



(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2024-0108186
(43) 공개일자 2024년07월09일

(51) 국제특허분류(Int. Cl.)
H04N 19/70 (2014.01) G06T 9/00 (2019.01)
H04N 19/132 (2014.01) H04N 19/174 (2014.01)
H04N 19/186 (2014.01) H04N 19/82 (2014.01)
(52) CPC특허분류
H04N 19/70 (2015.01)
G06T 9/002 (2013.01)
(21) 출원번호 10-2023-0030625
(22) 출원일자 2023년03월08일
심사청구일자 없음
(30) 우선권주장
63/436,521 2022년12월31일 미국(US)
18/095,876 2023년01월11일 미국(US)

(71) 출원인
샤프 가부시키키가이샤
일본국 오사카후 사카이시 사카이쿠 타쿠미쵸 1반
치
(72) 발명자
데시판데, 사친 지.
미국 98683 워싱턴주 밴쿠버 사우스이스트 밀 플
레인 블러바드 17200 스위트 200 샤프 래보라토리
스 오브 아메리카 인크. 내
셰이크 시디아, 아흐메드
미국 98683 워싱턴주 밴쿠버 사우스이스트 밀 플
레인 블러바드 17200 스위트 200 샤프 래보라토리
스 오브 아메리카 인크. 내
(74) 대리인
장수길, 박충범

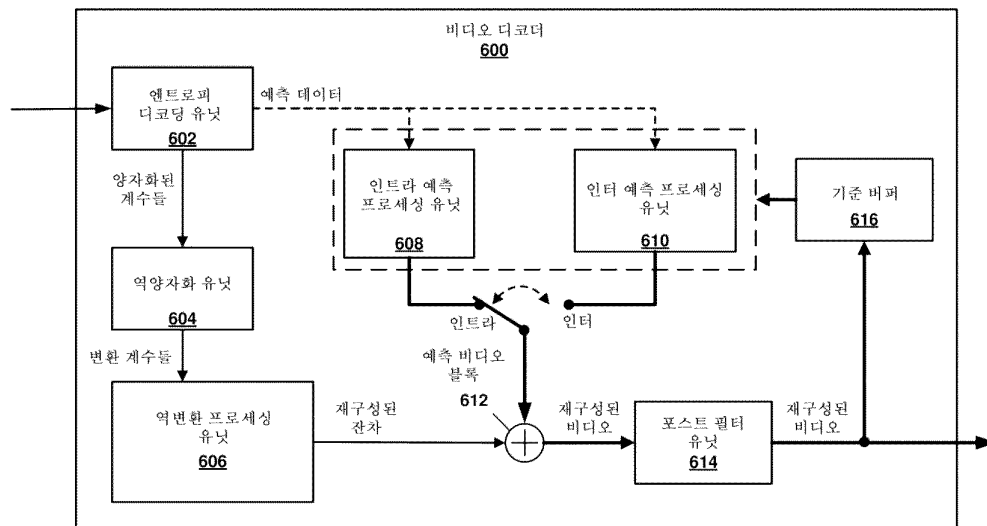
전체 청구항 수 : 총 5 항

(54) 발명의 명칭 비디오 코딩에서 신경 네트워크 포스트-필터 중첩 및 서브계층 프레임 레이트 업샘플링 정보
를 시그널링하기 위한 시스템들 및 방법들

(57) 요약

디바이스는 신경 네트워크 포스트-필터 특성 메시지에 포함된 정보에 기초하여 필터링을 수행하도록 구성될 수 있다. 하나의 예에서, 신경 네트워크 포스트-필터 특성들 메시지는 신경 네트워크 포스트-필터 특성 메시지와 연관된 다수의 서브계층들 각각에 대해 포스트-프로세싱 필터에 의해 생성된 보간된 픽처들의 수를 특징하는 신
택스 요소의 인스턴스를 포함한다.

대표도 - 도6



(52) CPC특허분류

H04N 19/132 (2015.01)

H04N 19/174 (2015.01)

H04N 19/186 (2015.01)

H04N 19/82 (2015.01)

G06T 2207/20084 (2013.01)

명세서

청구범위

청구항 1

비디오 데이터를 위한 신경 네트워크 필터링을 수행하는 방법으로서,

신경 네트워크 포스트-필터 특성 메시지를 수신하는 단계;

상기 신경 네트워크 포스트-필터 특성 메시지에서, 포스트-프로세싱 필터에 대한 입력에 필요한 패치 크기의 수평 샘플 카운트들을 나타내는 값을 갖는 제1 선택스 요소를 파싱하는 단계;

상기 신경 네트워크 포스트-필터 특성 메시지에서, 상기 포스트-프로세싱 필터에 대한 입력에 필요한 상기 패치 크기의 수직 샘플 카운트들을 나타내는 값을 갖는 제2 선택스 요소를 파싱하는 단계;

상기 신경 네트워크 포스트-필터 특성 메시지에서, 상기 포스트-프로세싱 필터의 인접 입력 텐서들의 중첩되는 수평 및 수직 샘플 카운트들을 나타내는 값을 갖는 제3 선택스 요소를 파싱하는 단계; 및

상기 신경 네트워크 포스트-필터 특성 메시지에서, 상기 포스트-프로세싱 필터가 입력으로서 (i) 상기 표시된 수평 샘플 카운트들과 동일한 폭 및 상기 표시된 수직 샘플 카운트들과 동일한 높이를 갖는 패치 크기를 수용하는지, 아니면 (ii) 상기 표시된 수평 샘플 카운트들의 배수 + 상기 표시된 중첩되는 수평 샘플 카운트들과 동일한 폭 및 상기 표시된 수직 샘플 카운트들의 배수 + 상기 표시된 중첩되는 수직 샘플 카운트들과 동일한 높이를 갖는 패치 크기를 수용하는지를 나타내는 제4 선택스 요소를 파싱하는 단계를 포함하는, 방법.

청구항 2

하나 이상의 프로세서들을 포함하는 디바이스로서, 상기 하나 이상의 프로세서들은,

신경 네트워크 포스트-필터 특성 메시지를 수신하도록;

상기 신경 네트워크 포스트-필터 특성 메시지에서, 포스트-프로세싱 필터에 대한 입력에 필요한 패치 크기의 수평 샘플 카운트들을 나타내는 값을 갖는 제1 선택스 요소를 파싱하도록;

상기 신경 네트워크 포스트-필터 특성 메시지에서, 상기 포스트-프로세싱 필터에 대한 입력에 필요한 상기 패치 크기의 수직 샘플 카운트들을 나타내는 값을 갖는 제2 선택스 요소를 파싱하도록;

상기 신경 네트워크 포스트-필터 특성 메시지에서, 상기 포스트-프로세싱 필터의 인접 입력 텐서들의 중첩되는 수평 및 수직 샘플 카운트들을 나타내는 값을 갖는 제3 선택스 요소를 파싱하도록; 그리고

상기 신경 네트워크 포스트-필터 특성 메시지에서, 상기 포스트-프로세싱 필터가 입력으로서 (i) 상기 표시된 수평 샘플 카운트들과 동일한 폭 및 상기 표시된 수직 샘플 카운트들과 동일한 높이를 갖는 패치 크기를 수용하는지, 아니면 (ii) 상기 표시된 수평 샘플 카운트들의 배수 + 상기 표시된 중첩되는 수평 샘플 카운트들과 동일한 폭 및 상기 표시된 수직 샘플 카운트들의 배수 + 상기 표시된 중첩되는 수직 샘플 카운트들과 동일한 높이를 갖는 패치 크기를 수용하는지를 나타내는 제4 선택스 요소를 파싱하도록 구성되는, 디바이스.

청구항 3

제2항에 있어서, 상기 디바이스는 비디오 디코더를 포함하는, 디바이스.

청구항 4

하나 이상의 프로세서들을 포함하는 디바이스로서, 상기 하나 이상의 프로세서들은,

신경 네트워크 포스트-필터 특성 메시지를 시그널링하도록;

상기 신경 네트워크 포스트-필터 특성 메시지에서, 포스트-프로세싱 필터에 대한 입력에 필요한 패치 크기의 수평 샘플 카운트들을 나타내는 값을 갖는 제1 선택스 요소를 시그널링하도록;

상기 신경 네트워크 포스트-필터 특성 메시지에서, 상기 포스트-프로세싱 필터에 대한 입력에 필요한 상기 패치

크기의 수직 샘플 카운트들을 나타내는 값을 갖는 제2 선택스 요소를 시그널링하도록;

상기 신경 네트워크 포스트-필터 특정 메시지에서, 상기 포스트-프로세싱 필터의 인접 입력 텐서들의 중첩되는 수평 및 수직 샘플 카운트들을 나타내는 값을 갖는 제3 선택스 요소를 시그널링하도록; 그리고

상기 신경 네트워크 포스트-필터 특정 메시지에서, 상기 포스트-프로세싱 필터가 입력으로서 (i) 상기 표시된 수평 샘플 카운트들과 동일한 폭 및 상기 표시된 수직 샘플 카운트들과 동일한 높이를 갖는 패치 크기를 수용하는지, 아니면 (ii) 상기 표시된 수평 샘플 카운트들의 배수 + 상기 표시된 중첩되는 수평 샘플 카운트들과 동일한 폭 및 상기 표시된 수직 샘플 카운트들의 배수 + 상기 표시된 중첩되는 수직 샘플 카운트들과 동일한 높이를 갖는 패치 크기를 수용하는지를 나타내는 제4 선택스 요소를 시그널링하도록 구성되는, 디바이스.

청구항 5

제4항에 있어서, 상기 디바이스는 비디오 인코더를 포함하는, 디바이스.

발명의 설명

기술 분야

[0001] 관련 출원

[0002] 본 출원은 2022년 12월 31일자로 출원된 미국 가출원 제63/436,521호의 이익을 주장하며, 이 출원은 그 전체가 참고로 포함된다.

[0003] 기술분야

[0004] 본 개시내용은 비디오 코딩에 관한 것으로, 더 구체적으로는, 코딩된 비디오에 대한 신경 네트워크 포스트-필터 파라미터 정보를 시그널링하기 위한 기법들에 관한 것이다.

배경 기술

[0005] 디지털 비디오 능력들은 디지털 텔레비전, 랩톱 또는 데스크톱 컴퓨터, 태블릿 컴퓨터, 디지털 기록 디바이스, 디지털 미디어 플레이어, 비디오 게이밍 디바이스, 소위 스마트폰을 포함하는 셀룰러 전화, 의료 이미징 디바이스 등을 비롯한 광범위한 디바이스들 내에 통합될 수 있다. 디지털 비디오는 비디오 코딩 표준에 따라 코딩될 수 있다. 비디오 코딩 표준들은 코딩된 비디오 데이터를 캡슐화하는 컴플라이언트 비트스트림(compliant bitstream)의 포맷을 정의한다. 컴플라이언트 비트스트림은 재구성된 비디오 데이터를 생성하기 위해 비디오 디코딩 디바이스에 의해 수신 및 디코딩될 수 있는 데이터 구조이다. 비디오 코딩 표준들은 비디오 압축 기법들을 포함할 수 있다. 비디오 코딩 표준들의 예는 ISO/IEC MPEG-4(Moving Picture Experts Group-4) 비주얼 및 ITU-T H.264(ISO/IEC MPEG-4 AVC로도 알려짐) 및 HEVC(High-Efficiency Video Coding)를 포함한다. HEVC는 문헌[High Efficiency Video Coding (HEVC), Rec. ITU-T H.265, December 2016]에 기재되어 있으며, 이는 참고로 포함되고, 본 명세서에서 ITU-T H.265로 지칭된다. 차세대 비디오 코딩 표준들의 개발을 위해 ITU-T H.265에 대한 확장들 및 개선들이 고려되고 있다. 예를 들어, ITU-T VCEG(Video Coding Experts Group) 및 ISO/IEC(MPEG)(JVET(Joint Video Exploration Team)로 총칭됨)는 현재 HEVC 표준의 압축 능력을 현저히 초과하는 압축 능력을 갖는 표준화된 비디오 코딩 기술을 갖는다. JEM 7(Joint Exploration Model 7), JEM 7의 알고리즘 설명, 본 명세서에 참고로 포함되는 문헌[ISO/IEC JTC1/SC29/WG11 Document: JVET-G1001, July 2017, Torino, IT]은 JVET에 의한 조정된 테스트 모델 연구 하에 있던 코딩 특징들을, ITU-T H.265의 능력들 이상으로 비디오 코딩 기술을 잠재적으로 향상시키는 것으로서 설명한다. JEM 7의 코딩 특징들은 JEM 기준 소프트웨어로 구현된다는 점에 유의해야 한다. 본 명세서에서 사용되는 바와 같이, 용어 "JEM"은 JEM 7에 포함된 알고리즘들 및 JEM 기준 소프트웨어의 구현들을 총칭할 수 있다. 또한, VCEG 및 MPEG에 의해 공동으로 발행된 문헌["Joint Call for Proposals on Video Compression with Capabilities beyond HEVC"]에 응답하여, 비디오 코딩 툴들의 다수의 설명들이 ISO/IEC JTC1/SC29/WG11의 제10차 회의(16-20 April 2018, San Diego, CA)에서 다양한 그룹들에 의해 제안되었다. 비디오 코딩 툴들의 다수의 설명들로부터, 비디오 코딩 규격의 생성된 초안 텍스트는 문헌["Versatile Video Coding (Draft 1)", 10th Meeting of ISO/IEC JTC1/SC29/WG11 16-20 April 2018, San Diego, CA, document JVET-J1001-v2]에 기재되어 있으며, 이는 본 명세서에 참고로 포함되고 JVET-J1001로 지칭된다. VCEG 및 MPEG에 의한 비디오 코딩 표준의 이러한 개발은 VVC(Versatile Video Coding) 프로젝트로 지칭된다. 본 명세서에 참고로 포함되고 JVET-T2001로 지칭되는 문헌["Versatile Video Coding (Draft 10)",

20th Meeting of ISO/IEC JTC1/SC29/WG11 7 - 16 October 2020, Teleconference, document JVET-T2001-v2]은 VVC 프로젝트에 대응하는 비디오 코딩 규격의 초안 텍스트의 현재 반복을 표현한다.

[0006] 비디오 압축 기법은 비디오 데이터를 저장하고 송신하기 위한 데이터 요건들이 감소되는 것을 가능하게 한다. 비디오 압축 기법은 비디오 시퀀스 내의 고유 리던던시(redundancy)를 활용함으로써 데이터 요건들을 감소시킬 수 있다. 비디오 압축 기법들은 비디오 시퀀스를 연속적으로 더 작은 부분들(즉, 비디오 시퀀스 내의 픽처들의 그룹들, 픽처들의 그룹 내의 픽처, 픽처 내의 영역들, 영역들 내의 서브-영역들 등)로 세분할 수 있다. 인트라 예측(intra prediction) 코딩 기법들(예컨대, 픽처 내에서의 공간 예측 기법들) 및 인터 예측(inter prediction) 기법들(즉, 인터-픽처 기법들(시간적))은 코딩될 비디오 데이터의 유닛과 비디오 데이터의 기준 유닛 사이의 차이 값들을 생성하는 데 사용될 수 있다. 차이 값들은 잔차 데이터로 치칭될 수 있다. 잔차 데이터는 양자화된 변환 계수들로서 코딩될 수 있다. 신택스 요소(syntax element)들은 잔차 데이터 및 기준 코딩 유닛(예컨대, 인트라 예측 모드 인덱스들, 및 모션 정보)과 관련될 수 있다. 잔차 데이터 및 신택스 요소들은 엔트로피 코딩될 수 있다. 엔트로피 인코딩된 잔차 데이터 및 신택스 요소들은 컴플라이언트 비트스트림을 형성하는 데이터 구조들에 포함될 수 있다.

발명의 내용

[0007] 대체적으로, 본 개시내용은 비디오 데이터를 코딩하기 위한 다양한 기법들을 기술한다. 특히, 본 개시내용은 코딩된 비디오 데이터에 대한 신경 네트워크 포스트-필터 파라미터 정보를 시그널링하기 위한 기법들을 기술한다. 본 개시내용의 기법들이 ITU-T H.264, ITU-T H.265, JEM, 및 JVET-T2001과 관련하여 기술되지만, 본 개시내용의 기법들은 대체로 비디오 코딩에 적용가능하다는 점에 유의해야 한다. 예를 들어, 본 명세서에서 기술되는 코딩 기법은 비디오 블록 구조들, 인트라 예측 기법, 인터 예측 기법, 변환 기법, 필터링 기법, 및/또는 ITU-T H.265, JEM, 및 JVET-T2001에 포함되는 것들 이외의 엔트로피 코딩 기법을 포함하는 비디오 코딩 시스템들(향후 비디오 코딩 표준들에 기초하는 비디오 코딩 시스템들을 포함함)에 통합될 수 있다. 따라서, ITU-T H.264, ITU-T H.265, JEM, 및/또는 JVET-T2001에 대한 언급은 설명 목적을 위한 것이며, 본 명세서에서 기술되는 기법들의 범주를 제한하는 것으로 해석되어서는 안 된다. 또한, 본 명세서에 문헌들을 참고로 통합하는 것은 설명 목적을 위한 것이며, 본 명세서에서 사용되는 용어들과 관련하여 불명료성을 제한하거나 생성하는 것으로 해석되어서는 안 된다는 점에 유의해야 한다. 예를 들어, 통합된 참고문헌이 다른 통합된 참고문헌과는 상이한 정의의 용어를 제공하는 경우에 그리고/또는 그 용어가 본 명세서에서 사용될 때, 이 용어는 개개의 정의 각각을 광범위하게 포함하는 방식으로 그리고/또는 대안에서의 특정 정의 각각을 포함하는 방식으로 해석되어야 한다.

[0008] 하나의 예에서, 비디오 데이터를 코딩하는 방법은 신경 네트워크 포스트-필터 특성 메시지를 시그널링하는 단계, 및 신경 네트워크 포스트-필터 특성 메시지와 연관된 다수의 서브계층들 각각에 대해, 신경 네트워크 포스트-필터 특성 메시지에서, 포스트-프로세싱 필터에 의해 생성된 보간된 픽처들의 수를 특징하는 신택스 요소를 시그널링하는 단계를 포함한다.

[0009] 하나의 예에서, 디바이스는 신경 네트워크 포스트-필터 특성 메시지를 시그널링하도록, 그리고 신경 네트워크 포스트-필터 특성 메시지와 연관된 다수의 서브계층들 각각에 대해, 신경 네트워크 포스트-필터 특성 메시지에서, 포스트-프로세싱 필터에 의해 생성된 보간된 픽처들의 수를 특징하는 신택스 요소를 시그널링하도록 구성된 하나 이상의 프로세서들을 포함한다.

[0010] 하나의 예에서, 비일시적 컴퓨터 판독가능 저장 매체는 그에 저장된 명령어들을 포함하고, 명령어들은, 실행될 때, 디바이스의 하나 이상의 프로세서들로 하여금, 신경 네트워크 포스트-필터 특성 메시지를 시그널링하게 하고, 신경 네트워크 포스트-필터 특성 메시지와 연관된 다수의 서브계층들 각각에 대해, 신경 네트워크 포스트-필터 특성 메시지에서, 포스트-프로세싱 필터에 의해 생성된 보간된 픽처들의 수를 특징하는 신택스 요소를 시그널링하게 한다.

[0011] 하나의 예에서, 장치는 신경 네트워크 포스트-필터 특성 메시지를 시그널링하기 위한 수단, 및 신경 네트워크 포스트-필터 특성 메시지에서, 신경 네트워크 포스트-필터 특성 메시지와 연관된 다수의 서브계층들 각각에 대해 포스트-프로세싱 필터에 의해 생성된 보간된 픽처들의 수를 특징하는 신택스 요소를 시그널링하기 위한 수단을 포함한다.

[0012] 하나의 예에서, 비디오 데이터를 디코딩하는 방법은 신경 네트워크 포스트-필터 특성 메시지를 수신하는 단계, 및 신경 네트워크 포스트-필터 특성 메시지와 연관된 다수의 서브계층들 각각에 대해, 신경 네트워크 포스트-필

터 특성 메시지에서부터, 포스트-프로세싱 필터에 의해 생성된 보간된 픽처들의 수를 특정하는 신택스 요소를 파싱하는 단계를 포함한다.

[0013] 하나의 예에서, 디바이스는 신경 네트워크 포스트-필터 특성 메시지를 수신하도록, 그리고 신경 네트워크 포스트-필터 특성 메시지와 연관된 다수의 서브계층들 각각에 대해, 신경 네트워크 포스트-필터 특성 메시지에서부터, 포스트-프로세싱 필터에 의해 생성된 보간된 픽처들의 수를 특정하는 신택스 요소를 파싱하도록 구성된 하나 이상의 프로세서들을 포함한다.

[0014] 하나의 예에서, 비일시적 컴퓨터 판독가능 저장 매체는 그에 저장된 명령어들을 포함하고, 명령어들은, 실행될 때, 디바이스의 하나 이상의 프로세서들로 하여금, 신경 네트워크 포스트-필터 특성 메시지를 수신하게 하고, 신경 네트워크 포스트-필터 특성 메시지와 연관된 다수의 서브계층들 각각에 대해, 신경 네트워크 포스트-필터 특성 메시지에서부터, 포스트-프로세싱 필터에 의해 생성된 보간된 픽처들의 수를 특정하는 신택스 요소를 파싱하게 한다.

[0015] 하나의 예에서, 장치는 신경 네트워크 포스트-필터 특성 메시지를 수신하기 위한 수단, 및 신경 네트워크 포스트-필터 특성 메시지에서부터, 신경 네트워크 포스트-필터 특성 메시지와 연관된 다수의 서브계층들 각각에 대해 포스트-프로세싱 필터에 의해 생성된 보간된 픽처들의 수를 특정하는 신택스 요소를 파싱하기 위한 수단을 포함한다.

[0016] 하나 이상의 예의 세부사항들이 첨부 도면들 및 다음의 설명에 기재된다. 다른 특징들, 목적들 및 이점들이 설명 및 도면들로부터 그리고 청구범위로부터 명백해질 것이다.

도면의 간단한 설명

[0017] 도 1은 본 개시내용의 하나 이상의 기법들에 따른, 비디오 데이터를 인코딩 및 디코딩하도록 구성될 수 있는 시스템의 일례를 예시하는 블록도이다.

도 2는 본 개시내용의 하나 이상의 기법들에 따른, 코딩된 비디오 데이터 및 대응하는 데이터 구조들을 예시하는 개념도이다.

도 3은 본 개시내용의 하나 이상의 기법들에 따른, 코딩된 비디오 데이터 및 대응하는 메타데이터를 캡슐화하는 데이터 구조를 예시하는 개념도이다.

도 4는 본 개시내용의 하나 이상의 기법들에 따른, 비디오 데이터를 인코딩 및 디코딩하도록 구성될 수 있는 시스템의 일 구현예에 포함될 수 있는 컴포넌트들의 일례를 예시하는 개념도이다.

도 5는 본 개시내용의 하나 이상의 기법들에 따른, 비디오 데이터를 인코딩하도록 구성될 수 있는 비디오 인코더의 일례를 예시하는 블록도이다.

도 6은 본 개시내용의 하나 이상의 기법들에 따른, 비디오 데이터를 디코딩하도록 구성될 수 있는 비디오 디코더의 일례를 예시하는 블록도이다.

도 7은 본 개시내용의 하나 이상의 기법들에 따른, 루마 성분의 패킹된 데이터 채널들의 일례를 예시하는 개념도이다.

발명을 실시하기 위한 구체적인 내용

[0018] 비디오 콘텐츠는 일련의 프레임들(또는 픽처들)로 구성된 비디오 시퀀스들을 포함한다. 일련의 프레임들은 또한 GOP(group of pictures)로 지칭될 수 있다. 각각의 비디오 프레임 또는 픽처는 하나 이상의 영역들로 나뉠 수 있다. 영역들은 기본 유닛(예컨대, 비디오 블록), 및 영역을 정의하는 규칙들의 세트들에 따라 정의될 수 있다. 예를 들어, 영역을 정의하는 규칙은 일정 영역이 직사각형으로 배열된 정수 개의 비디오 블록들이어야 한다는 것일 수 있다. 또한, 일정 영역 내의 비디오 블록들은 스캔 패턴(예컨대, 래스터 스캔)에 따라 정리될 수 있다. 본 명세서에서 사용되는 바와 같이, 용어 "비디오 블록"은 일반적으로 픽처의 영역을 지칭할 수 있거나, 또는 더 구체적으로, 예측적으로 코딩될 수 있는 샘플 값들의 최대 어레이, 그의 서브분할부(sub-division)들, 및/또는 대응하는 구조들을 지칭할 수 있다. 또한, 용어 "현재 비디오 블록"은 인코딩 또는 디코딩되고 있는 픽처의 영역을 지칭할 수 있다. 비디오 블록은 샘플 값들의 어레이로서 정의될 수 있다. 일부 경우에, 픽셀 값들은, 컬러 성분들(예컨대, 루마(Y) 및 크로마(Cb, Cr) 성분들 또는 적색, 녹색, 및 청색 성분들)로도 지칭될 수 있는, 비디오 데이터의 각각의 성분들에 대한 샘플 값들을 포함하는 것으로서 설명될 수 있

다는 점에 유의해야 한다. 일부 경우에, 용어들 "픽셀 값" 및 "샘플 값"은 상호교환가능하게 사용된다는 점에 유의해야 한다. 또는, 일부 경우들에 있어서, 픽셀 또는 샘플은 펠(pel)로 지칭될 수 있다. 크로마 포맷으로도 지칭될 수 있는 비디오 샘플링 포맷은 비디오 블록에 포함된 루마 샘플들의 수와 관련하여 비디오 블록에 포함된 크로마 샘플들의 수를 정의할 수 있다. 예를 들어, 4:2:0 샘플링 포맷의 경우, 루마 성분에 대한 샘플링 레이트는 수평 방향 및 수직 방향 양측 모두에 대해 크로마 성분들의 샘플링 레이트의 2배이다.

[0019] 비디오 인코더는 비디오 블록들 및 그의 서브분할부들에 대해 예측 인코딩을 수행할 수 있다. 비디오 블록들 및 그의 서브분할부들은 노드들로 지칭될 수 있다. ITU-T H.264는 16×16 개의 루마 샘플들을 포함하는 매크로 블록을 명시한다. 즉, ITU-T H.264에서, 픽처는 매크로블록들로 세그먼트화된다. ITU-T H.265는 유사한 코딩 트리 유닛(Coding Tree Unit, CTU) 구조(이는 최대 코딩 유닛(largest coding unit, LCU)으로 지칭될 수 있음)을 명시한다. ITU-T H.265에서, 픽처들은 CTU들로 세그먼트화된다. ITU-T H.265에서, 픽처에 대해, CTU 크기는 16×16 , 32×32 , 또는 64×64 개의 루마 샘플들을 포함하는 것으로 설정될 수 있다. ITU-T H.265에서, CTU는 비디오 데이터의 각각의 성분(예컨대, 루마(Y) 및 크로마(Cb, Cr))에 대한 각각의 코딩 트리 블록(Coding Tree Block, CTB)들로 구성된다. 하나의 루마 성분 및 2개의 대응하는 크로마 성분들을 갖는 비디오가 2개의 채널들, 즉 루마 채널 및 채널 채널을 갖는 것으로 기술될 수 있다는 점에 유의해야 한다. 또한, ITU-T H.265에서, CTU는 쿼드트리(QT) 분할 구조에 따라 분할될 수 있고, 이는 CTU의 CTB들이 코딩 블록(CB)들로 분할되는 결과를 야기한다. 즉, ITU-T H.265에서, CTU는 쿼드트리 리프 노드(quadtree leaf node)들로 분할될 수 있다. ITU-T H.265에 따르면, 2개의 대응하는 크로마 CB들 및 연관된 선택 요소들과 함께, 하나의 루마 CB는 코딩 유닛(coding unit, CU)으로 지칭된다. ITU-T H.265에서, CB의 최소 허용 크기가 시그널링될 수 있다. ITU-T H.265에서, 루마 CB의 가장 작은 최소 허용 크기는 8×8 개의 루마 샘플들이다. ITU-T H.265에서, 인트라 예측 또는 인터 예측을 사용하여 픽처 영역을 코딩하는 결정이 CU 레벨에서 이루어진다.

[0020] ITU-T H.265에서, CU는 CU에 루트(root)를 갖는 예측 유닛 구조와 연관된다. ITU-T H.265에서, 예측 유닛 구조들은 루마 및 크로마 CB들이 대응하는 기준 샘플들을 생성할 목적으로 분할되게 한다. 즉, ITU-T H.265에서, 루마 및 크로마 CB들은 각각의 루마 및 크로마 예측 블록(PB)들로 분할될 수 있으며, 여기서 PB는 동일한 예측이 적용되는 샘플 값들의 블록을 포함한다. ITU-T H.265에서, CB는 1개, 2개 또는 4개의 PB들로 분할될 수 있다. ITU-T H.265는 64×64 개의 샘플들로부터 4×4 개의 샘플들에 이르기까지의 PB 크기들을 지원한다. ITU-T H.265에서, 정사각형 PB들은 인트라 예측을 위해 지원되며, 여기서 CB는 PB를 형성할 수 있거나 또는 CB는 4개의 정사각형 PB들로 분할될 수 있다. ITU-T H.265에서, 정사각형 PB들에 더하여, 직사각형 PB들이 인터 예측을 위해 지원되며, 여기서 CB는 수직으로 또는 수평으로 반분되어 PB들을 형성할 수 있다. 또한, ITU-T H.265에서, 인터 예측을 위해, 4개의 비대칭 PB 분할부들이 지원되며, 여기서 CB는 CB의 (상단 또는 하단에서의) 높이 또는 (좌측 또는 우측에서의) 폭의 1/4에서 2개의 PB들로 분할된다는 점에 유의해야 한다. PB에 대응하는 인트라 예측 데이터(예컨대, 인트라 예측 모드 선택 요소들) 또는 인터 예측 데이터(예컨대, 모션 데이터 선택 요소들)가 PB에 대한 기준 및/또는 예측된 샘플 값들을 생성하는 데 사용된다.

[0021] JEM은 최대 크기가 256×256 개의 루마 샘플들인 CTU를 명시한다. JEM은 쿼드트리 플러스 이진 트리(quadtree plus binary tree, QTBT) 블록 구조를 명시한다. JEM에서, QTBT 구조는 쿼드트리 리프 노드들이 이진 트리(BT) 구조에 의해 추가로 분할되는 것을 가능하게 한다. 즉, JEM에서, 이진 트리 구조는 쿼드트리 리프 노드들이 수직으로 또는 수평으로 재귀적으로 나뉘는 것을 가능하게 한다. JVET-T2001에서, CTU들은 쿼드트리 플러스 쿼드트리 플러스 멀티 유형 트리(QTMT 또는 QT+MTT) 구조에 따라 분할된다. JVET-T2001에서의 QTMT는 JEM에서의 QTBT와 유사하다. 그러나, JVET-T2001에서, 이진 분할들을 나타내는 것 이외에, 멀티 유형 트리는 소위 삼진(ternary)(또는 삼중 트리(triple tree, TT)) 분할들을 나타낼 수 있다. 삼진 분할은 블록을 수직으로 또는 수평으로 3개의 블록들로 나눈다. 수직 TT 분할의 경우에, 블록은 좌측 에지로부터 그의 폭의 1/4에서 그리고 우측 에지로부터 그의 폭의 1/4에서 나뉘고, 수평 TT 분할의 경우에, 블록은 상단 에지로부터 그의 높이의 1/4에서 그리고 하단 에지로부터 그의 높이의 1/4에서 나뉜다.

[0022] 전술된 바와 같이, 각각의 비디오 프레임 또는 픽처는 하나 이상의 영역들로 나뉠 수 있다. 예를 들어, ITU-T H.265에 따르면, 각각의 비디오 프레임 또는 픽처는 하나 이상의 슬라이스들을 포함하도록 분할될 수 있고, 하나 이상의 타일(tile)들을 포함하도록 추가로 분할될 수 있는데, 여기서 각각의 슬라이스는 (예컨대, 래스터 스캔 순서의) CTU들의 시퀀스를 포함하고, 타일은 픽처의 직사각형 영역에 대응하는 CTU들의 시퀀스이다. ITU-T H.265에서, 슬라이스는 독립적인 슬라이스 세그먼트로 시작하여 다음의 독립적인 슬라이스 세그먼트(존재한다면)를 선행하는 모든 후속하는 종속적인 슬라이스 세그먼트들(존재한다면)을 포함하는 하나 이상의 슬라이스 세그먼트들의 시퀀스라는 점에 유의해야 한다. 슬라이스와 같이 슬라이스 세그먼트는 CTU들의

시퀀스이다. 따라서, 일부 경우에, 용어들 "슬라이스" 및 "슬라이스 세그먼트"는 래스터 스캔 순서로 배열된 CTU들의 시퀀스를 나타내기 위해 상호교환가능하게 사용될 수 있다. 추가로, ITU-T H.265에서, 타일은 하나 초과와 슬라이스 내에 포함된 CTU들로 이루어질 수 있고, 슬라이스는 하나 초과와 타일 내에 포함된 CTU들로 이루어질 수 있다는 점에 유의해야 한다. 그러나, ITU-T H.265는 하기 조건들 중 하나 또는 둘 모두가 충족되어야 한다는 것을 규정한다: (1) 슬라이스 내의 모든 CTU들은 동일한 타일에 속한다는 조건; 및 (2) 타일 내의 모든 CTU들은 동일한 슬라이스에 속한다는 조건.

[0023] JVET-T2001과 관련하여, 슬라이스들은 정수 개의 CTU들만으로 이루어지도록 요구되는 대신, 타일 내에 정수 개의 완전 타일들 또는 정수 개의 연속적인 완전 CTU 행들로 이루어지도록 요구된다. JVET-T2001에서, 슬라이스 설계는 슬라이스 세그먼트들을 포함하지 않는다(즉, 독립적/종속적 슬라이스 세그먼트들이 없다)는 점에 유의해야 한다. 따라서, JVET-T2001에서, 픽처는 단일 타일을 포함할 수 있으며, 여기서 단일 타일은 단일 슬라이스 내에 포함되거나 또는 픽처는 다수의 타일들을 포함할 수 있으며, 여기서 다수의 타일들(또는 그의 CTU 행들)은 하나 이상의 슬라이스들 내에 포함될 수 있다. JVET-T2001에서, 타일들로의 픽처의 분할은 타일 행들에 대한 각자의 높이들 및 타일 열들에 대한 각자의 폭들을 특정함으로써 명시된다. 따라서, JVET-T2001에서, 타일은 특정 타일 행 및 특정 타일 컬럼 포지션 내의 CTU들의 직사각형 영역이다. 또한, JVET-T2001은 픽처가 서브픽처들 - 여기서 서브픽처는 픽처 내의 CTU들의 직사각형 영역임 - 로 분할될 수 있는 경우를 규정한다는 점에 유의해야 한다. 서브픽처의 상단-좌측 CTU는 하나 이상의 슬라이스들을 포함하도록 제한되는 서브픽처들을 갖는 픽처 내의 임의의 CTU 포지션에 위치될 수 있다. 따라서, 타일과는 달리, 서브픽처는 반드시 특정 행 및 열 포지션으로 제한되지는 않는다. 서브픽처들은 픽처 내의 관심 영역들을 캡슐화하는 데 유용할 수 있고, 서브-비트스트림 추출 프로세스는 특정 관심 영역만을 디코딩 및 디스플레이하는 데 사용될 수 있다는 점에 유의해야 한다. 즉, 더 상세히 후술되는 바와 같이, 코딩된 비디오 데이터의 비트스트림은 네트워크 추상화 계층(network abstraction layer, NAL) 유닛들의 시퀀스를 포함하며, 여기서 NAL 유닛은 코딩된 비디오 데이터(즉, 픽처의 슬라이스에 대응하는 비디오 데이터)를 캡슐화하거나 또는 NAL 유닛은 비디오 데이터를 디코딩하는 데 사용되는 메타데이터(예컨대, 파라미터 세트)를 캡슐화하고, 서브-비트스트림 추출 프로세스는 비트스트림으로부터 하나 이상의 NAL 유닛들을 제거함으로써 새로운 비트스트림을 형성한다.

[0024] 도 2는 타일들, 슬라이스들, 및 서브픽처들에 따라 분할된 픽처들의 그룹 내의 픽처의 일례를 예시하는 개념도이다. 본 명세서에 기술된 기법들은 타일들, 슬라이스들, 서브픽처들, 이들의 서브분할부들 및/또는 이들에 대한 동등한 구조들에 적용가능할 수 있다는 점에 유의해야 한다. 즉, 본 명세서에 기술된 기법들은 대체적으로, 픽처가 영역들로 어떻게 분할되는지에 관계없이 적용가능할 수 있다. 예를 들어, 일부 경우들에 있어서, 본 명세서에 기술된 기법들은, 타일이 소위 브릭(brick)들로 분할될 수 있는 경우들에 적용가능할 수 있으며, 여기서 브릭은 특정 타일 내의 CTU 행들의 직사각형 영역이다. 또한, 예를 들어, 일부 경우들에 있어서, 본 명세서에 기술된 기법들은 하나 이상의 타일들이 소위 타일 그룹들에 포함될 수 있는 경우들에 적용가능할 수 있으며, 여기서 타일 그룹은 정수 개의 인접 타일들을 포함한다. 도 2에 도시된 예에서, Pic_3 은 16개의 타일들(즉, $Tile_0$ 내지 $Tile_{15}$) 및 3개의 슬라이스들(즉, $Slice_0$ 내지 $Slice_2$)을 포함하는 것으로 예시되어 있다. 도 2에 도시된 예에서, $Slice_0$ 은 4개의 타일들(즉, $Tile_0$ 내지 $Tile_3$)을 포함하고, $Slice_1$ 은 8개의 타일들(즉, $Tile_4$ 내지 $Tile_{11}$)을 포함하고, $Slice_2$ 은 4개의 타일들(즉, $Tile_{12}$ 내지 $Tile_{15}$)을 포함한다. 또한, 도 2의 예에 도시된 바와 같이, Pic_3 은 2개의 서브픽처들(즉, $Subpicture_0$ 및 $Subpicture_1$)을 포함하는 것으로 예시되어 있으며, 여기서 $Subpicture_0$ 은 $Slice_0$ 및 $Slice_1$ 을 포함하고, $Subpicture_1$ 은 $Slice_2$ 를 포함한다. 전술된 바와 같이, 서브픽처들은 픽처 내의 관심 영역들을 캡슐화하는 데 유용할 수 있고, 서브-비트스트림 추출 프로세스는 관심 영역을 선택적으로 디코딩(및 디스플레이)하기 위해 사용될 수 있다. 예를 들어, 도 2를 참조하면, $Subpicture_0$ 은 스포츠 이벤트 프레젠테이션(예컨대, 필드의 뷰)의 액션 부분에 대응할 수 있고, $Subpicture_1$ 은 스포츠 이벤트 프레젠테이션 동안 디스플레이된 스크롤링 배너에 대응할 수 있다. 이러한 방식으로 픽처를 서브픽처들로 조직화함으로써, 뷰어는 스크롤링 배너의 디스플레이를 디스에이블시킬 수 있다. 즉, 서브-비트스트림 추출 프로세스를 통해, $Slice_2$ NAL 유닛은 비트스트림으로부터 제거될 수 있고(따라서 디코딩 및/또는 디스플레이되지 않을 수 있음), $Slice_0$ NAL 유닛 및 $Slice_1$ NAL 유닛은 디코딩 및 디스플레이될 수 있다. 각자의 NAL 유닛 데이터 구조들의 픽처의 슬라이스들의 캡슐화 및 서브-비트스트림 추출이 더 상세히 후술된다.

[0025] 인트라 예측 코딩의 경우, 인트라 예측 모드는 픽처 내의 기준 샘플들의 위치를 특정할 수 있다. ITU-T H.265에서, 정의된 가능한 인트라 예측 모드들은 평면(즉, 표면 피팅) 예측 모드, DC(즉, 편평한 전체 평균) 예측 모

드, 및 33개의 각도 예측 모드들(predMode: 2 내지 34)을 포함한다. JEM에서, 정의된 가능한 인트라 예측 모드들은 평면 예측 모드, DC 예측 모드, 및 65개의 각도 예측 모드들을 포함한다. 평면 및 DC 예측 모드들이 비-방향성 예측 모드들로 지칭될 수 있고 각도 예측 모드들이 방향성 예측 모드들로 지칭될 수 있다는 점에 유의해야 한다. 본 명세서에서 기술되는 기법들은 정의된 가능한 예측 모드들의 수와는 무관하게 일반적으로 적용가능할 수 있다는 점에 유의해야 한다.

[0026] 인터 예측 코딩의 경우, 기준 픽처가 결정되고, 모션 벡터(motion vector, MV)가 현재 비디오 블록에 대한 예측을 생성하는 데 사용되는 기준 픽처 내의 샘플들을 식별한다. 예를 들어, 현재 비디오 블록은 하나 이상의 이전에 코딩된 픽처(들)에 위치한 기준 샘플 값들을 사용하여 예측될 수 있고, 모션 벡터는 현재 비디오 블록에 대한 기준 블록의 위치를 나타내는 데 사용된다. 모션 벡터는, 예를 들어, 모션 벡터의 수평 변위 성분(즉, MV_x), 모션 벡터의 수직 변위 성분(즉, MV_y), 및 모션 벡터에 대한 해상도(예컨대, 1/4 픽셀 정밀도, 1/2 픽셀 정밀도, 1 픽셀 정밀도, 2 픽셀 정밀도, 4 픽셀 정밀도)를 나타낼 수 있다. 현재 픽처 전에 또는 후에 출력된 픽처들을 포함할 수 있는 이전에 디코딩된 픽처들은 기준 픽처 리스트들 중 하나 이상으로 조직화될 수 있고, 기준 픽처 인덱스 값을 사용하여 식별될 수 있다. 추가로, 인터 예측 코딩에서, 단방향 예측은 단일 기준 픽처로부터의 샘플 값들을 사용하여 예측을 생성하는 것을 지칭하고, 양방향 예측은 2개의 기준 픽처들로부터의 각각의 샘플 값들을 사용하여 예측을 생성하는 것을 지칭한다. 즉, 단방향 예측에서는, 단일 기준 픽처 및 대응하는 모션 벡터가 현재 비디오 블록에 대한 예측을 생성하는 데 사용되고, 양방향 예측에서는, 제1 기준 픽처 및 대응하는 제1 모션 벡터 그리고 제2 기준 픽처 및 대응하는 제2 모션 벡터가 현재 비디오 블록에 대한 예측을 생성하는 데 사용된다. 양방향 예측에서, 각각의 샘플 값들은 예측을 생성하기 위해 조합된다(예컨대, 가산되고, 반올림되고, 클립핑되거나, 또는 가중치들에 따라 평균화됨). 픽처들 및 그 영역들은 그 비디오 블록들을 인코딩하기 위해 어느 타입들의 예측 모드들이 활용될 수 있는지에 기초하여 분류될 수 있다. 즉, B 타입을 갖는 영역들(예컨대, B 슬라이스)의 경우, 양방향 예측, 단방향 예측, 및 인트라 예측 모드들이 활용될 수 있고, P 타입을 갖는 영역들(예컨대, P 슬라이스)의 경우, 단방향 예측 및 인트라 예측 모드들이 활용될 수 있고, I 타입을 갖는 영역들(예컨대, I 슬라이스)의 경우, 인트라 예측 모드들만이 활용될 수 있다. 전술된 바와 같이, 기준 픽처들은 기준 인덱스들을 통해 식별된다. 예를 들어, P 슬라이스의 경우, 단일 기준 픽처 리스트 RefPicList0이 있을 수 있고, B 슬라이스의 경우, RefPicList0에 더하여, 제2의 독립적인 기준 픽처 리스트 RefPicList1이 있을 수 있다. B 슬라이스에서의 단방향 예측의 경우, RefPicList0 또는 RefPicList1 중 하나가 예측을 생성하는 데 사용될 수 있다는 점에 유의해야 한다. 또한, 디코딩 프로세스 동안, 픽처를 디코딩하기 시작할 시에, 기준 픽처 리스트(들)는 디코딩된 픽처 버퍼(decoded picture buffer, DPB)에 저장된 이전에 디코딩된 픽처들로부터 생성된다는 점에 유의해야 한다.

[0027] 추가로, 코딩 표준은 다양한 모드들의 모션 벡터 예측을 지원할 수 있다. 모션 벡터 예측은 현재 비디오 블록에 대한 모션 벡터의 값이 다른 모션 벡터에 기초하여 도출되는 것을 가능하게 한다. 예를 들어, 연관된 모션 정보를 갖는 후보 블록들의 세트는 현재 비디오 블록에 대한 공간적 이웃 블록들 및 시간적 이웃 블록들로부터 도출될 수 있다. 추가로, 생성된 (또는 디폴트) 모션 정보는 모션 벡터 예측에 사용될 수 있다. 모션 벡터 예측의 예들은 AMVP(advanced motion vector prediction), TMVP(temporal motion vector prediction), 소위 "병합(merge)" 모드, 및 "스킵(skip)" 및 "직접" 모션 추론을 포함한다. 추가로, 모션 벡터 예측의 다른 예들은 ATMVP(advanced temporal motion vector prediction) 및 STMVP(Spatial-temporal motion vector prediction)를 포함한다. 모션 벡터 예측을 위해, 비디오 인코더 및 비디오 디코더 둘 모두는 동일한 프로세스를 수행하여 후보들의 세트를 도출한다. 따라서, 현재 비디오 블록에 대해, 인코딩 및 디코딩 동안 후보들의 동일한 세트가 생성된다.

[0028] 전술된 바와 같이, 인터 예측 코딩을 위해, 이전에 코딩된 픽처 내의 기준 샘플들은 현재 픽처 내의 비디오 블록들을 코딩하는 데 사용된다. 현재 픽처를 코딩할 때 참조로서 사용하기 위해 이용가능한 이전에 코딩된 픽처들은 기준 픽처들로 지칭된다. 디코딩 순서는 픽처 출력 순서, 즉 비디오 시퀀스에서의 픽처들의 시간적 순서에 반드시 대응하지는 않는다는 점에 유의해야 한다. ITU-T H.265에서, 픽처가 디코딩될 때, 그것은 디코딩된 픽처 버퍼(DPB)(이는 프레임 버퍼, 기준 버퍼, 기준 픽처 버퍼 등으로 지칭될 수 있음)에 저장된다. ITU-T H.265에서, DPB에 저장된 픽처들은 이들이 출력될 때 DPB로부터 제거되고, 후속 픽처들을 코딩하는 데에는 더 이상 필요하지 않다. ITU-T H.265에서, 픽처들이 DPB로부터 제거되어야 하는지의 여부의 결정은, 슬라이스 헤더를 디코딩한 후, 즉 픽처를 디코딩하는 시작 시, 픽처당 1회 적용된다. 예를 들어, 도 2를 참조하면, Pic₂는 Pic₁을 참조하는 것으로 예시되어 있다. 유사하게, Pic₃은 Pic₀을 참조하는 것으로 예시되어 있다. 도 2와 관

련하여, 픽처 번호가 디코딩 순서에 대응한다고 가정하면, DPB는 하기와 같이 채워질 것이다: Pic_0 을 디코딩한 후, DPB는 $\{Pic_0\}$ 을 포함할 것이고; Pic_1 을 디코딩하는 시작 시, DPB는 $\{Pic_0\}$ 을 포함할 것이고; Pic_1 을 디코딩한 후, DPB는 $\{Pic_0, Pic_1\}$ 을 포함할 것이고; Pic_2 를 디코딩하는 시작 시, DPB는 $\{Pic_0, Pic_1\}$ 을 포함할 것이다. 이어서, Pic_2 는 Pic_1 을 참조하여 디코딩될 것이고, Pic_2 를 디코딩한 후, DPB는 $\{Pic_0, Pic_1, Pic_2\}$ 를 포함할 것이다. Pic_3 을 디코딩하는 시작 시, 픽처들 Pic_0 및 Pic_1 은 DPB로부터의 제거를 위해 마킹될 것인데, 이는 그들이 Pic_3 (또는 임의의 후속 픽처들, 도시되지 않음)을 디코딩하는 데 필요하지 않기 때문이며, Pic_1 및 Pic_2 가 출력되었다고 가정하면, DPB는 $\{Pic_0\}$ 을 포함하도록 업데이트될 것이다. 이어서, Pic_3 은 Pic_0 을 참조하여 디코딩될 것이다. DPB로부터 제거하기 위한 픽처들을 마킹하는 프로세스는 기준 픽처 세트(reference picture set, RPS) 관리로 지칭될 수 있다.

[0029] 전술된 바와 같이, 인트라 예측 데이터 또는 인터 예측 데이터는 샘플 값들의 블록에 대한 기준 샘플 값들을 생성하는 데 사용된다. 현재 PU 또는 다른 타입의 픽처 영역 구조에 포함된 샘플 값들과, 연관된 기준 샘플들(예컨대, 예측을 사용하여 생성된 것들) 사이의 차이는 잔차 데이터로 지칭될 수 있다. 잔차 데이터는 비디오 데이터의 각각의 성분에 대응하는 차이 값들의 각각의 어레이들을 포함할 수 있다. 잔차 데이터는 픽셀 도메인 내에 있을 수 있다. 이산 코사인 변환(discrete cosine transform, DCT), 이산 사인 변환(discrete sine transform, DST), 정수 변환(integer transform), 웨이블릿 변환(wavelet transform), 또는 개념적으로 유사한 변환과 같은 변환이 차이 값들의 어레이에 적용되어 변환 계수들을 생성할 수 있다. ITU-T H.265 및 JVET-T2001에서, CU는 CU 레벨에 루트를 갖는 변환 트리 구조와 연관된다는 점에 유의해야 한다. 변환 트리는 하나 이상의 변환 유닛(transform unit, TU)들로 분할된다. 즉, 차이 값들의 어레이는 변환 계수들을 생성할 목적으로 분할될 수 있다(예컨대, 4개의 8×8 변환들이 잔차 값들의 16×16 어레이에 적용될 수 있음). 비디오 데이터의 각각의 성분에 대해, 차이 값들의 그러한 서브분할부들은 변환 블록(Transform Block, TB)들로 지칭될 수 있다. 일부 경우들에 있어서, 변환 계수들을 생성하기 위해 코어 변환 및 후속 2차 변환들이 (비디오 인코더에서) 적용될 수 있다는 점에 유의해야 한다. 비디오 디코더의 경우, 변환 순서가 반전된다.

[0030] 양자화 프로세스는 (예컨대, 그 경우에 있어서, 팔레트 코딩 양자화의) 변환 계수들 또는 잔류 샘플 값들에 대해 직접 수행될 수 있다. 양자화는 특정된 값들의 세트에 제한된 진폭들만큼 변환 계수들을 근사화한다. 양자화는 변환 계수들의 그룹을 표현하는 데 요구되는 데이터의 양을 변경시키기 위해 변환 계수들을 반드시 스케일링한다. 양자화는 (예컨대, 가장 가까운 정수로 반올림하는) 임의의 연관된 반올림 함수들 및 양자화 스케일링 팩터(factor)에 의한 변환 계수들(또는 변환 계수들에 대한 오프셋 값의 덧셈으로부터 생성된 값들)의 나눗셈을 포함할 수 있다. 양자화된 변환 계수들은 계수 레벨 값들로 지칭될 수 있다. 역양자화(inverse quantization)(또는 "탈양자화")는 양자화 스케일링 팩터에 의한 계수 레벨 값들의 곱셈, 및 임의의 역 반올림(reciprocal rounding) 또는 오프셋 덧셈 연산들을 포함할 수 있다. 본 명세서에서 사용되는 바와 같이, 일부 경우에서의 용어 "양자화 프로세스"는, 일부 경우에, 레벨 값들을 생성하기 위한 스케일링 팩터에 의한 나눗셈 및 변환 계수들을 복구하기 위한 스케일링 팩터에 의한 곱셈을 지칭할 수 있다는 점에 유의해야 한다. 즉, 양자화 프로세스는, 일부 경우에는 양자화를 그리고 일부 경우에는 역양자화를 지칭할 수 있다. 또한, 아래의 예들 중 일부에서, 십진수 표기와 연관된 산술 연산들과 관련하여 양자화 프로세스들이 기술되지만, 그러한 설명들은 예시의 목적을 위한 것이고 제한하는 것으로 해석해서는 안 된다는 점에 유의해야 한다. 예를 들어, 본 명세서에서 기술되는 기법들은 이진 연산들 등을 사용하여 디바이스에서 구현될 수 있다. 예를 들어, 본 명세서에서 기술되는 곱셈 및 나눗셈 연산들은 비트 시프트 연산들 등을 사용하여 구현될 수 있다.

[0031] 양자화된 변환 계수들 및 선택스 요소들(예컨대, 비디오 블록에 대한 코딩 구조를 표시하는 선택스 요소들)은 엔트로피 코딩 기법에 따라 엔트로피 코딩될 수 있다. 엔트로피 코딩 프로세스는 무손실 데이터 압축 알고리즘들을 사용하여 선택스 요소들의 값들을 코딩하는 것을 포함한다. 엔트로피 코딩 기법들의 예들은 콘텐츠 적응적 가변 길이 코딩(content adaptive variable length coding, CAVLC), 콘텍스트 적응적 이진 산술 코딩(context adaptive binary arithmetic coding, CABAC), 확률 간격 분할 엔트로피 코딩(probability interval partitioning entropy coding, PIPE) 등을 포함한다. 엔트로피 인코딩된 양자화된 변환 계수들 및 대응하는 엔트로피 인코딩된 선택스 요소들은 비디오 디코더에서 비디오 데이터를 재생하는 데 사용될 수 있는 컴플라이언트 비트스트림을 형성할 수 있다. 엔트로피 코딩 프로세스, 예를 들어 CABAC는 선택스 요소들에 대해 이진화를 수행하는 것을 포함할 수 있다. 이진화는 선택스 요소의 값을 일련의 하나 이상의 비트들로 변환하는 프로세스를 지칭한다. 이러한 비트들은 "빈(bin)"들로 지칭될 수 있다. 이진화는 하기 코딩 기법들 중 하나 또는 이들의 조합을 포함할 수 있다: 고정 길이 코딩(fixed length coding), 일진 코딩(unary coding), 트런케이트

형 일진 코딩(truncated unary coding), 트런케이트형 라이스 코딩(truncated Rice coding), 골롬 코딩(Golomb coding), k-차 지수 골롬 코딩(k-th order exponential Golomb coding), 및 골롬-라이스 코딩(Golomb-Rice coding). 예를 들어, 이진화는, 8 비트 고정 길이 이진화 기법을 이용하여 선택스 요소에 대한 5의 정수 값을 00000101로서 표현하는 것 또는 일진 코딩 이진화 기법을 이용하여 5의 정수 값을 11110으로서 표현하는 것을 포함할 수 있다. 본 명세서에 사용되는 바와 같이, 용어들 "고정 길이 코딩", "일진 코딩", "트런케이트형 일진 코딩", "트런케이트형 라이스 코딩", "골롬 코딩", "k-차 지수 골롬 코딩", 및 "골롬-라이스 코딩" 각각은 이러한 코딩 기법의 일반적인 구현 및/또는 이러한 코딩 기법의 더 구체적인 구현을 지칭할 수 있다. 예를 들어, Golomb-Rice 코딩 구현은 비디오 코딩 표준에 따라 구체적으로 정의될 수 있다. CABAC의 예에서, 특정 빈에 대해, 콘텍스트는 빈에 대한 최고 가능성 상태(most probable state, MPS) 값(즉, 빈에 대한 MPS는 0 또는 1 중 하나임) 및 빈이 MPS 또는 최소 가능성 상태(least probable state, LPS)일 확률 값을 제공한다. 예를 들어, 콘텍스트는, 빈의 MPS가 0이고 빈이 1일 확률이 0.3임을 나타낼 수 있다. 현재 선택스 요소 및 이전에 코딩된 선택스 요소들에서의 빈들을 포함하는 이전에 코딩된 빈들의 값들에 기초하여 콘텍스트가 결정될 수 있다는 점에 유의해야 한다. 예를 들어, 이웃 비디오 블록들과 연관된 선택스 요소들의 값들은 현재 빈에 대한 콘텍스트를 결정하는 데 사용될 수 있다.

[0032]

전술된 바와 같이, 재구성된 블록의 샘플 값들은 인코딩되는 현재 비디오 블록의 샘플 값들과는 상이할 수 있다. 또한, 일부 경우들에 있어서, 블록 단위로 비디오 데이터를 코딩하는 것은 아티팩트들(예컨대, 소위 차단 아티팩트들, 밴딩 아티팩트들 등)을 초래할 수 있다는 점에 유의해야 한다. 예를 들어, 차단 아티팩트들은 재구성된 비디오 데이터의 코딩 블록 경계들이 사용자에게 시각적으로 인지가 가능하게 할 수 있다. 이러한 방식으로, 재구성된 샘플 값들은, 인코딩되는 현재 비디오 블록의 샘플 값들과 재구성된 블록 사이의 차이를 최소화하고/하거나 비디오 코딩 프로세스에 의해 도입된 아티팩트들을 최소화하도록 수정될 수 있다. 그러한 수정들은 대체적으로 필터링으로 지칭될 수 있다. 필터링은 인-루프 필터링 프로세스 또는 포스트-루프(또는 포스트-필터링) 필터링 프로세스의 일부로서 발생할 수 있다는 점에 유의해야 한다. 인-루프 필터링 프로세스를 위해, 필터링 프로세스의 생성된 샘플 값들은 예측 비디오 블록들(예컨대, 비디오 인코더에서의 후속 인코딩 및 비디오 디코더에서의 후속 디코딩을 위해 기준 프레임 버퍼에 저장됨)에 사용될 수 있다. 포스트-루프 필터링 프로세스를 위해, 필터링 프로세스의 생성된 샘플 값들은 단지, 디코딩 프로세스의 일부로서 출력된다(예컨대, 후속 코딩에 사용되지 않음). 예를 들어, 비디오 디코더의 경우, 인-루프 필터링 프로세스를 위해, 재구성된 블록을 필터링하는 것으로부터 생성된 샘플 값들은 후속 디코딩에 사용될 것이고(예컨대, 기준 버퍼에 저장됨), (예컨대, 디스플레이에) 출력될 것이다. 포스트-루프 필터링 프로세스를 위해, 재구성된 블록은 후속 디코딩을 위해 사용될 것이고, 재구성된 블록을 필터링하는 것으로부터 생성된 샘플 값들이 출력될 것이고 후속 디코딩에 사용되지 않을 것이다.

[0033]

디블록킹(또는 디-블록킹), 디블록 필터링(deblock filtering), 또는 디블록킹 필터 적용은 이웃하는 재구성된 비디오 블록들의 경계를 평활화시키는 프로세스(즉, 경계가 관찰자에게 덜 인지가 가능해지게 함)를 지칭한다. 이웃하는 재구성된 비디오 블록들의 경계를 평활화시키는 것은 경계에 인접한 행들 또는 열들에 포함된 샘플 값들을 수정하는 것을 포함할 수 있다. JVET-T2001은 인-루프 필터링 프로세스의 일부로서 재구성된 샘플 값들에 디블록킹 필터가 적용되는 경우를 규정한다. 인-루프 필터링 프로세스의 일부로서 디블록킹 필터를 적용하는 것에 더하여, JVET-T2001은 인-루프 필터링 프로세스에서 샘플 적응적 오프셋(Sample Adaptive Offset, SAO) 필터링이 적용될 수 있는 경우를 규정한다. 대체적으로, SAO는 오프셋 값을 조건부로 더함으로써 영역 내의 디블록킹된 샘플 값들을 수정하는 프로세스이다. 다른 유형의 필터링 프로세스는 소위 적응적 루프 필터(adaptive loop filter, ALF)를 포함한다. 블록 기반 적응을 갖는 ALF가 JEM에서 명시된다. JEM에서, ALF는 SAO 필터 후에 적용된다. ALF는 다른 필터링 기법들과는 독립적으로 재구성된 샘플들에 적용될 수 있다는 점에 유의해야 한다. 비디오 인코더에서의 JEM에서 명시된 ALF를 적용하기 위한 프로세스는 하기와 같이 요약될 수 있다: (1) 재구성된 픽처에 대한 루마 성분의 각각의 2x2 블록이 분류 인덱스에 따라 분류되고; (2) 필터 계수들의 세트들이 각각의 분류 인덱스에 대해 도출되고; (3) 루마 성분들에 대해 필터링 결정들이 결정되고; (4) 크로마 성분들에 대해 필터링 결정이 결정되고; (5) 필터 파라미터들(예컨대, 계수들 및 결정들)이 시그널링된다. JVET-T2001은 디블록킹, SAO, 및 ALF 필터들을 명시하는데, 이들은 대체적으로 ITU-T H.265 및 JEM에서 제공되는 디블록킹, SAO, 및 ALF 필터들에 기초하는 것으로 설명될 수 있다.

[0034]

JVET-T2001은 ITU-T H.266의 사전공개 버전으로 지칭되고, 따라서, VVC 프로젝트로부터 생성된 비디오 코딩 표준의 거의 완성된 초안이며, 따라서, VVC 표준의 제1 버전(또는 VVC 또는 VVC 버전 1 또는 ITU-H.266)으로 지칭될 수 있다는 점에 유의해야 한다. VVC 프로젝트 동안, 아티팩트 제거 및 객관적인 품질 개선의 잠재력을 보여주는 컨볼루션 신경 네트워크(Convolutional Neural Network, CNN) 기반 기법들이 조사되었지만, VVC 표준에

그러한 기법들을 포함하지 않는다는 것이 결정되었다는 점에 유의해야 한다. 그러나, 현재, VVC에 대한 확장들 및/또는 개선들에 대해 CNN 기반 기법들이 고려되고 있다. 일부 CNN 기반 기법들은 포스트-필터링에 관한 것이다. 예를 들어, 문헌["AHG11: Content-adaptive neural network post-filter", 26th Meeting of ISO/IEC JTC1/SC29/WG11 20-29 April 2022, Teleconference, document JVET-Z0082-v2](본 명세서에서 JVET-Z0082로 지칭됨)은 콘텐츠 적응적 신경 네트워크 기반 포스트-필터를 설명한다. JVET-Z0082에서, 테스트 비디오 상에 NN 포스트-필터를 오버피팅함으로써 콘텐츠 적응이 달성된다는 점에 유의해야 한다. 또한, JVET-Z0082에서의 오버피팅 프로세스의 결과는 가중치 업데이트를 유의해야 한다. JVET-Z0082는 가중치 업데이트가 ISO/IEC FDIS 15938-17에 따라 코딩되는 경우를 설명한다. 정보 기술 - 멀티미디어 콘텐츠 설명 인터페이스 - 부분 17: 멀티미디어 콘텐츠 설명 및 분석을 위한 신경 네트워크들의 압축 및 멀티미디어 콘텐츠 설명 및 분석을 위한 신경 네트워크들의 점진적 압축의 테스트 모델(Test Model of Incremental Compression of Neural Networks for Multimedia Content Description and Analysis, INCTM)(N0179, February 2022) - 이는 MPEG NNR(Neural Network Representation) 또는 신경 네트워크 코딩(Neural Network Coding, NNC) 표준으로 총칭될 수 있음. JVET-Z0082는, 코딩된 가중치 업데이트가 NNR 포스트-필터 SEI 메시지로써 비디오 비트스트림 내에서 시그널링되는 경우를 추가로 설명한다. 문헌["AHG9: NNR post-filter SEI message", 26th Meeting of ISO/IEC JTC1/SC29/WG11 20-29 April 2022, Teleconference, document JVET-Z0052-v1](본 명세서에서 JVET-Z0052로 지칭됨)은 JVET-Z0082에 의해 활용되는 NNR 포스트-필터 SEI 메시지를 설명한다. JVET-Z0082에서 설명된 NN 포스트-필터의 요소들 및 JVET-Z0052에서 설명된 NNR 포스트-필터 SEI 메시지는 문헌["Additional SEI messages for VSEI (Draft 2)", 27th Meeting of ISO/IEC JTC1/SC29/WG11 13-22 July 2022, Teleconference, document JVET-AA2006-v2](본 명세서에서 JVET-AA2006로 지칭됨)에서 채택되었다. JVET-AA2006은 코딩된 비디오 비트스트림들에 대한 범용 보충 향상 정보(versatile supplemental enhancement information, VSEI) 메시지들을 규정한다. JVET-AA2006은 신경 네트워크 포스트-필터 특성 SEI 메시지에 대한 그리고 신경 네트워크 포스트-필터 활성화 SEI 메시지에 대한 선택 및 시맨틱들을 명시한다. 신경 네트워크 포스트-필터 특성 SEI 메시지는 포스트-프로세싱 필터로서 사용될 수 있는 신경 네트워크를 명시한다. 특정 픽처들을 위한 특정된 포스트-프로세싱 필터들의 사용은 신경 네트워크 포스트-필터 활성화 SEI 메시지들로 표시된다. 또한, 문헌["Information technology - MPEG video technologies - Part 7: Versatile supplemental enhancement information messages for coded video bitstreams, AMENDMENT 1: Additional SEI messages" 28th Meeting of ISO/IEC JTC1/SC29/WG5 9, November 2022, Mainz, DE, document JVET-AB2006, m61498](본 명세서에서 JVET-AB2006)로 지칭됨). JVET-AB2006은 더 상세히 후술된다. 본 명세서에 설명된 기법들은 신경 네트워크 포스트-필터 메시지들의 시그널링을 위한 기법들을 제공한다.

[0035] 본 명세서에 사용되는 수학식들에 관하여, 하기의 산술 연산자들이 사용될 수 있다:

+ 덧셈
 - 뺄셈
 * 곱셈, 매트릭스 곱셈을 포함함
 x^y 지수화, x를 y번 거듭제곱하는 것을 명시한다. 다른 맥락들에서, 그러한 표기법은 지수로서의 해석을 위해 의도되지 않는 위첨자화(superscripting)에 사용된다.
 / 결과가 0으로 트런케이팅되는 정수 나눗셈. 예를 들어, $7/4$ 및 $-7/4$ 는 1로 트런케이팅되고, $-7/4$ 및 $7/-4$ 는 1로 트런케이팅된다.
 $\div$ 트런케이팅 또는 반올림이 의도되지 않는 수학식들에서 나눗셈을 나타내는 데 사용된다.
 $\frac{x}{y}$ 트런케이팅 또는 반올림이 의도되지 않는 수학식들에서 나눗셈을 나타내는 데 사용된다.

[0036]

[0037] 또한, 하기의 수학 함수들이 사용될 수 있다:

[0038] $\text{Log}_2(x)$ 밑이 2인 x의 로그;

$$\text{Min}(x, y) = \begin{cases} x & ; x \leq y \\ y & ; x > y \end{cases};$$

[0039]

$$\text{Max}(x, y) = \begin{cases} x & ; x \geq y \\ y & ; x < y \end{cases}$$

[0040]

[0041] $\text{Ceil}(x)$ x 이상인 최소 정수.

[0042] 본 명세서에서 사용되는 예시적인 선택스와 관련하여, 논리 연산자들의 하기의 정의들이 적용될 수 있다:

- [0043] $x \ \&\& \ y$ x 와 y 의 불 논리 "곱"(Boolean logical "and")
- [0044] $x|y$ x 와 y 의 불 논리 "합"(Boolean logical "or")
- [0045] $!$ 불 논리 "부정"(Boolean logical "not")
- [0046] $x? y:z$ x 가 참이거나 0과 동일하지 않으면, y 의 값으로 평가되고; 그렇지 않으면, z 의 값으로 평가된다.
- [0047] 추가로, 하기의 관계 연산자들이 사용될 수 있다:
- | | |
|------|---------|
| $>$ | 초과 |
| $>=$ | 이상 |
| $<$ | 미만 |
| $<=$ | 이하 |
| $==$ | 동일 |
| $!=$ | 동일하지 않음 |
- [0048]
- [0049] 또한, 본 명세서에서 사용되는 선택스 디스크립터들 중에서, 하기의 디스크립터들이 적용될 수 있다는 점에 유의해야 한다:
- [0050] - $b(8)$: 임의의 패턴의 비트 스트링을 갖는 바이트(8-비트) 이러한 디스크립터에 대한 파싱 프로세스는 함수 $read_bits(8)$ 의 반환 값에 의해 명시된다.
- [0051] - $f(n)$: 좌측 비트가 먼저인 (좌측으로부터 우측으로) 기록된 n 개의 비트들을 사용하는 고정 패턴 비트 스트링. 이러한 디스크립터에 대한 파싱 프로세스는 함수 $read_bits(n)$ 의 반환 값에 의해 명시된다.
- [0052] - $se(v)$: 좌측 비트가 먼저인 부호있는 정수 0차 Exp-Golomb-코딩된 선택스 요소.
- [0053] - $tb(v)$: 최대 $maxVal$ 개의 비트들을 사용하는 트런케이티된 이진수(truncated binary)(이때 $maxVal$ 은 선택스 요소의 시맨틱들에서 정의됨)
- [0054] - $tu(v)$: 최대 $maxVal$ 개의 비트들을 사용하는 트런케이티된 단항(truncated unary)(이때 $maxVal$ 은 선택스 요소의 시맨틱들에서 정의됨)
- [0055] - $u(n)$: n 개의 비트들을 사용하는 부호없는 정수. n 이 선택스 테이블에서 " v "일 때, 비트들의 수는 다른 선택스 요소들의 값에 종속적인 방식으로 변한다. 이러한 디스크립터에 대한 파싱 프로세스는 먼저 기록된 최상위 비트를 갖는 부호없는 정수의 이진수 표현으로서 해석되는 함수 $read_bits(n)$ 의 반환 값에 의해 명시된다.
- [0056] - $ue(v)$: 좌측 비트가 먼저인 부호없는 정수 0차 Exp-Golomb-코딩된 선택스 요소.
- [0057] 전술된 바와 같이, 비디오 콘텐츠는 일련의 픽처들로 구성된 비디오 시퀀스들을 포함하고, 각각의 픽처는 하나 이상의 영역들로 나뉠 수 있다. JVET-T2001에서, 픽처의 코딩된 표현은 AU 내에 특정 계층의 비디오 코딩 계층(video coding layer, VCL) NAL 유닛들을 포함하고, 픽처의 모든 CTU들을 포함한다. 예를 들어, 다시 도 2를 참조하면, Pic_3 의 코딩된 표현이 3개의 코딩된 슬라이스 NAL 유닛들(즉, $Slice_0$ NAL 유닛, $Slice_1$ NAL 유닛, 및 $Slice_2$ NAL 유닛)에 캡슐화된다. 용어 비디오 코딩 계층(VCL) NAL 유닛은 코딩된 슬라이스 NAL 유닛들에 대한 집합적 용어로서 사용되며, 즉 VCL NAL은 모든 유형들의 슬라이스 NAL 유닛들을 포함하는 집합적 용어라는 점에 유의해야 한다. 전술된 바와 같이, 그리고 더 상세히 후술되는 바와 같이, NAL 유닛은 비디오 데이터를 디코딩하는 데 사용되는 메타데이터를 캡슐화할 수 있다. 비디오 시퀀스를 디코딩하는 데 사용되는 메타데이터를 캡슐화하는 NAL 유닛은 대체적으로 비-VCL NAL 유닛으로 지칭된다. 따라서, JVET-T2001에서, NAL 유닛은 VCL NAL 유닛 또는 비-VCL NAL 유닛일 수 있다. VCL NAL 유닛은 특정 슬라이스를 디코딩하는 데 사용되는 정보를 제공하는 슬라이스 헤더 데이터를 포함한다는 점에 유의해야 한다. 따라서, JVET-T2001에서, 일부 경우들에 있어서 메타데이터로 지칭될 수 있는 비디오 데이터를 디코딩하는 데 사용되는 정보는 비-VCL NAL 유닛들에 포함되는 것으로 제한되지 않는다. JVET-T2001은, 픽처 유닛(picture unit, PU)이, 명시된 분류 규칙에 따라 서로 연관되고 디코딩 순서에서 연속적이고 정확히 하나의 코딩된 픽처를 포함하는 NAL 유닛들의 세트인 경우, 및 액세스 유닛(access unit, AU)이 상이한 계층들에 속하고 DPB로부터의 출력을 위해 동일한 시간과 연관된 코딩된 픽처들을 포함하는 PU들의 세트인 경우를 규정한다. JVET-T2001은, 일정 계층이, 연관된 비-VCL NAL 유닛들 및 계층 식별자의 특정 값을 모두가 갖는 VCL NAL 유닛들의 세트인 경우를 추가로 규정한다. 또한, JVET-T2001에서, PU는 0개 또는 1개의 PH NAL 유닛들, 하나 이상의 VCL NAL 유닛들을 포함하는 하나의 코딩된 픽처, 및 0개

이상의 다른 비-VCL NAL 유닛들로 이루어진다. 또한, JVET-T2001에서, 코딩된 비디오 시퀀스(coded video sequence, CVS)는, 디코딩 순서에서, 코딩된 비디오 시퀀스 시작(coded video sequence start, CVSS) AU 및 뒤이어, 모든 후속 AU들을 포함하지만 CVSS AU인 임의의 후속 AU까지는 포함하지 않는 CVSS AU들이 아닌 0개 이상의 AU들로 이루어진 AU들의 시퀀스이며, 여기서 코딩된 비디오 시퀀스 시작(CVSS) AU는 CVS 내의 각각의 계층에 대해 PU가 있는 AU이고, 각각의 현재 픽처 유닛 내의 코딩된 픽처는 코딩된 계층 비디오 시퀀스 시작(coded layer video sequence start, CLVSS) 픽처이다. JVET-T2001에서, 코딩된 계층 비디오 시퀀스(coded layer video sequence, CLVS)는, 디코딩 순서에서, CLVSS PU 및 뒤이어, 모든 후속 PU들을 포함하지만 CLVSS PU인 임의의 후속 PU까지는 포함하지 않는 CLVSS AU들이 아닌 0개 이상의 PU들로 이루어진 동일한 계층 내의 PU들의 시퀀스이다. 이는, JVET-T2001에서, 비트스트림이 하나 이상의 CVS들을 형성하는 AU들의 시퀀스를 포함하는 것으로 기술될 수 있다는 것이다.

[0058] 다계층 비디오 코딩은, 비디오 프레젠테이션이, 비디오 데이터의 기초 계층에 대응하는 프레젠테이션으로서 디코딩/디스플레이되는 것, 및 비디오 데이터의 향상 계층들에 대응하는 하나 이상의 추가적인 프리젠테이션들로서 디코딩/디스플레이되는 것을 가능하게 한다. 예를 들어, 기초 계층은 기본 레벨의 품질(예컨대, 고화질 렌더링 및/또는 30 Hz 프레임 레이트)을 갖는 비디오 프레젠테이션이 제시되는 것을 가능하게 할 수 있고, 향상 계층은 향상된 레벨의 품질(예컨대, 초고화질 렌더링 및/또는 60 Hz 프레임 레이트)을 갖는 비디오 프레젠테이션이 제시되는 것을 가능하게 할 수 있다. 향상 계층은 기초 계층을 참조하여 코딩될 수 있다. 즉, 예를 들어, 향상 계층 내의 픽처는 기초 계층 내의 하나 이상의 픽처들(그의 스케일링된 버전들을 포함함)을 참조함으로써 (예컨대, 계층간 예측 기법들을 사용하여) 코딩될 수 있다. 계층들은 또한, 서로 독립적으로 코딩될 수 있다는 점에 유의해야 한다. 이러한 경우, 2개의 계층들 사이에 계층간 예측이 없을 수 있다. 각각의 NAL 유닛은 NAL 유닛이 연관된 비디오 데이터의 계층을 나타내는 식별자를 포함할 수 있다. 전송된 바와 같이, 서브-비트스트림 추출 프로세스는 픽처의 특정 관심 영역만을 디코딩 및 디스플레이하는 데 사용될 수 있다. 또한, 서브-비트스트림 추출 프로세스는 비디오의 특정 계층만을 디코딩 및 디스플레이하는 데 사용될 수 있다. 서브-비트스트림 추출은, 컴플라이언트 또는 순응형 비트스트림을 수신하는 디바이스가 수신된 비트스트림에서 데이터를 폐기 및/또는 수정함으로써 새로운 컴플라이언트 또는 순응형 비트스트림을 형성하는 프로세스를 지칭할 수 있다. 예를 들어, 서브-비트스트림 추출은 비디오의 특정 표현(예컨대, 고품질 표현)에 대응하는 새로운 컴플라이언트 또는 순응형 비트스트림을 형성하는 데 사용될 수 있다.

[0059] JVET-T2001에서, 비디오 시퀀스, GOP, 픽처, 슬라이스, 및 CTU 각각은 비디오 코딩 속성들을 설명하는 메타데이터 및 비-VCL NAL 유닛들에 캡슐화된 일부 유형들의 메타데이터와 연관될 수 있다. JVET-T2001은 비디오 데이터 및/또는 비디오 코딩 속성들을 설명하는 데 사용될 수 있는 파라미터들 세트들을 정의한다. 특히, JVET-T2001은 하기의 4가지 유형들의 파라미터 세트들, 즉, 비디오 파라미터 세트(video parameter set, VPS), 시퀀스 파라미터 세트(sequence parameter set, SPS), 픽처 파라미터 세트(picture parameter set, PPS), 및 적응 파라미터 세트(adaption parameter set, APS)를 포함하며, 여기서 SPS는 0개 이상의 전체 CVS들에 적용되고, PPS는 0개 이상의 전체 코딩된 픽처들에 적용되고, APS는 0개 이상의 슬라이스들에 적용되고, VPS는 SPS에 의해 선택적으로 참조될 수 있다. PPS는 그것을 참조하는 개별 코딩된 픽처에 적용된다. JVET-T2001에서, 파라미터 세트들은 비-VCL NAL 유닛으로서 캡슐화될 수 있고/있거나 메시지로써 시그널링될 수 있다. JVET-T2001은 또한, 비-VCL NAL 유닛으로서 캡슐화된 픽처 헤더(picture header, PH)를 포함한다. JVET-T2001에서, 픽처 헤더는 코딩된 픽처의 모든 슬라이스들에 적용된다. JVET-T2001은 추가로, 디코딩 능력 정보(decoding capability information, DCI) 및 보충 향상 정보(supplemental enhancement information, SEI) 메시지들이 시그널링되는 것을 가능하게 한다. JVET-T2001에서, DCI 및 SEI 메시지들은 디코딩, 디스플레이, 또는 다른 목적들과 관련된 프로세스들을 보조하지만, DCI 및 SEI 메시지들은 디코딩 프로세스에 따라 루마 또는 크로마 샘플들을 구성하는 데 필요하지 않을 수 있다. JVET-T2001에서, DCI 및 SEI 메시지들은 비-VCL NAL 유닛들을 사용하여 비트스트림에서 시그널링될 수 있다. 또한, DCI 및 SEI 메시지들은 비트스트림에 존재하는 것 이외의 일부 메커니즘에 의해 전달될 수 있다(즉, 대역외 시그널링됨).

[0060] 도 3은 다수의 CVS들을 포함하는 비트스트림의 일례를 도시하는데, 여기서 CVS는 AU들을 포함하고, AU들은 픽처 유닛들을 포함한다. 도 3에 도시된 예는 도 2의 예에 예시된 슬라이스 NAL 유닛들을 비트스트림에 캡슐화하는 일례에 대응한다. 도 3에 도시된 예에서, Pic₃에 대한 대응하는 픽처 유닛은 3개의 VCL NAL 코딩된 슬라이스 NAL 유닛들, 즉 Slice₀ NAL 유닛, Slice₁ NAL 유닛, 및 Slice₂ NAL 유닛, 및 2개의 비-VCL NAL 유닛들, 즉, PPS NAL 유닛 및 PH NAL 유닛을 포함한다. 도 3에서, HEADER는 NAL 유닛 헤더(즉, 슬라이스 헤더와 혼동되어서는 안 됨)라는 점에 유의해야 한다. 또한, 도 3에서, 예시되지 않은 다른 비-VCL NAL 유닛들은 CVS들, 예컨대,

SPS NAL 유닛들, VPS NAL 유닛들, SEI 메시지 NAL 유닛들 등에 포함될 수 있다는 점에 유의해야 한다. 또한, 다른 예들에서, Pic_3 을 디코딩하는 데 사용되는 PPS NAL 유닛은 비트스트림 내의, 예컨대 Pic_0 에 대응하는 픽처 유닛 중의 어딘가에 포함될 수 있거나 또는 외부 메커니즘에 의해 제공될 수 있다는 점에 유의해야 한다. 더 상세히 후술되는 바와 같이, JVET-T2001에서, PH 선택스 구조는 VCL NAL 유닛의 슬라이스 헤더에 또는 현재 PU의 PH NAL 유닛에 존재할 수 있다.

[0061] JVET-T2001은 NAL 유닛에 포함된 원시 바이트 시퀀스 페이로드(Raw Byte Sequence Payload, RBSP) 데이터 구조의 유형을 특징하는 NAL 유닛 헤더 시맨틱들을 정의한다. 표 1은 JVET-T2001에서 규정된 NAL 유닛 헤더의 선택스를 예시한다.

[0062] [표 1]

nal_unit_header() {	디스크립터
forbidden_zero_bit	f(1)
nuh_reserved_zero_bit	u(1)
nuh_layer_id	u(6)
nal_unit_type	u(5)
nuh_temporal_id_plus1	u(3)
}	

[0063]

[0064] JVET-T2001은 표 1에 예시된 각자의 선택스 요소들에 대한 하기의 정의들을 규정한다.

[0065] **forbidden_zero_bit**는 0과 동일할 것이다.

[0066] **nuh_reserved_zero_bit**는 0과 동일해야 한다. **nuh_reserved_zero_bit**의 값 1은 ITU-T | ISO/IEC에 의해 향후에 특정될 수 있다. **nuh_reserved_zero_bit**의 값이 이러한 규격의 이러한 버전에서 0과 동일하도록 요구되지만, 이러한 규격의 이러한 버전에 순응하는 디코더들은 1과 동일한 **nuh_reserved_zero_bit**의 값이 선택스에 나타날 수 있게 할 것이고, 1과 동일한 **nuh_reserved_zero_bit**를 갖는 NAL 유닛들을 무시할 것이다(즉, 비트스트림으로부터 제거하고 폐기할 것임).

[0067] **nuh_layer_id**는 VCL NAL 유닛이 속하는 계층의 식별자 또는 비-VCL NAL 유닛이 적용되는 계층의 식별자를 특정한다. **nuh_layer_id**의 값은 0 내지 55(이를 포함함)의 범위 내에 있을 것이다. **nuh_layer_id**에 대한 다른 값들은 ITU-T | ISO/IEC에 의한 향후 사용을 위해 예약된다. **nuh_layer_id**의 값이 이러한 규격의 이러한 버전에서 0 내지 55(이를 포함함)의 범위가 되도록 요구되지만, 이러한 규격의 이러한 버전에 순응하는 디코더들은 55 초과인 **nuh_layer_id**의 값이 선택스에 나타날 수 있게 할 것이고, 55 초과인 **nuh_layer_id**를 갖는 NAL 유닛들을 무시할 것이다(즉, 비트스트림으로부터 제거하고 폐기할 것임).

[0068] **nuh_layer_id**의 값은 코딩된 픽처의 모든 VCL NAL 유닛들에 대해 동일할 것이다. 코딩된 픽처 또는 PU의 **nuh_layer_id**의 값은 코딩된 픽처 또는 PU의 VCL NAL 유닛들의 **nuh_layer_id**의 값이다.

[0069] **nal_unit_type**이 PH_NUT 또는 FD_NUT와 동일할 때, **nuh_layer_id**는 연관된 VCL NAL 유닛의 **nuh_layer_id**와 동일할 것이다.

[0070] **nal_unit_type**이 EOS_NUT과 동일할 때, **nuh_layer_id**는 CVS에 존재하는 계층들의 **nuh_layer_id** 값들 중 하나와 동일할 것이다.

[0071] 주 - DCI, OPI, VPS, AUD, 및 EOB 유닛들에 대한 **nuh_layer_id**의 값은 제한되지 않는다.

[0072] **nuh_temporal_id_plus1** - 1은 NAL 유닛에 대한 임시 식별자를 특정한다.

[0073] **nuh_temporal_id_plus1**의 값은 0과 동일하지 않을 것이다.

[0074] 변수 TemporalId는 하기와 같이 도출된다:

[0075] $TemporalId = nuh_temporal_id_plus1 - 1$

[0076] **nal_unit_type**이 IDR_W_RADL 내지 RSV_IRAP_11(이를 포함함)의 범위 내에 있을 때, TemporalId는 0과 동일할 것이다.

[0077] **nal_unit_type**이 STSA_NUT와 동일하고 **vps_independent_layer_flag[GeneralLayerIdx[nuh_layer_id]]**가 1과 동일할 때, TemporalId는 0 초과일 것이다.

[0078] TemporalId의 값은 AU의 모든 VCL NAL 유닛들에 대해 동일할 것이다. 코딩된 픽처, PU, 또는 AU의 TemporalId

의 값은 코딩된 픽처, PU, 또는 AU의 VCL NAL 유닛들의 TemporalId의 값이다. 서브계층 표현의 TemporalId의 값은 서브계층 표현에서 모든 VCL NAL 유닛들의 TemporalId의 최대 값이다.

- [0079] 비-VCL NAL 유닛들에 대한 TemporalId의 값은 하기와 같이 제한된다:
- [0080] - nal_unit_type이 DCI_NUT, OPI_NUT, VPS_NUT, 또는 SPS_NUT과 동일한 경우, TemporalId는 0과 동일할 것이고, ANL 유닛을 포함하는 AU의 TemporalId는 0과 동일할 것이다.
- [0081] - 이와 달리, nal_unit_type이 PH_NUT와 동일한 경우, TemporalId는 NAL 유닛을 포함하는 PU의 TemporalId와 동일할 것이다.
- [0082] - 이와 달리, nal_unit_type이 EOS_NUT 또는 EOB_NUT와 동일한 경우, TemporalId는 0과 동일할 것이다.
- [0083] - 이와 달리, nal_unit_type이 AUD_NUT, FD_NUT, PREFIX_SEI_NUT, 또는 SUFFIX_SEI_NUT와 동일한 경우, TemporalId는 NAL 유닛을 포함하는 AU의 TemporalId와 동일할 것이다.
- [0084] - 이와 달리, nal_unit_type이 PPS_NUT, PREFIX_APS_NUT, 또는 SUFFIX_APS_NUT와 동일할 때, TemporalId는 NAL 유닛을 포함하는 PU의 TemporalId 이상일 것이다.
- [0085] 주 - NAL 유닛이 비-VCL NAL 유닛일 때, TemporalId의 값은 비-VCL NAL 유닛이 적용되는 모든 AU들의 TemporalId 값들의 최소 값과 동일하다. nal_unit_type이 PPS_NUT, PREFIX_APS_NUT, 또는 SUFFIX_APS_NUT와 동일할 때, TemporalId는 포함하는 AU의 TemporalId 이상일 수 있는데, 이는 모든 PPS들 및 APS들이 (예컨대, 이들이 대역외로 수송되고 수신자가 이들을 비트스트림의 시작부에 놓을 때) 비트스트림의 시작부에 포함될 수 있기 때문이며, 여기서 제1 코딩된 픽처는 0과 동일한 TemporalId를 갖는다.
- [0086] **nal_unit_type**은 NAL 유닛 유형, 즉 표 2에 특정된 바와 같이 NAL 유닛에 포함된 Rbsp 데이터 구조의 유형을 명시한다.
- [0087] 시맨틱들이 명시되지 않는, UNSPEC28..UNSPEC31(이를 포함함)의 범위 내의 nal_unit_type을 갖는 NAL 유닛들은 이러한 규격에서 명시된 디코딩 프로세스에 영향을 주지 않을 것이다.
- [0088] 주 - UNSPEC_28..UNSPEC_31의 범위 내의 NAL 유닛 유형들은 애플리케이션에 의해 결정된 바와 같이 사용될 수 있다. 이러한 규격에는 nal_unit_type의 이러한 값들에 대한 어떠한 디코딩 프로세스도 명시되지 않는다. 상이한 애플리케이션들이 상이한 목적들을 위해 이러한 NAL 유닛 유형들을 사용할 수 있었기 때문에, 이러한 nal_unit_type 값들을 갖는 NAL 유닛들을 생성하는 인코더들의 설계에서 그리고 이러한 nal_unit_type 값들을 갖는 NAL 유닛들의 콘텐츠를 해석하는 디코더들의 설계에서 특별한 관심이 작용될 것으로 예상된다. 이러한 규격은 이러한 값들에 대한 어떠한 관리도 정의하지 않는다. 이러한 nal_unit_type 값들은, 사용의 "충돌들"(즉, 동일한 nal_unit_type 값에 대한 NAL 유닛 콘텐츠의 의미의 상이한 정의들)이 중요하지 않거나, 또는 가능하지 않거나, 또는 관리되는 - 예컨대, 제어용 애플리케이션 또는 수송 규격에서, 또는 비트스트림들이 분배되는 환경을 제어함으로써 정의되거나 관리됨 - 상황들에서 사용하는 데에만 적합할 수 있다.
- [0089] 비트스트림의 DU들에서 데이터의 양을 결정하는 것 이외의 목적들을 위해, 디코더들은 nal_unit_type의 예약된 값들을 사용하는 모든 NAL 유닛들의 콘텐츠들을 무시할 것이다(비트스트림으로부터 제거하고 폐기할 것임). 주 - 이러한 요건은 이러한 규격에 대한 호환가능한 확장들의 향후 정의를 허용한다.

[0090] [표 2]

nal_unit_type	nal_unit_type의 명칭	NAL 유닛 및 RBSP 선택스 구조의 콘텐츠	NAL 유닛 유형 클래스
0	TRAIL_NUT	트레일링 픽처 또는 서브픽처의 코딩된 슬라이스* slice_layer_rbsp()	VCL
1	STSA_NUT	STSA 픽처 또는 서브픽처의 코딩된 슬라이스* slice_layer_rbsp()	VCL
2	RADL_NUT	RADL 픽처 또는 서브픽처의 코딩된 슬라이스* slice_layer_rbsp()	VCL
3	RASL_NUT	RASL 픽처 또는 서브픽처의 코딩된 슬라이스* slice_layer_rbsp()	VCL
4..6	RSV_VCL_4.. RSV_VCL_6	예약된 비-IRAP VCL NAL 유닛 유형들	VCL
7	IDR_W_RADL	IDR 픽처 또는 서브픽처의 코딩된 슬라이스* slice_layer_rbsp()	VCL
8	IDR_N_LP		
9	CRA_NUT	CRA 픽처 또는 서브픽처의 코딩된 슬라이스* slice_layer_rbsp()	VCL
10	GDR_NUT	GDR 픽처 또는 서브픽처의 코딩된 슬라이스* slice_layer_rbsp()	VCL
11	RSV_IRAP_11	예약된 IRAP VCL NAL 유닛 유형	VCL
12	OPI_NUT	동작 포인트 정보 operating_point_information_rbsp()	비-VCL
13	DCI_NUT	디코딩 능력 정보 decoding_capability_information_rbsp()	비-VCL
14	VPS_NUT	비디오 파라미터 세트 video_parameter_set_rbsp()	비-VCL
15	SPS_NUT	시퀀스 파라미터 세트 seq_parameter_set_rbsp()	비-VCL
16	PPS_NUT	픽처 파라미터 세트 pic_parameter_set_rbsp()	비-VCL
17	PREFIX_APS_NUT	적응 파라미터 세트	비-VCL
18	SUFFIX_APS_NUT	adaptation_parameter_set_rbsp()	
19	PH_NUT	픽처 헤더 picture_header_rbsp()	비-VCL
20	AUD_NUT	AU 디리미터 access_unit_delimiter_rbsp()	비-VCL
21	EOS_NUT	시퀀스의 종단부 end_of_seq_rbsp()	비-VCL
22	EOB_NUT	비트스트림의 종단부 end_of_bitstream_rbsp()	비-VCL
23	PREFIX_SEI_NUT	보충 향상 정보 sei_rbsp()	비-VCL
24	SUFFIX_SEI_NUT		
25	FD_NUT	필러 데이터 filler_data_rbsp()	비-VCL
26	RSV_NVCL_26	예약된 비-VCL NAL 유닛 유형들	비-VCL
27	RSV_NVCL_27		
28..31	UNSPEC_28.. UNSPEC_31	명시되지 않은 비-VCL NAL 유닛 유형들	비-VCL

*는 pps_mixed_nalu_types_in_pic_flag 가 0 과 동일할 때의 픽처의 속성 및 pps_mixed_nalu_types_in_pic_flag 가 1 과 동일할 때의 서브픽처의 속성을 나타낸다.

[0091]

[0092]

주 - 클린 랜덤 액세스(clean random access, CRA) 픽처는 비트스트림에 존재하는 연관된 RASL 또는 RADL 픽처들을 가질 수 있다.

[0093]

주 - IDR_N_LP와 동일한 nal_unit_type을 갖는 순시 디코딩 리프레시(instantaneous decoding refresh, IDR) 픽처는 비트스트림에 존재하는 연관된 리딩 픽처(leading picture)들을 갖지 않는다. IDR_W_RADL과 동일한 nal_unit_type을 갖는 IDR 픽처는 비트스트림에 존재하는 연관된 RASL 픽처들을 갖지 않지만, 비트스트림에서 연관된 RADL 픽처들을 가질 수 있다.

[0094]

nal_unit_type의 값은 서브픽처의 모든 VCL NAL 유닛들에 대해 동일할 것이다. 서브픽처는 서브픽처의 VCL NAL 유닛들과 동일한 NAL 유닛 유형을 갖는 것으로 지칭된다.

[0095]

임의의 특정 픽처의 VCL NAL 유닛들의 경우, 하기가 적용된다:

[0096]

- pps_mixed_nalu_types_in_pic_flag가 0과 동일한 경우, nal_unit_type의 값은 픽처의 모든 VCL NAL 유닛들에 대해 동일할 것이고, 픽처 또는 PU는 픽처 또는 PU의 코딩된 슬라이스 NAL 유닛들과 동일한 NAL 유닛 유형을 갖는 것으로 지칭된다.

[0097]

- 그렇지 않은 경우(pps_mixed_nalu_types_in_pic_flag가 1과 동일함), 하기의 제약들 모두가 적용된다:

[0098]

- 픽처는 적어도 2개의 버스픽처들을 가질 것이다.

[0099]

- 픽처의 VCL NAL 유닛들은 2개 이상의 상이한 nal_unit_type 값들을 가질 것이다.

[0100]

- GDR_NUT와 동일한 nal_unit_type을 갖는 픽처의 VCL NAL 유닛이 없을 것이다.

- [0101] - 픽처의 VCL NAL 유닛이 IDR_W_RADL, IDR_N_LP, 또는 CRA_NUT와 동일한 nalUnitTypeA과 동일한 nal_unit_type을 가질 때, 픽처의 다른 VCL NAL 유닛들은 모두, nalUnitTypeA 또는 TRAIL_NUT와 동일한 nal_unit_type을 가질 것이다.
- [0102] nal_unit_type의 값은 IRAP 또는 GDR AU에서 모든 픽처들에 대해 동일할 것이다.
- [0103] sps_video_parameter_set_id가 0 초과이고, GeneralLayerIdx[nuh_layer_id]와 동일한 j에 대해, 그리고 j + 1 내지 vps_max_layers_minus1(이를 포함함)의 범위 내의 i의 임의의 값에 대해 vps_max_tid_il_ref_pics_plus1[i][j]가 0과 동일하고, pps_mixed_nalu_types_in_pic_flag가 1과 동일할 때, nal_unit_type의 값은 IDR_W_RADL, IDR_N_LP, 또는 CRA_NUT와 동일하지 않을 것이다.
- [0104] 하기의 제약들이 적용된다는 것이 비트스트림 순응성의 요건이다:
- [0105] - 픽처가 IRAP 픽처의 리딩 픽처일 때, 그것은 RADL 또는 RASL 픽처일 것이다.
- [0106] - 서브픽처가 IRAP 서브픽처의 리딩 서브픽처일 때, 그것은 RADL 또는 RASL 서브픽처일 것이다.
- [0107] - 픽처가 IRAP 픽처의 리딩 픽처가 아닐 때, 그것은 RADL 또는 RASL 픽처가 아닐 것이다.
- [0108] - 서브픽처가 IRAP 서브픽처의 리딩 서브픽처가 아닐 때, 그것은 RADL 또는 RASL 서브픽처가 아닐 것이다.
- [0109] - IDR 픽처와 연관된 비트스트림에 어떠한 RASL 픽처들도 존재하지 않을 것이다.
- [0110] - IDR 서브픽처와 연관된 비트스트림에 어떠한 RASL 서브픽처들도 존재하지 않을 것이다.
- [0111] - IDR_N_LP와 동일한 nal_unit_type을 갖는 IDR 픽처와 연관된 비트스트림에 어떠한 RADL 픽처들도 존재하지 않을 것이다.
- [0112] 주 - 각각의 파라미터 세트가 참조될 때 (이러한 규격에서 명시되지 않은 외부 수단에 의해 또는 비트스트림에서) 그것이 이용가능하다면, IRAP AU 전에 모든 PU들을 폐기함으로써 IRAP AU의 포지션에서 랜덤 액세스를 수행하는 것(및 디코딩 순서에서 IRAP AU 및 모든 후속 AU들 내의 비-RASL 픽처들을 정확하게 디코딩하는 것)이 가능하다.
- [0113] - IDR_N_LP와 동일한 nal_unit_type을 갖는 IDR 서브픽처와 연관된 비트스트림에 어떠한 RADL 서브픽처들도 존재하지 않을 것이다.
- [0114] - 디코딩 순서에서 layerId와 동일한 nuh_layer_id를 갖는 IRAP 픽처에 선행하는, 특정 값 layerId와 동일한 nuh_layer_id를 갖는 임의의 픽처는 출력 순서에서 IRAP 픽처에 선행할 것이고, 출력 순서에서 IRAP 픽처와 연관된 임의의 RADL 픽처에 선행할 것이다.
- [0115] - 디코딩 순서에서 layerId와 동일한 nuh_layer_id 및 subpicIdx와 동일한 서브픽처 인덱스를 갖는 IRAP 서브픽처에 선행하는, 특정 값 layerId와 동일한 nuh_layer_id 및 특정 값 subpicIdx와 동일한 서브픽처 인덱스를 갖는 임의의 서브픽처는 출력 순서에서, IRAP 서브픽처 및 모든 그의 연관된 RADL 서브픽처들에 선행할 것이다.
- [0116] - 디코딩 순서에서 layerId와 동일한 nuh_layer_id를 갖는 복구 포인트 픽처에 선행하는, 특정 값 layerId와 동일한 nuh_layer_id를 갖는 임의의 픽처는 출력 순서에서 복구 포인트 픽처에 선행할 것이다.
- [0117] - 디코딩 순서에서 복구 포인트 픽처에서 layerId와 동일한 nuh_layer_id 및 subpicIdx와 동일한 서브픽처 인덱스를 갖는 서브픽처에 선행하는, 특정 값 layerId와 동일한 nuh_layer_id 및 특정 값 subpicIdx와 동일한 서브픽처 인덱스를 갖는 임의의 서브픽처는 출력 순서에서, 복구 포인트 픽처 내의 해당 서브픽처에 선행할 것이다.
- [0118] - CRA 픽처와 연관된 임의의 RASL 픽처는 출력 순서에서, CRA 픽처와 연관된 임의의 RADL 픽처에 선행할 것이다.
- [0119] - CRA 서브픽처와 연관된 임의의 RASL 서브픽처는 출력 순서에서, CRA 서브픽처와 연관된 임의의 RADL 서브픽처에 선행할 것이다.
- [0120] - CRA 픽처와 연관된, 특정 값 layerId와 동일한 nuh_layer_id를 갖는 임의의 RASL 픽처는 출력 순서에서, 디코딩 순서에서 CRA 픽처에 선행하는 layerId와 동일한 nuh_layer_id를 갖는 임의의 IRAP 또는 GDR 픽처에 후행할 것이다.
- [0121] - CRA 서브픽처와 연관된, 특정 값 layerId와 동일한 nuh_layer_id 및 특정 값 subpicIdx와 동일한 서브픽처

인덱스를 갖는 임의의 RASL 서브픽처는 출력 순서에서, 디코딩 순서에서 CRA 서브픽처에 선행하는 layerId와 동일한 nuh_layer_id 및 subpicIdx와 동일한 서브픽처 인덱스를 갖는 임의의 IRAP 또는 GDR 서브픽처에 후행할 것이다.

[0122] - sps_field_seq_flag가 0과 동일한 경우, 하기가 적용된다: 특정 값 layerId와 동일한 nuh_layer_id를 갖는 현재 픽처가 IRAP 픽처와 연관된 리딩 픽처일 때, 그것은 디코딩 순서에서, 동일한 IRAP 픽처와 연관된 모든 비-리딩 픽처들에 선행할 것이다. 그렇지 않은 경우(sps_field_seq_flag가 1과 동일함), picA 및 picB가 디코딩 순서에서, 각각 IRAP 픽처와 연관된 제1 및 마지막 리딩 픽처들이라고 하면, 디코딩 순서에서 picA에 선행하는, layerId와 동일한 nuh_layer_id를 갖는 최대 하나의 비-리딩 픽처가 있을 것이고, 디코딩 순서에서 picA와 picB 사이에 layerId와 동일한 nuh_layer_id를 갖는 비-리딩 픽처가 없을 것이다.

[0123] - sps_field_seq_flag가 0인 경우, 하기가 적용된다: 특정 값 layerId와 동일한 nuh_layer_id 및 특정 값 subpicIdx와 동일한 서브픽처 인덱스를 갖는 현재 서브픽처가 IRAP 서브픽처와 연관된 리딩 서브픽처일 때, 그것은 디코딩 순서에서, 동일한 IRAP 서브픽처와 연관된 모든 비-리딩 서브픽처들에 선행할 것이다. 그렇지 않은 경우(sps_field_seq_flag가 1과 동일함), subpicA 및 subpicB가 디코딩 순서에서, 각각 IRAP 서브픽처와 연관된 제1 및 마지막 리딩 서브픽처들이라고 하면, 디코딩 순서에서 subpicA에 선행하는, layerId와 동일한 nuh_layer_id 및 subpicIdx와 동일한 서브픽처 인덱스를 갖는 최대 하나의 비-리딩 서브픽처가 있을 것이고, 디코딩 순서에서 picA와 picB 사이에 layerId와 동일한 nuh_layer_id 및 subpicIdx와 동일한 서브픽처 인덱스를 갖는 비-리딩 픽처가 없을 것이다.

[0124] 표 2에 제공된 바와 같이, NAL 유닛은 보충 향상 정보(SEI) 선택스 구조를 포함할 수 있다. 표 3 및 표 4는 JVET-T2001에서 규정된 보충 향상 정보(SEI) 선택스 구조를 예시한다.

[0125] [표 3]

sei_rbsp() {	디스크립터
do	
sei_message()	
while(more_rbsp_data())	
rbasp_trailing_bits()	
}	

[0126]

[0127] [표 4]

sei_message() {	디스크립터
payloadType = 0	
do {	
payload_type_byte	u(8)
payloadType += payload_type_byte	
} while(payload_type_byte == 0xFF)	
payloadSize = 0	
do {	
payload_size_byte	u(8)
payloadSize += payload_size_byte	
} while(payload_size_byte == 0xFF)	
sei_payload(payloadType, payloadSize)	
}	

[0128]

[0129] 표 3 및 표 4와 관련하여, JVET-T2001은 하기의 시맨틱들을 규정한다:

[0130] 각각의 SEI 메시지는 SEI 메시지 페이로드의 유형 payloadType 및 크기 payloadSize를 특정하는 변수들로 이루어진다. SEI 메시지 페이로드들이 특정된다. 도출된 SEI 메시지 페이로드 크기 payloadSize는 바이트 단위로 특정되고, SEI 메시지 페이로드에서 RBSP 바이트들의 수와 동일할 것이다.

[0131] 주 - SEI 메시지를 포함하는 NAL 유닛 바이트 시퀀스는 하나 이상의 에뮬레이션 방지 바이트들(emulation_prevention_three_byte 선택스 요소들로 표현됨)을 포함할 수 있다. SEI 메시지의 페이로드 크기가 RBSP 바이트 단위로 특정되므로, 에뮬레이션 방지 바이트들의 수량은 SEI 페이로드의 크기 payloadSize에 포함되지 않는다.

[0132] **payload_type_byte**는 SEI 메시지의 페이로드 유형의 바이트이다.

[0133] **payload_size_byte**는 SEI 메시지의 페이로드 크기의 바이트이다.

[0134] JVET-T2001은 페이로드 유형들을 정의하고, 문헌["Additional SEI messages for VSEI (Draft 6)", 25th Meeting of ISO/IEC JTC1/SC29/WG11 12 - 21 January 2022, Teleconference, document JVET-Y2006-v1](이는 본 명세서에 참고로 포함되고 JVET-Y2006으로 지칭됨)은 추가적인 페이로드 유형들을 정의한다는 점에 유의해야 한다. 표 5는 대체적으로 sei_payload() 선택스 구조를 예시한다. 즉, 표 5는 sei_payload() 선택스 구조를 예시하지만, 간결성을 위해, 가능한 유형들의 페이로드들 모두가 표 5에 포함되지는 않는다.

[0135] [표 5]

sei_payload(payloadType, payloadSize) {	더스크립터
if(nal_unit_type == PREFIX_SEI_NUT)	
if(payloadType == 0)	
buffering_period(payloadSize)	
...	
else if(payloadType == 204) /* Specified in Rec. ITU-T H.274 ISO/IEC 23002-7 */	
sample_aspect_ratio_info(payloadSize)	
else /* Specified in Rec. ITU-T H.274 ISO/IEC 23002-7 */	
reserved_message(payloadSize)	
else /* nal_unit_type == SUFFIX_SEI_NUT */	
if(payloadType == 3) /* Specified in Rec. ITU-T H.274 ISO/IEC 23002-7 */	
filler_payload(payloadSize)	
...	
else /* Specified in Rec. ITU-T H.274 ISO/IEC 23002-7 */	
reserved_message(payloadSize)	
if(more_data_in_payload()) {	
if(payload_extension_present())	
sei_reserved_payload_extension_data	u(v)
sei_payload_bit_equal_to_one /* equal to 1 */	f(1)
while(!byte_aligned())	
sei_payload_bit_equal_to_zero /* equal to 0 */	f(1)
}	
}	

[0136]

[0137] 표 5와 관련하여, JVET-T2001은 하기의 시맨틱들을 규정한다.

[0138] sei_reserved_payload_extension_data는 이러한 규격의 이러한 버전에 순응하는 비트스트림들에 존재하지 않을 것이다. 그러나, 이러한 규격의 이러한 버전에 순응하는 디코더들은 sei_reserved_payload_extension_data의 존재 및 값을 무시할 것이다. 존재할 때, sei_reserved_payload_extension_data의 길이(비트 단위)는 $8 * \text{payloadSize} - \text{nEarlierBits} - \text{nPayloadZeroBits} - 1$ 과 동일하며, 여기서 nEarlierBits는 sei_reserved_payload_extension_data 선택스 요소에 선행하는 sei_payload() 선택스 구조에서의 비트들의 수이고, nPayloadZeroBits는 sei_payload() 선택스 구조의 종단부에서의 sei_payload_bit_equal_to_zero 선택스 요소들의 수이다.

[0139] SEI 메시지 선택스 구조(예컨대, buffering_period() 선택스 구조)의 파싱 후에 more_data_in_payload()가 TRUE(참)이고, nPayloadZeroBits가 7과 동일하지 않은 경우, PayloadBits는 $8 * \text{payloadSize} - \text{nPayloadZeroBits} - 1$ 과 동일하게 설정되고; 그렇지 않은 경우, PayloadBits는 $8 * \text{payloadSize}$ 와 동일하게 설정된다.

[0140] payload_bit_equal_to_one은 1과 동일할 것이다.

[0141] payload_bit_equal_to_zero은 0과 동일할 것이다.

[0142] 주 - payloadType의 동일한 값을 갖는 SEI 메시지들은, 이들이 프리픽스 SEI NAL 유닛에 포함되는지 아니면 서픽스 SEI NAL 유닛에 포함되는지에 관계없이 개념적으로 동일한 SEI 메시지이다.

[0143] 주 - 이러한 규격 및 VSEI 규격(ITU-T H.274 | ISO/IEC 23002-7)에서 명시된 SEI 메시지들에 대해, payloadType 값들은 AVC(Rec. ITU-T H.264 | ISO/IEC 14496-10) 및 HEVC(Rec. ITU-T H.265 | ISO/IEC 23008-2)에서 명시된 유사한 SEI 메시지들과 정렬된다.

[0144] 각각의 SEI 메시지에 대한 시맨틱들 및 지속성 범주는 각각의 특정 SEI 메시지에 대한 시맨틱 규격에서 명시된다.

[0145] 주 - SEI 메시지들에 대한 지속성 정보가 유익하게 요약되어 있다.

[0146] JVET-T2001은 추가로 하기를 규정한다:

[0147] Rec. ITU-T H.274 | ISO/IEC 23002-7에서 명시된, [표 5]에서 식별된 선택스 구조들을 갖는 SEI 메시지들은 이

러한 규격에 의해 명시된 비트스트림들과 함께 사용될 수 있다.

[0148] 임의의 특정 Rec. ITU-T H.274 | ISO/IEC 23002-7 SEI 메시지가 이러한 규격에 의해 명시된 비트스트림에 포함될 때, SEI 페이로드 선택스는 [표 5]에 특정된 바와 같은 sei_payload() 선택스 구조에 포함될 것이고, [표 5]에 특정된 payloadType 값을 사용할 것이고, 추가적으로, 해당 특정 SEI 메시지에 대해 이러한 부록에서 명시된 임의의 SEI-메시지-특정적 제약들이 적용될 것이다.

[0149] 상기에 명시된 바와 같은 PayloadBits의 값은 Rec. ITU-T H.274 | ISO/IEC 23002-7에서 명시된 SEI 메시지 선택스 구조들의 파서(parser)로 전달된다.

[0150] 전술된 바와 같이, JVET-AB2006은 NN 포스트-필터 보충 향상 정보 메시지들을 규정한다. 특히, JVET-AB2006은 신경 네트워크 포스트-필터 특성 SEI 메시지(payloadType == 210) 및 신경 네트워크 포스트-필터 활성화 SEI 메시지(payloadType == 211)를 규정한다. 표 6은 JVET-AB2006에서 규정된 신경 네트워크 포스트-필터 특성 SEI 메시지의 선택스를 예시한다. 신경 네트워크 포스트-필터 특성 SEI 메시지는 NNPF SEI로 지칭될 수 있다는 점에 유의해야 한다.

[0151] [표 6]

nn_post_filter_characteristics(payloadSize) {	디스크립터
nnpf_id	ue(v)
nnpf_mode_idc	ue(v)
if(nnpf_mode_idc == 1) {	
while(!byte_aligned())	
nnpf_reserved_zero_bit_a	u(1)
nnpf_tag_uri	st(v)
nnpf_uri	st(v)
}	
nnpf_formatting_and_purpose_flag	u(1)
if(nnpf_formatting_and_purpose_flag) {	
nnpf_purpose	ue(v)
/* input and output formatting */	
if(nnpf_purpose == 2 nnpf_purpose == 4)	
nnpf_out_sub_c_flag	u(1)
else if(nnpf_purpose == 3 nnpf_purpose == 4) {	
nnpf_pic_width_in_luma_samples	ue(v)
nnpf_pic_height_in_luma_samples	ue(v)
}	
else if(nnpf_purpose == 5) {	
nnpf_num_input_pics_minus2	ue(v)
for(i = 0; i <= nnpf_num_input_pics_minus2; i++)	
nnpf_interpolated_pics[i]	ue(v)
}	
nnpf_component_last_flag	u(1)
nnpf_inp_format_idc	ue(v)
if(nnpf_inp_format_idc == 1)	
nnpf_inp_tensor_bitdepth_minus8	ue(v)
nnpf_inp_order_idc	ue(v)
nnpf_auxiliary_inp_idc	ue(v)
nnpf_separate_colour_description_present_flag	u(1)
if(nnpf_separate_colour_description_present_flag) {	
nnpf_colour primaries	u(8)
nnpf_transfer_characteristics	u(8)
nnpf_matrix_coeffs	u(8)
}	
nnpf_out_format_idc	ue(v)
if(nnpf_out_format_idc == 1)	
nnpf_out_tensor_bitdepth_minus8	ue(v)
nnpf_out_order_idc	ue(v)
nnpf_constant_patch_size_flag	u(1)
nnpf_patch_width_minus1	ue(v)
nnpf_patch_height_minus1	ue(v)
nnpf_overlap	ue(v)
nnpf_padding_type	ue(v)
if(nnpf_padding_type == 4) {	
nnpf_luma_padding_val	ue(v)
nnpf_cb_padding_val	ue(v)
nnpf_cr_padding_val	ue(v)
}	
nnpf_complexity_info_present_flag	ue(v)
if(nnpf_complexity_info_present_flag) {	
nnpf_parameter_type_idc	u(2)
if(nnpf_parameter_type_idc != 2)	
nnpf_log2_parameter_bit_length_minus3	u(2)
nnpf_num_parameters_idc	u(6)
nnpf_num_kmac_operations_idc	ue(v)
nnpf_total_kilobyte_size	ue(v)
}	
}	
/* ISO/IEC 15938-17 bitstream */	
if(nnpf_mode_idc == 0) {	
while(!byte_aligned())	
nnpf_reserved_zero_bit_b	u(1)
for(i = 0; more_data_in_payload(); i++)	
nnpf_payload_byte[i]	b(8)
}	
}	

[0152]

[0153] 표 6과 관련하여, JVET-AB2006은 하기의 시맨틱들을 규정한다:

[0154] 신경 네트워크 포스트-필터 특성(NNPF) SEI 메시지는 포스트-프로세싱 필터로서 사용될 수 있는 신경 네트워크를 명시한다. 특정 픽처들을 위한 특정된 포스트-프로세싱 필터들의 사용은 신경 네트워크 포스트-필터 활성화

SEI 메시지로 표시된다.

이러한 SEI 메시지의 사용은 하기의 변수들의 정의를 필요로 한다:

- 본 명세서에서 각각 CroppedWidth 및 CroppedHeight로 표기되는, 루마 샘플들의 단위의 크롭(crop)된 디코딩된 출력 픽처 폭 및 높이.

- 존재 시, 포스트-프로세싱 필터에 대한 입력으로 사용되는, 0 내지 $\text{numInputPics} - 1$ (이를 포함함)의 범위 내의 idx를 갖는 크롭된 디코딩된 출력 픽처들의 루마 샘플 어레이 $\text{CroppedYPic}[\text{idx}]$ 및 크로마 샘플 어레이를 $\text{CroppedCbPic}[\text{idx}]$ 및 $\text{CroppedCrPic}[\text{idx}]$.

- 크롭된 디코딩된 출력 픽처들의 루마 샘플 어레이에 대한 비트 깊이 BitDepth_Y .

- 존재하는 경우, 크롭된 디코딩된 출력 픽처들의 크로마 샘플 어레이들에 대한 비트 깊이 BitDepth_C .

- 본 명세서에서 ChromaFormatIdc로 표기되는 크로마 포맷 표시자.

- $\text{nnpfc_auxiliary_inp_idc}$ 가 1과 동일할 때, 0 내지 1(이를 포함함)의 범위 내의 실수일 필터링 강도 제어 값 $\text{StrengthControlVal}$.

변수들 SubWidthC 및 SubHeightC 는 표 7에 의해 특정된 바와 같은 ChromaFormatIdc로부터 도출된다.

[표 7]

$\text{sps_chroma_format_idc}$	크로마 포맷	SubWidthC	SubHeightC
0	단색	1	1
1	4:2:0	2	2
2	4:2:2	2	1
3	4:4:4	1	1

주 - 동일한 픽처에 대해 하나 초과인 NNPF SEI 메시지가 존재할 수 있다. nnpfc_id 의 상이한 값들을 갖는 하나 초과인 NNPF SEI 메시지가 존재하거나 동일한 픽처에 대해 활성화될 때, 그들은 nnpfc_purpose 및 nnpfc_mode_idc 의 동일한 또는 상이한 값들을 가질 수 있다.

nnpfc_id 는 포스트-프로세싱 필터를 식별하는 데 사용될 수 있는 식별 번호를 포함한다. nnpfc_id 의 값은 0 내지 $2^{32} - 2$ (이를 포함함)의 범위 내에 있을 것이다. 256 내지 511(이를 포함함)의 그리고 2^{31} 내지 $2^{32} - 2$ (이를 포함함)의 nnpfc_id 의 값들은 ITU-T | ISO/IEC에 의한 향후 사용을 위해 예약된다. 256 내지 511(이를 포함함)의 범위 내의, 또는 2^{31} 내지 $2^{32} - 2$ (이를 포함함)의 범위 내의 nnpfc_id 를 갖는 NNPF SEI 메시지와 접하는, 본 문헌의 이러한 편집에 순응하는 디코더들은 SEI 메시지를 무시할 것이다.

NNPF SEI 메시지가 디코딩 순서에서, 현재 CLVS 내의 특정 nnpfc_id 값을 갖는 제1 NNPF SEI 메시지일 때, 하기가 적용된다:

- 이러한 SEI 메시지는 기초 포스트-프로세싱 필터를 특정한다.

- 이러한 SEI 메시지는 현재 CLVS의 종료 때까지, 출력 순서에서, 현재 계층의 현재 디코딩된 픽처 및 모든 후속 디코딩된 픽처들에 관련된다.

NNPF SEI 메시지가 현재 CLVS에서, 디코딩 순서에서 이전의 NNPF SEI 메시지의 반복일 때, 후속 시맨틱들은 이러한 SEI 메시지가 현재 CLVS 내의 동일한 콘텐츠를 갖는 유일한 NNPF SEI 메시지였던 것처럼 적용된다.

NNPF SEI 메시지가 디코딩 순서에서, 현재 CLVS 내의 특정 nnpfc_id 값을 갖는 제1 NNPF SEI 메시지가 아닐 때, 하기가 적용된다:

- 이러한 SEI 메시지는 동일한 nnpfc_id 값으로 디코딩 순서에서 선행하는 기초 포스트-프로세싱 필터에 대한 업데이트를 정의한다.

- 이러한 SEI 메시지는 현재 CLVS의 종료 또는 현재 CLVS 내의 출력 순서에서 해당 특정 nnpfc_id 값을 갖는 다음 NNPF SEI 메시지 때까지, 출력 순서에서 현재 계층의 현재 디코딩된 픽처 및 모든 후속 디코딩된 픽처들에 관련된다.

0과 동일한 nnpfc_mode_idc 는, 이러한 SEI 메시지가 기초 포스트-프로세싱 필터를 특정하는 ISO/IEC 15938-17

비트스트림을 포함하거나 또는 동일한 `nnpfc_id` 값을 갖는 기초 포스트-프로세싱 필터에 대한 업데이트임을 나타낸다.

- [0175] NNPF SEI 메시지가 디코딩 순서에서, 현재 CLVS 내의 특정 `nnpfc_id` 값을 갖는 제1 NNPF SEI 메시지일 때, 1과 동일한 `nnpfc_mode_idc`는 `nnpfc_id` 값과 연관된 기초 포스트-프로세싱 필터가 태그 URI `nnpfc_tag_uri`에 의해 식별된 포맷을 갖는 `nnpfc_uri`에 의해 표시된 URI에 의해 식별되는 신경 네트워크임을 특징한다.
- [0176] NNPF SEI 메시지가 디코딩 순서에서, 현재 CLVS 내의 특정 `nnpfc_id` 값을 갖는 제1 NNPF SEI 메시지가 아닐 때, 1과 동일한 `nnpfc_mode_idc`는 동일한 `nnpfc_id` 값을 갖는 기초 포스트-프로세싱 필터에 대한 업데이트가 태그 URI `nnpfc_tag_uri`에 의해 식별된 포맷을 갖는 `nnpfc_uri`에 의해 표시된 URI에 의해 정의됨을 특징한다.
- [0177] `nnpfc_mode_idc`의 값은 본 문헌의 이러한 편집에 순응하는 비트스트림들에서 0 내지 1(이를 포함함)의 범위 내에 있을 것이다. `nnpfc_mode_idc`에 대한 2 내지 255(이를 포함함)의 값들은 ITU-T | ISO/IEC에 의한 향후 사용을 위해 예약되고, 본 문헌의 이러한 편집에 순응하는 비트스트림들에 존재하지 않을 것이다. 본 문헌의 이러한 편집에 순응하는 디코더들은 2 내지 255(이를 포함함)의 범위 내의 `nnpfc_mode_idc`를 갖는 NNPF SEI 메시지들을 무시할 것이다. 255 초과와 `nnpfc_mode_idc`의 값들은 본 문헌의 이러한 편집에 순응하는 비트스트림들에 존재하지 않을 것이고, 향후 사용을 위해 예약되지 않는다.
- [0178] 이러한 SEI 메시지가 디코딩 순서에서, 현재 CLVS 내의 특정 `nnpfc_id` 값을 갖는 제1 NNPF SEI 메시지일 때, 포스트-프로세싱 필터 `PostProcessingFilter()`는 기초 포스트-프로세싱 필터와 동일한 것으로 설정된다.
- [0179] 이러한 SEI 메시지가 디코딩 순서에서, 현재 CLVS 내의 특정 `nnpfc_id` 값을 갖는 제1 NNPF SEI 메시지가 아닐 때, 포스트-프로세싱 필터 `PostProcessingFilter()`는 포스트-프로세싱 필터에 이러한 SEI 메시지에 의해 정의된 업데이트를 적용함으로써 획득된다.
- [0180] 업데이트들은 누적되지 않지만, 오히려 디코딩 순서에서, 현재 CLVS 내의 특정 `nnpfc_id` 값을 갖는 제1 NNPF SEI 메시지에 의해 특정된 포스트-프로세싱 필터인 기초 포스트-프로세싱 필터에 대해 각각의 업데이트가 적용된다.
- [0181] `nnpfc_reserved_zero_bit_a`는 본 문헌의 이러한 편집에 순응하는 비트스트림들에서 0과 동일할 것이다. 디코더들은 `nnpfc_reserved_zero_bit_a`가 0과 동일하지 않은 NNPF SEI 메시지들을 무시할 것이다.
- [0182] `nnpfc_tag_uri`는 IETF RFC 4151에서 명시된 바와 같은 신택스 및 시맨틱들을 갖는 태그 URI를 포함하는데, 이는 `nnpfc_uri`에 의해 특정된 동일한 `nnpfc_id` 값을 갖는 기초 포스트-프로세싱 필터에 대한 업데이트 또는 기초 포스트-프로세싱 필터로서 사용되는 신경 네트워크에 관한 포맷 및 연관된 정보를 식별한다.
- [0183] `nnpfc_tag_uri`는 중앙 등록 기관을 필요로 하지 않고서 `nnpfc_uri`에 의해 특정된 신경 네트워크 데이터의 포맷을 고유하게 식별하는 것을 가능하게 한다.
- [0184] "tag:iso.org,2023:15938-17"과 동일한 `nnpfc_tag_uri`는 `nnpfc_uri`에 의해 식별된 신경 네트워크 데이터가 ISO/IEC 15938-17에 순응함을 나타낸다.
- [0185] `nnpfc_tag_uri`는 IETF 인터넷 표준 66에서 명시된 바와 같은 신택스 및 시맨틱들을 갖는 URI를 포함하는데, 이는 동일한 `nnpfc_id` 값을 갖는 기초 포스트-프로세싱 필터에 대한 업데이트 또는 기초 포스트-프로세싱 필터로서 사용되는 신경 네트워크를 식별한다.
- [0186] 1과 동일한 `nnpfc_formatting_and_purpose_flag`는 필터 목적, 입력 포맷화, 출력 포맷화, 및 복잡도와 관련된 신택스 요소들이 존재함을 특징한다. 0과 동일한 `nnpfc_formatting_and_purpose_flag`는 필터 목적, 입력 포맷화, 출력 포맷화, 및 복잡도와 관련된 신택스 요소들이 존재하지 않음을 특징한다.
- [0187] 이러한 SEI 메시지가 디코딩 순서에서, 현재 CLVS 내의 특정 `nnpfc_id` 값을 갖는 제1 NNPF SEI 메시지일 때, `nnpfc_formatting_and_purpose_flag`는 1과 동일할 것이다. 이러한 SEI 메시지가 디코딩 순서에서, 현재 CLVS 내의 특정 `nnpfc_id` 값을 갖는 제1 NNPF SEI 메시지가 아닐 때, `nnpfc_formatting_and_purpose_flag`는 0과 동일할 것이다.
- [0188] `nnpfc_purpose`는 표 8에 특정된 바와 같은 포스트-프로세싱 필터의 목적을 나타낸다.
- [0189] `nnpfc_purpose`의 값은 본 문헌의 이러한 편집에 순응하는 비트스트림들에서 0 내지 5(이를 포함함)의 범위 내에 있을 것이다. `nnpfc_purpose`에 대한 6 내지 1023(이를 포함함)의 값들은 ITU-T | ISO/IEC에 의한 향후 사용을

위해 예약되고, 본 문헌의 이러한 편집에 순응하는 비트스트림들에 존재하지 않을 것이다. 본 문헌의 이러한 편집에 순응하는 디코더들은 6 내지 1203(이를 포함함)의 범위 내의 `nnpfc_purpose`를 갖는 NNFC SEI 메시지들을 무시할 것이다. 1023 초과 `nnpfc_purpose`의 값들은 본 문헌의 이러한 편집에 순응하는 비트스트림들에 존재하지 않을 것이고, 향후 사용을 위해 예약되지 않는다.

[표 8]

값	해석
0	애플리케이션에 의해 결정된 바와 같이 사용될 수 있음
1	시각적 품질 개선
2	4:2:0 크로마 포맷으로부터 4:2:2 또는 4:4:4 크로마 포맷으로의, 또는 4:2:2 크로마 포맷으로부터 4:4:4 크로마 포맷으로의 크로마 업샘플링
3	크로마 포맷을 변경하지 않고서, 크롭된 디코딩된 출력 픽처의 폭 또는 높이를 증가시킴
4	크롭된 디코딩된 출력 픽처의 폭 또는 높이를 증가시키고 크로마 포맷을 업샘플링함
5	픽처 레이트 업샘플링

주 - `nnpfc_purpose`의 예약된 값이 ITU-T | ISO/IEC에 의해 향후에 사용될 때, 이러한 SEI 메시지의 선택스는 해당 값과 동일한 `nnpfc_purpose`에 의해 컨디셔닝되는 존재를 갖는 선택스 요소들을 사용하여 확장될 수 있다.

SubWidthC가 1과 동일하고 SubHeightC가 1과 동일할 때, `nnpfc_purpose`는 2 또는 4와 동일하지 않을 것이다.

1과 동일한 `nnpfc_out_sub_c_flag`는 outSubWidthC가 1과 동일하고 outSubHeightC가 1과 동일함을 특징한다. 0과 동일한 `nnpfc_out_sub_c_flag`는 outSubWidthC가 2와 동일하고 outSubHeightC가 1과 동일함을 특징한다. `nnpfc_out_sub_c_flag`가 존재하지 않을 때, outSubWidthC는 SubWidthC와 동일한 것으로 추론되고 outSubHeightC는 SubHeightC와 동일한 것으로 추론된다. ChromaFormatIdc가 2와 동일하고 `nnpfc_out_sub_c_flag`가 존재할 때, `nnpfc_out_sub_c_flag`의 값은 1과 동일할 것이다.

`nnpfc_pic_width_in_luma_samples` 및 `nnpfc_pic_height_in_luma_samples`는 `nnpfc_id`에 의해 식별된 포스트-프로세싱 필터를 크롭된 디코딩된 출력 픽처에 적용하는 것으로부터 생성된 픽처의 루마 샘플 어레이의 폭 및 높이를 각각 특징한다. `nnpfc_pic_width_in_luma_samples` 및 `nnpfc_pic_height_in_luma_samples`가 존재하지 않을 때, 이들은 각각 CroppedWidth 및 CroppedHeight와 동일한 것으로 추론된다. `nnpfc_pic_width_in_luma_samples`의 값은 CroppedWidth 내지 $\text{CroppedWidth} * 16 - 1$ (이를 포함함)의 범위 내에 있을 것이다. `nnpfc_pic_height_in_luma_samples`의 값은 CroppedHeight 내지 $\text{CroppedHeight} * 16 - 1$ (이를 포함함)의 범위 내에 있을 것이다.

`nnpfc_num_input_pics_minus2` + 2는 포스트-프로세싱 필터에 대한 입력으로서 사용되는 디코딩된 출력 픽처들의 수를 특징한다.

`nnpfc_interpolated_pics[ i ]`는 포스트-프로세싱 필터에 대한 입력으로서 사용되는 i-번째 픽처와 (i + 1)-번째 픽처 사이에서 포스트-프로세싱 필터에 의해 생성된 보간된 픽처들의 수를 특징한다.

포스트-프로세싱 필터에 대한 입력으로서 사용되는 픽처들의 수를 특징하는 변수 numInputPics 및 포스트-프로세싱 필터로부터 생성된 픽처들의 총 수를 특징하는 변수 numOutputPics가 하기와 같이 도출된다:

```

if( nnpfc_purpose == 5 ) {
    numInputPics = nnpfc_num_input_pics_minus2 + 2
    for( i = 0, numOutputPics = 0; i <= numInputPics - 2; i++ )
        numOutputPics += nnpfc_interpolated_pics[ i ]
} else
    numInputPics = 1

```

1과 동일한 `nnpfc_component_last_flag`는 포스트-프로세싱 필터로의 입력 텐서 inputTensor 및 포스트-프로세싱 필터로부터 생성된 출력 텐서 outputTensor의 마지막 치수가 현재 채널에 사용됨을 나타낸다. 0과 동일한 `nnpfc_component_last_flag`는 포스트-프로세싱 필터로의 입력 텐서 inputTensor 및 포스트-프로세싱 필터로부터 생성된 출력 텐서 outputTensor의 제3 치수가 채널에 현재 사용됨을 나타낸다.

주 - 입력 텐서 및 출력 텐서의 제1 치수는 배치 인덱스에 사용되는데, 이는 일부 신경 네트워크 프레임워크들에서의 관행이다. 이러한 SEI 메시지의 시맨틱들 내의 공식이 0과 동일한 배치 인덱스에 대응하는 배치 크기를 사용하지만, 신경 네트워크 추론에 대한 입력으로서 사용되는 배치 크기를 결정하는 것은 포스트-프로세싱 구현

에 달려 있다.

[0202] 주 - 예를 들어, `nnpfc_inp_order_idc`가 3과 동일하고 `nnpfc_auxiliary_inp_idc`가 1과 동일할 때, 입력 텐서에는 4개의 루마 매트릭스들, 2개의 크로마 매트릭스들, 및 하나의 보조 입력 매트릭스를 포함하는 7개의 채널들이 있다. 이러한 경우, 프로세스 `DeriveInputTensors()`는 입력 텐서의 이러한 7개의 채널들 각각을 하나씩 도출할 것이며, 이러한 채널들 중 특정 채널이 프로세싱될 때, 해당 채널은 프로세스 동안 현재 채널로 지칭된다.

[0203] `nnpfc_inp_format_idc`는 크롭된 디코딩된 출력 픽처의 샘플 값을 포스트-프로세싱 필터에 대한 입력 값으로 변환하는 방법을 나타낸다. `nnpfc_inp_format_idc`가 0과 동일할 때, 포스트-프로세싱 필터에 대한 입력 값들은 실수들이고, 함수들 `InpY()` 및 `InpC()`는 하기와 같이 특정된다:

[0204]
$$\text{InpY}(x) = x \div ((1 \ll \text{BitDepthY}) - 1)$$

[0205]
$$\text{InpC}(x) = x \div ((1 \ll \text{BitDepthC}) - 1)$$

[0206] `nnpfc_inp_format_idc`가 1과 동일할 때, 포스트-프로세싱 필터에 대한 입력 값들은 부호없는 정수들이고, 함수들 `InpY()` 및 `InpC()`는 하기와 같이 특정된다:

```

shiftY = BitDepthY - inpTensorBitDepth
if( inpTensorBitDepth >= BitDepthY )
    InpY( x ) = x << ( inpTensorBitDepth - BitDepthY )
else
    InpY( x ) = Clip3( 0, ( 1 << inpTensorBitDepth ) - 1, ( x + ( 1 << ( shiftY - 1 ) ) ) >> shiftY )

shiftC = BitDepthC - inpTensorBitDepth
if( inpTensorBitDepth >= BitDepthC )
    InpC( x ) = x << ( inpTensorBitDepth - BitDepthC )
else
    InpC( x ) = Clip3( 0, ( 1 << inpTensorBitDepth ) - 1, ( x + ( 1 << ( shiftC - 1 ) ) ) >> shiftC )

```

[0207]

[0208] 변수 `inpTensorBitDepth`는 하기에 특정되는 바와 같은 선택스 요소 `nnpfc_inp_tensor_bitdepth_minus8`로부터 도출된다.

[0209] 1 초과인 `nnpfc_inp_format_idc`의 값들은 ITU-T | ISO/IEC에 의한 향후 규격을 위해 예약되고, 본 문헌의 이러한 편집에 순응하는 비트스트림들에 존재하지 않을 것이다. 본 문헌의 이러한 편집에 순응하는 디코더들은 `nnpfc_inp_order_idc`의 예약된 값들을 포함하는 NNPFC SEI 메시지들을 무시할 것이다.

[0210] `nnpfc_inp_tensor_bitdepth_minus8` + 8은 입력 정수 텐서의 루마 샘플 값들의 비트 깊이를 특정한다. `inpTensorBitDepth`의 값은 하기와 같이 도출된다:

[0211]
$$\text{inpTensorBitDepth} = \text{nnpfc_inp_tensor_bitdepth_minus8} + 8$$

[0212] `nnpfc_inp_tensor_bitdepth_minus8`의 값은 0 내지 24(이를 포함함)의 범위 내에 있을 것이라는 것이 비트스트림 순응성의 요건이다.

[0213] `nnpfc_inp_order_idc`는 크롭된 디코딩된 출력 픽처의 샘플 어레이들을 포스트-프로세싱 필터에 대한 입력 픽처들 중 하나로서 순서화하는 방법을 나타낸다.

[0214] `nnpfc_inp_order_idc`의 값은 본 문헌의 이러한 편집에 순응하는 비트스트림들에서 0 내지 3(이를 포함함)의 범위 내에 있을 것이다. `nnpfc_inp_order_idc`에 대한 4 내지 255(이를 포함함)의 값들은 ITU-T | ISO/IEC에 의한 향후 사용을 위해 예약되고, 본 문헌의 이러한 편집에 순응하는 비트스트림들에 존재하지 않을 것이다. 본 문헌의 이러한 편집에 순응하는 디코더들은 4 내지 255(이를 포함함)의 범위 내의 `nnpfc_inp_order_idc`를 갖는 NNPFC SEI 메시지들을 무시할 것이다. 255 초과인 `nnpfc_inp_order_idc`의 값들은 본 문헌의 이러한 편집에 순응하는 비트스트림들에 존재하지 않을 것이고, 향후 사용을 위해 예약되지 않는다.

[0215] `ChromaFormatIdc`가 1과 동일하지 않을 때, `nnpfc_inp_order_idc`는 3과 동일하지 않을 것이다.

[0216] 표 9는 `nnpfc_inp_order_idc` 값들에 대한 유의한 설명을 포함한다.

[0217] [표 9]

nnpfc_inp_order_idc	설명
0	nnpfc_auxiliary_inp_idc 가 0 과 동일한 경우, 하나의 루마 매트릭스가 각각의 입력 픽처에 대한 입력 텐서에 존재하고, 채널들의 수는 1 이다. 이와 달리, nnpfc_auxiliary_inp_idc 가 1 과 동일할 때, 하나의 루마 매트릭스 및 하나의 보조 입력 매트릭스가 존재하고, 채널들의 수는 2 이다.
1	nnpfc_auxiliary_inp_idc 가 0 과 동일한 경우, 2 개의 크로마 매트릭스들이 입력 텐서에 존재하고, 채널들의 수는 2 이다. 이와 달리, nnpfc_auxiliary_inp_idc 가 1 과 동일할 때, 2 개의 크로마 매트릭스들 및 하나의 보조 입력 매트릭스가 존재하고, 채널들의 수는 3 이다.
2	nnpfc_auxiliary_inp_idc 가 0 과 동일한 경우, 하나의 루마 및 2 개의 크로마 매트릭스들이 입력 텐서에 존재하고, 채널들의 수는 3 이다. 그렇지 않고, nnpfc_auxiliary_inp_idc 가 1 과 동일하지 않은 경우, 하나의 루마 매트릭스, 2 개의 크로마 매트릭스들 및 하나의 보조 입력 매트릭스가 존재하고, 채널들의 수는 4 이다.
3	nnpfc_auxiliary_inp_idc 가 0 과 동일한 경우, 4 개의 루마 매트릭스들 및 2 개의 크로마 매트릭스들이 입력 텐서에 존재하고, 채널들의 수는 6 이다. 그렇지 않고, nnpfc_auxiliary_inp_idc 가 1 과 동일하지 않은 경우, 4 개의 루마 매트릭스들, 2 개의 크로마 매트릭스들 및 하나의 보조 입력 매트릭스가 입력 텐서에 존재하고, 채널들의 수는 7 이다. 루마 채널들은 도 7에 도시된 바와 같이 인터리빙된 방식으로 도출된다. 이러한 nnpfc_inp_order_idc 는 크로마 포맷이 4:2:0 일 때에만 사용될 수 있다.
4..255	예약됨

[0218]

[0219] 패치는 픽처의 성분(예컨대, 루마 또는 크로마 성분)으로부터의 샘플들의 직사각형 어레이이다.

[0220] 0 초과와 nnpfc_auxiliary_inp_idc는 보조 입력 데이터가 신경 네트워크 포스트-필터의 입력 텐서에 존재함을 나타낸다. 0과 동일한 nnpfc_auxiliary_inp_idc는 보조 입력 데이터가 입력 텐서에 존재하지 않음을 나타낸다. 1과 동일한 nnpfc_auxiliary_inp_idc는 보조 입력 데이터가 식 82에 특정된 바와 같이 도출됨을 특정한다.

[0221] nnpfc_auxiliary_inp_idc의 값은 본 문헌의 이러한 편집에 순응하는 비트스트림들에서 0 내지 1(이를 포함함)의 범위 내에 있을 것이다. nnpfc_inp_order_idc에 대한 2 내지 255(이를 포함함)의 값들은 ITU-T | ISO/IEC에 의한 향후 사용을 위해 예약되고, 본 문헌의 이러한 편집에 순응하는 비트스트림들에 존재하지 않을 것이다. 본 문헌의 이러한 편집에 순응하는 디코더들은 2 내지 255(이를 포함함)의 범위 내의 nnpfc_inp_order_idc를 갖는 NNPF SEI 메시지들을 무시할 것이다. 255 초과와 nnpfc_inp_order_idc의 값들은 본 문헌의 이러한 편집에 순응하는 비트스트림들에 존재하지 않을 것이고, 향후 사용을 위해 예약되지 않는다.

[0222] 입력 텐서에 포함된 샘플들의 패치에 대한 상단 좌측 샘플 위치를 특정하는 주어진 수직 샘플 좌표 cTop 및 수평 샘플 좌표 cLeft에 대한 입력 텐서 inputTensor를 도출하기 위한 프로세스 DeriveInputTensors()가 하기와 같이 특정된다:

```
for( i = 0; i < numInputPics; i++) {
    if( nnpfc_inp_order_idc == 0 )
        for( yP = -overlapSize; yP < inpPatchHeight + overlapSize; yP++)
```

[0223]

```

for( xP = -overlapSize; xP < inpPatchWidth + overlapSize; xP++ ) {
    inpVal = InpY( InpSampleVal( cTop + yP, cLeft + xP, CroppedHeight,
        CroppedWidth, CroppedYPic[ i ] ) )
    if( !nnpfc_component_last_flag )
        inputTensor[ 0 ][ i ][ 0 ][ yP + overlapSize ][ xP + overlapSize ] = inpVal
    else
        inputTensor[ 0 ][ i ][ yP + overlapSize ][ xP + overlapSize ][ 0 ] = inpVal
    if( nnpfc_auxiliary_inp_idc == 1 )
        if( !nnpfc_component_last_flag )
            inputTensor[ 0 ][ i ][ 1 ][ yP + overlapSize ][ xP + overlapSize ] = StrengthControlVal
        else
            inputTensor[ 0 ][ i ][ yP + overlapSize ][ xP + overlapSize ][ 1 ] = StrengthControlVal
    }
}
else if( nnpfc_inp_order_idc == 1 )
    for( yP = -overlapSize; yP < inpPatchHeight + overlapSize; yP++ )
        for( xP = -overlapSize; xP < inpPatchWidth + overlapSize; xP++ ) {
            inpCbVal = InpC( InpSampleVal( cTop + yP, cLeft + xP, CroppedHeight / SubHeightC,
                CroppedWidth / SubWidthC, CroppedCbPic[ i ] ) )
            inpCrVal = InpC( InpSampleVal( cTop + yP, cLeft + xP, CroppedHeight / SubHeightC,
                CroppedWidth / SubWidthC, CroppedCrPic[ i ] ) )
            if( !nnpfc_component_last_flag ) {
                inputTensor[ 0 ][ i ][ 0 ][ yP + overlapSize ][ xP + overlapSize ] = inpCbVal
                inputTensor[ 0 ][ i ][ 1 ][ yP + overlapSize ][ xP + overlapSize ] = inpCrVal
            } else {
                inputTensor[ 0 ][ i ][ yP + overlapSize ][ xP + overlapSize ][ 0 ] = inpCbVal
                inputTensor[ 0 ][ i ][ yP + overlapSize ][ xP + overlapSize ][ 1 ] = inpCrVal
            }
        }
    if( nnpfc_auxiliary_inp_idc == 1 )
        if( !nnpfc_component_last_flag )
            inputTensor[ 0 ][ i ][ 2 ][ yP + overlapSize ][ xP + overlapSize ] = StrengthControlVal
        else
            inputTensor[ 0 ][ i ][ yP + overlapSize ][ xP + overlapSize ][ 2 ] = StrengthControlVal
    }
}
else if( nnpfc_inp_order_idc == 2 )
    for( yP = -overlapSize; yP < inpPatchHeight + overlapSize; yP++ )
        for( xP = -overlapSize; xP < inpPatchWidth + overlapSize; xP++ ) {
            yY = cTop + yP
            xY = cLeft + xP
            yC = yY / SubHeightC
            xC = xY / SubWidthC
            inpYVal = InpY( InpSampleVal( yY, xY, CroppedHeight,

```

[0224]

```

        CroppedWidth, CroppedYPic[ i ] ) )
inpCbVal = InpC( InpSampleVal( yC, xC, CroppedHeight / SubHeightC,
        CroppedWidth / SubWidthC, CroppedCbPic[ i ] ) )
inpCrVal = InpC( InpSampleVal( yC, xC, CroppedHeight / SubHeightC,
        CroppedWidth / SubWidthC, CroppedCrPic[ i ] ) )
if( !nnpfc_component_last_flag ) {
    inputTensor[ 0 ][ i ][ 0 ][ yP + overlapSize ][ xP + overlapSize ] = inpYVal
    inputTensor[ 0 ][ i ][ 1 ][ yP + overlapSize ][ xP + overlapSize ] = inpCbVal
    inputTensor[ 0 ][ i ][ 2 ][ yP + overlapSize ][ xP + overlapSize ] = inpCrVal
} else {
    inputTensor[ 0 ][ i ][ yP + overlapSize ][ xP + overlapSize ][ 0 ] = inpYVal
    inputTensor[ 0 ][ i ][ yP + overlapSize ][ xP + overlapSize ][ 1 ] = inpCbVal
    inputTensor[ 0 ][ i ][ yP + overlapSize ][ xP + overlapSize ][ 2 ] = inpCrVal
}
if( nnpfc_auxiliary_inp_idc == 1 )
    if( !nnpfc_component_last_flag )
        inputTensor[ 0 ][ i ][ 3 ][ yP + overlapSize ][ xP + overlapSize ] = StrengthControlVal
    else
        inputTensor[ 0 ][ i ][ yP + overlapSize ][ xP + overlapSize ][ 3 ] = StrengthControlVal
}
else if( nnpfc_inp_order_idc == 3 )
for( yP = -overlapSize; yP < inpPatchHeight + overlapSize; yP++)
for( xP = -overlapSize; xP < inpPatchWidth + overlapSize; xP++) {
    yTL = cTop + yP * 2
    xTL = cLeft + xP * 2
    yBR = yTL + 1
    xBR = xTL + 1
    yC = cTop / 2 + yP
    xC = cLeft / 2 + xP
    inpTLVal = InpY( InpSampleVal( yTL, xTL, CroppedHeight,
        CroppedWidth, CroppedYPic[ i ] ) )
    inpTRVal = InpY( InpSampleVal( yTL, xBR, CroppedHeight,
        CroppedWidth, CroppedYPic[ i ] ) )
    inpBLVal = InpY( InpSampleVal( yBR, xTL, CroppedHeight,
        CroppedWidth, CroppedYPic[ i ] ) )
    inpBRVal = InpY( InpSampleVal( yBR, xBR, CroppedHeight,
        CroppedWidth, CroppedYPic[ i ] ) )
    inpCbVal = InpC( InpSampleVal( yC, xC, CroppedHeight / 2,
        CroppedWidth / 2, CroppedCbPic[ i ] ) )
    inpCrVal = InpC( InpSampleVal( yC, xC, CroppedHeight / 2,
        CroppedWidth / 2, CroppedCrPic[ i ] ) )

```

[0225]

```

    if( !nnpfc_component_last_flag ) {
        inputTensor[ 0 ][ i ][ 0 ][ yP + overlapSize ][ xP + overlapSize ] = inpTLVal
        inputTensor[ 0 ][ i ][ 1 ][ yP + overlapSize ][ xP + overlapSize ] = inpTRVal
        inputTensor[ 0 ][ i ][ 2 ][ yP + overlapSize ][ xP + overlapSize ] = inpBLVal
        inputTensor[ 0 ][ i ][ 3 ][ yP + overlapSize ][ xP + overlapSize ] = inpBRVal
        inputTensor[ 0 ][ i ][ 4 ][ yP + overlapSize ][ xP + overlapSize ] = inpCbVal
        inputTensor[ 0 ][ i ][ 5 ][ yP + overlapSize ][ xP + overlapSize ] = inpCrVal
    } else {
        inputTensor[ 0 ][ i ][ yP + overlapSize ][ xP + overlapSize ][ 0 ] = inpTLVal
        inputTensor[ 0 ][ i ][ yP + overlapSize ][ xP + overlapSize ][ 1 ] = inpTRVal
        inputTensor[ 0 ][ i ][ yP + overlapSize ][ xP + overlapSize ][ 2 ] = inpBLVal
        inputTensor[ 0 ][ i ][ yP + overlapSize ][ xP + overlapSize ][ 3 ] = inpBRVal
        inputTensor[ 0 ][ i ][ yP + overlapSize ][ xP + overlapSize ][ 4 ] = inpCbVal
        inputTensor[ 0 ][ i ][ yP + overlapSize ][ xP + overlapSize ][ 5 ] = inpCrVal
    }
    if( nnpfc_auxiliary_inp_idc == 1 )
        if( !nnpfc_component_last_flag )
            inputTensor[ 0 ][ i ][ 6 ][ yP + overlapSize ][ xP + overlapSize ] = StrengthControlVal
        else
            inputTensor[ 0 ][ i ][ yP + overlapSize ][ xP + overlapSize ][ 6 ] = StrengthControlVal
    }
}

```

[0226]

[0227]

1과 동일한 **nnpfc_separate_colour_description_present_flag**는 포스트-프로세싱 필터로부터 생성된 픽처에 대한 원색들, 전달 특성들, 및 매트릭스 계수들의 개별 조합이 SEI 메시지 신택스 구조에서 특정됨을 나타낸다. 0과 동일한 **nnpfc_separate_colour_description_present_flag**는 포스트-프로세싱 필터로부터 생성된 픽처에 대한 원색들, 전달 특성들, 및 매트릭스 계수들의 조합이 CLVS에 대한 VUI 파라미터들에 나타내진 바와 동일함을 나타낸다.

[0228]

nnpfc_colour_primaries는 **vui_colour_primaries** 신택스 요소에 대해 특정된 바와 동일한 시맨틱들을 갖는데, 이들은 하기와 같다: **vui_colour_primaries**는 소스 원색들의 색도 좌표들을 나타낸다. 그의 시맨틱들은 Rec. ITU-T H.273 | ISO/IEC 23091-2에서 ColourPrimaries 파라미터에 대해 명시된 바와 같다. **vui_colour_primaries** 신택스 요소가 존재하지 않을 때, **vui_colour_primaries**의 값은 2와 동일한 것으로 추론된다(색도는 알려져 있지 않거나 명시되어 있지 않거나, 또는 이러한 규격에서 명시되지 않은 다른 수단에 의해 결정됨). Rec. ITU-T H.273 | ISO/IEC 23091-2에서 향후 사용을 위해 예약된 것으로서 식별되는 **vui_colour_primaries**의 값들은 이러한 규격의 이러한 버전에 순응하는 비트스트림들에 존재하지 않을 것이다. 디코더들은 **vui_colour_primaries**의 예약된 값들을 값 2와 동등한 것으로서 해석할 것이다.

[0229]

하기를 제외한다:

[0230]

- **nnpfc_colour_primaries**는 CLVS에 사용되는 원색들보다는, SEI 메시지에서 특정된 신경 네트워크 포스트-필터를 적용하는 것으로부터 생성된 픽처의 원색들을 특정한다.

[0231]

- **nnpfc_colour_primaries**가 NNPFC SEI 메시지에 존재하지 않을 때, **nnpfc_colour_primaries**의 값은 **vui_colour_primaries**와 동일한 것으로 추론된다.

[0232]

nnpfc_transfer_characteristics는 **vui_transfer_characteristics** 신택스 요소에 대해서와 동일한 시맨틱들을 갖는데, 이들은 하기와 같다: **vui_transfer_characteristics**는 색상 표현의 전달 특성 함수를 나타낸다. 그의 시맨틱들은 Rec. ITU-T H.273 | ISO/IEC 23091-2에서 TransferCharacteristics 파라미터에 대해 명시된 바와 같다. **vui_transfer_characteristics** 신택스 요소가 존재하지 않을 때, **vui_transfer_characteristics**의 값은 2와 동일한 것으로 추론된다(전달 특성들은 알려져 있지 않거나 명시되어 있지 않거나 이러한 규격에서 명시되지 않은 다른 수단들에 의해 결정됨). Rec. ITU-T H.273 | ISO/IEC 23091-2에서 향후 사용을 위해 예약된 것으로서 식별되는 **vui_transfer_characteristics**의 값들은 이러한 규격의 이러한 버전에 순응하는 비트스트림들에 존재하지 않을 것이다. 디코더들은 **vui_transfer_characteristics**의 예약된 값들을 값 2와 동등한 것으로서 해석할 것이다.

[0233]

하기를 제외한다:

[0234]

- **nnpfc_transfer_characteristics**는 CLVS에 사용되는 전달 특성들보다는, SEI 메시지에서 특정된 신경 네트워크

크 포스트-필터를 적용하는 것으로부터 생성된 픽처의 전달 특성들을 특정한다.

[0235] - `nnpfc_transfer_characteristics`가 NNPFC SEI 메시지에 존재하지 않을 때, `nnpfc_transfer_characteristics`의 값은 `vui_transfer_characteristics`와 동일한 것으로 추론된다.

[0236] `nnpfc_matrix_coeffs`는 `vui_matrix_coeffs` 선택 요소에 대해서와 동일한 시맨틱들을 갖는데, 이들은 하기와 같다: `vui_matrix_coeffs`는 녹색, 청색, 및 적색, 또는 Y, Z, 및 X 원색들로부터 루마 및 크로마 신호들을 도출하는 데 사용되는 수학적식들을 설명한다. 그의 시맨틱들은 Rec. ITU-T H.273 | ISO/IEC 23091-2에서 `MatrixCoefficients`에 대해 명시된 바와 같다.

[0237] 하기를 제외한다:

[0238] - `nnpfc_matrix_coeffs`는 CLVS에 사용되는 매트릭스 계수들보다는, SEI 메시지에서 특정된 신경 네트워크 포스트-필터를 적용하는 것으로부터 생성된 픽처의 매트릭스 계수들을 특정한다.

[0239] - `nnpfc_matrix_coeffs`가 NNPFC SEI 메시지에 존재하지 않을 때, `nnpfc_matrix_coeffs`의 값은 `vui_matrix_coeffs`와 동일한 것으로 추론된다.

[0240] - `nnpfc_matrix_coeffs`에 대해 허용된 값들은 VUI 파라미터들의 시맨틱들에 대한 `ChromaFormatIdc`의 값에 의해 나타내지는 디코딩된 비디오 픽처들의 크로마 포맷에 의해 제한되지 않는다.

[0241] - `nnpfc_matrix_coeffs`가 0과 동일할 때, `nnpfc_out_order_idc`는 1 또는 3과 동일하지 않을 것이다.

[0242] 0과 동일한 `nnpfc_out_format_idc`는, 포스트-프로세싱 필터에 의해 출력된 샘플 값들이 실수인 것을 나타내며, 여기서 0 내지 1(이를 포함함)의 값 범위는 후속 포스트-프로세싱 또는 디스플레이를 위한 임의의 원하는 비트 깊이를 `bitDepth`를 위해, 0 내지 $(1 \ll \text{bitDepth}) - 1$ (이를 포함함)의 부호없는 정수 값 범위에 선형으로 맵핑된다.

[0243] 1과 동일한 `nnpfc_out_format_flag`는 포스트-프로세싱 필터에 의해 출력된 샘플 값들이 0 내지 $(1 \ll (\text{nnpfc_out_tensor_bitdepth_minus8} + 8)) - 1$ (이를 포함함)의 범위 내의 부호없는 정수들을 나타낸다.

[0244] 1 초과인 `nnpfc_out_format_idc`의 값들은 ITU-T | ISO/IEC에 의한 향후 규격을 위해 예약되고, 본 문헌의 이러한 편집에 순응하는 비트스트림들에 존재하지 않을 것이다. 본 문헌의 이러한 편집에 순응하는 디코더들은 `nnpfc_out_format_idc`의 예약된 값들을 포함하는 NNPFC SEI 메시지들을 무시할 것이다.

[0245] `nnpfc_out_tensor_bitdepth_minus8 + 8`은 출력 정수 텐서에서의 샘플 값들의 비트 깊이를 특정한다. `nnpfc_out_tensor_bitdepth_minus8`의 값은 0 내지 24(이를 포함함)의 범위 내에 있을 것이다.

[0246] `nnpfc_out_order_idc`는 포스트-프로세싱 필터로부터 생성된 샘플들의 출력 순서를 나타낸다.

[0247] `nnpfc_out_order_idc`의 값은 본 문헌의 이러한 편집에 순응하는 비트스트림들에서 0 내지 3(이를 포함함)의 범위 내에 있을 것이다. `nnpfc_out_order_idc`에 대한 4 내지 255(이를 포함함)의 값들은 ITU-T | ISO/IEC에 의한 향후 사용을 위해 예약되고, 본 문헌의 이러한 편집에 순응하는 비트스트림들에 존재하지 않을 것이다. 본 문헌의 이러한 편집에 순응하는 디코더들은 4 내지 255(이를 포함함)의 범위 내의 `nnpfc_out_order_idc`를 갖는 NNPFC SEI 메시지들을 무시할 것이다. 255 초과인 `nnpfc_out_order_idc`의 값들은 본 문헌의 이러한 편집에 순응하는 비트스트림들에 존재하지 않을 것이고, 향후 사용을 위해 예약되지 않는다.

[0248] `nnpfc_purpose`가 2 또는 4와 동일할 때, `nnpfc_out_order_idc`는 3과 동일하지 않을 것이다.

[0249] 표 10은 `nnpfc_out_order_idc` 값들에 대한 유익한 설명을 포함한다.

[0250] [표 10]

<code>nnpfc_out_order_idc</code>	설명
0	루마 매트릭스만이 출력 텐서에 존재하고, 이에 따라, 채널들의 수는 1이다.
1	크로마 매트릭스들만이 출력 텐서에 존재하고, 이에 따라, 채널들의 수는 2이다.
2	루마 및 크로마 매트릭스들이 출력 텐서에 존재하고, 이에 따라, 채널들의 수는 3이다.
3	4개의 루마 매트릭스들 및 2개의 크로마 매트릭스들이 출력 텐서에 존재하고, 이에 따라, 채널들의 수는 6이다. 이러한 <code>nnpfc_out_order_idc</code> 는 크로마 포맷이 4:2:0 일 때에만 사용될 수 있다.
4..255	예약됨

[0251]

[0252] 입력 텐서에 포함된 샘플들의 패치에 대한 상단 좌측 샘플 위치를 특정하는 주어진 수직 샘플 좌표 `cTop` 및 수

평 샘플 좌표 cLeft에 대한 출력 텐서 outputTensor로부터 필터링된 출력 샘플 어레이들 FilteredYPic, FilteredCbPic, 및 FilteredCrPic 내의 샘플 값들을 도출하기 위한 프로세스 StoreOutputTensors()가 하기와 같이 특징된다:

```

for( i = 0; i < numInputPics; i++ ) {
    if( nnpfc_out_order_idc == 0 )
        for( yP = 0; yP < outPatchHeight; yP++ )
            for( xP = 0; xP < outPatchWidth; xP++ ) {
                yY = cTop * outPatchHeight / inpPatchHeight + yP
                xY = cLeft * outPatchWidth / inpPatchWidth + xP
                if ( yY < nnpfc_pic_height_in_luma_samples && xY < nnpfc_pic_width_in_luma_samples )
                    if( !nnpfc_component_last_flag )
                        FilteredYPic[ i ][ xY ][ yY ] = outputTensor[ 0 ][ i ][ 0 ][ yP ][ xP ]
                    else
                        FilteredYPic[ i ][ xY ][ yY ] = outputTensor[ 0 ][ i ][ yP ][ xP ][ 0 ]
            }
    else if( nnpfc_out_order_idc == 1 )
        for( yP = 0; yP < outPatchCHeight; yP++ )
            for( xP = 0; xP < outPatchCWidth; xP++ ) {

                xSrc = cLeft * horCScaling + xP
                ySrc = cTop * verCScaling + yP
                if ( ySrc < nnpfc_pic_height_in_luma_samples / outSubHeightC &&
                    xSrc < nnpfc_pic_width_in_luma_samples / outSubWidthC )
                    if( !nnpfc_component_last_flag ) {
                        FilteredCbPic[ i ][ xSrc ][ ySrc ] = outputTensor[ 0 ][ i ][ 0 ][ yP ][ xP ]
                        FilteredCrPic[ i ][ xSrc ][ ySrc ] = outputTensor[ 0 ][ i ][ 1 ][ yP ][ xP ]
                    } else {
                        FilteredCbPic[ i ][ xSrc ][ ySrc ] = outputTensor[ 0 ][ i ][ yP ][ xP ][ 0 ]
                        FilteredCrPic[ i ][ xSrc ][ ySrc ] = outputTensor[ 0 ][ i ][ yP ][ xP ][ 1 ]
                    }
            }
    else if( nnpfc_out_order_idc == 2 )
        for( yP = 0; yP < outPatchHeight; yP++ )
            for( xP = 0; xP < outPatchWidth; xP++ ) {
                yY = cTop * outPatchHeight / inpPatchHeight + yP
                xY = cLeft * outPatchWidth / inpPatchWidth + xP
                yC = yY / outSubHeightC
                xC = xY / outSubWidthC
                yPc = ( yP / outSubHeightC ) * outSubHeightC
                xPc = ( xP / outSubWidthC ) * outSubWidthC
                if ( yY < nnpfc_pic_height_in_luma_samples && xY < nnpfc_pic_width_in_luma_samples )
                    if( !nnpfc_component_last_flag ) {
                        FilteredYPic[ i ][ xY ][ yY ] = outputTensor[ 0 ][ i ][ 0 ][ yP ][ xP ]
                        FilteredCbPic[ i ][ xC ][ yC ] = outputTensor[ 0 ][ i ][ 1 ][ yPc ][ xPc ]
                        FilteredCrPic[ i ][ xC ][ yC ] = outputTensor[ 0 ][ i ][ 2 ][ yPc ][ xPc ]
                    } else {
                        FilteredYPic[ i ][ xY ][ yY ] = outputTensor[ 0 ][ i ][ yP ][ xP ][ 0 ]
                        FilteredCbPic[ i ][ xC ][ yC ] = outputTensor[ 0 ][ i ][ yPc ][ xPc ][ 1 ]
                        FilteredCrPic[ i ][ xC ][ yC ] = outputTensor[ 0 ][ i ][ yPc ][ xPc ][ 2 ]
                    }
            }
    else if( nnpfc_out_order_idc == 3 )
        for( yP = 0; yP < outPatchHeight; yP++ )
            for( xP = 0; xP < outPatchWidth; xP++ ) {
                ySrc = cTop / 2 * outPatchHeight / inpPatchHeight + yP
                xSrc = cLeft / 2 * outPatchWidth / inpPatchWidth + xP
                if ( ySrc < nnpfc_pic_height_in_luma_samples / 2 &&
                    xSrc < nnpfc_pic_width_in_luma_samples / 2 )
                    if( !nnpfc_component_last_flag ) {
                        FilteredYPic[ i ][ xSrc * 2 ][ ySrc * 2 ] = outputTensor[ 0 ][ i ][ 0 ][ yP ][ xP ]

```

```

        FilteredYPic[ i ][ xSrc * 2 + 1 ][ ySrc * 2 ] = outputTensor[ 0 ][ i ][ 1 ][ yP ][ xP ]
        FilteredYPic[ i ][ xSrc * 2 ][ ySrc * 2 + 1 ] = outputTensor[ 0 ][ i ][ 2 ][ yP ][ xP ]
        FilteredYPic[ i ][ xSrc * 2 + 1 ][ ySrc * 2 + 1 ] = outputTensor[ 0 ][ i ][ 3 ][ yP ][ xP ]
        FilteredCbPic[ i ][ xSrc ][ ySrc ] = outputTensor[ 0 ][ i ][ 4 ][ yP ][ xP ]
        FilteredCrPic[ i ][ xSrc ][ ySrc ] = outputTensor[ 0 ][ i ][ 5 ][ yP ][ xP ]
    } else {
        FilteredYPic[ i ][ xSrc * 2 ][ ySrc * 2 ] = outputTensor[ 0 ][ i ][ yP ][ xP ][ 0 ]
        FilteredYPic[ i ][ xSrc * 2 + 1 ][ ySrc * 2 ] = outputTensor[ 0 ][ i ][ yP ][ xP ][ 1 ]
        FilteredYPic[ i ][ xSrc * 2 ][ ySrc * 2 + 1 ] = outputTensor[ 0 ][ i ][ yP ][ xP ][ 2 ]
        FilteredYPic[ i ][ xSrc * 2 + 1 ][ ySrc * 2 + 1 ] = outputTensor[ 0 ][ i ][ yP ][ xP ][ 3 ]
        FilteredCbPic[ i ][ xSrc ][ ySrc ] = outputTensor[ 0 ][ i ][ yP ][ xP ][ 4 ]
        FilteredCrPic[ i ][ xSrc ][ ySrc ] = outputTensor[ 0 ][ i ][ yP ][ xP ][ 5 ]
    }
}
}
}

```

[0255]

[0256]

1과 동일한 `nnpfc_constant_patch_size_flag`는 포스트-프로세싱 필터가 입력으로서 `nnpfc_patch_width_minus1` 및 `nnpfc_patch_height_minus1`에 의해 나타내지는, 정확히 패치 크기를 수용함을 나타낸다. 0과 동일한 `nnpfc_constant_patch_size_flag`는 포스트-프로세싱 필터가 입력으로서 `nnpfc_patch_width_minus1` 및 `nnpfc_patch_height_minus1`에 의해 나타내지는 패치 크기의 양의 정수 배수인 임의의 패치 크기를 수용함을 나타낸다.

[0257]

`nnpfc_patch_width_minus1 + 1`은 `nnpfc_constant_patch_size_flag`가 1과 동일할 때, 포스트-프로세싱 필터에 대한 입력을 위해 요구되는 패치 크기의 수평 샘플 카운트들을 나타낸다. `nnpfc_patch_width_minus1`의 값은 0 내지 $\text{Min}(32766, \text{CroppedWidth} - 1)$ (이를 포함함)의 범위 내에 있을 것이다.

[0258]

`nnpfc_patch_height_minus1 + 1`은 `nnpfc_constant_patch_size_flag`가 1과 동일할 때, 포스트-프로세싱 필터에 대한 입력을 위해 요구되는 패치 크기의 수직 샘플 카운트들을 나타낸다. `nnpfc_patch_height_minus1`의 값은 0 내지 $\text{Min}(32766, \text{CroppedHeight} - 1)$ (이를 포함함)의 범위 내에 있을 것이다.

[0259]

변수들 `inpPatchWidth` 및 `inpPatchHeight`가 각각 패치 크기 폭 및 패치 크기 높이라고 하자.

[0260]

`nnpfc_constant_patch_size_flag`가 0과 동일한 경우, 하기가 적용된다.

[0261]

- `inpPatchWidth` 및 `inpPatchHeight`의 값들은 본 문헌에서 특정되지 않은 외부 수단에 의해 제공되거나, 또는 포스트-프로세서 자체에 의해 설정된다.

[0262]

- `inpPatchWidth`의 값은 `nnpfc_patch_width_minus1 + 1`의 양의 정수 배수일 것이고, `CroppedWidth` 이하일 것이다. `inpPatchHeight`의 값은 `nnpfc_patch_height_minus1 + 1`의 양의 정수 배수일 것이고, `CroppedHeight` 이하일 것이다.

[0263]

이와 달리(`nnpfc_constant_patch_size_flag`가 1과 동일한 경우), `inpPatchWidth`의 값은 `nnpfc_patch_width_minus1 + 1`과 동일하게 설정되고, `inpPatchHeight`의 값은 `nnpfc_patch_height_minus1 + 1`과 동일하게 설정된다.

[0264]

`nnpfc_overlap`은 포스트-프로세싱 필터의 인접 입력 텐서들의 중첩되는 수평 및 수직 샘플 카운트들을 나타낸다. `nnpfc_overlap`의 값은 0 내지 16383(이를 포함함)의 범위 내에 있을 것이다.

[0265]

변수들 `outPatchWidth`, `outPatchHeight`, `horCScaling`, `verCScaling`, `outPatchCWidth`, `outPatchCHeight`, 및 `overlapSize`는 하기와 같이 도출된다:

[0266]

$$\text{outPatchWidth} = (\text{nnpfc_pic_width_in_luma_samples} * \text{inpPatchWidth}) / \text{CroppedWidth}$$

[0267]

$$\text{outPatchHeight} = (\text{nnpfc_pic_height_in_luma_samples} * \text{inpPatchHeight}) / \text{CroppedHeight}$$

[0268]

$$\text{horCScaling} = \text{SubWidthC} / \text{outSubWidthC}$$

[0269]

$$\text{verCScaling} = \text{SubHeightC} / \text{outSubHeightC}$$

[0270]

$$\text{outPatchCWidth} = \text{outPatchWidth} * \text{horCScaling}$$

[0271]

$$\text{outPatchCHeight} = \text{outPatchHeight} * \text{verCScaling}$$

[0272] overlapSize = nnpfc_overlap

[0273] outPatchWidth * CroppedWidth가 nnpfc_pic_width_in_luma_samples * inpPatchWidth와 동일할 것이고, outPatchHeight * CroppedHeight가 nnpfc_pic_height_in_luma_samples * inpPatchHeight와 동일할 것이라는 것이 비트스트림 순응성의 요건이다.

[0274] **nnpfc_padding_type**은 표 11에서 설명된 바와 같이 크롭된 디코딩된 출력 픽처의 경계들 외부의 샘플 위치들을 참조할 때 패딩(padding)하는 프로세스를 나타낸다. nnpfc_padding_type의 값은 0 내지 15(이를 포함함)의 범위 내에 있을 것이다.

[0275] [표 11]

nnpfc_padding_type	설명
0	제로 패딩
1	복제 패딩
2	반사 패딩
3	래어라운드 패딩
4	고정 패딩
5..15	예약됨

[0276]

[0277] **nnpfc_luma_padding_val**은 nnpfc_padding_type이 4와 동일할 때 패딩에 사용될 루마 값을 나타낸다.

[0278] **nnpfc_cb_padding_val**은 nnpfc_padding_type이 4와 동일할 때 패딩에 사용될 Cb 값을 나타낸다.

[0279] **nnpfc_cr_padding_val**은 nnpfc_padding_type이 4와 동일할 때 패딩에 사용될 Cr 값을 나타낸다.

[0280] 입력들이 수직 샘플 위치 y, 수평 샘플 위치 x, 픽처 높이 picHeight, 픽처 폭 picWidth, 및 샘플 어레이 croppedPic인 함수 InpSampleVal(y, x, picHeight, picWidth, croppedPic)은 하기와 같이 도출되는 sampleVal의 값을 반환한다:

[0281] 주 - 함수 InpSampleVal()에 대한 입력들에 대해, 수직 위치는 일부 추론 엔진들의 입력 텐서 협약들과의 호환성을 위해 수평 위치 앞에 열거된다.

```

if( nnpfc_padding_type == 0 )
    if( y < 0 || x < 0 || y >= picHeight || x >= picWidth )

        sampleVal = 0
    else
        sampleVal = croppedPic[ x ][ y ]
else if( nnpfc_padding_type == 1 )
    sampleVal = croppedPic[ Clip3( 0, picWidth - 1, x ) ][ Clip3( 0, picHeight - 1, y ) ]
else if( nnpfc_padding_type == 2 )
    sampleVal = croppedPic[ Reflect( picWidth - 1, x ) ][ Reflect( picHeight - 1, y ) ]
else if( nnpfc_padding_type == 3 )
    if( y >= 0 && y < picHeight )
        sampleVal = croppedPic[ Wrap( picWidth - 1, x ) ][ y ]
else if( nnpfc_padding_type == 4 )
    if( y < 0 || x < 0 || y >= picHeight || x >= picWidth )
        sampleVal[ 0 ] = nnpfc_luma_padding_val
        sampleVal[ 1 ] = nnpfc_cb_padding_val
        sampleVal[ 2 ] = nnpfc_cr_padding_val
    else
        sampleVal = croppedPic[ x ][ y ]

```

[0282]

[0283] 하기의 예시적인 프로세스는 nnpfc_out_order_idc에 의해 나타내진 바와 같이, Y, Cb, 및 Cr 샘플 어레이들 FilterdYPic, FilterdCbPic, 및 FilterdCrPic를 각각 포함하는 필터링된 픽처를 생성하기 위해 포스트-프로세싱 필터 PostProcessingFilter()를 사용하여, 크롭된 디코딩된 출력 픽처를 패치별로 필터링하는 데 사용될

수 있다.

```

if( nnpfc_inp_order_idc == 0 )
    for( cTop = 0; cTop < CroppedHeight; cTop += inpPatchHeight )
        for( cLeft = 0; cLeft < CroppedWidth; cLeft += inpPatchWidth ) {
            DeriveInputTensors( )
            outputTensor = PostProcessingFilter( inputTensor )
            StoreOutputTensors( )
        }
else if( nnpfc_inp_order_idc == 1 )
    for( cTop = 0; cTop < CroppedHeight / SubHeightC; cTop += inpPatchHeight )
        for( cLeft = 0; cLeft < CroppedWidth / SubWidthC; cLeft += inpPatchWidth ) {
            DeriveInputTensors( )
            outputTensor = PostProcessingFilter( inputTensor )
            StoreOutputTensors( )
        }
else if( nnpfc_inp_order_idc == 2 )
    for( cTop = 0; cTop < CroppedHeight; cTop += inpPatchHeight )
        for( cLeft = 0; cLeft < CroppedWidth; cLeft += inpPatchWidth ) {
            DeriveInputTensors( )
            outputTensor = PostProcessingFilter( inputTensor )
            StoreOutputTensors( )
        }
else if( nnpfc_inp_order_idc == 3 )
    for( cTop = 0; cTop < CroppedHeight; cTop += inpPatchHeight * 2 )
        for( cLeft = 0; cLeft < CroppedWidth; cLeft += inpPatchWidth * 2 ) {
            DeriveInputTensors( )
            outputTensor = PostProcessingFilter( inputTensor )
            StoreOutputTensors( )
        }
}

```

[0284]

[0285]

1과 동일한 **nnpfc_complexity_info_present_flag**는 nnpfc_id와 연관된 포스트-프로세싱 필터의 복잡도를 나타내는 하나 이상의 선택스 요소들이 존재함을 특정한다. 0과 동일한 nnpfc_complexity_info_present_flag와 연관된 포스트-프로세싱 필터의 복잡도를 나타내는 선택스 요소들이 존재하지 않음을 특정한다.

[0286]

0과 동일한 **nnpfc_parameter_type_idc**는 신경 네트워크가 정수 파라미터들만을 사용함을 나타낸다. 1과 동일한 nnpfc_parameter_type_idc는 신경 네트워크가 부동소수점 또는 정수 파라미터들을 사용할 수 있음을 나타낸다. 2와 동일한 nnpfc_parameter_type_idc는 신경 네트워크가 이진수 파라미터들만을 사용함을 나타낸다. 3과 동일한 nnpfc_parameter_type_idc는 ITU-T | ISO/IEC에 의한 향후 사용을 위해 예약되고, 본 문헌의 이러한 편집에 순응하는 비트스트림들에 존재하지 않을 것이다. 본 문헌의 이러한 편집에 순응하는 디코더들은 3과 동일한 nnpfc_parameter_type_idc를 갖는 NNPF SEI 메시지들을 무시할 것이다.

[0287]

0, 1, 2, 및 3과 동일한 **nnpfc_log2_parameter_bit_length_minus3**은 신경 네트워크가 각각 8, 16, 32, 및 64 초과 비트 길이의 파라미터들을 사용하지 않음을 나타낸다. nnpfc_parameter_type_idc가 존재하고 nnpfc_log2_parameter_bit_length_minus3이 존재하지 않을 때, 신경 네트워크는 1 초과 비트 길이의 파라미터들을 사용하지 않는다.

[0288]

nnpfc_num_parameters_idc는 포스트-프로세싱 필터에 대한 신경 네트워크 파라미터들의 최대 수를 2048의 제곱의 단위들로 나타낸다. 0과 동일한 nnpfc_num_parameters_idc는 신경 네트워크 파라미터들의 최대 수가 알려져 있지 않음을 나타낸다. nnpfc_num_parameters_idc의 값은 0 내지 52(이를 포함함)의 범위 내에 있을 것이다. 52 초과 nnpfc_num_parameters_idc의 값들은 ITU-T | ISO/IEC에 의한 향후 사용을 위해 예약되고, 본 문헌의 이러한 편집에 순응하는 비트스트림들에 존재하지 않을 것이다. 본 문헌의 이러한 편집에 순응하는 디코더들은 52 초과 nnpfc_num_parameters_idc를 갖는 NNPF SEI 메시지들을 무시할 것이다.

- [0289] nnpfc_num_parameters_idc의 값이 0 초과인 경우, 변수 maxNumParameters는 하기와 같이 도출된다:
- [0290] $\text{maxNumParameters} = (2048 \ll \text{nnpfc_num_parameters_idc}) - 1$
- [0291] 포스트-프로세싱 필터의 신경 네트워크 파라미터들의 수가 maxNumParameters 이하일 것이라는 것이 비트스트림 순응성의 요건이다.
- [0292] 0 초과인 nnpfc_num_kmac_operations_idc는 포스트-프로세싱 필터의 샘플당 곱셈-누산 연산들의 최대 수가 nnpfc_num_kmac_operations_idc * 1000 이하임을 나타낸다. 0과 동일한 nnpfc_num_kmac_operations_idc는 네트워크의 곱셈-누산 연산들의 최대 수가 알려져 있지 않음을 나타낸다. nnpfc_num_kmac_operations_idc의 값은 0 내지 $2^{32} - 1$ (이를 포함함)의 범위 내에 있을 것이다.
- [0293] 0 초과인 nnpfc_total_kilobyte_size는 신경 네트워크에 대한 비압축 파라미터들을 저장하는 데 필요한 총 크기를 킬로바이트 단위로 나타낸다. 비트 단위의 총 크기는 각각의 파라미터를 저장하는 데 사용되는 비트들의 합과 동일하거나 그보다 큰 수이다. nnpfc_total_kilobyte_size는 8000으로 나누어 반올림된, 비트 단위의 총 크기이다. 0과 동일한 nnpfc_total_kilobyte_size는 신경 네트워크에 대한 파라미터들을 저장하는 데 필요한 총 크기가 알려져 있지 않음을 나타낸다. nnpfc_total_kilobyte_size의 값은 0 내지 $2^{32} - 1$ (이를 포함함)의 범위 내에 있을 것이다.
- [0294] nnpfc_reserved_zero_bit_b는 본 문헌의 이러한 편집에 순응하는 비트스트림들에서 0과 동일할 것이다. 디코더들은 nnpfc_reserved_zero_bit_b가 0과 동일하지 않은 NNPFC SEI 메시지들을 무시할 것이다.
- [0295] nnpfc_payload_byte[i]는 ISO/IEC 15938-17에 순응하는 비트스트림의 i-번째 바이트를 포함한다. i의 모든 존재 값들에 대한 바이트 시퀀스 nnpfc_payload_byte[i]는 ISO/IEC 15938-17에 순응하는 완전 비트스트림일 것이다.
- [0296] 표 12는 JVET-AB2006에서 규정된 신경 네트워크 포스트-필터 활성화 SEI 메시지의 선택스를 예시한다.
- [0297] [표 12]

nn_post_filter_activation(payloadSize) {	디스크립터
nnpfa_target_id	ue(v)
nnpfa_cancel_flag	u(1)
if(!nnpfa_cancel_flag) {	
nnpfa_persistence_flag	u(1)
}	

- [0298]
- [0299] 표 12와 관련하여, JVET-AB2006은 하기의 시맨틱들을 규정한다.
- [0300] 신경 네트워크 포스트-필터 활성화(neural-network post-filter activation, NNPPFA) SEI 메시지는 픽처들의 세트의 포스트-프로세싱 필터링을 위해, nnpfa_target_id에 의해 식별된 표적 신경 네트워크 포스트-프로세싱 필터의 가능한 사용을 활성화시키거나 또는 비활성화시킨다.
- [0301] 주 - 예를 들어, 포스트-프로세싱 필터들이 상이한 목적들을 위해 의도되거나 상이한 색상 성분들을 필터링할 때, 동일한 픽처에 대해 존재하는 여러 개의 NNPPFA SEI 메시지들이 있을 수 있다.
- [0302] nnpfa_target_id는, 현재 픽처와 관련되고 nnpfa_target_id와 동일한 nnpfc_id를 갖는 하나 이상의 신경 네트워크 포스트-프로세싱 필터 특성 SEI 메시지들에 의해 특정되는 표적 신경 네트워크 포스트-프로세싱 필터를 나타낸다.
- [0303] nnpfa_target_id의 값은 0 내지 $2^{32} - 2$ (이를 포함함)의 범위 내에 있을 것이다. 256 내지 511(이를 포함함)의 그리고 2^{31} 내지 $2^{32} - 2$ (이를 포함함)의 nnpfa_target_id의 값들은 ITU-T | ISO/IEC에 의한 향후 사용을 위해 예약된다. 256 내지 511(이를 포함함)의 범위 내의, 또는 2^{31} 내지 $2^{32} - 2$ (이를 포함함)의 범위 내의 nnpfa_target_id를 갖는 NNPPFA SEI 메시지와 겹치는, 본 문헌의 이러한 편집에 순응하는 디코더들은 SEI 메시지를 무시할 것이다.
- [0304] nnpfa_target_id의 특정 값을 갖는 NNPPFA SEI 메시지는 하기의 조건들 중 하나 또는 둘 모두가 참이 아닌 경우 현재 PU에 존재하지 않을 것이다.

- [0305] - 현재 CLVS 내에는, 디코딩 순서에서 현재 PU에 선행하는 PU에 존재하는 `nnpfa_target_id`의 특정 값과 동일한 `nnpfc_id`를 갖는 NNPF SEI 메시지가 있다.
- [0306] - 현재 PU 내의 `nnpfa_target_id`의 특정 값과 동일한 `nnpfc_id`를 갖는 NNPF SEI 메시지가 있다.
- [0307] PU가 `nnpfc_id`의 특정 값을 갖는 NNPF SEI 메시지 및 `nnpfc_id`의 특정값과 동일한 `nnpfa_target_id`를 갖는 NNPF SEI 메시지 둘 모두를 포함할 때, NNPF SEI 메시지는 디코딩 순서에서 NNPF SEI 메시지에 선행할 것이다.
- [0308] 1과 동일한 `nnpfa_cancel_flag`는 현재 SEI 메시지와 동일한 `nnpfa_target_id`를 갖는 임의의 이전 NNPF SEI 메시지에 의해 확립된 표적 신경 네트워크 포스트-프로세싱 필터의 지속성이 취소됨, 즉 표적 신경 네트워크 포스트-프로세싱 필터가 현재 SEI 메시지와 동일한 `nnpfa_target_id` 및 0과 동일한 `nnpfa_cancel_flag`를 갖는 다른 NNPF SEI 메시지에 의해 활성화되지 않는 한, 그것이 더 이상 사용되지 않음을 나타낸다. 0과 동일한 `nnpfa_cancel_flag`는 `nnpfa_persistence_flag`가 후행함을 나타낸다.
- [0309] `nnpfa_persistence_flag`는 현재 계층에 대한 표적 신경 네트워크 포스트-프로세싱 필터의 지속성을 특정한다.
- [0310] 0과 동일한 `nnpfa_persistence_flag`는 표적 신경 네트워크 포스트-프로세싱 필터가 현재 픽처에 대한 포스트-프로세싱 필터링에만 사용될 수 있음을 특정한다.
- [0311] 1과 동일한 `nnpfa_persistence_flag`는 하기의 조건들 중 하나 이상이 참일 때까지, 표적 신경 네트워크 포스트-프로세싱 필터가 출력 순서에서 현재 계층의 현재 픽처 및 모든 후속 픽처들에 대한 포스트-프로세싱 필터링에 사용될 수 있음을 특정한다.
- [0312] - 현재 계층의 새로운 CLVS가 시작된다.
- [0313] - 비트스트림이 종료된다.
- [0314] - 현재 SEI 메시지와 동일한 `nnpfa_target_id` 및 1과 동일한 `nnpfa_cancel_flag`를 갖는 NNPF SEI 메시지와 연관된 현재 계층 내의 픽처가 출력되고, 이는 출력 순서에서 현재 픽처에 후행한다.
- [0315] 주 - 표적 신경 네트워크 포스트-프로세싱 필터는 현재 SEI 메시지와 동일한 `nnpfa_target_id` 및 1과 동일한 `nnpfa_cancel_flag`를 갖는 NNPF SEI 메시지와 연관된 현재 계층 내의 이러한 후속 픽처에 대해 적용되지 않는다.
- [0316] JVET-AB2006에서 규정된 신경 네트워크 포스트-필터 특성 SEI 메시지 메시지는 이상적이지 않을 수 있다. 특히, 예를 들어, JVET-AB2006에서의 시그널링은 서브계층별로 서브계층 상에서의 프레임 레이트 업샘플링을 지원하기에 불충분할 수 있다. 본 명세서에 기술된 기법들에 따르면, 시간적 확장성에 대해 프레임 레이트 업샘플링을 위한 더 양호한 지원을 제공하기 위해 선택스 및 시맨틱들이 제공된다.
- [0317] 도 1은 본 개시내용의 하나 이상의 기법들에 따른, 비디오 데이터를 코딩하도록(즉, 인코딩하고/하거나 디코딩하도록) 구성될 수 있는 시스템의 일례를 예시하는 블록도이다. 시스템(100)은 본 개시내용의 하나 이상의 기법들에 따른, 비디오 데이터를 캡슐화할 수 있는 시스템의 일례를 표현한다. 도 1에 도시된 바와 같이, 시스템(100)은 소스 디바이스(102), 통신 매체(110), 및 목적지 디바이스(120)를 포함한다. 도 1에 도시된 예에서, 소스 디바이스(102)는 비디오 데이터를 인코딩하도록 그리고 인코딩된 비디오 데이터를 통신 매체(110)에 송신하도록 구성된 임의의 디바이스를 포함할 수 있다. 목적지 디바이스(120)는 인코딩된 비디오 데이터를 통신 매체(110)를 통하여 수신하도록 그리고 인코딩된 비디오 데이터를 디코딩하도록 구성된 임의의 디바이스를 포함할 수 있다. 소스 디바이스(102) 및/또는 목적지 디바이스(120)는 유선 및/또는 무선 통신을 위해 구비된 컴퓨팅 디바이스들을 포함할 수 있고, 예를 들어, 셋톱 박스, 디지털 비디오 레코더, 텔레비전, 데스크톱, 랩톱 또는 태블릿 컴퓨터, 게이밍 콘솔, 의료용 이미징 디바이스, 및 예를 들어 스마트폰, 셀룰러 전화, 개인 게이밍 디바이스를 포함하는 모바일 디바이스를 포함할 수 있다.
- [0318] 통신 매체(110)는 무선 및 유선 통신 매체, 및/또는 저장 디바이스들의 임의의 조합을 포함할 수 있다. 통신 매체(110)는 동축 케이블, 광섬유 케이블, 연선 케이블, 무선 송신기 및 수신기, 라우터, 스위치, 리피터, 기지국, 또는 다양한 디바이스들 및 장소들 사이의 통신을 용이하게 하는 데 유용할 수 있는 임의의 다른 장비를 포함할 수 있다. 통신 매체(110)는 하나 이상의 네트워크들을 포함할 수 있다. 예를 들어, 통신 매체(110)는 월드 와이드 웹, 예를 들어 인터넷에의 액세스를 가능하게 하도록 구성된 네트워크를 포함할 수 있다. 네트워크는 하나 이상의 전기통신 프로토콜들의 조합에 따라 동작할 수 있다. 전기통신 프로토콜들은 독점적인 태양들

을 포함할 수 있고/있거나 표준화된 전기통신 프로토콜들을 포함할 수 있다. 표준화된 전기통신 프로토콜들의 예들은 DVB(Digital Video Broadcasting) 표준, ATSC(Advanced Television Systems Committee) 표준, ISDB(Integrated Services Digital Broadcasting) 표준, DOCSIS(Data Over Cable Service Interface Specification) 표준, GSM(Global System Mobile Communications) 표준, CDMA(code division multiple access) 표준, 3GPP(3rd Generation Partnership Project) 표준, ETSI(European Telecommunications standards Institute) 표준, IP(Internet Protocol) 표준, WAP(Wireless Application Protocol) 표준, 및 IEEE(Institute of Electrical and Electronics Engineers) 표준을 포함한다.

[0319] 저장 디바이스들은 데이터를 저장할 수 있는 임의의 유형의 디바이스 또는 저장 매체를 포함할 수 있다. 저장 매체는 유형적인 또는 비일시적인 컴퓨터 판독가능 매체를 포함할 수 있다. 컴퓨터 판독가능 매체는 광학 디스크, 플래시 메모리, 자기 메모리, 또는 임의의 다른 적합한 디지털 저장 매체들을 포함할 수 있다. 일부 예들에서, 메모리 디바이스 또는 그의 부분들은 비휘발성 메모리로서 기술될 수 있고, 다른 예들에서, 메모리 디바이스들의 부분들은 휘발성 메모리로서 기술될 수 있다. 휘발성 메모리들의 예들은 랜덤 액세스 메모리(RAM), 동적 랜덤 액세스 메모리(DRAM), 및 정적 랜덤 액세스 메모리(SRAM)를 포함할 수 있다. 비휘발성 메모리들의 예들은 자기 하드 디스크, 광 디스크, 플로피 디스크, 플래시 메모리, 또는 EPROM(electrically programmable memory) 또는 EEPROM(electrically erasable and programmable memory)의 형태를 포함할 수 있다. 저장 디바이스(들)는 메모리 카드(예컨대, SD(Secure Digital) 메모리 카드), 내부/외부 하드 디스크 드라이브, 및/또는 내부/외부 솔리드 스테이트 드라이브(solid state drive)를 포함할 수 있다. 데이터는 정의된 파일 포맷에 따라 저장 디바이스 상에 저장될 수 있다.

[0320] 도 4는 시스템(100)의 일 구현예에 포함될 수 있는 컴포넌트들의 일례를 예시하는 개념도이다. 도 4에 도시된 예시적인 구현예에서, 시스템(100)은 하나 이상의 컴퓨팅 디바이스들(402A 내지 402N), 텔레비전 서비스 네트워크(404), 텔레비전 서비스 제공자 사이트(406), 광역 네트워크(408), 근거리 통신 네트워크(410), 및 하나 이상의 콘텐츠 제공자 사이트들(412A 내지 412N)을 포함한다. 도 4에 도시된 구현예는, 예를 들어, 영화, 라이브 스포츠 이벤트 등과 같은 디지털 미디어 콘텐츠, 및 데이터 및 애플리케이션들 및 이들과 연관된 미디어 프레젠테이션들이 컴퓨팅 디바이스들(402A 내지 402N)과 같은 복수의 컴퓨팅 디바이스들에 분산되고 그들에 의해 액세스될 수 있게 하도록 구성될 수 있는 시스템의 일례를 표현한다. 도 4에 도시된 예에서, 컴퓨팅 디바이스들(402A 내지 402N)은 텔레비전 서비스 네트워크(404), 광역 네트워크(408), 및/또는 근거리 통신 네트워크(410) 중 하나 이상으로부터 데이터를 수신하도록 구성된 임의의 디바이스를 포함할 수 있다. 예를 들어, 컴퓨팅 디바이스들(402A 내지 402N)은 유선 및/또는 무선 통신을 위해 구비될 수 있고, 하나 이상의 데이터 채널들을 통해 서비스들을 수신하도록 구성될 수 있고, 소위 스마트 텔레비전을 포함한 텔레비전, 셋톱 박스 및 디지털 비디오 레코더를 포함할 수 있다. 또한, 컴퓨팅 디바이스들(402A 내지 402N)은 데스크톱, 랩톱, 또는 태블릿 컴퓨터들, 게이밍 콘솔, 예를 들어, "스마트"폰, 셀룰러 전화, 및 개인 게이밍 디바이스를 포함하는 모바일 디바이스를 포함할 수 있다.

[0321] 텔레비전 서비스 네트워크(404)는 텔레비전 서비스들을 포함할 수 있는 디지털 미디어 콘텐츠가 분배되는 것을 가능하게 하도록 구성된 네트워크의 일례이다. 예를 들어, 텔레비전 서비스 네트워크(404)는 공공 OTA(over-the-air) 텔레비전 네트워크, 공공 또는 가입 기반 위성 텔레비전 서비스 제공자 네트워크, 및 공공 또는 가입 기반 케이블 텔레비전 서비스 제공자 네트워크 및/또는 OTT(over the top) 또는 인터넷 서비스 제공자를 포함할 수 있다. 일부 예들에서, 텔레비전 서비스 네트워크(404)가 주로, 텔레비전 서비스들이 제공되는 것을 가능하게 하는 데 사용될 수 있지만, 텔레비전 서비스 네트워크(404)는 또한, 본 명세서에 기술된 전기통신 프로토콜들의 임의의 조합에 따라 다른 유형들의 데이터 및 서비스들이 제공되는 것을 가능하게 할 수 있다는 점에 유의해야 한다. 또한, 일부 예들에서, 텔레비전 서비스 네트워크(404)는 컴퓨팅 디바이스들(402A 내지 402N) 중 하나 이상과 텔레비전 서비스 제공자 사이트(406) 사이의 양방향 통신을 가능하게 할 수 있다는 점에 유의해야 한다. 텔레비전 서비스 네트워크(404)는 무선 및/또는 유선 통신 매체들의 임의의 조합을 포함할 수 있다. 텔레비전 서비스 네트워크(404)는 동축 케이블, 광섬유 케이블, 연선 케이블, 무선 송신기 및 수신기, 라우터, 스위치, 리피터, 기지국, 또는 다양한 디바이스들과 사이트들 사이의 통신을 용이하게 하는 데 유용할 수 있는 임의의 다른 장비를 포함할 수 있다. 텔레비전 서비스 네트워크(404)는 하나 이상의 전기통신 프로토콜들의 조합에 따라 동작할 수 있다. 전기통신 프로토콜들은 독점적인 태양들을 포함할 수 있고/있거나 표준화된 전기통신 프로토콜들을 포함할 수 있다. 표준화된 전기통신 프로토콜들의 예들은 DVB 표준, ATSC 표준, ISDB 표준, DTMB 표준, DMB 표준, DOCSIS(Data Over Cable Service Interface Specification) 표준, HbbTV 표준, W3C 표준 및 UPnP 표준을 포함한다.

- [0322] 다시 도 4를 참조하면, 텔레비전 서비스 제공자 사이트(406)는 텔레비전 서비스 네트워크(404)를 통해 텔레비전 서비스를 분배하도록 구성될 수 있다. 예를 들어, 텔레비전 서비스 제공자 사이트(406)는 하나 이상의 방송국, 케이블 텔레비전 제공자, 또는 위성 텔레비전 제공자, 또는 인터넷 기반 텔레비전 제공자를 포함할 수 있다. 예를 들어, 텔레비전 서비스 제공자 사이트(406)는 위성 업링크/다운링크를 통해 텔레비전 프로그래밍을 포함한 송신을 수신하도록 구성될 수 있다. 또한, 도 4에 도시된 바와 같이, 텔레비전 서비스 제공자 사이트(406)는 광역 네트워크(408)와 통신할 수 있고, 콘텐츠 제공자 사이트들(412A 내지 412N)로부터 데이터를 수신하도록 구성될 수 있다. 일부 예들에서, 텔레비전 서비스 제공자 사이트(406)는 텔레비전 스튜디오를 포함할 수 있고, 콘텐츠는 그로부터 유래될 수 있다는 점에 유의해야 한다.
- [0323] 광역 네트워크(408)는 패킷 기반 네트워크를 포함할 수 있고, 하나 이상의 전기통신 프로토콜들의 조합에 따라 동작할 수 있다. 전기통신 프로토콜들은 독점적인 태양들을 포함할 수 있고/있거나 표준화된 전기통신 프로토콜들을 포함할 수 있다. 표준화된 전기통신 프로토콜들의 예들은 GSM 표준, CDMA 표준, 3GPP 표준, ETSI 표준, EN(European standard), IP 표준, WAP 표준, 및 예를 들어, IEEE 802 표준들(예컨대, Wi-Fi) 중 하나 이상과 같은 IEEE 표준을 포함한다. 광역 네트워크(408)는 무선 및/또는 유선 통신 매체들의 임의의 조합을 포함할 수 있다. 광역 네트워크(408)는 동축 케이블, 광섬유 케이블, 연선 케이블, 이더넷 케이블, 무선 송신기 및 수신기, 라우터, 스위치, 리피터, 기지국, 또는 다양한 디바이스들과 사이트들 사이의 통신을 용이하게 하는 데 유용할 수 있는 임의의 다른 장비를 포함할 수 있다. 하나의 예에서, 광역 네트워크(408)는 인터넷을 포함할 수 있다. 근거리 통신 네트워크(410)는 패킷 기반 네트워크를 포함할 수 있고, 하나 이상의 전기통신 프로토콜들의 조합에 따라 동작할 수 있다. 근거리 통신 네트워크(410)는 액세스 및/또는 물리적 인프라구조의 레벨들에 기초하여 광역 네트워크(408)와 구별될 수 있다. 예를 들어, 근거리 통신 네트워크(410)는 보안 홈 네트워크를 포함할 수 있다.
- [0324] 다시 도 4를 참조하면, 콘텐츠 제공자 사이트들(412A 내지 412N)은 텔레비전 서비스 제공자 사이트(406) 및/또는 컴퓨팅 디바이스들(402A 내지 402N)에 멀티미디어 콘텐츠를 제공할 수 있는 사이트들의 예들을 표현한다. 예를 들어, 콘텐츠 제공자 사이트는 텔레비전 서비스 제공자 사이트(406)에 멀티미디어 파일들 및/또는 스트림들을 제공하도록 구성된 하나 이상의 스튜디오 콘텐츠 서버들을 갖는 스튜디오를 포함할 수 있다. 하나의 예에서, 콘텐츠 제공자 사이트들(412A 내지 412N)은 IP 스위트(suite)를 사용하여 멀티미디어 콘텐츠를 제공하도록 구성될 수 있다. 예를 들어, 콘텐츠 제공자 사이트는 RTSP(Real Time Streaming Protocol), HTTP 등에 따라 수신기 디바이스에 멀티미디어 콘텐츠를 제공하도록 구성될 수 있다. 또한, 콘텐츠 제공자 사이트들(412A 내지 412N)은 광역 네트워크(408)를 통해 수신기 디바이스 컴퓨팅 디바이스들(402A 내지 402N) 및/또는 텔레비전 서비스 제공자 사이트(406) 중 하나 이상에 하이퍼텍스트 기반 콘텐츠 등을 포함하는 데이터를 제공하도록 구성될 수 있다. 콘텐츠 제공자 사이트들(412A 내지 412N)은 하나 이상의 웹 서버들을 포함할 수 있다. 콘텐츠 제공자 사이트들(412A 내지 412N)에 의해 제공된 데이터는 데이터 포맷들에 따라 정의될 수 있다.
- [0325] 다시 도 1을 참조하면, 소스 디바이스(102)는 비디오 소스(104), 비디오 인코더(106), 데이터 캡슐화기(107) 및 인터페이스(108)를 포함한다. 비디오 소스(104)는 비디오 데이터를 캡처하고/하거나 저장하도록 구성된 임의의 디바이스를 포함할 수 있다. 예를 들어, 비디오 소스(104)는 비디오 카메라 및 그에 동작가능하게 커플링된 저장 디바이스를 포함할 수 있다. 비디오 인코더(106)는 비디오 데이터를 수신하도록 그리고 비디오 데이터를 표현하는 컴플라이언트 비트스트림을 생성하도록 구성된 임의의 디바이스를 포함할 수 있다. 컴플라이언트 비트스트림은 비디오 디코더가 그로부터 비디오 데이터를 수신 및 재생할 수 있는 비트스트림을 지칭할 수 있다. 컴플라이언트 비트스트림의 태양들은 비디오 코딩 표준에 따라 정의될 수 있다. 컴플라이언트 비트스트림을 생성할 때, 비디오 인코더(106)는 비디오 데이터를 압축할 수 있다. 압축은 (뷰어에게 식별가능하거나 식별불가능한) 손실이 있을 수 있거나 또는 손실이 없을 수 있다. 도 5는 본 명세서에서 기술된, 비디오 데이터를 인코딩하기 위한 기법들을 구현할 수 있는 비디오 인코더(500)의 일례를 도시한 블록도이다. 예시적인 비디오 인코더(500)가 별개의 기능 블록들을 갖는 것으로 예시되어 있지만, 그러한 예시는 설명의 목적을 위한 것이고 비디오 인코더(500) 및/또는 그의 서브컴포넌트들을 특정 하드웨어 또는 소프트웨어 아키텍처로 제한하지 않는다는 점에 유의해야 한다. 비디오 인코더(500)의 기능들은 하드웨어, 펌웨어, 및/또는 소프트웨어 구현들의 임의의 조합을 이용하여 실현될 수 있다.
- [0326] 비디오 인코더(500)는 픽처 영역들의 인트라 예측 코딩 및 인터 예측 코딩을 수행할 수 있고, 이 때문에, 하이브리드 비디오 인코더로 지칭될 수 있다. 도 5에 도시된 예에서, 비디오 인코더(500)는 소스 비디오 블록들을 수신한다. 일부 예들에서, 소스 비디오 블록들은 코딩 구조에 따라 나뉘었던 픽처의 영역들을 포함할 수 있다. 예를 들어, 소스 비디오 데이터는 매크로블록들, CTU들, CB들, 그의 서브분할부들, 및/또는 다른 등가 코딩 유

닛을 포함할 수 있다. 일부 예들에서, 비디오 인코더(500)는 소스 비디오 블록들의 추가 세분화들을 수행하도록 구성될 수 있다. 본 명세서에 기술되는 기법은, 일반적으로, 인코딩 전에 그리고/또는 인코딩 동안에 소스 비디오 데이터가 어떻게 분할되는지와는 무관하게 비디오 코딩에 적용가능하다는 점에 유의해야 한다. 도 5에 도시된 예에서, 비디오 인코더(500)는 합산기(502), 변환 계수 생성기(504), 계수 양자화 유닛(506), 역양자화 및 변환 계수 프로세싱 유닛(508), 합산기(510), 인트라 예측 프로세싱 유닛(512), 인터 예측 프로세싱 유닛(514), 필터 유닛(516), 및 엔트로피 인코딩 유닛(518)을 포함한다. 도 5에 도시된 바와 같이, 비디오 인코더(500)는 소스 비디오 블록들을 수신하고 비트스트림을 출력한다.

[0327]

도 5에 도시된 예에서, 비디오 인코더(500)는 소스 비디오 블록으로부터 예측 비디오 블록을 감산함으로써 잔차 데이터를 생성할 수 있다. 예측 비디오 블록의 선택은 상세히 후술된다. 합산기(502)는 이러한 감산 연산을 수행하도록 구성된 컴포넌트를 표현한다. 일례에서, 비디오 블록들의 감산은 픽셀 도메인에서 발생한다. 변환 계수 생성기(504)는 이산 코사인 변환(DCT), 이산 사인 변환(DST), 또는 개념적으로 유사한 변환과 같은 변환을, 잔차 블록 또는 그의 서브분할부들에 적용하여(예컨대, 4개의 8×8 변환들이 잔차 값들의 16×16 어레이에 적용될 수 있음), 잔차 변환 계수들의 세트를 생성한다. 변환 계수 생성기(504)는 이산 삼각법 변환들의 계열 - 그들의 근사화들을 포함함 - 에 포함되는 변환들의 임의의 그리고 모든 조합들을 수행하도록 구성될 수 있다. 변환 계수 생성기(504)는 변환 계수들을 계수 양자화 유닛(506)으로 출력할 수 있다. 계수 양자화 유닛(506)은 변환 계수들의 양자화를 수행하도록 구성될 수 있다. 양자화 프로세스는 계수들 중 일부 또는 전부와 연관된 비트 깊이를 감소시킬 수 있다. 양자화의 정도는 인코딩된 비디오 데이터의 레이트-왜곡(즉, 비트레이트 대 비디오 품질)을 변경할 수 있다. 양자화의 정도는 양자화 파라미터(QP)를 조정함으로써 수정될 수 있다. 양자화 파라미터는 슬라이스 레벨 값들 및/또는 CU 레벨 값들(예컨대, CU 델타 QP 값들)에 기초하여 결정될 수 있다. QP 데이터는 변환 계수들의 특정 세트를 양자화하기 위한 QP를 결정하는 데 사용되는 임의의 데이터를 포함할 수 있다. 도 5에 도시된 바와 같이, 양자화된 변환 계수들(이들은 레벨 값들로 지칭될 수 있음)은 역양자화 및 변환 계수 프로세싱 유닛(508)으로 출력된다. 역양자화 및 변환 계수 프로세싱 유닛(508)은 재구성된 잔차 데이터를 생성하기 위해 역양자화 및 역변환을 적용하도록 구성될 수 있다. 도 5에 도시된 바와 같이, 합산기(510)에서, 재구성된 잔차 데이터가 예측 비디오 블록에 추가될 수 있다. 이러한 방식으로, 인코딩된 비디오 블록은 재구성될 수 있고, 생성된 재구성된 비디오 블록은 주어진 예측, 변환, 및/또는 양자화에 대한 인코딩 품질을 평가하는 데 사용될 수 있다. 비디오 인코더(500)는 다수의 코딩 패스들을 수행하도록(예컨대, 예측, 변환 파라미터들, 및 양자화 파라미터들 중 하나 이상을 변경하면서 인코딩을 수행하도록) 구성될 수 있다. 비트스트림 또는 다른 시스템 파라미터들의 레이트-왜곡은 재구성된 비디오 블록들의 평가에 기초하여 최적화될 수 있다. 또한, 재구성된 비디오 블록들은 후속 블록들을 예측하기 위한 기준으로서 저장 및 사용될 수 있다.

[0328]

다시 도 5를 참조하면, 인트라 예측 프로세싱 유닛(512)은 비디오 블록이 코딩될 인트라 예측 모드를 선택하도록 구성될 수 있다. 인트라 예측 프로세싱 유닛(512)은 프레임을 평가하도록 그리고 현재 블록을 인코딩하는 데 사용할 인트라 예측 모드를 결정하도록 구성될 수 있다. 전술된 바와 같이, 가능한 인트라 예측 모드들은 평면 예측 모드들, DC 예측 모드들, 및 각도 예측 모드들을 포함할 수 있다. 또한, 일부 예들에서, 크로마 성분에 대한 예측 모드는 루마 예측 모드 동안의 예측 모드로부터 추론될 수 있다는 점에 유의해야 한다. 인트라 예측 프로세싱 유닛(512)은 하나 이상의 코딩 패스들을 수행한 후에 인트라 예측 모드를 선택할 수 있다. 또한, 하나의 예에서, 인트라 예측 프로세싱 유닛(512)은 레이트-왜곡 분석에 기초하여 예측 모드를 선택할 수 있다. 도 5에 도시된 바와 같이, 인트라 예측 프로세싱 유닛(512)은 인트라 예측 데이터(예컨대, 선택스 요소들)를 엔트로피 인코딩 유닛(518) 및 변환 계수 생성기(504)로 출력한다. 전술된 바와 같이, 잔차 데이터에 대해 수행되는 변환은 모드 종속적일 수 있다(예컨대, 2차 변환 매트릭스는 예측 모드에 기초하여 결정될 수 있음).

[0329]

다시 도 5를 참조하면, 인터 예측 프로세싱 유닛(514)은 현재 비디오 블록에 대한 인터 예측 코딩을 수행하도록 구성될 수 있다. 인터 예측 프로세싱 유닛(514)은 소스 비디오 블록들을 수신하도록 그리고 비디오 블록의 PU들에 대한 모션 벡터를 계산하도록 구성될 수 있다. 모션 벡터는 기준 프레임 내의 예측 블록에 대한 현재 비디오 프레임 내의 비디오 블록의 예측 유닛의 변위를 나타낼 수 있다. 인터 예측 코딩은 하나 이상의 기준 픽처를 사용할 수 있다. 또한, 모션 예측은 단방향 예측(하나의 모션 벡터를 사용) 또는 양방향 예측(2개의 모션 벡터를 사용함)일 수 있다. 인터 예측 프로세싱 유닛(514)은, 예를 들어 절대차의 합(sum of absolute difference, SAD), 제곱차의 합(sum of square difference, SSD), 또는 다른 차이 메트릭(metric)들에 의해 결정된 픽셀 차이를 계산함으로써 예측 블록을 선택하도록 구성될 수 있다. 전술된 바와 같이, 모션 벡터는 모션 벡터 예측에 따라 결정 및 특정될 수 있다. 인터 예측 프로세싱 유닛(514)은, 전술된 바와 같이, 모션 벡터 예

측을 수행하도록 구성될 수 있다. 인터 예측 프로세싱 유닛(514)은 모션 예측 데이터를 사용하여 예측 블록을 생성하도록 구성될 수 있다. 예를 들어, 인터 예측 프로세싱 유닛(514)은 프레임 버퍼(도 5에 도시되지 않음) 내에 예측 비디오 블록을 위치시킬 수 있다. 인터 예측 프로세싱 유닛(514)은, 추가로, 하나 이상의 보간 필터를 재구성된 잔차 블록에 적용하여 모션 추정에 사용하기 위한 서브-정수 픽셀 값들을 계산하도록 구성될 수 있다는 점에 유의해야 한다. 인터 예측 프로세싱 유닛(514)은 계산된 모션 벡터에 대한 모션 예측 데이터를 엔트로피 인코딩 유닛(518)에 출력할 수 있다.

[0330] 다시 도 5를 참조하면, 필터 유닛(516)은 재구성된 비디오 블록들 및 코딩 파라미터들을 수신하고, 수정된 재구성된 비디오 데이터를 출력한다. 필터 유닛(516)은 디블록킹 및/또는 샘플 적응적 오프셋(SAO) 필터링을 수행하도록 구성될 수 있다. SAO 필터링은 재구성된 비디오 데이터에 오프셋을 추가함으로써 재구성을 개선하는 데 사용될 수 있는 비선형 진폭 맵핑이다. 도 5에 도시된 바와 같이, 인트라 예측 프로세싱 유닛(512) 및 인터 예측 프로세싱 유닛(514)은 필터 유닛(516)을 통해, 수정된 재구성된 비디오 블록을 수신할 수 있다는 점에 유의해야 한다. 엔트로피 인코딩 유닛(518)은 양자화된 변환 계수들 및 예측 선택스 데이터(즉, 인트라 예측 데이터 및 모션 예측 데이터)를 수신한다. 일부 예들에서, 계수 양자화 유닛(506)은, 계수들이 엔트로피 인코딩 유닛(518)으로 출력되기 전, 양자화된 변환 계수들을 포함하는 매트릭스의 스캔을 수행할 수 있다는 점에 유의해야 한다. 다른 예들에서, 엔트로피 인코딩 유닛(518)은 스캔을 수행할 수 있다. 엔트로피 인코딩 유닛(518)은 본 명세서에서 기술된 기법들 중 하나 이상에 따라 엔트로피 인코딩을 수행하도록 구성될 수 있다. 이러한 방식으로, 비디오 인코더(500)는 본 개시내용의 하나 이상의 기법들에 따른, 인코딩된 비디오 데이터를 생성하도록 구성된 디바이스의 일례를 표현한다.

[0331] 다시 도 1을 참조하면, 데이터 캡슐화기(107)는 인코딩된 비디오 데이터를 수신할 수 있고, 정의된 데이터 구조에 따라 컴플라이언트 비트스트림, 예컨대, NAL 유닛들의 시퀀스를 생성할 수 있다. 컴플라이언트 비트스트림을 수신하는 디바이스는 그로부터 비디오 데이터를 재생할 수 있다. 또한, 전송된 바와 같이, 서브-비트스트림 추출은, 컴플라이언트 비트스트림을 수신하는 디바이스가 수신된 비트스트림에서 데이터를 폐기 및/또는 수정함으로써 새로운 컴플라이언트 비트스트림을 형성하는 프로세스를 지칭할 수 있다. 용어 "순응형 비트스트림"은 용어 "컴플라이언트 비트스트림" 대신에 사용될 수 있다는 점에 유의해야 한다. 하나의 예에서, 데이터 캡슐화기(107)는 본 명세서에 기술된 하나 이상의 기법들에 따라 선택스를 생성하도록 구성될 수 있다. 데이터 캡슐화기(107)는 비디오 인코더(106)와 반드시 동일한 물리적 디바이스에 위치될 필요가 없다는 점에 유의해야 한다. 예를 들어, 비디오 인코더(106) 및 데이터 캡슐화기(107)에 의해 수행되는 것으로 기술된 기능들은 도 4에 도시된 디바이스들 사이에 분배될 수 있다.

[0332] 전송된 바와 같이, JVET-AB2006에서의 시그널링은 서브계층별로 서브계층 상에서의 프레임 레이트 업샘플링을 지원하기에 불충분할 수 있다. 본 명세서의 기법들에 따르면, 시간적 서브계층들이 드롭되는 경우들에 대해 포스트-프로세싱 필터에 의해 생성된 보간된 픽처들의 수를 특정하기 위한 선택스가 제공된다. 즉, 하나의 예에서, 본 명세서의 기법들에 따르면, 선택스 요소 `nnpfc_interpolated_pics[ i ][ j ]`는 더 양호한 시간적 확장성을 위한 지원을 제공하기 위해 선택스 요소들 `nnpfc_interpolated_pics[ i ]` 대신에 사용된다.

[0333] 예시적인 예로서, 오리지널 비트스트림은 초당 60 프레임(fps)의 비트스트림일 수 있고, 2개의 시간적 서브계층들, 즉 0-번째 서브계층 및 1-번째 서브계층을 포함할 수 있으며, 이때 0-번째 서브계층은 30 fps이고, 0-번째 서브계층과 1-번째 서브계층은 함께 60 fps이다. 또한, JVET-AB2006에 따른 NNPF 메시지는 오리지널 60 fps 비트스트림의 경우에 0과 동일한 `nnpfc_num_input_pics_minus2` 및 1과 동일한 `nnpfc_interpolated_pics[ 0 ]`을 시그널링하여, 보간된 비트스트림과 오리지널 비트스트림이 함께, 픽처들의 세트가 120 fps로 되는 결과를 가져오도록 할 수 있다. 이 경우, 최고 시간적 서브계층이 오리지널 비트스트림으로부터 드롭되어, 오리지널 비트스트림이 30 fps 비트스트림이 되도록 하는 경우, 120 fps를 달성하기 위해, `nnpfc_interpolated_pics[ 0 ]`의 값은 3으로 변경되어야 한다. JVET-AB2006은 그러한 경우에 대해 `nnpfc_interpolated_pics[ 0 ]`의 값을 변경하기 위한 메커니즘을 제공하지 않는다. 본 명세서의 기법들에 따르면, 이 경우, 추가적인 루프가 `nnpfc_interpolated_pics[ 0 ][ 1 ] = 1` 및 `nnpfc_interpolated_pics[ 0 ][ 0 ] = 3`을 시그널링하는 것을 허용한다.

[0334] 표 13A 및 표 13B는 본 명세서의 기법들에 따라, 시간적 서브계층들이 드롭되는 경우들에 대해, 포스트-프로세싱 필터에 의해 생성된 보간된 픽처들의 수를 특정하기 위한 예시적인 신경 네트워크 포스트-필터 특성 SEI 메시지의 관련 선택스를 예시한다. 표 13A와 관련하여, JVET-T2001은 선택스 요소 `sps_max_sublayers_minus1`이 시퀀스 파라미터 세트 선택스 `seq_parameter_set_rbsp()`에서 어디에 포함되는지를 제공하고, 하기의 시맨틱들을

갖는다는 점에 유의해야 한다:

- [0335] **sps_max_sublayers_minus1** + 1은 SPS를 참조하는 각각의 CLVS에 존재할 수 있는 시간적 서브계층들의 최대 수를 특정한다.
- [0336] **sps_video_parameter_set_id**가 0 초과인 경우, **sps_max_sublayers_minus1**의 값은 0 내지 **vps_max_sublayers_minus1**(이를 포함함)의 범위 내에 있을 것이다.
- [0337] 그렇지 않은 경우(**sps_video_parameter_set_id**가 0과 동일한 경우), 하기가 적용된다:
- [0338] - **sps_max_sublayers_minus1**의 값은 0 내지 6(이를 포함함)의 범위 내에 있을 것이다.
- [0339] - **vps_max_sublayers_minus1**의 값은 **sps_max_sublayers_minus1**과 동일한 것으로 추론된다.
- [0340] - **NumSubLayersInLayerInOLS[0][0]**의 값은 **sps_max_sublayers_minus1** + 1과 동일한 것으로 추론된다.
- [0341] - **vps_ols_ptl_idx[0]**의 값은 0과 동일한 것으로 추론되고, **vps_ptl_max_tid[vps_ols_ptl_idx[0]]**의 값, 즉, **vps_ptl_max_tid[0]**은 **sps_max_sublayers_minus1**과 동일한 것으로 추론된다.
- [0342] 여기서,
- [0343] **vps_max_sublayers_minus1** + 1은 VPS에 의해 특정된 계층에 존재할 수 있는 시간적 서브계층들의 최대 수를 특정한다. **vps_max_sublayers_minus1**의 값은 0 내지 6(이를 포함함)의 범위 내에 있을 것이다.
- [0344] [표 13A]

nn_post_filter_characteristics(payloadSize) {	디스크립터
nnpfc_id	ue(v)
nnpfc_mode_idc	ue(v)
if(nnpfc_mode_idc == 1) {	
while(!byte_aligned())	
nnpfc_reserved_zero_bit_a	u(1)
nnpfc_tag_uri	st(v)
nnpfc_uri	st(v)
}	
nnpfc_formatting_and_purpose_flag	u(1)
if(nnpfc_formatting_and_purpose_flag) {	
nnpfc_purpose	ue(v)
/* input and output formatting */	
if(nnpfc_purpose == 2 nnpfc_purpose == 4)	
nnpfc_out_sub_c_flag	u(1)
else if(nnpfc_purpose == 3 nnpfc_purpose == 4) {	
nnpfc_pic_width_in_luma_samples	ue(v)
nnpfc_pic_height_in_luma_samples	ue(v)
}	
else if(nnpfc_purpose == 5) {	
nnpfc_num_input_pics_minus2	ue(v)
for(i = 0; i <= nnpfc_num_input_pics_minus2; i++)	
for(j = 0; j <= sps_max_sublayers_minus1; j++)	
nnpfc_interpolated_pics[i][j]	ue(v)
}	
nnpfc_component_last_flag	u(1)
...	
}	
}	

[0345]

[0346] [표 13B]

nn_post_filter_characteristics(payloadSize, NumSubLayersMinus1) {	디스크립터
nnpfc_id	ue(v)
nnpfc_mode_idc	ue(v)
if(nnpfc_mode_idc == 1) {	
while(!byte_aligned())	
nnpfc_reserved_zero_bit_a	u(1)
nnpfc_tag_uri	st(v)
nnpfc_uri	st(v)
}	
nnpfc_formatting_and_purpose_flag	u(1)
if(nnpfc_formatting_and_purpose_flag) {	
nnpfc_purpose	ue(v)
/* input and output formatting */	
if(nnpfc_purpose == 2 nnpfc_purpose == 4)	
nnpfc_out_sub_c_flag	u(1)
else if(nnpfc_purpose == 3 nnpfc_purpose == 4) {	
nnpfc_pic_width_in_luma_samples	ue(v)
nnpfc_pic_height_in_luma_samples	ue(v)
}	
else if(nnpfc_purpose == 5) {	
nnpfc_num_input_pics_minus2	ue(v)
for(i = 0; i <= nnpfc_num_input_pics_minus2; i++)	
for(j = 0; j <= NumSubLayersMinus1; j++)	
nnpfc_interpolated_pics[i][j]	ue(v)
}	
nnpfc_component_last_flag	u(1)
...	
}	
}	

[0347]

[0348] 표 13A 및 표 13B와 관련하여, 시맨틱들은 상기에 그리고 하기에 제공되는 시맨틱들에 기초할 수 있다:

[0349] 신경 네트워크 포스트-필터 특성(NNPFC) SEI 메시지는 포스트-프로세싱 필터로서 사용될 수 있는 신경 네트워크를 명시한다. 특정 픽처들을 위한 특정된 포스트-프로세싱 필터들의 사용은 신경 네트워크 포스트-필터 활성화 SEI 메시지들로 표시된다.

[0350] 이러한 SEI 메시지의 사용은 하기의 변수들의 정의를 필요로 한다:

[0351] - 본 명세서에서 각각 CroppedWidth 및 CroppedHeight로 표기되는, 루마 샘플들의 단위의 크롭(crop)된 디코딩된 출력 픽처 폭 및 높이.

[0352] - 존재 시, 포스트-프로세싱 필터에 대한 입력으로 사용되는, 0 내지 numInputPics - 1(이를 포함함)의 범위 내의 idx를 갖는 크롭된 디코딩된 출력 픽처들의 루마 샘플 어레이 CroppedYPic[idx] 및 크로마 샘플 어레이들 CroppedCbPic[idx] 및 CroppedCrPic[idx].

[0353] - 최고 시간적 서브계층 디코딩된 HighestTid(또는 하나의 예에서, 최고 시간적 서브계층 존재 HighestTid)

[0354] - 크롭된 디코딩된 출력 픽처들의 루마 샘플 어레이에 대한 비트 깊이 BitDepth_Y.

[0355] - 존재하는 경우, 크롭된 디코딩된 출력 픽처들의 크로마 샘플 어레이들에 대한 비트 깊이 BitDepth_C.

[0356] - 본 명세서에서 ChromaFormatIdc로 표기되는 크로마 포맷 표시자.

[0357] - nnpfc_auxiliary_inp_idc가 1과 동일할 때, 0 내지 1(이를 포함함)의 범위 내의 실수일 필터링 강도 제어 값 StrengthControlVal.

[0358] **nnpfc_interpolated_pics[i][j]**는 j가 비트스트림에 존재하는 또는 디코딩된 최고 시간적 서브계층일 때 포스트-프로세싱 필터에 대한 입력으로서 사용되는 i-번째 픽처와 (i + 1)-번째 픽처 사이에서 포스트-프로세싱 필터에 의해 생성된 보간된 픽처들의 수를 특정한다.

[0359] 포스트-프로세싱 필터에 대한 입력으로서 사용되는 픽처들의 수를 특정하는 변수 numInputPics 및 포스트-프로세싱 필터로부터 생성된 픽처들의 총 수를 특정하는 변수 numOutputPics가 하기와 같이 도출된다:

[0360] Htid를 최고 Tid로 한다.

```

if( nnpfc_purpose == 5 ) {
    numInputPics = nnpfc_num_input_pics_minus2 + 2
    for( i = 0, numOutputPics = 0; i <= numInputPics - 2; i++ )
        numOutputPics += nnpfc_interpolated_pics[ i ][ Htid ]
} else
    numInputPics = 1

```

[0361]

[0362] 여기서 JVET-T2001의 경우:

[0363] 디코딩될 최고 시간적 서브계층을 식별하는 변수 Htid는 하기와 같이 도출된다:

[0364] - 하기는 비트스트림의 제1 AU에서 적용된다:

[0365] - 이러한 규격에서 특정되지 않은 일부 외부 수단이 Htid를 설정하는 데 이용가능한 경우, Htid는 외부 수단에 의해 설정된다.

[0366] - 이와 달리, `opi_htid_plus1`이 비트스트림의 제1 AU 내의 OPI NAL 유닛에 존재하는 경우, Htid는 $((opi_htid_plus1 > 0) ? opi_htid_plus1 - 1 : 0)$ 과 동일하게 설정된다.

[0367] - 그렇지 않은 경우, Htid는 `vps_ptl_max_tid[ vps_ols_ptl_idx[ TargetOlsIdx ] ]`와 동일하게 설정된다.

[0368] 주 - `sps_video_parameter_set_id`가 0과 동일할 때, Htid는 `sps_max_sublayers_minus1`과 동일하게 설정될 것이다.

[0369] 다른 예에서, 상기 코드는 최고 Tid를 나타내기 위해 변수 HighestTid를 사용할 수 있고, 상기 코드의 일부는,

[0370] `numOutputPics += nnpfc_interpolated_pics[ i ][ Htid ]`

[0371] 로부터

[0372] `numOutputPics += nnpfc_interpolated_pics[ i ][ HighestTid ]`로 변경될 수 있다.

[0373] 여기서, 선택스 요소 `opi_htid_plus1`은 동작 포인트 정보 선택스 `operating_point_information_rbsp()`에 포함되고, 하기의 시맨틱들을 갖는다:

[0374] 0과 동일한 `opi_htid_plus1`은 OPI NAL 유닛에서 `opi_htid_plus1`이 제공되는 다음 CVS에 이르지만 이를 포함하지 않는, 디코딩 순서에서 현재 CVS 및 다음 CVS들 내의 모든 픽처들이 0과 동일한 `ph_recovery_poc_cnt`를 갖는 IRAP 픽처들 또는 GDR 픽처들임을 특징한다. 0 초과와 `opi_htid_plus1`은 OPI NAL 유닛에서 `opi_htid_plus1`이 제공되는 다음 CVS에 이르지만 이를 포함하지 않는, 디코딩 순서에서 현재 CVS 및 다음 CVS들 내의 모든 픽처들이 `opi_htid_plus1` 미만의 TemporalId를 가짐을 특징한다.

[0375] 여기서, 선택스 요소 `vps_ptl_max`는 비디오 파라미터 세트 선택스 `video_parameter_set_rbsp()`에 포함되고, 하기의 시맨틱들을 갖는다:

[0376] `vps_ptl_max_tid[ i ]`는 `vps_ols_ptl_idx[ olsIdx ]`가 *i*와 동일하도록, 레벨 정보가 VPS에서 *i*-번째 `profile_tier_level()` 선택스 구조에 존재하는 최고 서브계층 표현의 TemporalId, 및 OLS 인덱스 `olsIdx`를 갖는 OLS들에 존재하는 최고 서브계층 표현의 TemporalId를 특징한다. `vps_ptl_max_tid[ i ]`의 값은 0 내지 `vps_max_sublayers_minus1`(이를 포함함)의 범위 내에 있을 것이다. `vps_default_ptl_dpb_hrd_max_tid_flag`가 1과 동일할 때, `vps_ptl_max_tid[ i ]`의 값은 `vps_max_sublayers_minus1`과 동일한 것으로 추론된다.

[0377] 표 13B와 관련하여, 시간적 서브계층들의 수 - 1을 제공하는 입력 변수 `NumSubLayerMinus1`이 대신, `MaxNumSubLayersMinus1` 또는 `NumSubLayersInLayerInOLS`로 불릴 수 있다는 점에 주의해야 하며, 여기서, `MaxNumSubLayersMinus1` 또는 `NumSubLayersInLayerInOLS`은 JVET-T2001에서 제공된 정의들에 기초할 수 있다.

[0378] JVET-AB2006에 따르면, `nnpfc_constant_patch_size_flag`가 0과 동일할 때(즉, 포스트-프로세싱 필터가 `nnpfc_patch_width_minus1` 및 `nnpfc_patch_height_minus1`에 의해 나타내진 패치 크기의 양의 정수 배수인 임의의 패치 크기를 수용할 때), 포스트-프로세싱 필터의 인접 입력 텐서들의 중첩 수평 및 수직 샘플 카운트들이 패치 크기 폭 및 패치 크기 높이를 각각 표현하는 변수들 `inpPatchWidth` 및 `inpPatchHeight`와 관련하여 고려될 필요가 있다고 확인된다. JVET-AB2006에 따르면, 프로세스 `DeriveInputTensors()`에 의해 정의된 바와 같은 포

스트-프로세싱 필터에 대한 실제 입력 크기는 $yP = -\text{overlapSize}$ 내지 $yP < \text{inpPatchHeight} + \text{overlapSize}$ 의 범위의 yP 값들 및 $xP = -\text{overlapSize}$ 내지 $xP < \text{inpPatchWidth} + \text{overlapSize}$ 의 범위의 xP 값들에 대해 정의되는 "for loop"들에 의해 계산된다. 이는 전형적인 신경 네트워크 포스트 필터들에 의해 요구될 수 있는 바와 같이, 포스트-프로세싱 필터에 대한 실제 입력 크기가 특정 값(예컨대, 8)의 배수가 아닌 결과를 가져올 수 있다.

- [0379] 예를 들어, JVET-AB2006에 따르면, $\text{nnpfc_patch_width_minus1} + 1 = 8$, $\text{nnpfc_patch_height_minus1} + 1 = 8$, $\text{overlapSize} = 3$, 및 $\text{nnpfc_constant_patch_size_flag} = 0$ 인 경우가 있을 수 있다. 이러한 경우, 필터는 (8,8)의 크기 배수의 입력을 수용한다. 또한, $\text{InpPatchWidth} = (\text{nnpfc_patch_width_minus1} + 1) * k$ (즉, $8 * k$) 및 $\text{InpPatchHeight} = (\text{nnpfc_patch_height_minus1} + 1) * k$ (즉, $8 * k$)인 경우, 포스트-프로세싱 필터에 대한 입력은 $(8 * k + 2 * \text{overlapSize}, 8 * k + 2 * \text{overlapSize})$ (즉, $(8 * k + 6, 8 * k + 6)$)이다. $8 * k + 6$ 의 값은 반드시 8로 나눌 수 있는 것은 아닌데, 이는 예상치 못한 그리고/또는 잘못된 필터링 결과들을 초래할 수 있다.
- [0380] 본 명세서의 기법들에 따르면, 예시적인 신경 네트워크 포스트-필터 특성 SEI 메시지는 전형적인 신경 네트워크 포스트 필터에 의해 요구될 수 있는 바와 같이, 포스트-프로세싱 필터에 대한 실제 입력 크기가 특정 값의 배수임을 보장하는 것을 가능하게 하는 시맨틱들을 갖는 선택스 요소들을 포함할 수 있다. 하나의 예에서, 본 명세서의 기법들에 따르면, 예시적인 신경 네트워크 포스트-필터 특성 SEI 메시지는 표 6 및/또는 표 13A 및 표 13B의 예들에서 그리고 하기에 제공되는 선택스 및 시맨틱들에 기초할 수 있다:
- [0381] 1과 동일한 $\text{nnpfc_constant_patch_size_flag}$ 는 포스트-프로세싱 필터가 입력으로서 $\text{nnpfc_patch_width_minus1}$ 및 $\text{nnpfc_patch_height_minus1}$ 에 의해 나타내지는, 정확히 패치 크기를 수용함을 나타낸다. 0과 동일한 $\text{nnpfc_constant_patch_size_flag}$ 는 포스트-프로세싱 필터가 입력으로서 $\text{nnpfc_patch_width_minus1}$, $\text{nnpfc_patch_height_minus1}$, 및 nnpfc_overlap 에 의해 나타내지는 패치 크기의 양의 정수 배수인 임의의 패치 크기를 수용함을 나타낸다.
- [0382] $\text{nnpfc_patch_height_minus1} + 1$ 은 $\text{nnpfc_constant_patch_size_flag}$ 가 1과 동일할 때, 포스트-프로세싱 필터에 대한 입력을 위해 요구되는 패치 크기의 수직 샘플 카운트들을 나타낸다. $\text{nnpfc_patch_height_minus1}$ 의 값은 0 내지 $\text{Min}(32766, \text{CroppedHeight} - 1)$ (이를 포함함)의 범위 내에 있을 것이다.
- [0383] 변수들 inpPatchWidth 및 inpPatchHeight 가 각각 패치 크기 폭 및 패치 크기 높이라고 하자.
- [0384] $\text{nnpfc_constant_patch_size_flag}$ 가 0과 동일한 경우, 하기가 적용된다.
- [0385] - inpPatchWidth 및 inpPatchHeight 의 값들은 본 문헌에서 특정되지 않은 외부 수단에 의해 제공되거나, 또는 포스트-프로세서 자체에 의해 설정된다.
- [0386] - $\text{inpPatchWidth} + 2 * \text{overlapSize}$ 의 값은 $\text{nnpfc_patch_width_minus1} + 1$ 의 양의 정수 배수일 것이고, CroppedWidth 이하일 것이다. - $\text{inpPatchHeight} + 2 * \text{overlapSize}$ 의 값은 $\text{nnpfc_patch_height_minus1} + 1$ 의 양의 정수 배수일 것이고, CroppedHeight 이하일 것이다.
- [0387] 이와 달리($\text{nnpfc_constant_patch_size_flag}$ 가 1과 동일한 경우), inpPatchWidth 의 값은 $\text{nnpfc_patch_width_minus1} + 1$ 과 동일하게 설정되고, inpPatchHeight 의 값은 $\text{nnpfc_patch_height_minus1} + 1$ 과 동일하게 설정된다.
- [0388] 하나의 예에서, $\text{nnpfc_patch_height_minus1}$ 은 하기에 기초할 수 있다:
- [0389] $\text{nnpfc_patch_height_minus1} + 1$ 은 $\text{nnpfc_constant_patch_size_flag}$ 가 1과 동일할 때, 포스트-프로세싱 필터에 대한 입력을 위해 요구되는 패치 크기의 수직 샘플 카운트들을 나타낸다. $\text{nnpfc_patch_height_minus1}$ 의 값은 0 내지 $\text{Min}(32766, \text{CroppedHeight} - 1)$ (이를 포함함)의 범위 내에 있을 것이다.
- [0390] 변수들 inpPatchWidth 및 inpPatchHeight 가 각각 패치 크기 폭 및 패치 크기 높이라고 하자.
- [0391] $\text{nnpfc_constant_patch_size_flag}$ 가 0과 동일한 경우, 하기가 적용된다.
- [0392] - inpPatchWidth 및 inpPatchHeight 의 값들은 본 문헌에서 특정되지 않은 외부 수단에 의해 제공되거나, 또는 포스트-프로세서 자체에 의해 설정된다.
- [0393] - $\text{inpPatchWidth} + 2 * \text{overlapSize}$ 의 값은 $\text{nnpfc_patch_width_minus1} + 1$ 의 양의 정수 배수일 것이다. 그리고, inpPatchWidth 은 CroppedWidth 이하일 것이다. $\text{inpPatchHeight} + 2 * \text{overlapSize}$ 의 값은

$\text{nnpfc_patch_height_minus1} + 1$ 의 양의 정수 배수일 것이다. 그리고, inpPatchHeight 는 CroppedHeight 이하일 것이다.

- [0394] 이와 달리($\text{nnpfc_constant_patch_size_flag}$ 가 1과 동일한 경우), inpPatchWidth 의 값은 $\text{nnpfc_patch_width_minus1} + 1$ 과 동일하게 설정되고, inpPatchHeight 의 값은 $\text{nnpfc_patch_height_minus1} + 1$ 과 동일하게 설정된다.
- [0395] 하나의 예에서, $\text{nnpfc_patch_height_minus1}$ 은 하기에 기초할 수 있다:
- [0396] $\text{nnpfc_patch_height_minus1} + 1$ 은 $\text{nnpfc_constant_patch_size_flag}$ 가 1과 동일할 때, 포스트-프로세싱 필터에 대한 입력을 위해 요구되는 패치 크기의 수직 샘플 카운트들을 나타낸다. $\text{nnpfc_patch_height_minus1}$ 의 값은 0 내지 $\text{Min}(32766, \text{CroppedHeight} - 1)$ (이를 포함함)의 범위 내에 있을 것이다.
- [0397] 변수들 inpPatchWidth 및 inpPatchHeight 가 각각 패치 크기 폭 및 패치 크기 높이라고 하자.
- [0398] $\text{nnpfc_constant_patch_size_flag}$ 가 0과 동일한 경우, 하기가 적용된다.
- [0399] - inpPatchWidth 및 inpPatchHeight 의 값들은 본 문헌에서 특정되지 않은 외부 수단에 의해 제공되거나, 또는 포스트-프로세서 자체에 의해 설정된다.
- [0400] - inpPatchWidth 의 값은 $\text{nnpfc_patch_width_minus1} + 1 - 2 * \text{overlapSize}$ 의 양의 정수 배수일 것이고, CroppedWidth 이하일 것이다. - inpPatchHeight 의 값은 $\text{nnpfc_patch_height_minus1} + 1 - 2 * \text{overlapSize}$ 의 양의 정수 배수일 것이고, CroppedHeight 이하일 것이다.
- [0401] 이와 달리($\text{nnpfc_constant_patch_size_flag}$ 가 1과 동일한 경우), inpPatchWidth 의 값은 $\text{nnpfc_patch_width_minus1} + 1$ 과 동일하게 설정되고, inpPatchHeight 의 값은 $\text{nnpfc_patch_height_minus1} + 1$ 과 동일하게 설정된다.
- [0402] 하나의 예에서, 본 명세서의 기법들에 따르면, 예시적인 신경 네트워크 포스트-필터 특성 SEI 메시지는 표 6 및 /또는 표 13A 및 표 13B의 예들에서 그리고 하기에 제공되는 선택스 및 시맨틱들에 기초할 수 있다:
- [0403] 1과 동일한 $\text{nnpfc_constant_patch_size_flag}$ 는 포스트-프로세싱 필터가 입력으로서 $\text{nnpfc_patch_width_minus1}$ 및 $\text{nnpfc_patch_height_minus1}$ 에 의해 나타내지는, 정확히 패치 크기를 수용함을 나타낸다. 0과 동일한 $\text{nnpfc_constant_patch_size_flag}$ 는 포스트-프로세싱 필터가 입력으로서 $\text{nnpfc_patch_width_minus1}$, $\text{nnpfc_patch_height_minus1}$, 및 nnpfc_overlap 에 의해 나타내지는 패치 크기의 양의 정수 거듭제곱인 임의의 패치 크기를 수용함을 나타낸다.
- [0404] $\text{nnpfc_patch_height_minus1} + 1$ 은 $\text{nnpfc_constant_patch_size_flag}$ 가 1과 동일할 때, 포스트-프로세싱 필터에 대한 입력을 위해 요구되는 패치 크기의 수직 샘플 카운트들을 나타낸다. $\text{nnpfc_patch_height_minus1}$ 의 값은 0 내지 $\text{Min}(32766, \text{CroppedHeight} - 1)$ (이를 포함함)의 범위 내에 있을 것이다.
- [0405] 변수들 inpPatchWidth 및 inpPatchHeight 가 각각 패치 크기 폭 및 패치 크기 높이라고 하자.
- [0406] $\text{nnpfc_constant_patch_size_flag}$ 가 0과 동일한 경우, 하기가 적용된다.
- [0407] - inpPatchWidth 및 inpPatchHeight 의 값들은 본 문헌에서 특정되지 않은 외부 수단에 의해 제공되거나, 또는 포스트-프로세서 자체에 의해 설정된다.
- [0408] - $\text{inpPatchWidth} + 2 * \text{overlapSize}$ 의 값은 $\text{nnpfc_patch_width_minus1} + 1$ 의 양의 정수 거듭제곱일 것이고, CroppedWidth 이하일 것이다. - $\text{inpPatchHeight} + 2 * \text{overlapSize}$ 의 값은 $\text{nnpfc_patch_height_minus1} + 1$ 의 양의 정수 거듭제곱일 것이고, CroppedHeight 이하일 것이다.
- [0409] 이와 달리($\text{nnpfc_constant_patch_size_flag}$ 가 1과 동일한 경우), inpPatchWidth 의 값은 $\text{nnpfc_patch_width_minus1} + 1$ 과 동일하게 설정되고, inpPatchHeight 의 값은 $\text{nnpfc_patch_height_minus1} + 1$ 과 동일하게 설정된다.
- [0410] 이러한 방식으로, 비디오 인코더(500)는 신경 네트워크 포스트-필터 특성 메시지를 시그널링하도록, 그리고 신경 네트워크 포스트-필터 특성 메시지와 연관된 다수의 서브계층들 각각에 대해, 신경 네트워크 포스트-필터 특성 메시지에서, 포스트-프로세싱 필터에 의해 생성된 보간된 픽처들의 수를 특징하는 선택스 요소를 시그널링하도록 구성된 디바이스의 일례를 표현한다.

- [0411] 다시 도 1을 참조하면, 인터페이스(108)는 데이터 캡슐화기(107)에 의해 생성된 데이터를 수신하도록 그리고 데이터를 통신 매체에 송신 및/또는 저장하도록 구성된 임의의 디바이스를 포함할 수 있다. 인터페이스(108)는 이더넷 카드와 같은 네트워크 인터페이스 카드를 포함할 수 있고, 광학 송수신기, 무선 주파수 송수신기, 또는 정보를 전송하고/하거나 수신할 수 있는 임의의 다른 유형의 디바이스를 포함할 수 있다. 또한, 인터페이스(108)는 파일이 저장 디바이스 상에 저장되는 것을 가능하게 할 수 있는 컴퓨터 시스템 인터페이스를 포함할 수 있다. 예를 들어, 인터페이스(108)는 PCI(Peripheral Component Interconnect) 및 PCIe(Peripheral Component Interconnect Express) 버스 프로토콜들, 독점적인 버스 프로토콜들, USB(Universal Serial Bus) 프로토콜, I²C, 또는 피어(peer) 디바이스들을 상호접속시키는 데 사용될 수 있는 임의의 다른 논리적 및 물리적 구조를 지원하는 칩셋을 포함할 수 있다.
- [0412] 다시 도 1을 참조하면, 목적지 디바이스(120)는 인터페이스(122), 데이터 탈캡슐화기(123), 비디오 디코더(124), 및 디스플레이(126)를 포함한다. 인터페이스(122)는 통신 매체로부터 데이터를 수신하도록 구성된 임의의 디바이스를 포함할 수 있다. 인터페이스(122)는 이더넷 카드와 같은 네트워크 인터페이스 카드를 포함할 수 있고, 광학 송수신기, 무선 주파수 송수신기, 또는 정보를 수신하고/하거나 전송할 수 있는 임의의 다른 유형의 디바이스를 포함할 수 있다. 또한, 인터페이스(122)는 컴플라이언트 비디오 비트스트림이 저장 디바이스로부터 인출되는(retrieved) 것을 가능하게 하는 컴퓨터 시스템 인터페이스를 포함할 수 있다. 예를 들어, 인터페이스(122)는 PCI 및 PCIe 버스 프로토콜들, 독점적인 버스 프로토콜들, USB 프로토콜, I²C, 또는 피어 디바이스들을 상호접속시키는 데 사용될 수 있는 임의의 다른 논리적 및 물리적 구조를 지원하는 칩셋을 포함할 수 있다. 데이터 탈캡슐화기(123)는 본 명세서에 기술된 예시적인 선택 구조들 중 임의의 것을 수신 및 파싱하도록 구성될 수 있다.
- [0413] 비디오 디코더(124)는 비트스트림(예컨대, 서브-비트스트림 추출물) 및/또는 그의 허용가능한 변형들을 수신하도록 그리고 이들로부터 비디오 데이터를 복원하도록 구성된 임의의 디바이스를 포함할 수 있다. 디스플레이(126)는 비디오 데이터를 디스플레이하도록 구성된 임의의 디바이스를 포함할 수 있다. 디스플레이(126)는 액정 디스플레이(LCD), 플라스마 디스플레이, 유기 발광 다이오드(OLED) 디스플레이, 또는 다른 유형의 디스플레이와 같은 다양한 디스플레이 디바이스들 중 하나를 포함할 수 있다. 디스플레이(126)는 고화질 디스플레이 또는 초고화질 디스플레이를 포함할 수 있다. 도 1에 도시된 예에서, 비디오 디코더(124)가 디스플레이(126)에 데이터를 출력하는 것으로 기술되어 있지만, 비디오 디코더(124)는 비디오 데이터를 다양한 유형의 디바이스들 및/또는 그의 서브컴포넌트들로 출력하도록 구성될 수 있다는 점에 유의해야 한다. 예를 들어, 비디오 디코더(124)는, 본 명세서에 기술된 바와 같이, 비디오 데이터를 임의의 통신 매체에 출력하도록 구성될 수 있다.
- [0414] 도 6은 본 개시내용의 하나 이상의 기법들(예컨대, 전술된 기준 픽처 리스트 구성을 위한 디코딩 프로세스)에 따른, 비디오 데이터를 디코딩하도록 구성될 수 있는 비디오 디코더의 일례를 예시하는 블록도이다. 하나의 예에서, 비디오 디코더(600)는 변환 데이터를 디코딩하도록 그리고 디코딩된 변환 데이터에 기초하여 변환 계수들로부터 잔차 데이터를 재구성하도록 구성될 수 있다. 비디오 디코더(600)는 인트라 예측 디코딩 및 인터 예측 디코딩을 수행하도록 구성될 수 있고, 이 때문에, 하이브리드 디코더로 지칭될 수 있다. 비디오 디코더(600)는 표 1 내지 표 13B에서 전술된 선택 요소들의 임의의 조합을 파싱하도록 구성될 수 있다. 비디오 디코더(600)는 전술된 프로세스들에 기초하여 또는 이들에 따라, 그리고 표 1 내지 표 13B의 파싱된 값들에 추가로 기초하여 비디오를 디코딩할 수 있다.
- [0415] 도 6에 도시된 예에서, 비디오 디코더(600)는 엔트로피 디코딩 유닛(602), 역양자화 유닛(604), 역변환 계수 프로세싱 유닛(606), 인트라 예측 프로세싱 유닛(608), 인터 예측 프로세싱 유닛(610), 합산기(612), 포스트 필터 유닛(614), 및 기준 버퍼(616)를 포함한다. 비디오 디코더(600)는 비디오 코딩 시스템과 일치하는 방식으로 비디오 데이터를 디코딩하도록 구성될 수 있다. 예시적인 비디오 디코더(600)가 별개의 기능 블록들을 갖는 것으로 예시되지만, 그러한 예시는 설명의 목적을 위한 것이고 비디오 디코더(600) 및/또는 그의 서브컴포넌트들을 특정 하드웨어 또는 소프트웨어 아키텍처로 제한하지 않는다는 점에 유의해야 한다. 비디오 디코더(600)의 기능들은 하드웨어, 펌웨어, 및/또는 소프트웨어 구현들의 임의의 조합을 이용하여 실현될 수 있다.
- [0416] 도 6에 도시된 바와 같이, 엔트로피 디코딩 유닛(602)은 엔트로피 인코딩된 비트스트림을 수신한다. 엔트로피 디코딩 유닛(602)은 엔트로피 인코딩 프로세스에 상반되는 프로세스에 따라 비트스트림으로부터 선택 요소들 및 양자화된 계수들을 디코딩하도록 구성될 수 있다. 엔트로피 디코딩 유닛(602)은 전술된 엔트로피 코딩 기법들 중 임의의 것에 따라 엔트로피 디코딩을 수행하도록 구성될 수 있다. 엔트로피 디코딩 유닛(602)은 비디오 코딩 표준과 일치하는 방식으로, 인코딩된 비트스트림에서 선택 요소들에 대한 값들을 결정할 수 있다. 도 6

에 도시된 바와 같이, 엔트로피 디코딩 유닛(602)은 비트스트림으로부터 양자화 파라미터, 양자화된 계수 값들, 변환 데이터, 및 예측 데이터를 결정할 수 있다. 도 6에 도시된 예에서, 역양자화 유닛(604) 및 역변환 계수 프로세싱 유닛(606)은 엔트로피 디코딩 유닛(602)으로부터 양자화된 계수 값들을 수신하고, 재구성된 잔차 데이터를 출력한다.

[0417] 다시 도 6을 참조하면, 재구성된 잔차 데이터가 합산기(612)에 제공될 수 있다. 합산기(612)는 재구성된 잔차 데이터를 예측 비디오 블록에 추가할 수 있고, 재구성된 비디오 데이터를 생성할 수 있다. 예측 비디오 블록은 예측 비디오 기법(즉, 인트라 예측 및 인터 프레임 예측)에 따라 결정될 수 있다. 인트라 예측 프로세싱 유닛(608)은 인트라 예측 선택스 요소들을 수신하도록 그리고 기준 버퍼(616)로부터 예측 비디오 블록을 인출하도록 구성될 수 있다. 기준 버퍼(616)는 비디오 데이터의 하나 이상의 프레임들을 저장하도록 구성된 메모리 디바이스를 포함할 수 있다. 인트라 예측 선택스 요소들은 전송된 인트라 예측 모드들과 같은 인트라 예측 모드를 식별할 수 있다. 인터 예측 프로세싱 유닛(610)은 인터 예측 선택스 요소들을 수신할 수 있고, 모션 벡터들을 생성하여 기준 버퍼(616)에 저장된 하나 이상의 기준 프레임들 내의 예측 블록을 식별할 수 있다. 인터 예측 프로세싱 유닛(610)은 모션 보상된 블록들을 생성하여, 가능하게는 보간 필터들에 기초하여 보간을 수행할 수 있다. 서브-픽셀 정밀도의 모션 추정에 사용될 보간 필터들에 대한 식별자들이 선택스 요소들에 포함될 수 있다. 인터 예측 프로세싱 유닛(610)은 보간 필터들을 사용하여 기준 블록의 서브-정수 픽셀들에 대한 보간된 값들을 계산할 수 있다. 포스트 필터 유닛(614)은 재구성된 비디오 데이터에 대해 필터링을 수행하도록 구성될 수 있다. 예를 들어, 포스트 필터 유닛(614)은, 예컨대 비트스트림에 특정된 파라미터들에 기초하여, 디블록킹 및/또는 샘플 적응적 오프셋(SAO) 필터링을 수행하도록 구성될 수 있다. 또한, 일부 예들에서, 포스트 필터 유닛(614)은 독점적인 임의적 필터링(예컨대, 모스quito 노이즈(mosquito noise) 감소와 같은 시각적 향상들)을 수행하도록 구성될 수 있다는 점에 유의해야 한다. 도 6에 도시된 바와 같이, 재구성된 비디오 블록은 비디오 디코더(600)에 의해 출력될 수 있다. 이러한 방식으로, 비디오 디코더(600)는 신경 네트워크 포스트-필터 특성 메시지를 수신하도록, 그리고 신경 네트워크 포스트-필터 특성 메시지와 연관된 다수의 서브계층들 각각에 대해, 신경 네트워크 포스트-필터 특성 메시지로부터, 포스트-프로세싱 필터에 의해 생성된 보간된 픽처들의 수를 특징하는 선택스 요소를 파싱하도록 구성된 디바이스의 일례를 표현한다.

[0418] 하나 이상의 예들에서, 기술된 기능들은 하드웨어, 소프트웨어, 펌웨어, 또는 이들의 임의의 조합으로 구현될 수 있다. 소프트웨어로 구현되는 경우, 기능들은 컴퓨터 판독가능 매체 상에 저장될 수 있거나 또는 그 매체 상의 하나 이상의 명령어 또는 코드로서 전송될 수 있고, 하드웨어 기반 프로세싱 유닛에 의해 실행될 수 있다. 컴퓨터 판독가능 매체는 데이터 저장 매체와 같은 유형적 매체에 대응하는 컴퓨터 판독가능 저장 매체, 또는 예컨대 통신 프로토콜에 따라 하나의 장소에서 다른 장소로의 컴퓨터 프로그램의 이송을 용이하게 하는 임의의 매체를 포함하는 통신 매체를 포함할 수 있다. 이러한 방식으로, 컴퓨터 판독가능 매체는, 일반적으로, (1) 비일시적인 유형적 컴퓨터 판독가능 저장 매체 또는 (2) 신호 또는 반송파와 같은 통신 매체에 대응할 수 있다. 데이터 저장 매체는, 하나 이상의 컴퓨터 또는 하나 이상의 프로세서에 의해 액세스되어 본 개시내용에 기술된 기법들의 구현을 위한 명령어들, 코드 및/또는 데이터 구조들을 인출할 수 있는 임의의 이용가능한 매체일 수 있다. 컴퓨터 프로그램 제품은 컴퓨터 판독가능 매체를 포함할 수 있다.

[0419] 제한이 아닌 예로서, 그러한 컴퓨터 판독가능 저장 매체는 RAM, ROM, EEPROM, CD-ROM 또는 다른 광학 디스크 저장소, 자기 디스크 저장소, 또는 다른 자기 저장 디바이스, 플래시 메모리, 또는 명령어들 또는 데이터 구조들의 형태로 원하는 프로그램 코드를 저장하는 데 사용될 수 있고 컴퓨터에 의해 액세스될 수 있는 임의의 다른 매체를 포함할 수 있다. 또한, 임의의 접속이 사실상 컴퓨터 판독가능 매체로 지칭된다. 예를 들어, 명령어들이 동축 케이블, 광섬유 케이블, 연선, DSL(digital subscriber line), 또는 적외선, 무선, 및 마이크로파와 같은 무선 기술을 이용하여 웹사이트, 서버 또는 다른 원격 소스로부터 송신되는 경우, 동축 케이블, 광섬유 케이블, 연선, DSL, 또는 적외선, 무선, 및 마이크로파와 같은 무선 기술들이 매체의 정의에 포함된다. 그러나, 컴퓨터 판독가능 저장 매체 및 데이터 저장 매체는 접속, 반송파, 신호, 또는 다른 일시적 매체를 포함하는 것이 아니라, 그 대신, 비일시적인 유형적 저장 매체에 관한 것이라는 것을 이해해야 한다. 본 명세서에 사용되는 바와 같이, 디스크(disk) 및 디스크(disc)는 CD(compact disc), 레이저 디스크, 광 디스크, DVD(digital versatile disc), 플로피 디스크 및 블루-레이 디스크를 포함하는데, 여기서 디스크(disk)는 보통 데이터를 자기적으로 재생하는 반면 디스크(disc)는 레이저를 사용하여 광학적으로 데이터를 재생한다. 상기의 것들의 조합이 또한 컴퓨터 판독가능 매체의 범주 내에 포함되어야 한다.

[0420] 명령어들은 하나 이상의 DSP(digital signal processor), 범용 마이크로프로세서, ASIC(application specific integrated circuit), FPGA(field programmable logic array), 또는 다른 등가의 집적 또는 이산 로직 회로부

와 같은 하나 이상의 프로세서에 의해 실행될 수 있다. 따라서, 본 명세서에 사용되는 바와 같이, 용어 "프로세서"는 전술한 구조물 중 임의의 구조물 또는 본 명세서에 기술된 기법들의 구현에 적합한 임의의 다른 구조물을 지칭할 수 있다. 또한, 일부 태양들에서, 본 명세서에 기술된 기능은, 인코딩 및 디코딩을 위해 구성되거나, 또는 조합된 코덱에 포함된 전용 하드웨어 및/또는 소프트웨어 모듈들 내에 제공될 수 있다. 또한, 기법들은 하나 이상의 회로 또는 로직 요소에서 완전하게 구현될 수 있다.

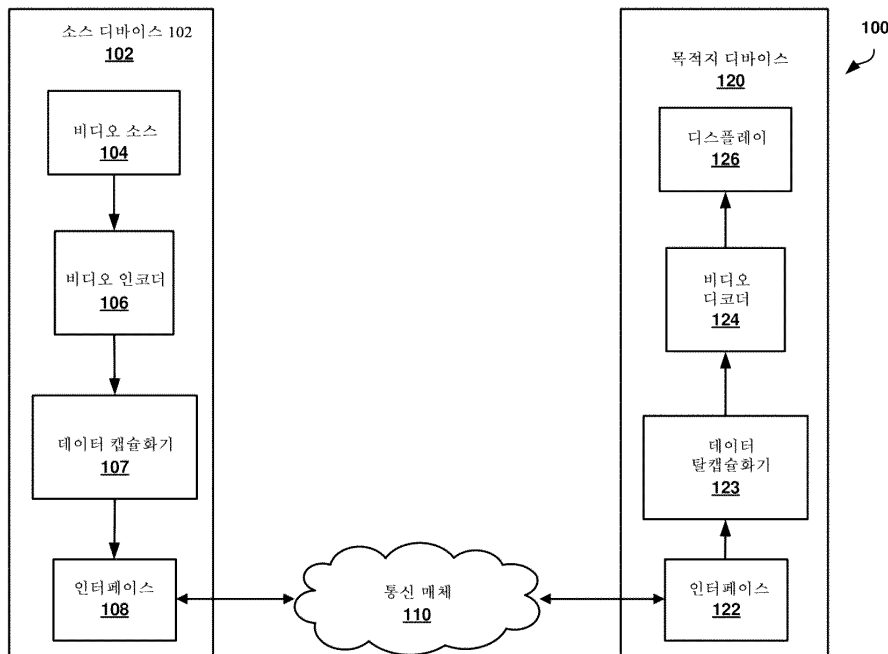
[0421] 본 개시내용의 기법들은 무선 핸드셋, 집적회로(IC) 또는 IC들의 세트(예컨대, 칩셋)를 포함하는 매우 다양한 디바이스들 또는 장치들에서 구현될 수 있다. 개시된 기법들을 수행하도록 구성된 디바이스들의 기능적 태양들을 강조하기 위해 다양한 컴포넌트들, 모듈들, 또는 유닛들이 본 개시내용에 기술되어 있지만, 반드시 상이한 하드웨어 유닛들에 의한 실현을 요구하는 것은 아니다. 오히려, 전술된 바와 같이, 다양한 유닛들이 코덱 하드웨어 유닛에서 조합될 수 있거나, 또는 적합한 소프트웨어 및/또는 펌웨어와 함께, 전술된 바와 같은 하나 이상의 프로세서를 포함하는 상호운용성 하드웨어 유닛들의 집합에 의해 제공될 수 있다.

[0422] 더욱이, 전술한 실시예들 각각에서 사용되는 기지국 디바이스 및 단말기 디바이스의 각각의 기능 블록 또는 다양한 특징부들은, 전형적으로 집적회로 또는 복수의 집적회로인 회로부에 의해 구현되거나 실행될 수 있다. 본 명세서에 설명된 기능들을 실행하도록 설계된 회로부는 범용 프로세서, DSP, ASIC(application specific or general application integrated circuit), FPGA(field programmable gate array), 또는 다른 프로그래밍가능 로직 디바이스, 이산 게이트 또는 트랜지스터 로직, 또는 이산 하드웨어 컴포넌트, 또는 이들의 조합을 포함할 수 있다. 범용 프로세서는 마이크로프로세서일 수 있거나, 또는, 대안적으로, 프로세서는 종래의 프로세서, 제어기, 마이크로제어기 또는 상태 머신(state machine)일 수 있다. 앞서 설명된 범용 프로세서 또는 각각의 회로는 디지털 회로에 의해 구성될 수 있거나, 또는 아날로그 회로에 의해 구성될 수 있다. 또한, 반도체 기술의 발전으로 인해, 현시대의 집적회로들을 대체하는 집적회로를 제조하는 기술이 나타날 때, 이러한 기술에 의한 집적회로가 또한 사용될 수 있다.

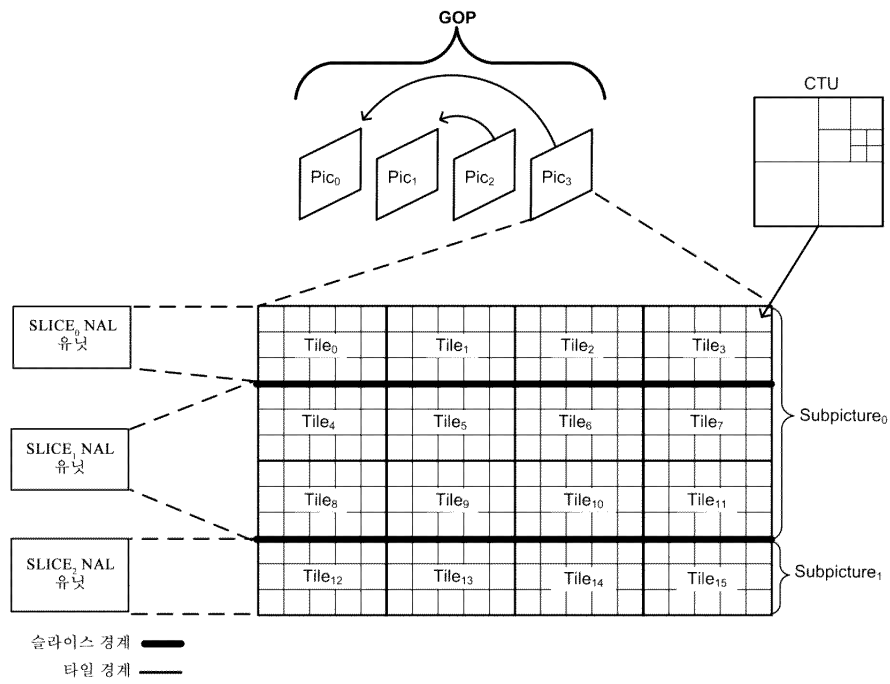
[0423] 다양한 예들이 기술되었다. 이들 및 다른 예들은 다음의 청구범위의 범주 내에 있다.

도면

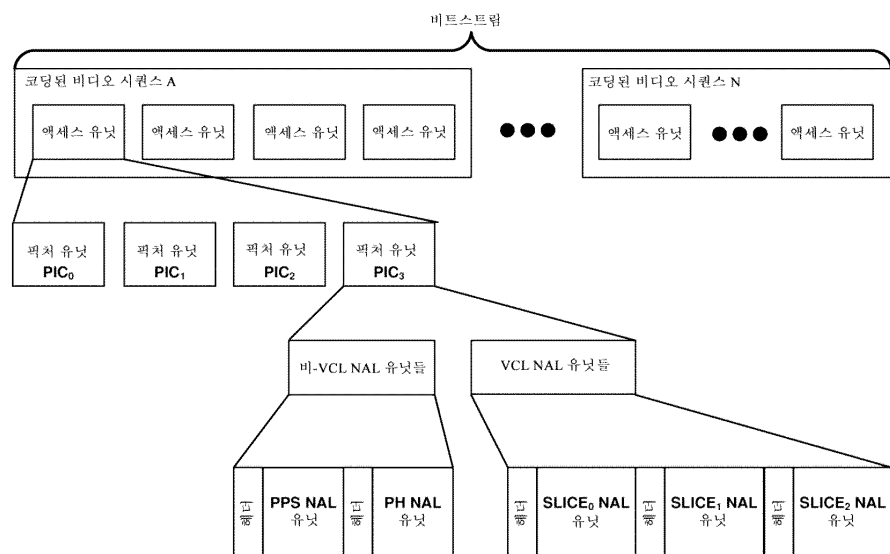
도면1



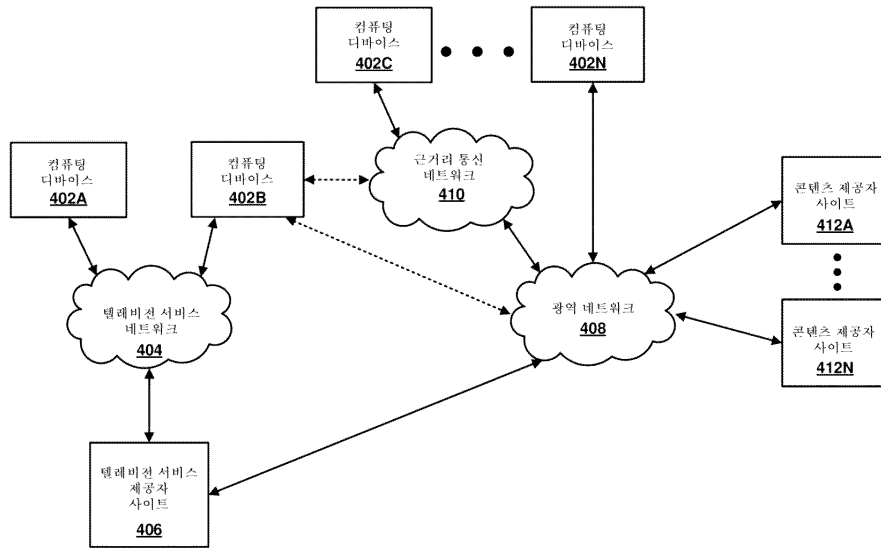
도면2



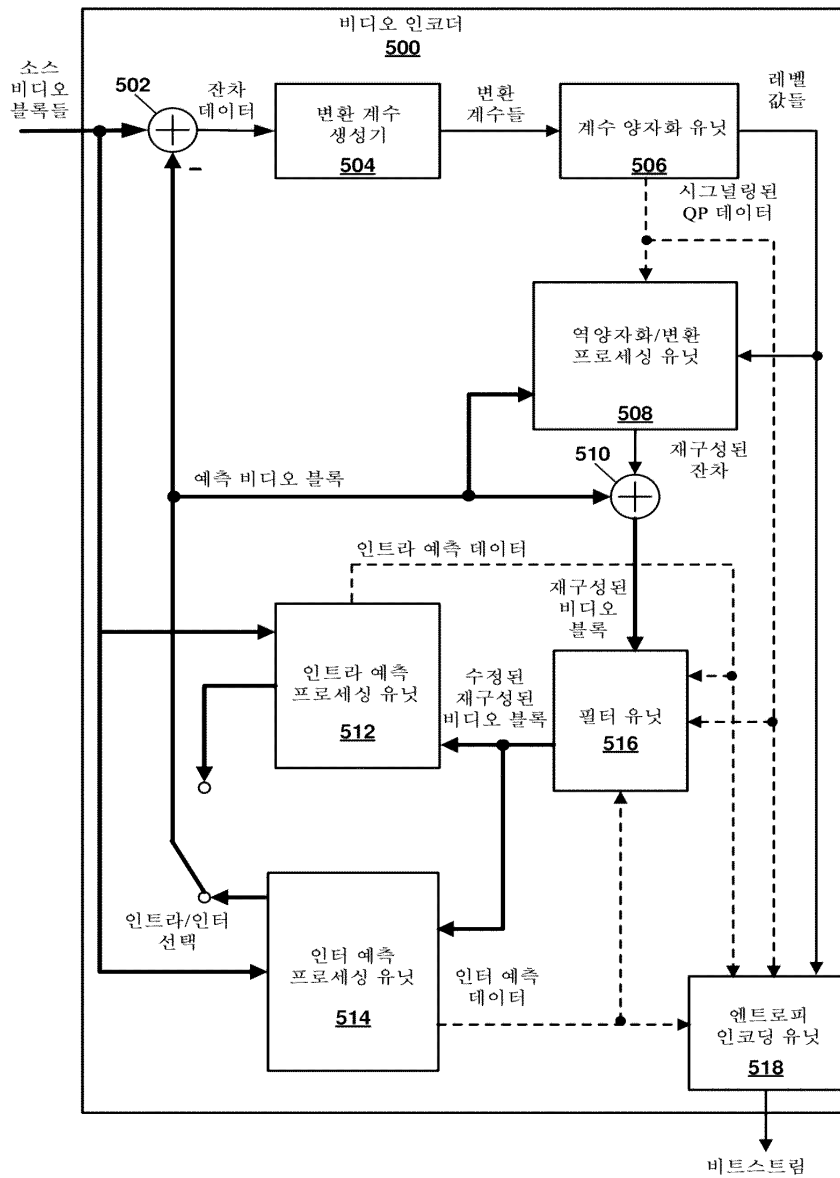
도면3



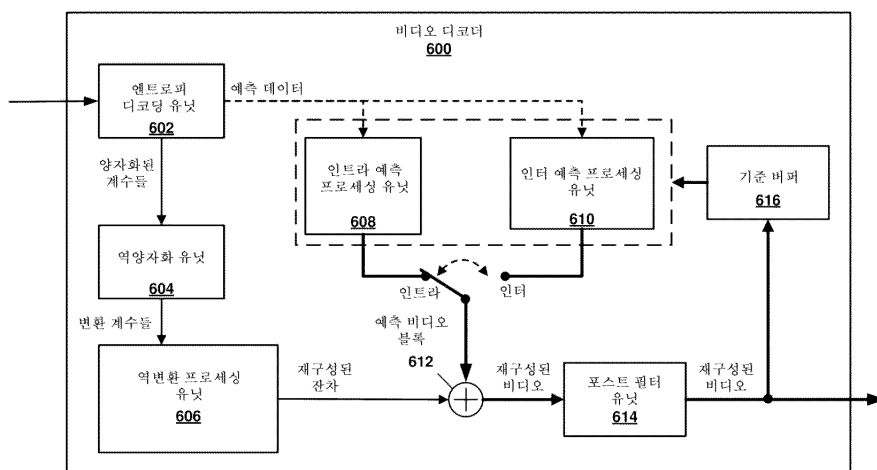
도면4



도면5



도면6



도면7

