한 조합으로 구현될 수 있다.
- [0453] 본 발명의 어떤 구현들은, 처리기들로 하여금 발명의 방법을 수행하게 하는 소프트웨어 명령어들을 실행하게 하는 컴퓨터 처리기들을 포함한다. 예를 들어, 표시장치, 인코더, 셋 톱 박스, 트랜스코더(transcoder) 등 내의 하나 이상의 처리기들은, 처리기들에 의해 액세스가능한 프로그램 메모리 내의 소프트웨어 명령어들을 실행함으로써, 전송된 바와 같은 이미지들의 통합된 신호 정형 및 코딩에 관한 방법들을 구현할 수 있다. 본 발명은 또한 프로그램 제품의 형태로 제공될 수도 있다. 프로그램 제품은, 데이터 처리기에 의해 실행될 때, 데이터 처리기로 하여금 본 발명의 방법을 실행하게 하는, 명령어들을 포함하는 컴퓨터-판독가능 신호들의 세트를 전달하는 임의의 비-일시적 매체를 포함할 수 있다. 본 발명에 따른 프로그램 제품은 매우 다양한 형태들 중 임의의 것일 수 있다. 프로그램 제품은, 예를 들어, 플로피 디스켓, 하드 디스크 드라이브를 포함하는 자기 데이터 저장 매체, CD ROM, DVD를 포함하는 광학 데이터 저장 매체, ROM, 플래시 RAM을 포함하는 전자 데이터 저장 매체 등과 같은 물리적인 매체를 포함할 수 있다. 프로그램 제품 상의 컴퓨터-판독가능 신호들은 선택적으로 압축 또는 암호화될 수 있다.
- [0454] 앞에서 구성요소(예를 들어, 소프트웨어 모듈, 처리기, 어셈블리(assembly), 장치, 회로 등)가 언급된 곳에서는, 이와 달리 언급되지 않는 한, ("수단"에 대한 언급을 포함하여) 그 구성요소에 대한 언급은, 그 구성요소에 대한 등가물들로서, 본 발명의 제시된 예시적인 실시예들 내의 기능을 수행하지만 개시된 구조와는 구조적으로 등가가 아닌 구성요소들을 포함하는, 그 기술된 구성요소의 기능을 수행하는 (예를 들어, 기능적으로 등가인) 임의의 구성요소를 포함하는 것으로 해석되어야 한다.
- [0455] 등가물, 확장물, 대안물 및 기타
- [0456] 이미지들의 효율적인 통합된 신호 정형 및 코딩에 관한 예시적인 실시예들이 이와 같이 기술된다. 앞의 명세서에서, 본 발명의 실시예들은 구현에 따라 다양할 수 있는 여러가지 구체적인 세부사항들을 참조하여 기술됐다. 따라서, 본 발명이 무엇인지 및 출원인이 본 발명이라고 의도하는 것이 무엇인지에 대한 단독의 및 독점적인 표시는 본 출원에 나타난 청구항들의 세트이고, 이들은 이러한 청구항들이 나타내는 특정한 형태를 가지며, 임의의 후속하는 정정을 포함한다. 이러한 청구항들에 포함된 용어들에 대해, 여기에 명확히 기술된 임의의 정의는 청구항들에 사용된 것과 같은 이러한 용어들의 의미를 좌우할 것이다. 따라서, 청구항에 명확히 언급되지 않은 어

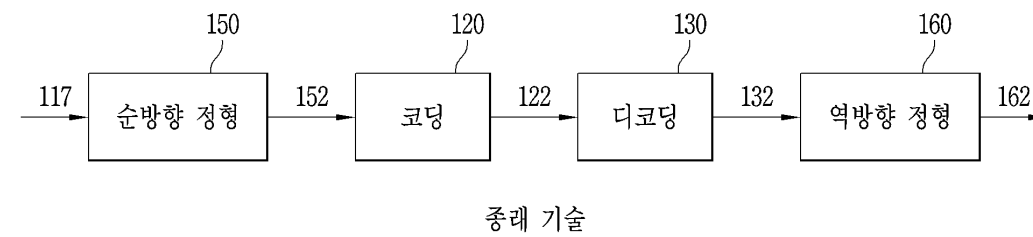
면 제한, 요소, 속성, 특징, 이점, 또는 속성은 어떤 방식으로든 이러한 청구항의 범위를 제한해서는 안된다. 따라서, 명세서 및 도면들은 제한하는 의미보다는 예시적인 것으로 간주되어야한다.

도면

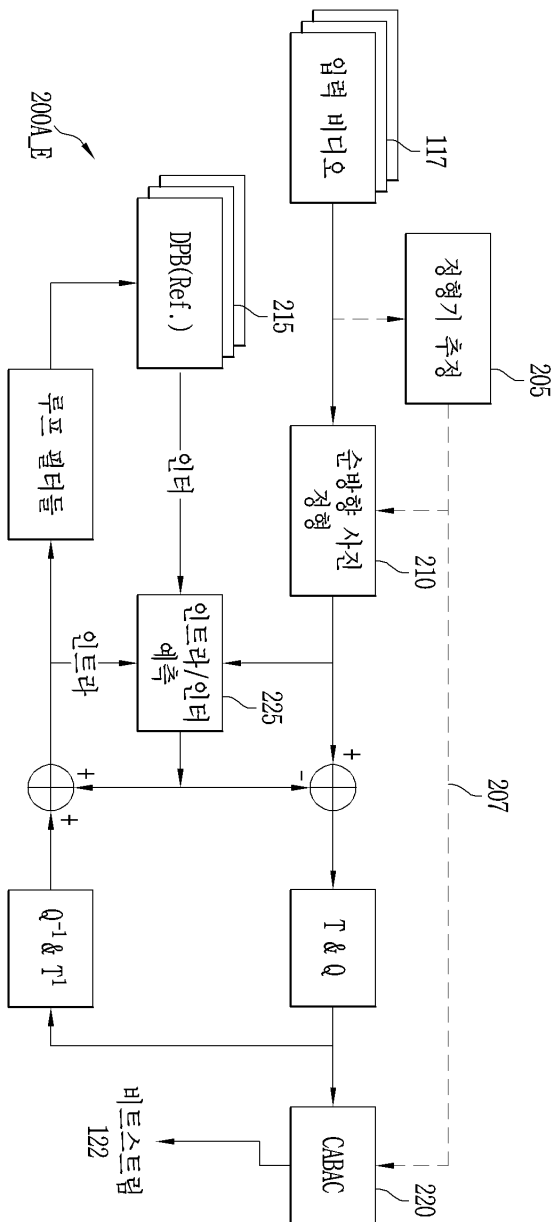
도면1a



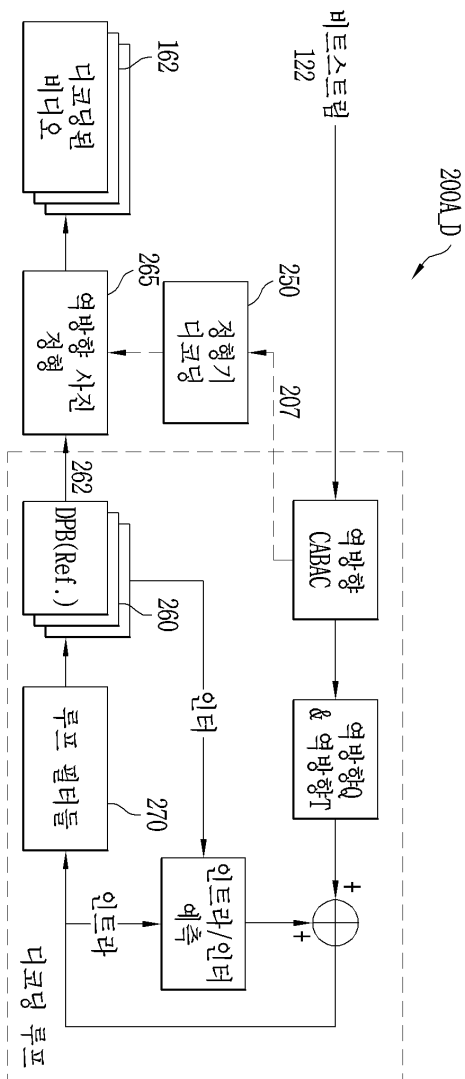
도면1b



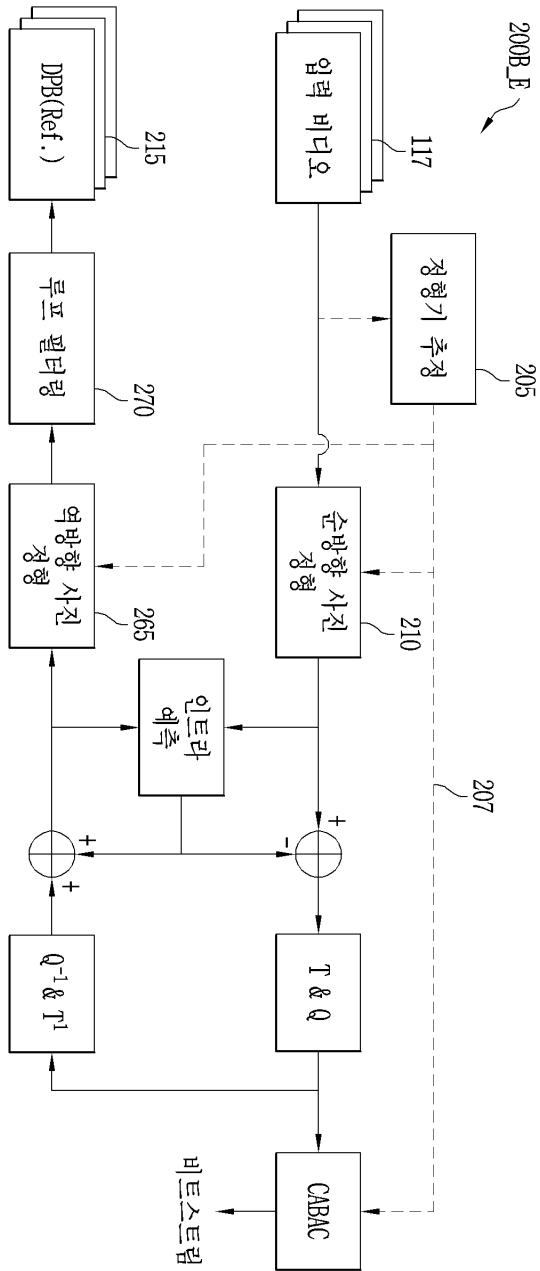
도면2a



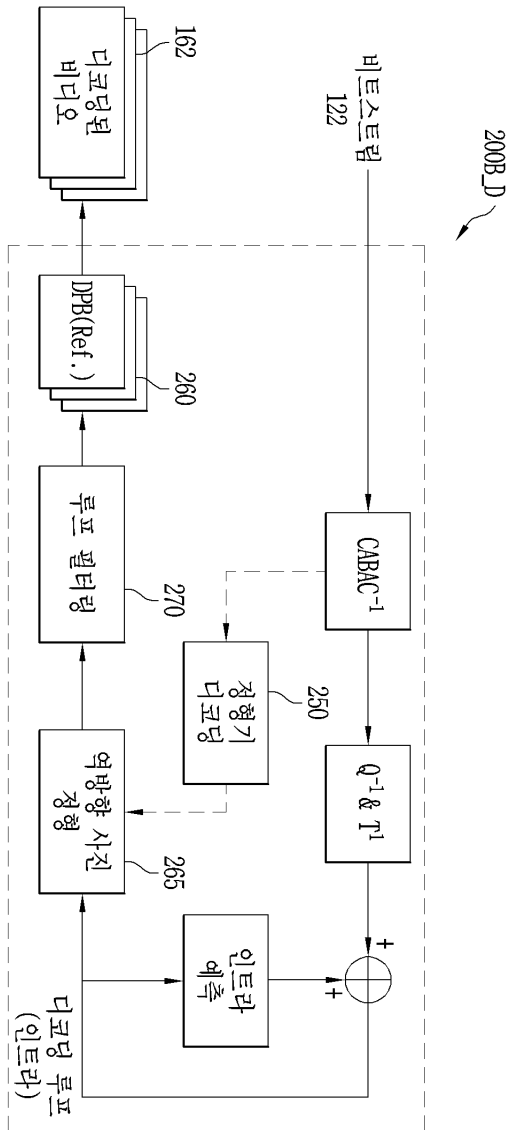
도면 2b



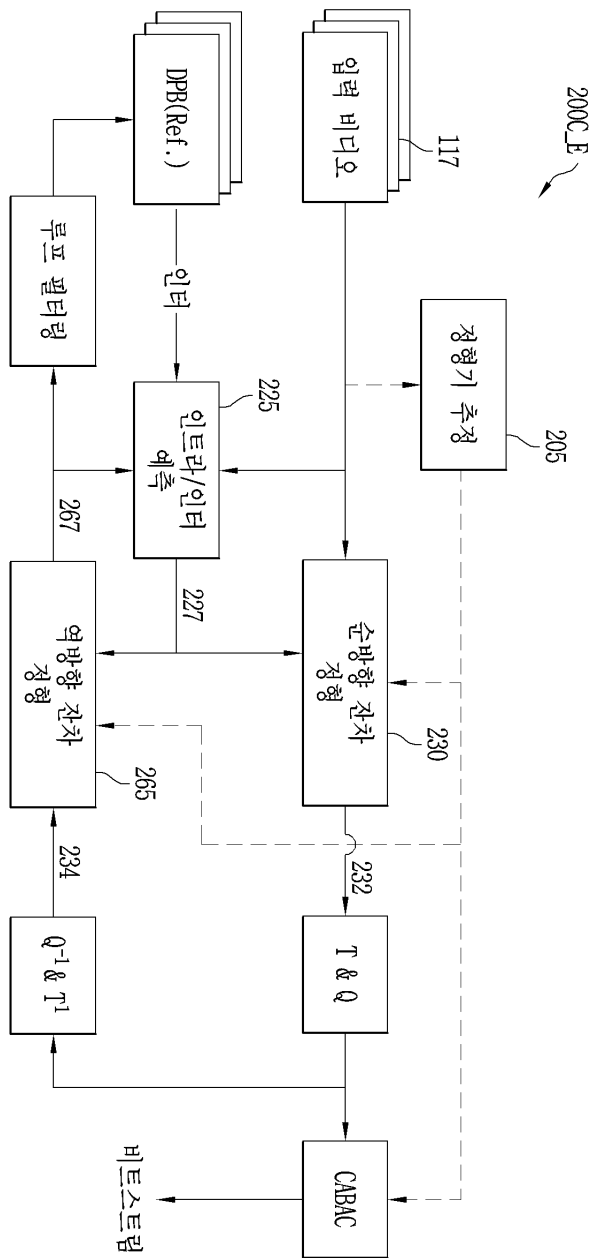
도면2c



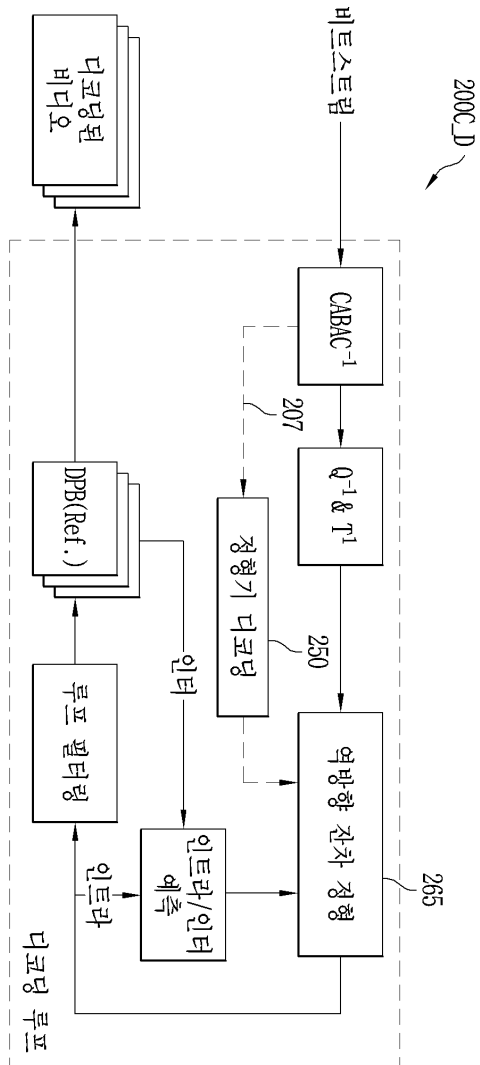
도면2d



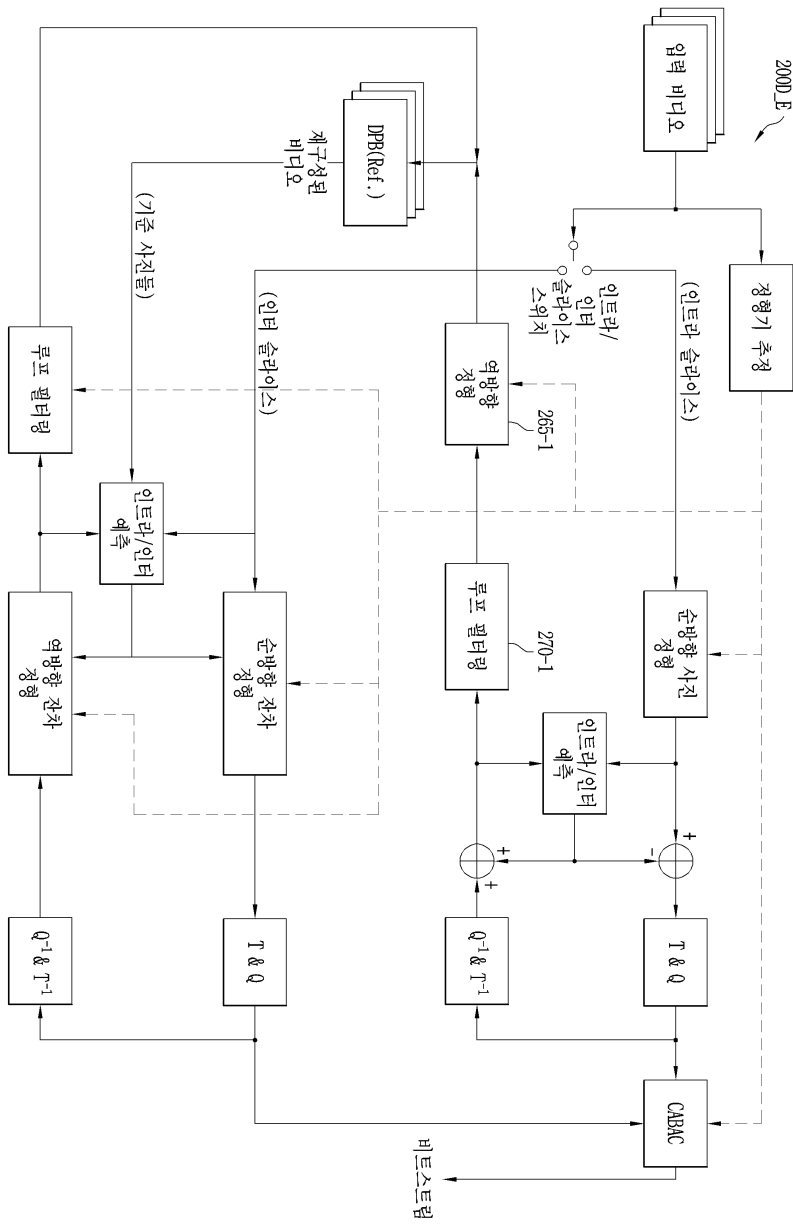
도면2e



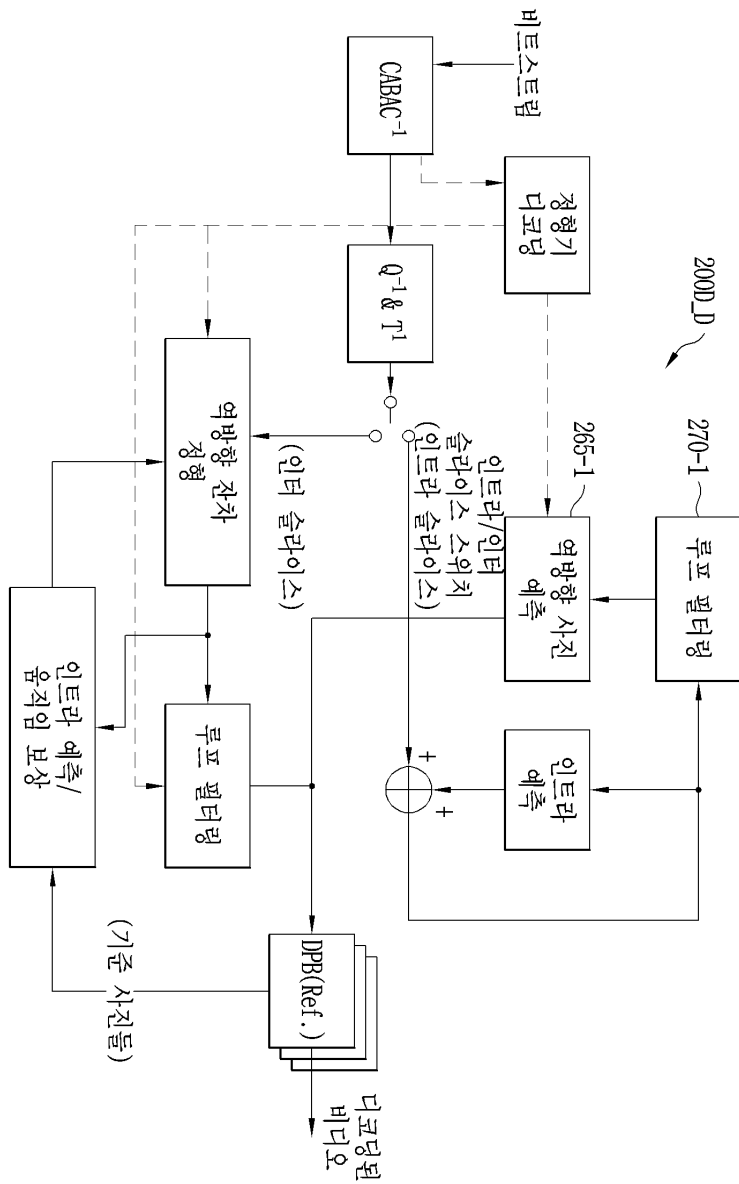
도면2f



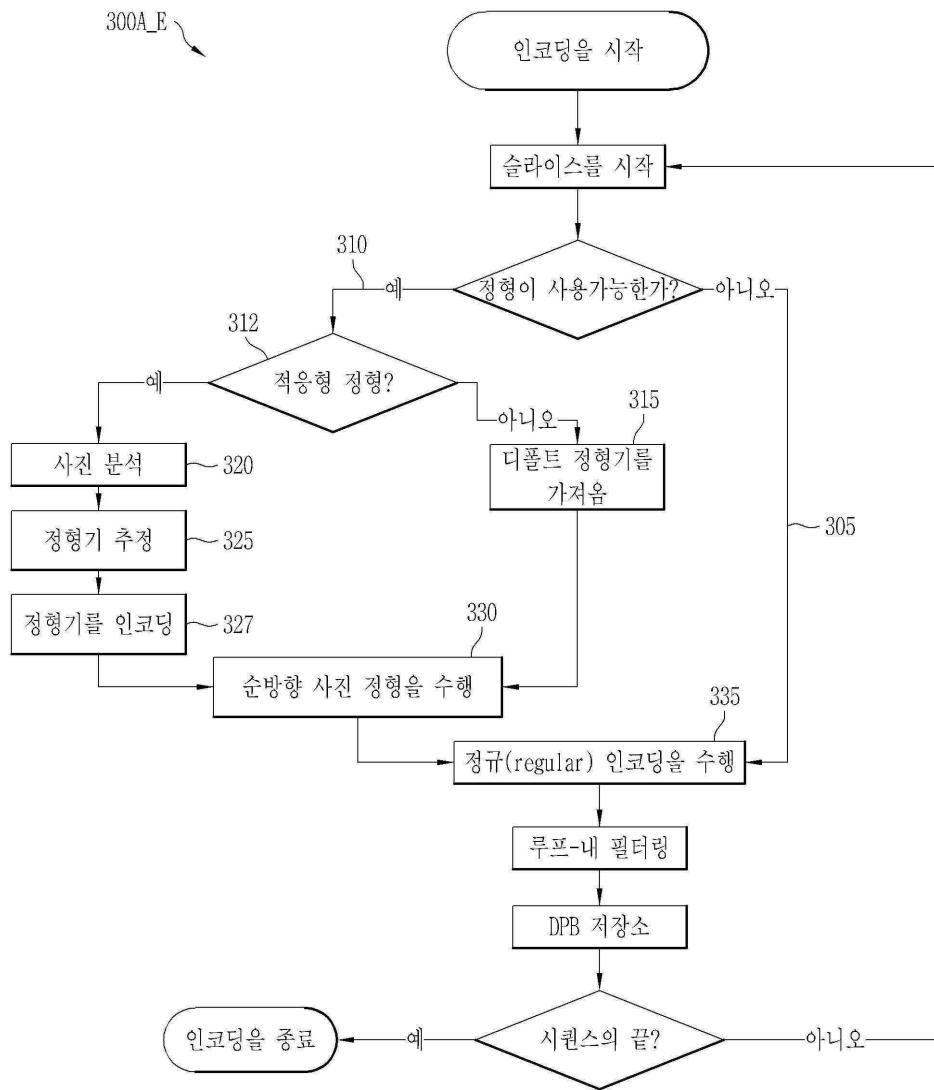
도면2g



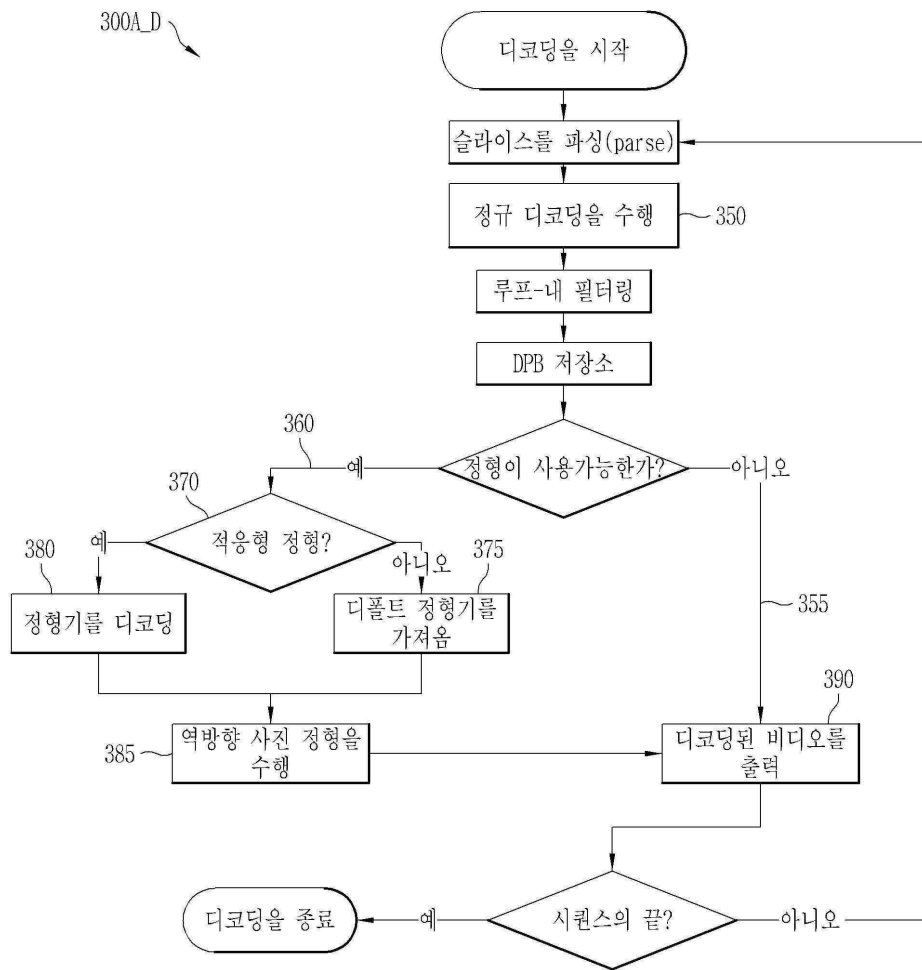
도면 2h



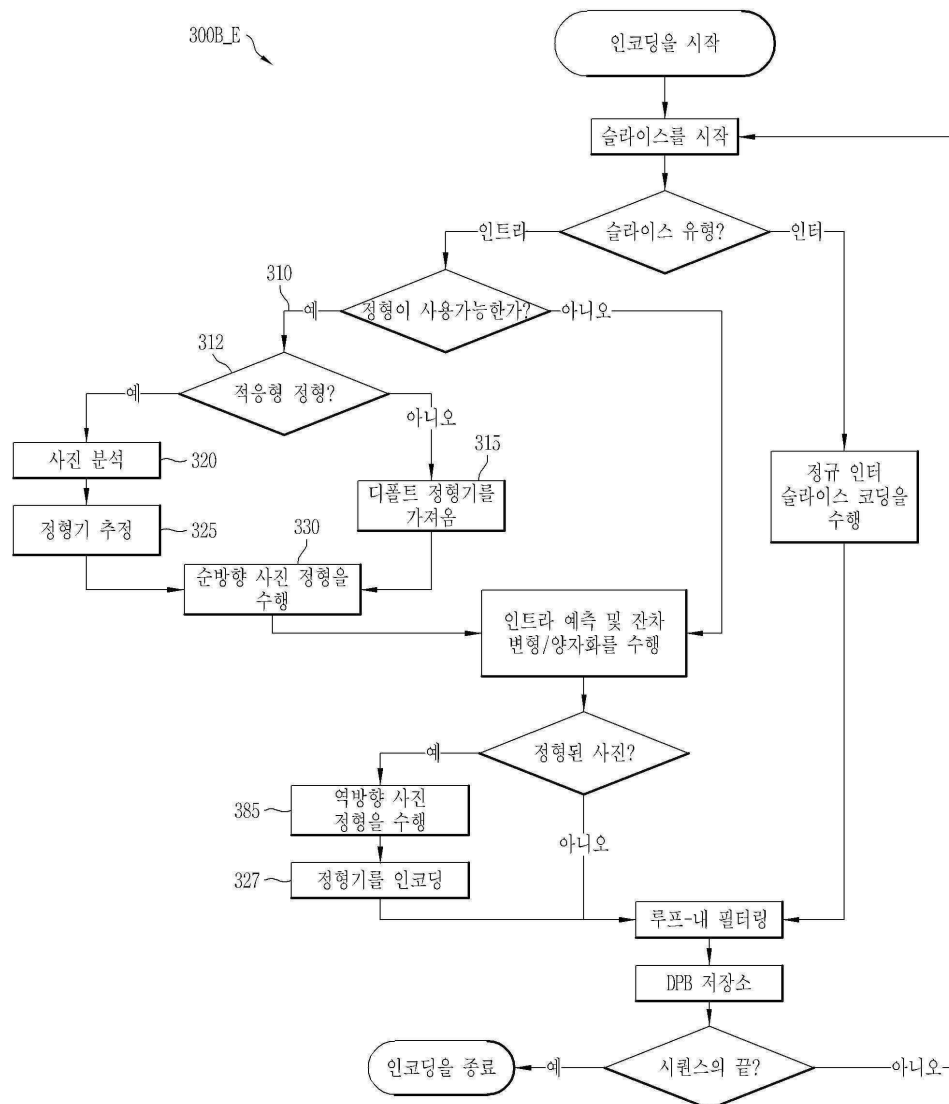
도면3a



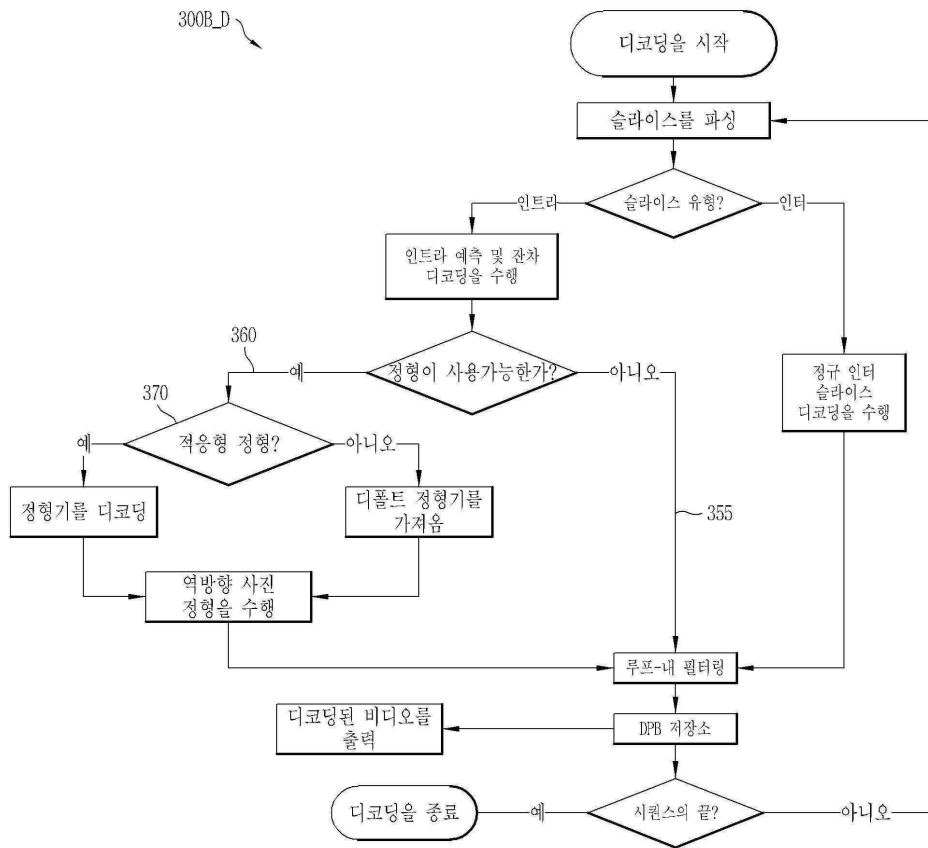
도면 3b



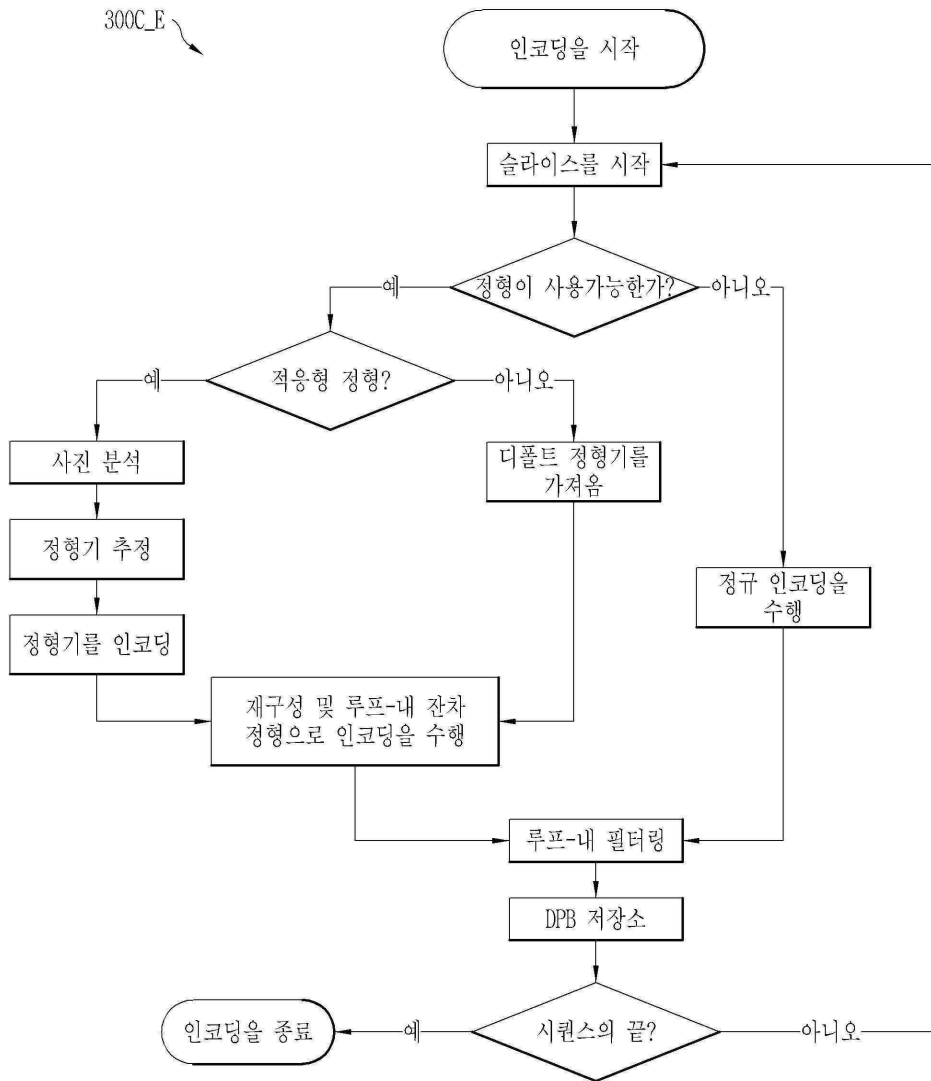
도면3c



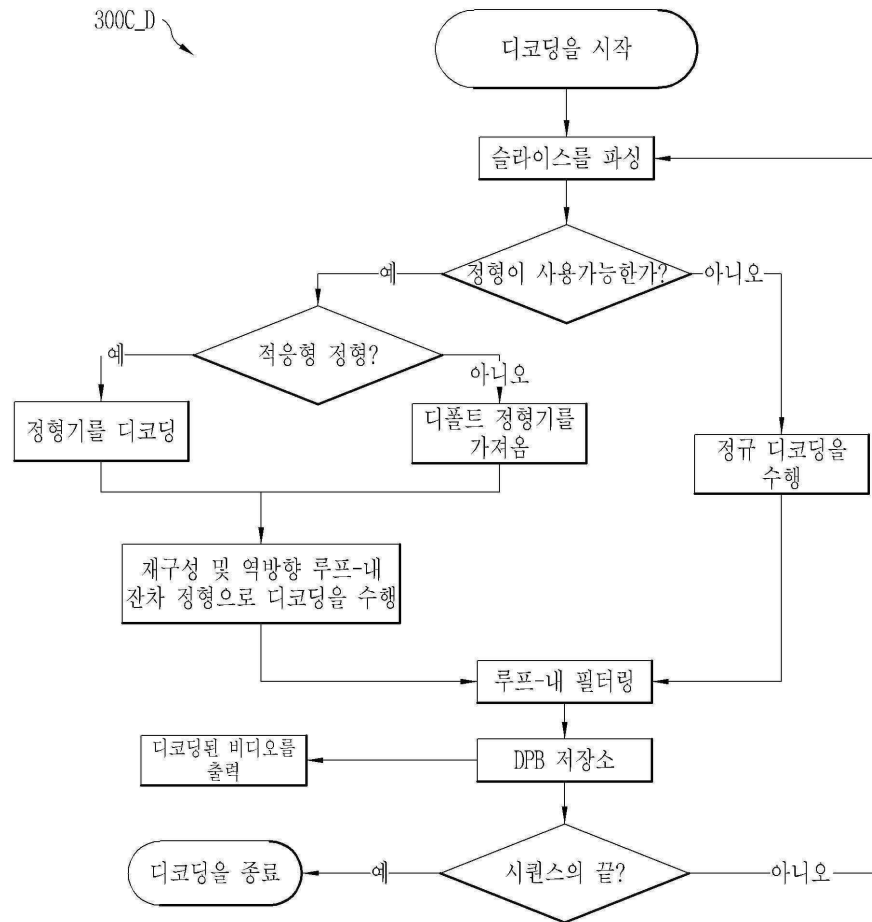
도면3d



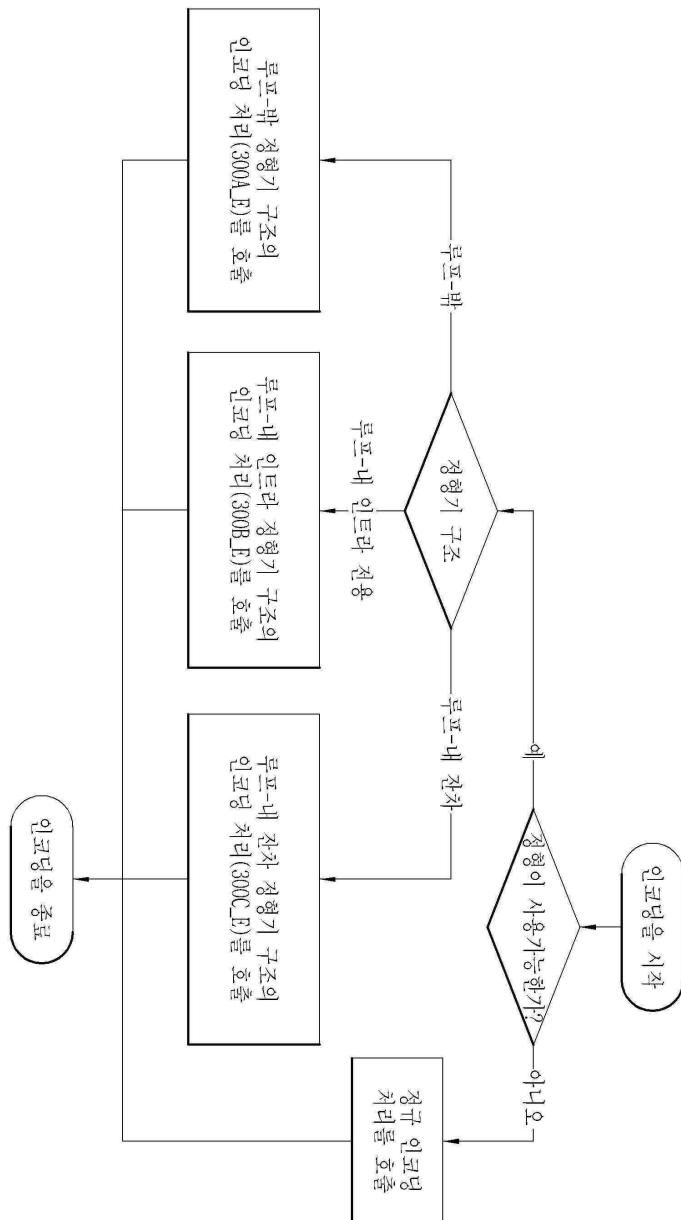
도면3e



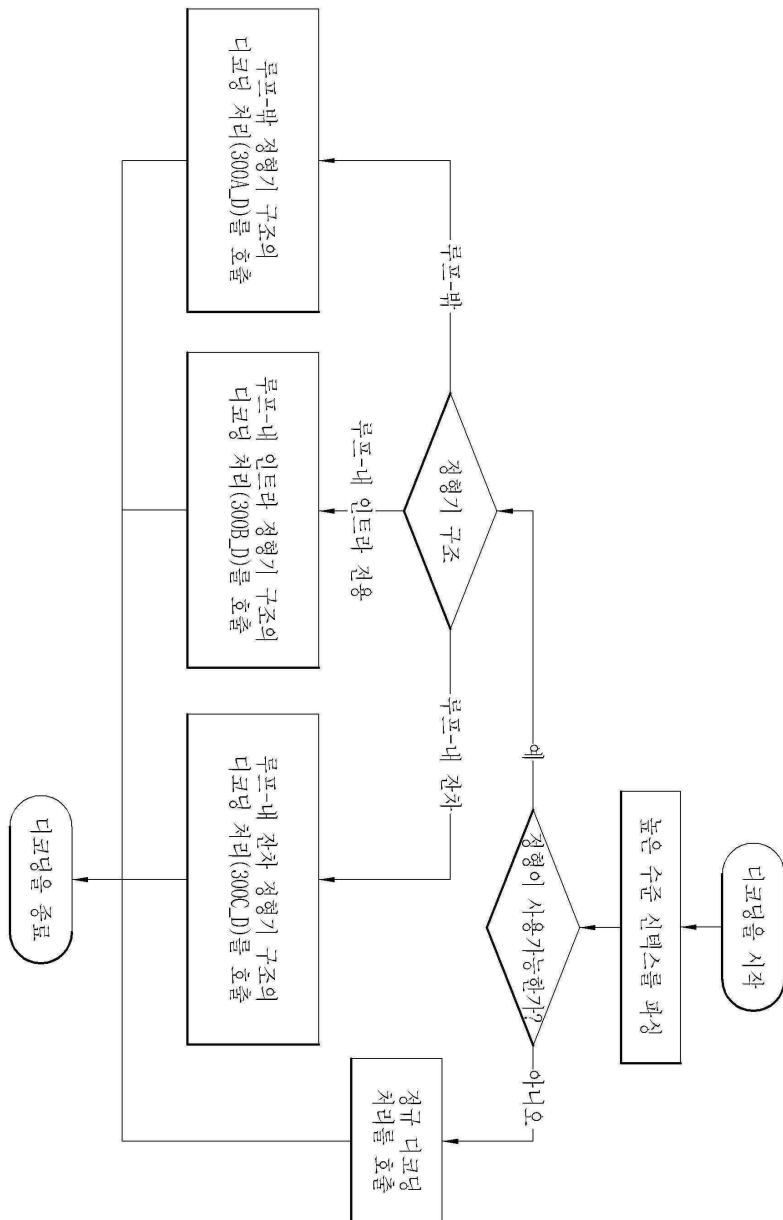
도면3f



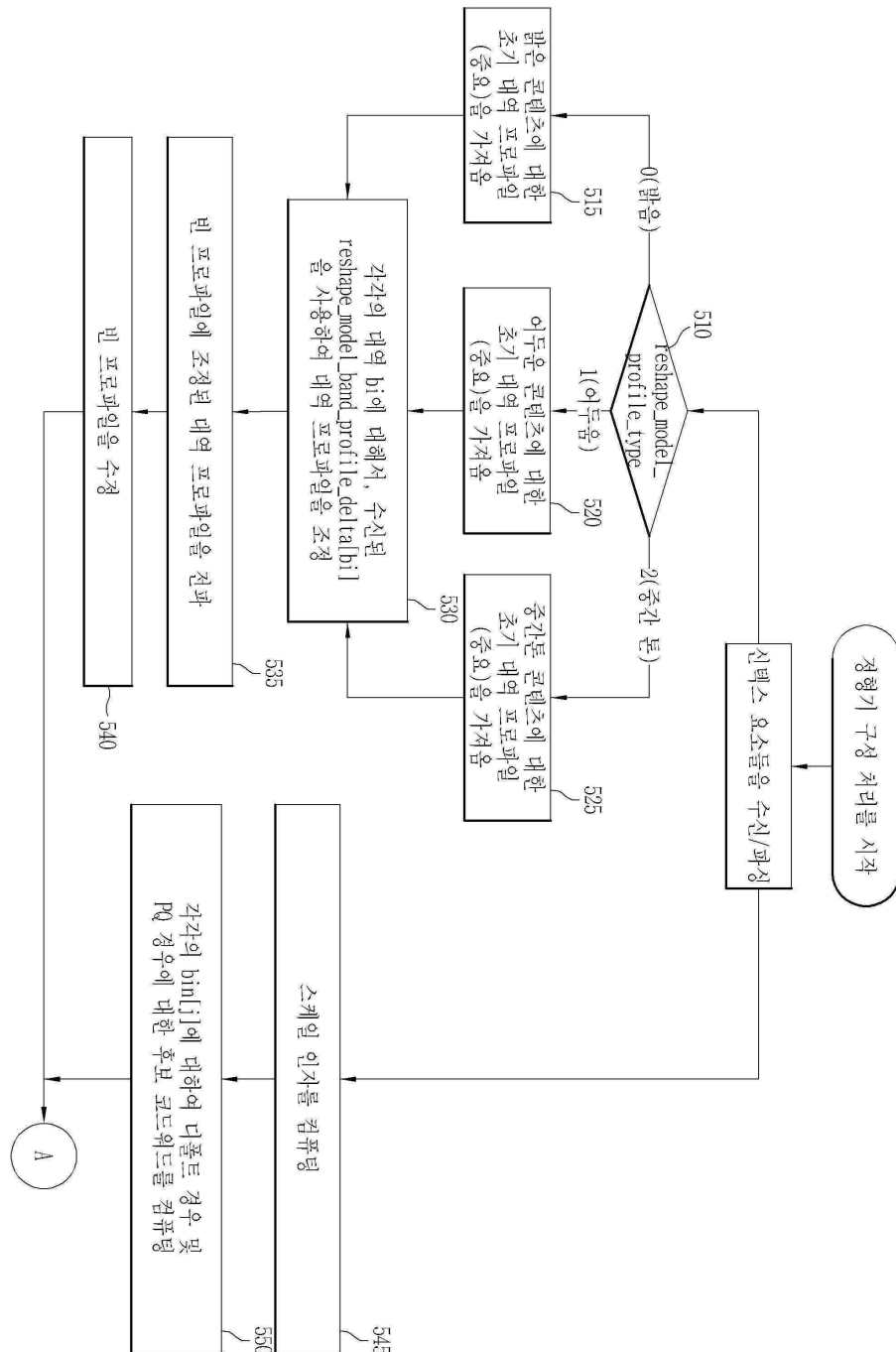
도면4a



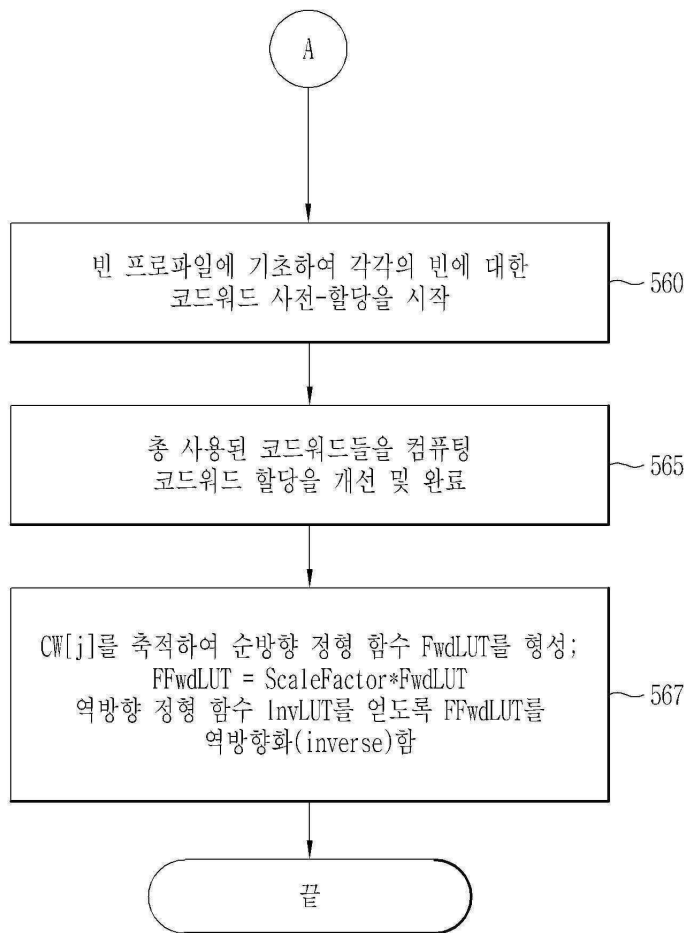
도면4b



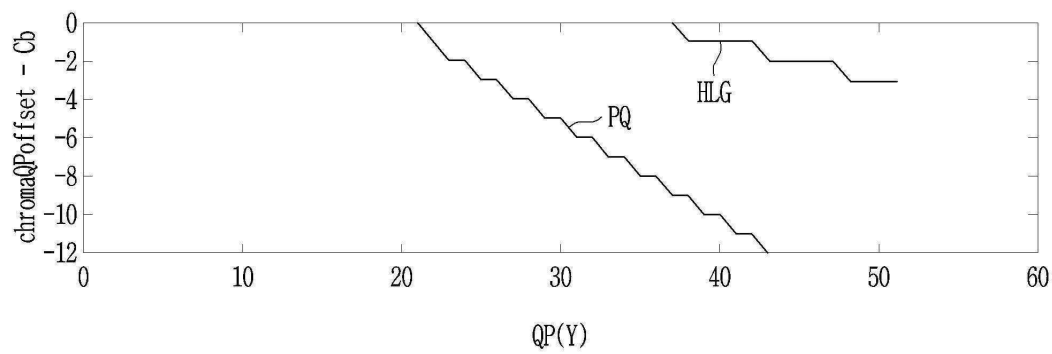
도면5a



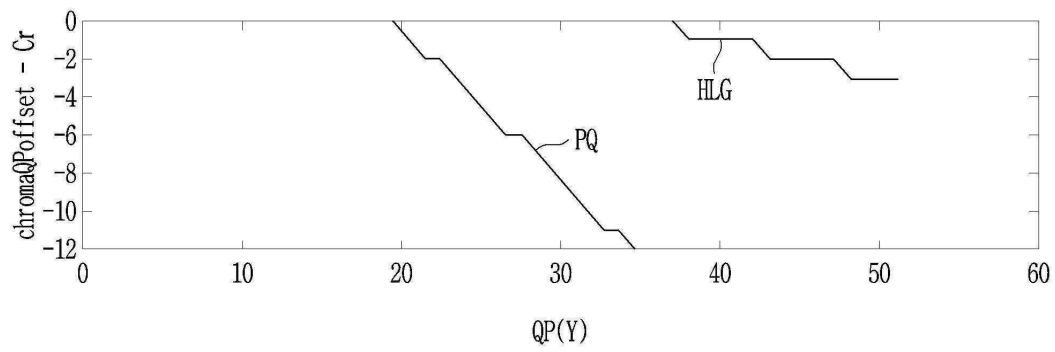
도면5b



도면6a



도면6b



도면7

