

(19) 대한민국특허청(KR)  
(12) 공개특허공보(A)(11) 공개번호 10-2024-0010576  
(43) 공개일자 2024년01월23일

- (51) 국제특허분류(Int. Cl.)  
H04N 19/513 (2014.01) H04N 19/105 (2014.01)  
H04N 19/132 (2014.01) H04N 19/157 (2014.01)  
H04N 19/176 (2014.01) H04N 19/567 (2014.01)
- (52) CPC특허분류  
H04N 19/513 (2015.01)  
H04N 19/105 (2015.01)
- (21) 출원번호 10-2024-7000656(분할)
- (22) 출원일자(국제) 2020년01월10일  
심사청구일자 없음
- (62) 원출원 특허 10-2021-7020500  
원출원일자(국제) 2020년01월10일  
심사청구일자 2022년12월22일
- (85) 번역문제출일자 2024년01월08일
- (86) 국제출원번호 PCT/CN2020/071332
- (87) 국제공개번호 WO 2020/143741  
국제공개일자 2020년07월16일
- (30) 우선권주장  
PCT/CN2019/071214 2019년01월10일 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  
장, 리  
미국 90066 캘리포니아주 로스엔젤레스 스위트 넘버 137 6층 웨스트 제퍼슨 블러바드 12655  
장, 카이  
미국 90066 캘리포니아주 로스엔젤레스 스위트 넘버 137 6층 웨스트 제퍼슨 블러바드 12655  
(뒷면에 계속)
- (74) 대리인  
양영준, 임규빈, 백만기

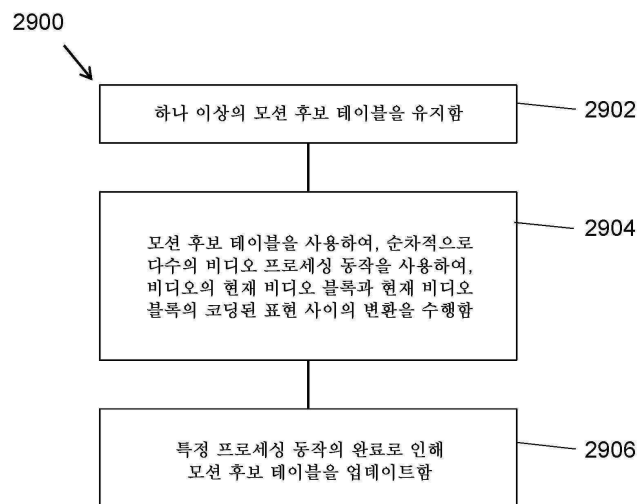
전체 청구항 수 : 총 18 항

## (54) 발명의 명칭 LUT 업데이트의 호출

## (57) 요약

하나 이상의 모션 후보 테이블을 유지하는 단계; 모션 후보 테이블을 사용하여, 순차적으로 다수의 비디오 프로세싱 동작을 사용하여, 비디오의 현재 비디오 블록과 현재 비디오 블록의 코딩된 표현 사이의 변환을 수행하는 단계; 및 특정 프로세싱 동작의 완료로 인해 모션 후보 테이블을 업데이트하는 단계를 포함하는 비디오 프로세싱 방법이 제공된다.

대표도 - 도29



(52) CPC특허분류

*H04N 19/132* (2015.01)

*H04N 19/157* (2015.01)

*H04N 19/176* (2015.01)

*H04N 19/567* (2015.01)

(72) 발명자

류, 홍빈

중국 100080 베이징 하이덴 디스트릭트 즈춘 로드  
넘버 63 차이나 세틀라이트 커뮤니케이션즈 타워  
진르터우타오 포스트 오피스

왕, 웨

중국 100080 베이징 하이덴 디스트릭트 즈춘 로드  
넘버 63 차이나 세틀라이트 커뮤니케이션즈 타워  
진르터우타오 포스트 오피스

## 명세서

### 청구범위

#### 청구항 1

비디오 프로세싱 방법으로서,

하나 이상의 모션 후보 테이블을 유지하는 단계 - 각각의 모션 후보 테이블은 하나 이상의 모션 후보를 포함하고, 상기 하나 이상의 모션 후보 중 적어도 하나는 적어도 하나의 이전 비디오 블록으로부터 도출되고, 상기 모션 후보 테이블 내의 상기 모션 후보들의 배열은 상기 모션 후보 테이블 내의 상기 모션 후보들의 추가의 시퀀스에 기초함 -;

제1 비디오 블록에 대한 모션 후보 리스트를 도출하기 위해, 모션 후보 리스트 도출 프로세스를 수행하는 단계;

상기 모션 후보 리스트에 기초하여, 상기 제1 비디오 블록에 대한 모션 정보를 도출하는 단계;

코딩 틀을 사용하여 상기 모션 정보를 리파이닝(refining)하는 단계;

상기 리파이닝된 모션 정보에 기초하여 상기 제1 비디오 블록과 상기 제1 비디오 블록을 포함하는 비디오의 비트스트림 사이의 변환을 수행하는 단계; 및

상기 하나 이상의 모션 후보 테이블 중의 모션 후보 테이블을 업데이트하는 단계

를 포함하고,

상기 모션 후보 테이블을 업데이트하기 위해, 상기 리파이닝된 모션 정보를 사용할지 또는 리파인먼트(refinement)를 이용하지 않은 상기 모션 정보를 사용할지는 상기 제1 비디오 블록과 연관된 상기 코딩 틀에 기초하고,

상기 업데이트된 모션 후보 테이블 내의 하나 이상의 모션 후보는 후속 블록에 대한 모션 후보 리스트를 구성하는 동안 순서대로 체크되는,

방법.

#### 청구항 2

제1항에 있어서,

상기 코딩 틀이 제1 리파인먼트 틀인 것에 대한 응답으로, 상기 모션 후보 테이블은 리파인먼트를 이용하지 않은 상기 모션 정보를 사용함으로써 업데이트되고, 상기 제1 리파인먼트 틀에서, 상기 리파이닝된 모션 정보는 상기 제1 비디오 블록의 참조 샘플들에 대응하는 절대 차이 합(sum of absolute differences)(SAD)의 계산에 기초하여 도출되는, 방법.

#### 청구항 3

제1항에 있어서,

상기 코딩 틀은 모션 벡터 차이와의 병합 틀이고, 상기 모션 후보 테이블은 상기 리파이닝된 모션 정보를 사용함으로써 업데이트되는, 방법.

#### 청구항 4

제1항 내지 제3항 중 어느 한 항에 있어서,

상기 제1 비디오 블록은 병합 모드로 코딩되는, 방법.

#### 청구항 5

제1항 내지 제4항 중 어느 한 항에 있어서,

상기 모션 후보 테이블은 적어도 하나의 특정 모드에 속하는 상기 제1 비디오 블록의 코딩 모드에 기초하여 업

데이트되는 것으로 결정되는, 방법.

#### 청구항 6

제5항에 있어서,

상기 적어도 하나의 특정 모드는 모션 벡터 예측 모드 또는 병합 모드 중 적어도 하나를 포함하는, 방법.

#### 청구항 7

제5항 또는 제6항에 있어서,

상기 적어도 하나의 특정 모드는 서브-블록 병합 모드 또는 삼각 병합 모드 중 적어도 하나를 배제하고, 상기 모션 후보 테이블은 상기 서브-블록 병합 모드 또는 상기 삼각 병합 모드를 사용하여 블록들로부터 도출된 모션 정보를 사용하여 업데이트되지 않는, 방법.

#### 청구항 8

제1항 내지 제7항 중 어느 한 항에 있어서,

상기 모션 후보 테이블은 이력 모션 벡터 예측(history motion vector prediction)(HMVP) 테이블인, 방법.

#### 청구항 9

제1항 내지 제8항 중 어느 한 항에 있어서,

상기 업데이트하는 단계는,

프루닝(pruning) 동작을 사용하여 상기 모션 후보 테이블을 업데이트하는 단계를 포함하고, 상기 프루닝 동작은, 상기 모션 후보 테이블로부터 적어도 하나의 모션 후보를 삭제할지 여부를 결정하기 위해, 상기 모션 후보 테이블에 추가될 새로운 모션 후보를 상기 모션 후보 테이블 내의 상기 적어도 하나의 모션 후보와 비교하는 단계를 포함하는, 방법.

#### 청구항 10

제1항 내지 제9항 중 어느 한 항에 있어서,

상기 코딩 틀이 조명 보상(illumination compensation) 틀 또는 중첩 블록 모션 보상(overlapped block motion compensation) 틀인 경우, 상기 모션 후보 테이블은 리파인먼트를 이용하지 않은 상기 모션 정보를 사용하여 업데이트되는, 방법.

#### 청구항 11

제1항 내지 제10항 중 어느 한 항에 있어서,

상기 모션 후보 리스트로부터 도출된 모션 정보와 연관된 새로운 후보가 상기 모션 후보 테이블로부터 상기 모션 후보 리스트 내에 추가된 모션 후보인 경우, 상기 모션 후보 테이블은 프루닝 동작을 이용하지 않고 재순서화되는, 방법.

#### 청구항 12

제1항 내지 제11항 중 어느 한 항에 있어서,

상기 모션 후보 리스트로부터 도출된 모션 정보와 연관된 새로운 후보가 상기 제1 비디오 블록을 포함하는 현재 픽처 내의 공간적 이웃 블록, 또는 상기 현재 픽처와 상이한 픽처 내의 시간적 블록과 관련된 모션 후보인 경우, 상기 모션 후보 테이블은 프루닝 동작을 이용하여 업데이트되는, 방법.

#### 청구항 13

제1항 내지 제12항 중 어느 한 항에 있어서,

모션 벡터 예측 모드로 코딩된 제2 비디오 블록에 대한 모션 후보 리스트를 도출하기 위해, 모션 후보 리스트 도출 프로세스를 수행하는 단계;



상기 제2 비디오 블록에 대해 도출된 상기 모션 후보 리스트에 기초하여, 상기 제2 비디오 블록에 대한 모션 정보를 도출하는 단계;

상기 제2 블록에 대해 도출된 상기 모션 정보에 기초하여, 상기 제2 비디오 블록과 상기 제2 비디오 블록을 포함하는 비디오의 비트스트림 사이의 변환을 수행하는 단계; 및

리파인먼트를 이용하지 않고 상기 제2 블록에 대해 도출된 상기 모션 정보를 사용하여 상기 모션 후보 테이블을 업데이트하는 단계

를 더 포함하는, 방법.

#### 청구항 14

제1항 내지 제12항 중 어느 한 항에 있어서,

상기 변환을 수행하는 단계는 상기 제1 비디오 블록에 기초하여 상기 비트스트림을 생성하는 것을 포함하는, 방법.

#### 청구항 15

제1항 내지 제12항 중 어느 한 항에 있어서,

상기 변환을 수행하는 단계는 상기 비트스트림으로부터 상기 제1 비디오 블록을 생성하는 것을 포함하는, 방법.

#### 청구항 16

비디오 데이터를 프로세싱하기 위한 장치로서,

프로세서, 및 명령어들을 갖는 비-일시적 메모리를 포함하고,

상기 명령어들은, 상기 프로세서에 의한 실행 시에, 상기 프로세서로 하여금,

하나 이상의 모션 후보 테이블을 유지하게 하고 — 각각의 모션 후보 테이블은 하나 이상의 모션 후보를 포함하고, 상기 하나 이상의 모션 후보 중 적어도 하나는 적어도 하나의 이전 비디오 블록으로부터 도출되고, 상기 모션 후보 테이블 내의 상기 모션 후보들의 배열은 상기 모션 후보 테이블 내의 상기 모션 후보들의 추가의 시퀀스에 기초함 —,

제1 비디오 블록에 대한 모션 후보 리스트를 도출하기 위해, 모션 후보 리스트 도출 프로세스를 수행하게 하고,

상기 모션 후보 리스트에 기초하여, 상기 제1 비디오 블록에 대한 모션 정보를 도출하게 하고,

코딩 툴을 사용하여 상기 모션 정보를 리파이닝하게 하고,

상기 리파이닝된 모션 정보에 기초하여 상기 제1 비디오 블록과 상기 제1 비디오 블록을 포함하는 비디오의 비트스트림 사이의 변환을 수행하게 하고,

상기 하나 이상의 모션 후보 테이블 중의 모션 후보 테이블을 업데이트하게 하고,

상기 모션 후보 테이블을 업데이트하기 위해, 상기 리파이닝된 모션 정보를 사용하지 또는 리파인먼트(refinement)를 이용하지 않은 상기 모션 정보를 사용할지는 상기 제1 비디오 블록과 연관된 상기 코딩 툴에 기초하고,

상기 업데이트된 모션 후보 테이블 내의 하나 이상의 모션 후보는 후속 블록에 대한 모션 후보 리스트를 구성하는 동안 순서대로 체크되는,

장치.

#### 청구항 17

명령어들을 저장한 비-일시적 컴퓨터 판독가능 저장 매체로서,

상기 명령어들은 프로세서로 하여금,

하나 이상의 모션 후보 테이블을 유지하게 하고 — 각각의 모션 후보 테이블은 하나 이상의 모션 후보를 포함하고, 상기 하나 이상의 모션 후보 중 적어도 하나는 적어도 하나의 이전 비디오 블록으로부터 도출되고, 상기 모

선 후보 테이블 내의 상기 모션 후보들의 배열은 상기 모션 후보 테이블 내로의 상기 모션 후보들의 추가의 시퀀스에 기초함 - ,

제1 비디오 블록에 대한 모션 후보 리스트를 도출하기 위해, 모션 후보 리스트 도출 프로세스를 수행하게 하고, 상기 모션 후보 리스트에 기초하여, 상기 제1 비디오 블록에 대한 모션 정보를 도출하게 하고,

코딩 툴을 사용하여 상기 모션 정보를 리파이닝하게 하고,

상기 리파이닝된 모션 정보에 기초하여 상기 제1 비디오 블록과 상기 제1 비디오 블록을 포함하는 비디오의 비트스트림 사이의 변환을 수행하게 하고,

상기 하나 이상의 모션 후보 테이블 중의 모션 후보 테이블을 업데이트하게 하고,

상기 모션 후보 테이블을 업데이트하기 위해, 상기 리파이닝된 모션 정보를 사용할지 또는 리파인먼트(refinement)를 이용하지 않은 상기 모션 정보를 사용할지는 상기 제1 비디오 블록과 연관된 상기 코딩 툴에 기초하고,

상기 업데이트된 모션 후보 테이블 내의 하나 이상의 모션 후보는 후속 블록에 대한 모션 후보 리스트를 구성하는 동안 순서대로 체크되는,

비-일시적 컴퓨터 판독가능 저장 매체.

### 청구항 18

비디오 프로세싱 장치에 의해 수행되는 방법에 의해 생성되는 비디오의 비트스트림을 저장한 비-일시적 컴퓨터 판독가능 기록 매체로서,

상기 방법은,

하나 이상의 모션 후보 테이블을 유지하는 단계 - 각각의 모션 후보 테이블은 하나 이상의 모션 후보를 포함하고, 상기 하나 이상의 모션 후보 중 적어도 하나는 적어도 하나의 이전 비디오 블록으로부터 도출되고, 상기 모션 후보 테이블 내의 상기 모션 후보들의 배열은 상기 모션 후보 테이블 내로의 상기 모션 후보들의 추가의 시퀀스에 기초함 - ;

제1 비디오 블록에 대한 모션 후보 리스트를 도출하기 위해, 모션 후보 리스트 도출 프로세스를 수행하는 단계;

상기 모션 후보 리스트에 기초하여, 상기 제1 비디오 블록에 대한 모션 정보를 도출하는 단계;

코딩 툴을 사용하여 상기 모션 정보를 리파이닝하는 단계;

상기 리파이닝된 모션 정보에 기초하여 상기 비트스트림을 생성하는 단계; 및

상기 하나 이상의 모션 후보 테이블 중의 모션 후보 테이블을 업데이트하는 단계

를 포함하고,

상기 모션 후보 테이블을 업데이트하기 위해, 상기 리파이닝된 모션 정보를 사용할지 또는 리파인먼트(refinement)를 이용하지 않은 상기 모션 정보를 사용할지는 상기 제1 비디오 블록과 연관된 상기 코딩 툴에 기초하고,

상기 업데이트된 모션 후보 테이블 내의 하나 이상의 모션 후보는 후속 블록에 대한 모션 후보 리스트를 구성하는 동안 순서대로 체크되는,

비-일시적 컴퓨터 판독가능 기록 매체.

### 발명의 설명

### 기술 분야

[0001]

관련 출원들에 대한 상호 참조

[0002]

본 출원은 2019년 1월 10일자로 출원된 국제 특허 출원 PCT/CN2019/071214호의 우선권 및 이익을 주장하는 2020년 1월 10일자로 출원된 국제 특허 출원 PCT/CN2020/071332호에 기초하는 한국 특허 출원 10-2021-7020500호의

분할 출원이다. 모든 전술된 특허 출원들은 이로써 그 전체가 참조로 포함된다.

[0003] 기술분야

[0004] 본 특허 문서는 비디오 코딩 기법들, 디바이스들, 및 시스템들에 관한 것이다.

## 배경 기술

[0005] 비디오 압축의 진보들에도 불구하고, 디지털 비디오는 여전히, 인터넷 및 다른 디지털 통신 네트워크들 상에서 가장 큰 대역폭 사용을 차지한다. 비디오를 수신 및 디스플레이할 수 있는 연결된 사용자 디바이스들의 수가 증가됨에 따라, 디지털 비디오 사용에 대한 대역폭 수요가 계속 증가될 것으로 예상된다.

## 발명의 내용

[0006] 본 문서는 모션 벡터들의 병합 리스트(merge list)를 사용하여 디지털 비디오를 인코딩 및 디코딩하기 위한 방법들, 시스템들, 및 디바이스들을 개시한다.

[0007] 일 예시적인 양태에서, 비디오 프로세싱 방법이 개시된다. 방법은, 하나 이상의 모션 후보 테이블을 유지하는 단계; 모션 후보 테이블을 사용하여, 순차적으로 다수의 비디오 프로세싱 동작을 사용하여, 비디오의 현재 비디오 블록과 현재 비디오 블록의 코딩된 표현 사이의 변환을 수행하는 단계; 및 특정 프로세싱 동작의 완료로 인해 모션 후보 테이블을 업데이트하는 단계를 포함한다.

[0008] 다른 예시적인 양태에서, 다른 비디오 프로세싱 방법이 개시된다. 방법은, 하나 이상의 모션 후보 테이블을 유지하는 단계; 코딩 모드, 치수, 비디오 프로세싱 데이터 유닛, 또는 저 지연 체크를 포함하는 현재 비디오 블록의 특성들에 기초하여, i) 하나 이상의 모션 후보 테이블 중 하나의 모션 후보 테이블을 업데이트할지 여부, 또는 ii) 모션 후보 테이블을 어떻게 업데이트할지 중 적어도 하나를 결정하는 단계; 및 결정에 기초하여, 비디오의 현재 비디오 블록과 비디오의 코딩된 표현 사이의 변환을 수행하는 단계를 포함한다.

[0009] 또 다른 예시적인 양태에서, 본원에서 설명되는 비디오 인코딩 방법을 구현하는 비디오 인코더 디바이스가 개시된다.

[0010] 또 다른 대표적인 양태에서, 본원에서 설명되는 다양한 기법들은 비-일시적 컴퓨터 판독가능 매체들 상에 저장되는 컴퓨터 프로그램 제품으로서 구현될 수 있다. 컴퓨터 프로그램 제품은 본원에서 설명되는 방법들을 수행하기 위한 프로그램 코드를 포함한다.

[0011] 또 다른 대표적인 양태에서, 비디오 디코더 장치는 본원에서 설명되는 봐와 같은 방법을 구현할 수 있다.

[0012] 하나 이상의 구현의 세부사항들은 동봉된 첨부물들, 도면들, 및 아래의 상세한 설명에서 제시된다. 다른 피쳐들은 상세한 설명 및 도면들 및 청구 범위로부터 명백하게 될 것이다.

## 도면의 간단한 설명

[0013] 도 1은 비디오 인코더 구현의 예를 도시하는 블록도이다.

도 2는 H.264 비디오 코딩 표준에서의 매크로블록 파티셔닝을 예시한다.

도 3은 코딩 블록(CB)들을 예측 블록(PU)들로 분할하는 예를 예시한다.

도 4는 CTB를 CB들 및 변환 블록(TB)들로 세분화하기 위한 예시적인 구현을 예시한다. 실선들은 CB 경계들을 표시하고, 점선들은 예시적인 CTB와 그의 파티셔닝 및 대응하는 쿼드트리(quadtrees)를 포함하여 TB 경계들을 표시한다.

도 5는 비디오 데이터를 파티셔닝하기 위한 쿼드트리 바이너리 트리(QTBT) 구조의 예를 도시한다.

도 6은 비디오 블록 파티셔닝의 예를 도시한다.

도 7은 쿼드트리 파티셔닝의 예를 도시한다.

도 8은 트리 타입 시그널링의 예를 도시한다.

도 9는 병합 후보 리스트 구성을 위한 도출 프로세스의 예를 도시한다.

도 10은 공간적 병합 후보들의 예시적인 위치들을 도시한다.

- 도 11은 공간적 병합 후보들의 중복 체크를 위해 고려되는 후보 쌍들의 예를 도시한다.
- 도 12는  $N \times 2N$  및  $2N \times N$  파티션들의 제2 PU에 대한 위치들의 예를 도시한다.
- 도 13은 시간적 병합 후보들에 대한 모션 벡터 스케일링을 예시한다.
- 도 14는 시간적 병합 후보들에 대한 후보 위치들 및 이들의 공동-위치된(co-located) 픽처를 도시한다.
- 도 15는 조합된 양방향 예측 병합 후보의 예를 도시한다.
- 도 16은 모션 벡터 예측 후보에 대한 도출 프로세스의 예를 도시한다.
- 도 17은 공간적 모션 벡터 후보들에 대한 모션 벡터 스케일링의 예를 도시한다.
- 도 18은 CU의 모션 예측을 위한 예시적인 대안적인 시간적 모션 벡터 예측(ATMVP)을 도시한다.
- 도 19는 소스 블록 및 소스 픽처의 식별의 예를 도식적으로 도시한다.
- 도 20은 4개의 서브-블록 및 이웃 블록들을 갖는 하나의 CU의 예를 도시한다.
- 도 21은 양방향 매칭(bilateral matching)의 예를 예시한다.
- 도 22는 템플릿 매칭의 예를 예시한다.
- 도 23은 FRUC(Frame Rate Up Conversion)에서의 단방향 모션 추정(ME)의 예를 도시한다.
- 도 24는 양방향 템플릿 매칭에 기초한 DMVR의 예를 도시한다.
- 도 25는 공간적 병합 후보들을 도출하기 위해 사용되는 공간적 이웃 블록들의 예를 도시한다.
- 도 26은 룩업 테이블 업데이트들을 위한 대표적인 위치의 선택 방법의 예를 도시한다.
- 도 27a 및 도 27b는 새로운 모션 정보 세트르 룩업 테이블을 업데이트하는 예를 예시한다.
- 도 28a 및 도 28b는 본 문서에서 설명되는 비주얼 미디어 디코딩 또는 비주얼 미디어 인코딩 기법을 구현하기 위한 하드웨어 플랫폼들의 예를 도시한다.
- 도 29는 개시되는 기술의 일부 구현들에 기초한 비디오 프로세싱 방법에 대한 흐름도의 예이다.
- 도 30은 개시되는 기술의 일부 구현들에 기초한 비디오 프로세싱 방법에 대한 흐름도의 예이다.
- 도 31은 제안되는 HMVP 방법을 이용하는 디코딩 흐름도의 예를 도시한다.
- 도 32는 제안되는 HMVP 방법을 사용하여 테이블들을 업데이트하는 예를 도시한다.
- 도 33a 및 도 33b는 (하나의 중복 모션 후보가 제거되는) 중복 제거 기반 LUT 업데이트 방법의 예를 도시한다.
- 도 34a 및 도 34b는 (다수의 중복 모션 후보가 제거되는) 중복 제거 기반 LUT 업데이트 방법의 예를 도시한다.

### 발명을 실시하기 위한 구체적인 내용

[0014] 비디오의 압축비를 개선하기 위해, 연구자들은 비디오를 인코딩하기 위한 새로운 기법들을 계속 찾고 있다.

#### [0015] 1. 서론

[0016] 본 문서는 비디오 코딩 기술들과 관련된다. 구체적으로, 본 문서는 비디오 코딩에서의 (병합 모드, AMVP 모드와 같은) 모션 정보 코딩과 관련된다. 이는 HEVC와 같은 기존의 비디오 코딩 표준 또는 완성될 표준(다용도 비디오 코딩(Versatile Video Coding))에 적용될 수 있다. 이는 또한, 향후의 비디오 코딩 표준들 또는 비디오 코덱에 적용가능할 수 있다.

#### [0017] 간단한 논의

[0018] 비디오 코딩 표준들은 주로, 널리 공지된 ITU-T 및 ISO/IEC 표준들의 개발을 통해 진화되었다. ITU-T는 H.261 및 H.263을 제작하였고, ISO/IEC는 MPEG-1 및 MPEG-4 비주얼을 제작하였고, 이러한 2개의 조직은 공동으로, H.262/MPEG-2 비디오 및 H.264/MPEG-4 AVC(Advanced Video Coding), 및 H.265/HEVC 표준들을 제작하였다. H.262 이후, 비디오 코딩 표준들은 시간적 예측 뿐만 아니라 변환 코딩이 활용되는 하이브리드 비디오 코딩 구

조에 기초한다. 전형적인 HEVC 인코더 프레임워크의 예가 도 1에 도시된다.

## 2.1 파티션 구조

### 2.1.1 H.264/AVC에서의 파티션 트리 구조

이전 표준들에서의 코딩 계층의 코어는, 루마 샘플들의 16x16 블록, 및 4:2:0 컬러 샘플링의 일반적인 경우 크로마 샘플들의 2개의 대응하는 8x8 블록을 포함하는 매크로블록이었다.

인트라 코딩된 블록은 픽셀들 사이의 공간적 상관을 활용하기 위해 공간적 예측을 사용한다. 2개의 파티션: 16x16 및 4x4가 정의된다.

인터 코딩된 블록은, 픽처들 사이의 모션을 추정함으로써, 공간적 예측 대신에 시간적 예측을 사용한다. 모션은 16x16 매크로블록, 또는 그의 서브-매크로블록 파티션들: 16x8, 8x16, 8x8, 8x4, 4x8, 4x4(도 2 참조) 중 임의의 것에 대해 독립적으로 추정될 수 있다. 서브-매크로블록 파티션당 하나의 모션 벡터(MV)만이 허용된다.

### 2.1.2 HEVC에서의 파티션 트리 구조

HEVC에서, CTU는 다양한 로컬 특성들에 적응하기 위한 코딩 트리로서 표시되는 쿼드트리 구조를 사용함으로써 CU들로 분할된다. 픽처 영역을 인터 픽처(시간적) 예측을 사용하여 코딩할지 또는 인트라 픽처(공간적) 예측을 사용하여 코딩할지에 대한 판정이 CU 레벨에서 이루어진다. 각각의 CU는 PU 분할 타입에 따라 1개, 2개, 또는 4개의 PU로 추가로 분할될 수 있다. 하나의 PU 내부에서, 동일한 예측 프로세스가 적용되고, 관련 정보가 PU 기초로 디코더로 송신된다. PU 분할 타입에 기초하여 예측 프로세스를 적용함으로써 잔차 블록을 획득한 후에, CU는 CU에 대한 코딩 트리와 유사한 다른 쿼드트리 구조에 따라 변환 유닛(transform unit)(TU)들로 파티셔닝될 수 있다. HEVC 구조의 핵심 피쳐 중 하나는 HEVC 구조가 CU, PU, 및 TU를 포함하는 다수의 파티션 개념을 갖는다는 것이다.

이하에서, HEVC를 사용하는 하이브리드 비디오 코딩에 관련된 다양한 피쳐들이 다음과 같이 강조된다.

1) 코딩 트리 유닛들 및 코딩 트리 블록(coding tree block)(CTB) 구조: HEVC에서의 유사한 구조는 코딩 트리 유닛(coding tree unit)(CTU)이고, 코딩 트리 유닛(CTU)은 인코더에 의해 선택된 크기를 갖고, 종래의 매크로블록보다 더 클 수 있다. CTU는 루마 CTB 및 대응하는 크로마 CTB들 및 신택스 요소들로 구성된다. 루마 CTB의 크기  $L \times L$ 은  $L = 16$ 개, 32개, 또는 64개의 샘플로서 선택될 수 있고, 더 큰 크기들은 전형적으로 더 양호한 압축을 가능하게 한다. 이어서, HEVC는, 트리 구조 및 쿼드트리-유사 시그널링을 사용하여, CTB들을 더 작은 블록들로 파티셔닝하는 것을 지원한다.

2) 코딩 유닛(coding unit)(CU)들 및 코딩 블록(coding block)(CB)들: CTU의 쿼드트리 신택스는 그의 루마 및 크로마 CB들의 크기 및 위치들을 지정한다. 쿼드트리의 루트(root)는 CTU와 연관된다. 따라서, 루마 CTB의 크기는 루마 CB에 대한 가장 큰 지원되는 크기이다. CTU를 루마 및 크로마 CB들로 분할하는 것은 공동으로 시그널링된다. 하나의 루마 CB 및 통상적으로는 2개의 크로마 CB는 연관된 신택스와 함께 코딩 유닛(CU)을 형성한다. CTB는 하나의 CU만을 포함할 수 있거나, 또는 다수의 CU를 형성하도록 분할될 수 있고, 각각의 CU는 예측 유닛(prediction unit)(PU) 및 변환 유닛(TU)들의 트리로의 연관된 파티셔닝을 갖는다.

3) 예측 유닛들 및 예측 블록(PB)들: 픽처 영역을 인터 픽처 예측을 사용하여 코딩할지 또는 인트라 픽처 예측을 사용하여 코딩할지에 대한 판정이 CU 레벨에서 이루어진다. PU 파티셔닝 구조는 CU 레벨에서 그의 루트를 갖는다. 기본 예측 타입 판정에 따라, 이어서, 루마 및 크로마 CB들은 크기에서 추가로 분할될 수 있고, 루마 및 크로마 예측 블록(PB)들로부터 예측될 수 있다. HEVC는 64x64로부터 4x4 샘플들까지의 가변 PB 크기들을 지원한다. 도 3은  $M \times M$  CU에 대한 허용된 PB들의 예들을 도시한다.

4) TU들 및 변환 블록들: 예측 잔차는 블록 변환들을 사용하여 코딩된다. TU 트리 구조는 CU 레벨에서 그의 루트를 갖는다. 루마 CB 잔차는 루마 변환 블록(TB)과 동일할 수 있거나, 또는 더 작은 루마 TB들로 추가로 분할될 수 있다. 이는 크로마 TB들에 동일하게 적용된다. 정사각형 TB 크기들 4x4, 8x8, 16x16, 및 32x32에 대해 이산 코사인 변환(DCT)의 것들과 유사한 정수 기반 함수들이 정의된다. 루마 인트라 픽처 예측 잔차들의 4x4 변환에 대해, 이산 사인 변환(DST)의 형태로부터 도출된 정수 변환이 대안적으로 지정된다.

도 4는 CB들 및 변환 블록(TB)들로의 CTB의 세분화의 예를 도시한다. 실선들은 CB 경계들을 표시하고, 점선들은 TB 경계들을 표시한다. (a) CTB와 그의 파티셔닝. (b) 대응하는 쿼드트리.

#### 2.1.2.1 변환 블록들 및 유닛들로의 트리-구조화 파티셔닝

- [0033] 잔차 코딩에 대해, CB는 변환 블록(TB)들로 재귀적으로 파티셔닝될 수 있다. 파티셔닝은 잔차 쿼드트리에 의해 시그널링된다. 도 4에 예시된 바와 같이, 정사각형 CB 및 TB 파티셔닝만이 지정되고, 여기서, 블록은 사분면들로 재귀적으로 분할될 수 있다. 크기  $M \times M$ 의 주어진 루마 CB에 대해, 플래그는 그것이 크기  $M/2 \times M/2$ 의 4개의 블록으로 분할되는지 여부를 시그널링한다. 추가 분할이 가능한 경우, SPS에서 표시되는 잔차 쿼드트리의 최대 깊이에 의해 시그널링되는 바와 같이, 각각의 사분면은 그것이 4개의 사분면으로 분할되는지 여부를 표시하는 플래그를 배정받는다. 잔차 쿼드트리로부터 기인하는 리프(leaf) 노드 블록들은 변환 코딩에 의해 추가로 프로세싱되는 변환 블록들이다. 인코더는 인코더가 사용할 최대 및 최소 루마 TB 크기들을 표시한다. 분할은 CB 크기가 최대 TB 크기보다 더 클 때 암시적이다. 분할은 표시된 최소치보다 더 작은 루마 TB 크기를 분할이 발생시킬 때 암시적이지 않다. 크로마 TB 크기는, 루마 TB 크기가  $4 \times 4$ 일 때를 제외하고, 각각의 치수에서 루마 TB 크기의 절반이고, 루마 TB 크기가  $4 \times 4$ 인 경우, 4개의  $4 \times 4$  루마 TB에 의해 커버되는 구역에 대해 단일  $4 \times 4$  크로마 TB가 사용된다. 인트라 픽처 예측된 CU들의 경우, (CB 내의 또는 외부의) 가장 근접한 이웃 TB들의 디코딩된 샘플들은 인트라 픽처 예측에 대한 참조 데이터로서 사용된다.
- [0034] 이전 표준들과 대조적으로, HEVC 설계는, 쿼드트리-구조화된 TB 파티셔닝의 잠재적인 코딩 효율 이익들을 최대화하기 위해, 인트라 픽처 예측된 CU들에 대해 TB가 다수의 PB에 걸쳐 있게 허용한다.
- [0035] **2.1.2.2 부모 및 자식 노드들**
- [0036] CTB는 쿼드트리 구조에 따라 분할되고, 그의 노드들은 코딩 유닛들이다. 쿼드트리 구조에서의 복수의 노드는 리프 노드들 및 비-리프 노드(non-leaf node)들을 포함한다. 리프 노드들은 트리 구조에서 자식 노드들을 갖지 않는다(즉, 리프 노드들은 추가로 분할되지 않음). 비-리프 노드들은 트리 구조의 루트 노드를 포함한다. 루트 노드는 비디오 데이터의 초기 비디오 블록(예컨대, CTB)에 대응한다. 복수의 노드의 각각의 비-루트 노드 각각에 대해, 각각의 비-루트 노드는 각각의 비-루트 노드의 트리 구조에서 부모 노드에 대응하는 비디오 블록의 서브-블록인 비디오 블록에 대응한다. 복수의 비-리프 노드의 각각의 비-리프 노드 각각은 트리 구조에서 하나 이상의 자식 노드를 갖는다.
- [0037] **2.1.3 JEM에서의 더 큰 CTU들을 갖는 쿼드트리 플러스(plus) 바이너리 트리 블록 구조**
- [0038] HEVC 이후 향후의 비디오 코딩 기술들을 연구하기 위해, JVET(Joint Video Exploration Team)가 VCEG 및 MPEG에 의해 2015년에 공동으로 설립되었다. 그 이후로, 다수의 새로운 방법들이 JVET에 의해 채택되어 JEM(Joint Exploration Model)으로 명명된 기준 소프트웨어에 넣어졌다.
- [0039] **2.1.3.1 QTBT 블록 파티셔닝 구조**
- [0040] HEVC와 달리, QTBT 구조는 다수의 파티션 타입의 개념들을 제거하고, 즉, 이는 CU, PU, 및 TU 개념들의 분리를 제거하고, CU 파티션 형상들에 대한 더 많은 유연성을 지원한다. QTBT 블록 구조에서, CU는 정사각형 또는 직사각형 형상을 가질 수 있다. 도 5에 도시된 바와 같이, 코딩 트리 유닛(CTU)은 쿼드트리 구조에 의해 먼저 파티셔닝된다. 쿼드트리 리프 노드들은 바이너리 트리 구조에 의해 추가로 파티셔닝된다. 바이너리 트리 분할에는 대칭 수평 분할 및 대칭 수직 분할의 2개의 분할 타입이 존재한다. 바이너리 트리 리프 노드들은 코딩 유닛(CU)들로 지칭되고, 그 세그먼트화는 어떠한 추가 파티셔닝도 없이 예측 및 변환 프로세싱을 위해 사용된다. 이는 QTBT 코딩 블록 구조에서 CU, PU, 및 TU가 동일한 블록 크기를 갖는다는 것을 의미한다. JEM에서, CU는 때때로 상이한 컬러 컴포넌트들의 코딩 블록(CB)들로 구성되고, 예컨대, 4:2:0 크로마 포맷의 P 및 B 슬라이스들의 경우, 하나의 CU는 하나의 루마 CB 및 2개의 크로마 CB를 포함하고, 때때로 단일 컴포넌트의 CB로 구성되고, 예컨대, I 슬라이스들의 경우, 하나의 CU는 단 하나의 루마 CB 또는 단 2개의 크로마 CB를 포함한다.
- [0041] QTBT 파티셔닝 체계에 대해 다음의 파라미터들이 정의된다.
- [0042] - CTU 크기: 쿼드트리의 루트 노드 크기, HEVC에서와 동일한 개념
- [0043] -  $MinQTSIZE$ : 최소 허용 쿼드트리 리프 노드 크기
- [0044] -  $MaxBTSIZE$ : 최대 허용 바이너리 트리 루트 노드 크기
- [0045] -  $MaxBTDepth$ : 최대 허용 바이너리 트리 깊이
- [0046] -  $MinBTSIZE$ : 최소 허용 바이너리 트리 리프 노드 크기
- [0047] QTBT 파티셔닝 구조의 일 예에서, CTU 크기는 크로마 샘플들의 2개의 대응하는  $64 \times 64$  블록을 갖는  $128 \times 128$  루마 샘플들로서 설정되고,  $MinQTSIZE$ 는  $16 \times 16$ 으로서 설정되고,  $MaxBTSIZE$ 는  $64 \times 64$ 로서 설정되고, (폭과 높이 둘 모



두에 대한) *MinBTSIZE*는 4x4로서 설정되고, *MaxBTDpth*는 4로서 설정된다. 쿼드트리 파티셔닝은 쿼드트리 리프 노드들을 생성하기 위해 CTU에 먼저 적용된다. 쿼드트리 리프 노드들은 16x16(즉, *MinQTSIZE*) 내지 128x128(즉, CTU 크기)의 크기를 가질 수 있다. 리프 쿼드트리 노드가 128x128인 경우, 이는 크기가 *MaxBTSIZE*(즉, 64x64)를 초과하기 때문에 바이너리 트리에 의해 추가로 분할되지 않을 것이다. 그렇지 않은 경우, 리프 쿼드트리 노드는 바이너리 트리에 의해 추가로 파티셔닝될 수 있다. 따라서, 쿼드트리 리프 노드는 또한 바이너리 트리에 대한 루트 노드이고, 이는 바이너리 트리 깊이를 0으로 갖는다. 바이너리 트리 깊이가 *MaxBTDpth*(즉, 4)에 도달할 때, 추가 분할이 고려되지 않는다. 바이너리 트리 노드가 *MinBTSIZE*(즉, 4)와 동일한 폭을 갖을 때, 추가 수평 분할이 고려되지 않는다. 유사하게, 바이너리 트리 노드가 *MinBTSIZE*와 동일한 높이를 갖을 때, 추가 수직 분할이 고려되지 않는다. 바이너리 트리의 리프 노드들은, 어떠한 추가 파티셔닝도 없이, 예측 및 변환 프로세싱에 의해 추가로 프로세싱된다. JEM에서, 최대 CTU 크기는 256x256 루마 샘플들이다.

[0048] 도 5(좌측)는 QTBT를 사용한 블록 파티셔닝의 예를 예시하고, 도 5(우측)는 대응하는 트리 표현을 예시한다. 실선들은 쿼드트리 분할을 표시하고, 점선들은 바이너리 트리 분할을 표시한다. 바이너리 트리의 각각의 분할(즉, 비-리프) 노드에서, 어떤 분할 타입(즉, 수평 또는 수직)이 사용되는지를 표시하기 위해 하나의 플래그가 시그널링되고, 여기서, 0은 수평 분할을 표시하고, 1은 수직 분할을 표시한다. 쿼드트리 분할에 대해, 분할 타입을 표시할 필요가 없는데, 이는 쿼드트리 분할은 항상, 동일한 크기를 갖는 4개의 서브-블록을 생성하기 위해 블록을 수평과 수직 둘 모두로 분할하기 때문이다.

[0049] 추가하여, QTBT 체계는 루마 및 크로마가 별개의 QTBT 구조를 갖는 능력을 지원한다. 현재, P 및 B 슬라이스들에 대해, 하나의 CTU 내의 루마 및 크로마 CTB들은 동일한 QTBT 구조를 공유한다. 그러나, I 슬라이스들에 대해, 루마 CTB는 QTBT 구조에 의해 CU들로 파티셔닝되고, 크로마 CTB들은 다른 QTBT 구조에 의해 크로마 CU들로 파티셔닝된다. 이는 I 슬라이스 내의 CU가 루마 컴포넌트의 코딩 블록 또는 2개의 크로마 컴포넌트의 코딩 블록들로 구성되고, P 또는 B 슬라이스 내의 CU가 모든 3개의 컬러 컴포넌트의 코딩 블록으로 구성된다는 것을 의미한다.

[0050] HEVC에서, 작은 블록들에 대한 인터 예측은 모션 보상의 메모리 액세스를 감소시키기 위해 제한되고, 그에 따라, 양방향 예측(bi-prediction)이 4x8 및 8x4 블록들에 대해 지원되지 않고, 인터 예측이 4x4 블록들에 대해 지원되지 않는다. JEM의 QTBT에서, 이러한 제한들이 제거된다.

#### [0051] 2.1.4 VVC에 대한 터너리(ternary) 트리

[0052] 쿼드트리 및 바이너리 트리 이외의 트리 타입들이 지원된다. 구현에서, 도 6의 (d) 및 (e)에 도시된 바와 같이, 2개 이상의 터너리 트리(ternary tree)(TT) 파티션, 즉, 수평 및 수직 중앙-측부(center-side) 터너리 트리가 추가로 도입된다.

[0053] 도 6은: (a) 쿼드트리 파티셔닝, (b) 수직 바이너리 트리 파티셔닝, (c) 수평 바이너리 트리 파티셔닝, (d) 수직 중앙-측부 터너리 트리 파티셔닝, (e) 수평 중앙-측부 터너리 트리 파티셔닝을 도시한다.

[0054] 2개의 트리 레벨, 즉, 구역 트리(쿼드트리) 및 예측 트리(바이너리 트리 또는 터너리 트리)가 존재한다. CTU는 구역 트리(region tree)(RT)에 의해 먼저 파티셔닝된다. RT 리프는 예측 트리(PT)에 의해 추가로 분할될 수 있다. PT 리프는 또한, 최대 PT 깊이가 도달될 때까지 PT에 의해 추가로 분할될 수 있다. PT 리프는 기본 코딩 유닛이다. 이는 편의상 여전히 CU로 지칭된다. CU는 추가로 분할될 수 없다. 예측 및 변환은 둘 모두 JEM과 동일한 방식으로 CU에 적용된다. 전체 파티션 구조는 '다중-타입-트리'로 명명된다.

#### [0055] 2.1.5 파티셔닝 구조

[0056] 다중-트리 타입(Multi-Tree Type)(MTT)으로 지칭되는 이러한 대응으로 사용되는 트리 구조는 QTBT의 일반화이다. QTBT에서, 도 5에 도시된 바와 같이, 코딩 트리 유닛(CTU)은 쿼드트리 구조에 의해 먼저 파티셔닝된다. 쿼드트리 리프 노드들은 바이너리 트리 구조에 의해 추가로 파티셔닝된다.

[0057] 도 7에 도시된 바와 같이, MTT의 기본 구조는 9개의 타입의 파티션을 지원하는 2개의 타입의 트리 노드: 구역 트리(RT) 및 예측 트리(PT)로 구성된다.

[0058] 도 7은: (a) 쿼드트리 파티셔닝, (b) 수직 바이너리 트리 파티셔닝, (c) 수평 바이너리 트리 파티셔닝, (d) 수직 터너리 트리 파티셔닝, (e) 수평 터너리 트리 파티셔닝, (f) 수평-상측 비대칭 바이너리 트리 파티셔닝, (g) 수평-하측 비대칭 바이너리 트리 파티셔닝, (h) 수직-좌측 비대칭 바이너리 트리 파티셔닝(i) 수직-우측 비대칭

바이너리 트리 파티셔닝을 도식한다.

[0059] 구역 트리는 CTU를 4x4 크기 구역 트리 리프 노드까지 정사각형 블록들로 재귀적으로 분할할 수 있다. 구역 트리 내의 각각의 노드에서, 예측 트리는 3개의 트리 타입: 바이너리 트리(BT), 터너리 트리(TT), 및 비대칭 바이너리 트리(ABT) 중 하나로부터 형성될 수 있다. PT 분할에서, 예측 트리의 분지들에서 쿼드트리 파티션을 갖는 것이 금지된다. JEM에서와 같이, 루마 트리 및 크로마 트리는 I 슬라이스들에서 분리된다. RT 및 PT에 대한 시그널링 방법들이 도 8에 예시된다.

## [0060] 2.2 HEVC/H.265에서의 인터 예측

[0061] 각각의 인터 예측된 PU는 1개 또는 2개의 참조 픽처 리스트에 대한 모션 파라미터들을 갖는다. 모션 파라미터들은 모션 벡터 및 참조 픽처 인덱스를 포함한다. 2개의 참조 픽처 리스트 중 하나의 사용이 *inter\_pred\_idc*를 사용하여 또한 시그널링될 수 있다. 모션 벡터들은 예측자(predictor)들에 대한 델타들로서 명시적으로 코딩될 수 있고, 이러한 코딩 모드는 AMVP 모드로 지칭된다.

[0062] CU가 스킵 모드로 코딩될 때, 하나의 PU가 CU와 연관되고, 유의미한 잔차 계수들, 코딩된 모션 벡터 델타, 또는 참조 픽처 인덱스가 존재하지 않는다. 병합 모드가 지정되고, 이에 의해, 공간적 및 시간적 후보들을 포함하는 이웃 PU들로부터 현재 PU에 대한 모션 파라미터들이 획득된다. 병합 모드는, 스킵 모드 뿐만 아니라, 임의의 인터 예측된 PU에 적용될 수 있다. 병합 모드에 대한 대안은 모션 파라미터들의 명시적 송신이고, 여기서, 모션 벡터, 각각의 참조 픽처 리스트에 대한 대응하는 참조 픽처 인덱스, 및 참조 픽처 리스트 사용이 각각의 PU마다 명시적으로 시그널링된다.

[0063] 시그널링이 2개의 참조 픽처 리스트 중 하나가 사용될 것임을 표시할 때, PU는 샘플들의 하나의 블록으로부터 생성된다. 이는 '단방향 예측(uni-prediction)'으로 지칭된다. P-슬라이스들과 B-슬라이스들 둘 모두에 대해 단방향 예측이 이용가능하다.

[0064] 시그널링이 참조 픽처 리스트들 둘 모두가 사용될 것임을 표시할 때, PU는 샘플들의 2개의 블록으로부터 생성된다. 이는 '양방향 예측'으로 지칭된다. 양방향 예측은 B-슬라이스들에 대해서만 이용가능하다.

[0065] 다음의 텍스트는 HEVC에서 지정된 인터 예측 모드들에 대한 세부사항들을 제공한다. 설명은 병합 모드로 시작될 것이다.

### [0066] 2.2.1 병합 모드

#### [0067] 2.2.1.1 병합 모드에 대한 후보들의 도출

[0068] PU가 병합 모드를 사용하여 예측될 때, 병합 후보 리스트 내의 엔트리를 지적하는 인덱스가 비트스트림으로부터 파싱되고(parsed), 모션 정보를 검색하기 위해 사용된다. 이 리스트의 구성은 HEVC 표준에서 지정되고, 단계들의 다음의 시퀀스에 따라 요약될 수 있다:

[0069] ● 단계 1: 초기 후보 도출

[0070] ○ 단계 1.1: 공간적 후보 도출

[0071] ○ 단계 1.2: 공간적 후보들에 대한 중복 체크

[0072] ○ 단계 1.3: 시간적 후보 도출

[0073] ● 단계 2: 추가적인 후보 삽입

[0074] ○ 단계 2.1: 양방향 예측 후보들의 생성

[0075] ○ 단계 2.2: 제로(zero) 모션 후보들의 삽입

[0076] 이러한 단계들은 또한, 도 9에 개략적으로 도시된다. 공간적 병합 후보 도출에 대해, 5개의 상이한 위치에 위치한 후보들 중에서 최대 4개의 병합 후보가 선택된다. 시간적 병합 후보 도출에 대해, 2개의 후보 중에서 최대 하나의 병합 후보가 선택된다. 디코더에서 각각의 PU에 대한 일정한 수의 후보가 가정되기 때문에, 슬라이스 헤더에서 시그널링되는 병합 후보의 최대 수(MaxNumMergeCand)에 후보의 수가 도달하지 않을 때, 추가적인 후보들이 생성된다. 후보의 수가 일정하기 때문에, 최상의 병합 후보의 인덱스는 절단 단항 바이너리화(truncated unary binarization)(TU)를 사용하여 인코딩된다. CU의 크기가 8과 동일한 경우, 현재 CU의 모든 PU들은 2Nx2N 예측 유닛의 병합 후보 리스트와 동일한 단일 병합 후보 리스트를 공유한다.



[0077] 이하에서, 전술된 단계들과 연관된 동작들이 상세히 설명된다.

#### [0078] 2.2.1.2 공간적 후보 도출

[0079] 공간적 병합 후보들의 도출에서, 도 10에 도시된 위치들에 위치한 후보들 중에서 최대 4개의 병합 후보가 선택된다. 도출의 순서는  $A_1$ ,  $B_1$ ,  $B_0$ ,  $A_0$ , 및  $B_2$ 이다. 위치  $B_2$ 는 위치  $A_1$ ,  $B_1$ ,  $B_0$ ,  $A_0$ 의 임의의 PU가 (예컨대, 그 임의의 PU가 다른 슬라이스 또는 타일에 속하기 때문에) 이용가능하지 않거나 또는 인트라 코딩될 때에만 고려된다. 위치  $A_1$ 에서의 후보가 추가된 후에, 나머지 후보들의 추가는 중복 체크를 거치고, 이는, 코딩 효율이 개선되도록, 동일한 모션 정보를 갖는 후보들이 리스트로부터 배제되는 것을 보장한다. 계산 복잡도를 감소시키기 위해, 언급된 중복 체크에서 모든 가능한 후보 쌍들이 고려되지는 않는다. 대신에, 도 11에서 화살표로 링크된 쌍들만이 고려되고, 중복 체크에 대해 사용된 대응하는 후보가 동일한 모션 정보를 갖지 않는 경우에만 후보가 리스트에 추가된다. 중복 모션 정보의 다른 소스는  $2N \times 2N$ 과 상이한 파티션들과 연관된 "제2 PU"이다. 예로서, 도 12는 각각  $N \times 2N$  및  $2N \times N$ 의 경우에 대한 제2 PU를 도시한다. 현재 PU가  $N \times 2N$ 으로서 파티셔닝될 때, 위치  $A_1$ 에서의 후보는 리스트 구성을 위해 고려되지 않는다. 실제로, 이 후보를 추가하는 것에 의해 동일한 모션 정보를 갖는 2개의 예측 유닛이 발생될 것이고, 이는 중복되고, 그에 따라, 코딩 유닛에서 단지 하나의 PU만을 갖게 된다. 유사하게, 위치  $B_1$ 은 현재 PU가  $2N \times N$ 으로서 파티셔닝될 때 고려되지 않는다.

#### [0080] 2.2.1.3 시간적 후보 도출

[0081] 이 단계에서, 하나의 후보만이 리스트에 추가된다. 특히, 이러한 시간적 병합 후보의 도출에서, 주어진 참조 픽처 리스트 내에서 현재 픽처와 가장 작은 POC 차이를 갖는 픽처에 속하는 공동-위치된 PU에 기초하여, 스케일링된 모션 벡터가 도출된다. 공동-위치된 PU의 도출을 위해 사용될 참조 픽처 리스트는 슬라이스 헤더에서 명시적으로 시그널링된다. 시간적 병합 후보에 대한 스케일링된 모션 벡터는 도 13에서 파선에 의해 예시된 바와 같이 획득되고, 그 스케일링된 모션 벡터는, POC 거리들( $t_b$  및  $t_d$ )을 사용하여, 공동-위치된 PU의 모션 벡터로부터 스케일링되고, 여기서,  $t_b$ 는 현재 픽처의 참조 픽처와 현재 픽처 사이의 POC 차이인 것으로 정의되고,  $t_d$ 는 공동-위치된 픽처의 참조 픽처와 공동-위치된 픽처 사이의 POC 차이인 것으로 정의된다. 시간적 병합 후보의 참조 픽처 인덱스는 제로와 동일하게 설정된다. 스케일링 프로세스의 실제 실현은 HEVC 사양에서 설명된다. B-슬라이스에 대해, 하나는 참조 픽처 리스트 0에 대한 것이고 다른 하나는 참조 픽처 리스트 1에 대한 것인 2개의 모션 벡터가 획득되고 조합되어 양방향-예측 병합 후보를 형성한다. 시간적 병합 후보에 대한 모션 벡터 스케일링이 예시된다.

[0082] 참조 프레임에 속하는 공동-위치된 PU (Y)에서, 시간적 후보에 대한 위치는, 도 14에 도시된 바와 같이, 후보들  $C_0$ 과  $C_1$  사이에서 선택된다. 위치  $C_0$ 에서의 PU가 이용가능하지 않거나, 인트라 코딩되거나, 또는 현재 CTU 외부에 있는 경우, 위치  $C_1$ 이 사용된다. 그렇지 않은 경우, 시간적 병합 후보의 도출에서 위치  $C_0$ 이 사용된다.

#### [0083] 2.2.1.4 추가적인 후보 삽입

[0084] 공간-시간적(spatio-temporal) 병합 후보들 이외에, 2개의 추가적인 타입의 병합 후보: 조합된 양방향 예측 병합 후보 및 제로 병합 후보가 존재한다. 조합된 양방향 예측 병합 후보들은 공간-시간적 병합 후보들을 활용함으로써 생성된다. 조합된 양방향 예측 병합 후보는 B-슬라이스에 대해서만 사용된다. 조합된 양방향 예측 후보들은, 초기 후보의 제1 참조 픽처 리스트 모션 파라미터들을 다른 후보의 제2 참조 픽처 리스트 모션 파라미터들과 조합함으로써, 생성된다. 이러한 2개의 튜플(tuple)이 상이한 모션 가설들을 제공하는 경우, 이들은 새로운 양방향 예측 후보를 생성할 것이다. 예로서, 도 15는, 최종 리스트(우측에 있음)에 추가되는 조합된 양방향 예측 병합 후보를 생성하기 위해,  $mvL0$ 과  $refIdxL0$  또는  $mvL1$ 과  $refIdxL1$ 을 갖는 본래 리스트(좌측에 있음) 내의 2개의 후보가 사용되는 경우를 도시한다. 이러한 추가적인 병합 후보들을 생성하기 위해 고려되는 조합들에 관한 다수의 규칙이 존재한다.

[0085] 병합 후보 리스트 내의 나머지 엔트리들을 채우기 위해 제로 모션 후보들이 삽입되고, 그에 따라,  $MaxNumMergeCand$  용량에 도달한다. 이러한 후보들은 제로 공간 변위, 및 제로로부터 시작하여 새로운 제로 모션 후보가 리스트에 추가될 때마다 증가하는 참조 픽처 인덱스를 갖는다. 이러한 후보들에 의해 사용되는 참조 프레임의 수는 단방향 및 양방향 예측에 대해 각각 1개 및 2개이다. 마지막으로, 이러한 후보들에 대해 중복 체크는 수행되지 않는다.

#### [0086] 2.2.1.5 병렬 프로세싱을 위한 모션 추정 구역들

[0087] 인코딩 프로세스를 가속시키기 위해, 모션 추정이 병렬로 수행될 수 있고, 이에 의해, 주어진 구역 내부의 모든 예측 유닛들에 대한 모션 벡터들이 동시에 도출된다. 공간적 이웃으로부터의 병합 후보들의 도출은 병렬 프로세싱을 방해할 수 있는데, 이는 하나의 예측 유닛이 그의 연관된 모션 추정이 완료될 때까지 인접 PU로부터 모션 파라미터들을 도출할 수 없기 때문이다. 코딩 효율과 프로세싱 레이턴시 사이의 절충을 완화하기 위해, HEVC는 모션 추정 구역(motion estimation region)(MER)을 정의하고, 그의 크기는 "log2\_parallel\_merge\_level\_minus2" 선택 요소를 사용하여 픽처 파라미터 세트에서 시그널링된다. MER이 정의될 때, 동일한 구역에 속하는 병합 후보들은 이용불가능한 것으로서 마킹되고, 그에 따라, 리스트 구성에서 고려되지 않는다.

### [0088] 7.3.2.3 픽처 파라미터 세트 RBSP 선택스

#### [0089] 7.3.2.3.1 일반적인 픽처 파라미터 세트 RBSP 선택스

|   |       |
|---|-------|
| pic_parameter_set_rbsp() {                  | 디스크립터 |
| pps_pic_parameter_set_id                    | ue(v) |
| pps_seq_parameter_set_id                    | ue(v) |
| dependent_slice_segments_enabled_flag       | u(1)  |
| ...   |       |
| pps_scaling_list_data_present_flag          | u(1)  |
| if( pps_scaling_list_data_present_flag )    |       |
| scaling_list_data()                         |       |
| lists_modification_present_flag             | u(1)  |
| log2_parallel_merge_level_minus2            | ue(v) |
| slice_segment_header_extension_present_flag | u(1)  |
| pps_extension_present_flag                  | u(1)  |
| ...   |       |
| rbbsp_trailing_bits()                       |       |
| }   |       |

[0090]

[0091] log2\_parallel\_merge\_level\_minus2 플러스 2는 변수 Log2ParMrgLevel의 값을 지정하고, 그 값은 조항 8.5.3.2.2에서 지정된 바와 같은 병합 모드를 위한 루마 모션 벡터들에 대한 도출 프로세스 및 조항 8.5.3.2.3에서 지정된 바와 같은 공간적 병합 후보들에 대한 도출 프로세스에서 사용된다. log2\_parallel\_merge\_level\_minus2의 값은 0 내지 CtbLog2SizeY - 2의 범위(경계값을 포함함)에 있어야 한다.

[0092] 변수 Log2ParMrgLevel은 다음과 같이 도출된다:

[0093]  $\text{Log2ParMrgLevel} = \text{log2\_parallel\_merge\_level\_minus2} + 2 \text{ (7-37)}$

[0094] 주 3 - Log2ParMrgLevel의 값은 병합 후보 리스트들의 병렬 도출의 내장된 능력을 표시한다. 예컨대, Log2ParMrgLevel이 6과 동일할 때, 64x64 블록에 포함된 모든 예측 유닛(PU)들 및 코딩 유닛(CU)들에 대한 병합 후보 리스트들이 병렬로 도출될 수 있다.

### [0095] 2.2.2 AMVP 모드에서의 모션 벡터 예측

[0096] 모션 벡터 예측은 이웃 PU들과의 모션 벡터의 공간-시간적 상관을 활용하고, 이는 모션 파라미터들의 명시적 송신을 위해 사용된다. 이는, 좌측, 상측의 시간적으로 이웃하는 PU 위치들의 이용가능성을 먼저 체크하고, 중복 후보들을 제거하고, 제로 벡터를 추가하여 후보 리스트를 일정한 길이가 되도록 함으로써, 모션 벡터 후보 리스트를 구성한다. 이어서, 인코더는 후보 리스트로부터 최상의 예측자를 선택할 수 있고, 선택된 후보를 표시하는 대응하는 인덱스를 송신할 수 있다. 병합 인덱스 시그널링과 유사하게, 최상의 모션 벡터 후보의 인덱스는 절단 단항을 사용하여 인코딩된다. 이 경우에 인코딩될 최대 값은 2이다(예컨대, 도 2 내지 도 8). 이하의 섹션들에서, 모션 벡터 예측 후보의 도출 프로세스에 관한 세부사항들이 제공된다.

#### [0097] 2.2.2.1 모션 벡터 예측 후보들의 도출

[0098] 도 16은 모션 벡터 예측 후보에 대한 도출 프로세스를 요약한다.

[0099] 모션 벡터 예측에서, 2개의 타입의 모션 벡터 후보: 공간적 모션 벡터 후보 및 시간적 모션 벡터 후보가 고려된다. 공간적 모션 벡터 후보 도출에 대해, 도 11에 도시된 바와 같은 5개의 상이한 위치에 위치한 각각의 PU의

모션 벡터들에 기초하여, 2개의 모션 벡터 후보가 결과적으로 도출된다.

[0100] 시간적 모션 벡터 후보 도출에 대해, 2개의 상이한 공동-위치된 위치에 기초하여 도출되는 2개의 후보로부터 하나의 모션 벡터 후보가 선택된다. 공간-시간적 후보들의 제1 리스트가 형성된 후에, 리스트 내의 중복 모션 벡터 후보들이 제거된다. 잠재적 후보의 수가 2개보다 더 많은 경우, 연관된 참조 픽처 리스트 내의 참조 픽처 인덱스가 1보다 더 큰 모션 벡터 후보들이 리스트로부터 제거된다. 공간-시간적 모션 벡터 후보의 수가 2개보다 더 적을 경우, 추가적인 제로 모션 벡터 후보들이 리스트에 추가된다.

#### [0101] 2.2.2.2 공간적 모션 벡터 후보들

[0102] 공간적 모션 벡터 후보들의 도출에서, 도 11에 도시된 바와 같은 위치들에 위치된 PU로부터 도출되는 5개의 잠재적 후보 중에서 최대 2개의 후보가 고려되고, 이러한 위치들은 모션 병합의 위치들과 동일하다. 현재 PU의 좌측에 대한 도출의 순서는  $A_0$ ,  $A_1$ , 및 스케일링된  $A_0$ , 스케일링된  $A_1$ 로서 정의된다. 현재 PU의 상측에 대한 도출의 순서는  $B_0$ ,  $B_1$ ,  $B_2$ , 스케일링된  $B_0$ , 스케일링된  $B_1$ , 스케일링된  $B_2$ 로서 정의된다. 따라서, 각각의 측에 대해, 모션 벡터 후보로서 사용될 수 있는 4개의 경우가 있고, 2개의 경우는 공간적 스케일링을 사용하도록 요구되지 않고, 2개의 경우에는 공간적 스케일링이 사용된다. 이러한 4개의 상이한 경우는 다음과 같이 요약된다.

[0103] ● 공간적 스케일링 부재

[0104] - (1) 동일한 참조 픽처 리스트, 및 동일한 참조 픽처 인덱스(동일한 POC)

[0105] - (2) 상이한 참조 픽처 리스트, 그러나 동일한 참조 픽처(동일한 POC)

[0106] ● 공간적 스케일링

[0107] - (3) 동일한 참조 픽처 리스트, 그러나 상이한 참조 픽처(상이한 POC)

[0108] - (4) 상이한 참조 픽처 리스트, 및 상이한 참조 픽처(상이한 POC)

[0109] 공간적-스케일링-부재 경우들은 먼저 체크되고, 이어서 공간적 스케일링이 후속된다. 참조 픽처 리스트와 상관 없이, 이웃 PU의 참조 픽처와 현재 PU의 참조 픽처 사이에서 POC가 상이할 때, 공간적 스케일링이 고려된다. 좌측 후보들의 모든 PU들이 이용가능하지 않거나 또는 인트라 코딩되는 경우, 좌측 및 상측 MV 후보들의 병렬 도출을 돕기 위해, 상측 모션 벡터에 대한 스케일링이 허용된다. 그렇지 않은 경우, 상측 모션 벡터에 대해 공간적 스케일링이 허용되지 않는다.

[0110] 공간적 스케일링 프로세스에서, 이웃 PU의 모션 벡터는, 도 17에 도시된 바와 같이, 시간적 스케일링에 대한 것과 유사한 방식으로 스케일링된다. 주된 차이는 현재 PU의 참조 픽처 리스트 및 인덱스가 입력으로서 주어지는 것이고; 실제 스케일링 프로세스는 시간적 스케일링의 것과 동일하다.

#### [0111] 2.2.2.3 시간적 모션 벡터 후보들

[0112] 참조 픽처 인덱스 도출 이외에, 시간적 병합 후보들의 도출에 대한 모든 프로세스들은 공간적 모션 벡터 후보들의 도출에 대한 것(예컨대, 도 6 참조)과 동일하다. 참조 픽처 인덱스는 디코더로 시그널링된다.

#### [0113] 2.2.2.4 AMVP 정보의 시그널링

[0114] AMVP 모드에 대해, 4개의 부분, 즉, 예측 방향, 참조 인덱스, MVD, 및 mv 예측자 후보 인덱스가 비트스트림에서 시그널링될 수 있다.

[0115] 신택스 테이블들:

|   |       |
|---|-------|
| prediction_unit( x0, y0, nPbW, nPbH ) {                         | 디스크립터 |
| if( cu_skip_flag[ x0 ][ y0 ] ) {                                |       |
| if( MaxNumMergeCand > 1 )                                       |       |
| <b>merge_idx</b> [ x0 ][ y0 ]                                   | ae(v) |
| } else { /* MODE_INTER */                                       |       |
| <b>merge_flag</b> [ x0 ][ y0 ]                                  | ae(v) |
| if( merge_flag[ x0 ][ y0 ] ) {                                  |       |
| if( MaxNumMergeCand > 1 )                                       |       |
| <b>merge_idx</b> [ x0 ][ y0 ]                                   | ae(v) |
| } else {  |       |
| if( slice_type == B )   |       |
| <b>inter_pred_idc</b> [ x0 ][ y0 ]                              | ae(v) |
| if( inter_pred_idc[ x0 ][ y0 ] != PRED_L1 ) {                   |       |
| if( num_ref_idx_l0_active_minus1 > 0 )                          |       |
| <b>ref_idx_l0</b> [ x0 ][ y0 ]                                  | ae(v) |
| mvd_coding( x0, y0, 0 )   |       |
| <b>mvp_l0_flag</b> [ x0 ][ y0 ]                                 | ae(v) |
| }   |       |
| if( inter_pred_idc[ x0 ][ y0 ] != PRED_L0 ) {                   |       |
| if( num_ref_idx_l1_active_minus1 > 0 )                          |       |
| <b>ref_idx_l1</b> [ x0 ][ y0 ]                                  | ae(v) |
| if( mvd_l1_zero_flag && inter_pred_idc[ x0 ][ y0 ] == PRED_BI ) |       |
| {   |       |
| MvdL1[ x0 ][ y0 ][ 0 ] = 0                                      |       |
| MvdL1[ x0 ][ y0 ][ 1 ] = 0                                      |       |
| } else  |       |
| mvd_coding( x0, y0, 1 )   |       |
| <b>mvp_l1_flag</b> [ x0 ][ y0 ]                                 | ae(v) |
| }   |       |
| }   |       |
| }   |       |
| }   |       |
| }   |       |

[0116]

## 7.3.8.9 모션 벡터 차이 선택스

|                                    |       |
|------------------------------------|-------|
| mvd_coding( x0, y0, refList ) {    | 디스크립터 |
| abs_mvd_greater0_flag[ 0 ]         | ae(v) |
| abs_mvd_greater0_flag[ 1 ]         | ae(v) |
| if( abs_mvd_greater0_flag[ 0 ] )   |       |
| abs_mvd_greater1_flag[ 0 ]         | ae(v) |
| if( abs_mvd_greater0_flag[ 1 ] )   |       |
| abs_mvd_greater1_flag[ 1 ]         | ae(v) |
| if( abs_mvd_greater0_flag[ 0 ] ) { |       |
| if( abs_mvd_greater1_flag[ 0 ] )   |       |
| abs_mvd_minus2[ 0 ]                | ae(v) |
| mvd_sign_flag[ 0 ]                 | ae(v) |
| }                                  |       |
| if( abs_mvd_greater0_flag[ 1 ] ) { |       |
| if( abs_mvd_greater1_flag[ 1 ] )   |       |
| abs_mvd_minus2[ 1 ]                | ae(v) |
| mvd_sign_flag[ 1 ]                 | ae(v) |
| }                                  |       |
| }                                  |       |

## 2.3 JEM(Joint Exploration Model)에서의 새로운 인터 예측 방법들

## 2.3.1 서브-CU 기반 모션 벡터 예측

QTBT를 이용하는 JEM에서, 각각의 CU는 각각의 예측 방향에 대해 최대 하나의 모션 파라미터 세트를 가질 수 있다. 큰 CU를 서브-CU들로 분할하고 큰 CU의 모든 서브-CU들에 대한 모션 정보를 도출하는 것에 의해, 인코더에서 2개의 서브-CU 레벨 모션 벡터 예측 방법들이 고려된다. 대안적인 시간적 모션 벡터 예측(alternative temporal motion vector prediction)(ATMVP) 방법은 각각의 CU로 하여금 공동-위치된 참조 픽처에서 현재 CU보다 더 작은 다수의 블록으로부터 다수의 모션 정보 세트를 인출하게 허용한다. 공간-시간적 모션 벡터 예측(spatial-temporal motion vector prediction)(STMVP) 방법에서, 서브-CU들의 모션 벡터들은 시간적 모션 벡터 예측자 및 공간적 이웃 모션 벡터를 사용함으로써 재귀적으로 도출된다.

서브-CU 모션 예측에 대한 더 정확한 모션 필드를 보존하기 위해, 참조 프레임들에 대한 모션 압축은 현재 디스에이블된다.

## 2.3.1.1 대안적인 시간적 모션 벡터 예측

대안적인 시간적 모션 벡터 예측(ATMVP) 방법에서, 모션 벡터 시간적 모션 벡터 예측(TMVP)은 현재 CU보다 더 작은 블록들로부터 다수의 모션 정보 세트(모션 벡터들 및 참조 인덱스들을 포함함)를 인출함으로써 수정된다. 도 18에 도시한 바와 같이, 서브-CU는 정사각형 NxN 블록들(디폴트로 N은 4로 설정됨)이다.

ATMVP는 2개의 단계에서 CU 내의 서브-CU들의 모션 벡터들을 예측한다. 제1 단계는 소위 시간적 벡터로 참조 픽처 내의 대응하는 블록을 식별하는 것이다. 참조 픽처는 모션 소스 픽처로 지칭된다. 제2 단계는, 도 18에 도시된 바와 같이, 현재 CU를 서브-CU들로 분할하고, 각각의 서브-CU에 대응하는 블록으로부터 각각의 서브-CU의 모션 벡터들 뿐만 아니라 참조 인덱스들을 획득하는 것이다.

제1 단계에서, 참조 픽처 및 대응하는 블록은 현재 CU의 공간적 이웃 블록들의 모션 정보에 의해 결정된다. 이웃 블록들의 반복적인 스캐닝 프로세스를 피하기 위해, 현재 CU의 병합 후보 리스트 내의 제1 병합 후보가 사용된다. 제1 이용가능한 모션 벡터 뿐만 아니라 그의 연관된 참조 인덱스는 모션 소스 픽처에 대한 **시간적 벡터** 및 인덱스로 설정된다. 이러한 방식으로, ATMVP에서, 대응하는 블록이 TMVP에 비해 더 정확하게 식별될 수 있고, 여기서, 대응하는 블록(때때로, 공동-위치된 블록으로 지칭됨)은 항상 현재 CU에 대해 하단-우측 또는 중앙 위치에 있다. 일 예에서, 제1 병합 후보가 좌측 이웃 블록(즉, 도 19의 A<sub>1</sub>)으로부터 유래한 경우, 연관된 MV 및 참조 픽처는 소스 블록 및 소스 픽처를 식별하기 위해 활용된다.



- [0127] 도 19는 소스 블록 및 소스 픽처의 식별의 예를 도시한다.
- [0128] 제2 단계에서, 현재 CU의 좌표에 시간적 벡터를 추가함으로써, 모션 소스 픽처에서의 시간적 벡터에 의해, 서브-CU의 대응하는 블록이 식별된다. 각각의 서브-CU에 대해, 그의 대응하는 블록(중앙 샘플을 커버하는 가장 작은 모션 그리드)의 모션 정보는 서브-CU에 대한 모션 정보를 도출하기 위해 사용된다. 대응하는 NxN 블록의 모션 정보가 식별된 후에, 그 모션 정보는, HEVC의 TMVP와 동일한 방식으로, 현재 서브-CU의 모션 벡터들 및 참조 인덱스들로 변환되고, 여기서, 모션 스케일링 및 다른 절차들이 적용된다. 예컨대, 디코더는 저 지연 조건(즉, 현재 픽처의 모든 참조 픽처들의 POC들이 현재 픽처의 POC보다 더 작은 것)이 충족되는지 여부를 체크하고, 가능하게는, 각각의 서브-CU에 대해, 모션 벡터  $MV_x$ (참조 픽처 리스트 X에 대응하는 모션 벡터)를 사용하여, 모션 벡터  $MV_y$ (X는 0 또는 1과 동일하고 Y는 1-X와 동일함)를 예측한다.
- [0129] **2.3.1.2 공간-시간적 모션 벡터 예측**
- [0130] 이 방법에서, 서브-CU들의 모션 벡터들은 래스터 스캔 순서에 따라 재귀적으로 도출된다. 도 20은 이 개념을 예시한다. 4개의 4x4 서브-CU A, B, C, 및 D를 포함하는 8x8 CU를 고려한다. 현재 프레임 내의 이웃하는 4x4 블록들은 a, b, c, 및 d로서 표시된다(labelled).
- [0131] 서브-CU A에 대한 모션 도출은 그의 2개의 공간적 이웃을 식별하는 것에 의해 시작된다. 제1 이웃은 서브-CU A의 상측의 NxN 블록(블록 c)이다. 이 블록 c가 이용가능하지 않거나 또는 인트라 코딩되는 경우, 서브-CU A의 상측의 다른 NxN 블록들이(블록 c에서 시작하여 좌측에서 우측으로) 체크된다. 제2 이웃은 서브-CU A의 좌측의 블록(블록 b)이다. 블록 b가 이용가능하지 않거나 또는 인트라 코딩되는 경우, 서브-CU A의 좌측의 다른 블록들이(블록 b에서 시작하여 상단으로부터 하단으로) 체크된다. 각각의 리스트에 대해 이웃 블록들로부터 획득된 모션 정보는 주어진 리스트에 대해 제1 참조 프레임으로 스케일링된다. 다음으로, HEVC에서 지정된 바와 같은 TMVP 도출의 동일한 절차를 따르는 것에 의해, 서브-블록 A의 시간적 모션 벡터 예측자(TMVP)가 도출된다. 위치 D에서의 공동-위치된 블록의 모션 정보가 그에 따라 인출 및 스케일링된다. 마지막으로, 모션 정보를 검색 및 스케일링한 후에, 모든 이용가능한 모션 벡터들(최대 3개)은 각각의 참조 리스트에 대해 개별적으로 평균된다. 평균된 모션 벡터는 현재 서브-CU의 모션 벡터로서 지정된다.
- [0132] 도 20은 4개의 서브-블록(A-D) 및 그의 이웃 블록들(a-d)을 갖는 하나의 CU의 예를 도시한다.
- [0133] **2.3.1.3 서브-CU 모션 예측 모드 시그널링**
- [0134] 서브-CU 모드들은 추가적인 병합 후보들로서 인에이블되고, 모드들을 시그널링하기 위해 요구되는 추가적인 인덱스 요소는 없다. ATMVP 모드 및 STMVP 모드를 표현하기 위해, 각각의 CU의 병합 후보 리스트에 2개의 추가적인 병합 후보가 추가된다. ATMVP 및 STMVP가 인에이블되는 것을 시퀀스 파라미터 세트가 표시하는 경우, 최대 7개의 병합 후보가 사용된다. 추가적인 병합 후보들의 인코딩 로직은 HM에서의 병합 후보들에 대한 것과 동일하고, 이는, P 또는 B 슬라이스 내의 각각의 CU에 대해, 2개의 추가적인 병합 후보를 위해 2개의 RD 체크가 추가로 필요하다는 것을 의미한다.
- [0135] JEM에서, 병합 인덱스의 모든 bin(bin)들은 CABAC에 의해 컨텍스트 코딩된다. 한편 HEVC에서는, 제1 bin만이 컨텍스트 코딩되고, 나머지 bin들은 컨텍스트 바이-패스 코딩된다(context by-pass coded).
- [0136] **2.3.2 적응적 모션 벡터 차이 해상도**
- [0137] HEVC에서, (PU의 예측된 모션 벡터와 모션 벡터 사이의) 모션 벡터 차이(motion vector difference)(MVD)들은, 슬라이스 헤더에서 use\_integer\_mv\_flag가 0과 동일할 때, 쿼터 루마 샘플의 단위로 시그널링된다. JEM에서, 로컬 적응적 모션 벡터 해상도(locally adaptive motion vector resolution)(LAMVR)가 도입된다. JEM에서, MVD는 쿼터 루마 샘플, 정수 루마 샘플, 또는 4 루마 샘플의 단위로 코딩될 수 있다. MVD 해상도는 코딩 유닛(CU) 레벨에서 제어되고, MVD 해상도 플래그들은 적어도 하나의 비-제로 MVD 컴포넌트를 갖는 각각의 CU에 대해 조건부로 시그널링된다.
- [0138] 적어도 하나의 비-제로 MVD 컴포넌트를 갖는 CU에 대해, 쿼터 루마 샘플 MV 정밀도가 CU에서 사용되는지 여부를 표시하기 위해 제1 플래그가 시그널링된다. 쿼터 루마 샘플 MV 정밀도가 사용되지 않는다는 것을 제1 플래그(1과 동일함)가 표시할 때, 정수 루마 샘플 MV 정밀도 또는 4 루마 샘플 MV 정밀도가 사용되는지 여부를 표시하기 위해 다른 플래그가 시그널링된다.
- [0139] CU의 제1 MVD 해상도 플래그가 제로이거나 또는 CU에 대해 코딩되지 않을 때(CU에서의 모든 MVD들이 제로인 것

을 의미함), 쿼터 루마 샘플 MV 해상도가 CU에 대해 사용된다. CU가 정수 루마 샘플 MV 정밀도 또는 4 루마 샘플 MV 정밀도를 사용할 때, CU에 대한 AMVP 후보 리스트에서의 MVP들은 대응하는 정밀도로 라운딩된다(rounded).

[0140] 인코더에서, 어떤 MVD 해상도가 CU에 대해 사용될지를 결정하기 위해 CU 레벨 RD 체크들이 사용된다. 즉, CU 레벨 RD 체크는 각각의 MVD 해상도에 대해 3회 수행된다. 인코더 속도를 가속시키기 위해, 다음의 인코딩 체계들이 JEM에 적용된다.

[0141] 정상적인 쿼터 루마 샘플 MVD 해상도에 대한 CU의 RD 체크 동안, 현재 CU의 모션 정보(정수 루마 샘플 정확도)가 저장된다. (라운딩 후의) 저장된 모션 정보는, 정수 루마 샘플 및 4 루마 샘플 MVD 해상도에 대한 동일한 CU에 대한 RD 체크 동안, 더 작은 범위 모션 벡터 리파인먼트(refinement)에 대한 시작 포인트로서 사용되고, 그에 따라, 시간 소모적인 모션 추정 프로세스가 3회 중복되지 않게 된다.

[0142] 4 루마 샘플 MVD 해상도에 대한 CU의 RD 체크가 조건부로 호출된다(invoked). CU에 대해, RD 비용 정수 루마 샘플 MVD 해상도가 쿼터 루마 샘플 MVD 해상도의 것보다 훨씬 더 클 때, CU에 대한 4 루마 샘플 MVD 해상도의 RD 체크는 스킵된다.

### [0143] 2.3.3 패턴 매칭된 모션 벡터 도출

[0144] 패턴 매칭된 모션 벡터 도출(pattern matched motion vector derivation)(PMMVD) 모드는 FRUC(Frame-Rate Up Conversion) 기법들에 기초한 특수 병합 모드이다. 이 모드에서, 블록의 모션 정보는 시그널링되는 것이 아니라 디코더 측에서 도출된다.

[0145] FRUC 플래그는 CU의 병합 플래그가 참일 때 CU에 대해 시그널링된다. FRUC 플래그가 거짓일 때, 병합 인덱스가 시그널링되고, 정규 병합 모드가 사용된다. FRUC 플래그가 참일 때, 어떤 방법(양방향 매칭 또는 템플릿 매칭)이 블록에 대한 모션 정보를 도출하기 위해 사용될지를 표시하기 위해, 추가적인 FRUC 모드 플래그가 시그널링된다.

[0146] 인코더 측에서, CU에 대해 FRUC 병합 모드를 사용하는지 여부에 대한 판정은 정상적인 병합 후보에 대해 행해지는 RD 비용 선택에 기초한다. 즉, 2개의 매칭 모드(양방향 매칭 및 템플릿 매칭)는 둘 모두, RD 비용 선택을 사용하여 CU에 대해 체크된다. 최소 비용을 발생시키는 것은 다른 CU 모드들과 추가로 비교된다. FRUC 매칭 모드가 가장 효율적인 것인 경우, FRUC 플래그가 CU에 대해 참으로 설정되고, 관련 매칭 모드가 사용된다.

[0147] FRUC 병합 모드에서의 모션 도출 프로세스는 2개의 단계를 갖는다. CU 레벨 모션 탐색이 먼저 수행된 후에, 서브-CU 레벨 모션 리파인먼트가 후속된다. CU 레벨에서, 초기 모션 벡터는 양방향 매칭 또는 템플릿 매칭에 기초하여 전체 CU에 대해 도출된다. 먼저, MV 후보들의 리스트가 생성되고, 최소 매칭 비용을 발생시키는 후보가 추가 CU 레벨 리파인먼트에 대한 시작 포인트로서 선택된다. 이어서, 시작 포인트 주위에서 양방향 매칭 또는 템플릿 매칭에 기초한 로컬 탐색이 수행되고, 최소 매칭 비용을 발생시키는 MV가 전체 CU에 대한 MV로서 취해진다. 후속하여, 모션 정보는 도출된 CU 모션 벡터들을 시작 포인트들로 하여 서브-CU 레벨에서 추가로 리파이닝된다(refined).

[0148] 예컨대, WxH CU 모션 정보 도출을 위해 다음의 도출 프로세스가 수행된다. 제1 단계에서, 전체 WxH CU에 대한 MV가 도출된다. 제2 스테이지에서, CU는 MxM 서브-CU들로 추가로 분할된다. M의 값은 (16)에서와 같이 계산되고, D는 JEM에서 디폴트로 3으로 설정된 미리 정의된 분할 깊이이다. 이어서, 각각의 서브-CU에 대한 MV가 도출된다.

$$M = \max\{4, \min\{\frac{M}{2^D}, \frac{N}{2^D}\}\} \quad (1)$$

[0150] 도 21에 도시된 바와 같이, 양방향 매칭은, 2개의 상이한 참조 픽처 내의 현재 CU의 모션 궤적을 따르는 2개의 블록 사이의 가장 근접한 매칭을 발견함으로써, 현재 CU의 모션 정보를 도출하기 위해 사용된다. 연속적인 모션 궤적의 가정 하에서, 2개의 참조 블록을 지적하는 모션 벡터들 MV0 및 MV1은 현재 픽처와 2개의 참조 픽처 사이의 시간적 거리들(즉, TD0 및 TD1)에 비례할 것이다. 특별한 경우로서, 현재 픽처가 시간적으로 2개의 참조 픽처 사이에 있고, 현재 픽처로부터 2개의 참조 픽처까지의 시간적 거리가 동일할 때, 양방향 매칭은 미리 기반 양방향 MV가 된다.

[0151] 도 22에 도시된 바와 같이, 템플릿 매칭은, 현재 픽처 내의 템플릿(현재 CU의 상단 및/또는 좌측 이웃 블록들)과 참조 픽처 내의 블록(템플릿과 동일한 크기) 사이의 가장 근접한 매칭을 발견함으로써, 현재 CU의 모션 정보

를 도출하기 위해 사용된다. 전술된 FRUC 병합 모드 이외에, 템플릿 매칭은 AMVP 모드에 또한 적용된다. JEM에서, HEVC에서 행해지는 바와 같이, AMVP는 2개의 후보를 갖는다. 템플릿 매칭 방법에 의해, 새로운 후보가 도출된다. 템플릿 매칭에 의해 새로 도출된 후보가 제1 기존 AMVP 후보와 상이한 경우, 이는 AMVP 후보 리스트의 맨 처음에 삽입되고, 이어서, 리스트 크기는 2로 설정된다(이는 제2 기존 AMVP 후보의 제거를 의미함). AMVP 모드에 적용될 때, CU 레벨 탐색만이 적용된다.

### [0152] 2.3.3.1 CU 레벨 MV 후보 세트

[0153] CU 레벨에서의 MV 후보 세트는 다음의 것들로 구성된다:

[0154] (i) 현재 CU가 AMVP 모드에 있는 경우 본래의 AMVP 후보들,

[0155] (ii) 모든 병합 후보들,

[0156] (iii) 보간된 MV 필드 내의 여러 개의 MV,

[0157] (iv) 상단 및 좌측 이웃 모션 벡터들

[0158] 양방향 매칭을 사용할 때, 병합 후보의 각각의 유효 MV는 양방향 매칭을 가정하여 MV 쌍을 생성하기 위한 입력으로서 사용된다. 예컨대, 병합 후보의 하나의 유효 MV는 참조 리스트 A에서 (MVa, refa)이다. 이어서, 그의 쌍을 이루는 양방향 MV의 참조 픽처 refb가 다른 참조 리스트 B에서 발견되고, 그에 따라, refa 및 refb는 시간적으로 현재 픽처의 상이한 측들에 있다. 이러한 refb가 참조 리스트 B에서 이용가능하지 않은 경우, refb는 refa와 상이한 참조로서 결정되고, 현재 픽처까지의 그의 시간적 거리는 리스트 B에서 최소 시간적 거리이다. refb가 결정된 후에, 현재 픽처와 refa, refb 사이의 시간적 거리에 기초하여, MVa를 스케일링함으로써, MVb가 도출된다.

[0159] 보간된 MV 필드로부터의 4개의 MV가 또한, CU 레벨 후보 리스트에 추가된다. 더 구체적으로, 현재 CU의 위치 (0, 0), (W/2, 0), (0, H/2), 및 (W/2, H/2)에서 보간된 MV들이 추가된다.

[0160] FRUC가 AMVP 모드에서 적용될 때, 본래의 AMVP 후보들이 또한 CU 레벨 MV 후보 세트에 추가된다.

[0161] CU 레벨에서, AMVP CU들에 대한 최대 15개의 MV 및 병합 CU들에 대한 최대 13개의 MV가 후보 리스트에 추가된다.

### [0162] 2.3.3.2 서브-CU 레벨 MV 후보 세트

[0163] 서브-CU 레벨에서의 MV 후보 세트는 다음의 것들로 구성된다:

[0164] (i) CU-레벨 탐색으로부터 결정된 MV,

[0165] (ii) 상단, 좌측, 상단-좌측, 및 상단-우측 이웃 MV들,

[0166] (iii) 참조 픽처들로부터의 공동-위치된 MV들의 스케일링된 버전들,

[0167] (iv) 최대 4개의 ATMVP 후보,

[0168] (v) 최대 4개의 STMVP 후보

[0169] 참조 픽처들로부터의 스케일링된 MV들은 다음과 같이 도출된다. 리스트들 둘 모두 내의 모든 참조 픽처가 순회된다(traversed). 참조 픽처 내의 서브-CU의 공동-위치된 위치에서의 MV들은 시작 CU 레벨 MV의 참조로 스케일링된다.

[0170] ATMVP 및 STMVP 후보들은 4개의 제1 후보로 제한된다.

[0171] 서브-CU 레벨에서, 최대 17개의 MV가 후보 리스트에 추가된다.

### [0172] 2.3.3.3 보간된 MV 필드의 생성

[0173] 프레임을 코딩하기 전에, 보간된 모션 필드가 단방향(unilateral) ME에 기초하여 전체 픽처에 대해 생성된다. 이어서, 모션 필드는 CU 레벨 또는 서브-CU 레벨 MV 후보들로서 나중에 사용될 수 있다.

[0174] 먼저, 참조 리스트들 둘 모두 내의 각각의 참조 픽처들의 모션 필드가 4x4 블록 레벨에서 순회된다. 각각의 4x4 블록에 대해, (도 23에 도시된 바와 같이) 블록과 연관된 모션이 현재 픽처 내의 4x4 블록을 통과하고, 블록이 어떠한 보간된 모션도 배정받지 않은 경우, 참조 블록의 모션은 (HEVC에서 TMVP의 MV 스케일링의 것과 동



일한 방식으로) 시간적 거리 TD0 및 TD1에 따라 현재 픽처로 스케일링되고, 스케일링된 모션은 현재 프레임 내의 블록에 지정된다. 스케일링된 MV가 4x4 블록에 지정되지 않은 경우, 블록의 모션은 보간된 모션 필드에서 이용불가능한 것으로 마킹된다.

#### [0175] 2.3.3.4 보간 및 매칭 비용

[0176] 모션 벡터가 프랙션(fractional) 샘플 위치를 지칭할 때, 모션 보상된 보간이 필요하다. 복잡도를 감소시키기 위해, 정규 8-탭 HEVC 보간 대신에 쌍선형 보간이 양방향 매칭과 템플릿 매칭 둘 모두에 대해 사용된다.

[0177] 매칭 비용의 계산은 상이한 단계들에서 다소 상이하다. CU 레벨에서 후보 세트로부터 후보를 선택할 때, 매칭 비용은 양방향 매칭 또는 템플릿 매칭의 절대 차이 합(SAD)이다. 시작 MV가 결정된 후에, 서브-CU 레벨 탐색에서의 양방향 매칭의 매칭 비용 C는 다음과 같이 계산된다:

$$C = SAD + w \cdot (|MV_x - MV_x^s| + |MV_y - MV_y^s|) \quad (2)$$

[0179] 여기서, w는 경험적으로 4로 설정된 가중 인자이고, MV 및  $MV^s$ 는 각각 현재 MV 및 시작 MV를 표시한다. SAD는 서브-CU 레벨 탐색에서의 템플릿 매칭의 매칭 비용으로서 여전히 사용된다.

[0180] FRUC 모드에서, MV는 루마 샘플들만을 사용하여 도출된다. 도출된 모션은 MC 인터 예측을 위해 루마와 크로마 둘 모두에 대해 사용될 것이다. MV가 판정된 후에, 루마에 대한 8-탭 보간 필터 및 크로마에 대한 4-탭 보간 필터를 사용하여 최종 MC가 수행된다.

#### [0181] 2.3.3.5 MV 리파인먼트

[0182] MV 리파인먼트는 양방향 매칭 비용 또는 템플릿 매칭 비용의 기준에 의한 패턴 기반 MV 탐색이다. JEM에서, 2개의 탐색 패턴 - 각각 CU 레벨 및 서브-CU 레벨에서의 MV 리파인먼트를 위한 UCBDS(unrestricted center-biased diamond search) 및 적응적 교차 탐색이 지원된다. CU 및 서브-CU 레벨 MV 리파인먼트 둘 모두에 대해, MV는 쿼터 루마 샘플 MV 정확도로 직접적으로 탐색되고, 그 후에, 1/8 루마 샘플 MV 리파인먼트가 후속된다. CU 및 서브-CU 단계에 대한 MV 리파인먼트의 탐색 범위는 8개의 루마 샘플과 동일하게 설정된다.

#### [0183] 2.3.3.6 템플릿 매칭 FRUC 병합 모드에서의 예측 방향의 선택

[0184] 양방향 매칭 병합 모드에서, 양방향 예측이 항상 적용되는데, 이는 2개의 상이한 참조 픽처 내의 현재 CU의 모션 궤적을 따르는 2개의 블록 사이의 가장 근접한 매칭에 기초하여, CU의 모션 정보가 도출되기 때문이다. 템플릿 매칭 병합 모드의 경우 그러한 제한은 없다. 템플릿 매칭 병합 모드에서, 인코더는 list0으로부터의 단방향 예측, list1로부터의 단방향 예측, 또는 CU에 대한 양방향 예측 중에서 선택할 수 있다. 선택은 다음과 같은 템플릿 매칭 비용에 기초한다:

[0185]  $costBi \leq factor * \min(cost0, cost1)$ 인 경우,

[0186] 양방향 예측이 사용되고;

[0187] 그렇지 않고,  $cost0 \leq cost1$ 인 경우,

[0188] list0으로부터의 단방향 예측이 사용되고;

[0189] 그렇지 않은 경우,

[0190] list1로부터의 단방향 예측이 사용된다;

[0191] 여기서, cost0은 list0 템플릿 매칭의 SAD이고, cost1은 list1 템플릿 매칭의 SAD이고, costBi는 양방향 예측 템플릿 매칭의 SAD이다. factor의 값은 1.25와 동일하고, 이는 선택 프로세스가 양방향 예측으로 편향되는 것을 의미한다. 인터 예측 방향 선택은 CU 레벨 템플릿 매칭 프로세스에만 적용된다.

#### [0192] 2.3.4 디코더 측 모션 벡터 리파인먼트

[0193] 양방향 예측 동작에서, 하나의 블록 구역의 예측을 위해, 각각 list0의 모션 벡터(MV) 및 list1의 MV를 사용하여 형성된 2개의 예측 블록이 단일 예측 신호를 형성하기 위해 조합된다. 디코더 측 모션 벡터 리파인먼트(DMVR) 방법에서, 양방향 예측의 2개의 모션 벡터는 양방향 템플릿 매칭 프로세스에 의해 추가로 리파인링된다. 추가적인 모션 정보의 송신 없이, 리파인링된 MV를 획득하도록, 참조 픽처들 내의 재구성 샘플들과 양방향 템플릿 사이의 왜곡 기반 탐색을 수행하기 위해, 디코더에 양방향 템플릿 매칭이 적용된다.

- [0194] DMVR에서, 도 23에 도시된 바와 같이, 각각 list0의 초기 MV0 및 list1의 MV1로부터의 2개의 예측 블록의 가중된 조합(즉, 평균)으로서 양방향 템플릿이 생성된다. 템플릿 매칭 동작은 참조 픽처 내의 (초기 예측 블록 주위의) 샘플 구역과 생성된 템플릿 사이의 비용 측정들을 계산하는 것으로 구성된다. 2개의 참조 픽처 각각에 대해, 최소 템플릿 비용을 산출하는 MV는 본래의 MV를 대체하기 위한 그 리스트의 업데이트된 MV로서 고려된다. JEM에서, 각각의 리스트에 대해 9개의 MV 후보가 탐색된다. 9개의 MV 후보는 본래의 MV, 및 수평 또는 수직 방향 또는 이들 둘 모두로 본래의 MV에 대해 하나의 루마 샘플 오프셋을 갖는 8개의 주위 MV를 포함한다. 마지막으로, 최종 양방향 예측 결과들을 생성하기 위해, 도 24에 도시된 바와 같은 2개의 새로운 MV, 즉 MV0' 및 MV1'이 사용된다. 절대 차이 합(SAD)이 비용 측정으로서 사용된다.
- [0195] DMVR은, 추가적인 신택스 요소들의 송신 없이, 과거의 참조 픽처로부터의 하나의 MV 및 미래의 참조 픽처로부터의 다른 MV를 이용하는 양방향 예측의 병합 모드에 대해 적용된다. JEM에서, LIC, 아핀(affine) 모션, FRUC, 또는 서브-CU 병합 후보가 CU에 대해 인에이블될 때, DMVR은 적용되지 않는다.
- [0196] **2.3.5 양방향 매칭 리파인먼트를 이용하는 병합/스킵 모드**
- [0197] 병합 후보 리스트는 먼저, 이용가능한 후보의 수가 19의 최대 후보 크기에 도달할 때까지, 중복 체크를 이용하여, 후보 리스트 내에 공간적 이웃 및 시간적 이웃 블록들의 모션 벡터들 및 참조 인덱스들을 삽입함으로써 구성된다. 병합/스킵 모드에 대한 병합 후보 리스트는, 미리 정의된 삽입 순서에 따라, 공간적 후보들(도 11), 시간적 후보들, 아핀 후보들, 고급 시간적 MVP(ATMVP) 후보, 공간-시간적 MVP(STMVP) 후보, 및 HEVC에서 사용되는 추가적인 후보들(조합된 후보들 및 제로 후보들)을 삽입함으로써 구성된다:
- [0198] - 블록 1-4에 대한 공간적 후보들.
  - [0199] - 블록들 1-4에 대한 외삽된 아핀 후보들.
  - [0200] - ATMVP.
  - [0201] - STMVP.
  - [0202] - 가상 아핀 후보.
  - [0203] - 공간적 후보(블록 5)(이용가능한 후보의 수가 6개보다 더 적을 때에만 사용됨).
  - [0204] - 외삽된 아핀 후보(블록 5).
  - [0205] - 시간적 후보(HEVC에서와 같이 도출됨).
  - [0206] - 비-인접 공간적 후보들에 이어지는 외삽된 아핀 후보(도 25에 도시된 바와 같은 블록들 6 내지 49).
  - [0207] - 조합된 후보들.
  - [0208] - 제로 후보들.
  - [0209] IC 플래그들이 STMVP 및 아핀을 제외하고 병합 후보들로부터 또한 승계된다는 점에 유의한다. 더욱이, 처음 4개의 공간적 후보에 대해, 양방향 예측 후보들은 단방향 예측을 갖는 후보들 전에 삽입된다.
  - [0210] 현재 블록과 연결되지 않은 블록들이 액세스될 수 있다. 비-인접 블록이 비-인트라 모드로 코딩되는 경우, 연관된 모션 정보는 추가적인 병합 후보로서 추가될 수 있다.
  - [0211] **3. 본원에서 개시되는 실시예들에 의해 해소되는 문제들의 예들**
  - [0212] 현재 HEVC 설계는, 모션 정보를 더 양호하게 코딩하기 위해, 현재 블록과 그의 이웃 블록들(현재 블록 옆에 있음)의 상관을 취할 수 있다. 그러나, 이웃 블록들이 상이한 모션 궤적들을 갖는 상이한 객체들에 대응하는 것이 가능하다. 이 경우, 그의 이웃 블록들로부터의 예측은 효율적이지 않다.
  - [0213] 비-인접 블록들의 모션 정보로부터의 예측은, 하드웨어 구현에 대한 복잡도를 상당히 증가시키는, 모든 모션 정보(전형적으로는 4x4 레벨)를 캐시 내에 저장하는 것의 대가로, 추가적인 코딩 이득을 가져올 수 있다.
  - [0214] **4. 일부 예들**
  - [0215] 아래의 예들은 일반적인 개념들을 설명하기 위한 예들로서 고려되어야 한다. 이러한 예들은 협소하게 해석되지 않아야 한다. 게다가, 이러한 예들은 임의의 방식으로 조합될 수 있다.

- [0216] 일부 실시예들은 블록의 모션 정보를 예측하기 위해 적어도 하나의 모션 후보가 저장된 하나 이상의 룩업 테이블을 사용할 수 있다. 실시예들은 룩업 테이블에 저장된 모션 정보의 세트를 표시하기 위해 모션 후보를 사용할 수 있다. 종래의 AMVP 또는 병합 모드들에 대해, 실시예들은 모션 정보를 저장하기 위해 AMVP 또는 병합 후보들을 사용할 수 있다.
- [0217] 아래의 예들은 일반적인 개념들을 설명한다.
- [0218] **룩업 테이블들의 예들**
- [0219] 예 A1: 각각의 룩업 테이블은 하나 이상의 모션 후보를 포함할 수 있고, 여기서, 각각의 후보는 그의 모션 정보와 연관된다.
- [0220] a. 여기서, 모션 후보의 모션 정보는 예측 방향, 참조 인덱스들/픽처들, 모션 벡터들, LIC 플래그, 아핀 플래그, MVD 정밀도, MVD 값들 중 일부 또는 전부를 포함할 수 있다.
- [0221] b. 모션 정보는 모션 정보가 유래한 곳을 표시하기 위해 블록 위치 정보 및/또는 블록 형상을 더 포함할 수 있다.
- [0222] c. 각각의 룩업 테이블에 대해 카운터가 추가로 지정될 수 있다.
- [0223] i. 카운터는 픽처/슬라이스/LCU(CTU) 행/타일의 인코딩/디코딩의 시작 시에 제로로 초기화될 수 있다.
- [0224] ii. 일 예에서, 카운터는 CTU/CTB/CU/CB/PU/특정 구역 크기(예컨대, 8x8 또는 16x16)를 인코딩/디코딩한 후에 업데이트될 수 있다.
- [0225] iii. 일 예에서, 카운터는 하나의 후보가 룩업 테이블에 추가될 때마다 1씩 증가된다.
- [0226] iv. 일 예에서, 카운터는 테이블 크기(허용되는 모션 후보의 수)보다 더 크지 않아야 한다.
- [0227] v. 대안적으로, 카운터는 얼마나 많은 모션 후보들이 룩업 테이블들에 추가되려고 시도되었는지(이들 중 일부는 룩업 테이블에 있었지만 나중에 테이블로부터 제거될 수 있음)를 표시하기 위해 사용될 수 있다. 이 경우, 카운터는 테이블 크기보다 더 클 수 있다.
- [0228] d. 테이블 크기(허용되는 모션 후보의 수) 및/또는 테이블의 수는 고정될 수 있거나 또는 적응적일 수 있다. 테이블 크기는 모든 테이블에 대해 동일할 수 있거나, 또는 상이한 테이블들에 대해 상이할 수 있다.
- [0229] i. 대안적으로, 상이한 크기들이 상이한 룩업 테이블들에 대해 사용될 수 있다(예컨대, 1 또는 2).
- [0230] ii. 일 예에서, 테이블 크기들 및/또는 테이블의 수는 미리 정의될 수 있다.
- [0231] iii. 일 예에서, 테이블 크기들 및/또는 테이블의 수는 비디오 파라미터 세트(VPS), 시퀀스 파라미터 세트(SPS), 픽처 파라미터 세트(PPS), 슬라이스 헤더, 타일 헤더, 코딩 트리 유닛(CTU), 코딩 트리 블록(CTB), 코딩 유닛(CU), 또는 예측 유닛(PU), 다수의 CTU/CTB/CU/PU를 커버하는 구역에서 시그널링될 수 있다.
- [0232] iv. 테이블 크기 및/또는 테이블의 수는 추가로, 슬라이스 타입, 픽처의 시간적 계층 인덱스, 하나의 슬라이스와 가장 근접한 인트라 슬라이스 사이의 POC(picture order count) 거리에 의존할 수 있다.
- [0233] e. 코딩 스레드에 대해 사용되는 N개의 테이블이 있는 것으로 가정하면, 슬라이스를 코딩하기 위해 N\*P개의 테이블이 요구될 수 있고, 여기서, P는 LCU 행의 수 또는 타일의 수를 표시한다.
- [0234] i. 대안적으로, 슬라이스를 코딩하기 위해 P개의 테이블만이 요구될 수 있고, 여기서, P는 LCU 행의 수를 표시하고, 여기서, 각각의 LCU 행은, 타일이 디스에이블될 때, N이 1보다 더 클 수 있더라도, 하나의 룩업 테이블만을 사용한다.
- [0235] **LUT들의 선택**
- [0236] 예 B1: 블록을 코딩하기 위해, 하나의 룩업 테이블로부터의 모션 후보들 중 일부 또는 전부가 순서대로 체크될 수 있다. 블록을 코딩하는 동안 하나의 모션 후보가 체크될 때, 이는 모션 후보 리스트(예컨대, AMVP, 병합 후보 리스트들)에 추가될 수 있다.
- [0237] a. 대안적으로, 다수의 룩업 테이블로부터의 모션 후보들이 순서대로 체크될 수 있다.
- [0238] b. 룩업 테이블 인덱스들은 CTU, CTB, CU 또는 PU, 또는 다수의 CTU/CTB/CU/PU를 커버하는 구역에서 시그널링될

수 있다.

- [0239] 예 B2: 룩업 테이블들의 선택은 블록의 위치에 의존할 수 있다.
- [0240] a. 이는 블록을 커버하는 CTU 어드레스에 의존할 수 있다. 여기서, 아이디어를 예시하기 위한 예를 위해 2개의 룩업 테이블(DLUT(Dual Look Up Tables))을 이용한다:
- [0241] i. 블록이 CTU 행 내의 처음 M개의 CTU 중 하나에 위치된 경우, 블록을 코딩하기 위해 제1 룩업 테이블이 활용될 수 있는 한편, CTU 행 내의 나머지 CTU들에 위치된 블록들에 대해 제2 룩업 테이블이 활용될 수 있다.
- [0242] ii. 블록이 CTU 행 내의 처음 M개의 CTU 중 하나에 위치된 경우, 블록을 코딩하기 위해 제1 룩업 테이블의 모션 후보들이 먼저 체크될 수 있고, 제1 테이블에 충분한 후보들이 있지 않은 경우, 제2 룩업 테이블이 추가로 활용될 수 있다. 한편, CTU 행 내의 나머지 CTU들에 위치된 블록들에 대해 제2 룩업 테이블이 활용될 수 있다.
- [0243] iii. 대안적으로, CTU 행 내의 나머지 CTU들에 위치된 블록들에 대해, 블록을 코딩하기 위해 제2 룩업 테이블의 모션 후보들이 먼저 체크될 수 있고, 제2 테이블에 충분한 후보들이 있지 않은 경우, 제1 룩업 테이블이 추가로 활용될 수 있다.
- [0244] b. 이는 블록의 위치와 하나 이상의 룩업 테이블 내의 하나의 모션 후보와 연관된 위치 사이의 거리에 의존할 수 있다.
- [0245] iv. 일 예에서, 하나의 모션 후보가 코딩될 블록까지의 더 작은 거리와 연관되는 경우, 이는 다른 모션 후보에 비해 더 먼저 체크될 수 있다.
- [0246] **룩업 테이블들의 사용**
- [0247] 예 C1: 체크될 룩업 테이블 내의 모션 후보의 총 수는 미리 정의될 수 있다.
- [0248] a. 이는 추가로, 코딩된 정보, 블록 크기, 블록 형상 등에 의존할 수 있다. 예컨대, AMVP 모드에 대해, m개의 모션 후보만이 체크될 수 있는 한편, 병합 모드에 대해, n개의 모션 후보가 체크될 수 있다(예컨대,  $m = 2$ ,  $n = 44$ ).
- [0249] b. 일 예에서, 체크될 모션 후보의 총 수는 비디오 파라미터 세트(VPS), 시퀀스 파라미터 세트(PS), 픽처 파라미터 세트(PPS), 슬라이스 헤더, 타일 헤더, 코딩 트리 유닛(CTU), 코딩 트리 블록(CTB), 코딩 유닛(CU), 또는 예측 유닛(PU), 다수의 CTU/CTB/CU/PU를 커버하는 구역에서 시그널링될 수 있다.
- [0250] 예 C2: 룩업 테이블에 포함된 모션 후보(들)는 블록에 의해 직접적으로 승계될 수 있다.
- [0251] a. 이들은 병합 모드 코딩에 대해 사용될 수 있고, 즉, 모션 후보들은 병합 후보 리스트 도출 프로세스에서 체크될 수 있다.
- [0252] b. 이들은 아핀 병합 모드 코딩에 대해 사용될 수 있다.
- [0253] i. 룩업 테이블 내의 모션 후보는 그의 아핀 플래그가 1인 경우 아핀 병합 후보로서 추가될 수 있다.
- [0254] c. 룩업 테이블들 내의 모션 후보들의 체크는 다음과 같은 경우 인에이블될 수 있다:
- [0255] i. TMVP 후보를 삽입한 후에, 병합 후보 리스트가 가득 차지 않는다;
- [0256] ii. 공간적 병합 후보 도출을 위해 특정 공간적 이웃 블록을 체크한 후에, 병합 후보 리스트가 가득 차지 않는다;
- [0257] iii. 모든 공간적 병합 후보들 후에, 병합 후보 리스트가 가득 차지 않는다;
- [0258] iv. 조합된 양방향 예측 병합 후보들 후에, 병합 후보 리스트가 가득 차지 않는다;
- [0259] v. 다른 코딩 방법들(예컨대, JEM 설계 또는 HEVC 설계의 병합 도출 프로세스)로부터 병합 후보 리스트 내에 배치된 공간적 또는 시간적(예컨대, 인접 공간 및 비-인접 공간, TMVP, STMVP, ATMVP 등을 포함함) 병합 후보의 수가 최대 허용 병합 후보들 마이너스 주어진 임계치보다 더 작은 경우.
- [0260] 1. 일 예에서, 임계치는 1 또는 0으로 설정된다.
- [0261] 2. 대안적으로, 임계치는 SPS/PPS/시퀀스, 픽처, 슬라이스 헤더/타일에서 시그널링될 수 있거나 또는 미리 정의될 수 있다.

- [0262] 3. 대안적으로, 임계치는 블록마다 적응적으로 변경될 수 있다. 예컨대, 이는, 블록 크기/블록 형상/슬라이스 타입과 같은 코딩된 블록 정보에 의존할 수 있고/있거나, 이용가능한 공간적 또는 시간적 병합 후보의 수에 의존할 수 있다.
- [0263] 4. 다른 예에서, 병합 후보 리스트 내에 배치된 특정 종류의 병합 후보의 수가 최대 허용 병합 후보들 마이너스 주어진 임계치보다 더 작은 경우, "특정 종류의 병합 후보들"은 HEVC에서와 같은 공간적 후보들 또는 비-인접 병합 후보들일 수 있다.
- [0264] vi. 병합 후보 리스트에 모션 후보를 추가하기 전에 프루닝(pruning)이 적용될 수 있다.
- [0265] 1. 일 예에서, 모션 후보는 병합 후보 리스트 내의 다른 코딩 방법들로부터의 이용가능한 공간적 또는 시간적 (예컨대, 인접 공간적 및 비-인접 공간적, TMVP, STMVP, ATMVP 등을 포함함) 병합 후보들 중 일부 또는 전부로 프루닝될 수 있다.
- [0266] 2. 모션 후보는 서브-블록 기반 모션 후보들, 예컨대, ATMVP, STMVP로 프루닝되지 않을 수 있다.
- [0267] 3. 일 예에서, 현재 모션 후보는 병합 후보 리스트 내의 (현재 모션 후보 전에 삽입된) 이용가능한 모션 후보들 중 일부 또는 전부로 프루닝될 수 있다.
- [0268] 4. 모션 후보들과 관련된 프루닝 동작의 수(즉, 모션 후보들이 병합 리스트 내의 다른 후보들과 비교될 필요가 있는 횟수)는 이용가능한 공간적 또는 시간적 병합 후보의 수에 의존할 수 있다. 예컨대, 새로운 모션 후보를 체크할 때, 병합 리스트에 이용가능한 M개의 후보가 있는 경우, 새로운 모션 후보는 처음 K개( $K \leq M$ )의 후보와만 비교될 수 있다. 프루닝 함수가 거짓(예컨대, 처음 K개의 후보 중 어느 것과도 동일하지 않음)을 리턴하는 경우, 새로운 모션 후보는 모든 M개의 후보와 상이한 것으로 고려되고, 병합 후보 리스트에 추가될 수 있다. 일 예에서, K는  $\min(K, 2)$ 로 설정된다.
- [0269] 5. 일 예에서, 새로 부가된 모션 후보는 병합 후보 리스트 내의 처음 N개의 후보와만 비교된다. 예컨대,  $N = 3, 4$ , 또는 5이다. N은 인코더로부터 디코더로 시그널링될 수 있다.
- [0270] 6. 일 예에서, 체크될 새로운 모션 후보는 병합 후보 리스트 내의 마지막 N개의 후보와만 비교된다. 예컨대,  $N = 3, 4$ , 또는 5이다. N은 인코더로부터 디코더로 시그널링될 수 있다.
- [0271] 7. 일 예에서, 테이블로부터의 새로운 모션 후보와 비교될, 리스트에 이전에 추가된 후보들을 선택하기 위한 방법은, 이전에 추가된 후보들이 도출된 위치에 의존할 수 있다.
- [0272] a. 일 예에서, 룩업 테이블 내의 모션 후보는 주어진 시간적 및/또는 공간적 이웃 블록으로부터 도출된 후보들과 비교될 수 있다.
- [0273] b. 일 예에서, 룩업 테이블 내의 모션 후보들의 상이한 엔트리들은 상이한 이전에 추가된 후보들(즉, 상이한 위치들로부터 도출됨)과 비교될 수 있다.
- [0274] 예 C3: 룩업 테이블에 포함된 모션 후보(들)는 블록의 모션 정보를 코딩하기 위한 예측자로서 사용될 수 있다.
- [0275] a. 이들은 AMVP 모드 코딩에 대해 사용될 수 있고, 즉, 모션 후보들은 AMVP 후보 리스트 도출 프로세스에서 체크될 수 있다.
- [0276] b. 룩업 테이블들 내의 모션 후보들의 체크는 다음과 같은 경우 인에이블될 수 있다:
- [0277] i. TMVP 후보를 삽입한 후에, AMVP 후보 리스트가 가득 차지 않는다;
- [0278] ii. TMVP 후보를 삽입하기 직전에, 공간적 이웃들로부터 선택하고 프루닝한 후에, AMVP 후보 리스트가 가득 차지 않는다;
- [0279] iii. 스케일링 없이 상측 이웃 블록들로부터 AMVP 후보가 없는 경우 및/또는 스케일링 없이 좌측 이웃 블록들로부터 AMVP 후보가 없는 경우.
- [0280] iv. AMVP 후보 리스트에 모션 후보를 추가하기 전에 프루닝이 적용될 수 있다.
- [0281] v. 불릿(bullet) 5.(3)(4)에서 언급된 바와 유사한 규칙들이 AMVP 모드에 적용될 수 있다.
- [0282] c. 현재 참조 픽처와 동일한 참조 픽처를 갖는 모션 후보들이 체크된다.
- [0283] i. 대안적으로, 추가하여, 현재 참조 픽처와 상이한 참조 픽처들을 갖는 모션 후보들(MV 스케일링됨)이 또한 체



크된다.

- [0284] ii. 대안적으로, 현재 참조 픽처와 동일한 참조 픽처를 갖는 모든 모션 후보들이 먼저 체크된 후에, 현재 참조 픽처와 상이한 참조 픽처를 갖는 모션 후보들이 체크된다.
- [0285] iii. 대안적으로, 모션 후보들은 병합에서 동일하게 체크된다.
- [0286] 예 C4: 룩업 테이블 내의 모션 후보들의 체크 순서는 다음과 같이 정의된다( $K(K \geq 1)$ 개의 모션 후보가 체크되게 허용되는 것으로 가정함):
- [0287] a. 룩업 테이블 내의 마지막  $K$ 개의 모션 후보는, 예컨대, LUT에 대한 엔트리 인덱스의 내림차순으로 이루어진다.
- [0288] b. 처음  $K$ 개의 후보는, 예컨대, LUT에 대한 엔트리 인덱스들의 내림차순으로 이루어지고, 여기서,  $L$ 은  $K \geq L$ 일 때 룩업 테이블 크기이다.
- [0289] c.  $K \geq L$ 일 때, 룩업 테이블 내의 모든 후보들( $L$ 개의 후보)은 순서에 기초한다. 일 예에서, 테이블 내의 처음  $K$ 개의 후보는 LUT에 대한 엔트리 인덱스들의 내림차순으로 체크되고, 이어서, 마지막( $L - K$ ) 후보들이 엔트리 인덱스들의 내림차순으로 체크된다.
- [0290] d. 대안적으로, 게다가, 모션 후보 인덱스들의 내림차순에 기초한다.
- [0291] e. 대안적으로, 게다가, 모션 후보 인덱스들의 오름차순에 기초한다.
- [0292] f. 대안적으로, 모션 후보들과 연관된 위치들과 현재 블록의 거리와 같은 후보 정보에 기초하여  $K$ 개의 모션 후보를 선택한다.
- [0293] i. 일 예에서,  $K$ 개의 가장 근접한 모션 후보가 선택된다.
- [0294] ii. 일 예에서, 후보 정보는 거리를 계산할 때 블록 형상을 추가로 고려할 수 있다.
- [0295] g. 일 예에서,  $L$ 개의 후보를 포함하는 테이블로부터의 모션 후보들 중  $K$ 개의 모션 후보의 체크 순서는: 순서대로  $a_0, a_0+T_0, a_0+T_0+T_1, a_0+T_0+T_1+T_2, \dots, a_0+T_0+T_1+T_2+\dots+T_{K-1}$ 과 동일한 인덱스를 갖는 그러한 후보들을 선택하는 것으로서 정의될 수 있고, 여기서,  $a_0$  및  $T_i$ ( $i$ 는  $0 \dots K-1$ 임)는 정수 값들이다.
- [0296] i. 일 예에서,  $a_0$ 은  $0$ (즉, 테이블 내의 모션 후보의 제1 엔트리)으로 설정된다. 대안적으로,  $a_0$ 은  $(K - L/K)$ 로 설정된다. 산술 연산 '/'는 결과를 제로로 절단하는 정수 나눗셈으로서 정의된다. 대안적으로,  $a_0$ 은  $0$ 과  $L/K$  사이의 임의의 정수로 설정된다.
- [0297] 1. 대안적으로,  $a_0$ 의 값은 현재 블록 및 이웃 블록들의 코딩 정보에 의존할 수 있다.
- [0298] ii. 일 예에서, 모든 간격들  $T_i$ ( $i$ 는  $0 \dots K-1$ 임)는  $L/K$ 와 같이 동일하다. 산술 연산 '/'는 결과를 제로로 절단하는 정수 나눗셈으로서 정의된다.
- [0299] iii. 일 예에서,  $(K, L, a_0, T_i)$ 는  $(4, 16, 0, 4)$ , 또는  $(4, 12, 0, 3)$  또는  $(4, 8, 0, 1)$  또는  $(4, 16, 3, 4)$  또는  $(4, 12, 2, 3)$ , 또는  $(4, 8, 1, 2)$ 로 설정된다.  $T_i$ 는 모든  $i$ 에 대해 동일하다.
- [0300] iv. 이러한 방법은  $K$ 가  $L$ 보다 더 작을 때에만 적용될 수 있다.
- [0301] v. 대안적으로, 게다가,  $K$ 가 임계치 이상일 때, 불릿 7.c.가 적용될 수 있다. 임계치는  $L$ 로서 정의될 수 있거나, 또는 이는  $K$ 에 의존할 수 있거나 또는 블록마다 적응적으로 변경될 수 있다. 일 예에서, 임계치는, 룩업 테이블로부터 새로운 모션 후보를 추가하기 전의, 리스트 내의 이용가능한 모션 후보의 수에 의존할 수 있다.
- [0302] h. 일 예에서,  $L$ 개의 후보를 포함하는 테이블로부터의 모션 후보들 중  $K$ 개의 모션 후보의 체크 순서는: 순서대로  $a_0, a_0-T_0, a_0-T_0-T_1, a_0-T_0-T_1-T_2, \dots, a_0-T_0-T_1-T_2-\dots-T_{K-1}$ 과 동일한 인덱스를 갖는 그러한 후보들을 선택하는 것으로서 정의될 수 있고, 여기서,  $a_0$  및  $T_i$ ( $i$ 는  $0 \dots K-1$ 임)는 정수 값들이다.
- [0303] i. 일 예에서,  $a_0$ 은  $L-1$ (즉, 테이블 내의 모션 후보의 마지막 엔트리)로 설정된다. 대안적으로,  $a_0$ 은  $L-1-L/K$ 와  $L-1$  사이의 임의의 정수로 설정된다.

- [0304] ii. 일 예에서, 모든 간격들  $T_i$  ( $i$ 는  $0 \dots K-1$ 임)는  $L/K$ 와 같이 동일하다.
- [0305] iii. 일 예에서,  $(K, L, a_0, T_i)$ 는  $(4, 16, L-1, 4)$ , 또는  $(4, 12, L-1, 3)$  또는  $(4, 8, L-1, 1)$  또는  $(4, 16, L-4, 4)$  또는  $(4, 12, L-3, 3)$ , 또는  $(4, 8, L-2, 2)$ 로 설정된다.  $T_i$ 는 모든  $i$ 에 대해 동일하다.
- [0306] iv. 이러한 방법은  $K$ 가  $L$ 보다 더 작을 때에만 적용될 수 있다. 대안적으로, 게다가,  $K$ 가 임계치 이상일 때, 블럭 7.c가 적용될 수 있다. 임계치는  $L$ 로서 정의될 수 있거나, 또는 이는  $K$ 에 의존할 수 있거나 또는 블록마다 적응적으로 변경될 수 있다. 일 예에서, 임계치는, 룩업 테이블로부터 새로운 모션 후보를 추가하기 전의, 리스트 내의 이용가능한 모션 후보의 수에 의존할 수 있다.
- [0307] i. 룩 테이블로부터 모션 후보를 얼마나 많이 및/또는 어떻게 선택할지는 블록 크기/블록 형상과 같은 코딩된 정보에 의존할 수 있다.
- [0308] i. 일 예에서, 더 작은 블록 크기에 대해, 마지막  $K$ 개의 모션 후보를 선택하는 대신에, 다른  $K$ 개의 모션 후보 (마지막 모션 후보로부터 시작하지 않음)가 선택될 수 있다.
- [0309] ii. 일 예에서, 코딩된 정보는 AMVP 또는 병합 모드일 수 있다.
- [0310] iii. 일 예에서, 코딩된 정보는 아핀 모드 또는 비-아핀 AMVP 모드 또는 비-아핀 병합 모드일 수 있다.
- [0311] iv. 일 예에서, 코딩된 정보는 아핀 AMVP(인터) 모드, 아핀 병합 모드 또는 비-아핀 AMVP 모드 또는 비-아핀 병합 모드일 수 있다.
- [0312] v. 일 예에서, 코딩된 정보는 현재 픽처 참조(Current Picture Reference)(CPR) 모드일 수 있거나 또는 CPR 모드가 아닐 수 있다.
- [0313] vi. 대안적으로, 룩업 테이블로부터 모션 후보들을 선택하기 위한 방법은 추가로, 룩업 테이블 내의 모션 후보의 수, 및/또는 룩업 테이블로부터 새로운 모션 후보를 추가하기 전의 리스트 내의 이용가능한 모션 후보의 수에 의존할 수 있다.
- [0314] j. 일 예에서, 체크될(즉, 병합/amvp 후보 리스트에 추가될 수 있는) 룩업 테이블 내의 모션 후보의 최대 수는 룩업 테이블 내의 이용가능한 모션 후보의 수( $N_{\text{availMCinLUT}}$ 에 의해 표시됨), 및/또는 추가될(미리 정의되거나 또는 시그널링될 수 있는) 최대 허용 모션 후보들( $NUM_{\text{maxMC}}$ 에 의해 표시됨), 및/또는 룩업 테이블로부터의 후보들을 체크하기 전의 후보 리스트 내의 이용가능한 후보의 수( $N_{\text{availC}}$ 에 의해 표시됨)에 의존할 수 있다.
- [0315] i. 일 예에서, 체크될 룩업 테이블 내의 모션 후보의 최대 수는  $(N_{\text{availMCinLUT}}, NUM_{\text{maxMC}}, N_{\text{availC}})$ 의 최소 값으로 설정된다.
- [0316] ii. 대안적으로, 체크될 룩업 테이블 내의 모션 후보의 최대 수는  $(N_{\text{availMCinLUT}}, NUM_{\text{maxMC}} - N_{\text{availC}})$ 의 최소 값으로 설정된다.
- [0317] iii. 일 예에서,  $N_{\text{availC}}$ 는 공간적 또는 시간적(인접 및/또는 비-인접) 이웃 블록들로부터 도출된 삽입된 후보의 수를 표시한다. 대안적으로, 게다가, (AMTVP, STMVP와 같은) 서브-블록 후보의 수는  $N_{\text{availC}}$ 에서 카운트되지 않는다.
- [0318] iv.  $NUM_{\text{maxMC}}$ 는 코딩된 모드에 의존할 수 있고, 예컨대, 병합 모드 및 AMVP 모드에 대해,  $NUM_{\text{maxMC}}$ 는 상이한 값들로 설정될 수 있다. 일 예에서, 병합 모드에 대해,  $NUM_{\text{maxMC}}$ 는 4, 6, 8, 10 등으로 설정될 수 있고, AMVP 모드에 대해,  $NUM_{\text{maxMC}}$ 는 1, 2, 4 등으로 설정될 수 있다.
- [0319] v. 대안적으로,  $NUM_{\text{maxMC}}$ 는 블록 크기, 블록 형상, 슬라이스 타입 등과 같은 다른 코딩된 정보에 의존할 수 있다.
- [0320] k. 상이한 룩업 테이블들의 체크 순서는 다음의 서브섹션에서 룩업 테이블들의 사용에서 정의된다.
- [0321] l. 체크 프로세스는 병합/AMVP 후보 리스트가 최대 허용 후보 수에 도달하면 종료될 것이다.
- [0322] m. 체크 프로세스는 병합/AMVP 후보 리스트가 최대 허용 후보 수 마이너스 임계치( $Th$ )에 도달하면 종료될 것이다. 일 예에서,  $Th$ 는 양의 정수 값, 예컨대, 1 또는 2 또는 3으로서 미리 정의될 수 있다. 대안적으로,  $Th$ 는

블록마다 적응적으로 변경될 수 있다. 대안적으로, Th는 SPS/PPS/슬라이스 헤더 등에서 시그널링될 수 있다. 대안적으로, Th는 추가로, 블록 형상/블록 크기/코딩된 모드들 등에 의존할 수 있다. 대안적으로, Th는 LUT들로부터 모션 후보들을 추가하기 전의 이용가능한 후보의 수에 의존할 수 있다.

- [0323] n. 대안적으로, 이는 추가된 모션 후보의 수가 최대 허용 모션 후보 수에 도달하면 종료될 것이다. 최대 허용 모션 후보 수는 시그널링 또는 미리 정의될 수 있다. 대안적으로, 최대 허용 모션 후보 수는 추가로, 블록 형상/블록 크기/코딩된 모드들 등에 의존할 수 있다.
- [0324] o. 체크되게 허용되는 모션 후보의 수(즉,  $K = L$ ) 뿐만 아니라 테이블 크기를 표시하기 위한 하나의 선택스 요소가 SPS, PPS, 슬라이스 헤더, 타일 헤더에서 시그널링될 수 있다.
- [0325] 예 C5: 블록의 모션 정보 코딩을 위한 룩업 테이블들의 사용의 인에이블/디스에이블은 SPS, PPS, 슬라이스 헤더, 타일 헤더, CTU, CTB, CU 또는 PU, 다수의 CTU/CTB/CU/PU를 커버하는 구역에서 시그널링될 수 있다.
- [0326] 예 C6: 룩업 테이블들로부터의 예측을 적용할지 여부는 추가로, 코딩된 정보에 의존할 수 있다. 블록에 대해 적용하지 않는 것으로 추론될 때, 예측의 표시들의 추가적인 시그널링이 스킵된다. 대안적으로, 블록에 대해 적용하지 않는 것으로 추론될 때, 룩업 테이블의 모션 후보들에 액세스할 필요가 없고, 관련 모션 후보들의 체크가 생략된다.
- [0327] a. 룩업 테이블들로부터의 예측을 적용할지 여부는 블록 크기/블록 형상에 의존할 수 있다. 일 예에서, 4x4, 8x4, 또는 4x8 블록들과 같은 더 작은 블록들에 대해, 룩업 테이블들로부터의 예측을 수행하는 것이 허용되지 않는다.
- [0328] b. 룩업 테이블들로부터의 예측을 적용할지 여부는 블록이 AMVP 또는 병합 모드로 코딩되는지에 의존할 수 있다. 일 예에서, AMVP 모드에 대해, 룩업 테이블들로부터의 예측을 수행하는 것이 허용되지 않는다.
- [0329] c. 룩업 테이블들로부터의 예측을 적용할지 여부는 블록이 아핀 모션 또는 다른 종류의 모션(이러테면, 병진 모션)으로 코딩되는 것에 의존할 수 있다. 일 예에서, 아핀 모드에 대해, 룩업 테이블들로부터의 예측을 수행하는 것이 허용되지 않는다.
- [0330] 예 C7: 이전에 코딩된 프레임들/슬라이스들/타일들 내의 룩업 테이블의 모션 후보들은 상이한 프레임/슬라이스/타일 내의 블록의 모션 정보를 예측하기 위해 사용될 수 있다.
- [0331] a. 일 예에서, 현재 블록의 참조 픽처들과 연관된 룩업 테이블들만이 현재 블록을 코딩하는 데 활용될 수 있다.
- [0332] b. 일 예에서, 현재 블록의 동일한 슬라이스 타입 및/또는 동일한 양자화 파라미터들을 갖는 픽처들과 연관된 룩업 테이블들만이 현재 블록을 코딩하는 데 활용될 수 있다.
- [0333] **룩업 테이블들의 업데이트**
- [0334] 예 D1: 모션 정보(즉, 인트라BC 모드, 인터 코딩된 모드)로 블록을 코딩한 후에, 하나 이상의 룩업 테이블이 업데이트될 수 있다.
- [0335] a. 일 예에서, 룩업 테이블을 업데이트할지 여부는 룩업 테이블들을 선택하기 위한 규칙들을 재사용할 수 있고, 예컨대, 룩업 테이블이 현재 블록을 코딩하기 위해 선택될 수 있을 때, 블록을 코딩/디코딩한 후에, 선택된 룩업 테이블이 추가로 업데이트될 수 있다.
- [0336] b. 업데이트된 룩업 테이블들은 코딩된 정보 및/또는 블록/LCU의 위치들에 기초하여 선택될 수 있다.
- [0337] c. 블록이 (AMVP 모드, 정상/아핀 인터 모드에 대한 MMVD 모드, 정상/아핀 인터 모드에 대한 AMVR 모드와 같이) 직접적으로 시그널링된 모션 정보로 코딩되는 경우, 블록에 대한 모션 정보가 룩업 테이블에 추가될 수 있다.
- [0338] i. 대안적으로, 블록이 임의의 리파인먼트를 이용하지 않은 공간적 이웃 블록(예컨대, 리파인먼트를 이용하지 않은 공간적 병합 후보)으로부터 직접적으로 승계된 모션 정보로 코딩되는 경우, 블록에 대한 모션 정보는 룩업 테이블에 추가되지 않아야 한다.
- [0339] ii. 대안적으로, 블록이 (DMVR, FRUC와 같이) 리파인먼트를 이용한 공간적 이웃 블록으로부터 직접적으로 승계된 모션 정보로 코딩되는 경우, 블록에 대한 모션 정보는 임의의 룩업 테이블에 추가되지 않아야 한다.
- [0340] iii. 대안적으로, 블록이 룩업 테이블에 저장된 모션 후보로부터 직접적으로 승계된 모션 정보로 코딩되는 경우, 블록에 대한 모션 정보는 임의의 룩업 테이블에 추가되지 않아야 한다.



- [0341] iv. 일 예에서, 이러한 모션 정보는 록업 테이블에, 이를테면, 테이블의 마지막 엔트리에, 또는 다음의 이용가능한 모션 후보를 저장하기 위해 사용되는 엔트리에 직접적으로 추가될 수 있다.
- [0342] v. 대안적으로, 그러한 모션 정보는 프루닝을 이용하지 않고, 예컨대, 임의의 프루닝을 이용하지 않고 록업 테이블에 직접적으로 추가될 수 있다.
- [0343] vi. 대안적으로, 그러한 모션 정보는 록업 테이블을 재순서화하기 위해 사용될 수 있다.
- [0344] vii. 대안적으로, 그러한 모션 정보는 제한된 프루닝으로(예컨대, 록업 테이블 내의 최근의 것과 비교하여) 록업 테이블을 업데이트하기 위해 사용될 수 있다.
- [0345] d. 블록 내의  $M(M \geq 1)$ 개의 대표적인 위치가 선택되고, 그 대표적인 위치와 연관된 모션 정보가 록업 테이블들을 업데이트하기 위해 사용된다.
- [0346] i. 일 예에서, 대표적인 위치는 블록 내의 4개의 코너 위치(예컨대, 도 26의 C0-C3) 중 하나로서 정의된다.
- [0347] ii. 일 예에서, 대표적인 위치는 블록 내의 중앙 위치(예컨대, 도 26의 Ca-Cd)로서 정의된다.
- [0348] iii. 블록에 대해 서브-블록 예측이 허용되지 않을 때,  $M$ 은 1로 설정된다.
- [0349] iv. 블록에 대해 서브-블록 예측이 허용될 때,  $M$ 은 1 또는 서브-블록의 총 수 또는  $[1, \text{서브-블록의 수}]$  사이의 임의의 다른 값(경계값을 포함하지 않음)으로 설정될 수 있다.
- [0350] v. 대안적으로, 블록에 대해 서브-블록 예측이 허용될 때,  $M$ 은 1로 설정될 수 있고, 대표적인 서브-블록의 선택은 다음의 것들에 기초한다.
- [0351] 1. 활용된 모션 정보의 주파수,
- [0352] 2. 그 서브-블록이 양방향 예측 블록인지 여부,
- [0353] 3. 참조 픽처 인덱스/참조 픽처에 기초함,
- [0354] 4. 다른 모션 벡터들과 비교한 모션 벡터 차이들(예컨대, 최대 MV 차이들의 선택)
- [0355] 5. 다른 코딩된 정보.
- [0356] e.  $M(M \geq 1)$ 개의 대표적인 위치 세트가 록업 테이블들을 업데이트하기 위해 선택될 때, 록업 테이블들에 추가적인 모션 후보들로서 이들을 추가하기 전에 추가 조건들이 체크될 수 있다.
- [0357] i. 록업 테이블 내의 기존의 모션 후보들에 대한 새로운 모션 정보 세트들에 프루닝이 적용될 수 있다.
- [0358] ii. 일 예에서, 새로운 모션 정보 세트는 록업 테이블 내의 기존의 모션 후보들 중 임의의 것 또는 일부와 동일하지 않아야 한다.
- [0359] iii. 대안적으로, 새로운 모션 정보 세트 및 하나의 기존 모션 후보로부터의 동일한 참조 픽처들에 대해, MV 차이는 하나/다수의 임계치 이상이어야 한다. 예컨대, MV 차이의 수평 및/또는 수직 컴포넌트는 1-픽셀 거리보다 더 커야 한다.
- [0360] iv. 대안적으로, 새로운 모션 정보 세트들은, 기존의 모션 후보들을 재활성화하는 것을 허용하기 위해,  $K > L$ 일 때, 마지막  $K$ 개의 후보 또는 처음  $K \setminus L$ 개의 기존 모션 후보에 대해서만 프루닝된다.
- [0361] v. 대안적으로, 프루닝이 적용되지 않는다.
- [0362] f.  $M$ 개의 모션 정보 세트가 록업 테이블을 업데이트하기 위해 사용되는 경우, 대응하는 카운터는  $M$ 만큼 증가되어야 한다.
- [0363] g. 업데이트될 록업 테이블의 카운터가 현재 블록을 코딩하기 전에  $K$ 로 표시되는 것으로 가정하면, 블록을 코딩한 후에, (위에서 언급된 방법들에 의한) 하나의 선택된 모션 정보 세트에 대해, 이는  $K \setminus L$ 과 동일한 인덱스를 갖는 추가적인 모션 후보로서 추가된다(여기서,  $L$ 은 록업 테이블 크기임). 예들이 도 27a 및 도 27b에 도시된다.
- [0364] i. 대안적으로, 이는  $\min(K+1, L-1)$ 과 동일한 인덱스를 갖는 추가적인 모션 후보로서 추가된다. 대안적으로, 게다가,  $K \geq L$ 인 경우, 제1 모션 후보(0과 동일한 인덱스)가 록업 테이블로부터 제거되고, 후속하는  $K$ 개의 후보 인덱스가 1만큼 감소된다.

- [0365] ii. 위의 방법들 둘 모두( $K$ 와 동일한 엔트리 인덱스로 새로운 모션 후보를 추가하는 것 또는  $\min(K+1, L-1)$ 과 동일한 인덱스로 새로운 모션 후보를 추가하는 것)에 대해, 이들은, 동일한/유사한 모션 후보들이 존재하는지 여부와 상관없이, 이전에 코딩된 블록들로부터의 최근의 몇 개의 모션 정보 세트를 유지하려고 시도하고 있다.
- [0366] iii. 대안적으로, 모션 후보로서 새로운 모션 정보 세트를 LUT에 추가할 때, 중복 체크가 먼저 적용된다. 이 경우, LUT는 이전에 코딩된 블록들로부터의 최근의 몇 개의 모션 정보 세트를 유지할 것이지만, 중복된 것들은 LUT들로부터 제거될 수 있다. 이러한 방법은 중복 제거 기반 LUT 업데이트 방법으로 지칭된다.
- [0367] 1. LUT에 중복 모션 후보들이 있는 경우, LUT와 연관된 카운터가 증가 또는 감소되지 않을 수 있다.
- [0368] 2. 중복 체크는 병합 후보 리스트 구성 프로세스에서의 프루닝 프로세스, 예컨대, 참조 픽처들/참조 픽처 인덱스들이 동일한지 여부, 및 모션 벡터 차이들이 범위 내에 있는지 또는 동일한지 여부를 체크하는 것으로서 정의될 수 있다.
- [0369] 3. LUT에서 중복 모션 후보가 발견되는 경우, 중복 모션 후보는 그의 현재 위치로부터 LUT의 마지막 위치로 이동된다.
- [0370] a. 유사하게, LUT에서 중복 모션 후보가 발견되는 경우, 이 중복 모션 후보는 LUT로부터 제거된다. 추가하여, 중복 모션 후보가 이동한 후에, LUT에 삽입된 모든 모션 후보들은 중복 모션 후보의 제거된 엔트리를 다시 채우기 위해 앞으로 이동한다. 시프팅 후에, 새로운 모션 후보가 LUT에 추가된다.
- [0371] b. 이 경우, 카운터는 변경 없이 유지된다.
- [0372] c. 중복 모션 후보가 LUT에서 식별되면, 중복 체크 프로세스가 종료된다.
- [0373] 4. 다수의 중복 모션 후보가 식별될 수 있다. 이 경우, 이들 모두는 LUT로부터 제거된다. 추가하여, 모든 나머지 모션 후보들은 순서대로 앞으로 이동할 수 있다.
- [0374] a. 이 경우, 카운터는 (중복 모션 후보의 수 마이너스 1)만큼 감소된다.
- [0375] b. 중복 체크 프로세스는  $maxR$ 개의 중복 모션 후보( $maxR$ 은 양의 정수 변수임)를 식별한 후에 종료된다.
- [0376] 5. 중복 체크 프로세스는 제1 모션 후보로부터 마지막 모션 후보로(즉, LUT들에 추가된 순서로, 모션 정보가 유래한 블록들의 디코딩 프로세스의 순서로) 시작될 수 있다.
- [0377] 6. 대안적으로, LUT에 중복 모션 후보들이 있을 때, LUT들로부터 중복 모션 후보들 중 하나 이상을 제거하는 대신에, 중복 모션 후보들로부터 가상 모션 후보들이 도출될 수 있고, 가상 모션 후보들이 중복 모션 후보들을 대체하기 위해 사용될 수 있다.
- [0378] a. 가상 모션 후보들은, 동일한 참조 픽처들을 지적하는 경우 2개의 모션 벡터의 평균; 또는 하나 이상의 모션 벡터의 수평 및/또는 수직 컴포넌트에 오프셋(들)을 추가하는 것에 의해, 중복 모션 후보로부터 도출될 수 있다. 대안적으로, 가상 모션 후보는 룩업 테이블 내의 모션 벡터들을 입력으로 하는 임의의 함수로부터 도출될 수 있다. 예시적인 함수들은: 2개 이상의 모션 벡터를 함께 추가하는 것; 2개 이상의 모션 벡터를 평균하는 것이다. 모션 벡터들은 함수에 입력되기 전에 스케일링될 수 있다.
- [0379] b. 가상 모션 후보들은 중복 모션 후보들과 동일한 위치에 추가될 수 있다.
- [0380] c. 가상 모션 후보들은 모든 다른 모션 후보들 전에(예컨대, 제로와 같은 가장 작은 엔트리 인덱스들로부터 시작하여) 추가될 수 있다.
- [0381] d. 일 예에서, 이는 현재 LUT가 가득 차지 않을 때와 같은 특정 조건들 하에서만 적용된다.
- [0382] 7. 중복 제거 기반 LUT 업데이트 방법은 다음과 같은 특정 조건들 하에서 호출될 수 있다.
- [0383] a. 현재 블록이 병합 모드로 코딩된다.
- [0384] b. 현재 블록이 AMVP 모드로 코딩되지만 MV 차이의 적어도 하나의 컴포넌트가 비-제로이다.
- [0385] c. 현재 블록은 서브-블록 기반 모션 예측/모션 보상 방법들로 코딩되거나 또는 코딩되지 않는다(예컨대, 아핀 모드로 코딩되지 않음).
- [0386] d. 현재 블록은 병합 모드로 코딩되고, 모션 정보는 (예컨대, 공간적 이웃 블록들로부터의, 좌측 이웃 블록으로부터의, 시간적 블록으로부터의) 특정 타입과 연관된다.

- [0387] h. 하나의 블록을 인코딩/디코딩한 후에, 하나 이상의 룩업 테이블은 테이블의 끝에, 즉, 모든 기존 후보들 뒤에 M개의 모션 정보 세트를 삽입하는 것만으로 업데이트될 수 있다.
- [0388] i. 대안적으로, 게다가, 테이블 내의 일부 기존 모션 후보들이 제거될 수 있다.
- [0389] 1. 일 예에서, M개의 모션 정보 세트를 삽입한 후에 테이블이 가득 찬 경우, 모션 후보들의 처음 몇 개의 엔트리가 테이블로부터 제거될 수 있다.
- [0390] 2. 일 예에서, M개의 모션 정보 세트를 삽입하기 전에 테이블이 가득 찬 경우, 모션 후보들의 처음 몇 개의 엔트리가 테이블로부터 제거될 수 있다.
- [0391] ii. 대안적으로, 게다가, 테이블로부터의 모션 후보로 블록이 코딩되는 경우, 테이블 내의 모션 후보들은 선택된 모션 후보가 테이블의 마지막 엔트리에 놓이도록 재순서화될 수 있다.
- [0392] i. 일 예에서, 블록을 인코딩/디코딩하기 전에, 룩업 테이블은  $HMVP_0, HMVP_1, HMVP_2, \dots, HMVP_{k-1}, HMVP_k, HMVP_{k+1}, \dots, HMVP_{L-1}$ 로 표시된 모션 후보들을 포함할 수 있고, 여기서,  $HMVP_i$ 는 룩업 테이블 내의 i번째 엔트리를 표시한다. 블록이  $HMVP_k$ 로부터 예측되는 경우(K는 범위  $[0, L-1]$  내에 포함됨(경계값을 포함함)), 이 블록을 인코딩/디코딩한 후에, 룩업 테이블은  $HMVP_0, HMVP_1, HMVP_2, \dots, HMVP_{k-1}, HMVP_k, HMVP_{k+1}, \dots, HMVP_{L-1}, HMVP_k$ 로 재순서화된다.
- [0393] j. 룩업 테이블은 하나의 인트라 제약된 블록을 코딩한 후에 비워질 수 있다.
- [0394] k. 모션 정보 엔트리가 룩업 테이블 내에 추가되는 경우, 모션 정보로부터의 도출에 의해, 더 많은 모션 정보 엔트리들이 또한 테이블 내에 추가될 수 있다. 이 경우, 룩업 테이블과 연관된 카운터는 1보다 더 많이 증가될 수 있다.
- [0395] i. 일 예에서, 모션 정보 엔트리의 MV가 스케일링되고 테이블 내에 배치된다;
- [0396] ii. 일 예에서, 모션 정보 엔트리의 MV는 (dx, dy)에 의해 추가되고 테이블 내에 배치된다;
- [0397] iii. 일 예에서, 2개 이상의 모션 정보 엔트리의 MV들의 평균이 계산되고 테이블 내에 배치된다.
- [0398] 예 D2: 하나의 블록이 픽처/슬라이스/타일 경계에 위치한 경우, 룩업 테이블들의 업데이트는 항상 허용되지 않을 수 있다.
- [0399] 예 D3: 위의 LCU 행들의 모션 정보가 현재 LCU 행을 코딩하기 위해 디스에이블될 수 있다.
- [0400] a. 이 경우, 새로운 슬라이스/타일/LCU 행의 시작 시에, 이용가능한 모션 후보의 수가 0으로 리셋될 수 있다.
- [0401] 예 D4: 새로운 시간적 계층 인덱스를 갖는 슬라이스/타일을 코딩하기 시작할 시에, 이용가능한 모션 후보의 수가 0으로 리셋될 수 있다.
- [0402] 예 D5: 룩업 테이블은 동일한 시간적 계층 인덱스를 갖는 하나의 슬라이스/타일/LCU 행/슬라이스들로 연속적으로 업데이트될 수 있다.
- [0403] a. 대안적으로, 룩업 테이블은 각각의  $S(S \geq 1)$ 개의 CTU/CTB/CU/CB를 코딩/디코딩한 후에만, 또는 특정 구역(예컨대, 8x8 또는 16x16과 동일한 크기)을 코딩/디코딩한 후에만 업데이트될 수 있다.
- [0404] b. 대안적으로, 룩업 테이블은 특정 모드들로 각각의  $S(S \geq 1)$ 개의 블록(예컨대, CU들/CB들)(예컨대, S개의 인터 코딩된 블록)을 코딩/디코딩한 후에만 업데이트될 수 있다. 대안적으로, 룩업 테이블은 서브-블록 기반 모션 예측/모션 보상 방법으로 코딩되지 않은(예컨대, 아핀 및/또는 ATMVP 모드로 코딩되지 않은) 각각의  $S(S \geq 1)$ 개의 인터 코딩된 블록(예컨대, CU들/CB들)을 코딩/디코딩한 후에만 업데이트될 수 있다.
- [0405] c. 대안적으로, 룩업 테이블은 코딩된/디코딩된 블록의 좌측-상단 좌표가 일부 조건들을 만족시킬 때에만 업데이트될 수 있다. 예컨대, 룩업 테이블은  $(x \& M = 0) \& (y \& M = 0)$ 일 때에만 업데이트되고, 여기서, (x, y)는 코딩된/디코딩된 블록의 좌측-상단 좌표이다. M은 2, 4, 8, 16, 32, 또는 64와 같은 정수이다.
- [0406] d. 대안적으로, 하나의 룩업 테이블이 최대 허용 카운터에 도달하면 그 룩업 테이블은 업데이트가 중지될 수 있다.
- [0407] e. 일 예에서, 카운터는 미리 정의될 수 있다. 대안적으로, 이는 비디오 파라미터 세트(VPS), 시퀀스 파라미터

세트(SPS), 픽처 파라미터 세트(PPS), 슬라이스 헤더, 타일 헤더, 코딩 트리 유닛(CTU), 코딩 트리 블록(CTB), 코딩 유닛(CU), 또는 예측 유닛(PU), 다수의 CTU/CTB/CU/PU를 커버하는 구역에서 시그널링될 수 있다.

[0408] 예 D6: 룩업 테이블 업데이트 프로세스는 상이한 절차들 내에서 호출될 수 있다.

[0409] a. 일 예에서, 병합 모드로 코딩된 블록에 대해, 병합 후보를 디코딩한 후에, 또는 병합 리스트를 구성한 후에, 또는 리파인먼트를 이용하고/이용하거나 리파인먼트를 이용하지 않고 모션 정보를 디코딩한 후에, 룩업 테이블 업데이트 프로세스가 호출될 수 있다.

[0410] b. 일 예에서, AMVP 모드로 코딩된 블록에 대해, 리파인먼트를 이용하고/이용하거나 리파인먼트를 이용하지 않고 모션 정보를 디코딩한 후에, 룩업 테이블 업데이트 프로세스가 호출될 수 있다.

[0411] c. 룩업 테이블을 언제 및/또는 어떻게 업데이트할지는 코딩된 모드, 블록 치수, 비디오 프로세싱 데이터 유닛, 저 지연 체크 등에 의존할 수 있다.

[0412] i. 일 예에서, 하나의 블록이 AMVP 모드로 코딩될 때, 룩업 테이블은 프루닝을 이용하지 않고 직접적으로 업데이트될 수 있다.

[0413] ii. 대안적으로, 하나의 블록이 병합 모드로 코딩될 때, 룩업 테이블은 프루닝을 이용하여 업데이트될 수 있다.

[0414] iii. 대안적으로, 하나의 블록이 병합 모드로 코딩되고, 그의 모션 정보가 공간적 및/또는 시간적 블록들로부터 도출될 때, 룩업 테이블은 프루닝을 이용하여 업데이트될 수 있다.

[0415] iv. 대안적으로, 하나의 블록이 병합 모드로 코딩되고, 그의 모션 정보가 룩업 테이블 내의 모션 후보들로부터 도출될 때, 룩업 테이블은 프루닝을 이용하지 않고 재순서화될 수 있다.

[0416] v. 대안적으로, 하나의 블록이 병합 모드로 코딩되고, 그의 모션 정보가 룩업 테이블 내의 가상 후보들(예컨대, 조합된 양방향, 쌍방향, 제로 모션 벡터 후보들)로부터 도출될 때, 룩업 테이블은 업데이트되지 않을 수 있다.

[0417] vi. 대안적으로, 하나의 블록이 서브-블록 병합 모드 및/또는 삼각 병합 모드로 코딩될 때, 룩업 테이블은 업데이트되지 않을 수 있다.

[0418] vii. 대안적으로, 하나의 블록이 모션 벡터 차이와의 병합(merge with motion vector differences)(MMVD) 모드로 코딩되고, 그의 모션 정보가 공간적 및/또는 시간적 블록들로부터 도출될 때, 룩업 테이블은 직접적으로 업데이트될 수 있다.

[0419] viii. 일 예에서, 하나의 블록이 조명 보상(illumination compensation)(IC) 모드 및/또는 중첩 블록 모션 보상(Overlapped Block Motion Compensation)(OBMC) 모드 및/또는 디코드 측 모션 벡터 도출(Decode-side Motion Vector Derivation)(DMVD) 모드로 코딩될 때, 룩업 테이블은 업데이트되지 않을 수 있다. 대안적으로, 하나의 블록이 그러한 모드로 코딩될 때, 룩업 테이블은 업데이트되지 않을 수 있다.

#### [0420] 추가적인 예시적인 실시예들

[0421] 이력 기반 MVP(history-based MVP)(HMVP) 방법이 제안되고, 여기서, HMVP 후보는 이전에 코딩된 블록의 모션 정보로서 정의된다. 인코딩/디코딩 프로세스 동안 다수의 HMVP 후보들을 갖는 테이블이 유지된다. 테이블은 새로운 슬라이스와 마주칠 때 비워진다. 인터 코딩된 블록이 존재할 때마다, 연관된 모션 정보가 새로운 HMVP 후보로서 테이블의 마지막 엔트리에 추가된다. 전체 코딩 흐름이 도 31에 도시된다.

[0422] 일 예에서, 테이블 크기는 L(예컨대, L = 16 또는 6 또는 44)로 설정되고, 이는 최대 L개의 HMVP 후보가 테이블에 추가될 수 있다는 것을 표시한다.

[0423] (예 11.g.i에 대응하는) 일 실시예에서, 이전에 코딩된 블록들로부터 L개보다 더 많은 HMVP 후보가 존재하는 경우, FIFO(First-In-First-Out) 규칙이 적용되고, 그에 따라, 테이블은 항상 최근의 이전에 코딩된 L개의 모션 후보를 포함한다. 도 32는, HMVP 후보를 제거하고 제안된 방법에서 사용되는 테이블에 새로운 HMVP 후보를 추가하기 위해, FIFO 규칙이 적용되는 예를 도시한다.

[0424] (발명 11.g.iii에 대응하는) 다른 실시예에서, 새로운 모션 후보(이를테면, 현재 블록은 인터 코딩되고 비-아핀 모드임)를 추가할 때마다, LUT들에서 동일한 또는 유사한 모션 후보들이 있는지 여부를 식별하기 위해, 중복 체크 프로세스가 먼저 적용된다.

[0425] 일부 예들은 다음과 같이 도시된다:

- [0426] 도 33a는 새로운 모션 후보를 추가하기 전에 LUT가 가득 찰 때의 예를 도시한다.
- [0427] 도 33b는 새로운 모션 후보를 추가하기 전에 LUT가 가득 차지 않을 때의 예를 도시한다.
- [0428] 도 33a 및 도 33b는 (하나의 중복 모션 후보가 제거되는) 중복 제거 기반 LUT 업데이트 방법의 예를 함께 도시한다.
- [0429] 도 34a 및 도 34b는 (다수의 중복 모션 후보가 제거되고 도면들에서는 2개의 후보가 제거되는) 중복 제거 기반 LUT 업데이트 방법의 2개의 경우에 대한 예시적인 구현을 도시한다.
- [0430] 도 34a는 새로운 모션 후보를 추가하기 전에 LUT가 가득 찰 때의 예시적인 경우를 도시한다.
- [0431] 도 34b는 새로운 모션 후보를 추가하기 전에 LUT가 가득 차지 않을 때의 예시적인 경우를 도시한다.
- [0432] 병합 후보 리스트 구성 프로세스에서 HMVP 후보들이 사용될 수 있다. 테이블 내의 마지막 엔트리로부터 첫 번째 엔트리까지의 모든 HMVP 후보들(또는 마지막 K0개의 HMVP, 예컨대, K0은 16 또는 6과 동일함)은 TMVP 후보 후에 삽입된다. 프루닝이 HMVP 후보들에 적용된다. 이용가능한 병합 후보의 총 수가 시그널링된 최대 허용 병합 후보들에 도달하면, 병합 후보 리스트 구성 프로세스가 종료된다. 대안적으로, 추가된 모션 후보의 총 수가 주어진 값에 도달하면, LUT들로부터의 모션 후보들의 인출이 종료된다.
- [0433] 유사하게, HMVP 후보들은 또한, AMVP 후보 리스트 구성 프로세스에서 사용될 수 있다. 테이블 내의 마지막 K1개의 HMVP 후보의 모션 벡터들은 TMVP 후보 후에 삽입된다. AMVP 타겟 참조 픽처와 동일한 참조 픽처를 갖는 HMVP 후보들만이 AMVP 후보 리스트를 구성하는 데 사용된다. 프루닝이 HMVP 후보들에 적용된다. 일 예에서, K1은 4로 설정된다.
- [0434] **인용 문헌들**
- [0435] [1] "Overview of the High Efficiency Video Coding (HEVC) Standard", Gary J. Sullivan, Jens-Rainer Ohm, Woo-Jin Han, and Thomas Wiegand, IEEE Transactions on Circuits and Systems for Video Technology, Vol. 22, No. 12, December 2012.
- [0436] [2] "Overview of the H.264/AVC video coding standard", Ajay Luthra, Pankaj Topiwala, Proceedings of SPIE Vol. 5203 Applications of Digital Image Processing XXVI.
- [0437] [3] J. Chen, E. Alshina, G. J. Sullivan, J.-R. Ohm, J. Boyce, "Algorithm description of Joint Exploration Test Model 7 (JEM7)," JVET-G1001, Aug. 2017.
- [0438] [4] JEM-7.0: [https://jvet.hhi.fraunhofer.de/svn/svn\\_HMJEMSoftware/tags/HM-16.6-JEM-7.0](https://jvet.hhi.fraunhofer.de/svn/svn_HMJEMSoftware/tags/HM-16.6-JEM-7.0).
- [0439] [5] H.265/HEVC, <https://www.itu.int/rec/T-REC-H.265>
- [0440] [6] "Multi-Type-Tree", JVET-D0117
- [0441] [7] 국제 특허 출원 WO2016/091161
- [0442] [8] "Description of SDR, HDR and 360° video coding technology proposal by Qualcomm and Technicolor - low and high complexity versions", JVET-J0021.
- [0443] 도 28a는 비디오 프로세싱 장치(2800)의 블록도이다. 장치(2800)는 본원에서 설명되는 방법들 중 하나 이상을 구현하는 데 사용될 수 있다. 장치(2800)는 스마트폰, 태블릿, 컴퓨터, 사물 인터넷(IoT) 수신기 등에서 구현될 수 있다. 장치(2800)는 하나 이상의 프로세서(2802), 하나 이상의 메모리(2804), 및 비디오 프로세싱 하드웨어(2806)를 포함할 수 있다. 프로세서(들)(2802)는 본 문서에서 설명되는 하나 이상의 방법을 구현하도록 구성될 수 있다. 메모리(메모리들)(2804)는 본원에서 설명되는 방법들 및 기법들을 구현하기 위해 사용되는 데이터 및 코드를 저장하기 위해 사용될 수 있다. 비디오 프로세싱 하드웨어(2806)는, 하드웨어 회로부에서, 본 문서에서 설명되는 일부 기법들을 구현하는 데 사용될 수 있다.
- [0444] 도 28b는 개시되는 기법들이 구현될 수 있는 비디오 프로세싱 시스템의 블록도의 다른 예이다. 도 28b는 본원에서 개시되는 다양한 기법들이 구현될 수 있는 예시적인 비디오 프로세싱 시스템(3100)을 도시하는 블록도이다. 다양한 구현들은 시스템(3100)의 컴포넌트들 중 일부 또는 전부를 포함할 수 있다. 시스템(3100)은 비디오 콘텐츠를 수신하기 위한 입력(3102)을 포함할 수 있다. 비디오 콘텐츠는 원시(raw) 또는 압축되지 않은 포맷, 예컨대, 8 또는 10 비트 다중-컴포넌트 픽셀 값들로 수신될 수 있거나, 또는 압축된 또는 인코딩된



포맷될 수 있다. 입력(3102)은 네트워크 인터페이스, 주변 버스 인터페이스, 또는 저장 인터페이스를 표현할 수 있다. 네트워크 인터페이스의 예들은 유선 인터페이스들, 이더넷, 이더넷, 수동 광 네트워크(passive optical network)(PON) 등, 및 무선 인터페이스들, 이더넷, Wi-Fi 또는 셀룰러 인터페이스들을 포함한다.

[0445] 시스템(3100)은 본 문서에서 설명되는 다양한 코딩 또는 인코딩 방법들을 구현할 수 있는 코딩 컴포넌트(3104)를 포함할 수 있다. 코딩 컴포넌트(3104)는, 비디오의 코딩된 표현을 생성하기 위해, 입력(3102)으로부터 코딩 컴포넌트(3104)의 출력으로 비디오의 평균 비트레이트를 감소시킬 수 있다. 따라서, 코딩 기법들은 때때로, 비디오 압축 또는 비디오 트랜스코딩 기법들로 지칭된다. 코딩 컴포넌트(3104)의 출력은, 컴포넌트(3106)로 표현된 바와 같이, 저장될 수 있거나 또는 연결된 통신을 통해 송신될 수 있다. 입력(3102)에서 수신된 비디오의 저장 또는 통신된 비트스트림(또는 코딩된) 표현은 디스플레이 인터페이스(3110)로 전송되는 디스플레이가능 비디오 또는 픽셀 값들을 생성하기 위해 컴포넌트(3108)에 의해 사용될 수 있다. 비트스트림 표현으로부터 사용자-시청가능 비디오를 생성하는 프로세스는 때때로 비디오 압축 해제로 지칭된다. 게다가, 특정 비디오 프로세싱 동작들이 "코딩" 동작들 또는 툴들로 지칭되지만, 코딩 툴들 또는 동작들은 인코더에서 사용되고, 코딩의 결과들을 역전시키는 대응하는 디코딩 툴들 또는 동작들은 디코더에 의해 수행될 것이라는 것이 인식될 것이다.

[0446] 주변 버스 인터페이스 또는 디스플레이 인터페이스의 예들은 USB(universal serial bus) 또는 HDMI(high definition multimedia interface) 또는 디스플레이포트(Displayport) 등을 포함할 수 있다. 저장 인터페이스들의 예들은 SATA(serial advanced technology attachment), PCI, IDE 인터페이스 등을 포함한다. 본 문서에서 설명되는 기법들은 다양한 전자 디바이스들, 이더넷, 모바일 폰들, 랩톱들, 스마트폰들, 또는 디지털 데이터 프로세싱 및/또는 비디오 디스플레이를 수행할 수 있는 다른 디바이스들에서 구현될 수 있다.

[0447] 일부 실시예들에서, 비디오 코딩 방법들은 도 28a 또는 도 28b에 대해 설명된 바와 같은 하드웨어 플랫폼 상에 구현되는 장치를 사용하여 구현될 수 있다.

[0448] 도 29는 비디오 프로세싱 방법(2900)의 예에 대한 흐름도이다. 방법(2900)은, 단계(2902)에서, 하나 이상의 모션 후보 테이블을 유지하는 단계를 포함한다. 방법(2900)은, 단계(2904)에서, 모션 후보 테이블을 사용하여, 순차적으로 다수의 비디오 프로세싱 동작을 사용하여, 비디오의 현재 비디오 블록과 현재 비디오 블록의 코딩된 표현 사이의 변환을 수행하는 단계를 더 포함한다. 방법(2900)은, 단계(2906)에서, 특정 프로세싱 동작의 완료로 인해 모션 후보 테이블을 업데이트하는 단계를 더 포함한다.

[0449] 도 30은 비디오 프로세싱 방법(3000)의 예에 대한 흐름도이다. 방법(3000)은, 단계(3002)에서, 하나 이상의 모션 후보 테이블을 유지하는 단계를 포함한다. 방법(3000)은, 단계(3004)에서, 코딩 모드, 치수, 비디오 프로세싱 데이터 유닛, 또는 저 지연 체크를 포함하는 현재 비디오 블록의 특성들에 기초하여, i) 하나 이상의 모션 후보 테이블 중 하나의 모션 후보 테이블을 업데이트할지 여부, 또는 ii) 모션 후보 테이블을 어떻게 업데이트할지 중 적어도 하나를 결정하는 단계들 더 포함한다. 방법(3000)은, 단계(3006)에서, 결정에 기초하여, 비디오의 현재 비디오 블록과 비디오의 코딩된 표현 사이의 변환을 수행하는 단계를 더 포함한다.

[0450] 방법들(2900 및 3000)에 대해, 일부 실시예들에서, 모션 정보는: 예측 방향, 참조 픽처 인덱스, 모션 벡터 값들, 강도 보상 플래그, 아핀 플래그, 모션 벡터 차이 정밀도, 및 모션 벡터 차이 값 중 적어도 하나를 포함한다. 추가로, 모션 정보는 모션 정보의 소스를 표시하는 블록 위치 정보를 더 포함할 수 있다. 일부 실시예들에서, 비디오 블록은 CU 또는 PU일 수 있고, 비디오의 일부는 하나 이상의 비디오 슬라이스 또는 하나 이상의 비디오 픽처에 대응할 수 있다.

[0451] 일부 실시예들에서, 각각의 LUT는 연관된 카운터를 포함하고, 여기서, 카운터는 비디오의 부분의 시작 시에 제로 값으로 초기화되고, 비디오의 부분 내의 각각의 인코딩된 비디오 구역에 대해 증가된다. 비디오 구역은 코딩 트리 유닛, 코딩 트리 블록, 코딩 유닛, 코딩 블록, 또는 예측 유닛 중 하나를 포함한다. 일부 실시예들에서, 카운터는, 대응하는 LUT에 대해, 대응하는 LUT로부터 제거된 모션 후보의 수를 표시한다. 일부 실시예들에서, 모션 후보들의 세트는 모든 LUT들에 대해 동일한 크기를 가질 수 있다. 일부 실시예들에서, 비디오의 부분은 비디오의 슬라이스에 대응하고, 여기서, LUT의 수는  $N \times P$ 와 동일하고, 여기서,  $N$ 은 디코딩 스테드당 LUT들을 표현하는 정수이고,  $P$ 는 최대 코딩 유닛 행의 수 또는 비디오의 슬라이스 내의 타일의 수를 표현하는 정수이다. 방법들(2900 및 3000)의 추가적인 세부사항들은 섹션 4에서 제공된 예들 및 아래에 열거된 예들에서 설명된다.

[0452] 1. 비디오 프로세싱 방법은, 하나 이상의 모션 후보 테이블을 유지하는 단계; 모션 후보 테이블을 사용하여, 순차적으로 다수의 비디오 프로세싱 동작을 사용하여, 비디오의 현재 비디오 블록과 현재 비디오 블록의 코딩된 표현 사이의 변환을 수행하는 단계; 및 특정 프로세싱 동작의 완료로 인해 모션 후보 테이블을 업데이트하는 단

계를 포함한다.

- [0453] 2. 예 1의 방법에서, 특정 프로세싱 동작의 아이덴티티는 현재 비디오 블록의 코딩 특성에 의존한다.
- [0454] 3. 예 1의 방법에서, 현재 비디오 블록은 병합 모드로 코딩되고, 모션 후보 테이블은 변환의 일부인 이하의 동작들:
  - [0455] (i) 병합 후보를 디코딩하는 동작;
  - [0456] (ii) 상기 병합 후보를 포함하는 병합 리스트를 구성하는 동작;
  - [0457] (iii) 리파인먼트를 이용하지 않고 모션 정보를 디코딩하는 동작; 또는
  - [0458] (iv) 리파인먼트를 이용하여 모션 정보를 디코딩하는 동작
- [0459] 중 적어도 하나 후에 업데이트된다.
- [0460] 4. 예 3의 방법에서, 현재 비디오 블록은 병합 모드로 코딩되고, 모션 후보 테이블은 디코드 측 모션 리파인먼트(DMVR)로 리파인먼트되기 전의 디코딩된 모션 정보를 사용하여 업데이트된다.
- [0461] 5. 예 3의 방법에서, 현재 비디오 블록은 병합 모드로 코딩되고, 모션 후보 테이블은 모션 벡터 차이와의 병합(merge with motion vector differences)(MMVD)으로 리파인먼트된 후의 디코딩된 모션 정보를 사용하여 업데이트된다.
- [0462] 6. 예 1의 방법에서, 현재 비디오 블록은 고급 모션 벡터 예측(advanced motion vector prediction)(AMVP) 모드로 코딩되고, 모션 후보 테이블은 리파인먼트를 이용하거나 또는 이용하지 않고 모션 정보를 디코딩한 후에 업데이트된다.
- [0463] 7. 비디오 프로세싱을 위한 방법은, 하나 이상의 모션 후보 테이블을 유지하는 단계; 코딩 모드, 치수, 비디오 프로세싱 데이터 유닛, 또는 저 지연 체크를 포함하는 현재 비디오 블록의 특성들에 기초하여, i) 하나 이상의 모션 후보 테이블 중 하나의 모션 후보 테이블을 업데이트할지 여부, 또는 ii) 모션 후보 테이블을 어떻게 업데이트할지 중 적어도 하나를 결정하는 단계; 및 결정에 기초하여, 비디오의 현재 비디오 블록과 비디오의 코딩된 표현 사이의 변환을 수행하는 단계를 포함한다.
- [0464] 8. 예 7의 방법에서, 현재 비디오 블록은 고급 모션 벡터 예측(AMVP) 모드로 코딩되고, 모션 후보 테이블은 프루닝을 이용하지 않고 업데이트된다.
- [0465] 9. 예 7의 방법에서, 현재 비디오 블록은 병합 모드로 코딩되고, 모션 후보 테이블은 프루닝을 이용하여 업데이트된다.
- [0466] 10. 예 8 또는 예 9의 방법에서, 프루닝은 현재 비디오 블록과 연관된 모션 정보를 모션 후보 테이블 내의 적어도 하나의 모션 후보와 비교하는 것을 포함한다.
- [0467] 11. 예 9의 방법에서, 현재 비디오 블록의 모션 정보는 공간적 블록들 또는 시간적 블록들로부터 도출된다.
- [0468] 12. 예 7의 방법에서, 현재 비디오 블록은 병합 모드로 코딩되고, 현재 비디오 블록의 모션 정보는 모션 후보 테이블 내의 모션 후보들로부터 도출되고, 모션 후보 테이블은 프루닝을 이용하지 않고 재순서화된다.
- [0469] 13. 예 7의 방법에서, 현재 비디오 블록은 병합 모드로 코딩되고, 현재 비디오 블록의 모션 정보는 모션 후보 테이블 내의 가상 후보들로부터 도출되고, 모션 후보 테이블은 업데이트되지 않는다.
- [0470] 14. 예 7의 방법에서, 현재 비디오 블록은 서브-블록 병합 모드 및/또는 삼각 병합 모드로 코딩되고, 모션 후보 테이블은 업데이트되지 않는다.
- [0471] 15. 예 7의 방법에서, 현재 비디오 블록은 모션 벡터 차이와의 병합(MMVD) 모드로 코딩되고, 현재 비디오 블록의 모션 정보는 공간적 블록들 또는 시간적 블록들로부터 도출되고, 모션 후보 테이블은 업데이트된다.
- [0472] 16. 예 7의 방법에서, 현재 비디오 블록은 조명 보상(IC) 모드, 중첩 블록 모션 보상(OBMC) 모드, 또는 디코드 측 모션 벡터 도출(DMVD) 모드 중 하나 이상으로 코딩되고, 모션 후보 테이블은 업데이트되지 않는다.
- [0473] 17. 예 7의 방법에서, 현재 비디오 블록은 조명 보상(IC) 모드, 중첩 블록 모션 보상(OBMC) 모드, 또는 디코드 측 모션 벡터 도출(DMVD) 모드 중 하나 이상으로 코딩되고, 모션 후보 테이블은 업데이트된다.

- [0474] 18. 예1 내지 예 17 중 어느 한 예의 방법에서, 모션 후보 테이블은 이력 모션 벡터 예측(HMVP) 테이블이다.
- [0475] 19. 예 1 내지 예 18 중 어느 한 예의 방법에서, 변환을 수행하는 단계는 현재 비디오 블록으로부터 코딩된 표현을 생성하는 단계를 포함한다.
- [0476] 20. 예 1 내지 예 18 중 어느 한 예의 방법에서, 변환을 수행하는 단계는 코딩된 표현으로부터 현재 비디오 블록을 생성하는 단계를 포함한다.
- [0477] 21. 비디오 시스템 내의 장치는, 프로세서, 및 명령어들을 갖는 비-일시적 메모리를 포함하고, 명령어들은, 프로세서에 의한 실행 시에, 프로세서로 하여금 예 1 내지 예 20 중 하나 이상의 예에 기재된 방법을 구현하게 한다.
- [0478] 22. 비-일시적 컴퓨터 판독가능 매체들 상에 저장되는 컴퓨터 프로그램 제품에서, 컴퓨터 프로그램 제품은 예 1 내지 예 20 중 하나 이상의 예에 기재된 방법을 수행하기 위한 프로그램 코드를 포함한다.
- [0479] 전술된 바로부터, 현재 개시된 기술의 특정 실시예들이 예시의 목적으로 본원에서 설명되었지만, 본 발명의 범위를 벗어나지 않으면서 다양한 변형들이 이루어질 수 있음이 이해될 것이다. 따라서, 현재 개시된 기술은 첨부된 청구 범위에 의한 것을 제외하고는 제한되지 않는다.
- [0480] 본 문서에서 설명되는 개시된 및 다른 실시예들, 모듈들, 및 기능 동작들은 디지털 전자 회로로 구현될 수 있거나, 또는 본 문서에서 개시된 구조들 및 이들의 구조적 등가물들을 포함하는, 컴퓨터 소프트웨어, 펌웨어, 또는 하드웨어로 구현될 수 있거나, 또는 이들 중 하나 이상의 조합들로 구현될 수 있다. 개시된 실시예 및 다른 실시예들은, 하나 이상의 컴퓨터 프로그램 제품들, 즉, 데이터 프로세싱 장치에 의한 실행을 위해 또는 데이터 프로세싱 장치의 동작을 제어하기 위해 컴퓨터 판독가능 매체 상에 인코딩된 컴퓨터 프로그램 명령어들의 하나 이상의 모듈들로서 구현될 수 있다. 컴퓨터 판독가능 매체는 머신 판독가능 저장 디바이스, 머신 판독가능 저장 기관, 메모리 디바이스, 머신 판독가능 전파 신호에 영향을 미치는 물질의 조성, 또는 이들 중 하나 이상의 조합일 수 있다. "데이터 프로세싱 장치"라는 용어는, 예컨대, 프로그램가능 프로세서, 컴퓨터, 또는 다수의 프로세서 또는 컴퓨터를 포함하는, 데이터를 프로세싱하기 위한 모든 장치, 디바이스들, 및 머신들을 포함한다. 장치는, 하드웨어 이외에, 해당 컴퓨터 프로그램에 대한 실행 환경을 생성하는 코드, 예컨대, 프로세서 펌웨어, 프로토콜 스택, 데이터베이스 관리 시스템, 운영 체제, 또는 이들 중 하나 이상의 조합을 구성하는 코드를 포함할 수 있다. 전파 신호는 적절한 수신기 장치로 송신하기 위한 정보를 인코딩하기 위해 생성되는 인공적으로 생성된 신호, 예컨대, 머신-생성 전기, 광학 또는 전자기 신호이다.
- [0481] 컴퓨터 프로그램(프로그램, 소프트웨어, 소프트웨어 애플리케이션, 스크립트, 또는 코드라고 또한 알려져 있음)은 컴파일형 또는 해석형 언어들을 포함하는 임의의 형태의 프로그래밍 언어로 기록될 수 있으며, 컴퓨터 프로그램은, 독립형 프로그램 또는 모듈로서, 컴포넌트, 서브루틴, 또는 컴퓨팅 환경에서 사용하는 데 적합한 다른 유닛을 포함하는 임의의 형태로 배포될 수 있다. 컴퓨터 프로그램은 반드시 파일 시스템 내의 파일에 대응할 필요는 없다. 프로그램은, 다른 프로그램들 또는 데이터(예컨대, 마크업 언어 문서에 저장된 하나 이상의 스크립트들)를 보유하는 파일의 일부, 해당 프로그램에 전용되는 단일 파일, 또는 다수의 협력 파일(예컨대, 하나 이상의 모듈, 하위 프로그램들, 또는 코드의 부분들을 저장하는 파일들)로 저장될 수 있다. 컴퓨터 프로그램은 하나의 컴퓨터 상에서, 또는 하나의 위치에 위치되거나 또는 다수의 위치에 걸쳐 분산되어 통신 네트워크에 의해 상호연결되는 다수의 컴퓨터 상에서 실행되도록 배포될 수 있다.
- [0482] 본 문서에서 설명되는 프로세스들 및 로직 흐름들은, 입력 데이터에 대해 동작하여 출력을 생성함으로써 기능들을 수행하도록 하나 이상의 컴퓨터 프로그램을 실행하는 하나 이상의 프로그램 가능 프로세서들에 의해 수행될 수 있다. 프로세스들 및 로직 흐름들은 또한, 특수 목적 로직 회로부, 예컨대 FPGA(field programmable gate array) 또는 ASIC(application specific integrated circuit)에 의해 수행될 수 있으며, 장치는 또한, 특수 목적 로직 회로부, 예컨대 FPGA 또는 ASIC로서 구현될 수 있다.
- [0483] 컴퓨터 프로그램의 실행에 적합한 프로세서들은, 예로서, 범용 및 특수 목적 마이크로프로세서들 둘 모두, 및 임의의 종류의 디지털 컴퓨터의 임의의 하나 이상의 프로세서를 포함한다. 일반적으로, 프로세서는 판독 전용 메모리 또는 랜덤 액세스 메모리 또는 이들 둘 모두로부터 명령어 및 데이터를 수신할 것이다. 컴퓨터의 필수 엘리먼트들은 명령어들을 수행하기 위한 프로세서, 및 명령어 및 데이터를 저장하기 위한 하나 이상의 메모리 디바이스이다. 일반적으로, 컴퓨터는 또한, 데이터를 저장하기 위한 하나 이상의 대용량 저장 디바이스, 예컨대 자기, 광자기 디스크들 또는 광 디스크들을 포함할 것이거나, 또는 이들로부터 데이터를 수신하도록 또는 데이터를 전송하도록, 또는 이들 둘 모두를 수행하도록 동작적으로 커플링될 것이다. 그러나, 컴퓨터는 그



러한 디바이스들을 가질 필요가 없다. 컴퓨터 프로그램 명령어들 및 데이터를 저장하기에 적합한 컴퓨터 판독 가능 매체는, 예로서, 반도체 메모리 디바이스, 예를 들어 EPROM, EEPROM 및 플래시 메모리 디바이스들; 자기 디스크들, 예를 들어 내부 하드 디스크들 또는 제거가능 디스크들; 광자기 디스크들; 및 CD ROM 및 DVD-ROM 디스크들을 포함하는 모든 형태의 비-휘발성 메모리, 매체 및 메모리 디바이스들을 포함한다. 프로세서 및 메모리는 특수 목적 로직 회로부에 의해 보완되거나 또는 이에 통합될 수 있다.

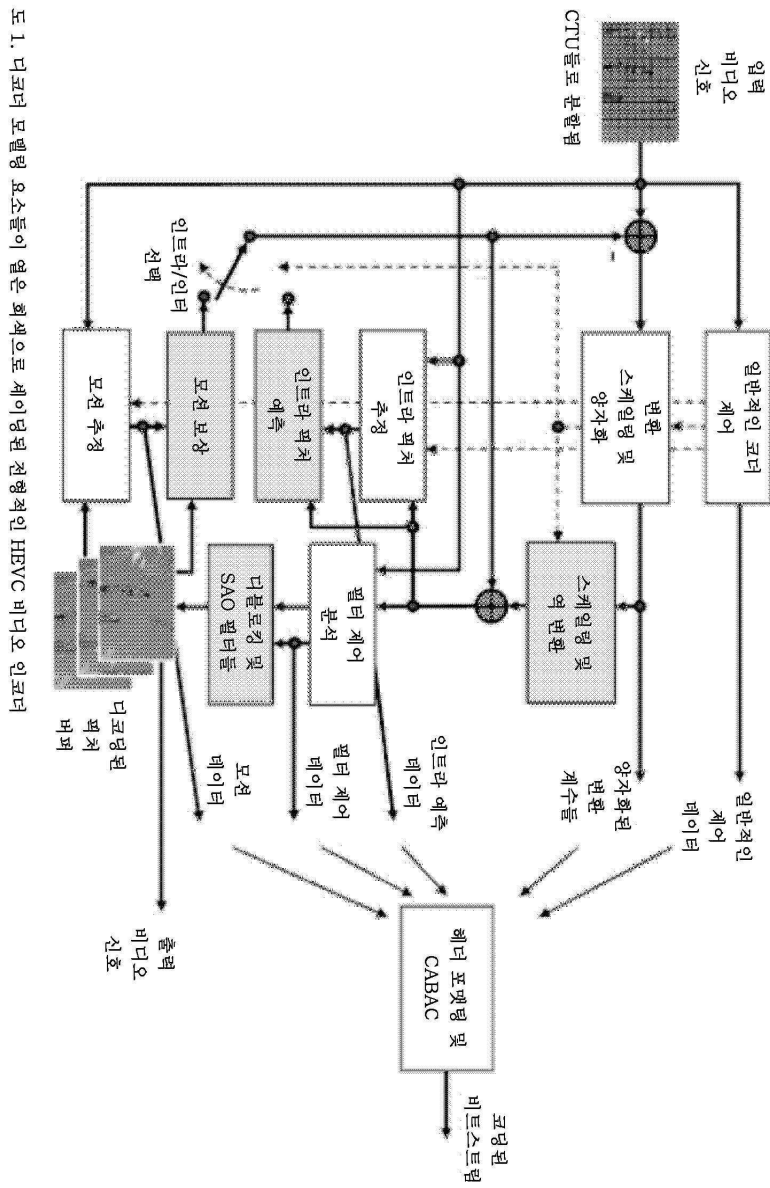
[0484] 이 특허 문서는 다수의 세부사항을 포함하지만, 이들은 임의의 발명의 범위 또는 청구될 수 있는 것에 대한 제한들로 해석되지 않아야 하며, 오히려, 특정 발명들의 특정 실시예들에 특정될 수 있는 피처들의 설명들로 해석되어야 한다. 별개의 실시예들의 맥락에서 본 특허 문서에 설명된 특정 피처들은 또한, 단일 실시예로 조합하여 구현될 수 있다. 대조적으로, 단일 실시예의 맥락에서 설명된 다양한 피처들은 또한, 개별적으로 다수의 실시예로 또는 임의의 적절한 서브조합으로 구현될 수 있다. 게다가, 피처들이 특정 조합들로 동작하는 것으로서 위에서 설명될 수 있고 심지어 그와 같이 처음에 청구될 수 있지만, 청구되는 조합으로부터의 하나 이상의 피처는 일부 경우들에서 조합으로부터 제거될 수 있고, 그리고 청구되는 조합은 서브조합 또는 서브조합의 변형에 관련될 수 있다.

[0485] 유사하게, 동작들이 특정한 순서로 도면들에 도시되어 있지만, 이는, 바람직한 결과들을 달성하기 위해, 그러한 동작들이 도시된 특정한 순서 또는 순차적인 순서로 수행될 필요가 있거나, 모든 예시된 동작들이 수행될 필요가 있는 것으로 이해되지 않아야 한다. 더욱이, 본 특허 문서에서 설명된 실시예들에서의 다양한 시스템 컴포넌트들의 분리가 모든 실시예들에서 이러한 분리를 요구하는 것으로 이해되지 않아야 한다.

[0486] 몇몇 구현들 및 예들만이 설명되고, 다른 구현들, 개선들, 및 변형들은 본 특허 문서에 설명 및 예시된 것에 기초하여 이루어질 수 있다.

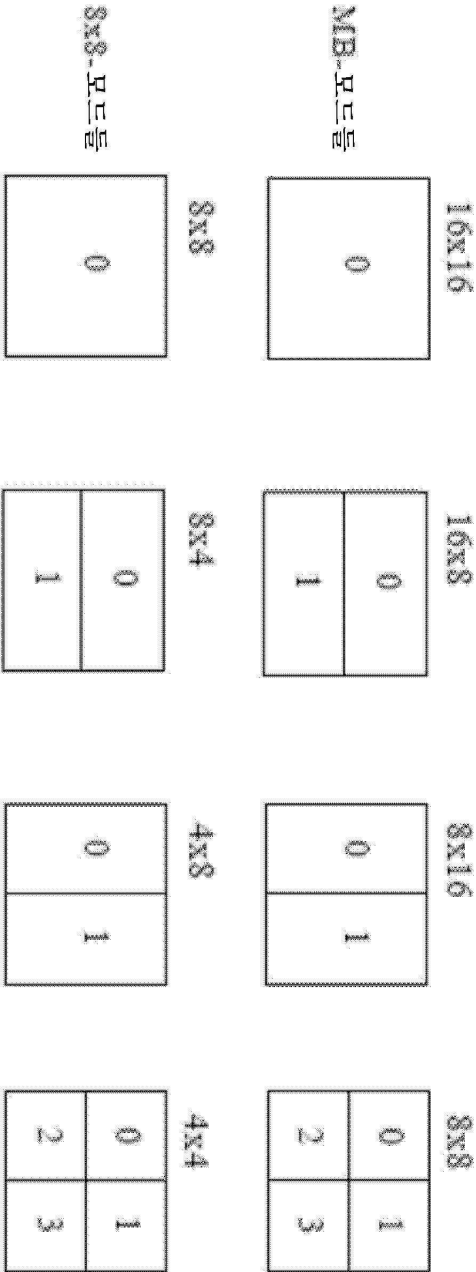
도면

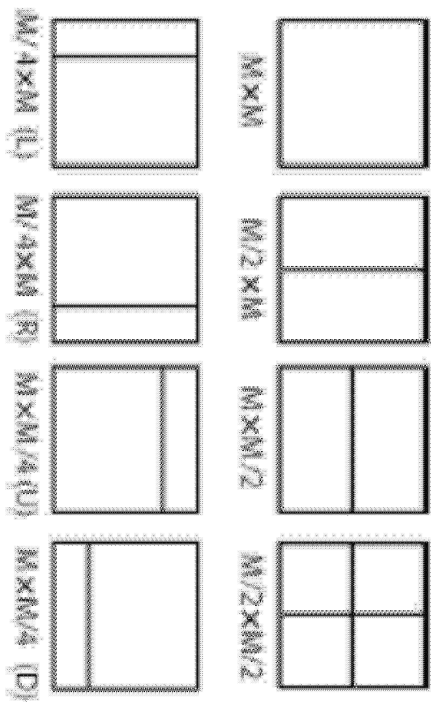
도면1



도 1. 디코더 모델링 요소들이 열린 회색으로 표시된 전형적인 HEVC 비디오 인코더

도면2

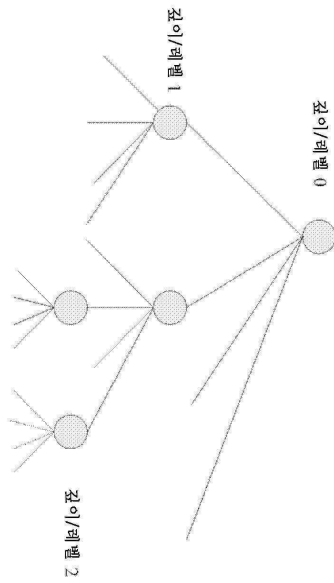
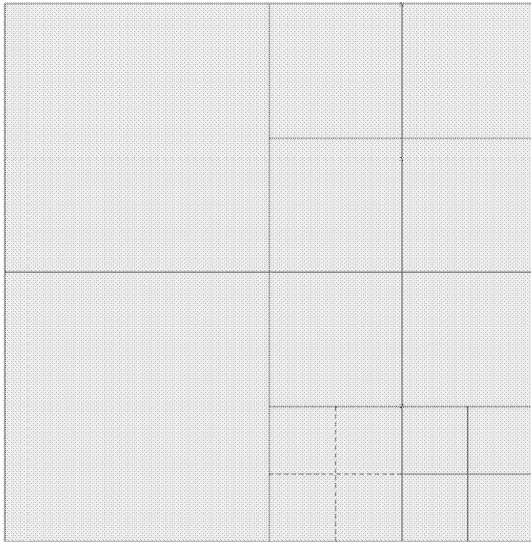




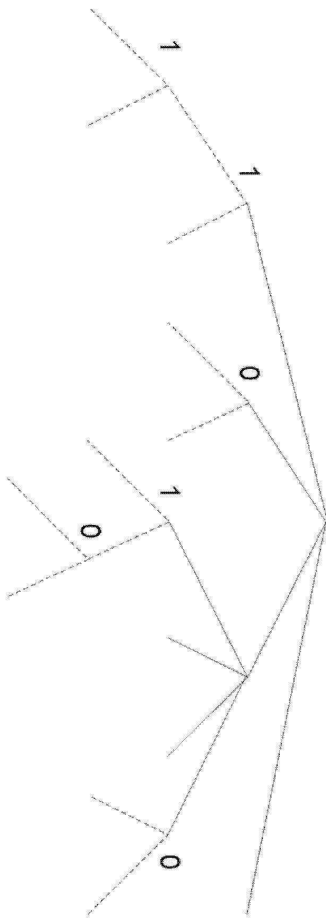
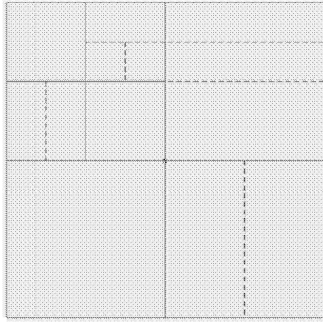
도면3

도 3. 특정 크기 제약들을 받는, CB를 PB들로 분할하기 위한 모드들. 인트라 픽처 예측된 CB들에 대해,  $M \times M$  및  $M/2 \times M/2$ 만이 지원됨.

도면4

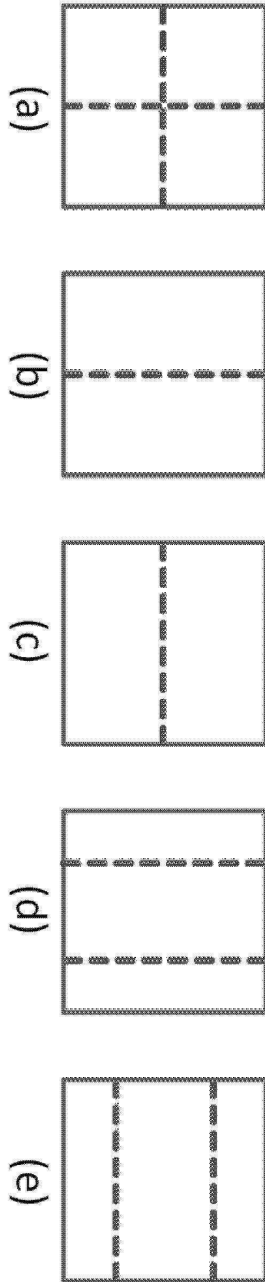


도면5

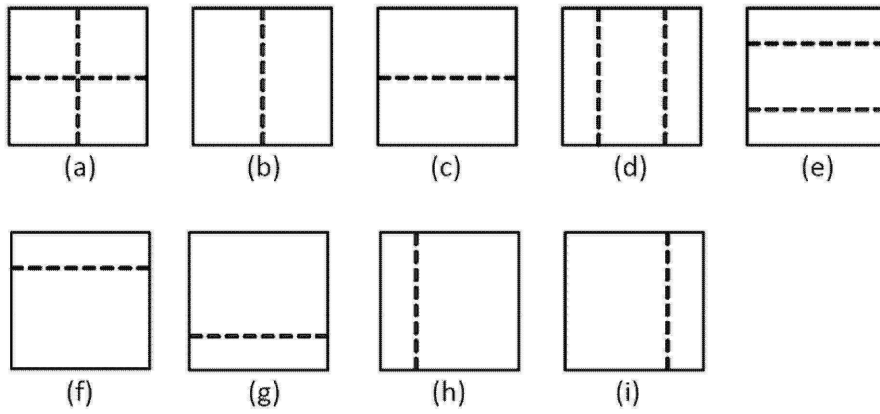




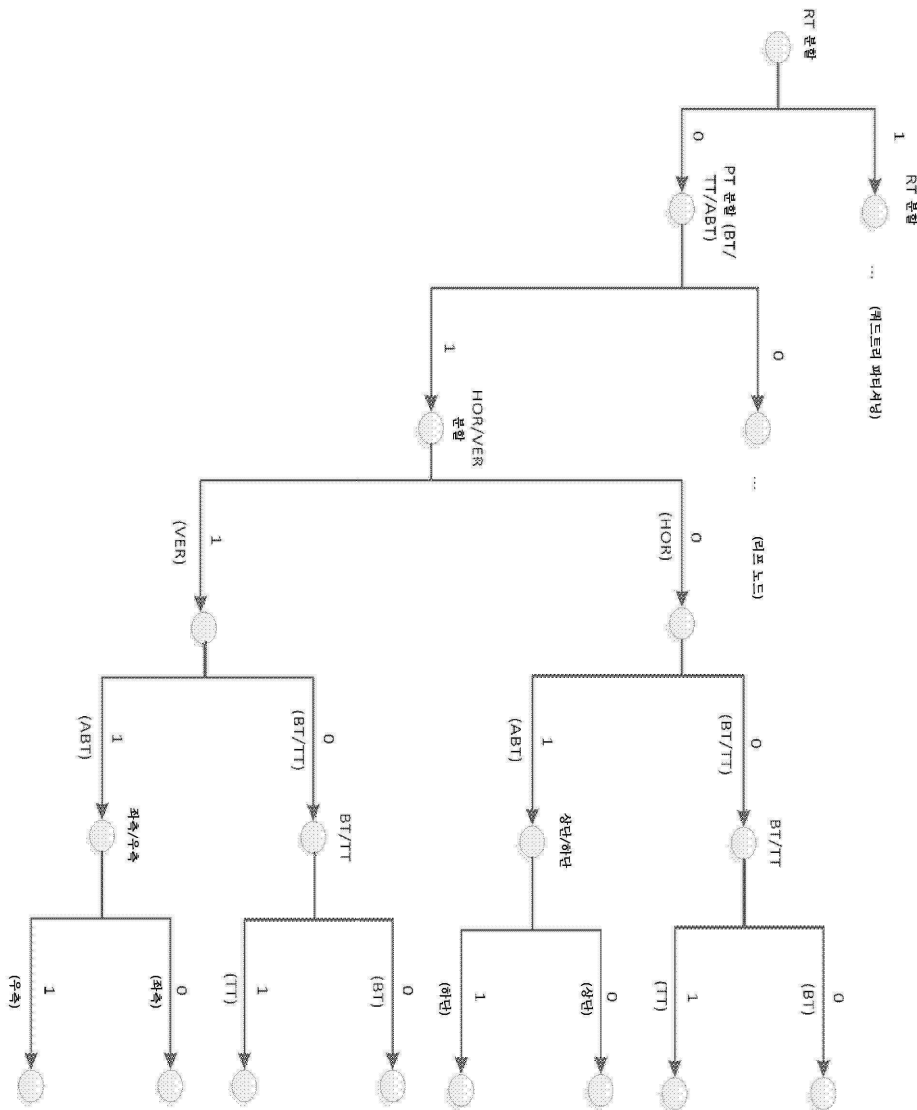
도면6



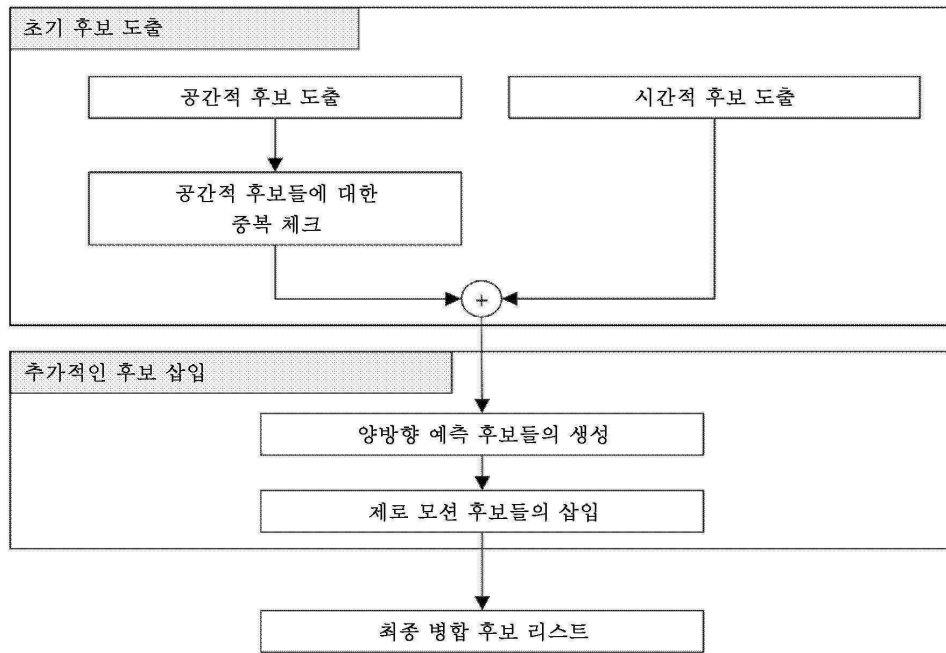
도면7



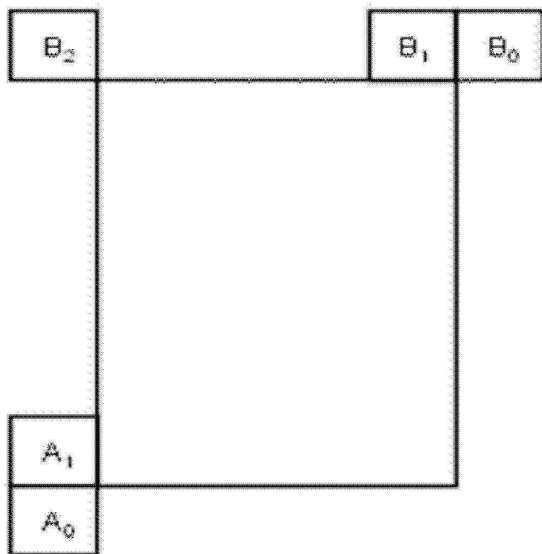
도면8



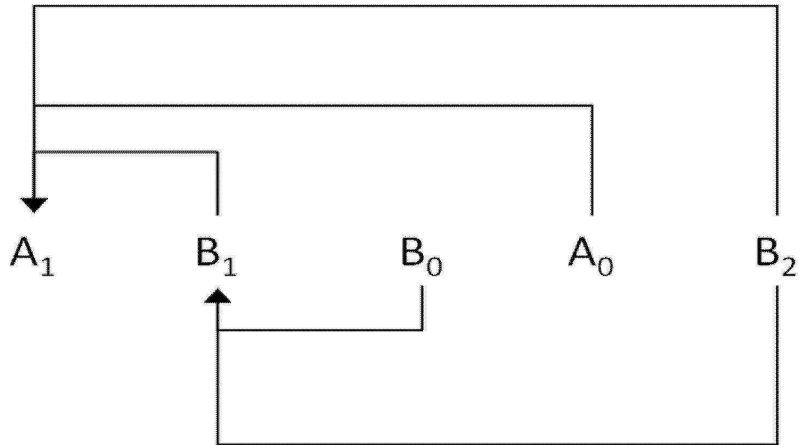
도면9



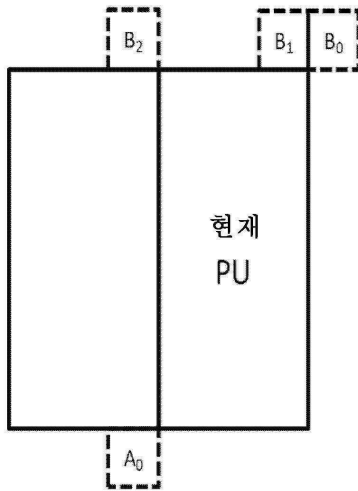
도면10



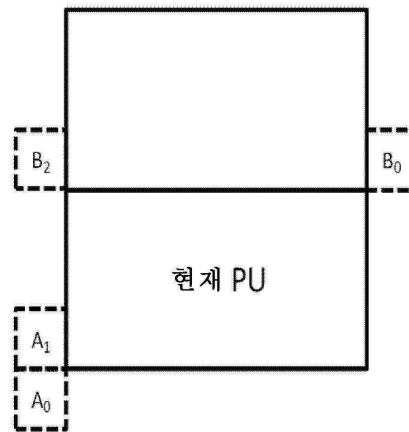
도면11



도면12

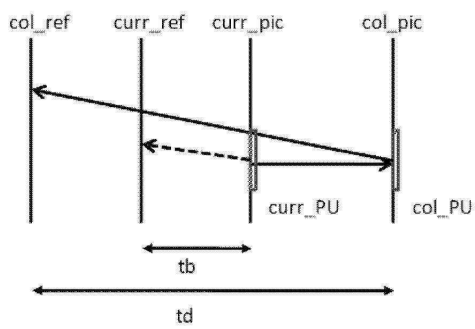


(a) Nx2N의 제2 PU

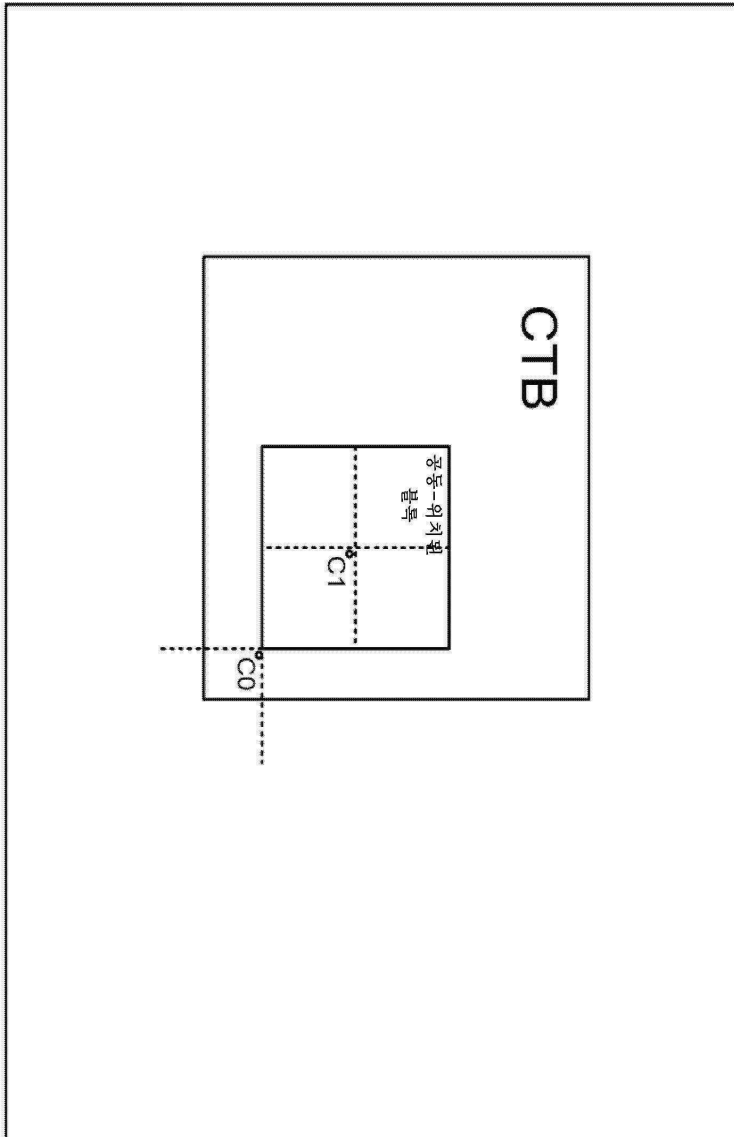


(b) 2NxN의 제2 PU

도면13

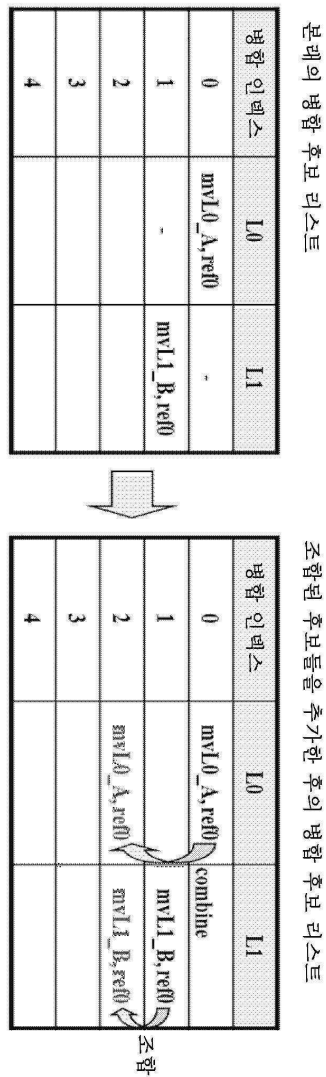


도면14

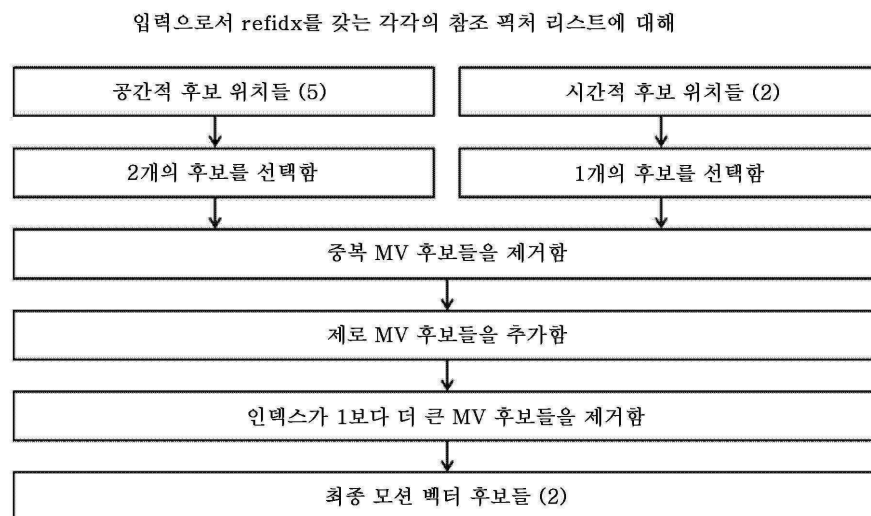




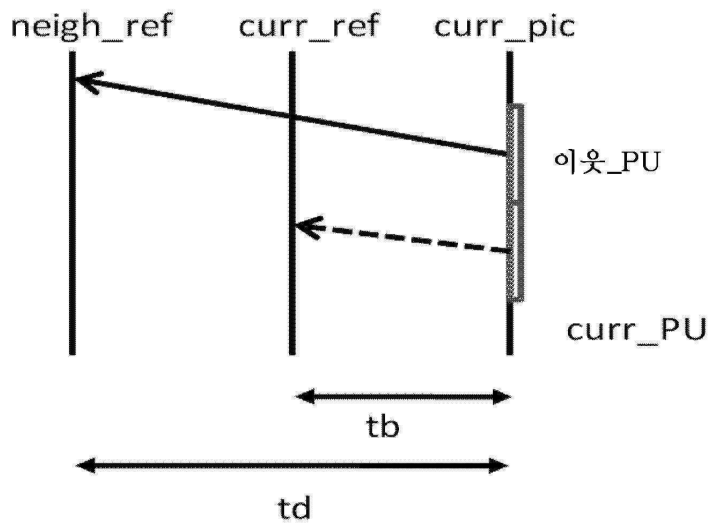
도면15



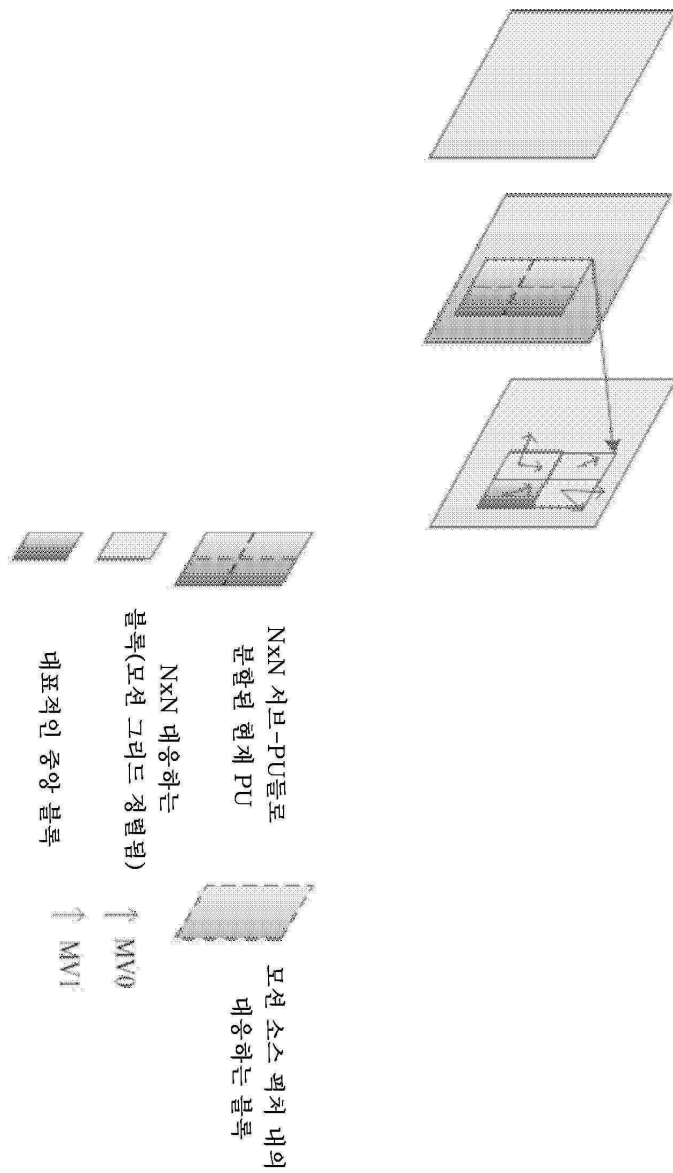
도면16



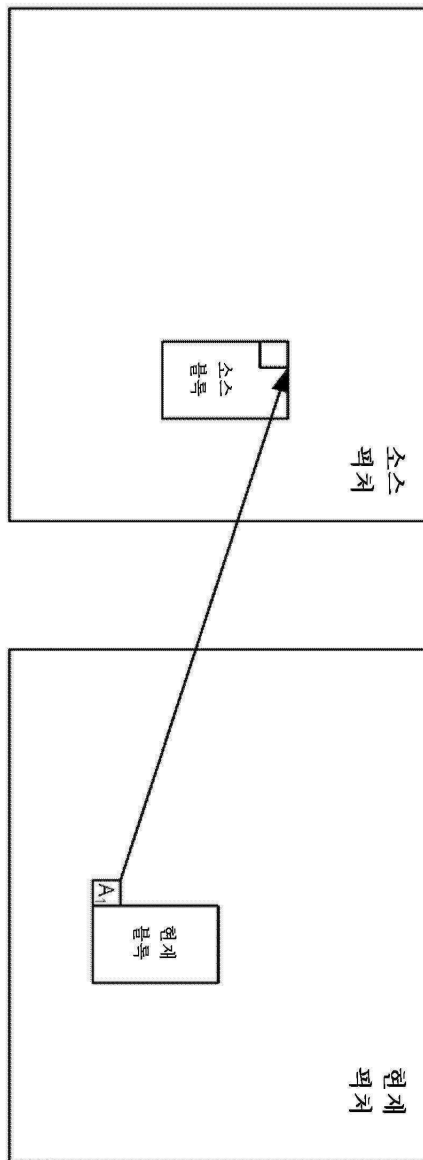
도면17



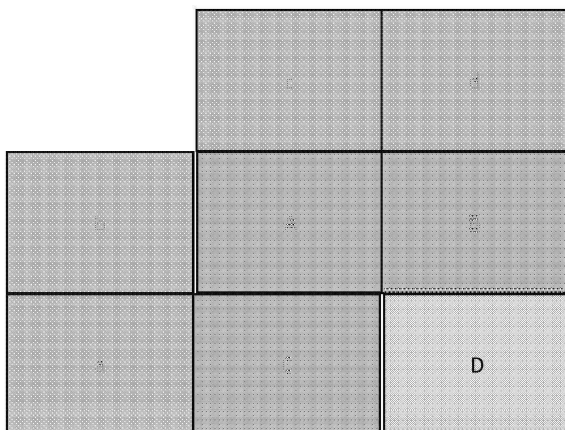
도면18



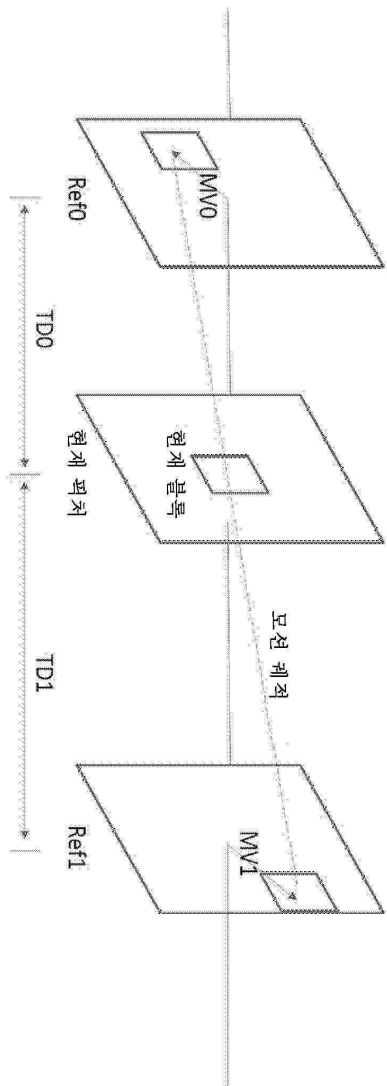
도면19



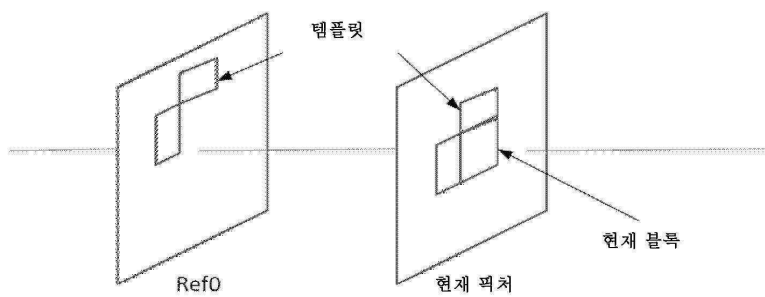
도면20



도면21

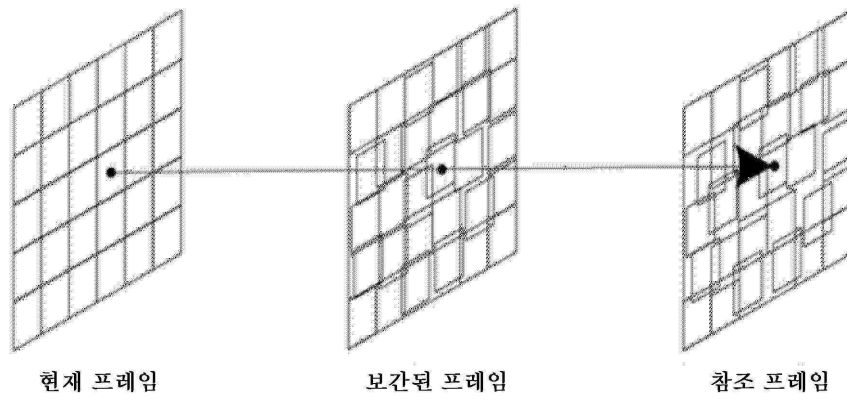


도면22

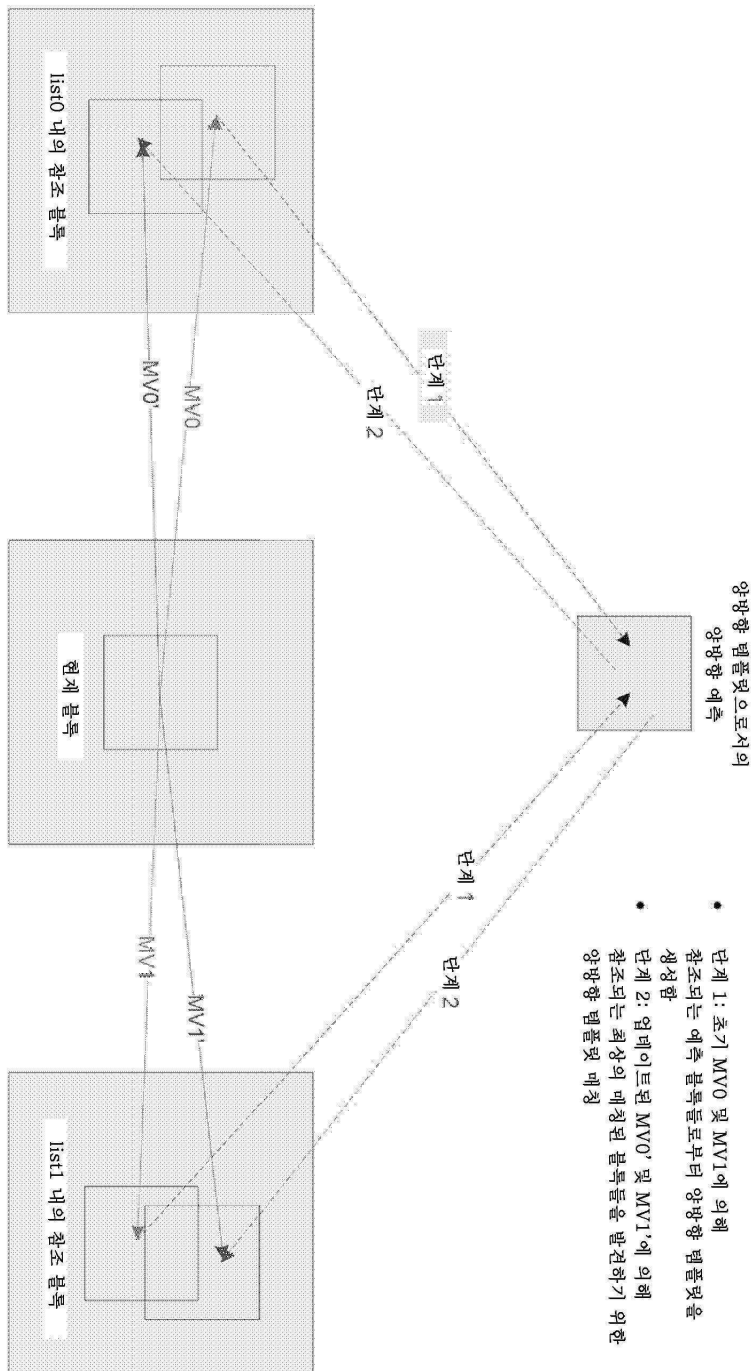




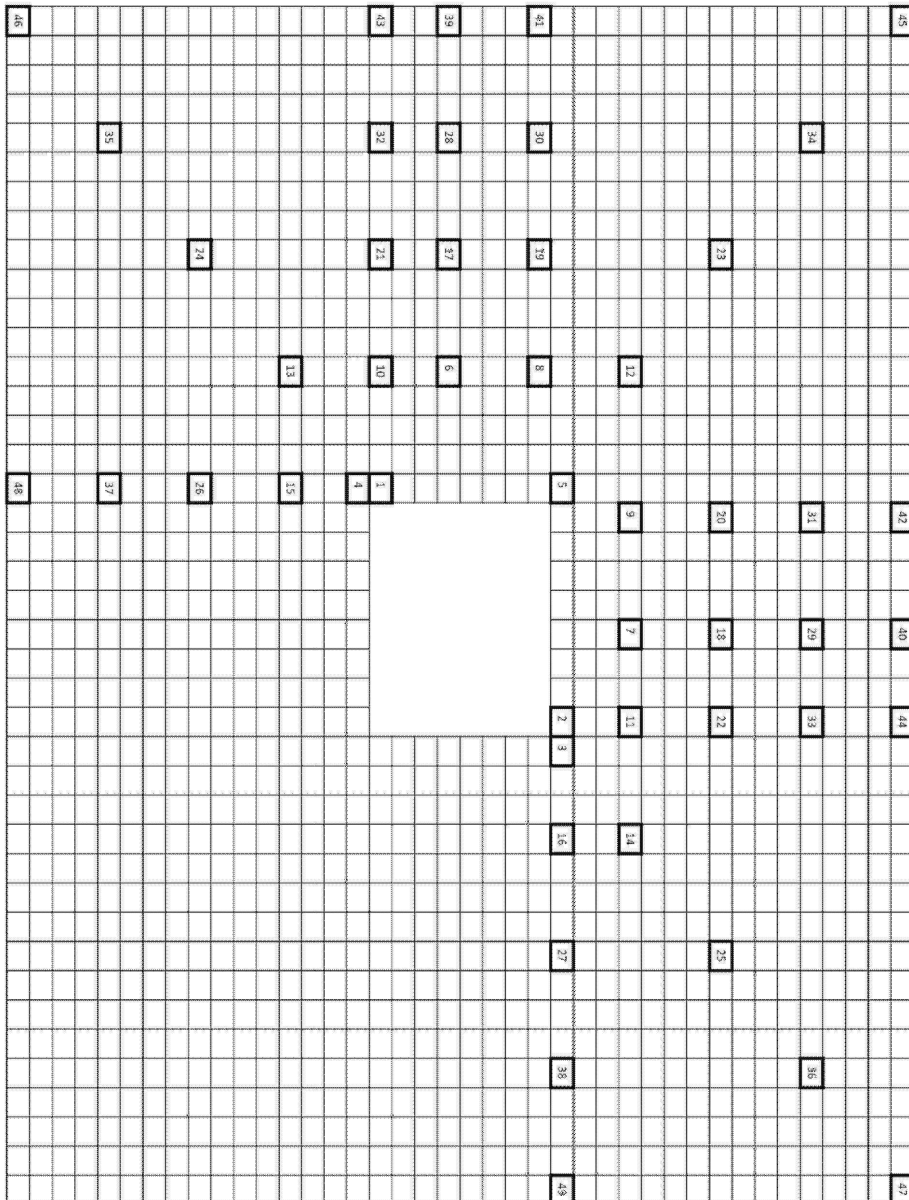
도면23



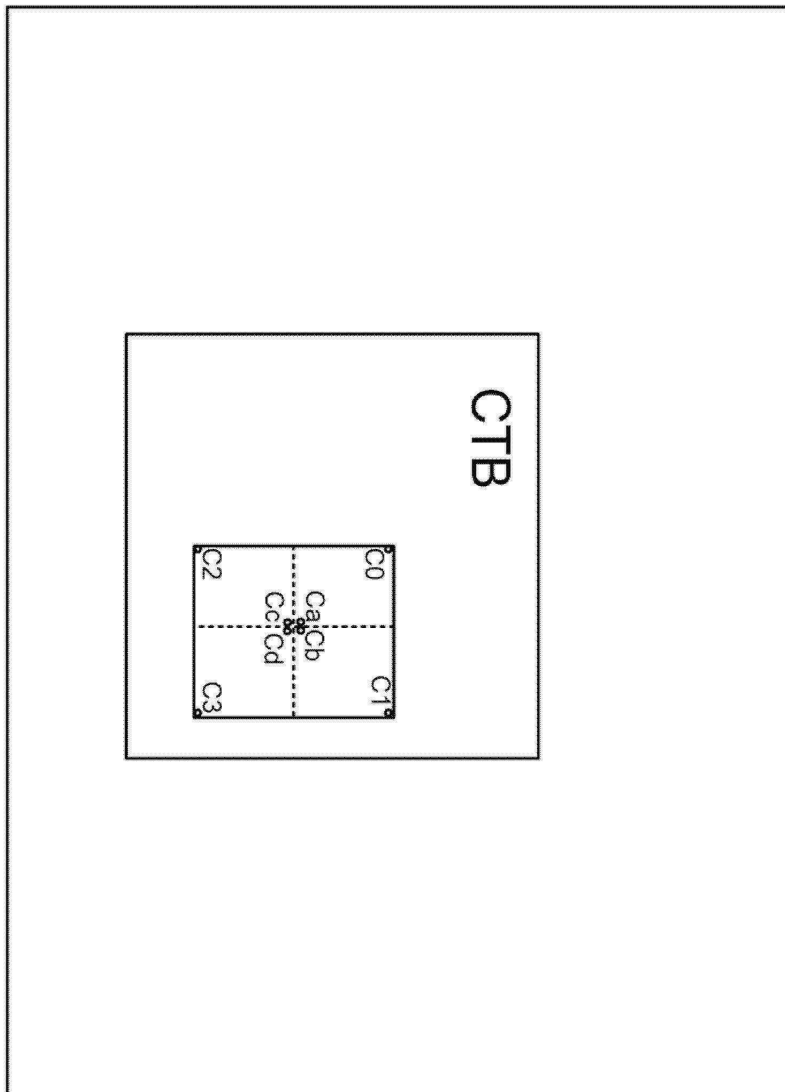
도면24



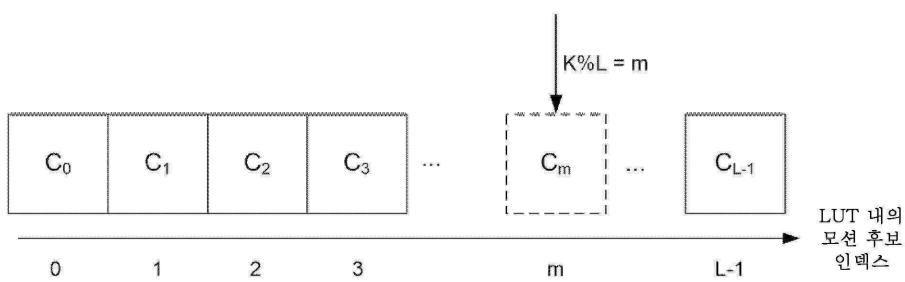
도면25



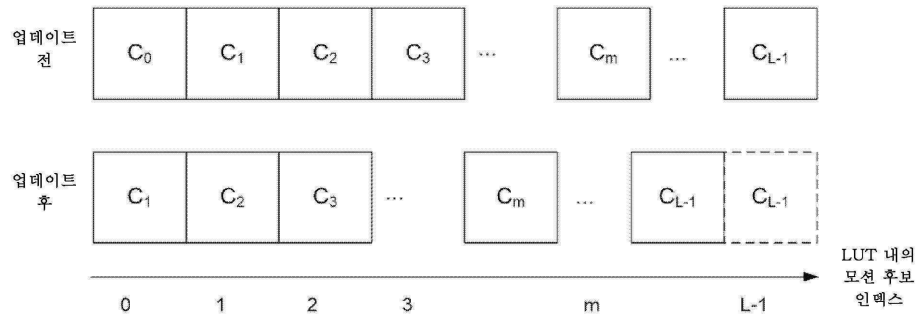
도면26



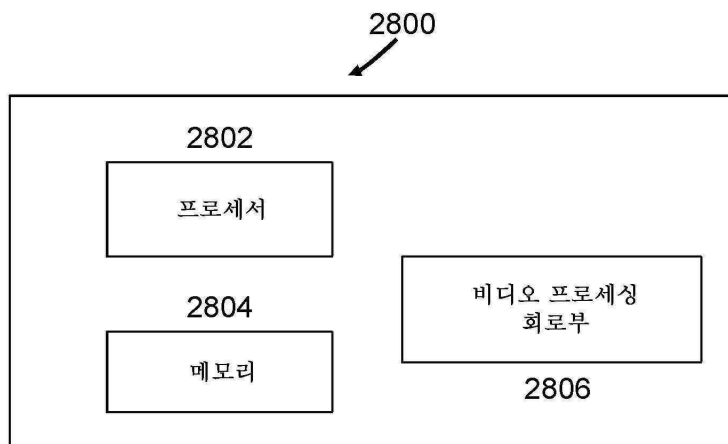
도면27a



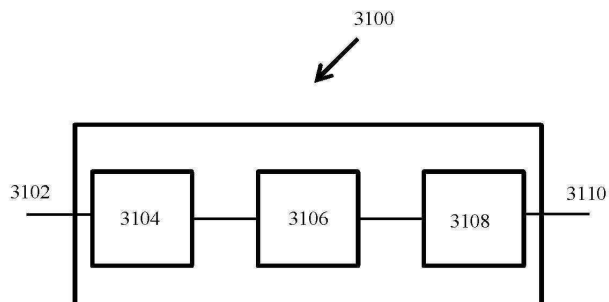
도면27b



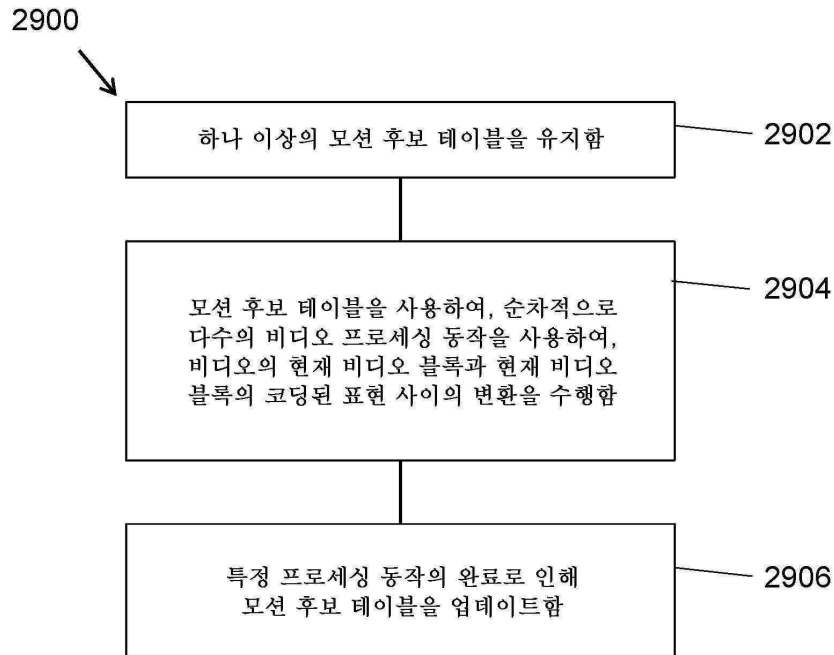
도면28a



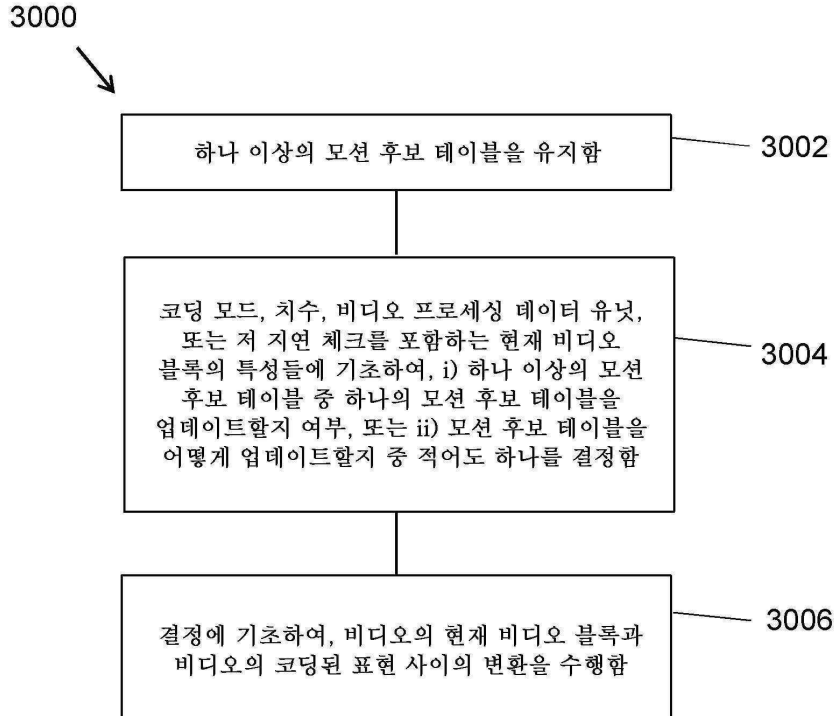
도면28b



도면29

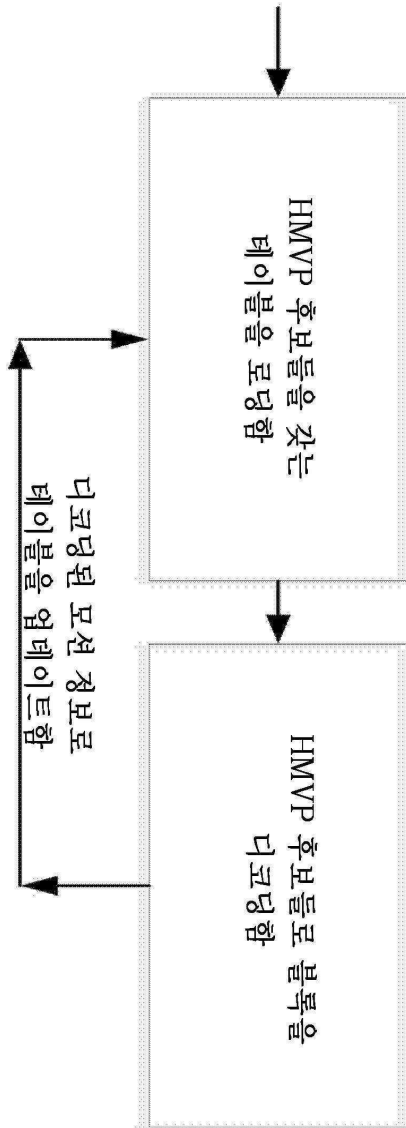


도면30

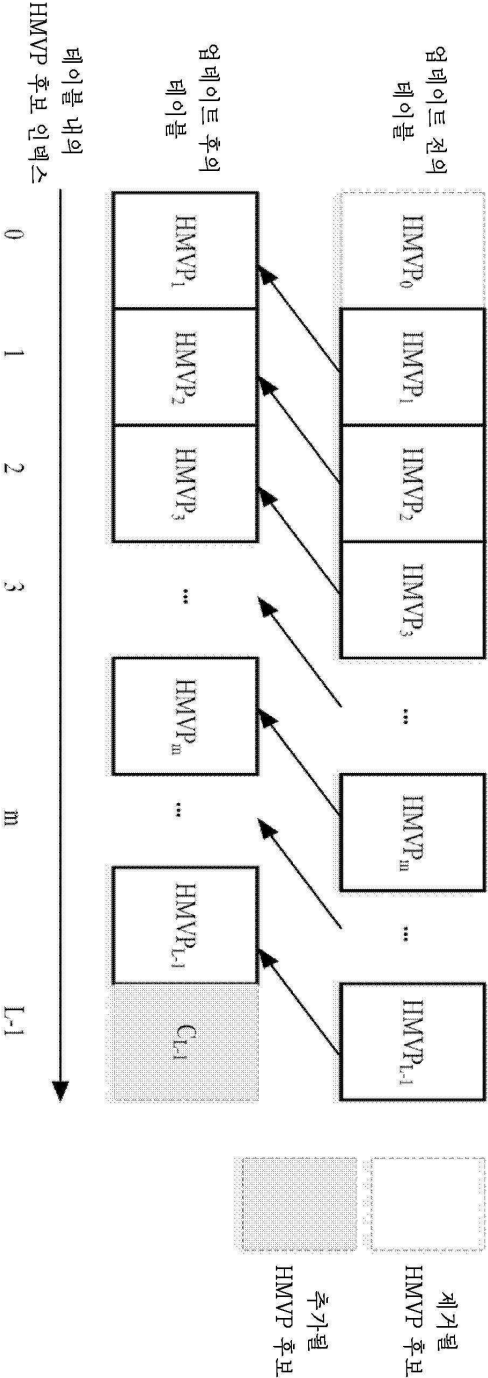




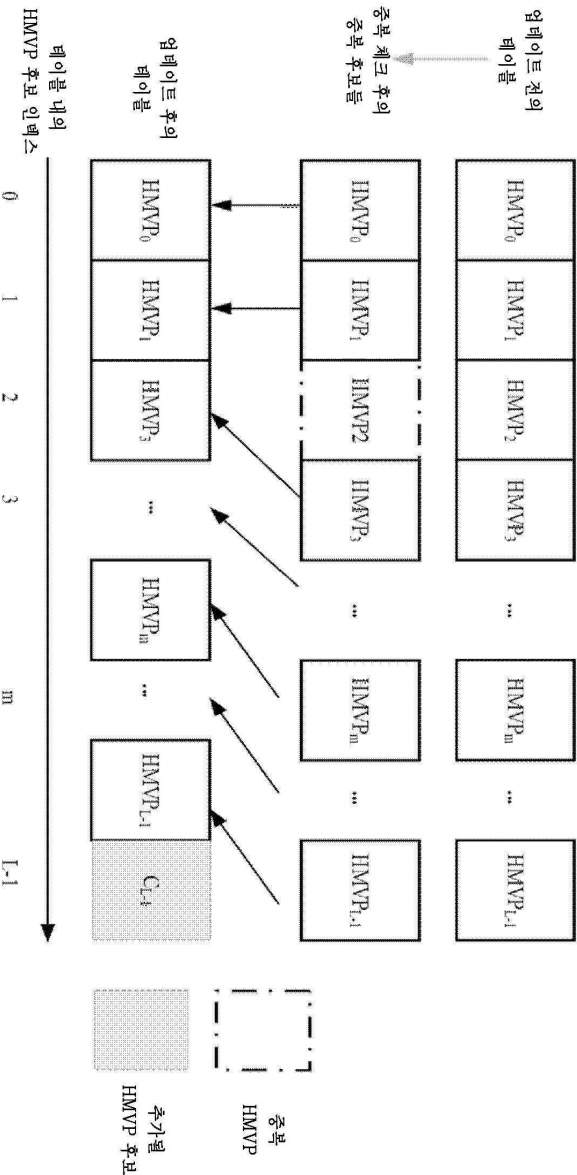
도면31



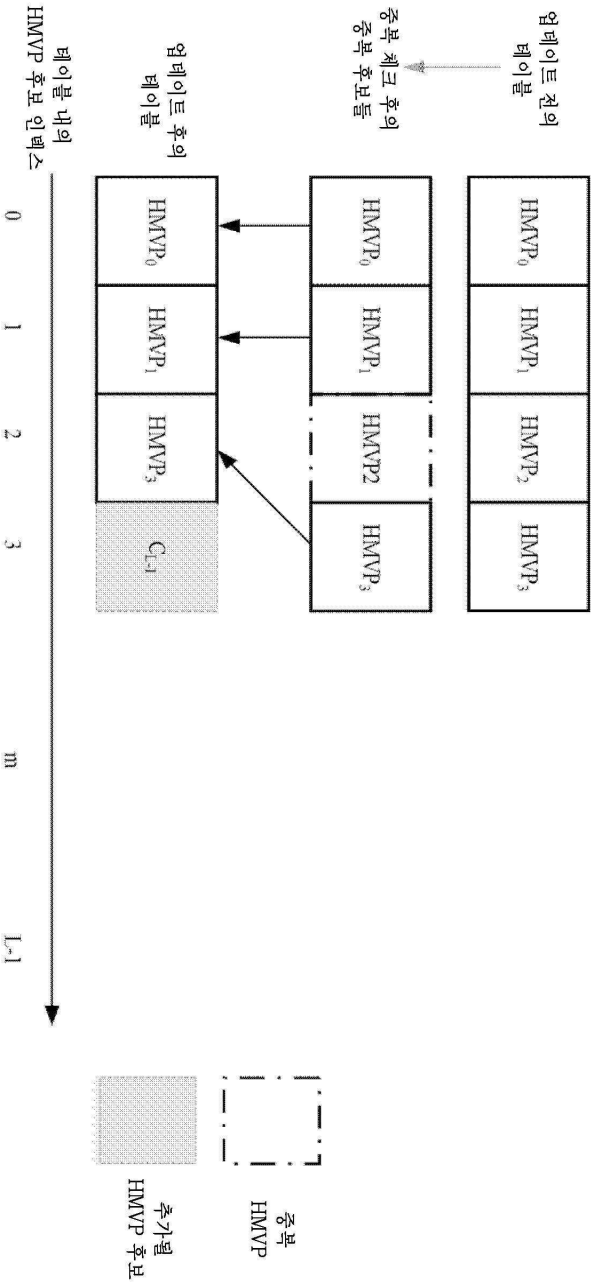
도면32



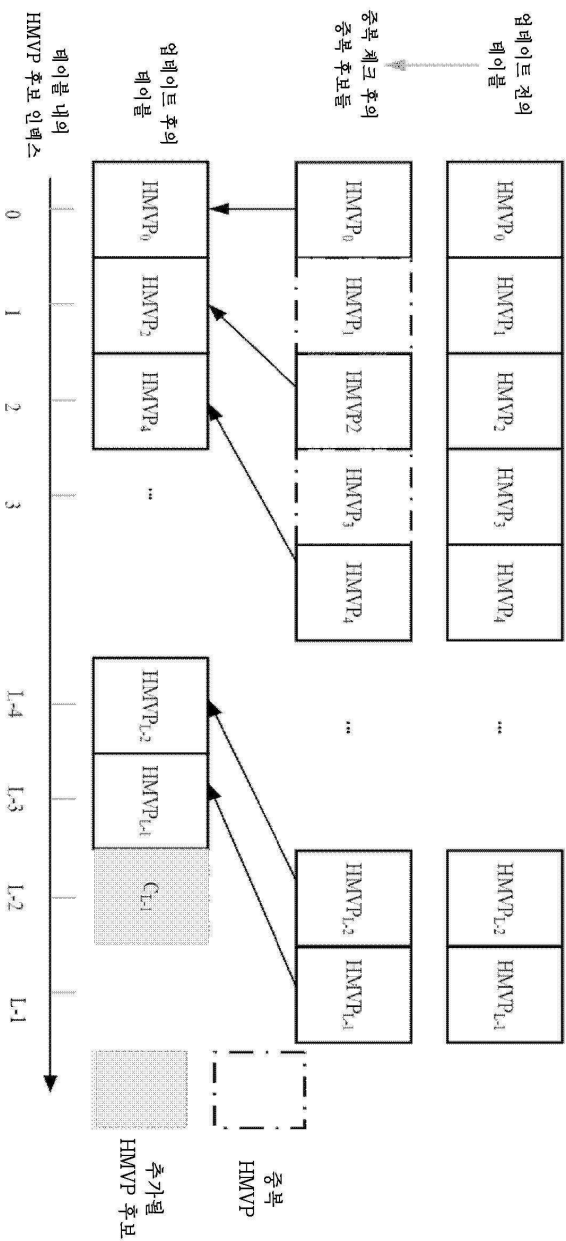
도면33a



도면33b



도면34a



도면34b

