(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2023-0165888
(43) 공개일자 2023년12월05일

(51) 국제특허분류(Int. Cl.)
H04N 19/577 (2014.01) *H04N 19/105* (2014.01)
H04N 19/132 (2014.01) *H04N 19/523* (2014.01)
H04N 19/82 (2014.01)
(52) CPC특허분류
H04N 19/577 (2015.01)
H04N 19/105 (2015.01)
(21) 출원번호 10-2023-7041404(분할)
(22) 출원일자(국제) 2020년04월02일
심사청구일자 없음
(62) 원출원 특허 10-2021-7030716
원출원일자(국제) 2020년04월02일
심사청구일자 2023년03월10일
(85) 번역문제출일자 2023년11월30일
(86) 국제출원번호 PCT/CN2020/082941
(87) 국제공개번호 WO 2020/200270
국제공개일자 2020년10월08일
(30) 우선권주장
PCT/CN2019/080914 2019년04월02일 중국(CN)

(71) 출원인
베이징 바이트댄스 네트워크 테크놀로지 컴퍼니, 리미티드
중국, 베이징 100041, 스징산 디스트릭트, 스싱 로드, 넘버 30, 넘버 3빌딩, 2층, 룸 비-0035
바이트댄스 아이엔씨
미국, 90066 캘리포니아 로스엔젤레스 스위트룸 137호 6층 제퍼슨 블러바드 웨스트 12655
(72) 발명자
리우 홍빈
진르터우탸오 포스트 오피스 차이나 세틀라이트 커뮤니케이션 타워 63 지춘 로드 하이디안 디스트릭트 베이징 100080 중국
장 리
미국 90066 캘리포니아 로스엔젤레스 스위트룸 137호 6층 제퍼슨 블러바드 웨스트 12655
(뒷면에 계속)
(74) 대리인
성병기

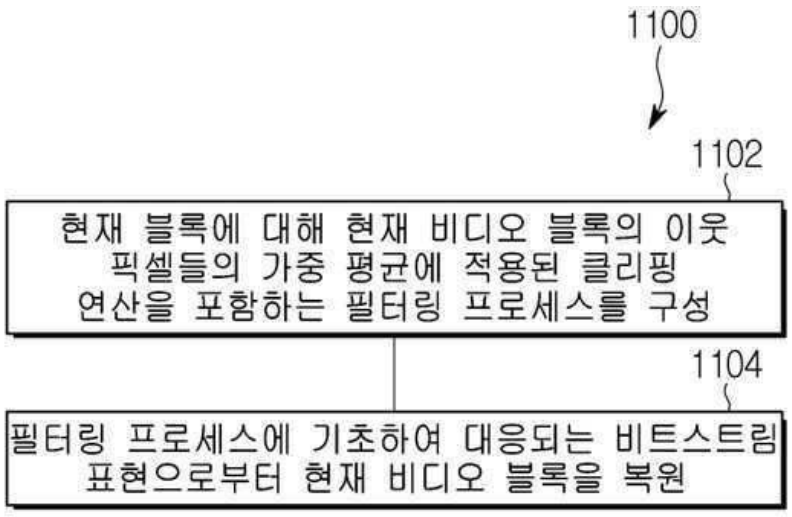
전체 청구항 수 : 총 12 항

(54) 발명의 명칭 **양방향 광학 흐름 기반 비디오 코딩 및 디코딩**

(57) 요 약

비디오 코딩을 위한 샘플 개선 및 필터링 장치, 시스템 및 방법이 설명된다. 예시적인 양태에서, 비디오 프로세싱 방법은 비디오의 블록 및 비디오의 비트스트림 표현 간의 변환을 위해 클리핑 연산을 개선 값에 적용함으로써 블록의 예측 샘플에 대한 개선 값을 수정하는 단계를 포함한다. 개선 값은 광학 흐름 코딩 프로세스의 그라디언트 값에 기초하여 유도된다. 클리핑 연산의 출력은 범위 내이다. 이 방법은 또한 개선 값에 기초하여 예측 샘플을 개선하는 단계 및 개선된 예측 샘플에 기초하여 변환을 수행하는 단계를 포함한다.

대 표 도 - 도11a



(52) CPC특허분류
H04N 19/132 (2015.01)
H04N 19/523 (2015.01)
H04N 19/82 (2015.01)

(72) 발명자
장 카이
미국 90066 캘리포니아 로스엔젤레스 스위트룸 137호 6층 제퍼슨 블러바드 웨스트 12655
수 지장
미국 90066 캘리포니아 로스엔젤레스 스위트룸 137호 6층 제퍼슨 블러바드 웨스트 12655
왕 유에
진르터우탸오 포스트 오피스 차이나 세틀라이트 커뮤니케이션 타워 63 지춘 로드 하이디안 디스트릭트 베이징 100080 중국

명 세 서

청구범위

청구항 1

비디오 데이터 프로세싱 방법에 있어서,

비디오의 제1 블록에 대해 제1 예측 샘플을 결정하는 단계;

제1 광학 흐름 개선 과정에 기초하여 제1 최종 예측 샘플을 획득하기 위해 제1 예측 샘플 오프셋을 상기 제1 예측 샘플에 더하는, 제1 예측 샘플 오프셋을 이용한 상기 제1 예측 샘플을 개선하는 단계, 여기서 상기 제1 예측 샘플 오프셋은 상기 제1 예측 샘플의 적어도 하나의 제1 그라디언트에 기초하여 결정되고, 상기 제1 그라디언트는 상기 제1 예측 샘플의 위치에 기초하여 획득된 두 개의 예측 샘플들 간의 차이에 적어도 기초하여 계산되고, 상기 제1 예측 샘플 오프셋을 상기 제1 예측 샘플에 더하기 전에 상기 제1 예측 샘플 오프셋은 제1 범위로 클리핑 됨; 및

상기 제1 최종 예측 샘플에 기초하여 상기 비디오의 상기 제1 블록과 비트스트림 간의 변환을 수행하는 단계;를 포함하되,

상기 제1 범위는 상기 제1 블록의 입력 비트 깊이에 기초하여 결정되고, 상기 제1 범위는 [min, max] 이되, min은 0보다 작고 max 는 0 보다 큰, 비디오 데이터 프로세싱 방법.

청구항 2

제1 항에 있어서, 상기 방법은,

상기 비디오의 제2 블록에 대해 제2 예측 샘플을 결정하는 단계;

제2 광학 흐름 개선 과정에 기초하여 제2 최종 예측 샘플을 획득하기 위해 제2 예측 샘플 오프셋을 이용하여 상기 제2 예측 샘플을 개선하는 단계, 여기서 상기 제2 예측 샘플 오프셋은 상기 제2 예측 샘플의 적어도 하나의 제2 그라디언트에 기초하여 결정되고, 상기 제2 그라디언트는 상기 제2 예측 샘플의 위치에 기초하여 획득된 두 개의 예측 샘플들 간의 차이에 기초하여 계산됨;

상기 제2 최종 예측 샘플을 제2 범위로 클리핑하는 단계; 및

상기 제2 최종 예측 샘플에 기초하여 상기 제2 블록과 상기 비트스트림 간의 변환을 수행하는 단계;를 더 포함하는, 비디오 데이터 프로세싱 방법.

청구항 3

제2 항에 있어서,

상기 제1 비디오 블록은 어파인 모드로 코딩되고, 상기 제2 비디오 블록은 논-어파인(non-affine) 모드로 코딩되는, 비디오 데이터 프로세싱 방법.

청구항 4

제2 항에 있어서,

상기 제2 범위는 상기 제2 비디오 블록의 입력 비트 깊이에 기반하여 결정되는, 비디오 데이터 프로세싱 방법.

청구항 5

제2 항에 있어서,

상기 제2 범위는 [min, max] 이되, min 은 0보다 작고 max 는 0 보다 큰, 비디오 데이터 프로세싱 방법.

청구항 6

제2 항에 있어서,

상기 제2 범위는 기 정의되는, 비디오 데이터 프로세싱 방법.

청구항 7

제2 항에 있어서,

상기 제1 예측 샘플의 위치에 기초하여 획득된 상기 두 개의 예측 샘플은 동일한 참조 픽처 리스트에서 유도되고, 상기 제2 예측 샘플의 위치에 기초하여 획득된 상기 두 개의 예측 샘플은 서로 다른 참조 픽처 리스트에서 유도되는, 비디오 데이터 프로세싱 방법.

청구항 8

제1 항에 있어서,

상기 변환을 수행하는 단계는 상기 비트스트림으로부터 상기 제1 블록을 디코딩하는 것을 포함하는, 비디오 데이터 프로세싱 방법.

청구항 9

제1 항에 있어서,

상기 변환을 수행하는 단계는 상기 제1 블록을 상기 비트스트림으로 인코딩 하는 것을 포함하는, 비디오 데이터 프로세싱 방법.

청구항 10

명령이 저장된 비 일시적 메모리와 프로세서를 포함하는 비디오 데이터 프로세싱 장치에 있어서, 상기 명령은 상기 프로세서에 의해 실행되되, 상기 프로세서가,

비디오의 제1 블록에 대해 제1 예측 샘플을 결정하게 하고;

제1 광학 흐름 개선 과정에 기초하여 제1 최종 예측 샘플을 획득하기 위해 제1 예측 샘플 오프셋을 상기 제1 예측 샘플에 더하는, 제1 예측 샘플 오프셋을 이용한 상기 제1 예측 샘플을 개선하게 하되, 여기서 상기 제1 예측 샘플 오프셋은 상기 제1 예측 샘플의 적어도 하나의 제1 그라디언트에 기초하여 결정되고, 상기 제1 그라디언트는 상기 제1 예측 샘플의 위치에 기초하여 획득된 두 개의 예측 샘플들 간의 차이에 적어도 기초하여 계산되고, 상기 제1 예측 샘플 오프셋을 상기 제1 예측 샘플에 더하기 전에 상기 제1 예측 샘플 오프셋은 제1 범위로 클리핑 됨; 및

상기 제1 최종 예측 샘플에 기초하여 상기 비디오의 상기 제1 블록과 비트스트림 간의 변환을 수행하게 하되,

상기 제1 범위는 상기 제1 블록의 입력 비트 깊이에 기초하여 결정되고, 상기 제1 범위는 [min, max] 이되, min 은 0보다 작고 max 는 0 보다 큰, 비디오 데이터 프로세싱 장치.

청구항 11

명령을 저장하는 비 일시적 컴퓨터 판독 가능한 저장 매체에 있어서, 상기 명령은 프로세서가,

비디오의 제1 블록에 대해 제1 예측 샘플을 결정하게 하고;

제1 광학 흐름 개선 과정에 기초하여 제1 최종 예측 샘플을 획득하기 위해 제1 예측 샘플 오프셋을 상기 제1 예측 샘플에 더하는, 제1 예측 샘플 오프셋을 이용한 상기 제1 예측 샘플을 개선하게 하되, 여기서 상기 제1 예측 샘플 오프셋은 상기 제1 예측 샘플의 적어도 하나의 제1 그라디언트에 기초하여 결정되고, 상기 제1 그라디언트는 상기 제1 예측 샘플의 위치에 기초하여 획득된 두 개의 예측 샘플들 간의 차이에 적어도 기초하여 계산되고, 상기 제1 예측 샘플 오프셋을 상기 제1 예측 샘플에 더하기 전에 상기 제1 예측 샘플 오프셋은 제1 범위로 클리핑 됨; 및

상기 제1 최종 예측 샘플에 기초하여 상기 비디오의 상기 제1 블록과 비트스트림 간의 변환을 수행하게 하되,

상기 제1 범위는 상기 제1 블록의 입력 비트 깊이에 기초하여 결정되고, 상기 제1 범위는 [min, max] 이되, min은 0보다 작고 max 는 0 보다 큰, 비 일시적 컴퓨터 판독 가능한 저장 매체.

청구항 12

비디오 프로세싱 장치에 의해 수행되는 방법에 의해 생성된 비디오의 비트스트림을 저장하는 비 일시적 컴퓨터 판독 가능한 기록 매체에 있어서, 상기 방법은,

비디오의 제1 블록에 대해 제1 예측 샘플을 결정하는 단계;

제1 광학 흐름 개선 과정에 기초하여 제1 최종 예측 샘플을 획득하기 위해 제1 예측 샘플 오프셋을 상기 제1 예측 샘플에 더하는, 제1 예측 샘플 오프셋을 이용한 상기 제1 예측 샘플을 개선하는 단계, 여기서 상기 제1 예측 샘플 오프셋은 상기 제1 예측 샘플의 적어도 하나의 제1 그라디언트에 기초하여 결정되고, 상기 제1 그라디언트는 상기 제1 예측 샘플의 위치에 기초하여 획득된 두 개의 예측 샘플들 간의 차이에 적어도 기초하여 계산되고, 상기 제1 예측 샘플 오프셋을 상기 제1 예측 샘플에 더하기 전에 상기 제1 예측 샘플 오프셋은 제1 범위로 클리핑 됨; 및

상기 제1 최종 예측 샘플에 기초하여 상기 비트스트림을 생성하는 단계;를 포함하되,

상기 제1 범위는 상기 제1 블록의 입력 비트 깊이에 기초하여 결정되고, 상기 제1 범위는 [min, max] 이되, min은 0보다 작고 max 는 0 보다 큰, 비 일시적 컴퓨터 판독 가능한 기록 매체.

발명의 설명

기 술 분 야

[0001] 본 출원은 2019년 4월 2일에 제출된 국제특허 출원 No. PCT/CN2019/080914의 우선권 혜택을 주장하는, 2020년 4월 2일에 제출된 국제특허 출원 No. PCT/CN2020/082941의 국내단계 한국 출원 No. 10-2021-7030716의 분할 출원이다. 상술한 모든 특허 출원은 그 전체가 참고로서 여기에 포함된다.

[0002] 본 특허 문서는 비디오 코딩 기술들, 장치들 및 시스템들에 관한 것이다.

배 경 기 술

[0003] 비디오 압축의 발전에도 불구하고, 디지털 비디오는 여전히 인터넷 및 기타 디지털 통신 네트워크에서 가장 큰 대역폭의 사용을 차지하고 있다. 비디오를 수신하고 디스플레이할 수 있는 연결된 사용자 장치의 수가 증가함에 따라, 디지털 비디오 사용에 대한 대역폭의 수요는 계속해서 증가할 것으로 예상된다.

발명의 내용

해결하려는 과제

과제의 해결 수단

[0004] 디지털 비디오 코딩, 특히 비디오 코딩을 위한 샘플 정제 및 필터링 방법과 관련된 장치, 시스템 및 방법이 설명된다. 설명된 방법은 HEVC(High Efficiency Video Coding)와 같은 기존의 비디오 코딩 표준과 VVC(Versatile Video Coding) 또는 코덱과 같은 미래의 비디오 코딩 표준 모두에 적용될 수 있다.

[0005] 일 대표 양상에서, 개시된 기술은 비디오 프로세싱 방법을 제공하는 데 사용될 수 있다. 이 방법은 비디오의 현재 블록과 비디오의 비트스트림 표현 간의 변환을 위해, 현재 블록의 픽셀과 픽셀의 인접 픽셀들 간의 차이의 가중 평균에 클리핑 연산을 적용하는 단계를 포함한다. 이 방법은 또한 클리핑 연산의 출력에 기초하여 변환을 수행하는 단계를 포함한다.

[0006] 다른 대표적인 양상에서, 개시된 기술은 비디오 프로세싱 방법을 제공하는 데 사용될 수 있다. 이 방법은, 비디오의 현재 블록과 비디오의 비트스트림 표현 간의 변환을 위해, 현재 블록의 인접 픽셀들의 세트에 기초하여 필터링 프로세스를 구성하는 단계를 포함한다. 이 방법은 또한 필터링 프로세스의 출력에 기초하여 변환을 수행하는 단계를 포함한다.

[0007] 다른 대표적인 양상에서, 개시된 기술은 비디오 프로세싱 방법을 제공하는 데 사용될 수 있다. 이 방법은 비디오의 블록과 비디오의 비트스트림 표현 간의 변환을 위해 개선 값에 클리핑 연산을 적용함으로써 블록에서 예측 샘플에 대한 개선 값을 수정하는 단계를 포함한다. 개선 값은 광학 흐름 코딩 프로세스의 그라디언트 값에 기초하여 유도된다. 클리핑 연산의 출력은 범위 내이다. 이 방법은 또한 수정된 개선 값에 기초하여 예측 샘플을 개선하는 단계 및 개선된 예측 샘플에 기초하여 변환을 수행하는 단계를 포함한다.

[0008] 다른 대표적인 양상에서, 개시된 기술은 비디오 프로세싱 방법을 제공하는 데 사용될 수 있다. 이 방법은 비디오의 블록과 비디오의 비트스트림 표현 간의 변환을 위해, 광학 흐름 코딩 프로세스에서 그라디언트 값을 수정하는 단계를 포함한다. 이 방법은 또한 변환을 위해 수정된 그라디언트 값에 기초하여 후속 예측 연산을 수행하는 단계를 포함한다.

[0009] 다른 대표적인 양상에 있어서, 개시된 기술은 비디오 프로세싱 방법을 제공하는 데 사용될 수 있다. 이 방법은, 현재 비디오 블록에 대해, 1차원 필터링 프로세스를 구성하는 단계, 및 1차원 필터링 프로세스에 기초하여 대응 비트스트림 표현으로부터 현재 비디오 블록을 복원하는 단계를 포함한다.

[0010] 또 다른 대표적인 양상에 있어서, 상술한 방법은 프로세서 실행가능 코드의 형태로 구현되어 컴퓨터 판독가능한 프로그램 매체에 저장된다.

[0011] 또 다른 대표적인 양상에 있어서, 상술한 방법을 수행하도록 구성되거나 동작 가능한 장치가 개시된다. 이 장치는 이 방법을 구현하도록 프로그래밍된 프로세서를 포함할 수 있다.

[0012] 또 다른 대표적인 양상에 있어서, 비디오 디코더 장치는 여기에 개시된 방법을 구현할 수 있다.

[0013] 개시된 기술의 상기 및 기타 양태 및 특징들은 도면들, 상세한 설명 및 청구 범위에서 보다 상세하게 설명된다.

도면의 간단한 설명

[0014] 도 1은 비디오 코딩을 위한 인코더 블록도의 예를 도시한 것이다.

도 2a, 2b 및 2c는 GALF(geometry transformation-based adaptive loop filter) 필터 형태들의 예를 도시한 것이다.

도 3은 GALF 인코더 결정을 위한 흐름도의 예를 도시한 것이다.

도 4a-4d는 ALF(adaptive loop filter) 분류를 위한 서브 샘플링된 라플라시안(Laplacian) 계산의 예를 도시한 것이다.

도 5는 루마 필터 형태의 예를 도시한 것이다.

도 6은 WVGA(Wide Video Graphic Array) 시퀀스의 영역 분할의 예를 도시한 것이다.

도 7은 정형(reshaping) 디코딩 흐름의 흐름도의 예를 도시한 것이다.

도 8은 양 방향 광학 흐름(bi-directional optical flow: BDOF 혹은 BIO) 알고리즘에 의해 사용되는 광학 흐름 궤적의 예를 도시한 것이다.

도 9a 및 9b는 블록 확장 없이 양 방향 광학 흐름(BIO) 알고리즘의 사용의 예시적인 스냅샷들을 도시한 것이다.

도 10은 PROF(prediction refinement with optical flow)의 예를 도시한 것이다.

도 11a 및 11b는 개시된 기술에 따라 비디오 코딩에서 샘플 개선 및 필터링을 위한 예시적인 방법들의 흐름도들을 도시한 것이다.

도 12는 본 문서에 개시된 시각 매체 디코딩 혹은 시각 매체 인코딩 기술을 실현하기 위한 하드웨어 플랫폼의 일 예를 블록도로 도시한 것이다.

도 13은 개시된 기술이 구현될 수 있는 예시적인 비디오 프로세싱 시스템의 블록도를 도시한 것이다.

도 14는 본 기술에 따른 비디오 프로세싱 방법의 흐름도 표현을 도시한 것이다.

도 15는 본 기술에 따른 비디오 프로세싱을 위한 다른 방법의 흐름도 표현을 도시한 것이다.

도 16은 본 기술에 따른 비디오 프로세싱을 위한 다른 방법의 흐름도 표현을 도시한 것이다.

도 17은 본 기술에 따른 비디오 프로세싱을 위한 또다른 방법의 흐름도 표현을 도시한 것이다.

발명을 실시하기 위한 구체적인 내용

[0015] 고해상도 비디오에 대한 수요 증가로 인해, 비디오 코딩 방법 및 기술은 현대 기술이 도처에 존재하고 있다. 비디오 코덱은 일반적으로 디지털 비디오를 압축하거나 압축 해제하는 전자 회로 또는 소프트웨어를 포함하며 더 높은 코딩 효율성을 제공하기 위해 지속적으로 개선되고 있다. 비디오 코덱은 압축되지 않은 비디오를 압축된 형식으로 변환하고, 역 변환한다. 비디오 품질, 비디오를 나타내는 데 사용되는 데이터의 양(비트 전송율에 따라 결정됨), 인코딩 및 디코딩 알고리즘의 복잡성, 데이터 손실 및 오류에 대한 민감도, 편집 용이성, 랜덤 액세스 및 종단 간 지연(latency) 간에는 복잡한 관계가 있다. 압축된 포맷은 주로, 예를 들어, 고효율 비디오 코딩(High Efficiency Video Coding: HEVC) 표준 (H.265 혹은 MPEG-H 파트 2로도 알려짐), 완결될 다용도 비디오 코딩(Versatile Video Coding: VVC) 표준 혹은 다른 현재 및/혹은 미래의 비디오 코딩 표준들과 같은 표준 비디오 압축 사양을 따른다.

[0016] 일부 실시예들에서, 미래의 비디오 코딩 기술들은 JEM(Joint Exploration Model)로 알려진 참조 소프트웨어를 사용하여 탐색된다. JEM에서, 서브 블록 기반 예측은 다양한 코딩 도구들, 예를 들어, 어파인 예측, 대체 시간적 움직임 벡터 예측(alternative temporal motion vector prediction: ATMVP), 시공간적 움직임 벡터 예측(spatial-temporal motion vector prediction: STMVP), 양 방향 광학 흐름(bi-directional optical flow: BIO), 프레임 율 상향 변환(Frame-Rate Up Conversion: FRUC), 지역적 적응 움직임 벡터 해상도(Locally Adaptive Motion Vector Resolution: LAMVR), 중첩된 블록 움직임 보상(Overlapped Block Motion Compensation: OBMC), 지역 조명 보상(Local Illumination Compensation: LIC), 및 디코더 측 움직임 벡터 개선(Decoder-side Motion Vector Refinement: DMVR)에서 채택된 바 있다.

[0017] 개시된 기술의 실시예들은 기존 비디오 코딩 표준(예를 들어, HEVC, H.265)과 런타임 성능을 향상하기 위한 미래의 표준들에 적용될 수 있으며 섹션 표제는 설명의 가독성을 향상시키기 위해 본 문서에서 사용되며 논의 또는 실시예(및/또는 구현)를 각각의 섹션으로만 제한하는 것은 아니다.

[0018] **1색상 공간 및 크로마 서브샘플링의 예들**

[0019] 색상 모델(혹은 색상 시스템)으로도 알려진 색상 공간은 색상 범위를 숫자 튜플, 일반적으로 3개 또는 4개 값 또는 색상 구성 요소들(예를 들어, RGB)로 간단히 설명하는 추상적 수학 모델이다. 기본적으로 색상 공간은 좌표계와 하위 공간의 정교화이다.

[0020] 비디오 압축에서, 가장 많이 사용되는 색상 공간들은 YCbCR 및 RGB이다.

[0021] YCBCR 혹은 Y'CBCR로도 지칭되는 YCbCr, Y′ CbCr, 혹은 Y Pb/Cb Pr/Cr은 비디오 및 디지털 사진 시스템들에서 색상 이미지 파이프라인의 일부로 사용되는 색 공간들의 제품군이다. Y'는 루마 구성요소이고 CB 및 CR은 청색차 및 적색차 크로마 구성요소들이다. Y'(프라임 포함)는 휘도인 Y와 구별되며, 이는 광 강도가 감마 보정된 RGB 원색을 기반으로 비선형적으로 인코딩됨을 의미한다.

[0022] 크로마 서브샘플링은 휘도보다 색상 차이에 대한 인간 시각 시스템의 낮은 선명도를 이용하여 루마 정보보다 크

로마 정보에 대해 더 낮은 해상도를 구현함으로써 이미지를 인코딩하는 방식이다.

[0023] **1.14:4:4 색상 형식**

[0024] 3개의 Y'CbCr 구성 요소 각각은 동일한 샘플률을 가지므로 크로마 서브샘플링이 없다. 이 방식은 고급 필름 스캐너와 영화 후반 제작에 가끔 사용된다.

[0025] **1.24:2:2 색상 형식**

[0026] 두 개의 크로마 구성요소들은 루마의 샘플율의 절반으로 샘플링되는데, 예를 들어, 수평 크로마 해상도가 절반으로 줄어든다. 이는 시각적 차이가 거의 또는 전혀 없이 압축되지 않은 비디오 신호의 대역폭을 1/3으로 감소시킨다.

[0027] **1.34:2:0 색상 형식**

[0028] 4:2:0에서, 수평 샘플링은 4:1:1:에 비해 두배이지만, 이 방식에서는 Cb 및 Cr 채널들이 각 교대 라인에서만 샘플링되기 때문에 수직 해상도가 절반으로 줄어든다. 그래서 데이터 율은 동일하다. Cb 및 Cr은 수평으로 및 수직으로 각각 2의 인자로 서브샘플링된다. 4:2:0 방식에는 세 가지 변형이 있으며 수평 및 수직 위치가 서로 상이하다.

[0029] ○ MPEG-2에서, Cb 및 Cr은 수평으로 나란히 배치되어 있다. Cb 및 Cr은 수직 방향으로 픽셀 사이에 배치된다(간격 배치됨).

[0030] ○ JPEG/JFIF, H.261, 및 MPEG-1에서, Cb 및 Cr은 교대 휘도 샘플들 사이의 중간인 틈새에 위치한다.

[0031] ○ 4:2:0 DV에서, Cb 및 Cr은 수평방향으로 공존한다. 수직 방향으로, 교대 라인에 함께 배치된다.

[0032] **2일반적인 비디오 코덱의 코딩 흐름의 예들**

[0033] 도 1은 3개의 인루프 필터링 블록인 디블록킹 필터(DF), 샘플 적응 오프셋(SAO) 및 적응 루프 필터(ALF)를 포함하는 VVC의 인코더 블록 다이어그램의 예를 도시한 것이다. 미리 정의된 필터를 사용하는 DF와 달리 SAO 및 ALF는 오프셋들 및 필터 계수들을 시그널링하는 코딩된 부가 정보에서 각각 오프셋을 추가하고 유한 임펄스 응답(FIR) 필터를 적용하여, 원본 샘플과 복원된 샘플 간의 평균 제곱 오차를 줄이기 위해 현재 픽처의 원본 샘플을 활용한다. ALF는 각 그림의 마지막 처리 단계에 위치하며 이전 단계에서 생성된 아티팩트를 잡아서 수정하는 도구라고 볼 수 있다.

[0034] **3JEM에서 기하 변환 기반 적응적 루프 필터(geometry transformation-based adaptive loop filter)의 예들**

[0035] JEM에서, 블록 기반 필터 적용에 의한 GALF(geometry transformation-based adaptive loop filter)이 적용된다. 루마 구성요소의 경우 지역 그라디언트들의 방향과 활성에 기초하여 각 2x2 블록 마다 25개의 필터들 중 하나가 선택된다.

[0036] **3.1필터 형태의 예들**

[0037] JEM에서, 최대 세 개의 다이아몬드 필터 형태들(도 2A, 2B 및 2C에 각각 도시된 5x5 다이아몬드, 7x7 다이아몬드 및 9x9 다이아몬드와 같이)이 루마 구성요소에 대해 선택될 수 있다. 루마 구성요소에 사용되는 필터 형태를 지시하기 위해 픽처 레벨에서 인덱스가 시그널링된다. 픽처에서 크로마 구성요소들에 대하여는, 5x5 다이아몬트 형태가 사용된다.

[0038] **3.1.1블록 분류**

[0039] 각 2x2 블록은 25 개의 클래스들 중 하나로 분류된다. 분류 인덱스 C 는 방향성 D와 활성 $\hat{A}$의 양자화된 값을 기반으로 하기와 같이 유도된다.

[0040] $$C = 5D + \hat{A}. \qquad (1)$$

[0041] D 및 $\hat{A}$를 계산하기 위해, 수평, 수직 및 두 대각선 방향의 그라디언트 들은 먼저 1-D 라플라시안을 이용하여

계산된다.

$$g_v = \sum_{k=i-2}^{i+3} \sum_{l=j-2}^{j+3} V_{k,l}, \quad V_{k,l} = |2R(k,l) - R(k,l-1) - R(k,l+1)|, \tag{2}$$

$$g_h = \sum_{k=i-2}^{i+3} \sum_{l=j-2}^{j+3} H_{k,l}, \quad H_{k,l} = |2R(k,l) - R(k-1,l) - R(k+1,l)|, \tag{3}$$

$$g_{d1} = \sum_{k=i-2}^{i+3} \sum_{l=j-3}^{j+3} D1_{k,l}, \; D1_{k,l} \tag{4}$$

$$= |2R(k,l) - R(k-1,l-1) - R(k+1,l+1)|$$

$$g_{d2} = \sum_{k=i-2}^{i+3} \sum_{j=j-2}^{j+3} D2_{k,l}, \; D2_{k,l} \tag{5}$$

[0042] $$= |2R(k,l) - R(k-1,l+1) - R(k+1,l-1)|$$

[0043] 인덱스 i 및 j 는 2x2 블록에서 좌상단 샘플의 좌표들을 나타내고, $R(i,j)$ 는 좌표 (i,j)에서 복원된 샘플을 지시한다.

[0044] 그런 다음 수평 및 수직 방향의 그라디언트들의 D 최대값 및 최소값이 다음과 같이 설정된다:

$$g_{h,v}^{max} = max(g_h, g_v), \quad g_{h,v}^{min} = min(g_h, g_v), \tag{6}$$

[0045] 또한 두 대각선 방향의 그라디언트의 최대값 및 최소값은 다음과 같이 설정된다:

$$g_{d0,d1}^{max} = max(g_{d0}, g_{d1}), \quad g_{d0,d1}^{min} = min(g_{d0}, g_{d1}), \tag{7}$$

[0046]

[0047] 방향성 D 의 값을 유도하기 위해, 이러한 값들은 서로 그리고 두 개의 임계값 t_1 및 t_2와 비교된다:

[0048] **단계 1.** $g_{h,v}^{max} \le t_1 \cdot g_{h,v}^{min}$ 및 $g_{d0,d1}^{max} \le t_1 \cdot g_{d0,d1}^{min}$ 둘 모두가 참이면, D는 0으로 설정된다.

[0049] **단계 2.** $g_{h,v}^{max} / g_{h,v}^{min} > g_{d0,d1}^{max} / g_{d0,d1}^{min}$ 이면, 단계 3으로 이어진다. 그 외의 경우, 단계 4로 이어진다.

[0050] **단계 3.** $g_{h,v}^{max} > t_2 \cdot g_{h,v}^{min}$ 이면, D 는 2로 설정된다; 그 외의 경우 D는 1로 설정된다.

[0051] **단계 4.** $g_{d0,d1}^{max} > t_2 \cdot g_{d0,d1}^{min}$ 이면, D 는 4로 설정된다; 그 외의 경우 D는 3으로 설정된다.

[0052] 활성 값 A 는 하기와 같이 계산된다:

$$A = \sum_{k=i-2}^{i+3} \sum_{l=j-2}^{j+3} (V_{k,l} + H_{k,l}). \quad (8)$$

[0053]

[0054] A 는 0 내지 4의 범위 내로 양자화 되고, 양자화 값은 $\hat{A}$ 로 지시된다. 픽처에서 두 크로마 구성요소들에 대해, 분류 방법이 적용되지 않는다. 즉, 단일 세트의 ALF 계수들은 각 크로마 구성요소에 대해 적용된다.

[0055] **3.1.2필터 계수들의 기하 변환들**

[0056] 각 2x2 블록 필터링 전에, 회전 혹은 대각선 혹은 시각적 플리핑과 같은 기하 변환들은 그 블록에 대해 계산된 그라디언트 값들에 따라 필터 계수들 $f(k,l)$ 에 적용된다. 이것은 필터 지지 영역의 샘플들에 이러한 변환을 적용하는 것과 동일하다. 이러한 컨셉은 방향성을 맞춤으로써 ALF가 적용되는 다른 블록들을 더 유사하게 만드는 것이다.

[0057] 대각선, 수직 플립(vertical flip) 및 회전을 포함한 세 개의 기하 변환들이 개시된다:

$$\text{대각선(Diagonal)}: f_D(k,l) = f(l,k),$$

$$\text{수직 플립(Vertical flip)}: f_V(k,l) = f(k,K-l-1), \quad (9)$$

$$\text{회전(Rotation)}: f_R(k,l) = f(K-l-1,k).$$

[0058]

[0059] 여기서, K 는 필터의 크기이고, $0 \le k,l \le K-1$ 은 계수 좌표로서, 위치 $(0,0)$ 가 좌측 상단 모서리에 있고, 위치 $(K-1,K-1)$ 가 우측 하단 모서리에 있다. 변환들은 해당 블록에 대해 계산된 그라디언트 값들에 따라 필터 계수들 $f(k,l)$ 에 적용된다. 변환과 네 방향들의 네 그라디언트들 간의 관계는 표 1에 요약되어 있다.

[0060] **표 1: 한 블록에 대해 계산된 그라디언트와 변환의 매핑**

그라디언트 값	변환
$g_{d2} < g_{d1}$ and $g_h < g_v$	변환 없음(No transformation)
$g_{d2} < g_{d1}$ and $g_v < g_h$	대각선(Diagonal)
$g_{d1} < g_{d2}$ and $g_h < g_v$	수직 플립(Vertical flip)
$g_{d1} < g_{d2}$ and $g_v < g_h$	회전(Rotation)

[0061]

[0062] **3.1.3 필터 파라미터들의 시그널링**

[0063] JEM에서, GALF 필터 파라미터들은 첫 번째 CTU(제1 CTU)에 대해, 예를 들어, 슬라이스 헤더 이후 및 첫 번째 CTU의 SAO 파라미터들 이전에 시그널링된다. 최대 25개 의 루마 필터 계수 세트들이 시그널링될 수 있다. 비트 오버헤드를 감소시키기 위해, 상이한 분류에 대한 필터 계수들이 머지된다. 또한, 참조 픽처들의 GALF 계수들은 저장되고 현재 픽처의 GALF 계수들로서 재사용된다. 현재 픽처는 참조 픽처들에 대해 저장된 GALF 계수들을 사용하고, GALF 계수 시그널링을 우회하도록 선택할 수 있다. 이러한 경우, 참조 픽처들 중 하나에 대한 인덱스만이 시그널링되고, 지시된 참조 픽처의 저장된 GALF 계수들은 현재 픽처에 대해 상속된다.

[0064] GALF 시간적 예측을 지원하기 위해, GALF 필터 세트들의 후보 리스트가 유지된다. 새로운 시퀀스에 대한 디코딩의 시작에서, 후보 리스트는 비어있다. 한 픽처를 디코딩한 이후, 필터들의 대응 세트는 후보 리스트에 추가될 수 있다. 후보 리스트의 크기가 최대 허용된 값(예를 들어, 현재 JEM에서는 6)에 도달하면, 새로운 필터 세트가 디코딩 순서에서 가장 오래된 세트를 덮어쓴다. 즉, 후보 리스트를 업데이트하기 위해 선입선출FIFO (First-in-First-out) 규칙이 적용된다. 반복을 피하기 위해, 대응 픽처가 GALF 시간적 예측을 사용하지 않는 경우에만 세트가 리스트에 추가될 수 있다. 시간적 확장성을 지원하기 위해, 필터 세트들의 복수의 후보 리스트들이 존재하고, 각 후보 리스트는 시간적 레이어에 연관된다. 보다 상세하게는, 시간적 레이어 인덱스(TempIdx)에 의해 할당된 각 배열은 더 낮은 TempIdx와 동일한, 이전에 디코딩된 픽처들의 필터 세트들을 구성할 수 있다. 예를 들어, k번째 배열은 k와 동일한 TempIdx와 연관되도록 할당되고, TempIdx가 k보다 작거나 동일한 픽처들로부터의 필터 세트들만을 포함한다. 특정 픽처의 코딩 이후, 픽처와 연관된 필터 세트들은 동일한 혹은 더 높은 TempIDx와 연관된 배열들을 업데이트하기 위해 사용될 것이다.

[0065] GALF 계수들의 시간적 예측은 시그널링 오버헤드를 최소화하기 위해 인터 코딩된 프레임에 사용된다. 인트라 프레임들의 경우, 시간적 예측이 허용되지 않으며, 고정 필터 16개를 갖는 세트가 각 클래스에 할당된다. 고정 필터의 사용을 지시하기 위해, 각 클래스에 대한 플래그가 시그널링되고, 필요한 경우, 선택된 고정 필터의 인덱스. 주어진 클래스에 대해 고정 필터가 선택되는 경우더라도, 적응적 필터 f(k, l)의 계수들은 여전히 이 클래스에 전송될 수 있으며, 이 경우 복원 픽처에 적용될 필터의 계수들은 두 계수 세트들의 합이다.

[0066] 루마 구성요소의 필터링 프로세스는 CU 레벨에서 제어될 수 있다. 플래그는 GALF가 CU의 루마 구성요소에 적용되는지 여부를 지시하기위해 시그널링된다. 크로마 구성요소의 경우, GALF가 적용되는지 혹은 아닌지 여부는 픽처 레벨에서만 지시된다.

[0067] **3.1.4필터링 프로세스**

[0068] 디코더 측에서, GALF가 블록에 대해 활성화되면, 블록 내의 각 샘플 $R(i,j)$ 이 필터링 되어 하기와 같이 샘플 값

$R'(i,j)$ $f_{m,n}$ $f(k,l)$

이 출력된다. 여기서 L은 필터 길이를, 는 필터 계수를 나타내고, 는 디코딩된 필터 계수들을 나타낸다.

$$R'(i,j) = \sum_{k=-L/2}^{L/2} \sum_{l=-L/2}^{L/2} f(k,l) \times R(i+k, j+l) \quad (10)$$

[0069]

[0070] **3.1.5인코더 측 필터 파라미터들에 대한 결정 프로세스**

[0071] GALF에 대한 전반적인 인코더 결정 프로세스는 도 3에 도해되어 있다. 각 CU의 루마 샘플들의 경우, 인코더는 GALF가 적용되고 적절한 시그널링 플래그가 슬라이스 헤더에 포함되는지 혹은 아닌지 여부에 대한 결정을 내린다. 크로마 샘플들의 경우, 필터를 적용한다는 결정은 CU 레벨 대신 픽처 레벨에 기초하여 내려진다. 또한, 픽처에 대한 크로마 GALF는 루마 GALF가 픽처에 대해 활성화된 때에만 확인된다.

[0072] **4VVC에서 기하 변환 기반 적응적 루프 필터의 예들**

[0073] VVC에서 GALF의 현재 설계는 JEM에서의 설계와 비교했을 때 다음과 같은 주요 변경 사항이 있다:

[0074] (1) 적응적 필터 형태는 제거된다. 오직 7x7 필터 형태만이 루마 구성요소에 대해 허용되고, 5x5 필터 형태는 크로마 구성요소에 대해 허용된다.

[0075] (2) ALF 파라미터들의 시간적 예측 및 고정 필터들로부터의 예측은 모두 제거된다.

[0076] (3) 각 CTU에 대해, 1비트 플래그로 ALF가 활성화되었는지 혹은 비활성화되었는지 여부를 시그널링한다.

[0077] (4) 클래스 인덱스의 계산은 2x2 대신 4x4 레벨에서 수행된다. 또한, 일부 실시예들에서, ALF 분류를 위한 서브 샘플링된 라플라시안 계산 방법이 활용된다. 보다 상세하게는, 한 블록 내의 각 샘플에 대해 수평/수직/45 대각선/ 135도 그라디언트들을 계산할 필요가 없다. 대신, 1:2 서브 샘플링이 활용된다.

[0078] **5AVS2에서 영역 기반 적응적 루프 필터의 예들**

[0079] ALF는 루프 내 필터링의 마지막 단계이다. 이 프로세스에는 두 단계가 있다. 첫 번째 단계는 필터 계수 유도이다. 필터 계수들을 훈련하기 위해, 인코더는 휘도 구성요소의 복원 픽셀들을 16개의 영역으로 분류하고, 필터 계수들의 한 세트는 원본(original) 프레임 및 복원된 프레임 간의 평균 제곱 오차를 최소화하기 위해 위너-호프(wiener-hopf) 방정식을 이용하여 각 카테고리에 대해 필터 계수의 한 세트가 훈련된다. 16 개의 필터 계수 세트 간의 중복성을 감소시키기 위해 인코더는 비율 왜곡 성능에 기초하여 적응적으로 이들을 머지(merge)할 수 있다. 최대 16개의 서로 다른 필터 세트가 휘도 구성요소에 할당될 수 있으며 색차 구성요소에는 하나만이 할당될 수 있다. 두 번째 단계는 필터 결정으로서, 프레임 레벨 및 LCU 레벨을 모두 포함한다. 먼저 인코더는 프레임 레벨 적응적 루프 필터링이 수행되는지 여부를 결정한다. 프레임 레벨 ALF가 온(on)되어 있으면, 인코더는 LCU 레벨 ALF가 수행되는지 여부를 추가로 결정한다.

[0080] **5.1필터 형태**

[0081] 예시적인 필터 형태는 도 5에 도시된 바와 같이 휘도 및 색차 구성요소들 모두에 대해 3x3 정사각형 형태를 중첩하는 7x7 크로스 형태이다. 도 5의 각 정사각형은 샘플에 대응된다. 따라서, 총 17개의 샘플들이 위치 C8의 샘플에 대해 필터링된 값을 유도하는 데 사용된다. 계수 전송의 오버헤드를 고려할 때, 점 대칭 필터는 오직 아홉 개의 계수들 {C0, C1, ..., C8}만이 남겨졌을 때에 활용되므로, 필터 계수의 수와 필터링의 곱셈 수를 절반으로 줄일 수 있다. 점 대칭 필터는 또한 하나의 필터링된 샘플에 대한 계산의 절반을 줄일 수 있다. 예를 들어, 필터링된 샘플 하나에 대한 9개의 곱셈과 14개의 가산 연산을 줄일 수 있다.

[0082] **5.2영역 기반 적응적 머지**

[0083] 다른 코딩 오류를 적용하기 위해, 일부 구현에서는 휘도 구성요소에 대해 영역 기반 다중 적응적 루프 필터들을 채택한다. 휘도 구성요소는 도 6에 도시된 바와 같이 각 기본 영역이 최대 코딩 유닛(LCU) 경계로 정렬되고 각 영역에 대해 하나의 위너(wiener) 필터가 유도되는, 대략적으로 동일한 크기인 기본 영역들 16개로 분할된다. 필터들이 더 많이 사용될 수록, 더 많은 왜곡이 감소되나, 이러한 계수들을 인코딩하는 데 사용되는 비트들이 필터들의 수에 따라 증가한다. 최상의 율 왜곡 성능을 달성하기 위해, 이러한 영역들은 동일한 필터 계수들을 공유하는, 소수의 보다 큰 영역들로 머지될 수 있다. 머지 프로세스를 단순화하기 위해, 각 영역에는 영상 사전

상관 관계에 기초하여 수정된 힐버트 순서에 따라 인덱스가 할당된다. 연이은 인덱스를 가진 두 영역은 율 왜곡 비용에 기초하여 머지될 수 있다.

[0084] 영역 간의 매핑 정보는 디코더에 시그널링되어야 한다. AVS-2에서, 기본 영역들의 수는 머지 결과들을 나타내기 위해 사용되고, 필터 계수들은 그 영역 순서에 따라 순차적으로 압축된다. 예를 들어, {0, 1}, {2, 3, 4}, {5, 6, 7, 8, 9} 및 좌측 기본 영역들이 각각 한 영역으로 머지될 때, 오직 세개의 정수(즉, 2, 3, 5)만이 이러한 머지 맵을 표현하기 위해 코딩 된다.

[0085] **5.3사이드 정보의 시그널링**

[0086] 다중 스위치 플래그들 또한 사용된다. 시퀀스 스위치 플래그인 adaptive_loop_filter_enable는 적응적 루프 필터가 전체 시퀀스에 적용되는지 여부를 제어하는 데 사용된다. 이미지 스위치 플래그인 picture_alf_enable[i]는 ALF가 대응 i번째 이미지 구성요소에 적용되는지 여부를 제어한다. 만약 picture_alf_enable[i]만이 활성화되면, 그 색상 구성요소에 대한 대응 LCU 레벨 플래그 및 필터 계수들이 전송될 것이다. LCU 레벨 플래그인 lcu_alf_enable[k]는 ALF가 대응 k번째 LCU에 대해 활성화되는지 여부를 제어하고, 슬라이스 데이터로 인터리브(interleave)된다. 서로 다른 레벨 규제 플래그의 결정은 모두 율 왜곡 비용에 기초한다. 높은 유연성으로 인해 ALF가 코딩 효율성을 훨씬 더 크게 향상시킬 수 있게 한다.

[0087] 일부 실시예들에서, 루마 구성요소에 대해, 최대 16 개의 필터 계수 세트가 존재할 수 있다.

[0088] 일부 실시예들에서, 각 크로마 구성요소(Cb 및 Cr)에 대해, 한 개의 필터 계수 세트 만이 전송될 수 있다.

[0089] **6예시적인 GALF**

[0090] 일부 실시예들에서, 적응적 루프 필터의 필터링 프로세스는 하기와 같이 수행된다:

[0091] $$O(x,y) = \sum_{(i,j)} w(i,j).I(x+i,y+j) \qquad (11)$$

[0092] 여기서 샘플들 $I(x+i,y+j)$ 은 입력 샘플들이고, $O(x,y)$ 는 필터링된 출력 샘플(즉, 필터 결과)이고, $w(i,j)$ 는 필터 계수를 나타낸다. 일부 실시예들에서, 고정 소수점 정밀도 계산을 위해 정수 산술을 이용하여 구현된다:

[0093] $$O(x,y) = \left(\sum_{i=-\frac{L}{2}}^{\frac{L}{2}} \sum_{j=-\frac{L}{2}}^{\frac{L}{2}} w(i,j).I(x+i,y+j) + 64\right) \gg 7 \qquad (12)$$

[0094] L 은 필터 길이를, $w(i,j)$ 는 고정 소수점 정밀도에서 필터 계수를 나타낸다.

[0095] **7비선형 적응적 루프 필터링(ALF)**

[0096] **7.1필터링 재구성(Filtering reformulation)**

[0097] 방정식 (11)은 코딩 효율성에 대한 영향 없이, 하기 식으로 재구성될 수 있다:

[0098] $$O(x,y) = I(x,y) + \sum_{(i,j)\neq(0,0)} w(i,j).\left(I(x+i,y+j) - I(x,y)\right) \qquad (13)$$

[0099] 여기서, $w(i,j)$ 는 식 (11)에서와 동일한 필터 계수이다 [식 (11)에서는 $1 - \sum_{(i,j)\neq(0,0)} w(i,j)$, 식 (13)에서는 1과 동일한 $w(0,0)$ 는 제외됨].

[0100] **7.2수정된 필터**

[0101] 상기의 필터 공식 (13)을 이용하면, 인접 샘플 값 $(I(x+i,y+j))$ 이 필터링된 현재 샘플 값

$(I(x+i,y+j))$ 과 너무 많이 다른 경우, 인접 샘플 값들의 영향을 줄이기 위해 간단한 클리핑 함수를 사용하여 ALF를 보다 효율적으로 만들기 위해 비선형성이 도입될 수 있다.

[0102] 이러한 제안에서, ALF 필터는 하기와 같이 수정된다:

[0103] $$O'(x,y) = I(x,y) + \sum_{(i,j)\neq(0,0)} w(i,j).K(I(x+i,y+j) - I(x,y), k(i,j)) \quad (14)$$

[0104] 여기서, $K(d,b) = \min(b, \max(-b,d))$ 는 클리핑 함수이고, $k(i,j)$ 는 (i,j) 필터 계수에 따라 달라지는 클리핑 파라미터들이다. 인코더는 최적의 $k(i,j)$ 를 찾기 위해 최적화를 수행한다.

[0105] 일부 실시예들에서, 클리핑 파라미터들 $k(i,j)$ 은 각 ALF 필터들에 대해 특정되고, 하나의 클리핑 값이 필터 계수마다 시그널링된다. 이는 루마 필터당 비트스트림에서 최대 12개의 클리핑 값을, 크로마 필터에 대해 최대 6개의 클리핑 값을 시그널링할 수 있음을 의미한다.

[0106] 시그널링 비용 및 인코더 복잡도를 제한하기 위해, 클리핑 값의 평가는 가능한 값의 작은 세트로 제한될 수 있다. 인터(INTER) 및 인트라(INTRA) 타일 그룹에 대해 동일한 4개의 고정 값들만이 사용될 수 있다.

[0107] 종종, 로컬 차이의 분산이 크로마에 대한 것보다 루마에 대해 더 높기 때문에, 루마 및 크로마 필터들에 대해 서로 상이한 2개의 세트가 사용될 수 있다. 필요하지 않은 경우 클리핑을 비활성화할 수 있도록 각 세트에 최대 샘플 값(여기서는 10비트 비트 깊이의 경우, 1024)이 포함될 수 있다.

[0108] 일부 실시예들에서 사용된 클리핑 값 세트가 표 2에 제공된다. 4개의 값은 로그 영역(logarithmic domain)에서 루마에 대한 샘플 값(10비트로 코딩됨)의 전체 범위를, 크로마의 경우 4에서 1024 사이의 범위를 대략적으로 균등하게 분할하여 선택되었다.

[0109] 보다 상세하게는, 클리핑 값들의 루마 표는 하기의 공식에 의해 획득된다:

[0110] $$\mathrm{AlfClip_L} = \left\{ \mathrm{round}\left(\left((M)^{\frac{1}{N}} \right)^{N-n+1} \right) \text{ for } n \in 1..N \right\}$$, 여기서 $M=2^{10}$ 및 $N=4$.

[0111] 마찬가지로, 클리핑 값들의 크로마 표는 하기의 공식에 따라 획득된다:

[0112] $$\mathrm{AlfClip_C} = \left\{ \mathrm{round}\left(A.\left(\left(\frac{M}{A}\right)^{\frac{1}{N-1}} \right)^{N-n} \right) \text{ for } n \in 1..N \right\}$$, 여기서 $M=2^{10}$, $N=4$ 및 $A=4$.

[0113] **표 2: 승인된 클리핑 값들**

	인트라/인터 타일 그룹
루마	{ 1024, 181, 32, 6 }
크로마	{ 1024, 161, 25, 4 }

[0114]

[0115] 선택된 클리핑 값들은 상기 표 2에서 클리핑 값의 인덱스에 대응되는 골룸(Golomb) 인코딩 방식을 이용하여 신택스 요소 "alf_data"에 코딩된다. 이 코딩 방식은 필터 인덱스에 대한 인코딩 방식과 동일하다.

[0116] **8 예시적인 루프내 정형 (in-loop reshaping: ILR)**

[0117] 루프 내 정형(in-loop reshaping: ILR)의 기본적인 개념은 원본(제1 구역에서) 신호 (예측/복원 신호)를 제2 구

역(정형된 구역)으로 변환하는 것이다.

[0118] 루프 내 루마 정형기는 한 쌍의 룩업 표들(LUTs)로서 구현되나, 다른 하나는 시그널링된 LUT로부터 계산될 수 있으므로, 두 LUT 중 오직 하나만이 시그널링된다. 각 LUT는 1차원, 10-비트, 1024-엔트리 매핑 표(1D-LUT)이다. 한 LUT는 입력 루마 코드 값들 Y_i를 변경된 값들인 Y_r: $Y_r = FwdLUT[Y_i]$에 매핑하는 순방향 LUT, *FwdLUT* 이다. 다른 LUT는 변경된 코드 값들인 Y_r를 $\hat{Y}_i = InvLUT[Y_r]$로 매핑하는 역방향 LUT, *InvLUT* 이다($\hat{Y}_i$는 Y_i의 복원 값을 나타낸다).

[0119] **8.1예시적인 조각별 선형(Piece-wise Linear: PWL) 모델**

[0120] 개념적으로, 조각별 선형(PWL)은 다음과 같은 방식으로 구현된다:

[0121] x1, x2를 두 개의 입력 피벗(pivot) 점들이라고 하고, y1, y2를 한 조각에 대한 대응 출력 피벗 점들이라고 하자. x1 및 x2 사이의 임의의 입력 값 x에 대한 출력 값 y는 다음 방정식으로 보간될 수 있다:

[0122] y = ((y2-y1)/(x2-x1)) * (x-x1) + y1

[0123] 고정 소수점 구현에 있어서, 방정식은 하기와 같이 다시 쓸 수 있다:

[0124] y = ((m * x + 2FP_PREC-1) >> FP_PREC) + c

[0125] m은 스칼라이고, c는 오프셋이고, FP_PREC는 정밀도를 지정하는 상수 값이다.

[0126] 일부 실시예들에서, PWL 모델은 1024-엔트리 FwdLUT 및 InvLUT 매핑 표들을 미리 계산하는 데 사용된다. 그러나 PWL 모델은 구현 시 LUT들을 미리 계산하지 않고도 즉석에서 동일한 매핑 값들을 계산할 수 있도록 한다.

[0127] **8.2예시적인 정형(reshaping)**

[0128] **8.2.1 루마 정형**

[0129] 루프 내 루마 정형의 일부 실시예는 인터 슬라이스 복원에서 블록 단위 인트라 예측을 위한 디코딩 대기 시간도 제거하는 더 낮은 복잡도의 파이프라인을 제공한다. 인트라 예측은 인터 및 인트라 슬라이스 모두에 대해 정형된 구역(domain)에서 수행된다.

[0130] 인트라 예측은 슬라이스 유형에 관계없이 정형된 구역에서 수행된다. 이와 같이 구성되면, 이전의 TU 복원이 완료된 직후 인트라 예측을 곧바로 시작할 수 있다. 이러한 구성은 슬라이스에 종속되는 대신 인트라 모드에 대한 통합된 프로세스를 제공할 수 있다. 도 7은 모드에 기초하여 예시적인 디코딩 프로세스의 블록도를 도시한 것이다.

[0132] *일부 실시예들은 32-조각 PWL 모델들 대신, 루마 및 크로마 잔차에 대한 16-조각 조각 별 선형(PWL) 모델들을 테스트한다.

[0133] 일부 실시예들에서 루프 내 정형기에 의한 인터 슬라이스 복원(밝게 음영 처리된 블록들은 정형된 구역의 신호를 지시한다: 루마 잔차, 인트라 루마 예측됨 및 인트라 루마 복원됨)

[0134] **8.2.2루마 기반 크로마 잔차 스케일링**

[0135] 루마 기반 크로마 잔차 스케일링은 고정 소수점 정수 연산으로 구현되는 곱셈 프로세스이다. 크로마 잔차 스케일링은 크로마 신호와 루마 신호의 상호 작용을 보상한다. 크로마 잔차 스케일링은 TU레벨에서 적용된다. 보다 구체적으로, 다음이 적용된다:

[0136] ○ 인트라의 경우, 복원된 루마가 평균화된다.

[0137] ○ 인터의 경우, 예측 루마가 평균화된다.

[0138] 평균은 PWL 모델의 인덱스를 식별하는 데 사용된다. 인덱스는 스케일링 인자 cScaleInv를 특정한다. 크로마 잔차에 해당 숫자를 곱한다.

[0139] 크로마 스케일링 인자는 복원된 루마 값들 보다는 순방향 매핑되고 예측된 루마 값들로부터 계산된다는 점에 유의하여야 한다.

[0140] 8.2.3ILR 사이드 정보의 시그널링

[0141] 파라미터들은 (현재) 타일 그룹 헤더(ALF와 유사)로 전송된다. 이는 40-100비트들을 필요로 하는 것으로 알려져 있다.

[0142] *다음 사양은 JVET-L1001 버전 9를 기준으로 한다. 추가된 구문은 노란색으로 강조 표시된다.*

[0143] *7.3.2.1 시퀀스 파라미터 세트 RBSP 신택스에서*

seq_parameter_set_rbsp( ) {	Descriptor
sps_seq_parameter_set_id	ue(v)
…	
sps_triangle_enabled_flag	u(1)
sps_ladf_enabled_flag	u(1)
if (sps_ladf_enabled_flag) {	
sps_num_ladf_intervals_minus2	u(2)
sps_ladf_lowest_interval_qp_offset	se(v)
for(i = 0; i < sps_num_ladf_intervals_minus2 + 1; i++) {	
sps_ladf_qp_offset[i]	se(v)
sps_ladf_delta_threshold_minus1[i]	ue(v)
}	
}	
sps_reshaper_enabled_flag	u(1)
rbsp_trailing_bits( )	
}	

[0144]

[0145] *7.3.3.1 일반적인 타일 그룹 헤더 신택스에서*

tile_group_header( ) {	Descriptor
…	
if(num_tiles_in_tile_group_minus1 > 0) {	
offset_len_minus1	ue(v)
for(i = 0; i < num_tiles_in_tile_group_minus1; i++)	
entry_point_offset_minus1[i]	u(v)
}	
if (sps_reshaper_enabled_flag) {	
tile_group_reshaper_model_present_flag	u(1)
if (tile_group_reshaper_model_present_flag)	
tile_group_reshaper_model ( )	
tile_group_reshaper_enable_flag	u(1)
if (tile_group_reshaper_enable_flag && (!(qtbtt_dual_tree_intra_flag && tile_group_type == I)))	
tile_group_reshaper_chr oma_residual_scale_flag	u(1)
}	
byte_alignment( )	
}	

[0146]

[0147] *새로운 신택스 표 타일 그룹 정형기(reshaper) 모델을 추가한다:*

tile_group_reshaper_model () {	Descriptor
reshaper_model_min_bin_idx	ue(v)
reshaper_model_delta_max_bin_idx	ue(v)
reshaper_model_bin_delta_abs_cw_prec_minus1	ue(v)
for (i = reshaper_model_min_bin_idx; i <= reshaper_model_max_bin_idx; i++) {	
reshape_model_bin_delta_abs_CW [i]	u(v)
if (reshaper_model_bin_delta_abs_CW[i]) > 0)	
reshaper_model_bin_delta_sign_CW_flag[i]	u(1)
}	
}	

[0148]

[0149] *일반적인 시퀀스 파라미터 세트 RBSP 시맨틱에서, 하기의 시맨틱(semantics)을 추가한다:*

[0150] **sps_reshaper_**enabled**_flag**이 1이면, 정형기(reshaper)가 코딩된 비디오 시퀀스(coded video sequence: CVS)에서 사용되도록 지정한다. sps_reshaper_enabled_flag가 0이면, 정형기가 CVS에서 사용되지 않음을 지정한다.

[0151] *타일 그룹 헤더 신택스에서, 하기의 시맨틱들이 추가된다:*

[0152] **tile_group_reshaper_model_present_flag**이 1이면 tile_group_reshaper_model()이 타일 그룹 헤더에 존재함을 지정한다. tile_group_reshaper_model_present_flag이 0이면, tile_group_reshaper_model()이 타일 그룹 헤더에 존재하지 않음을 지정한다. tile_group_reshaper_model_present_flag이 존재하지 않으면 0과 동일한 것으로 추론된다.

[0153] **tile_group_reshaper_enabled_flag**이 1이면, 현재 타일 그룹에 대해 정형기(reshaper)가 활성화됨을 지정한다. tile_group_reshaper_enabled_flag 이0이면, 정형기가 현재 타일 그룹에 대해 활성화되지 않음을 지정한다. tile_group_reshaper_enable_flag이 존재하지 않는 경우, 0과 동일한 것으로 추론된다.

[0154] **tile_group_reshaper_chroma_residual_scale_flag**이 1이면 현재 타일 그룹에 대해 크로마 잔차 스케일링이 활성화됨을 지정한다. tile_group_reshaper_chroma_residual_scale_flag이 0이면 크로마 잔차 스케일링이 현재 타일 그룹에 대해 활성화되지 않음을 지정한다. tile_group_reshaper_chroma_residual_scale_flag이 존재하지 않으면, 0과 동일한 것으로 추론된다.

[0155] *tile_group_reshaper_model( ) 신택스 추가*

[0156] **reshape_model_min_bin_idx**는 정형기(reshaper) 구성 프로세스에서 사용할 최소 빈(혹은 조각) 인덱스를 지정한다. reshape_model_min_bin_idx의 값은 0에서 MaxBinIdx까지의 범위 내에 있다. MaxBinIdx의 값은 15와 동일하다.

[0157] **reshape_model_delta_max_bin_idx** 는 최대 허용 빈(혹은 조각) 인덱스 MaxBinIdx 에서 정형기(reshaper) 구성 프로세스에 사용할 최대 빈 인덱스를 뺀 값으로 지정한다. reshape_model_max_bin_idx의 값은 MaxBinIdx - reshape_model_delta_max_bin_idx와 동일하게 설정된다.

[0158] **reshaper_model_bin_delta_abs_cw_prec_minus1** +1은 신택스 reshape_model_bin_delta_abs_CW[i]의 표현에 사용되는 비트 수를 지정한다.

[0159] **reshape_model_bin_delta_abs_CW**[i] 는 i번째 빈에 대한 절대 델타 코드워드 값을 지정한다.

[0160] **reshaper_model_bin_delta_sign_CW_flag**[i]는 다음과 같이 reshape_model_bin_delta_abs_CW[i]의 부호를 지정한다:

[0161] - reshape_model_bin_delta_sign_CW_flag[i]이 0과 동일한 경우, 대응 변수 RspDeltaCW[i] 은 양수이다.

[0162] - 그 외의 경우 (reshape_model_bin_delta_sign_CW_flag[i]가 0과 동일하지 않은 경우), 대응 변수 RspDeltaCW[i] 은 음수이다.

[0163] reshape_model_bin_delta_sign_CW_flag[i] 이 존재하지 않으면, 0과 같은 것으로 추론된다.

[0164] 변수 RspDeltaCW[i] = (1 - 2*reshape_model_bin_delta_sign_CW [i]) *

reshape_model_bin_delta_abs_CW [i];

[0165] 변수 RspCW[i] 는 하기의 단계를 따라 유도된다:

[0166] 변수 OrgCW 는 (1 << $BitDepth_Y$) / (MaxBinIdx + 1)와 동일하게 설정된다.

[0167] - reshaper_model_min_bin_idx < = i <= reshaper_model_max_bin_idx 이면

[0168] RspCW[i] = OrgCW + RspDeltaCW[i].

[0169] - 그 외의 경우 , RspCW[i] = 0.

[0170] $BitDepth_Y$ 의 값이 10과 동일하면, RspCW [i]의 값이 32 부터 2 * OrgCW - 1 까지의 범위 내일 수 있다.

[0171] i가 0부터 MaxBinIdx + 1인 변수들 InputPivot[i] 은 하기와 같이 유도된다:

[0172] InputPivot[i] = i * OrgCW

[0173] i가 0부터 MaxBinIdx+1인 변수 ReshapePivot[i], i가 0부터 MaxBinIdx인 변수 ScaleCoef[i] 및 InvScaleCoeff[i] 은 하기와 같이 유도된다:

[0174] shiftY = 14

[0175] ReshapePivot[0] = 0;

[0176] for(i = 0; i <= MaxBinIdx ; i++) {

[0177] ReshapePivot[i + 1] = ReshapePivot[i] + RspCW[i]

[0179] * ScaleCoef[i] = (RspCW[i] * (1 << shiftY) + (1 << (Log2(OrgCW) - 1))) >> (Log2(OrgCW))

[0180] if (RspCW[i] == 0)

[0181] InvScaleCoeff[i] = 0

[0182] else

[0183] InvScaleCoeff[i] = OrgCW * (1 << shiftY) / RspCW[i]

[0184] }

[0185] i가 0에서 MaxBinIdx인 변수 ChromaScaleCoef[i]는 다음과 같이 유도된다:

[0186] ChromaResidualScaleLut[64] = {16384, 16384, 16384, 16384, 16384, 16384, 16384, 8192, 8192, 8192, 8192, 5461, 5461, 5461, 5461, 4096, 4096, 4096, 4096, 3277, 3277, 3277, 3277, 2731, 2731, 2731, 2731, 2341, 2341, 2341, 2048, 2048, 2048, 1820, 1820, 1820, 1638, 1638, 1638, 1638, 1489, 1489, 1489, 1489, 1365, 1365, 1365, 1365, 1260, 1260, 1260, 1260, 1170, 1170, 1170, 1170, 1092, 1092, 1092, 1092, 1024, 1024, 1024, 1024};

[0187] shiftC = 11

[0188] - if (RspCW[i] == 0)

[0189] ChromaScaleCoef [i] = (1 << shiftC)

[0190] - Otherwise (RspCW[i] != 0), ChromaScaleCoef[i] = ChromaResidualScaleLut[RspCW[i] >> 1]

[0191] **8.2.4ILR의 사용**

[0192] 인코더 측에서, 각 픽처(혹은 타일 그룹)은 먼저 정형된 구역으로 변환된다. 그리고 모든 코딩 프로세스들이 정형된 구역에서 수행된다. 인트라 예측의 경우, 인접 블록은 정형된 구역 내이다. 인터 예측의 경우, 참조 블록들(디코딩된 픽처 버퍼에서 원본 구역으로부터 생성됨)은 정형된 구역으로 먼저 변환된다. 그런 다음 잔차가 생

성되고 비트스트림으로 코딩된다.

[0193] 전체 픽처(혹은 타일 그룹)이 인코딩/디코딩을 마치고 나면, 정형된 구역 내의 샘플들은 원본(original) 구역으로 변환되고, 디블록킹 필터 및 다른 필터들이 적용된다.

[0195] *예측 신호에 대한 순방향 정형은 하기의 경우들에 대해 비활성화된다:

[0196] ○ 현재 블록은 인트라 코딩된다

[0197] ○ 현재 블록은 CPR로 코딩된다(현재 픽처 참조(current picture referencing), 인트라 블록 복사(intra block copy: IBC)로도 알려짐)

[0198] ○ 현재 블록은 결합된 인터-인트라 모드(combined inter-intra mode: CIIP)로 코딩되고 순방향 정형은 인트라 예측 블록에 대해 비활성화 된다

[0199] **9양방향 광학 흐름(Bi-directional optical flow: BIO 혹은 BDOF)**

[0200] **9.1BIO 개요 및 분석**

[0201] BDOF(BIO로도 알려짐)에서, 움직임 보상이 현재 블록의 첫 번째 예측들(각 예측 방향에서)을 생성하기 위해 먼저 수행된다. 첫 번째 예측들은 공간적 그라디언트, 시간적 그라디언트 및 블록 내 각 서브 블록/픽셀의 광학 흐름을 유도하는 데 사용되며, 이는 두 번째 예측, 즉 서브 블록/픽셀의 최종 예측을 생성하기 위해 사용된다. 자세한 내용은 하기에서 설명된다.

[0202] 양 방향 광학 흐름(bi-directional optical flow: BIO)은 양 예측을 위해 블록 별 움직임 보상 위에 수행되는 샘플 별 움직임 개선이다. 일부 실시예들에서, 샘플 레벨 움직임 개선은 시그널링을 사용하지 않는다.

[0203] $I^{(k)}$을 블록 움직임 보상 후 레퍼런스 k (k=0, 1)로부터의 루마 값이라 하고, $\partial I^{(k)}/\partial x$ 및 $\partial I^{(k)}/\partial y$ 을 $I^{(k)}$ 그라디언트(gradient)의 각각 수평 및 수직 구성요소들이라 하자. 광학 흐름이 유효하다고 가정하면, 움직임 벡터 필드 (v_x, v_y) 는 아래 식에 의해 주어진다.

[0204] $$\partial I^{(k)}/\partial t + v_x \partial I^{(k)}/\partial x + v_y \partial I^{(k)}/\partial y = 0. \quad \text{식. (15)}$$

[0205] 각 샘플의 운동 궤적에 대해 이러한 광학 흐름 방정식과 헤르미트(Hermite) 보간법을 결합하면, 끝에서 도함수들 $\partial I^{(k)}/\partial x$과 $\partial I^{(k)}/\partial y$ 및 함수 값들 $I^{(k)}$ 모두를 매칭시키는 고유한 3차 다항식이 획득된다. t=0에서 이 다항식의 값은 BIO 예측이다.

[0206] $$pred_{BIO} = 1/2 \cdot (I^{(0)} + I^{(1)} + v_x/2 \cdot (\tau_1 \partial I^{(1)}/\partial x - \tau_0 \partial I^{(0)}/\partial x) + v_y/2 \cdot (\tau_1 \partial I^{(1)}/\partial y - \tau_0 \partial I^{(0)}/\partial y)). \quad \text{식. (16)}$$

[0207] 도 8은 BIO(Bi-directional Optical Flow) 방법에서 예시적인 광학 흐름 궤적을 도시한 것이다. 여기서, τ_0 및 τ_1 은 참조 프레임까지의 거리들을 나타낸다. 거리 τ_0 및 τ_1 는 Ref_0 및 Ref_1에 기반하여 계산된다. 즉, t_0=POC(현재) POC(Ref_0), t_1= POC(Ref_1) POC(현재). 만약 두 예측 모두 같은 시간 방향에서(예를 들어, 둘 다 과거로부터 또는 미래로부터) 획득된다면, 부호들은 다르다(예를 들어, e.g., $\tau_0 \cdot \tau_1 < 0$). 이때, 예측이 동일한 순간으로부터 오는 것이 아니라면(예를 들어, e.g., $\tau_0 \neq \tau_1$), BIO가 적용된다. 참조된 양 영역 모두 논-제로 움직임을 가지며(예를 들어, $MVx_0, MVy_0, MVx_1, MVy_1 \neq 0$), 블록 움직임 벡터들은 시간 거리에 비례한다

(예를 들어, $MVx_0/MVx_1 = MVy_0/MVy_1 = -\tau_0/\tau_1$).

[0208] 움직임 벡터 필드(v_x, v_y)는 점 A 및 점 B에서의 값들 사이의 차이 Δ를 최소화함으로써 결정된다. 도 8은 움직임 궤적과 참조 프레임 평면들의 교차점의 일 예를 도시한 것이다. 모델은 Δ에 대해 로컬 테일러 전개식의 제1 선형항만을 사용한다:

$$\Delta = \left(I^{(0)} - I^{(1)}_0 + v_x\left(\tau_1 \partial I^{(1)}/\partial x + \tau_0 \partial I^{(0)}/\partial x\right) + v_y\left(\tau_1 \partial I^{(1)}/\partial y + \tau_0 \partial I^{(0)}/\partial y\right)\right) \quad \text{식. (17)}$$

[0209]

[0210] 상기 식에서의 모든 값들은 (i', j')로 표시된 샘플 위치에 의존한다. 움직임이 로컬 주변 환경에서 일관적이라고 가정하면, 현재 예측된 점 (i, j)에 중심이 맞춰진 (2M+1)x(2M+1) 정사각형 윈도우 Ω 내의 Δ는 최소화될 수도 있다. 여기서 M은 2와 같다:

$$(v_x, v_y) = \underset{v_x, v_y}{\operatorname{argmin}} \sum_{[i', j] \in \Omega} \Delta^2[i', j'] \quad \text{식. (18)}$$

[0211]

[0212] 이러한 최적화 문제에 대해, JEM은 먼저 수직 방향에서 최소화한 다음 수평 방향에서 최소화를 행하는 단순화된 접근법을 사용한다. 그 결과는 아래와 같다:

$$v_x = (s_1 + r) > m ? clip3\left(-thBIO, thBIO, -\frac{s_3}{(s_1 + r)}\right) : 0 \quad \text{식. (19)}$$

$$v_y = (s_5 + r) > m ? clip3\left(-thBIO, thBIO, -\frac{s_6 - v_x s_2/2}{(s_5 + r)}\right) : 0 \quad \text{식. (20)}$$

여기서,

$$\begin{aligned} s_1 &= \sum_{[i', j] \in \Omega} \left(\tau_1 \partial I^{(1)}/\partial x + \tau_0 \partial I^{(0)}/\partial x\right)^2; s_3 = \sum_{[i', j] \in \Omega} \left(I^{(1)} - I^{(0)}\right)\left(\tau_1 \partial I^{(1)}/\partial x + \tau_0 \partial I^{(0)}/\partial x\right) \\ s_2 &= \sum_{[i', j] \in \Omega} \left(\tau_1 \partial I^{(1)}/\partial x + \tau_0 \partial I^{(0)}/\partial x\right)\left(\tau_1 \partial I^{(1)}/\partial y + \tau_0 \partial I^{(0)}/\partial y\right) \\ s_5 &= \sum_{[i', j] \in \Omega} \left(\tau_1 \partial I^{(1)}/\partial y + \tau_0 \partial I^{(0)}/\partial y\right)^2; s_6 = \sum_{[i', j] \in \Omega} \left(I^{(1)} - I^{(0)}\right)\left(\tau_1 \partial I^{(1)}/\partial y + \tau_0 \partial I^{(0)}/\partial y\right) \end{aligned} \quad \text{식. (21)}$$

[0213]

[0214] 제로 혹은 매우 작은 값에 의한 나눗셈을 피하기 위해, 정규화(regularization) 파라미터들 r 및 m이 식 19 및 20에 도입될 수 있다. 여기서:

$$r = 500 \cdot 4^{d-8} \quad \text{식. (22)}$$

$$m = 700 \cdot 4^{d-8} \quad \text{식. (23)}$$

[0215]

[0216] d는 비디오 샘플들의 비트 깊이이다 .

[0217] BIO에 대한 메모리 액세스를 정규 양 예측 움직임 보상에 대한 것과 같게 하기 위해, 모든 예측 및 그라디언트 값들인 $I^{(k)}, \partial I^{(k)}/\partial x, \partial I^{(k)}/\partial y$ 은 현재 블록 내부의 위치들에 대해 계산된다. 도 9a는 블록(900) 외부의 액세스 위치들의 일 예를 도시한 것이다. 도 9a에 도시된 것과 같이, 식 17에서, 예측된 블록의 경계 상의 현재 예측된 점에 중심이 맞춰진 (2M+1)x(2M+1) 정사각형 윈도우 Ω는 상기 블록 외부의 위치들에 접근할 필요가 있다.

$I^{(k)}, \partial I^{(k)}/\partial x, \partial I^{(k)}/\partial y$

JEM에서 상기 블록 외부의 의 값들은 상기 블록 내부의 가장 가까운 가용 값과 같도록 설정된다. 예를 들어, 도 9b에 도시된 바와 같이, 이것은 패딩 영역(901)으로 구현될 수 있다.

[0218] BIO를 사용하면, 움직임 필드가 각 샘플에 대해 개선될 수 있다. 계산 복잡도를 줄이기 위해, BIO의 블록 기반 디자인이 JEM에서 사용된다. 움직임 개선은 4x4 블록에 기반하여 계산될 수 있다. 블록 기반 BIO에서, 4x4 블록 상의 모든 샘플들의 식 (21)에 따른 s_n의 값들이 합산될 수 있으며, 그 다음 합산된 s_n의 값들은 상기 4x4 블록에 대한 BIO 움직임 벡터 오프셋을 유도하는 데에 사용된다. 구체적으로, 아래 식이 블록 기반 BIO 유도를 위해 사용될 수 있다:

[0219]
$$s_{1,b_k} = \sum_{(x,y)\in b_k} \sum_{[i',j]\in\Omega(x,y)} \left(\tau_1 \partial I^{(1)}/\partial x + \tau_0 \partial I^{(0)}/\partial x\right)^2; s_{3,b_k} = \sum_{(x,y)\in b_k} \sum_{[i',j]\in\Omega} \left(I^{(1)} - I^{(0)}\right)\left(\tau_1 \partial I^{(1)}/\partial x + \tau_0 \partial I^{(0)}/\partial x\right);$$
$$s_{2,b_k} = \sum_{(x,y)\in b_k} \sum_{[i',j]\in\Omega} \left(\tau_1 \partial I^{(1)}/\partial x + \tau_0 \partial I^{(0)}/\partial x\right)\left(\tau_1 \partial I^{(1)}/\partial y + \tau_0 \partial I^{(0)}/\partial y\right); \quad \text{식. (24)}$$
$$s_{5,b_k} = \sum_{(x,y)\in b_k} \sum_{[i',j]\in\Omega} \left(\tau_1 \partial I^{(1)}/\partial y + \tau_0 \partial I^{(0)}/\partial y\right)^2; s_{6,b_k} = \sum_{(x,y)\in b_k} \sum_{[i',j]\in\Omega} \left(I^{(1)} - I^{(0)}\right)\left(\tau_1 \partial I^{(1)}/\partial y + \tau_0 \partial I^{(0)}/\partial y\right)$$

[0220] 여기서, b_k는 예측된 블록의 k번째 4x4 블록에 속한 샘플들의 세트를 표시한다. 관련된 움직임 벡터 오프셋들을 유도하기 위해, 식 7 및 식 8에서 s_n는 $((s_{n,bk}) >> 4)$로 교체된다.

[0221] 일부 시나리오에서, BIO의 MV 레지먼트 (regiment)는 노이즈 또는 불규칙한 움직임으로 인해 신뢰할 수 없을 수도 있다. 그러므로, BIO에서, MV 레지먼트의 크기는 특정 임계값 thBIO로 클립된다. 임계값은 현재 픽처의 모든 레퍼런스 픽처들이 모두 한 방향에서 오는지의 여부에 기초하여 결정된다. 예를 들어, 현재 픽처의 모든 참조 픽처들이 한 방향으로부터 오는 경우, 상기 임계값은 $12 \times 2^{14-d}$로 설정되고, 그렇지 않을 경우, $12 \times 2^{13-d}$로 설정된다.

[0222] BIO에 대한 그라디언트들(gradients)은 HEVC 움직임 보상 절차(예를 들어, 2D 분리가능한 유한 임펄스 응답(Finite Impulse Response: FIR)과 부합하는 동작들을 사용하는 움직임 보상 보간과 동시에 계산될 수 있다. 일부 실시예들에서, 2D 분리가능한 FIR에 대한 입력은 움직임 보상 절차에 대한 것과 같은 참조 프레임 샘플 및 블록 움직임 벡터의 분수 부분에 따른 분수 위치 (*fracX*, *fracY*) 이다. 수평 그라디언트 $\partial I/\partial x$에 대해, 신호는 먼저 역스케일링 시프트 $d-8$을 갖는 분수 위치 *fracY*에 해당하는 *BIOfilterS*를 사용하여 수직으로 보간된다. 그 다음에, 보간 필터 *BIOfilterG*는 *18-d* 만큼의 역스케일링 시프트를 갖는 분수 위치 *fracX*에 해당하는 수평 방향으로 적용된다. 수직 그라디언트 $\partial I/\partial y$에 대해, 그라디언트 필터는 역스케일링 시프트 $d-8$을 갖는 분수 위치 *fracY*에 해당하는 *BIOfilterG*를 사용하여 수직으로 적용된다. 그 다음으로, 신호 변위는 역스케일링 쉬프트 18*d*을 갖는 분수 위치 *fracX*에 해당하는 수평 방향으로 *BIOfilterS*를 사용하여 실행된다. 그라디언트 계산을 위한 보간 필터 BIOfilterG 및 신호 변위를 위한 BIOfilterF의 길이는 적절한 복잡도를 유지하기 위해 더 짧을 수 있다(예를 들어, 6-탭). 표 1은 BIO에서 블록 움직임 벡터의 상이한 분수 위치들의 그라디언트 계산을 위해 사용될 수 있는 필터들의 예를 보여준다. 표 2는 BIO에서 예측 신호 생성을 위해 사용될 수 있는 보간 필터들을 보여준다.

[0223] **표 3: BIO에서 그라디언트 계산을 위한 필터들의 예**

분수 화소 위치	그라디언트를 위한 보간 필터(BIOfilterG)
0	{ 8, -39, -3, 46, -17, 5}
1/16	{ 8, -32, -13, 50, -18, 5}
1/8	{ 7, -27, -20, 54, -19, 5}
3/16	{ 6, -21, -29, 57, -18, 5}
1/4	{ 4, -17, -36, 60, -15, 4}
5/16	{ 3, -9, -44, 61, -15, 4}
3/8	{ 1, -4, -48, 61, -13, 3}
7/16	{ 0, 1, -54, 60, -9, 2}
1/2	{ -1, 4, -57, 57, -4, 1}

[0224]

[0225] **표 4: BIO에서 예측 신호 생성을 위한 보간 필터들의 예**

분수 화소 위치	예측 신호를 위한 보간 필터(BIOfilterS)
0	{ 0, 0, 64, 0, 0, 0}
1/16	{ 1, -3, 64, 4, -2, 0}
1/8	{ 1, -6, 62, 9, -3, 1}
3/16	{ 2, -8, 60, 14, -5, 1}
1/4	{ 2, -9, 57, 19, -7, 2}
5/16	{ 3, -10, 53, 24, -8, 2}
3/8	{ 3, -11, 50, 29, -9, 2}
7/16	{ 3, -11, 44, 35, -10, 3}
1/2	{ 3, -10, 35, 44, -11, 3}

[0226]

[0227] JEM에서, BIO는 2 개의 예측들이 상이한 참조 픽처들로부터 올 경우, 모든 양 예측된 블록들에 적용될 수 있다. LIC (local illumination compensation) 이 CU에 대해 활성화되는(enabled) 경우, BIO는 비활성화 될 수 있다 (disabled).

[0228] JEM에서, OBMC는 정상 MC 절차 후에 블록에 대해 적용된다. 계산 복잡성을 줄이기 위해, BIO는 OBMC 절차 동안에는 적용되지 않을 수 있다. 이는 BIO는 블록에 대한 MC 절차에서 그것 자체의 MV를 사용하는 경우 적용되며,

주변 블록의 MC가 OBMC 절차 동안 사용되는 경우 MC 절차에서 적용되지 않음을 의미한다.

[0229] **10예시적인 광학 흐름 예측 개선(prediction refinement with optical flow: PROF)**

[0230] 일부 실시예들에서 광학 흐름으로 서브 블록 기반 어파인 움직임 보상된 예측을 개선할 수 있다. 서브 블록 기반 움직임 보상이 수행된 이후, 예측 샘플은 광학 흐름 방정식에서 유도되는 차이를 추가하여 개선되며, 이를 PROF(prediction refinement with optical flow)라 한다. 이러한 기술들은 메모리 액세스 대역폭의 증가 없이 픽셀 레벨 입도(granularity)에서 인터 예측을 달성할 수 있다.

[0231] 움직임 보상의 더 미세한 입도를 달성하기 위해, 광학 흐름을 사용하여 서브-블록 기반 어파인 움직임 보상된 예측을 개선하는 방법이 제안된다. 서브블록 기반 어파인 움직임 보상이 수행된 후, 광학 흐름 방정식에 의해 유도된 차이를 추가함으로써 루마 예측 샘플이 개선된다. PROF는 다음 4단계로 설명된다.

[0232] 단계 1) 서브 블록 기반 어파인 움직임 보상은 서브 블록 예측 $I(i,j)$를 생성하기 위해 수행된다.

[0233] 단계 2) 서브 블록 예측의 공간적 그라디언트들 $g_x(i,j)$ 및 $g_y(i,j)$는 3-탭 필터 [-1, 0, 1]을 이용하여 각 샘플 위치에서 계산된다.

$$g_x(i,j) = I(i+1,j) - I(i-1,j)$$

[0234] $$g_y(i,j) = I(i,j+1) - I(i,j-1)$$

[0235] 서브 블록 예측은 그라디언트 계산을 위해 각 측면에서 하나의 픽셀만큼씩 확장된다. 메모리 대역폭 및 복잡성을 감소시키기 위해, 확장된 경계들에서의 픽셀들은 참조 픽처에서 가장 가까운 정수 픽셀 위치들로부터 복사된다. 따라서, 패딩 영역에 대한 추가적인 보간이 회피된다.

[0236] 단계 3) 루마 예측 개선은 광학 흐름 방정식에 의해 계산된다.

[0238] $$\Delta I(i,j) = g_x(i,j) * \Delta v_x(i,j) + g_y(i,j) * \Delta v_y(i,j)$$
*.

[0239] 여기서, 도 10에서 도시된 바와 같이, $\Delta v(i,j)$는 샘플 위치 (i,j)에 대해 계산된 픽셀 MV($v(i,j)$로 표시됨)와 픽셀 (i,j)이 속한 서브 블록의 서브 블록 MV 간의 차이이다.

[0240] 어파인 모델 파라미터들과 서브 블록의 중심에 대한 픽셀 위치는 서브 블록으로부터 서브 블록으로 변경되지 않기 때문에, 제1 서브 블록에 대해 $\Delta v(i,j)$가 계산되고 동일 CU 내의 다른 서브 블록들에 대해 재사용 될 수 있다. x 및 y가 픽셀 위치로부터 서브 블록의 중심까지의 수평 및 수직 오프셋이라고 하자. $\Delta v(i,j)$는 하기 방정식에 의해 유도될 수 있다.

$$\begin{cases} \Delta v_x(x,y) = c*x + d*y \\ \Delta v_y(x,y) = e*x + f*y \end{cases}$$

[0241]

[0242] 4-파라미터 어파인 모델의 경우,

$$\begin{cases} c = f = \frac{v_{1x} - v_{0x}}{w} \\ e = -d = \frac{v_{1y} - v_{0y}}{w} \end{cases}$$

[0243]

[0244] 6-파라미터 어파인 모델의 경우,

[0245] $$\begin{cases} c = \frac{v_{1x}-v_{0x}}{w} \\ d = \frac{v_{2x}-v_{0x}}{h} \\ e = \frac{v_{1y}-v_{0y}}{w} \\ f = \frac{v_{2y}-v_{0y}}{h} \end{cases}$$

[0246] 여기서, (v_{0x}, v_{0y}), (v_{1x}, v_{1y}), (v_{2x}, v_{2y}) 는 좌측 상단, 우측 상단, 좌측 하단의 제어 점 움직임 벡터들이고, w 및 h 는 CU의 너비 및 높이이다.

[0247] 단계 4) 최종적으로, 루마 예측 개선은 서브 블록 예측 $I(i,j)$ 에 추가된다. 최종 예측 I' 는 하기 방정식으로써 생성된다:

[0248] $$I'(i,j) = I(i,j) + \Delta I(i,j)$$

[0249] **11기존 실시예들의 단점들**

[0250] 비선형 ALF(NLALF), BIO(BDOF) 및/혹은 PROF 설계들은 다음과 같은 문제점들을 갖는다:

[0251] (1) BDOF 혹은 PROF에서, 예측 샘플들에 적용되는 유도된 오프셋 값들은 원본 샘플들과 지나치게 차이가 나서 다소 큰 잔차를 야기할 수 있다.

[0252] (2) 강인한 수평 혹은 수직 패턴들이 있는 시퀀스들에 대해 차선책일 수 있는 2-D 필터링이 적용된다.

[0253] **12 비디오 코딩에서 샘플 개선 및 필터링을 위한 예시적인 방법들**

[0254] 현재 개시된 기술의 실시예들은 기존 구현들의 단점들을 극복함으로써, 더 높은 코딩 효율을 갖는 비디오 코딩을 제공한다. 개시된 기술을 기반으로 하는 비디오 코딩을 위한 비디오 코딩을 위한 샘플 개선 및 필터링 방법들은 기존 및 미래의 비디오 코딩 표준들 모두를 향상시킬 수 있으며, 다양한 구현에 대해 하기 예들에서 설명된다. 하기에 제공된 개시된 기술의 예들은 일반적인 개념을 설명하는 것이며 제한하는 것으로 해석되지 않아야 한다. 일 예로서, 반대로 명시적으로 나타내지 않는 한, 이러한 예들에서 설명된 다양한 특징들이 결합될 수 있다.

[0255] 아래의 예들에서 언급된 '필터링 방법'은 적응적 루프 필터/포스트 복원 필터들(예를 들어, 양방향 필터, 확산 필터 등)을 참조할 수 있다.

[0256] 1. 각 인접 픽셀 및 현재 픽셀 간의 차이를 클리핑하는 대신, 클리핑은 인접 픽셀들의 가중 평균에 적용될 수 있다.

[0257] a. 일부 실시예들에서, 클리핑은 하기와 같이 수행될 수 있으며, K가 클리핑 연산이다.

[0258] $$O(x,y) = I(x,y) + K\left(\sum_{(i,j)\neq(0,0)} w(i,j).\left(I(x+i,y+j) - I(x,y)\right)\right)$$

[0259] 2. 1-D 필터링 방법이 활용될 수 있다.

[0260] a. 일부 실시예들에서, 1-D 필터는 수평적으로 인접한 샘플들만을 사용할 수 있다.

[0261] i. 일부 실시예들에서, 필터링 프로세스는 하기와 같은 방식으로 표현될 수 있다.

[0262] $$O(x,y) = \sum_{(i,j)} w(i,j) * \sum_{si=-T}^{T} ws(k) * I(x+i+si, y+j)$$

[0263] ii. 샘플들/필터 탭(예를 들어, T)의 수는 픽셀들에 따라 달라질 수 있다. 예를 들어, CTU 경계의 픽셀의 경

우, 현재 CTU 내에 위치하지 않은 픽셀들은 사용되지 않는다.

[0264] b. 일부 실시예들에서, 짧은 탭 필터는 수직 인접 샘플들만 사용할 수 있다.

$$O(x,y) = \sum_{(i,j)} w(i,j) * \sum_{si=-T}^{T} ws(k) * I(x+i, y+j+si)$$

[0265]

[0266] i. T는 픽셀마다 달라질 수 있다. 예를 들어, CTU 경계의 픽셀들의 경우, 현재 CTU 내에 있지 않은 픽셀들은 사용되지 않는다.

[0267] c. 일부 실시예들에서, 1-D 탭 필터는 수평/수직 방향을 제외한 한 방향을 따라 인접하는 샘플들만을 사용할 수 있다.

[0268] d. 일부 실시예들에서, ws(si)는 픽셀들의 클래스마다 달라질 수 있다.

[0269] e. 일부 실시예들에서, ws(si)는 픽셀들의 상이한 클래스에 대해 동일할 수 있다.

[0270] f. 일부 실시예들에서, ws(si)는 디코더에 시그널링될 수 있다.

[0271] g. 일부 실시예들에서 ws(si)는 기 정의될 수 있다.

[0272] 3. 1-D 비선형 적응적 루프 필터링 방법이 활용될 수 있다.

[0273] a. 일부 실시예들에서, 짧은 탭 필터는 수평으로 인접한 샘플들만을 사용할 수 있다.

[0274] i. 일부 실시예들에서, 비선형 필터링 방법은 수평 인접 샘플들과 함께 다음과 같은 방식으로 표현될 수 있다.

$$O'(x,y) = I(x,y) + \sum_{(i,j)\neq(0,0)} w(i,j).K\left(\sum_{si=-T}^{T} ws(si) * I(x+i+si, y+j) - I(x,y), k(i,j)\right)$$

[0275]

[0276] ii. T는 픽셀마다 달라질 수 있다. 예를 들어, CTU 경계의 픽셀의 경우, 현재 CTU 내에 있지 않은 픽셀들은 사용되지 않는다.

[0277] b. 일부 실시예들에서, 1-D 비선형 필터는 수직 인접 샘플들만을 사용할 수 있다.

[0278] i. 일부 실시예들에서, 비선형 필터링 방법은 수직 인접 샘플들과 함께 다음과 같은 방식으로 표현될 수 있다.

$$O'(x,y) = I(x,y) + \sum_{(i,j)\neq(0,0)} w(i,j).K\left(\sum_{si=-T}^{T} ws(si) * I(x+i, y+j+si) - I(x,y), k(i,j)\right)$$

[0279]

[0280] ii. T는 픽셀들마다 달라질 수 있다. 예를 들어 CTU 경계의 픽셀의 경우 현재 CTU 내에 있지 않은 픽셀들은 사용되지 않는다.

[0281] c. 일부 실시예들에서, 1-D 비선형 탭 필터는 수평/수직 방향을 제외한 한 방향으로 인접한 샘플들만을 사용할 수 있다.

[0282] d. 일부 실시예들에서, ws(si)는 픽셀들의 클래스마다 상이할 수 있다.

[0283] e. 일부 실시예들에서, ws(si)는 상이한 픽셀들의 클래스에 대해 동일할 수 있다.

[0284] f. 일부 실시예들에서, ws(si)는 디코더로 시그널링될 수 있다.

[0285] g. 일부 실시예들에서, ws(si)는 기 정의될 수 있다.

[0286] 4. 양방향 필터 혹은 아다마르(Hadamard) 필터에서, 픽셀들은 서로 다른 방향으로 필터링될 수 있다.

[0287] a. 일부 실시예들에서, 픽셀들은 그라디언트 정보에 기초하여 서로 다른 방향으로 필터링될 수 있다.

[0288] b. 일부 실시예들에서, 각 MxN 서브 블록에 대해, 수평 그라디언트 및 수직 그라디언트가 계산될 수 있는데, 수평 그라디언트의 절대 합이 수직 그라디언트의 절대 합의 이상인 경우, 서브 블록 내의 픽셀들은 수평적으로 필터링될 수 있다.

[0289] i. 또는, 서브 블록 내의 픽셀들은 수직적으로 필터링될 수 있다.

[0291] * c. 일부 실시예들에서, 각 MxN 서브 블록에 대해, 수평 그라디언트 및 수직 그라디언트가 계산될 수 있는데, 수평 그라디언트의 절대 합이 수직 그라디언트의 절대 합의 이하인 경우, 서브 블록 내의 픽셀들은 수직적으로 필터링될 수 있다.

[0292] i. 또는, 서브 블록 내 픽셀들은 수평적으로 필터링될 수 있다.

[0293] d. 일부 실시예들에서, 각 MxN 서브 블록에 대해, 수평 그라디언트, 수직 그라디언트, 45도 대각선 그라디언트 및 135도 대각선 그라디언트가 계산될 수 있으며, 서브 블록 내의 픽셀들은 그라디언트의 절대 합이 가장 큰 방향을 따라 필터링될 수 있다.

[0294] e. 일부 실시예들에서, 각 MxN 서브 블록에 대해, 수평 그라디언트, 수직 그라디언트, 45도 대각선 그라디언트 및 135도 대각선 그라디언트가 계산될 수 있으며, 서브 블록 내의 픽셀들은 그라디언트의 절대 합이 가장 작은 방향을 따라 필터링될 수 있다.

[0295] 5. BDOF에서, 그라디언트는 샘플 개선 혹은/및 MV 오프셋 유도에 사용되기 이전에 수정될 수 있다.

[0296] a. 일부 실시예들에서, BDOF에서 계산된 공간적 혹은/및 시간적 그라디언트들은 범위 [min, max]로 클리핑될 수 있다.

[0297] i. 변수들 min 및 max는 공간적 및 시간적 그라디언트들에 대해 상이할 수 있다.

[0298] ii. 변수들 min은 0보다 작고, max는 0보다 클 수 있다.

[0299] iii. 변수들 min 및 max는 샘플들의 입력 비트 깊이에 의존할 수 있다.

[0300] iv. 변수들 min 및 max는 중간 샘플들의 생성에 사용된 보간 필터들에 의존할 수 있다.

[0301] v. 변수들 min 및 max는 시그널링될 수 있다.

[0302] vi. 변수들 min 및 max는 기 정의될 수 있다.

[0303] b. 일부 실시예들에서, BDOF에서 계산된 공간적 혹은/및 시간적 그라디언트들은 비선형 함수에 의해 수정될 수 있다.

[0304] i. 일 예로서, 로지스틱 시그모이드 함수가 사용될 수 있다.

[0305] c. 마찬가지로, PROF에서, 그라디언트는 샘플 개선 혹은/및 MV 오프셋 유도에 사용되기 전에 수정될 수 있다.

[0306] 6. BDOF/PROF에서, 개선된 샘플 값 혹은 예측 샘플 간의 차이(혹은 오프셋) 및 그것의 개선된 샘플 값은 최종 복원 샘플 값의 유도에 사용되기 전에 더 수정될 수 있다.

[0307] a. BDOF/PROF에서, 클리핑 연산은 예측 샘플과 그것의 개선된 샘플 값 간의 차이에 추가될 수 있다.

[0308] b. 일부 실시예들에서, 차이는 범위 [min, max]로 클리핑될 수 있다.

[0309] i. 변수들 min 및 max는 공간적 및 시간적 그라디언트들에 대해 상이할 수 있다.

[0310] ii. 변수들 min은 0보다 작고, max는 0보다 클 수 있다.

[0311] iii. 변수들 min 및 max는 샘플들의 입력 비트 깊이에 의존할 수 있다.

[0312] iv. 변수들 min 및 max는 중간 샘플들을 생성하기 위해 사용되는 보간 필터에 의존할 수 있다.

[0313] v. 변수들 min 및 max는 시그널링될 수 있다.

[0314] vi. 변수들 min 및 max는 기 정의될 수 있다.

[0315] c. 또는, 또한, 한 예측 샘플의 최종 복원 값은 클리핑된 차이에 의존할 수 있다.

[0316] d. 또는, 또한, 한 예측 샘플의 최종 복원 값은 수정되고 개선된 샘플/수정되고 개선된 차이에 의존할 수 있다.

[0317] 7. 상기 방법들에서 활용되는 클리핑 파라미터들은, 예를 들어, 시퀀스 레벨, 픽처 레벨, CTU 레벨 등과 같은 일부 비디오 코딩 유닛 레벨에서 시그널링될 수 있다.

[0318] a. 일부 실시예들에서, 그들은 SPS/VPS/PPS/타일 그룹 헤더/CTU 행들/영역들에서 시그널링될 수 있다.

[0319] b. 또는, 파라미터들은 즉석에서 유도될 수 있다.

[0320] c. 또는, 파라미터들은 양자화 파라미터들, 타일 그룹 유형, 코딩된 모드 정보, 정형 파라미터들 등에 따라 유도될 수 있다.

[0321] 8. 상기 방법들을 적용할 지 여부는 색상 형식에 의존할 수 있다.

[0322] a. 제안된 방법은, 예를 들어, G 혹은 루마 색상 구성요소 같은, 하나의 색상 구성요소에 대해서만 적용가능할 수 있다.

[0323] b. 또는, 제안된 방법은 4:4:4 색상 형식의 모든 색상 구성요소들에 적용가능할 수 있다.

[0324] 상기에서 설명된 예시들은, 비디오 디코더 혹은 비디오 인코더에서 실행될 수 있는, 예를 들어, 방법들 1100 및 1150과 같은 하기에서 설명되는 방법들의 내용과 통합될 수 있다.

[0325] 도 11a는 예시적인 비디오 프로세싱 방법의 흐름도를 도시한 것이다. 방법(1100)은 단계 1102에서 현재 비디오 블록에 대해, 현재 비디오 블록의 인접 픽셀들의 가중 평균에 적용되는 클리핑 연산을 포함하는 필터링 프로세스를 구성하는 단계를 포함한다.

[0326] 일부 실시예들에서, 필터링 프로세스는 다음을 포함한다.

[0327] $$O(x,y) = I(x,y) + K\left(\sum_{(i,j)\neq(0,0)} w(i,j).\left(I(x+i,y+j) - I(x,y)\right)\right),$$

[0328] 여기서 $I(x,y)$는 현재 비디오 블록의 (x,y)번째 픽셀이고, $O(x,y)$는 (x,y)번째 픽셀의 출력 값이며, $w(i,j)$는 가중 인자이며 $K(\cdot)$는 클리핑 연산이다.

[0329] 방법 (1100)은 단계 1104에서 필터링 프로세스에 기초하여 대응 비트스트림 표현으로부터 현재 비디오 블록을 복원하는 단계를 포함한다.

[0330] 일부 실시예들에서, 클리핑 연산의 하나 이상의 파라미터들은 SPS(sequence parameter set), VPS(video parameter set), PPS(picture parameter set), 타일 그룹 헤더, 코딩 트리 유닛(CTU) 행 혹은 CTU 영역에서 시그널링될 수 있다.

[0331] 일부 실시예들에서, 클리핑 연산의 하나 이상의 파라미터들은 즉석에서 유도된다.

[0332] 일부 실시예들에서, 클리핑 연산의 하나 이상의 파라미터들은 하나 이상의 양자화 파라미터들, 하나 이상의 정형 파라미터들, 타일 그룹 유형 혹은 현재 비디오 블록의 코딩된 모드에 기초하여 유도된다.

[0333] 일부 실시예들에서, 필터링 프로세스는 적응적 루프 필터링 프로세스 혹은 복원 후 필터링 프로세스에 대응된다. 일 예로서, 복원 후 필터링 프로세스는 양방향 필터, 아다마르 필터 혹은 확산 필터 중 적어도 하나를 사용한다.

[0334] 도 11b는 예시적인 비디오 프로세싱 방법의 흐름도를 도시한 것이다. 방법(1150)은 단계 1152에서 현재 비디오 블록에 대해 1차원 필터링 프로세스를 구성하는 단계를 포함한다.

[0335] 방법(1150)은 단계 1154에서 1차원 필터링 프로세스에 기초하여 대응 비트스트림 표현으로부터 현재 비디오 블록을 복원하는 단계를 포함한다.

[0336] 일부 실시예들에서, 1차원 필터링 프로세스는 현재 블록의 수평적으로 인접한 샘플들, 수직적으로 인접한 샘플들, 45도 대각선으로 인접한 샘플들, 혹은 135도 대각선으로 인접한 샘플들 중 적어도 하나에 적용된다. 일 예

로서, 코딩 트리 유닛(CTU) 경계의 픽셀에 적용되는 1차원 필터링 프로세스의 필터탭들에 대한 첫 번째(제1) 세트는 전체적으로 CTU 내인 픽셀에 적용되는 1차원 필터링 프로세스의 필터 탭들의 두 번째(제2) 세트와 상이하다. 다른 일 예로서, 필터 탭들의 첫 번째(제1) 세트 혹은 필터 탭들의 두 번째(제2) 세트는 기 정의되거나 대응 비트스트림 표현으로 시그널링된다.

[0337] 일부 실시예들에서, 1차원 필터링 프로세스를 구성하는 단계는 그라디언트 정보에 기초한다. 일 예로서, 그라디언트 정보는 현재 비디오 블록의 MxN 서브 블록에 대한 복수의 수평 그라디언트들 및 복수의 수직 그라디언트들을 포함한다. 다른 일 예로서, 복수의 수평 그라디언트들의 절대 값 합은 복수의 수직 그라디언트들의 절대 값 합 이상이다. 또다른 일 예로서, 복수의 수평 그라디언트들의 절대 값 합은 복수의 수직 그라디언트들의 절대 값 합 이하이다. 또다른 일 예로서, 1차원 필터링 프로세스는 수평적으로 인접한 샘플들에 적용된다. 또다른 일 예로서, 1차원 필터링 프로세스는 수직적으로 인접한 샘플들에 적용된다.

[0338] 일부 실시예들에서, 그라디언트 정보는 MxN 서브 블록에 대한 복수의 45도 대각선 그라디언트들 및 정보는 MxN 서브 블록에 대한 복수의 135도 대각선 그라디언트들을 더 포함한다. 일 예로서, 1차원 필터링 프로세스는 복수의 수평 그라디언트들, 수직 그라디언트들, 45도 대각선 그라디언트들 및 135도 대각선 그라디언트들의 절대 값 합 중 최대에 대응되는 방향을 따라 적용된다. 다른 일 예로서, 1차원 필터링 프로세스는 복수의 수평 그라디언트들, 수직 그라디언트들, 45도 대각선 그라디언트들 및 135도 대각선 그라디언트들의 절대 값 합 중 최소에 대응되는 방향을 따라 적용된다.

[0339] 일부 실시예들에서, 1차원 필터링 프로세스를 구성하는 단계는 그라디언트 정보의 수정에 기초한다. 일 예로서, 수정은 범위 [min, max]의 클리핑 연산을 그라디언트 정보에 적용하는 것을 포함한다. 다른 일 예로서, min 및 max는 공간적 그라디언트들에 대한 값들의 첫 번째(제1) 세트로 구성되고, 여기서 min 및 max는 시간적 그라디언트들에 대한 값들의 첫 번째 세트와 상이한 값들의 두 번째(제2) 세트로 구성된다. 또다른 예로서, min<0 및 max>0이다. 또다른 예로서, min 및 max에 대한 값들은 현재 비디오 블록의 값들의 입력 비트 깊이에 기초하여 선택된다. 또다른 예로서, min 및 max에 대한 값들은 기 정의되거나 대응 비트스트림 표현으로 시그널링된다.

[0340] 일부 실시예들에서, 수정은 비선형 함수를 그라디언트 정보에 적용하는 것을 포함한다. 일 예로서, 비선형 함수는 로지스틱 시그모이드 함수일 수 있다.

[0341] 일부 실시예들에서, 1차원 필터링 프로세스는 적응적 루프 필터링 프로세스 혹은 복원 후 필터링 프로세스에 대응된다. 일 예로서, 복원 후 필터링 프로세스는 양방향 필터, 아다마르 필터 혹은 확산 필터 중 적어도 하나를 사용한다.

[0342] **13개시된 기술의 예시적인 구현예들**

[0343] 도 12은 비디오 프로세싱 장치(1200)의 블록도를 도시한 것이다. 장치(1200)는 이 명세서에 기재된 하나 이상의 방법들을 구현하는 데 사용될 수 있다. 장치(1200)는 스마트폰, 태블릿, 컴퓨터, 사물인터넷(Internet of Things; IoT) 수신기 등 내에 구현될 수 있다. 장치(1200)는 하나 이상의 프로세서(1202)들과, 하나 이상의 메모리(1204)들과, 및 비디오 프로세싱 하드웨어(1206)를 포함할 수 있다. 프로세서(들)(1202)는 이 명세서에 기재된 (방법(1100) 및 방법(1150)을 포함하되, 이에 제한되는 것은 아님) 하나 이상의 방법들을 구현하도록 구성될 수 있다. 메모리(메모리들)(1204)는 이 명세서에 기재된 방법과 기법들을 구현하는 데 사용되는 데이터와 코드를 저장하는 데 사용될 수 있다. 비디오 프로세싱 하드웨어(1206)는 하드웨어 회로 내에서 이 명세서에 기재된 일부 기법들을 구현하는 데 사용될 수 있다.

[0344] 일부 실시예들에서, 비디오 코딩 방법들은 도 12에 개시된 하드웨어 플랫폼에서 구현되는 장치를 이용하여 구현될 수 있다.

[0345] 도 13은 여기에 개시된 다양한 기술들이 구현될 수 있는 예시적인 비디오 프로세싱 시스템 1300을 나타내는 블록도를 도시한다. 다양한 실시예들은 시스템 1300의 일부 혹은 모든 컴포넌트들을 포함할 수 있다. 시스템 1300은 비디오 컨텐츠를 수신하기 위해 입력 1302을 포함할 수 있다. 비디오 콘텐츠는 예를 들어, 8 또는 10 비트 다중 컴포넌트 픽셀 값들과 같은 raw 혹은 압축되지 않은 포맷으로 수신될 수 있고, 압축되거나 인코딩된 포맷일 수도 있다. 입력 1302는 네트워크 인터페이스, 주변 버스 인터페이스, 혹은 스토리지 인터페이스를 나타낼 수 있다. 네트워크 인터페이스 예시들은 이더넷(ethernet), PON(passive optical network) 등과 같은 유선 인터페이스들을 포함하고, Wi-Fi 혹은 셀룰러 인터페이스들과 같은 무선 인터페이스들도 포함한다.

[0346] 시스템 1300은 본 문서에 개시된 다양한 코딩 혹은 인코딩 방법들을 구현할 수 있는 코딩 컴포넌트 1304를 포함할 수 있다. 코딩 컴포넌트 1304는 비디오의 코딩된 표현을 생성하기 위해 입력 1302로부터 코딩 컴포넌트 1304

의 출력으로의 비디오의 평균 비트율(bitrate)을 감소시킬 수 있다. 따라서 코딩 기술들은 종종 비디오 압축 혹은 비디오 트랜스코딩 기술들로 불리기도 한다. 코딩 컴포넌트 1304의 출력은 컴포넌트 1306에 의해 표현되는 바와 같이 저장되거나 연결된 통신을 통해 전송될 수 있다. 입력 1302에서 수신된 비디오의 저장된 혹은 통신된 비트스트림(혹은 코딩된) 표현은 컴포넌트 1308에 의해 디스플레이 인터페이스 1310로 송신되는 픽셀 값들 혹은 디스플레이 가능한 비디오를 생성하는 데 사용될 수 있다. 비트스트림 표현으로부터 사용자가 시청 가능한 비디오를 생성하는 프로세스는 비디오 종종 압축 해제로 불린다. 또한, 특정 비디오 프로세싱 동작들이 "코딩" 동작들 혹은 도구들로 불리지만, 코딩 도구 혹은 동작은 인코더에서 사용되고, 코딩 결과를 역전시키는 해당 디코딩 도구 혹은 동작은 디코더에 의해 수행될 것이다.

[0347] 주변 버스 인터페이스 또는 디스플레이 인터페이스의 예는 범용 직렬 버스(universal serial bus; USB), 고화질 멀티미디어 인터페이스(high definition multimedia interface; HDMI), 또는 Displayport 등을 포함할 수 있다. 저장 인터페이스의 예는 SATA(serial advanced technology attachment; 직렬 고급 결합 기술; 직렬 ATA), PCI, IDE 인터페이스 등을 포함할 수 있다. 이 명세서에 기재된 기법들은 모바일폰, 랩탑, 스마트폰 또는 디지털 데이터 처리 및/또는 비디오 디스플레이가 가능한 다른 기기들 등의 여러 전자 기기들에 구현될 수 있다.

[0348] 도 14는 현 기술에 따른 비디오 프로세싱 방법(1400)의 흐름도 표현이다. 방법(1400)은 동작 1410에서, 비디오의 현재 블록과 비디오의 비트스트림 표현 간의 변환을 위해 현재 블록에서 픽셀 및 그 픽셀의 인접 픽셀들 간의 차이의 가중 평균에 클리핑 연산을 적용하는 단계를 포함한다. 방법 (1400)은 동작 1420에서 클리핑 연산의 출력에 기초하여 변환을 수행하는 단계를 포함한다.

[0349] 일부 실시예들에서, 클리핑 연산은

$$O(x,y) = I(x,y) + K\left(\sum_{(i,j)\neq(0,0)} w(i,j).\left(I(x+i,y+j) - I(x,y)\right)\right)$$

[0350] 로써 적용되고, 여기서 $I(x,y)$ 는 현재 블록의 (x,y) 번째 픽셀이고, $O(x,y)$ 는 (x,y) 번째 픽셀의 출력 값이며, $w(i,j)$ 는 가중 인자이고 $K(\cdot)$ 는 클리핑 연산이다. 일부 실시예들에서, 클리핑 연산의 하나 이상의 파라미터들은 SPS(sequence parameter set), VPS(Video parameter set), PPS(picture parameter set), 타일 그룹 헤더, 코딩 트리 유닛(CTU) 행 혹은 CTU 영역에서 시그널링된다. 일부 실시예들에서, 클리핑 연산의 하나 이상의 파라미터들은 즉석에서 유도된다. 일부 실시예들에서, 클리핑 연산의 하나 이상의 파라미터들은 하나 이상의 양자화 파라미터들, 하나 이상의 정형 파라미터들, 타일 그룹 유형 혹은 현재 비디오 블록의 코딩된 모드에 기초하여 유도된다.

[0351] 도 15는 현 기술에 따른 비디오 프로세싱 방법(1500)의 흐름도 표현이다. 방법(1500)은 동작 1510에서, 비디오의 현재 블록 및 비디오의 비트스트림 표현 간의 변환을 위해 현재 블록의 인접 픽셀들의 세트에 기초하는 필터링 프로세스를 구성하는 단계를 포함한다. 방법 (1500)은 동작 1520에서 필터링 프로세스의 출력에 기초하여 변환을 수행하는 단계를 포함한다.

[0352] 일부 실시예들에서, 필터 프로세싱은 현재 블록의 인접 픽셀들의 세트에 기초한 1차원 필터링 연산을 포함한다. 일부 실시예들에서, 1차원 필터링 연산은 비선형 적응적 루프 필터링 방법을 포함한다.

[0353] 일부 실시예들에서, 인접 픽셀들의 세트는 현재 블록의 수평 인접 픽셀들을 포함한다. 일부 실시예들에서, 필터링 프로세스는 $O(x,y) = \sum_{(i,j)} w(i,j) * \sum_{si=-T}^{T} ws(si) * I(x+i+si, y+j)$ 를 포함하고, $I(x,y)$ 는 현재 블록의 (x,y) 번째 픽셀이고, $O(x,y)$ 는 (x,y) 번째 픽셀의 출력 값이며, $w(i,j)$ 는 가중 인자이고, T는 필터링 프로세스에 사용되는 복수개의 픽셀들의 수를 지시하고, $ws(si)$ 는 T에 대응되는 필터링 계수들을 나타낸다. 일부 실시예들에서, 필터링 프로세스는

$$O(x,y) = I(x,y) + \sum_{(i,j)\neq(0,0)} w(i,j).K\left(\sum_{si=-T}^{T} ws(si) * I(x+i+si,y+j) - I(x,y), k(i,j)\right)$$ 를

$I(x,y)$ 포함하고, (x,y) 는 현재 블록의 번째 픽셀이고, $O(x,y)$ (x,y) 는 번째 픽셀의 출력값이고, $w(i,j)$ 는 가중 인자이고, $K(\cdot)$ 는 클리핑 연산이며, T는 필터링 프로세스에 사용될 픽셀들의 수이고, $ws(si)$ 는 T에 대응되는 필터링 계수들을 나타내며 $k(i,j)$ 는 클리핑 파라미터들을 나타낸다.

[0354] 일부 실시예들에서, 인접 픽셀들의 세트는 현재 블록의 수직 인접 픽셀들을 포함한다. 일부 실시예들에서, 필터링 프로세스는 $O(x,y) = \sum_{(i,j)} w(i,j) * \sum_{si=-T}^{T} ws(si) * I(x+i, y+j+si)$ 를 포함하고, $I(x,y)$ 는 현재 블록의 (x,y) 번째 픽셀이고, $O(x,y)$ (x,y) 는 번째 픽셀의 출력값이고, $w(i,j)$ 는 가중 인자이며, T 는 필터링 프로세스에 사용될 픽셀들의 수를 나타내고, $ws(si)$ 는 T에 대응되는 필터링 계수들을 나타낸다. 일부 실시예들에서, 필터링 프로세스는 $O(x,y) = I(x,y) + \sum_{(i,j)\neq(0,0)} w(i,j).K(\sum_{si=-T}^{T} ws(si) * I(x+i, y+j+si) - I(x,y), k(i,j))$ 를 $I(x,y)$ 포함하고, (x,y) 는 현재 블록의 번째 픽셀이며, $O(x,y)$ (x,y) 는 번째 픽셀의 출력값이고, $w(i,j)$ 는 가중 인자이며, $K(\cdot)$ 는 클리핑 연산이고, T는 필터링 프로세스에 사용될 픽셀들의 수를 나타내며, $ws(si)$ 는 T에 대응되는 필터링 계수들을 나타내고, $k(i,j)$ 는 클리핑 파라미터들을 나타낸다.

[0355] 일부 실시예들에서, T는 현재 블록의 픽셀들에 따라 달라진다. 일부 실시예들에서, 현재 블록의 (x,y) 번째 픽셀이 현재 블록의 경계에 위치한 경우, 필터링 프로세스에 사용될 픽셀들의 수 T는 현재 블록의 외부에 위치한 픽셀들을 배제한다. 일부 실시예들에서, $ws(k)$ 는 현재 블록의 픽셀 마다 달라진다. 일부 실시예들에서, $ws(k)$ 는 비트스트림 표현에서 시그널링된다. 일부 실시예들에서, $ws(k)$ 는 기 정의된다.

[0356] 일부 실시예들에서, 1차원 필터링 연산은 필터를 수평 혹은 수직 방향에 따라 적용하는 것을 배제한다.

[0357] 일부 실시예들에서, 필터링 프로세싱은 상이한 방향들에 따라 필터를 적용하는 것을 포함한다. 일부 실시예들에서, 필터는 양방향 필터 혹은 아다마르(Hadamard) 필터를 포함한다. 일부 실시예들에서, 필터는 현재 블록과 연관된 그라디언트 정보에 기초하여 서로 다른 방향에 적용된다. 일부 실시예들에서, 방법은 현재 블록 내의 서브 블록에 대해 서브 블록의 픽셀들의 수평 그라디언트들 및 수직 그라디언트들을 결정하는 단계; 및 수평 그라디언트들의 절대 합이 수직 그라디언트들의 절대 합 이상인 경우 서브 블록에 필터를 수평적으로 적용하는 단계;를 포함한다. 일부 실시예들에서, 현재 블록 내의 서브블록에 대해, 서브 블록의 픽셀들의 수평 그라디언트들 및 수직 그라디언트들을 결정하는 단계; 및 수평 그라디언트들의 절대 합이 수직 그라디언트들의 절대 합 이상인 경우 서브블록에 필터를 수직적으로 적용하는 단계;를 포함한다. 일부 실시예들에서, 현재 블록 내의 서브 블록에 대해, 서브블록의 픽셀들의 수평 그라디언트들 및 수직 그라디언트들을 결정하는 단계; 및 수평 그라디언트들의 절대 합이 수직 그라디언트들의 절대 합 이하인 경우 서브블록에 필터를 수평적으로 적용하는 단계;를 포함한다. 일부 실시예들에서, 현재 블록 내의 서브 블록에 대해, 서브블록의 픽셀들의 수평 그라디언트들 및 수직 그라디언트들을 결정하는 단계; 및 수평 그라디언트들의 절대 합이 수직 그라디언트들의 절대 합 이하인 경우 서브블록에 필터를 수직적으로 적용하는 단계;를 포함한다. 일부 실시예들에서, 방법은 현재 블록 내의 서브 블록에 대해 서브블록의 픽셀들의 수평 그라디언트들, 수직 그라디언트들, 45도 대각선 그라디언트들, 135도 대각선 그라디언트들을 결정하는 단계; 및 결정된 수평 그라디언트들, 수직 그라디언트들, 45도 대각선 그라디언트들, 및 135도 대각선 그라디언트들에 기초하여 그라디언트의 절대 합이 가장 큰 방향을 따라 서브 블록에 필터를 적용하는 단계;를 포함한다. 일부 실시예들에서, 방법은 현재 블록 내의 서브 블록에 대해 서브블록의

픽셀들의 수평 그라디언트들, 수직 그라디언트들, 45도 대각선 그라디언트들, 135도 대각선 그라디언트들을 결정하는 단계; 및 결정된 수평 그라디언트들, 수직 그라디언트들, 45도 대각선 그라디언트들, 및 135도 대각선 그라디언트들에 기초하여 그라디언트의 절대 합이 가장 작은 방향을 따라 서브 블록에 필터를 적용하는 단계;를 포함한다.

[0358] 일부 실시예들에서, 필터링 프로세스가 현재 블록에 적용 가능한지 여부는 현재 블록의 색상 형식에 기초한다. 일부 실시예들에서, 필터링 프로세스는 현재 블록의 단일 색상 구성요소에 적용 가능하다. 일부 실시예들에서, 단일 색상 구성요소는 루마 색상 구성요소 혹은 녹색 구성요소이다. 일부 실시예들에서, 필터링 프로세스는 현재 블록의 색상 형식이 4:4:4 색상 형식인 경우 현재 블록의 모든 색상 구성요소들에 적용 가능하다.

[0359] 도 16은 현 기술에 따른 비디오 프로세싱 방법(1600)의 흐름도 표현이다. 방법(1600)은 동작 1610에서, 비디오의 블록과 비디오의 비트스트림 표현 간의 변환을 위해 클리핑 연산을 개선 값에 적용함으로써 블록의 예측 샘플에 대한 개선 값을 수정하는 단계를 포함한다. 개선 값은 광학 흐름 코딩 프로세스의 그라디언트 값에 기초하여 유도된다. 클리핑 연산의 출력은 범위 내이다. 이 방법(1600)은 동작 1620에서, 수정된 개선 값에 기초하여 예측 샘플을 개선하는 단계를 포함한다. 이 방법(1600)은 동작 1630에서, 개선된 예측 샘플에 기초하여 변환을 수행하는 단계를 포함한다.

[0360] 일부 실시예들에서, 광학 흐름 코딩 프로세스는 광학 흐름 프로세스에 의한 예측 개선을 포함한다. 일부 실시예들에서, 광학 흐름 코딩 프로세스는 양방향 광학 흐름 프로세스를 포함한다.

[0361] 일부 실시예들에서, 범위는 샘플의 입력 비트 깊이에 기초하여 개선된다. 일부 실시예들에서, 범위는 샘플의 공간적 그라디언트 혹은 시간적 그라디언트에 기초한 변수이다. 일부 실시예들에서, 범위는 [min, max]이고, min은 0보다 작고 max는 0보다 크다. 일부 실시예들에서, 범위는 샘플과 연관된 중간(intermediate) 샘플들을 생성하기 위한 보간 필터에 기초한다. 일부 실시예들에서, 범위는 비트스트림 표현에서 시그널링된다. 일부 실시예들에서, 범위는 기정의된다.

[0362] 일부 실시예들에서, 샘플의 최종 복원 값은 클리핑 연산의 출력에 기초하여 결정된다. 일부 실시예들에서, 샘플의 최종 복원 값은 샘플에 대한 수정된 예측 값에 기초하여 결정된다. 일부 실시예들에서, 블록은 어파인 움직임으로 코딩된다. 일부 실시예들에서, 개선 값은 블록 내 복수의 샘플들 각각에 대해 유도된다.

[0363] 도 17은 현 기술에 따른 비디오 프로세싱 방법(1700)의 흐름도 표현이다. 방법 (1700)은 동작 1710에서 비디오의 블록과 비디오의 비트스트림 표현 간의 변환을 위해, 광학 흐름 코딩 프로세스에서 그라디언트 값을 수정하는 단계를 포함한다. 또한 방법(1700)은 동작 1720에서 변환을 위해 수정된 그라디언트 값에 기초하여 후속 예측 연산을 수행하는 단계를 포함한다. 일부 실시예들에서, 광학 흐름 코딩 프로세스는 광학 흐름 프로세스 혹은 양방향 광학 흐름 프로세스에 따른 예측 개선을 포함한다. 일부 실시예들에서, 후속 예측 연산은 샘플 개선 연산 혹은 움직임 벡터 오프셋 유도 연산을 포함한다.

[0364] 일부 실시예들에서, 그라디언트 값을 수정하는 단계는 그라디언트 값에 클리핑 연산을 적용하는 단계를 포함하고, 여기서 클리핑 연산의 출력은 범위 내이다. 일부 실시예들에서, 범위는 그라디언트가 예측 샘플의 공간적 그라디언트 혹은 시간적 그라디언트인지 여부에 기초하는 변수이다. 일부 실시예들에서, 범위는 [min, max]이고, min은 0보다 작고 max는 0보다 크다. 일부 실시예들에서, 범위는 샘플의 입력 비트 깊이에 기초하여 결정된다. 일부 실시예들에서, 범위는 범위는 샘플에 연관된 중간(intermediate) 샘플들의 생성을 위한 보간 필터에 기초한다. 일부 실시예들에서, 범위는 비트스트림 표현에서 시그널링된다. 일부 실시예들에서, 범위는 기정의된다. 일부 실시예들에서, 그라디언트 값의 수정은 그라디언트 값에 비선형 함수를 적용하는 것을 포함한다. 일부 실시예들에서, 비선형 함수는 로지스틱 시그모이드(logistic sigmoid) 함수를 포함한다.

[0365] 일부 실시예들에서, 블록은 어파인 움직임으로 코딩된다. 일부 실시예들에서, 그라디언트 값은 블록 내 복수의 샘플들 각각에 대해 유도된다.

[0366] 일부 실시예들에서, 클리핑 연산의 하나 이상의 파라미터들은 SPS(sequence parameter set), VPS(video parameter set), PPS(picture parameter set), 타일 그룹 헤더, 코딩 트리 유닛(CTU) 행 혹은 CTU 영역에서 시그널링된다. 일부 실시예들에서, 클리핑 연산의 하나 이상의 파라미터들은 즉석에서 유도된다. 일부 실시예들에서, 클리핑 연산의 하나 이상의 파라미터들은 하나 이상의 양자화 파라미터들, 하나 이상의 정형 파라미터들, 타일 그룹 유형 혹은 현재 비디오 블록의 코딩된 모드에 기초하여 시그널링된다.

[0367] 일부 실시예들에서, 수정이 현재 블록에 적용 가능한지 여부는 현재 블록의 색상 형식에 기초한다. 일부 실시예들에서, 수정은 현재 블록의 단일 색상 구성요소에 적용 가능합니다. 일부 실시예들에서, 단일 색상 구성요소는

루마 색상 구성요소 또는 녹색 구성요소이다. 일부 실시예들에서, 수정은 현재 블록의 색상 형식이 4:4:4 색상 형식인 경우 현재 블록의 모든 색상 구성요소들에 적용 가능하다.

[0368] 일부 실시예들에서, 광학 흐름 코딩 프로세스는 블록 내의 움직임이 광학 흐름으로 모델링되는 프로세스이다.

[0369] 일부 실시예들에서, 변환을 수행하는 단계는 비디오의 현재 블록에 기초하여 비트스트림 표현을 생성하는 단계를 포함한다. 일부 실시예들에서, 변환을 수행하는 단계는 비트스트림 표현으로부터 비디오의 현재 블록을 생성하는 단계를 포함한다.

[0370] 개시된 기술의 일부 실시예는 비디오 처리 도구 또는 모드를 활성화하기 위한 결정 또는 결론을 내리는 것을 포함한다. 일 예에서, 비디오 처리 도구 또는 모드가 활성화되면 인코더는 비디오 블록의 프로세싱에서 도구 또는 모드를 사용하거나 실행할 수 있지만 도구 또는 모드의 사용에 따라 결과 비트스트림을 반드시 수정해야하는 것은 아니다. 즉, 비디오 블록에서 비디오의 비트스트림 표현으로의 변환은 결정 또는 결론에 따라 활성화될 때 비디오 프로세싱 도구 또는 모드를 사용할 수 있다. 다른 예에서, 비디오 프로세싱 도구 또는 모드가 활성화될 때, 디코더는 비디오 프로세싱 도구 또는 모드에 기초하여 비트스트림이 수정되었다는 지식으로 비트스트림을 프로세싱할 것이다. 즉, 비디오의 비트스트림 표현에서 비디오의 블록으로의 변환은 결론 또는 결정에 기초하여 활성화된 비디오 프로세싱 도구 또는 모드를 사용하여 수행될 것이다.

[0371] 개시된 기술의 일부 실시예들은 비디오 프로세싱 도구 또는 모드를 비활성화하기 위한 결정 또는 결론을 내리는 것을 포함한다. 일 예로서, 비디오 프로세싱 도구 또는 모드가 비활성화되면 인코더는 비디오 블록을 비디오의 비트스트림 표현으로 변환하는 데 도구 또는 모드를 사용하지 않을 수 있다. 다른 예로서, 비디오 프로세싱 도구 또는 모드가 비활성화되면, 디코더는 결정 또는 결론에 기초하여 활성화된 비디오 프로세싱 도구 또는 모드를 사용하여 비트스트림이 수정되지 않았다는 지식으로 비트스트림을 프로세싱할 것이다.

[0372] 이 명세서에서 "비디오 프로세싱(처리)(video processing)"라는 용어는 비디오 인코딩, 비디오 디코딩, 비디오 압축, 또는 비디오 압축해제를 지칭할 수 있다. 예를 들어, 비디오 압축 알고리즘들이 비디오의 화소 표현으로부터 대응 비트스트림 표현으로의 변환 동안 적용될 수 있으며 그 역도 마찬가지이다. 현재 비디오 블록의 비트스트림 표현은 예를 들어 신택스에 정의된 바와 같이 비트스트림의 다른 위치들에 공존(co-located)하거나 분산된 비트들에 대응할 수 있다. 예를 들어 매크로블록(macroblock)은 변환 및 코딩된 오류 잔차 값(error residual value)들로, 또한 헤더 또는 비트스트림 내의 다른 필드들의 비트를 사용하여 인코딩될 수 있다. 또한, 변환 동안, 디코더는 상기 솔루션에서 설명된 바와 같이 결정에 기초하여 일부 필드들이 존재하거나 없을 수 있다는 생각 하에 비트스트림을 파싱할 수 있다. 유사하게, 인코더는 특정 신택스 필드가 포함되는 지 여부를 결정할 수 있고, 이에 따라 코딩된 표현으로부터 신택스 필드를 포함하거나 제외함으로써 코딩된 표현을 생성할 수 있다.

[0373] 이상에서, 현재 개시되는 본 개시된 기술의 특정한 실시예들이 예시의 목적으로 기재되었지만 발명의 범위를 벗어나지 않고도 다양한 변경들이 이뤄질 수 있음을 이해해야 할 것이다. 이에 따라, 현재 개시되는 기술은 첨부된 청구항들에 의해서가 아니면 한정되지 않는다.

[0374] 이 명세서에 기재된 주제의 구현과 기능적 작동들은, 이 명세서에 개시된 구조들과 그 구조적 등가물들, 또는 그 하나 이상의 조합들을 포함하여 여러 가지 시스템, 디지털 전자 회로, 또는 컴퓨터 소프트웨어, 펌웨어, 또는 하드웨어로 구현될 수 있다. 이 명세서에 기재된 주제의 구현과 기능적 작동들은 하나 이상의 컴퓨터 프로그램 제품들, 즉 데이터 처리 장치에서 실행되거나 그 작동을 제어하는 유형 및 비일시적인 컴퓨터 독출 가능한 매체 상에 인코딩된 컴퓨터 프로그램 명령의 하나 이상의 모듈들로 구현될 수 있다. 컴퓨터 독출 가능한 매체는 기계 독출 가능한 저장 장치, 기계 독출 가능한 저장 기판, 메모리 장치, 기계 독출 가능한 전파 신호(propagated signal)를 일으키는(effecting) 물질의 조성(composition of matter), 또는 그 하나 이상의 조합이 될 수 있다. "데이터 처리 유닛(data processing unit)" 또는 "데이터 처리 장치(data processing apparatus)"라는 용어는 예를 들어 프로그래밍 가능한 프로세서, 컴퓨터, 또는 복수의 프로세서들 또는 컴퓨터들을 포함하여 데이터를 처리하는 모든 장치, 기기, 또는 기계들을 포괄한다. 장치는 하드웨어에 추가하여 예를 들어 프로세서 펌웨어를 구성하는 코드, 프로토콜 스택(protocol stack), 데이터베이스 관리 시스템, 운영체제, 또는 그 하나 이상의 조합 등 문제의 컴퓨터 프로그램에 대한 실행 환경을 생성하는 코드를 포함할 수 있다.

[0375] 컴퓨터 프로그램(프로그램, 소프트웨어, 소프트웨어 애플리케이션, 스크립트, 또는 코드라고도 알려져 있음)은 컴파일되거나 해석된 언어를 포함하는 임의의 형태의 프로그래밍 언어로 작성될 수 있고, 이것은 독립형 프로그램과 같은 또는 컴퓨팅 환경에서 사용되기에 적합한 모듈, 성분, 서브루틴, 또는 다른 유닛과 같은 임의의 형태

로 전개될 수 있다. 컴퓨터 프로그램은 반드시 파일 시스템 내의 파일에 대응하는 것은 아니다. 프로그램은 다른 프로그램 또는 데이터를 보유하는 파일(예를 들어, 마크업 언어 문서에 저장된 하나 이상의 스크립트) 내에, 문제가 되는 프로그램 전용의 단일 파일 내에, 또는 여러 개의 조율된 파일들)예를 들어, 하나 이상의 모듈, 서브-프로그램, 또는 코드의 일부를 저장하는 파일) 내에 저장될 수 있다. 컴퓨터 프로그램은 하나의 컴퓨터 상에서 또는 하나의 사이트에 위치되거나 여러 사이트들에 걸쳐서 분산되고 통신 네트워크에 의하여 상호 연결되는 여러 컴퓨터들 상에서 실행되도록 구현될 수 있다.

[0376] 본 명세서에서 설명되는 프로세스 및 로직 흐름은 입력 데이터 상에 작용하여 출력을 생성함으로써 기능을 수행하기 위한 하나 이상의 컴퓨터 프로그램을 실행하는 하나 이상의 프로그래밍가능한 프로세서에 의하여 수행될 수 있다. 또한 프로세스 및 로직 흐름은 특수 목적 로직 회로부, 예를 들어 FPGA(field programmable gate array), 또는 ASIC(application-상세 integrated circuit)에 의하여 수행될 수 있으며, 장치는 또한 이러한 것으로서 구현될 수 있다.

[0377] 컴퓨터 프로그램을 실행하기에 적합한 프로세서는, 예를 들어 범용 및 주문형 마이크로프로세서 양자 모두, 및 임의의 종류의 디지털 컴퓨터의 임의의 하나 이상의 프로세서를 포함한다. 일반적으로, 프로세서는 판독-전용 메모리 또는 랜덤 액세스 메모리 또는 이들 모두로부터 명령 및 데이터를 수신할 것이다. 컴퓨터의 본질적인 요소는 명령을 수행하기 위한 프로세서 및 명령 및 데이터를 저장하기 위한 하나 이상의 메모리 디바이스이다. 일반적으로, 컴퓨터는 데이터를 저장하기 위한 하나 이상의 대용량 저장 디바이스, 예를 들어 자기적, 자기-광학적 디스크, 또는 광학적 디스크를 더욱 포함하거나, 이것으로부터 데이터를 수신하거나 이것으로 데이터를 전송하거나, 또는 이들 모두를 하기 위하여 이것에 동작가능하도록 커플링될 것이다. 그러나, 컴퓨터는 이러한 디바이스를 가져야 하는 것은 아니다. 컴퓨터 프로그램 명령 및 데이터를 저장하기에 적합한 컴퓨터 판독가능 매체는, 예를 들자면 반도체 메모리 디바이스, 예를 들어 EPROM, EEPROM, 및 플래시 메모리 디바이스; 자기적 디스크, 예를 들어 내부 하드 디스크 또는 착탈식 디스크; 자기-광학적 디스크; 및 CD-ROM 및 DVD-ROM 디스크를 포함하는 비-휘발성 메모리, 미디어 및 메모리 디바이스의 모든 형태를 포함한다. 프로세서 및 메모리는 특수 목적 로직 회로부에 의하여 보충되거나 그 내부에 통합될 수 있다.

[0378] 도면과 함께 명세서는 단지 예시적인 것으로 간주되어야 하며, 여기서 예시적인 것은 예시를 의미합니다. 본 명세서에 사용된 바와 같이, 단수 형태는 문맥이 명백하게 달리 나타내지 않는 한 복수 형태도 포함하는 것으로 의도된다. 추가적으로, "또는(혹은)"의 사용은 문맥이 명백하게 달리 나타내지 않는 한 "및/또는"을 포함하도록 의도된다.

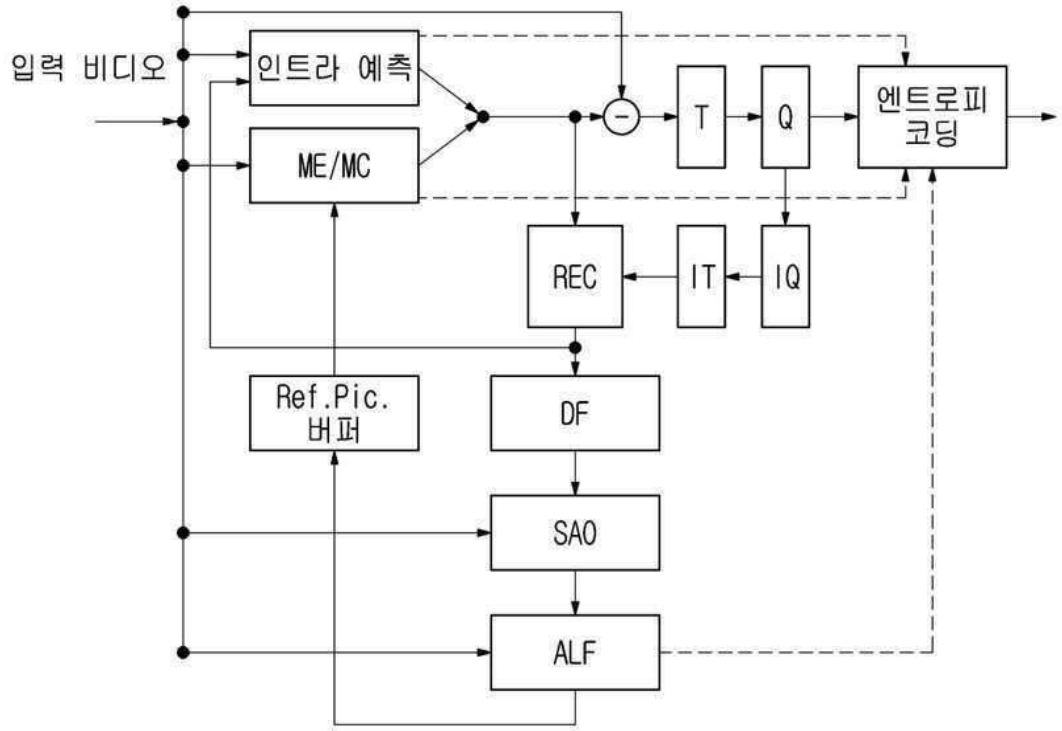
[0379] 본 특허 명세서가 많은 특징을 포함하고 있지만, 이들은 임의의 기술 요지 또는 청구될 내용의 범위를 한정하는 것으로 해석되어서는 안 되고, 오히려 특정 기법의 특정 실시형태들에 특유할 수 있는 특징들의 설명으로서 해석되어야 한다. 본 특허 명세서에서 별개의 실시형태들의 콘텍스트에서 설명되는 일부 특징들은, 단일 실시형태 내에 조합되어 구현될 수도 있다. 반대로, 단일 실시예의 콘텍스트에서 설명되는 다양한 특징들은 또한 따로따로 다수의 실시예들에서 또는 임의의 적합한 서브컴비네이션에서 구현될 수 있다. 더구나, 비록 특징들이 특정한 조합들로 작용하는 것으로 위에서 설명될 수도 있고 그와 같이 처음에 청구된 경우에도, 청구된 조합들로부터의 하나 이상의 특징들은 일부 경우들에서 조합으로부터 삭제될 수 있고, 청구된 조합은 서브컴비네이션 또는 서브컴비네이션의 변형예를 위한 것일 수도 있다.

[0380] 이와 유사하게, 동작들이 도면들에서 특정한 순서로 묘사되어 있으나, 원하는 결과들을 획득하기 위하여 이러한 동작들이 도시된 특정한 순서로 또는 순차적인 순서로 수행되어야 하거나, 또는 도시된 모든 동작들이 수행되어야 한다고 요구하는 것으로 이해되어서는 안 된다. 더욱이, 본 특허 명세서에서 설명되는 실시형태들 내의 다양한 시스템 성분들을 분리하는 것은, 모든 실시형태에서 이러한 분리를 요구하는 것으로 이해되어서는 안 된다.

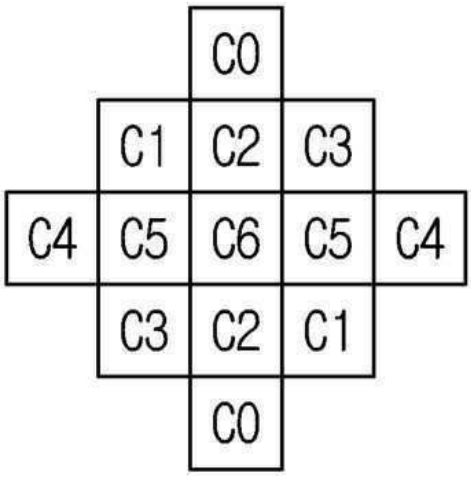
[0381] 몇 가지의 구현형태와 예들만이 설명되고, 다른 구현형태, 개선형태, 및 변형예들이 본 특허 명세서에서 설명되고 예시된 것에 기반하여 이루어질 수 있다.

도면

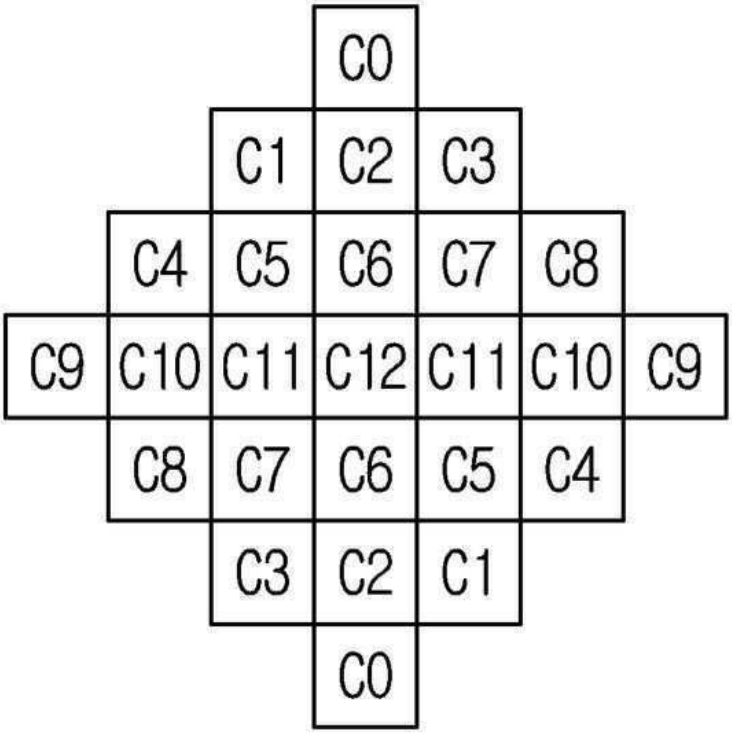
도면1


입력 비디오
인트라 예측
ME/MC
T
Q
엔트로피 코딩
REC
IT
IQ
Ref.Pic. 버퍼
DF
SAO
ALF

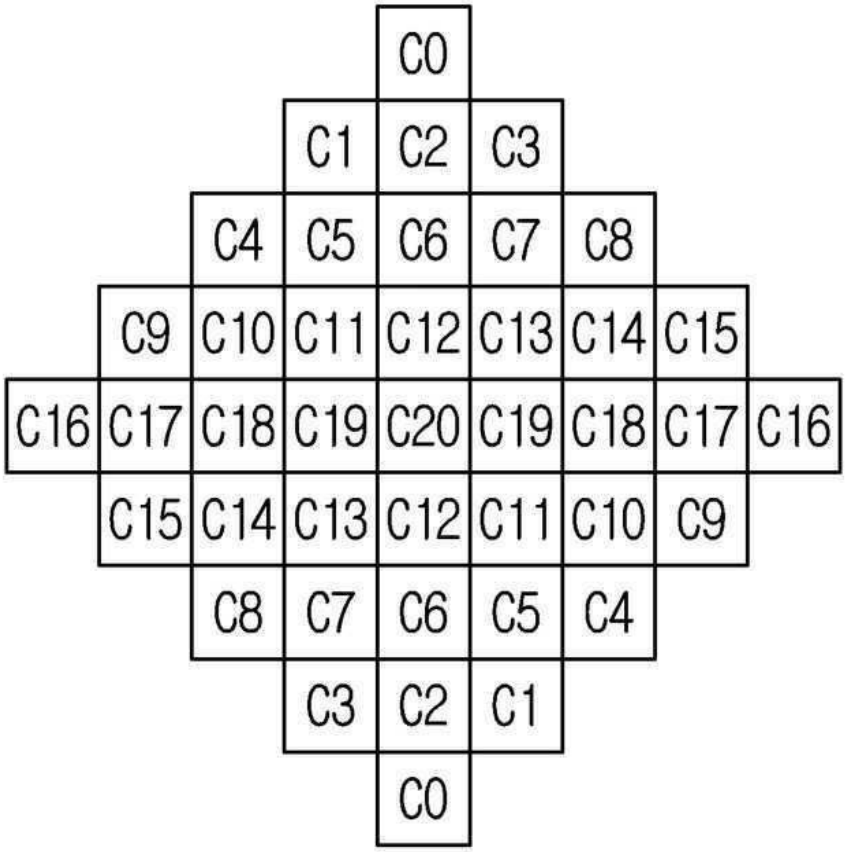
도면2a


C0
C1
C2
C3
C4
C5
C6
C5
C4
C3
C2
C1
C0

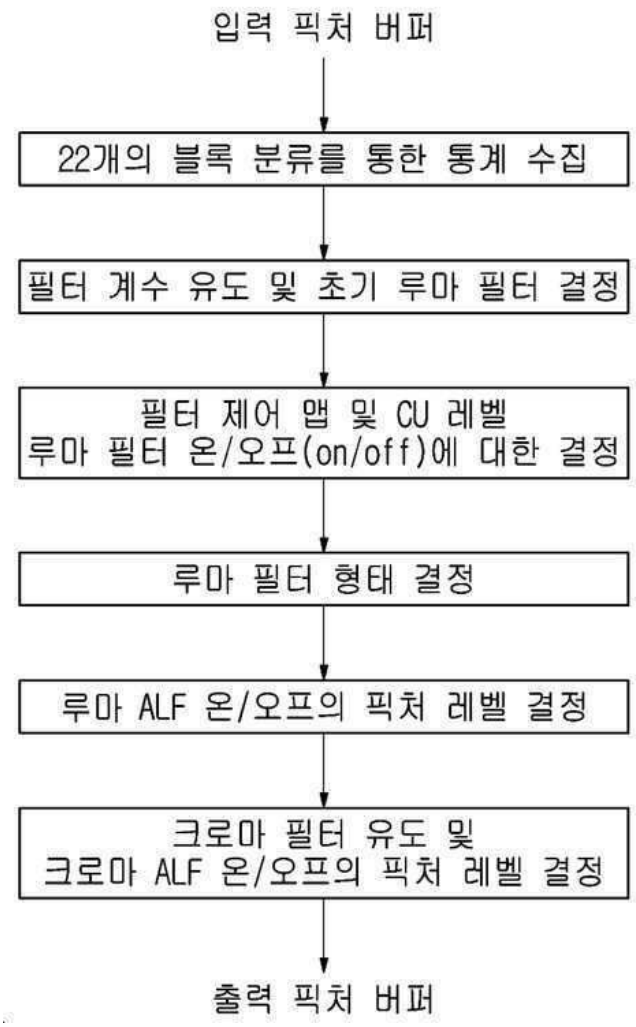
도면2b


C0
C1 C2 C3
C4 C5 C6 C7 C8
C9 C10 C11 C12 C11 C10 C9
C8 C7 C6 C5 C4
C3 C2 C1
C0

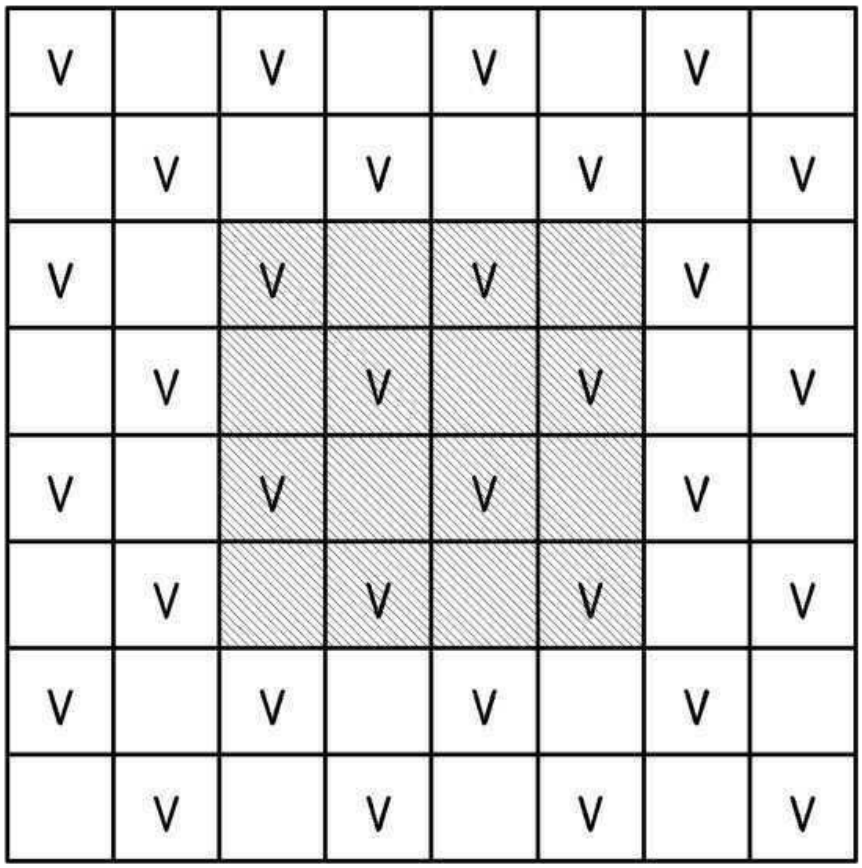
도면2c


C0
C1 C2 C3
C4 C5 C6 C7 C8
C9 C10 C11 C12 C13 C14 C15
C16 C17 C18 C19 C20 C19 C18 C17 C16
C15 C14 C13 C12 C11 C10 C9
C8 C7 C6 C5 C4
C3 C2 C1
C0

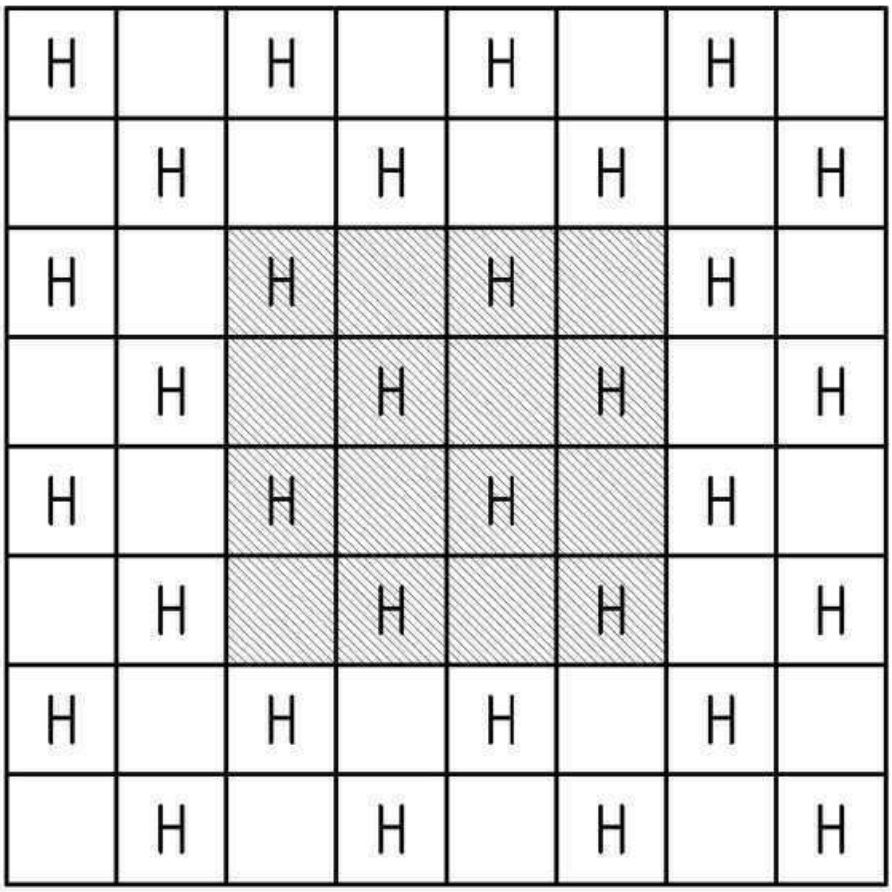
도면3


입력 픽처 버퍼
22개의 블록 분류를 통한 통계 수집
필터 계수 유도 및 초기 루마 필터 결정
필터 제어 맵 및 CU 레벨
루마 필터 온/오프(on/off)에 대한 결정
루마 필터 형태 결정
루마 ALF 온/오프의 픽처 레벨 결정
크로마 필터 유도 및
크로마 ALF 온/오프의 픽처 레벨 결정
출력 픽처 버퍼

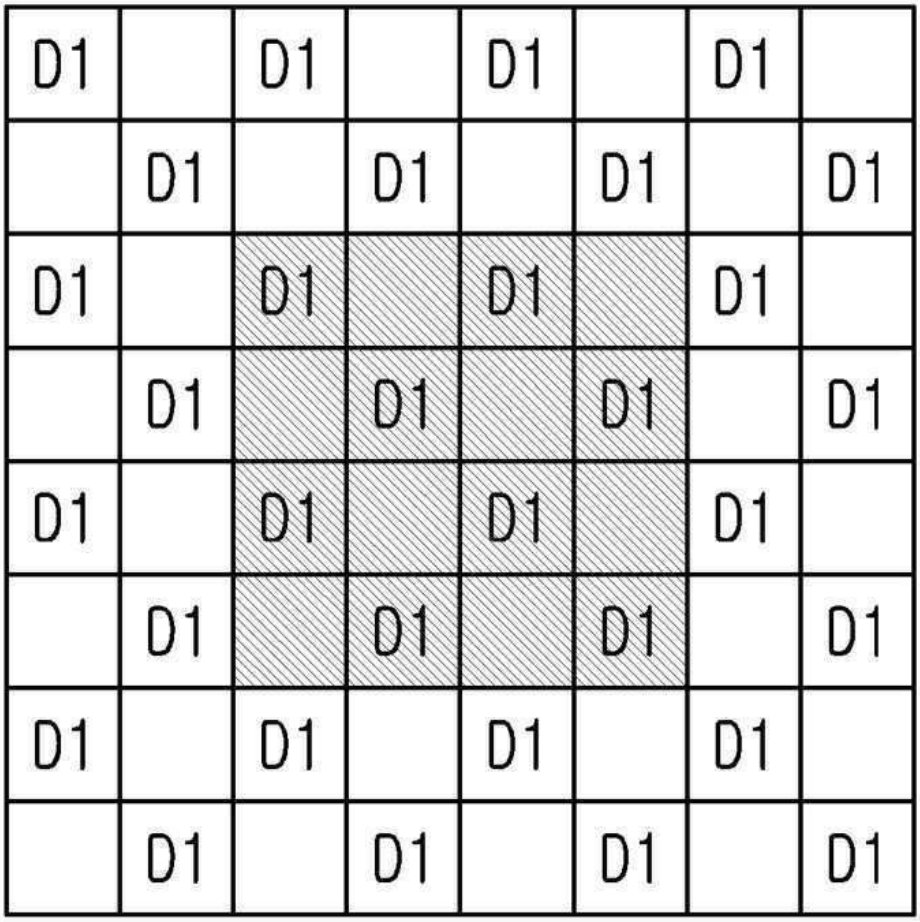
도면4a



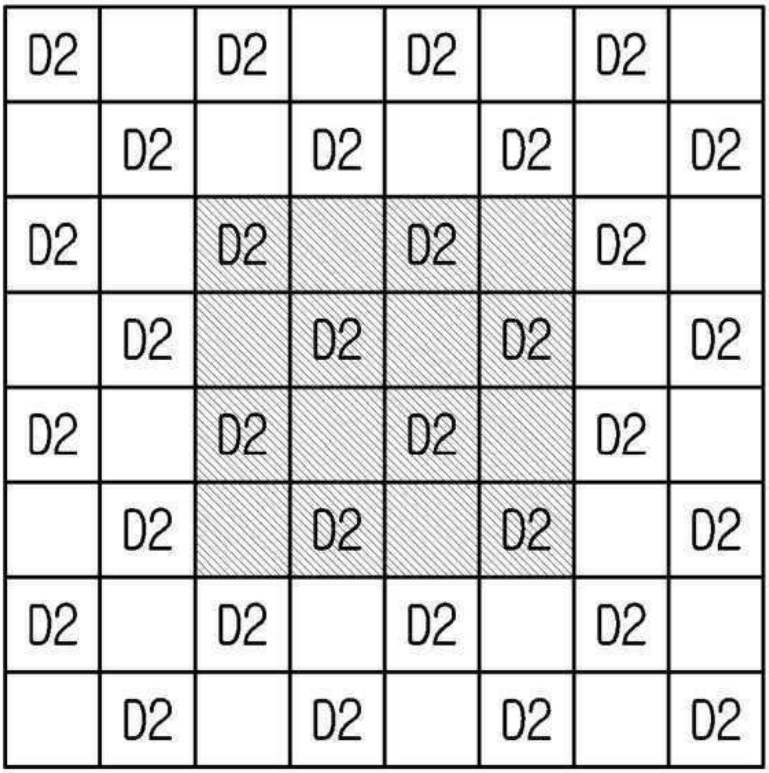
도면4b



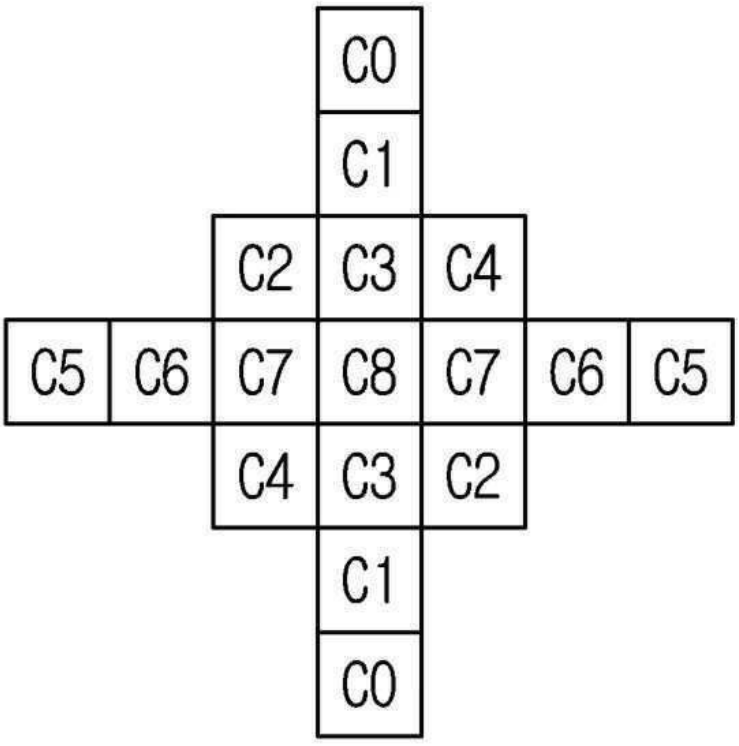
도면4c


D1
D1
D1
D1
D1
D1
D1
D1
D1
D1
D1
D1
D1
D1
D1
D1
D1
D1
D1
D1
D1
D1
D1
D1
D1
D1
D1
D1
D1
D1
D1
D1

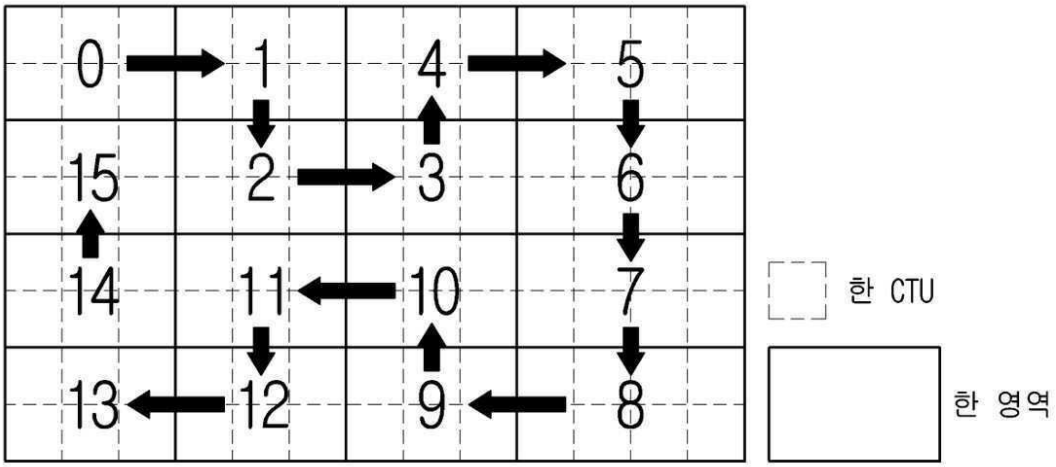
도면4d


D2
D2
D2
D2
D2
D2
D2
D2
D2
D2
D2
D2
D2
D2
D2
D2
D2
D2
D2
D2
D2
D2
D2
D2
D2
D2
D2
D2
D2
D2
D2
D2

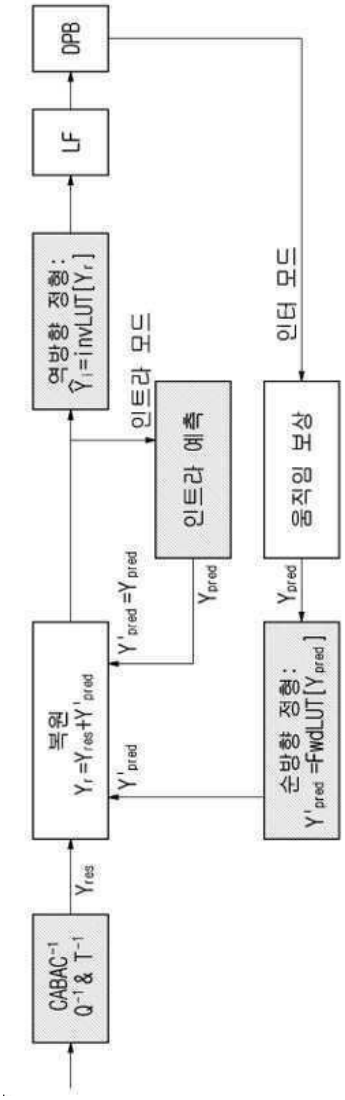
도면5


C0
C1
C2
C3
C4
C5
C6
C7
C8
C7
C6
C5
C4
C3
C2
C1
C0

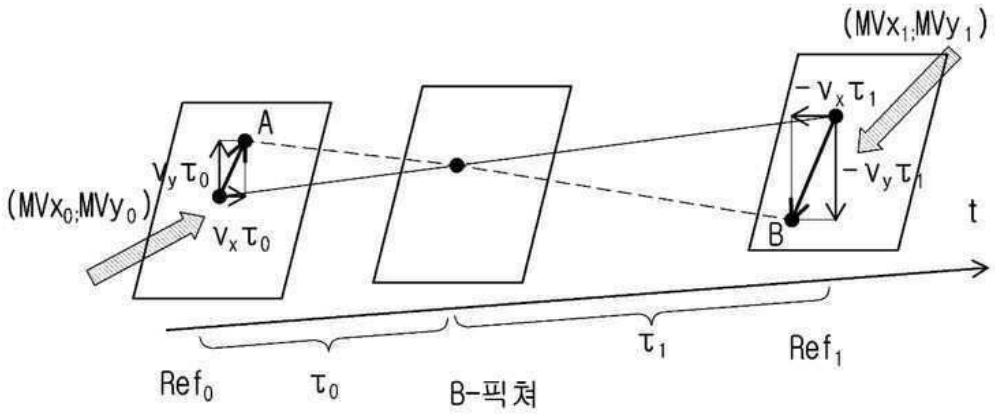
도면6


0
1
4
5
15
2
3
6
14
11
10
7
13
12
9
8
한 CTU
한 영역

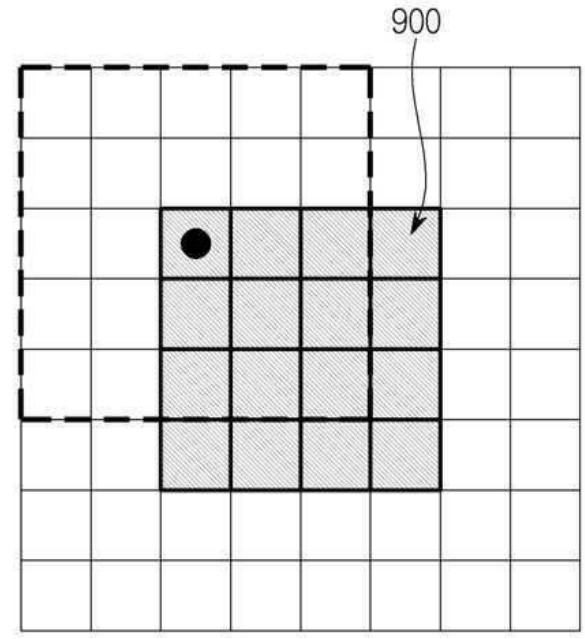
도면7


DPB
LF
역방향 정형:
$\hat{Y}_r = invLUT[Y_r]$
인트라 모드
인터 모드
인트라 예측
움직임 보상
Y_{pred}
$Y'_{pred} = Y_{pred}$
복원
$Y_r = Y_{res} + Y'_{pred}$
Y'_{pred}
순방향 정형:
$Y'_{pred} = FwdLUT[Y_{pred}]$
Y_{res}
CABAC^{-1}
Q^{-1} & T^{-1}

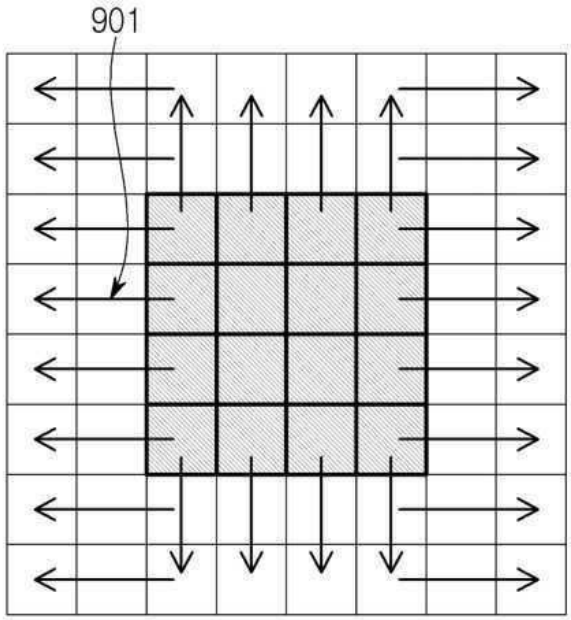
도면8


(MVx1;MVy1)
-VxT1
-VyT1
A
VyT0
(MVx0;MVy0)
VxT0
B
t
T1
Ref1
Ref0
T0
B-픽쳐

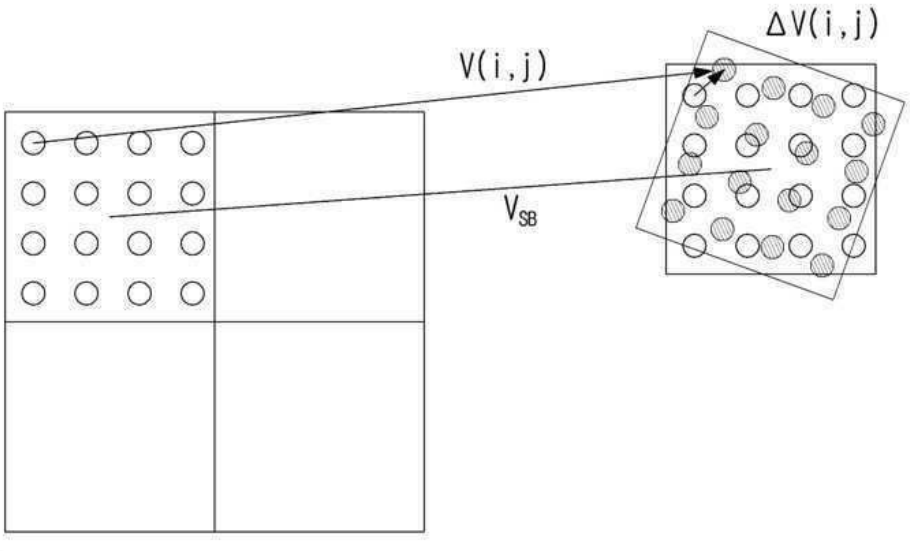
도면9a


900

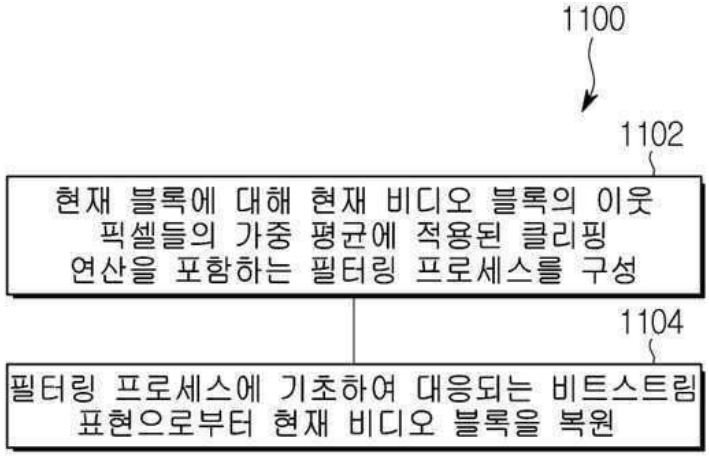
도면9b



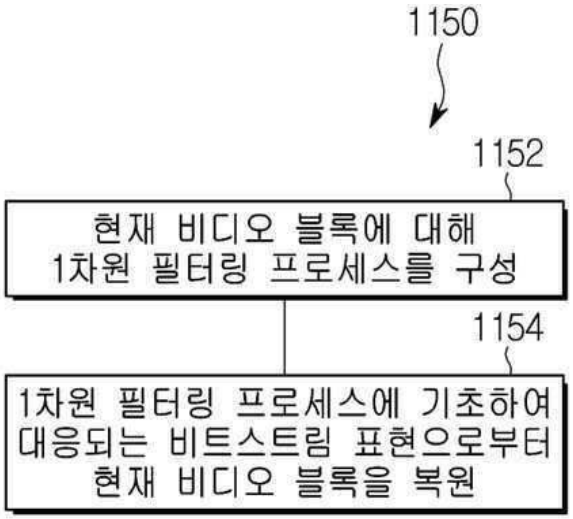
도면10



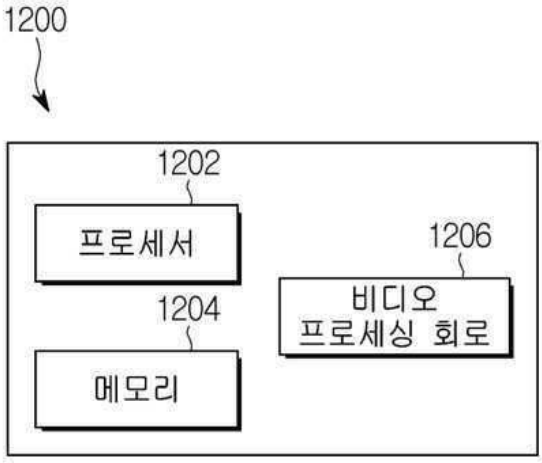
도면11a



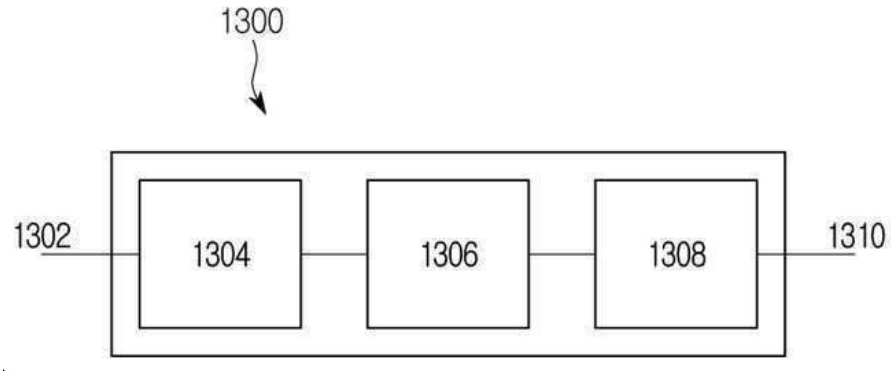
도면11b



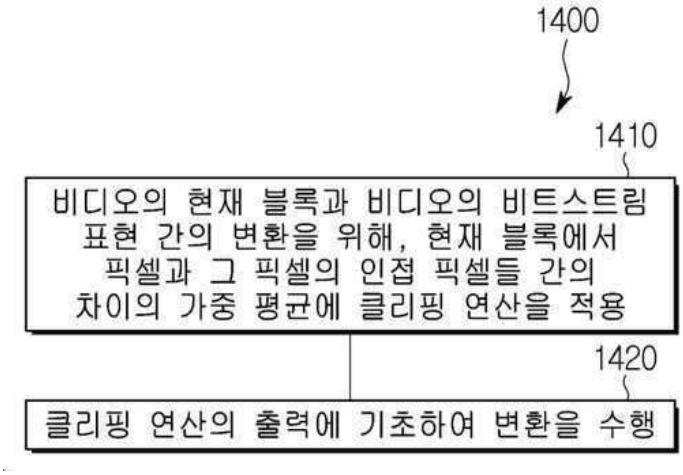
도면12



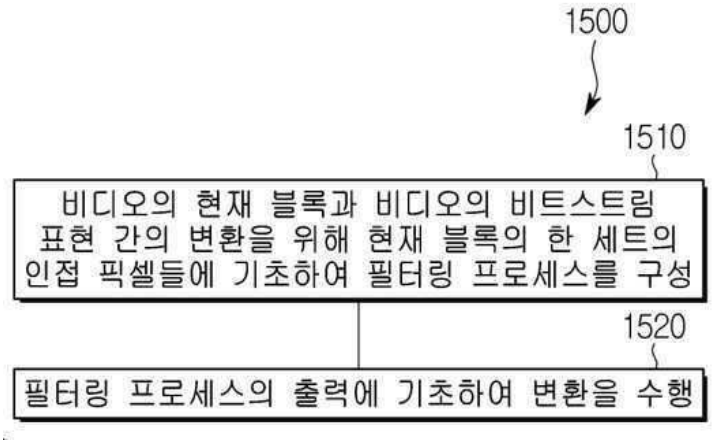
도면13



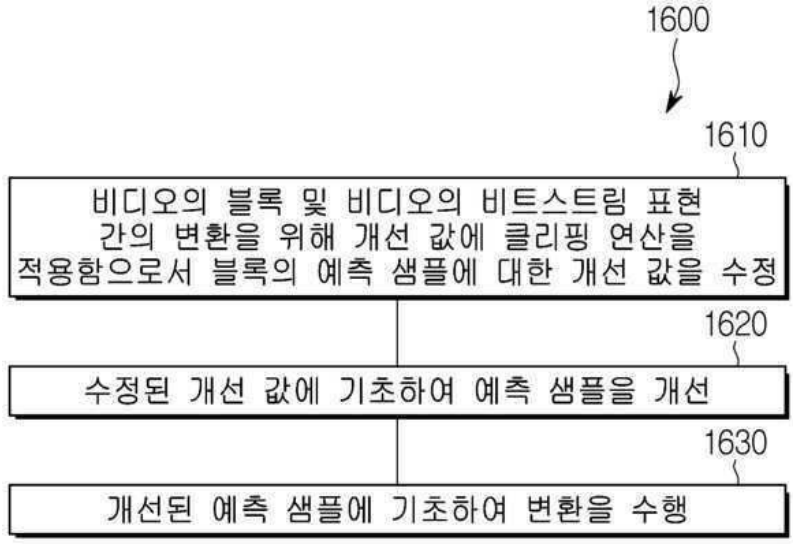
도면14



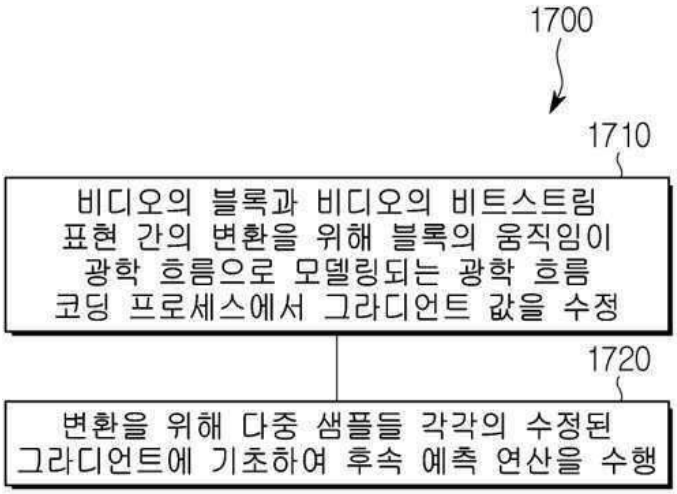
도면15



도면16


1600
1610
비디오의 블록 및 비디오의 비트스트림 표현 간의 변환을 위해 개선 값에 클리핑 연산을 적용함으로서 블록의 예측 샘플에 대한 개선 값을 수정
1620
수정된 개선 값에 기초하여 예측 샘플을 개선
1630
개선된 예측 샘플에 기초하여 변환을 수행

도면17


1700
1710
비디오의 블록과 비디오의 비트스트림 표현 간의 변환을 위해 블록의 움직임이 광학 흐름으로 모델링되는 광학 흐름 코딩 프로세스에서 그라디언트 값을 수정
1720
변환을 위해 다중 샘플들 각각의 수정된 그라디언트에 기초하여 후속 예측 연산을 수행