



(19) 대한민국특허청(KR)

(12) 등록특허공보(B1)

(45) 공고일자 2016년03월29일

(11) 등록번호 10-1607426

(24) 등록일자 2016년03월23일

(51) 국제특허분류(Int. Cl.)

H04N 19/70 (2014.01) H04N 19/50 (2014.01)

(21) 출원번호 10-2014-7030835(분할)

(22) 출원일자(국제) 2011년11월04일

심사청구일자 2014년10월31일

(85) 번역문제출일자 2014년10월31일

(65) 공개번호 10-2014-0142363

(43) 공개일자 2014년12월11일

(62) 원출원 특허 10-2013-7014189

원출원일자(국제) 2011년11월04일

심사청구일자 2013년06월03일

(86) 국제출원번호 PCT/EP2011/069408

(87) 국제공개번호 WO 2012/059577

국제공개일자 2012년05월10일

(30) 우선권주장

61/410,246 2010년11월04일 미국(US)

(56) 선행기술조사문헌

US20130279577 A1

WO2012059577 A1

Anonymous: "Test Model under Consideration",

JCT-VC Meeting, 2010.4.15-23. JCT-VC A205.*

*는 심사관에 의하여 인용된 문헌

(73) 특허권자

지이 비디오 컴프레션, 엘엘씨

미국 뉴욕 12211 올버니 사우스우즈 블러바드 8

(72) 발명자

슈바르츠 헤이코

독일 베를린 13187 노이에 손홀저 스트라쎄 12

하이너 키르쉴페르

독일 베를린 10555 고츠코브스키 스트라쎄 5

(뒷면에 계속)

(74) 대리인

특허법인 정안

전체 청구항 수 : 총 4 항

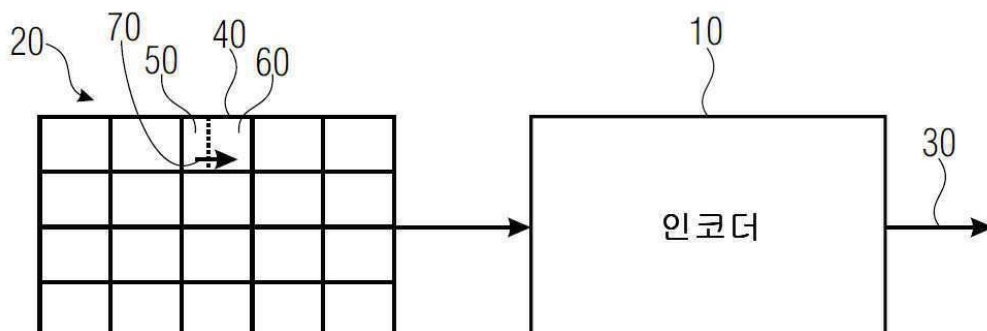
심사관 : 김영태

(54) 발명의 명칭 블록 병합 및 스킵 모드를 지원하는 화상 코딩

(57) 요약

코딩 효율성 증가는 병합의 활성화와 스킵 모드의 활성화 모두와 관련하여 비트스트림 내에서의 공동의 신호전달을 이용함으로써 달성된다. 즉, 비트스트림 내에서의 하나 이상의 구문 엘리먼트(syntax element)들의 잠재적인 상태들 중 하나의 상태는 화상의 현재 샘플 세트에 대해 각각의 샘플 세트가 병합될 것이고 비트스트림 내로 인코딩되고 삽입되는 예측 잔차가 없다는 것을 신호전달할 수 있다. 달리 말하면, 공동 플래그는 현재 샘플 세트와 연관된 코딩 파라미터들이 비트스트림으로부터 검색되거나 또는 병합 후보에 따라 설정될 것인지 여부와, 화상의 현재 샘플 세트가, 어떠한 잔차 데이터 없이, 현재 샘플 세트와 연관된 코딩 파라미터들에 의존하는 예측 신호에만 기초하여 재구성될 것인지 또는 비트스트림 내의 잔차 데이터에 의해 현재 샘플 세트와 연관된 코딩 파라미터들에 의존하여 예측 신호를 정제시킴으로써 재구성될 것인지 여부를 공동으로 신호전달할 수 있다.

대표도 - 도1



(72) 발명자

필립 헬레

독일 베를린 14129 아흐렌쇼페르 자일 4

사이먼 오우던

독일 베를린 10245 세우메스트라췌 29

얀 스테게만

독일 베를린 10245 심프론스트라췌 55

벤자민 브로스

독일 베를린 10245 가브리엘-막스-스트라췌 17

데트레브 마르페

독일 베를린 12161 쇼트베스트코르소 70

토마스 비에간드

독일 베를린 14195 오토-아펠-스트라췌 52

명세서

청구범위

청구항 1

블록 병합 및 스킵 모드를 지원하는 코더로서,

상기 코더는:

현재 블록에 대하여 상기 현재 블록에 대한 구문 엘리먼트(syntax element)의 가능한 상태들 중에서 미리결정된 하나의 상태를 이용하여 블록 병합 및 스킵 모드의 동시적인 활성화(concurrent activation)를 비트스트림 내에서 시그널링(signal)하도록 구성되는,

코더.

청구항 2

블록 병합 및 스킵 모드를 지원하는 디코더로서,

상기 디코더는:

비트스트림 내의 현재 블록에 대하여 블록 병합 및 스킵 모드를 동시적으로 활성화시키기 위하여, 구문 엘리먼트의 가능한 상태들 중에서 미리결정된 하나의 상태를 취하여 상기 현재 블록에 대한 구문 엘리먼트에 응답하도록(responsive) 구성되는,

디코더.

청구항 3

제 1 항에 있어서,

상기 코더는:

만약 상기 현재 블록에 대한 구문 엘리먼트가 상기 가능한 상태들 중의 미리결정된 하나의 상태를 취하지(assume) 않는 경우, 상기 현재 블록에 대하여 플래그를 비트스트림 내에서 조건적으로 시그널링하도록 구성되고,

상기 플래그는 스킵 모드가 비활성화된 상태에서 블록 병합의 활성화를 지시하는,

코더.

청구항 4

제 2 항에 있어서,

상기 디코더는:

만약 상기 현재 블록에 대한 구문 엘리먼트가 상기 가능한 상태들 중의 미리결정된 하나의 상태를 취하지 않는 경우, 상기 현재 블록에 대하여, 상기 비트스트림 내의 플래그에 대해 조건적으로 응답하도록 구성되고, 그리하여 상기 플래그에 의존하여 상기 스킵 모드가 비활성화된 상태에서 상기 블록 병합을 활성화시키도록 구성되는,

디코더.

발명의 설명

기술분야

[0001]

본 출원은 화상 및/또는 비디오 코딩에 관한 것이며, 보다 구체적으로는 블록 파티션화 및 스킵 모드를 지원하

는 코덱들에 관한 것이다.

배경 기술

- [0002] 많은 화상 및/또는 비디오 코덱들은 블록 단위로 화상들을 처리한다. 예를 들어, 예측 코덱(predictive codec)들은 한편으로 높은 공간 해상도로 설정되지만 예측 파라미터들을 위한 너무 많은 보조 정보를 소모시키는 매우 미세하게 설정되는 예측 파라미터들과, 다른 한편으로 예측 파라미터들의 보다 낮은 공간 해상도로 인해 예측 잔차(prediction residual)를 인코딩하는데 필요한 비트들의 양을 증가시키는 너무 거칠게(coarsely) 설정된 예측 파라미터들간에 양호한 절충을 달성하기 위해 블록 입도(block granularity)를 이용한다. 사실상, 예측 파라미터들에 대한 최적의 설정들은 양극단 사이에 놓여 있다.
- [0003] 위에서 개술한 문제에 대한 최적의 해법을 획득하기 위해 여러가지 시도들이 취해져 왔다. 예를 들어, 행들과 열들로 정규적으로 배열된 블록들로의 화상의 정규적인 하위분할(subdivision)을 이용하는 것 대신에, 멀티 트리 파티션화 하위분할은 하위분할 정보에 대한 합당한 요구로 화상을 블록들로 하위분할하는 자유도를 증가시키고자 한다. 그럼에도 불구하고, 심지어 멀티 트리 하위분할은 주목할만한 데이터양의 신호전달을 필요로 하며 화상을 하위분할하는데 있어서의 자유도는 심지어 이러한 멀티 트리 하위분할을 이용하는 경우에서도 매우 제한된다.
- [0004] 한편으로 화상 하위분할을 신호전달하기 위해 필요한 보조 정보의 양과, 다른 한편으로 화상을 하위분할하는데 있어서의 자유도간의 보다 나은 트레이드오프가 가능해지도록 하기 위해, 병합 정보를 신호전달하는데 필요한 적절한 양의 추가적인 데이터로 잠재적인 화상 하위분할의 횟수를 증가시키기 위해 블록들의 병합이 이용될 수 있다. 블록들이 병합되는 경우에, 코딩 파라미터들은, 마치 결과적으로 병합된 블록들의 그룹이 직접적으로 하위분할된 해당 화상의 부분이었던 것 처럼, 비트스트림 내에서 단 한번 완전히 송신될 필요가 있다.
- [0005] 화상 콘텐츠를 인코딩하는데 있어서 효율성을 추가적으로 증가시키기 위해, 스킵(skip) 모드가 몇몇의 블록 기반 화상 코덱들 내에 도입되었는데, 이러한 스킵 모드는 인코더가 일정한 블록의 잔차 데이터를 디코더로 송신하지 못하게 한다. 즉, 스킵 모드는 일정한 블록들에 대한 잔차 데이터 송신을 억제시킬 가능성이 있다. 일정한 블록들에 대한 잔차 데이터의 송신을 억제시키는 능력은 코딩/예측 파라미터들을 인코딩하기 위한 보다 넓은 입도 간격을 초래시키며 이 간격 내에서는 한편으로는 코딩 품질과 다른 한편으로는 소비된 전체 비트 레이트간의 최적의 트레이드오프가 예상될 수 있으며, 당연히, 코딩/예측 파라미터들의 인코딩의 공간적 해상도를 증가시키는 것은 보조 정보 레이트의 증가를 불러일으키는 반면에 레지둠(residuum)을 감소시켜서 잔차 데이터를 인코딩하는데 필요한 레이트를 낮춘다. 하지만, 스킵 모드의 이용가능성으로 인해, 레지둠은 매우 작아서 레지둠의 개별적인 송신이 생략될 수 있도록 코딩/예측 파라미터들이 송신될 때의 입도를 단순히 적절하게 한층 더 증가시킴으로써 급격한 코딩 레이트 절감을 획득하는 것이 바람직할 수 있다.
- [0006] 하지만, 블록 병합과 스킵 모드 활용의 결합에 의해 새롭게 야기된 남아있는 리던던시들로 인해, 보다 나은 코딩 효율을 달성할 필요성은 여전히 존재한다.

발명의 내용

해결하려는 과제

- [0007] 따라서, 본 발명의 목적은 증가된 코딩 효율을 갖는 코딩 개념을 제공하는 것이다. 이 목적은 계류중인 독립 청구항들에 의해 달성된다.

과제의 해결 수단

- [0008] 본 발명에 깔려있는 아이디어는, 병합의 활성화와 스킵 모드의 활성화 모두와 관련하여 비트스트림 내에서 공동의 시그널링(signaling)이 이용된다면 추가적인 코딩 효율 증가가 달성될 수 있다는 것이다. 즉, 비트스트림 내에서의 하나 이상의 구문 엘리먼트(syntax element)들의 잠재적인 상태들 중 하나의 상태는 화상의 현재 샘플 세트에 대해 각각의 샘플 세트가 병합될 것이고 비트스트림 내로 인코딩되고 삽입되는 예측 잔차를 갖지 않는다는 것을 시그널링할 수 있다. 달리 말하면, 공동 플래그는 현재 샘플 세트와 연관된 코딩 파라미터들이 비트스트림으로부터 검색되거나 또는 병합 후보에 따라 설정될 것인지 여부와, 화상의 현재 샘플 세트가, 어떠한 잔차 데이터 없이, 현재 샘플 세트와 연관된 코딩 파라미터들에 의존하는 예측 신호에만 기초하여 재구성될 것인지 또는 비트스트림 내의 잔차 데이터에 의해 현재 샘플 세트와 연관된 코딩 파라미터들에 의존하여 예측 신호를

정제(refining)시킴으로써 재구축될 것인지 여부를 공동으로 시그널링할 수 있다.

[0009] 병합과 스킵 모드가 동시에 활성화되지 않는 경우에 병합 및/또는 스킵 모드의 활성화를 서로 개별적으로 시그널링하기 위한 추가적인 오버헤드는 감소될 수 있거나 또는 단순 소모될 필요만 있을 뿐이기 때문에, 한편으로 병합 활성화의 공동 시그널링과 다른 한편으로 스킵 모드의 활성화의 이러한 도입은 비트 레이트를 절감시킨다는 것을 본 발명의 발명자들은 발견하였다.

[0010] 본 발명의 유리한 구현들은 첨부된 종속 청구항들의 발명내용이다.

도면의 간단한 설명

[0011] 아래에서는 본 출원의 바람직한 실시예들을 도면들과 관련하여 자세하게 설명하며, 이러한 도면들 중에서, 도 1은 실시예에 따른 인코딩을 위한 장치의 블록도를 도시한다. 도 2는 보다 상세한 실시예에 따른 인코딩을 위한 장치의 블록도를 도시한다. 도 3은 실시예에 따른 디코딩을 위한 장치의 블록도를 도시한다. 도 4는 보다 상세한 실시예에 따른 디코딩을 위한 장치의 블록도를 도시한다. 도 5는 도 1 또는 도 2의 인코더의 잠재적인 내부 구조의 블록도를 도시한다. 도 6은 도 3 또는 도 4의 디코더의 잠재적인 내부 구조의 블록도를 도시한다. 도 7a는 화상의 트리 루트 블록들, 코딩 유닛들(블록들) 및 예측 유닛들(파티션들)로의 잠재적인 하위분할을 개략적으로 도시한다. 도 7b는 일례에 따라, 7a에서 도시된 트리 루트 블록의 하위분할 트리들, 파티션들의 레벨 내림순으로 도시한다. 도 8은 실시예에 따른 잠재적인 지원된 파티션화 패턴들의 세트에 대한 실시예를 도시한다. 도 9는 도 8에 따른 블록 파티션화를 이용할 때 블록 병합과 블록 파티션화를 결합함으로써 결과적으로 초래되는 잠재적인 파티션화 패턴들을 도시한다. 도 10은 실시예에 따른 SKIP/DIRECT 모드에 대한 후보 블록들을 개략적으로 도시한다. 도 11 내지 도 13은 실시예에 따른 구문(syntax)의 구문 부분들을 도시한다. 도 14는 실시예에 따른 파티션을 위한 이웃 파티션들의 정의를 개략적으로 도시한다.

발명을 실시하기 위한 구체적인 내용

[0012] 아래의 설명과 관련하여, 상이한 도면들과 관련하여 동일한 참조 부호가 이용될 때 마다 이러한 도면들 중 하나의 도면과 관련하여 표현된 각각의 엘리먼트에 관한 설명들은 하나의 도면으로부터 다른 도면으로의 설명들의 이러한 전달이 이러한 다른 도면의 나머지 설명과 충돌하지 않는 한 다른 도면들에도 똑같이 적용될 것이라는 점을 유념해둔다.

[0013] 도 1은 화상(20)을 비트스트림(30) 내로 인코딩하기 위한 장치(10)를 도시한다. 당연히, 화상(20)은 비디오의 일부일 수 있고, 이 경우 인코더(10)는 비디오 인코더일 것이다.

[0014] 화상(20)은, 비록 도 1에서는 명시적으로 도시되어 있지 않지만, 샘플들의 어레이로서 표현된다. 화상(20)의 샘플 어레이는, 화상(20)의 오버랩하지 않는 단일 연결된 영역들을 커버하는 샘플 세트들과 같은 임의의 샘플 세트들일 수 있는 샘플 세트들(40)로 파티션화된다. 이해를 용이하게 하기 위해, 아래에서는 샘플 세트들(40)을 블록들(40)로서 도시하고 호칭할 것이지만, 아래의 설명은 임의의 특정 종류의 샘플 세트들(40)로 국한되는 것으로서 간주되어서는 안된다. 구체적인 실시예에 따르면, 샘플 세트들(40)은 직사각형 및/또는 정방형 블록들이다.

[0015] 예를 들어, 화상(20)은, 블록들(40)이 도 1에서 예시적으로 도시된 바와 같은 열과 행으로 배열되도록, 블록들(40)의 규칙적인 배열로 하위분할될 수 있다. 하지만, 블록들(40)로의 화상(20)의 임의의 다른 하위분할이 또한 가능할 수 있다. 특히, 블록들(40)로의 화상(20)의 하위분할은 고정될 수 있고, 즉 디폴트에 의해 디코더에 알려져 있거나 또는 비트스트림(30) 내에서 디코더에 신호전달될 수 있다. 특히, 화상(20)의 블록들(40)은 크기가

다를 수 있다. 예를 들어, 쿼드 트리 하위분할과 같은 멀티 트리 하위분할이 화상(20)에 적용될 수 있거나 또는 규칙적으로 배열된 트리 루트 블록들로의 화상(20)의 규칙적인 사전 하위분할에 적용되어서 블록들(40)을 획득할 수 있는데, 이 경우, 이 블록들(40)은 트리 루트 블록들의 멀티 트리 하위분할의 리프(leaf) 블록들을 형성한다.

[0016]

어떠한 경우든지, 인코더(10)는, 현재의 샘플 세트(40)에 대해, 현재 샘플 세트(40)와 연관된 코딩 파라미터들이 병합 후보에 따라 설정될 것인지 또는 비트스트림(30)으로부터 검색될 것인지 여부와, 화상(20)의 현재 샘플 세트가, 어떠한 잔차 데이터도 없이, 현재 샘플 세트와 연관된 코딩 파라미터들에 의존하는 예측 신호에만 기초하여 재구성될 것인지, 또는 비트 스트림(30)내의 잔차 데이터에 의해 현재 샘플 세트(40)와 연관된 코딩 파라미터들에 의존하여 예측 신호를 정제시킴으로써 재구성될 것인지 여부를 공동으로 신호전달하는 플래그를 비트스트림(30) 내로 인코딩하도록 구성된다. 예를 들어, 인코더(10)는, 현재의 샘플 세트(40)에 대해, 제1 상태를 취하는 경우, 현재 샘플 세트(40)와 연관된 코딩 파라미터들이 비트스트림(30)으로부터 검색되기 보다는 병합 후보에 따라 설정될 것이라는 것과, 화상(20)의 현재 샘플 세트가, 어떠한 잔차 데이터도 없이, 현재 샘플 세트와 연관된 코딩 파라미터들에 의존하는 예측 신호에만 기초하여 재구성될 것이라는 것과, 임의의 다른 상태를 취하는 경우, 현재 샘플 세트(40)와 연관된 코딩 파라미터들이 비트스트림(30)으로부터 검색될 것이거나, 또는 화상(20)의 현재 샘플 세트가 비트스트림(30) 내의 잔차 데이터에 의해 현재 샘플 세트(40)와 연관된 코딩 파라미터들에 의존하여 예측 신호를 정제시킴으로써 재구성될 것이라는 것을 공동으로 신호전달하는 플래그를 비트스트림(30) 내로 인코딩하도록 구성된다. 이것은 다음을 의미한다. 인코더(10)는 블록들(40)의 병합을 지원한다. 병합은 임의적이다. 즉, 모든 블록(40)이 병합을 거치는 것만은 아니다. 몇몇의 블록들(40)의 경우, 예컨대 어느정도 레이트 왜곡 최적화 관점에서, 현재 블록(40)을 병합 후보와 병합시키는 것이 바람직하지만, 다른 블록들의 경우에는 그 반대가 바람직하다. 어떠한 블록(40)이 병합을 거쳐야 하는지 여부를 결정하기 위해, 인코더(10)는 병합 후보들의 세트 또는 리스트를 결정하고, 이러한 병합 후보들 각각에 대해, 현재 블록(40)을 해당 병합 후보와 병합시키는 것이 예컨대 레이트 왜곡 최적화 관점에서 가장 바람직한 코딩 옵션을 형성하는지 여부를 체크한다. 인코더(10)는 비트스트림(30)의 이전에 인코딩된 부분들에 기초하여 현재 블록(40)에 대한 병합 후보들의 리스트 또는 세트를 결정하도록 구성된다. 예를 들어, 인코더(10)는 인코더(10)에 의해 적용된 인코딩 순서에 따라 이전에 인코딩되었던 국부적으로 및/또는 시간적으로 이웃하는 블록들(40)과 연관된 코딩 파라미터들을 채택함으로써 병합 후보들의 리스트 또는 세트의 적어도 일부분을 유도해낸다. 시간적 이웃은, 예컨대 화상(20)이 속해있는 비디오의 이전에 인코딩된 화상들의 블록들을 가리키며, 시간적으로 이웃한 블록들은 현재 화상(20)의 현재 블록(40)과 공간적으로 오버랩하도록 공간적으로 위치한다. 따라서, 병합 후보들의 리스트 또는 세트의 이러한 부분에 대해, 각각의 병합 후보와 공간적으로 및/또는 시간적으로 이웃하는 블록들간의 일대일 연관이 존재한다. 각각의 병합 후보는 연관된 코딩 파라미터들을 갖는다. 현재 블록(40)이 임의의 병합 후보들과 병합되면, 인코더(10)는 병합 후보에 따라 현재 블록(40)의 코딩 파라미터들을 설정한다. 예를 들어, 인코더(10)는 현재 블록(40)의 코딩 파라미터들을 각각의 병합 후보와 동등해지도록 설정할 수 있는데, 즉 인코더(10)는 각각의 병합 후보로부터 현재 블록(40)의 코딩 파라미터들을 복사할 수 있다. 따라서, 병합 후보들의 세트 또는 리스트의 이러한 방금 개술한 부분에 대해, 병합 후보의 코딩 파라미터들은, 공간적으로 및/또는 시간적으로 이웃하는 블록으로부터 직접 채용되거나, 또는 각각의 병합 후보의 코딩 파라미터들은 위 사항을 채용함으로써, 즉 병합 후보를 이웃 블록들에 대해 동등하게 설정하되 도메인 변동들을 고려함으로써, 예컨대 도메인 변동에 따라 상기 채용된 코딩 파라미터들을 스케일링(scaling)함으로써, 이러한 공간적으로 및/또는 시간적으로 이웃하는 블록의 코딩 파라미터들로부터 획득된다. 예를 들어, 병합되는 코딩 파라미터들의 적어도 일부분은 모션 파라미터들을 망라할 수 있다. 하지만, 모션 파라미터들은 상이한 참조 화상 인덱스들을 가리킬 수 있다. 보다 정확하게 하기 위해, 채용될 모션 파라미터들은 현재 화상과 참조 화상사이의 일정한 시간 간격을 가리킬 수 있고, 각각의 모션 파라미터들을 갖는 각각의 병합 후보를 현재 블록과 병합할 때, 인코더(10)는 병합 후보의 시간 간격을 현재 블록을 위해 선택된 시간 간격으로 적응시키기 위해 각각의 병합 후보의 모션 파라미터들을 스케일링하도록 구성될 수 있다.

[0017]

어떠한 경우던간에, 지금까지 설명한 병합 후보들은 이들 모두가 연관된 코딩 파라미터들을 가지며, 이러한 병합 후보들과 이웃 블록들간의 일대일 연관이 존재한다는 점을 공통적으로 갖는다. 따라서, 코딩 파라미터들이 스케일링 적응 등을 제외하고, 블록들(40)의 이러한 그룹들 내에서 화상(20)에 걸쳐 달라지지 않도록, 블록들(40)을 방금 개술한 임의의 병합 후보들과 병합시키는 것은 이러한 블록들을 하나 이상의 블록들(40)의 그룹들로 병합시키는 것으로서 간주될 수 있다. 사실상, 방금 개술한 임의의 병합 후보들과의 병합은 화상(20)에 걸쳐 코딩 파라미터들이 변하는 입도(granularity)를 감소시킨다. 이 외에, 방금 개술한 임의의 병합 후보들과의 병합은 화상(40)을 블록들(40)과 블록들(40)의 그룹들로 각각 하위분할시키는데 있어서 추가적인 자유도를 불러일

으킨다. 따라서, 이러한 관점에서 이러한 블록들의 그룹들로의 블록들(40)의 병합은 인코더(10)로 하여금 화상(20)에 걸쳐 변하는 코딩 파라미터들을 이용하여 이러한 블록들(40)의 그룹들의 유닛들에서 화상(20)을 인코딩하게 하는 것으로서 간주될 수 있다.

[0018] 방금 언급한 병합 후보들 이외에, 인코더(10)는 또한 병합 후보들의 리스트/세트에, 두 개 이상의 이웃 블록들의 코딩 파라미터들, 즉 코딩 파라미터들의 산술 평균, 기하 평균 또는 이웃 블록들의 코딩 파라미터들의 중간값 등의 조합의 결과물인 병합 후보들을 추가시킬 수 있다.

[0019] 따라서, 사실상, 인코더(10)는 블록들(40)로의 화상(20)의 하위분할에 의해 정의된 입도에 비교하여 코딩 파라미터들이 비트스트림(30) 내에서 명시적으로 송신되는 입도를 감소시킨다. 이러한 블록들(40) 중 몇몇은 위에서 개술한 병합 옵션의 이용에 의해 하나의 동일한 코딩 파라미터들을 이용하여 블록들의 그룹들을 형성한다. 몇몇의 블록들은 병합을 통해 서로 결합되지만, 각각의 스케일링 적응들 및/또는 결합 함수들을 통해 서로간에 상관된 상이한 코딩 파라미터들을 이용한다. 몇몇의 블록들(40)은 병합처리되지 않으며, 이에 따라 인코더(10)는 코딩 파라미터들을 비트스트림(30) 내로 곧바로 인코딩한다.

[0020] *인코더(10)는 화상(20)에 대한 예측 신호를 결정하기 위해 이에 따라 정의된 블록들(40)의 코딩 파라미터들을 이용한다. 인코더(10)는 예측 신호가 각각의 블록(40)과 연관된 코딩 파라미터들에 의존한다는 점에서 이러한 예측 신호의 결정을 블록별로 수행한다.

[0021] 인코더(10)에 의해 수행되는 또다른 결정은 레지둠(residuum), 즉 현재 블록(40)의 각각의 로컬 영역에서의 예측 신호와 원래의 화상 콘텐츠간의 차이가 비트스트림(30) 내에서 송신되는지 아닌지의 여부이다. 즉, 인코더(10)는 블록들(40)에 대해 스킵 모드가 각각의 블록에 적용되어야 하는지 아닌지 여부를 결정한다. 스킵 모드가 적용되는 경우, 인코더(10)는 현재 부분(40) 내의 화상(20)을 단지 각각의 블록(40)과 연관된 코딩 파라미터들로부터 유도된 예측 신호의 형태로, 또는 이러한 코딩 파라미터들에 의존하여 인코딩하고, 스킵 모드가 선택해제되는 경우, 인코더(10)는 예측 신호뿐만 아니라 잔차 데이터 모두를 이용하여 블록(40) 내의 비트스트림(30) 내로 화상(20)을 인코딩한다.

[0022] 한편으로는 병합과 관련하여 다른 한편으로는 스킵 모드와 관련한 결정을 신호전달하기 위한 비트 레이트를 절감시키기 위해, 인코더(10)는 블록(40)에 대한 하나의 플래그를 이용하여 이 모두의 결정들을 공동으로 신호전달한다. 보다 정확히 하기 위해, 공동의 신호전달은, 제1 잠재적 플래그 상태를 취하는 경우, 병합과 스킵 모드 모두의 활성화가 각각의 블록(40)의 플래그에 의해 비트스트림(30) 내에서 공동으로 표시되도록 실현될 수 있는 반면에, 플래그의 나머지 다른 플래그 상태는 병합 또는 스킵 모드 중 어느 하나가 활성화되어 있지 않다는 것을 디코더에게 단순히 표시한다. 예를 들어, 인코더(10)는 일정한 블록(40)에 대해 병합을 활성화시키되, 스킵 모드를 비활성화시킬 것을 결정할 수 있다. 이 경우, 인코더(10)는 비트스트림(30) 내에서 병합과 스킵 모드 중 적어도 하나의 비활성화를 신호전달하고, 그 후 또다른 플래그의 이용에 의해 예컨대 병합의 활성화를 비트스트림(30) 내에서 신호전달하기 위해 다른 플래그 상태를 이용한다. 따라서, 인코더(10)는 블록(40)에 대해 병합과 스킵 모드가 동시에 활성화되지 않은 경우에만 이러한 추가적인 플래그를 송신해야 한다. 아래에서 추가적으로 설명되는 실시예들에서, 제1 플래그를 `mr_g_cbf` 또는 `skip_flag`라고 부르는 반면에 보조 병합 표시자 플래그를 `mr_g` 또는 `merge_flag`라고 부른다. 병합 및 스킵 모드의 활성화를 공동으로 신호전달하기 위한 이러한 하나의 신호전달 상태의 공동 이용은 비트스트림(30)의 총체적인 비트 레이트를 감소시킨다는 것을 본 출원의 발명자들에 의해 발견되어왔다.

[0023] 방금 언급한 신호전달 상태와 관련하여, 이러한 신호전달 상태는 비트스트림(30)의 하나의 비트의 상태에 의해 결정될 수 있다는 것을 유념해야 한다. 하지만, 인코더(10)는 비트스트림(30)을 엔트로피 인코딩하도록 구성될 수 있고, 이에 따라 비트스트림(30)과 플래그의 신호전달 상태간의 대응은 보다 복잡해질 수 있다. 이 경우, 신호전달 상태는 엔트로피 디코딩된 도메인에서 비트스트림(30)의 하나의 비트에 대응할 수 있다. 더 나아가, 신호전달 상태는 가변 길이 코딩 방식에 따라 코드 워드들이 할당된 플래그의 두 개의 상태들 중 하나의 상태에 대응할 수 있다. 산술 코딩의 경우, 병합과 스킵 모드의 활성화를 공동으로 신호전달하는 신호전달 상태는 산술 인코딩 방식의 근간을 이루는 심볼 알파벳의 심볼들 중 하나에 대응할 수 있다.

[0024] 위에서 개술한 바와 같이, 인코더(10)는 비트스트림(30) 내의 플래그를 이용하여 병합 및 스킵 모드의 동시적인 활성화를 신호전달한다. 아래에서 보다 자세하게 개술하겠지만, 이러한 플래그는 두 개보다 많은 잠재적인 상태들을 갖는 구문 엘리먼트(syntax element) 내에서 송신될 수 있다. 이러한 구문 엘리먼트는, 예컨대, 다른 코딩

옵션들도 신호전달할 수 있다. 아래에서는 이러한 내용을 보다 자세하게 설명한다. 하지만, 이 경우, 하나 이상의 구문 엘리먼트들의 잠재적인 상태들 중 하나의 상태는 동시적인 활성화를 신호전달한다. 즉, 방금 언급한 현재 블록(40)의 구문 엘리먼트가 이러한 미리결정된 잠재적인 상태를 취할 때 마다, 인코더(10)는 병합 및 스킵 모드 모두의 활성화를 신호전달한다. 따라서 디코더는 병합의 활성화 및 스킵 모드의 활성화 각각에 관한 추가적인 신호전달을 필요로 하지 않는다.

[0025]

위에서 개술한 설명과 관련하여, 블록들(40)로의 화상(20)의 파티션화는 화상(20)에 대해 코딩 파라미터들이 결정되는 최고로 미세한 해상도를 표현하지 않을 수 있다. 이보다는, 인코더(10)는 현재 블록(40)을 하위블록들(50, 60), 즉 샘플 하위세트들로 각각 파티션화하기 위한 지원된 파티션화 패턴들 중 하나를 비트스트림(30) 내에서 신호전달하기 위해, 각각의 블록(40)에 추가적인 파티션화 정보를 동반시킬 수 있다. 이 경우, 동시적인 병합/스킵 결정은 블록들(40)의 유닛들에서 인코더(10)에 의해 수행되는 반면에, 예컨대 서로 분리된 보조 병합 결정 및/또는 스킵 모드 결정과 함께 코딩 파라미터들은 화상(20)에 대해 블록들(40)의 하위파티션의 유닛들, 즉 도 1에서 예시적으로 도시된 블록(40) 내의 하위블록들(50, 60)의 유닛들로 정의된다. 당연히, 지원된 파티션화 패턴들 중 하나를 비파티션화 모드가 나타낼 수 있고, 이에 따라 인코더(10)가 블록(40)에 대한 코딩 파라미터들의 하나의 세트를 단순히 결정하는 것을 초래시킨다. 각각의 파티션화 패턴의 하위블록들(50, 60)의 갯수와 상관없이, 병합 결정은 모든 하위블록들, 즉 하나 이상의 하위블록들에 적용될 수 있다. 즉, 병합이 블록(40)에 대해 활성화되면, 이 활성화는 모든 하위블록들에 대해 유효할 수 있다. 아래에서 추가적으로 개술되는 실시예에 따르면, 병합과 스킵 모드의 활성화를 공동으로 신호전달하는 앞서언급한 공동 상태는 현재 블록(40)에 대해 지원된 파티션화 패턴들 중에서 비파티션화 패턴을 동시에 추가적으로 신호전달할 수 있고, 이로써 구문 엘리먼트 또는 플래그가 이러한 상태를 취하는 경우, 현재 블록에 대한 파티션화 정보의 추가적인 송신은 불필요하다. 당연히, 병합 및 스킵 모드의 활성화에 더하여 지원된 파티션화 패턴 중 임의의 다른 파티션화 패턴이 대안적으로 동시에 표시될 수 있다.

[0026]

본 출원의 몇몇의 실시예들에 따르면, 인코더(10)는 한편으로 블록들(40)의 블록 파티션화와 다른 한편으로 하위블록들(50, 60)의 병합의 공동 이용으로부터 초래된 비트 효율성 패널티를 피한다. 보다 정확히 하기 위해, 인코더(10)는 예컨대 어느정도 레이트 왜곡 최적화 관점에서 추가적인 파티션 블록(40)에 비해 더 나은지 여부에 대해 결정내릴 수 있고, 일정한 코딩 파라미터들이 화상(20)의 현재 블록(40) 내에서 정의되거나 또는 설정된 입도를 적응시키기 위해 현재 블록(40)에 대해 지원된 파티션화 패턴들 중 어느 것을 이용해야 하는지에 대해 결정내릴 수 있다. 아래에서 보다 자세하게 개술하는 바와 같이, 코딩 파라미터들은, 예컨대 인터(inter) 예측 파라미터들과 같은 예측 파라미터들을 나타낼 수 있다. 이러한 인터 예측 파라미터들은, 예컨대 참조 화상 인덱스, 모션 벡터 등을 포함할 수 있다. 지원된 파티션화 패턴들은, 예컨대, 비파티션화 모드, 즉 현재 블록(40)이 추가적으로 파티션화되지 않는 옵션, 수평적 파티션화 모드, 즉 현재 블록(40)이 수평하게 연장하는 라인을 따라 상부 또는 상단 부분과 바닥 또는 하단 부분으로 하위분할되는 옵션, 및 수직적 파티션화 모드, 즉 현재 블록(40)이 수직하게 연장하는 라인을 따라 좌측 부분과 우측 부분으로 수직하게 하위분할되는 옵션을 포함할 수 있다. 이 외에, 지원된 파티션화 패턴들은 또한 현재 블록(40)의 사분의 일을 각각 취하는 네 개의 추가적인 블록들로 현재 블록(40)이 규칙적으로 더 하위분할되는 옵션을 포함할 수 있다. 또한, 파티션화는 화상(20)의 모든 블록들(40)과 관계될 수 있거나 또는 인터 예측 모드와 같은 어떠한 연관된 코딩 모드를 갖는 블록들의 하위세트와 같이 모든 블록들(40) 중에서 단지 적절한 하위세트와 관계될 수 있다. 마찬가지로, 병합 그 자체는 인터 예측 모드에서 코딩된 것과 같은, 일정한 블록들에 대해서만 이용가능할 수 있다는 것을 유념한다. 아래에서 추가적으로 개술된 실시예에 따르면, 앞서언급한 공동으로 해석된 상태는 또한 각각의 블록이 인트라(intra) 예측 모드보다는 인터 예측 모드를 갖는다는 것을 동시에 신호전달한다. 따라서, 블록(40)에 대한 앞서 언급한 플래그의 하나의 상태는 이러한 블록이 더 이상 파티션되지 않는 인터 예측 코딩 블록이라는 것과 이에 대해 병합 및 스킵 모드 모두가 활성화되어 있다는 것을 신호전달할 수 있다. 하지만, 플래그가 다른 상태를 취하는 경우 부수적인 결정으로서, 각각의 파티션 또는 샘플 하위세트(50, 60)는 병합이 각각의 파티션(50, 60)에 적용되어야할지 또는 그렇지 않을지 여부를 신호전달하기 위한 추가적인 플래그를 비트스트림(30) 내에서 개별적으로 수반할 수 있다. 또한, 지원된 파티션화 모드들의 상이한 하위세트들은 예컨대 블록 크기, 블록(40)의 하위분할의 레벨(이 경우 블록은 멀티 트리 하위분할 리프 블록이다), 이들의 조합 또는 이들에 개별적으로 의존하여 블록들(40)에 대해 이용가능해질 수 있다.

[0027]

즉, 그 중에서도 블록(40)을 획득하기 위한 블록들로의 화상(20)의 하위분할은 고정될 수 있거나 또는 비트스트림 내에서 신호전달될 수 있다. 마찬가지로, 현재 블록(40)을 추가적으로 파티션화하기 위해 이용될 파티션화 패턴은 파티션화 정보의 형태로 비트스트림(30) 내에서 신호전달될 수 있다. 따라서, 파티션화 정보는 블록들(40)로의 화상(20)의 하위분할의 확장의 한 종류인 것으로서 간주될 수 있다. 한편, 블록들(40)로의 화상(20)의

하위분할의 원래 입도의 추가적인 관련성이 여전히 남아있을 수 있다. 예를 들어, 인코더(10)는 블록(40)에 의해 정의된 입도로 화상(20)의 블록(40) 또는 각각의 부분을 위해 이용될 코딩 모드를 비트스트림(30) 내에서 신호전달하도록 구성될 수 있으면서, 인코더(10)는 각각의 블록(40)을 위해 선택된 각각의 파티션화 패턴에 의해 정의된 증가된(보다 미세한) 입도로 각각의 블록(40) 내에서 각각의 코딩 모드의 코딩 파라미터들을 변경하도록 구성될 수 있다. 예를 들어, 블록들(40)의 입도로 신호전달된 코딩 모드는 시간간(temporal inter) 예측 모드, 장면간(inter-view) 예측 모드 등과 같은, 인터 예측 모드, 인트라 예측 모드 등 사이에서 구별될 수 있다. 각각의 블록(40)의 파티션화로부터 초래된 하나 이상의 하위블록들(파티션들)과 연관된 코딩 파라미터들의 종류는 각각의 블록(40)에 할당된 코딩 모드에 의존한다. 예를 들어, 인트라 코딩된 블록(40)의 경우, 코딩 파라미터들은 공간 방향을 포함할 수 있는데, 이러한 공간 방향을 따라, 화상(20)의 이전에 디코딩된 부분들의 화상 콘텐츠가 각각의 블록(40)을 채우기 위해 이용된다. 인터 코딩된 블록(40)의 경우, 코딩 파라미터들은, 그 중에서도, 모션 보정된 예측을 위한 모션 벡터를 포함할 수 있다.

[0028]

도 1은 현재 블록(40)이 두 개의 하위블록들(50, 60)로 하위분할된 것을 예시적으로 도시한다. 특히, 수직적 파티션화 모드가 예시적으로 도시된다. 보다 작은 블록들(50, 60)은 또한 하위블록들(50, 60) 또는 파티션들(50, 60) 또는 예측 유닛들(50, 60)이라고 불릴 수 있다. 특히, 인코더(10)는, 지원된 파티션화 패턴들 중 신호전달된 패턴이 두 개 이상의 추가적인 블록들(50, 60)로의 현재 블록(40)의 하위분할을 규정하는 경우들에서, 하위블록들(50, 60)의 코딩 순서로 첫번째 하위블록을 제외한 모든 추가적인 블록들에 대해, 각각의 하위블록을 위한 코딩 파라미터 후보들의 세트로부터, 각각의 하위블록들과 병합될 때 지원된 파티션화 패턴들 중 하나의 패턴을 불러일으킬 임의의 하위블록들과 연관된 코딩 파라미터들과 동일한 코딩 파라미터들을 갖는 코딩 파라미터 후보들을 제거하도록 구성될 수 있다. 보다 정확하게 하기 위해, 지원된 파티션화 패턴들 각각에 대해, 결과적인 하나 이상의 파티션들(50, 60)간에 코딩 순서가 정의된다. 도 1의 경우, 이러한 코딩 순서는 화살표(70)에 의해 예시적으로 나타나며, 이것은 좌측 파티션(50)이 우측 파티션(60) 이전에 코딩되는 것을 정의한다. 수평적 파티션화 모드의 경우에는, 하부 파티션 이전에 상부 파티션이 코딩되는 것이 정의될 수 있다. 어떠한 경우이던 간에, 인코더(10)는, 이러한 병합의 결과, 즉 양측 파티션들(50, 60)이, 실제로 보다 낮은 코딩 레이트로 현재 블록(40)에 대한 비파티션화 모드를 선택함으로써 똑같이 산출될 수 있는, 자신과 연관된 동일한 코딩 파라미터들을 가질 것이라는 사실을 회피하기 위해, 코딩 순서(70)로 두번째의 파티션(60)에 대해, 각각의 두번째 파티션(60)에 대한 코딩 파라미터 후보들의 세트로부터, 첫번째 파티션(50)과 연관된 코딩 파라미터들과 동일한 코딩 파라미터들을 갖는 코딩 파라미터 후보들을 제거하도록 구성된다.

[0029]

보다 정확히 하기 위해, 인코더(10)는 블록 파티션화와 더불어 효율적인 방식으로 블록 병합을 이용하도록 구성될 수 있다. 블록 병합과 관련되는 한, 인코더(10)는 각각의 파티션(50, 60)에 대해, 각각의 코딩 파라미터 후보들의 세트를 결정할 수 있다. 인코더는 이전에 디코딩된 블록들과 연관된 코딩 파라미터들에 기초하여 파티션들(50, 60) 각각에 대한 코딩 파라미터 후보들의 세트들을 결정하도록 구성될 수 있다. 특히, 코딩 파라미터 후보들의 세트들 내의 적어도 몇몇의 코딩 파라미터 후보들은 이전에 디코딩된 파티션들의 코딩 파라미터들과 동등할 수 있다(즉, 이전에 디코딩된 파티션들의 코딩 파라미터들로부터 채용될 수 있다). 추가적으로 또는 대안적으로, 적어도 몇몇의 코딩 파라미터 후보들은 중간값, 평균 등과 같은 적절한 조합에 의해, 하나 이상의 이전에 코딩된 파티션과 연관된 코딩 파라미터 후보들로부터 유도될 수 있다. 하지만, 인코더(10)는 코딩 파라미터 후보들의 감소된 세트의 결정을 수행하며, 제거 후 이러한 코딩 파라미터 후보가 하나보다 많이 남아있는 경우, 제거되지 않고 남아있는 코딩 파라미터 후보들 중의 선택을 수행하도록 구성될 수 있기 때문에, 첫번째 파티션 이외의 다른 파티션들(60) 각각에 대해, 제거되지 않은 코딩 파라미터 후보 또는 선택된 코딩 파라미터 후보에 의존하여 각각의 파티션과 연관된 코딩 파라미터들을 설정하기 위해, 인코더(10)는, 파티션들(50, 60)의 리유닛화를 사실상 야기시킬 코딩 파라미터 후보들이 제거되도록 이러한 제거를 수행하도록 구성된다. 즉, 단순히 파티션화 정보만의 이용에 의해 이러한 파티션화를 곧바로 신호전달하는 경우보다 더 복잡하게 실제의 파티션화 상황이 코딩되는 구문 정상도들은 효과적으로 회피가 된다.

[0030]

게다가, 코딩 파라미터 후보들의 세트가 작아짐에 따라, 병합 정보를 비트스트림(30) 내로 인코딩하는데 필요한 보조 정보의 양은 이러한 후보 세트들에서의 보다 작은 수의 엘리먼트들로 인해 감소될 수 있다. 특히, 디코더는 도 1의 인코더가 행하는 것과 동일한 방식으로 코딩 파라미터 후보들의 세트들을 결정하고 이어서 이들을 감소시킬 수 있기 때문에, 도 1의 인코더(10)는 예컨대, 제거되지 않은 코딩 파라미터 후보들 중 어느 것이 병합을 위해 활용될 것인지를 규정하는 구문 엘리먼트를 비트스트림(30) 내로 삽입하기 위해 보다 작은 비트들을 이용함으로써 감소된 코딩 파라미터 후보들의 세트들을 활용할 수 있다. 당연히, 각각의 파티션을 위한 제거되지 않은 코딩 파라미터 후보들의 갯수가 단하나인 경우에는 비트스트림(30) 내로의 구문 엘리먼트의 도입은 완전히 억제될 수 있다. 어떠한 경우이던 간에, 병합으로 인해, 즉 각각의 파티션과 연관된 코딩 파라미터들을 제거되지

많은 코딩 파라미터 후보들 중의 선택된 후보 또는 남아있는 후보에 의존하여 설정함으로써 인해, 인코더(10)는 비트스트림(30)으로의 각각의 파티션을 위한 코딩 파라미터들의 완전히 새로운 삽입을 억제하여, 보조 정보도 감소시킬 수 있다. 본 출원의 몇몇의 실시예들에 따르면, 인코더(10)는 각각의 파티션들을 위한 코딩 파라미터 후보들 중의 남아있는 코딩 파라미터 후보, 또는 선택된 코딩 파라미터 후보를 정제시키기 위한 정제 정보를 비트스트림(30) 내에서 신호전달하도록 구성될 수 있다.

[0031]

병합 후보 리스트를 감소시키는 방법 언급한 가능성에 따라, 인코더(10)는 병합 후보들의 코딩 파라미터들과 파티션의 코딩 파라미터들의 비교에 의해 제거될 병합 후보들을 결정하도록 구성될 수 있으며, 이러한 병합은 또 다른 지원된 파티션화 패턴을 산출시킬 것이다. 코딩 파라미터 후보들을 처리하는 이러한 방식은 예컨대, 좌측 파티션(50)의 코딩 파라미터들이 우측 파티션(60)을 위한 코딩 파라미터 후보들의 세트의 하나의 엘리먼트를 형성하는 경우, 도 1의 예시적인 경우에서 적어도 하나의 코딩 파라미터 후보를 효과적으로 제거시킬 것이다. 하지만, 추가적인 코딩 파라미터 후보들은 또한 좌측 파티션(50)의 코딩 파라미터들과 동등한 경우에도 제거될 수 있다. 하지만, 본 발명의 또다른 실시예에 따르면, 인코더(10)는 코딩 순서로 두번째의 파티션 및 후속 파티션 각각에 대한 후보 블록들의 세트를 결정하는 것과 함께, 각각의 파티션과 병합될 때, 지원된 파티션화 패턴들 중 하나의 패턴을 초래시키는, 이러한 후보 블록들의 세트로부터 그 후보 블록들을 제거하도록 구성될 수 있다. 몇몇의 의미에서, 이것은 다음을 의미한다. 인코더(10)는 후보 세트의 각 엘리먼트가 현재 블록(40) 또는 이전에 코딩된 임의의 블록들(40)의 연관된 단하나의 파티션을 갖고 병합 후보가 연관된 파티션의 각각의 코딩 파라미터들을 채택하도록, 각각의 파티션(50 또는 60)(즉 코딩 순서로 첫번째 파티션 및 후속 파티션)을 위한 병합 후보들을 결정하도록 구성될 수 있다. 예를 들어, 후보 세트의 각 엘리먼트는 이전에 코딩된 파티션들의 이러한 코딩 파라미터들 중의 하나와 동등해질 수 있고, 즉 이로부터 채용될 수 있거나, 또는 추가적으로 보내진 정제 정보를 이용한 정제 또는 추가적인 스케일링에 의한 것과 같이 단하나의 이러한 이전에 코딩된 파티션의 코딩 파라미터들로부터 적어도 유도될 수 있다. 하지만, 인코더(10)는 또한 이러한 후보 세트를 추가적인 엘리먼트들 또는 후보들, 즉 이전에 코딩된 하나보다 많은 파티션의 코딩 파라미터들의 결합으로부터 유도되었거나 또는 하나의 모션 파라미터 리스트의 코딩 파라미터들만을 취함으로써 이전에 코딩된 하나의 파티션의 코딩 파라미터들로부터 수정을 통해 유도되었던 코딩 파라미터 후보들과 함께 수반하도록 구성될 수 있다. "결합된" 엘리먼트들에서는, 각각의 후보 엘리먼트와 각각의 파티션의 코딩 파라미터들간에는 1:1 연관이 존재하지 않는다. 도 1의 설명의 제1 대안구성에 따르면, 인코더(10)는 전체 후보 세트로부터 모든 후보들을 제거하도록 구성될 수 있으며, 이 경우 후보 세트의 코딩 파라미터들은 파티션(50)의 코딩 파라미터들과 동일하다. 도 1의 설명 중의 후자의 대안구성에 따르면, 인코더(10)는 파티션(50)과 연관된 후보 세트의 엘리먼트만을 제거하도록 구성될 수 있다. 양쪽의 관점들을 조화시켜서, 인코더(10)는 이러한 제거(및 동등한 코딩 파라미터들을 갖는 후보들에 대한 검색)를 결합에 의해 획득되는 코딩 파라미터들을 갖는 후보 세트의 남아있는 부분까지 확장시키지 않고, 몇몇의 (예컨대, 이웃하는) 이전에 코딩된 파티션들에 대한 1:1 연관을 보여주는 후보들을 후보 세트의 부분으로부터 제거하도록 구성될 수 있다. 하지만, 물론, 하나의 결합이 중복된 표현을 야기시킬 것이라면, 이것은 결합된 후보들에 대한 리던던시 체크도 수행함으로써 또는 리스트로부터 중복 코딩 파라미터들을 제거함으로써 해결될 수 있다.

[0032]

도 1의 방법 개술한 실시예에 맞는 디코더의 실시예를 설명하기에 앞서, 보다 상세한 구현에서의 도 1에 따른 인코딩을 위한 장치, 즉 인코더를 도 2를 참조하여 아래에서 보다 자세하게 개술한다. 도 2는 화상(20)을 블록들(40)로 하위분할하도록 구성된 하위분할기(72), 위에서 개술한 하나 이상의 샘플 세트들의 그룹들로 블록(40)을 병합하도록 구성된 병합기(74), 샘플 세트들의 그룹들의 유닛들에서 화상(20)에 걸쳐 변하는 코딩 파라미터들을 이용하여 화상(20)을 인코딩하도록 구성된 인코더 또는 인코딩 스테이지(76), 및 스트림 생성기(78)를 포함하는 인코더를 도시한다. 인코더(76)는 화상(20)을 예측하고 미리결정된 블록들에 대한 예측 잔차(residual)를 인코딩함으로써 화상(20)을 인코딩하도록 구성된다. 즉, 인코더(76)는 상술한 바와 같이, 모든 블록들(40)에 대한 예측 잔차를 인코딩하는 것만은 아니다. 이보다는, 이들 중 몇몇은 활성화된 스킵 모드를 갖는다. 스트림 생성기(78)는 각각의 블록(40)이 또다른 블록과 더불어 그룹들 중 하나의 그룹으로 병합되는지 아닌지의 여부 및 각각의 블록이 스킵 모드를 이용하거나 아닌지의 여부를 신호전달하는, 적어도 블록들(40)의 하위 세트 각각에 대한 하나 이상의 구문 엘리먼트들과 함께, 예측 잔차 및 코딩 파라미터들을 비트스트림(30) 내로 삽입하도록 구성된다. 상술한 바와 같이, 하위분할기(72)의 하위분할에 기초를 이루는 하위분할 정보는 또한 스트림 생성기(78)에 의해 화상(20)에 대한 비트스트림(30) 내로 인코딩될 수 있다. 이것은 도 2에서 점선으로 표시된다. 병합기(74)에 의한 병합 결정 및 인코더(76)에 의해 수행되는 스킵 모드 결정은, 위에서 설명한 바와 같이, 현재 블록(40)의 하나 이상의 구문 엘리먼트들의 잠재적인 상태들 중 하나의 상태가 각각의 블록이 화상(20)의 또다른 블록과 함께 블록들의 그룹들 중 하나의 그룹으로 병합될 것이며 비트스트림(30) 내로 인코딩되

고 삽입되는 예측 잔차를 갖지 않는다는 것을 신호전달하도록, 스트림 생성기(78)에 의해 비트스트림(30) 내로 공동으로 인코딩된다. 스트림 생성기(78)는 예컨대, 삽입을 수행하기 위해 엔트로피 코딩을 이용할 수 있다. 하위분할기(72)는 블록들(40)로의 화상(20)의 하위분할뿐만 아니라 각각의 파티션들(50, 60)로의 택일적인 추가적 파티션화를 담당할 수 있다. 병합기(74)는 위에서 개술한 병합 결정을 담당하는 반면에 인코더(76)는 예컨대 블록들(40)에 대한 스킵 모드에 대해 결정할 수 있다. 당연히, 이러한 결정들 모두는 레이트/왜곡 수치에 조합 형태로 영향을 미치고, 이에 따라 장치(10)는 어느 옵션이 바람직할 수 있는지를 확인하기 위해 여러 개의 결정 옵션들을 시도하도록 구성될 수 있다.

[0033]

도 1 및 도 2와 관련하여 본 발명의 실시예에 따른 인코더를 설명한 후, 실시예에 따른 디코딩을 위한 장치, 즉 디코더(80)를 도 3을 참조하여 설명한다. 도 3의 디코더(80)는 상술한 바와 같이, 화상(20)이 인코딩되어 있는 비트스트림(30)을 디코딩하도록 구성된다. 특히, 디코더(80)는, 현재 샘플 세트 또는 블록(40)에 대해, 현재 블록(40)과 연관된 코딩 파라미터들이 병합 후보에 따라 설정될 것인지 또는 비트스트림(30)으로부터 검색될 것인지 여부에 대한 제1 결정과, 화상(20)의 현재 블록(40)이, 어떠한 잔차 데이터도 없이, 현재 블록(40)과 연관된 코딩 파라미터들에 의존하는 예측 신호에만 기초하여 재구성될 것인지, 또는 비트 스트림(30)내의 잔차 데이터에 의해 현재 블록(40)과 연관된 코딩 파라미터들에 의존하여 예측 신호를 정제시킴으로써 재구성될 것인지 여부에 대한 제2 결정에 관해, 비트스트림(30) 내의 앞서 언급한 플래그에 공동으로 응답하도록(commonly responsive) 구성될 수 있다.

[0034]

즉, 디코더의 기능성은 도 1 및 도 2와 관련하여 설명된 인코더의 기능성과 대체로 일치한다. 예를 들어, 디코더(80)는 블록들(40)로의 화상(20)의 하위분할을 수행하도록 구성될 수 있다. 이러한 하위분할은 디플트에 의해 디코더(80)에게 알려질 수 있거나, 또는 디코더(80)는 비트스트림(30)으로부터 각각의 하위분할 정보를 추출하도록 구성될 수 있다. 블록(40)이 병합될 때 마다, 디코더(80)는 병합 후보에 따라 블록의 코딩 파라미터들을 설정함으로써 블록(40)과 연관된 코딩 파라미터들을 획득하도록 구성될 수 있다. 병합 후보를 결정하기 위해, 디코더(80)는 인코더가 수행했던 것과 정확히 동일한 방식으로 병합 후보들의 리스트 또는 세트의 위에서 개술한 결정을 수행할 수 있다. 이것은 심지어, 본 출원의 몇몇의 실시예들에 따라, 한편으로 블록 파티션화와 다른 한편으로 블록 병합간의 위에서 개술된 리던던시를 회피하기 위해 병합 후보들의 예비 세트/리스트의 감소를 포함한다. 병합 후보들의 결정된 세트 또는 리스트 중의 선택은 병합이 활성화될 때 마다 비트스트림(30)으로부터 각각의 병합 인덱스를 추출함으로써 디코더(80)에 의해 수행될 수 있다. 병합 인덱스는 상술한 바와 같이 결정된 병합 후보들의 (감소된) 세트 또는 리스트 중에서 이용될 병합 후보를 가리킨다. 게다가, 또한 상술한 바와 같이, 디코더(80)는 또한 지원된 파티션화 패턴들 중의 하나의 패턴에 따른 파티션화를 블록들(40)에 대해 수행하도록 구성될 수 있다. 당연히, 이러한 파티션화 패턴들 중 하나는 블록(40)이 더 이상 파티션화되지 않는 비 파티션화 모드를 수반할 수 있다. 완전히 설명한 플래그가 어떠한 블록(40)에 대한 병합 및 스킵 모드의 활성화를 표시하는 공동으로 정의된 상태를 취하는 경우, 디코더(80)는 임의의 잔차 신호와의 결합보다는 예측 신호에만 기초하여 현재 블록(40)을 재구성하도록 구성될 수 있다. 달리 말하면, 디코더(80)는 이 경우 현재 블록(40)에 대한 잔차 데이터 추출을 억제시키며 단지 현재 블록의 코딩 파라미터들로부터 유도된 예측 신호의 이용에 의해 현재 블록(40) 내에서 화상(20)을 재구성한다. 이미 또한 상술한 바와 같이, 디코더(80)는 또한 현재 블록(40)에 대한 신호전달로서의 플래그의 공동 상태를, 이러한 블록이 더 이상 파티션화되지 않은 블록 및/또는 인터 예측 블록이라고 해석할 수 있다. 즉, 디코더(80)는 병합 후보에 따라 이러한 코딩 파라미터들을 설정함으로써 현재 블록(40)과 연관된 코딩 파라미터들을 획득하고, 비트스트림(30) 내의 현재 블록(40)의 해당 플래그가 현재 블록(40)과 연관된 코딩 파라미터들이 병합을 이용하여 설정될 것이라는 것을 신호전달하는 경우 어떠한 잔차 데이터도 없이 현재 블록(40)의 코딩 파라미터들에 의존하는 예측 신호만에 기초하여 화상(20)의 현재 블록(40)을 재구성하도록 구성될 수 있다. 하지만, 현재 블록(40)이 병합을 거치지 않거나 또는 스킵 모드가 이용되지 않는다는 것을 해당 플래그가 신호전달하는 경우, 디코더(80)는 비트스트림(30) 내의 또다른 플래그에 응답할 수 있으며, 이로써 디코더(80)는, 이러한 다른 플래그에 의존하여, 각각의 병합 후보에 따라 코딩 파라미터들을 설정함으로써 현재 블록과 연관된 코딩 파라미터들을 획득하고, 비트스트림(30)으로부터 현재 블록에 대한 잔차 데이터를 획득하고 예측 신호와 잔차 데이터에 기초하여 화상(20)의 현재 블록(40)을 재구성하거나, 또는 비트스트림(30)으로부터 현재 블록(40)과 연관된 코딩 파라미터들을 추출하고, 비트스트림(30)으로부터 현재 블록(40)에 대한 잔차 데이터를 획득하며 예측 신호와 잔차 데이터에 기초하여 화상(20)의 현재 블록(40)을 재구성한다. 상술한 바와 같이, 디코더(80)는, 제1 플래그가 병합 및 스킵 모드의 활성화를 동시에 신호전달하는 공동의 신호전달 상태를 취하지 않는 경우에만, 비트스트림(30) 내의 다른 플래그의 존재를 예상하도록 구성될 수 있다. 그 후에만, 디코더(80)는 스킵 모드없이 병합이 발생할지 여부를 확인하기 위해 비트스트림으로부터 다른 플래그를 추출한다. 당연히, 디코더(80)는 대안적으로, 제2 플래그가 병합의 비활성화를 신호전달하

는 경우 현재 블록(40)에 대한 비트스트림(30) 내의 또다른 제3 플래그를 기다리도록 구성될 수 있으며, 이 제3 플래그는 스킵 모드 활성화 또는 비활성화를 신호전달한다.

[0035]

도 2와 유사하게, 도 4는 도 3의 디코딩을 위한 장치의 잠재적인 구현을 도시한다. 따라서, 도 4는 디코딩을 위한 장치, 즉, 비트스트림(30) 내로 인코딩된 화상(20)을 블록들(40)로 하위분할하도록 구성된 하위분할기(82), 하나 이상의 블록들 각각의 그룹들로 블록들(40)을 병합하도록 구성된 병합기(84), 샘플 세트들의 그룹들의 유닛들에서 화상(20)에 걸쳐 변하는 코딩 파라미터들을 이용하여 화상(20)을 디코딩하거나 재구성하도록 구성된 디코더(86), 및 추출기(88)를 포함하는 디코더(80)를 도시한다. 디코더(86)는 또한 스킵 모드를 스위치 오프시킨 미리결정된 블록들(40)에 대해, 화상(20)을 예측하고, 미리결정된 블록들(40)에 대한 예측 잔차를 디코딩하며 화상(20)을 예측한 것으로부터 초래된 예측치와 예측 잔차를 결합시킴으로써 화상(20)을 디코딩하도록 구성된다. 추출기(88)는 비트스트림(30)으로부터, 각각의 블록(40)이 또다른 블록(40)과 함께 그룹들 중 하나의 그룹으로 병합될 것인지 아닌지의 여부를 신호전달하는, 블록들(40)의 적어도 하위세트 각각에 대한 하나 이상의 구문 엘리먼트들과 더불어, 코딩 파라미터들과 예측 잔차를 추출하도록 구성되며, 병합기(84)는 하나 이상의 구문 엘리먼트들에 응답하여 병합을 수행하도록 구성되며, 하나 이상의 구문 엘리먼트들의 잠재적인 상태들 중 하나의 상태는 각각의 블록(40)이 또다른 블록(40)과 함께 블록들의 그룹들 중 하나의 그룹으로 병합될 것이며 비트스트림(30) 내로 인코딩되고 삽입되는 예측 잔차를 갖지 않는다는 것을 신호전달한다.

[0036]

따라서, 도 4를 도 2와 비교하면, 하위분할기(82)는 하위분할기(72)에 의해 생성된 하위분할을 저장하기 위해 하위분할기(72) 처럼 동작한다. 하위분할기(82)는 디폴트에 의해 화상(20)의 하위분할에 대해 알거나 또는 추출기(88)를 통해 비트스트림(30)으로부터 하위분할 정보를 추출한다. 마찬가지로, 병합기(84)는 블록들(40)의 병합을 수행하고, 비트스트림(30) 내의 상술한 신호전달을 통해 블록들(40) 및 블록 부분들에 대해 활성화된다. 디코더(86)는 비트스트림(30) 내의 코딩 파라미터들을 이용하여 화상(20)의 예측 신호의 생성을 수행한다. 병합의 경우, 디코더(86)는 현재 블록(40) 또는 현재 블록 파티션의 코딩 파라미터들을 이웃 블록들/파티션들로부터 복사하거나 또는 그렇지 않고 병합 후보에 따라 상기 코딩 파라미터들을 설정한다.

[0037]

이미 위에서 설명한 바와 같이, 추출기(88)는 현재 블록에 대한 플래그 또는 구문 엘리먼트의 잠재적인 상태들 중 하나의 상태를 병합 및 스킵 모드의 활성화를 동시에 신호전달하는 신호로서 해석하도록 구성된다. 이와 동시에, 추출기(88)는 상기 상태를 현재 블록(40)에 대한 지원된 파티션화 패턴들 중 미리결정된 패턴을 신호전달하는 것이라고 해석할 수 있다. 예를 들어, 미리결정된 파티션화 패턴은 블록(40)이 파티션화되지 않은 상태로 남아 있도록 하고 이에 따라 파티션 그 자체를 형성케하는 비파티션화 모드일 수 있다. 따라서, 추출기(88)는 각각의 플래그 또는 구문 엘리먼트가 동시적인 신호전달 상태를 취하지 않는 경우에만 블록(40)의 파티션화를 신호전달하는 파티션화 정보를 비트스트림(30)이 포함할 것으로 예상한다. 아래에서 보다 자세하게 설명할 바와 같이, 파티션화 정보는 비트스트림(30) 내에서 구문 엘리먼트를 통해 운송될 수 있으며, 이와 동시에 현재 블록(40)의 코딩 모드를 제어하는데, 즉 블록들(40)을 인터 코딩된 것들과 인트라 코딩된 것들로 분할한다. 이 경우, 제1 플래그/구문 엘리먼트의 공동의 신호전달 상태는 또한 인터 예측 코딩 모드의 신호전달로서 해석될 수 있다. 신호전달된 파티션화 정보로부터 초래된 파티션들 각각에 대해, 추출기(88)는 블록(40)을 위한 제1 플래그/구문 엘리먼트가 병합 및 스킵 모드의 활성화를 동시에 신호전달하는 공동의 신호전달 상태를 취하지 않는 경우 비트스트림(30)으로부터 또다른 병합 플래그를 추출할 수 있다. 이 경우, 스킵 모드는 추출기(88)에 의해 스위치 오프될 것으로 불가피하게 해석될 수 있지만, 병합은 파티션들에 대해 개별적으로 비트스트림(30)에 의해 활성화될 수 있으며, 잔차 신호는 이러한 현재 블록(40)에 대해 비트스트림(30)으로부터 추출된다.

[0038]

따라서, 도 3 또는 도 4의 디코더(80)는 비트스트림(30)을 디코딩하도록 구성된다. 상술한 바와 같이, 비트스트림(30)은 화상(20)의 현재 블록(40)에 대한 지원된 파티션화 패턴들 중 하나를 신호전달할 수 있다. 디코더(80)는, 지원된 파티션화 패턴들 중 신호전달된 패턴이 두 개 이상의 파티션들(50, 60)로의 현재 블록(40)의 하위분할을 규정하는 경우, 코딩 순서(70)로 파티션들의 첫번째 파티션(50)을 제외한 모든 파티션들, 즉 도 1과 도 3의 도시된 예시에서 파티션(60)을, 각각의 파티션과 병합될 때 지원된 파티션화 패턴들 중 하나, 즉 비트스트림(30) 내에서 신호전달되지 않았음에도 불구하고 지원된 파티션화 패턴들 중 하나의 패턴인 패턴을 초래시킬 임의의 파티션들과 연관된 코딩 파라미터들과 동일하거나 또는 동등한 코딩 파라미터들을 갖는 각각의 파티션 코딩 파라미터 후보들에 대한 코딩 파라미터 후보들의 세트로부터 제거하도록 구성될 수 있다.

[0039]

예를 들어, 디코더(80)는, 제거되지 않은 코딩 파라미터 후보들의 수가 제로가 아닌 경우, 각각의 파티션(60)과 연관된 코딩 파라미터들을 제거되지 않은 파라미터 후보들 중 하나에 의존하여 설정하도록 구성될 수 있다. 예를 들어, 디코더(80)는 추가적인 정제를 갖거나 또는 추가적인 정제 없이 및/또는 코딩 파라미터들이 각각 가리키는 시간적 거리에 따른 스케일링을 갖거나 또는 그러한 스케일링 없이 파티션(60)의 코딩 파라미터들을 제거

되지 않은 코딩 파라미터 후보 중 하나와 동등해지도록 설정한다. 예를 들어, 제거되지 않은 후보들과 병합될 코딩 파라미터 후보는 파티션(60)을 위한 비트스트림(30) 내에서 명시적으로 신호전달된 참조 화상 인덱스보다는 연관된 또다른 참조 화상 인덱스를 가질 수 있다. 이 경우, 코딩 파라미터 후보들의 코딩 파라미터들은 각각의 참조 화상 인덱스와 각각 관련된 모션 벡터들을 정의할 수 있고, 디코더(80)는 참조 화상 인덱스들간의 비율에 따라 최종적으로 선택된 제거되지 않은 코딩 파라미터 후보의 모션 벡터를 스케일링하도록 구성될 수 있다. 따라서, 방금 언급한 대안구성에 따르면, 병합을 거칠 코딩 파라미터들은 모션 파라미터들을 망라할 것인 반면에, 참조 화상 인덱스들은 이로부터 구분될 것이다. 하지만, 위에서 나타낸 바와 같이, 대안적인 실시예들에 따르면, 참조 화상 인덱스들은 또한 병합을 거치게 될 코딩 파라미터들의 일부일 수 있다.

[0040] 병합 동작은 인터 예측 블록들(40)에 제한적일 수 있다는 것은 도 1과 도 2의 인코더와 도 3과 도 4의 디코더에 똑같이 적용된다. 따라서, 디코더(80)와 인코더(10)는 현재 블록(40)에 대한 인트라 및 인터 예측 모드들을 지원하며, 현재 블록(40)이 인터 예측 모드로 코딩되는 경우에만 병합을 수행하도록 구성될 수 있다. 따라서, 인터 예측된 이전에 코딩된 파티션들의 코딩/예측 파라미터들만이 후보 리스트를 결정/구축하는데 이용될 수 있다.

[0041] 위에서 이미 논의한 바와 같이, 코딩 파라미터들은 예측 파라미터들일 수 있고, 디코더(80)는 각각의 파티션을 위한 예측 신호를 유도해내기 위해 파티션들(50, 60)의 예측 파라미터들을 이용하도록 구성될 수 있다. 당연히, 인코더(10)도 이와 동일한 방식으로 예측 신호의 유도를 수행한다. 하지만, 인코더(10)는 적절한 최적화 관점에서 일부 최적화를 달성하기 위해 비트스트림(30) 내의 다른 모든 구문 엘리먼트들과 더불어 예측 파라미터들을 추가적으로 설정한다.

[0042] 또한, 위에서 이미 설명한 바와 같이, 인코더는 각각의 파티션을 위한 (제거되지 않은) 코딩 파라미터 후보의 갯수가 하나보다 큰 경우에만 인덱스를 (제거되지 않은) 코딩 파라미터 후보에 삽입하도록 구성될 수 있다. 따라서, 디코더(80)는, (제거되지 않은) 코딩 파라미터 후보들, 예컨대 파티션(60)의 갯수에 의존하여, (제거되지 않은) 코딩 파라미터 후보들의 갯수가 하나보다 큰 경우, (제거되지 않은) 코딩 파라미터 후보 중 어느 것이 병합을 위해 활용되는지를 명시하는 구문 엘리먼트를 비트스트림(30)이 포함할 것으로 단지 예상하도록 구성될 수 있다. 하지만, 후보 세트가 둘보다 작은 갯수로 작아지는 경우는 상술한 바와 같이, 결합된 코딩 파라미터들, 즉 하나보다 많거나 또는 두 개보다 많은 이전에 코딩된 파티션들의 코딩 파라미터들의 결합에 의해 유도되었던 파라미터들을 이용하여 후보들의 리스트/세트를 확장시키고, 후보 세트 감소의 성능을 정확히 이전에 코딩된 하나의 파티션의 코딩 파라미터들로부터의 유도 또는 채움에 의해 획득되었던 후보들로 제한시킴으로써 발생으로부터 일반적으로 배제될 수 있다. 반대의 경우, 즉 일반적으로 또다른 지원된 파티션화 패터를 조래하는 파티션과 동일한 값을 갖는 모든 코딩 파라미터 후보들을 제거하는 것도 가능하다.

[0043] 이러한 결정과 관련하여, 디코더(80)는 인코더(10)가 동작하는 것처럼 동작한다. 즉, 디코더(80)는 이전에 디코딩된 파티션들과 연관된 코딩 파라미터들에 기초하여 블록(40)의 파티션 또는 파티션들을 위한 병합 후보들의 세트를 결정하도록 구성될 수 있다. 즉, 코딩 순서는 각각의 블록(40)의 파티션들(50, 60) 중에서 정의되는 것뿐만이 아니라, 화상(20) 자체의 블록들(40) 중에서도 정의될 수 있다. 따라서, 파티션(60) 이전에 코딩되었던 모든 파티션들은 도 3의 경우에서의 파티션(60)과 같은, 임의의 후속 파티션들을 위한 병합 후보들의 세트의 결정에 대한 기초로서 역할을 할 수 있다. 위에서 또한 설명된 바와 같이, 인코더와 디코더는 병합 후보들의 세트의 결정을 일정한 공간적 및/또는 시간적 이웃의 파티션들로 제한시킬 수 있다. 예를 들어, 디코더(80)는 현재 파티션에 이웃해 있는 이전에 디코딩된 파티션들과 연관된 코딩 파라미터들에 기초하여 병합 후보들의 세트를 결정하도록 구성될 수 있고, 이러한 파티션들은 현재 블록(40)의 안과 밖에 놓여있을 수 있다. 당연히, 병합 후보들의 결정은 또한 코딩 순서로 첫번째의 파티션에 대해 수행될 수 있다. 단지 제거만이 생략될 수 있다.

[0044] 도 1의 설명에 따르면, 디코더(80)는 인트라 예측 모드에서 코딩된 것들을 제외하고, 이전에 디코딩된 파티션들의 초기 세트 중에서 첫번째 파티션 이외의 각각의 파티션(60)을 위한 코딩 파라미터 후보들의 세트를 결정하도록 구성될 수 있다.

[0045] 또한, 인코더가 화상(20)을 블록들(40)로 하위분할하기 위해 하위분할 정보를 비트스트림 내에 도입시키는 경우, 디코더(80)는 비트스트림(30)에서의 하위분할 정보에 따라 이러한 코딩 블록들(40)로의 화상(20)의 하위분할을 복구하도록 구성될 수 있다.

[0046] 도 1 내지 도 4를 참조하면, 현재 블록(40)에 대한 잔차 신호는 코딩 파라미터들과 관련하여 파티션들에 의해 정의된 입도와는 다를 수 있는 입도로 비트스트림(30)을 통해 송신될 수 있어야 한다는 것을 유념해야 한다. 예를 들어, 스킵 모드가 비활성화된 블록들의 경우, 도 1의 인코더(10)는, 파티션들(50, 60)로의 파티션과 병행해

서, 또는 이와는 독립적으로, 블록(40)을 하나 이상의 변환 블록들로 하위분할하도록 구성될 수 있다. 인코더는 추가적인 하위분할 정보를 통해 블록(40)에 대한 각각의 변환 블록 하위분할을 신호전달할 수 있다. 이어서 디코더(80)는 비트스트림에서의 추가적인 하위분할 정보에 따라 하나 이상의 변환 블록들로의 블록(40)의 이러한 추가적인 하위분할을 복구시키고, 이러한 변환 블록들의 유닛들에서의 비트스트림으로부터 현재 블록(40)의 잔차 신호를 유도하도록 구성될 수 있다. 변환 블록 파티션화의 중요성은 인코더에서의 DCT와 같은 변환과, 디코더에서의 IDCT와 같은 대응하는 역변환이 블록(40)의 각각의 변환 블록 내에서 개별적으로 수행된다는 것일 수 있다. 그 후 블록(40) 내의 화상(20)을 재구성하기 위해, 인코더(10)는 각각의 파티션들(50, 60)에서 코딩 파라미터들을 적용함으로써 유도된 예측 신호와, 잔차 신호를 각각 덧셈과 같이 결합시킨다. 하지만, 잔차 코딩은 임의의 변환 및 역변환을 각각 수반하지 않을 수 있고, 예측 레지듀م(residuum)은 이 대신에 예측대 공간 도메인에서 코딩된다는 것을 유념해야 한다.

[0047]

추가적인 실시예들의 추가적인 잠재적 상세사항을 아래에서 설명하기 전에, 도 1 내지 도 4의 인코더와 디코더의 잠재적인 내부 구조를 도 5와 도 6을 참조하여 설명할 것이지만, 하이브리드 코딩 특성에 초점을 맞추기 위해 본 도면들에서는 병합기와 하위분할기는 도시되지 않는다. 도 5는 어떻게 인코더(10)가 내부적으로 구축될 수 있는지를 예시적으로 도시한다. 도시된 바와 같이, 인코더(10)는 감산기(108), 변환기(100), 및 비트스트림 생성기(102)를 포함할 수 있으며, 비트스트림 생성기(102)는, 도 5에서 표시된 바와 같이, 엔트로피 코딩을 수행할 수 있다. 엘리먼트들(108, 100, 102)은 화상(20)을 수신하는 입력부(112)와, 앞서 언급한 비트스트림(30)을 출력하는 출력부(114) 사이에 직렬로 연결된다. 특히, 감산기(108)는 입력부(112)에 연결된 비반전 입력부를 가지며, 변환기(100)는 감산기(108)의 출력부와 비트스트림 생성기(102)의 제1 입력부 사이에 연결되어 있으며, 비트스트림 생성기(102)는 이어서 출력부(114)에 연결된 출력부를 갖는다. 도 5의 인코더(10)는 역변환기(104)와 가산기(110)를 더 포함하며, 역변환기(104)와 가산기(110)는, 이러한 순서로, 변환기(100)의 출력부에 직렬로 연결되어 있다. 인코더(10)는 가산기(110)의 출력부와 가산기(110)의 추가적인 입력부와 감산기(108)의 반전 입력부 사이에 연결된 예측기(106)를 더 포함한다.

[0048]

도 5의 엘리먼트들은 다음과 같이 상호작용한다: 예측기(106)는 화상(20)의 부분들을 예측하고, 이러한 예측의 결과, 즉 예측 신호는 감산기(108)의 반전 입력에 인가된다. 이어서 감산기(108)의 출력은 화상(20)의 각각의 부분과 예측 신호간의 차이, 즉 잔차 신호를 나타낸다. 잔차 신호는 변환기(100)에서 변환 코딩처리를 거친다. 즉, 변환기(100)는 변환 계수 레벨들을 획득하기 위해, DCT 등과 같은 변환, 및 변환된 잔차 신호, 즉 변환 계수에 대한 후속 양자화를 수행할 수 있다. 역변환기(104)는 변환기(100)에 의해 출력된 최종적인 잔차 신호를 재구성하여, 변환기(100)에서의 양자화로 인한 정보 손실을 제외하고 변환기(100)로 입력된 잔차 신호에 대응하는 재구성된 잔차 신호를 획득한다. 예측기(106)에 의해 출력된 예측 신호와 재구성된 잔차 신호의 합산은 화상(20)의 각각의 부분의 재구축을 초래시키고, 이것은 가산기(110)의 출력부로부터 예측기(106)의 입력부로 포워딩된다. 예측기(106)는 인트라 예측 모드, 인터 예측 모드 등과 같은, 상술한 바와 같은 상이한 모드들에서 동작한다. 예측 신호를 획득하기 위해 예측기(106)에 의해 적용된 예측 모드 및 대응하는 코딩 또는 예측 파라미터들은 비트스트림 내로의 삽입을 위해 예측기(106)에 의해 엔트로피 인코더(102)에 포워딩된다.

[0049]

인코더에 대한 도 5에서 도시된 가능성에 대응하는, 도 3과 도 4의 디코더(80)의 내부 구조의 잠재적인 구현이 도 6에서 도시된다. 여기서 도시된 바와 같이, 디코더(80)는, 도 6에서 도시된 바와 같이 엔트로피 디코더로서 구현될 수 있는 비트스트림 추출기(150), 역 변환기(152)와 가산기(154)를 포함할 수 있으며, 역 변환기(152)와 가산기(154)는 이 순서대로 디코더의 출력부(160)와 입력부(158) 사이에 연결된다. 더 나아가, 도 6의 디코더는 가산기(154)의 출력부와 가산기(154)의 추가적인 입력부 사이에 연결된 예측기(156)를 포함한다. 엔트로피 디코더(150)는 예측기(156)의 파라미터 입력부에 연결된다.

[0050]

도 6의 디코더의 기능을 간략하게 설명하면, 엔트로피 디코더(150)는 비트스트림(30) 내에 포함된 모든 정보를 추출하기 위한 것이다. 이용되는 엔트로피 코딩 방식은 가변 길이 코딩 또는 산술 코딩일 수 있다. 이것에 의해, 엔트로피 디코더(150)는 비트스트림으로부터 잔차 신호를 나타내는 변환 계수 레벨들을 복구시키고, 이 복구된 것을 역 변환기(152)에 포워딩한다. 더 나아가, 엔트로피 디코더(150)는 상술한 추출기(88)로서 역할을 하며 비트스트림으로부터 모든 코딩 모드들 및 연관된 코딩 파라미터들을 복구시켜서 이것들을 예측기(156)에 포워딩한다. 추가적으로, 파티션화 정보 및 병합 정보는 추출기(150)에 의해 비트스트림으로부터 추출된다. 역으로 변환된, 즉 재구성된 잔차 신호 및 예측기에 의해 유도된 예측 신호는 가산기(154)에 의해 가산되는 것과 같이 결합되고, 이어서 가산기(154)는 이에 따라 복구되고 재구성된 신호를 출력부(160)에서 출력하여 이것을 예측기(156)에 포워딩한다.

[0051]

도 5와 도 6의 비교로부터 명백해지는 바와 같이, 엘리먼트들(152, 154, 156)은 도 5의 엘리먼트들(104, 110,

106)에 기능적으로 대응한다.

[0052]

도 1 내지 도 6의 위 설명에서, 코딩 화상(20)에 수반된 파라미터들 중 일부를 변경시킴으로써 화상(20)의 잠재적인 하위분할들 및 대응 입도에 대한 여러가지 상이한 가능성들이 제시되었다. 이러한 한가지 가능성을 다시 도 7a와 도 7b를 참조하여 설명한다. 도 7a는 화상(20) 중의 일부분을 도시한다. 도 7a의 실시예에 따르면, 인코더와 디코더는 제일먼저 화상(20)을 트리 루트 블록들(200)로 하위분할하도록 구성된다. 이러한 한가지 트리 루트 블록이 도 7a에서 도시된다. 트리 루트 블록들로의 화상(20)의 하위분할은 점선들로 나타난 바와 같이 열과 행으로 규칙적으로 행해진다. 트리 루트 블록들(200)의 크기는 인코더에 의해 선택되고 비트스트림(30)에 의해 디코더로 신호전달될 수 있다. 대안적으로, 이러한 트리 루트 블록들(200)의 크기는 디폴트에 의해 고정될 수 있다. 코딩 블록들 또는 코딩 유닛들이라고 칭해질 수 있는 위에서 확인된 블록들(40)을 산출시키기 위해 트리 루트 블록들(200)은 쿼드(quad) 트리 파티션화의 이용에 의해 하위분할된다. 이러한 코딩 블록들 또는 코딩 유닛들은 도 7a에서 얇은 실선들로 그려진다. 이에 의해, 인코더는 각각의 트리 루트 블록(200)에 하위분할 정보를 수반시키고 하위분할 정보를 비트스트림 내에 삽입한다. 이러한 하위분할 정보는 트리 루트 블록(200)이 어떻게 블록들(40)로 하위분할되는지를 나타내준다. 이러한 블록들(40)의 입도 및 유닛들에서, 예측 모드는 화상(20) 내에서 변한다. 위에서 나타난 바와 같이, 각각의 블록(40) - 또는 인터 예측 모드와 같은 일정한 예측 모드를 갖는 각각의 블록 - 에는 지원된 어느 파티션화 패턴이 각각의 블록(40)에 대해 이용될 것인지에 대한 파티션화 정보가 수반된다. 하지만, 이와 관련하여, 앞서언급한 플러그/구문 엘리먼트는 또한, 공동의 신호전달 상태를 취할 때, 각각의 블록(40)에 대한 지원된 파티션화 모드들 중 하나를 동시에 신호전달할 수 있고, 이로써 이러한 블록(40)에 대한 또다른 파티션화 정보의 명시적 송신이 인코더측에서 억제될 수 있고 이에 따라 디코더측에서 예상되지 않을 수 있다. 도 7a의 예시의 경우에서, 많은 코딩 블록들(40)에 대해, 코딩 블록(40)이 대응 파티션과 공간적으로 일치하도록 비파티션화 모드가 선택되었다. 다시 말하면, 코딩 블록(40)은, 동시에, 연관된 예측 파라미터들의 각각의 세트를 갖는 파티션이다. 이어서 예측 파라미터들의 종류는 각각의 코딩 블록(40)과 연관된 모드에 의존한다. 하지만, 더욱 더 파티션화되도록 다른 코딩 블록들이 예시적으로 도시된다. 예컨대 트리 루트 블록(200)의 우측 상단 모서리에 있는 코딩 블록(40)은 네 개의 파티션들로 파티션화되는 것이 도시되는 반면에, 트리 루트 블록(200)의 우측 하단 모서리에 있는 코딩 블록은 두 개의 파티션들로 수직하게 하위분할되는 것이 예시적으로 도시된다. 파티션들로의 파티션화를 위한 하위분할은 점선들로 도시된다. 도 7a는 또한 이에 따라 정의된 파티션들간의 코딩 순서를 도시한다. 도시된 바와 같이, 깊이 우선 탐색(depth-first traversal) 순서가 이용된다. 트리 루트 블록 경계들을 가로질러, 코딩 순서는 트리 루트 블록들(200)의 행들이 화상(20)의 위에서 아래로 행별로 스캐닝되는 스캔 순서로 계속될 수 있다. 이러한 조치에 의해, 어떠한 파티션이 그 왼쪽 경계와 좌측 경계에 인접해 있는 이전에 코딩된 파티션을 가질 최대 가능성을 갖는 것이 가능하다. 각각의 블록(40) - 또는 인터 예측 모드와 같은 일정한 예측 모드를 갖는 각각의 블록 - 은 대응 파티션들에 대한 병합이 활성화되어 있는지 아닌지의 여부에 대해 표시하는 비트스트림 내의 병합 스위치 표시자를 가질 수 있다. 파티션들/예측 유닛들로의 블록들의 파티션화는 최대 두 개의 파티션들의 파티션화로 제한될 수 있으며, 이러한 규칙의 유일한 예외는 블록들(40)의 가장 작은 잠재적인 블록 크기의 경우일 뿐이라는 것을 유념해야 한다. 이것은, 블록들(40)을 획득하기 위해 쿼드 트리 하위분할을 이용하는 경우, 화상(20)을 블록(40)으로 하위분할하기 위한 하위분할 정보와 블록(40)을 파티션들로 하위분할하기 위한 파티션화 정보간의 리던던시를 회피한다. 대안적으로, 비대칭적인 것들을 포함하거나 또는 포함하지 않고서, 하나 또는 두 개의 파티션들로의 파티션화만이 허용될 수 있다.

[0053]

도 7b는 하위분할 트리를 도시한다. 트리 루트 블록(200)의 하위분할은 실선들로 도시되는 반면에, 점선들은 코딩 블록들(40)인, 쿼드 트리 하위분할의 리프(leaf) 블록들의 파티션화를 상징한다. 즉, 코딩 블록들의 파티션화는 일종의 쿼드 하위분할의 확장을 나타낸다.

[0054]

이미 위에서 언급한 바와 같이, 변환 블록들이 각각의 코딩 블록(40)의 상이한 하위분할을 나타낼 수 있도록 각각의 코딩 블록(40)은 변환 블록들로 병렬적으로 하위분할될 수 있다. 도 7a와 도 7b에서는 도시되지 않은 이러한 변환 블록들 각각에 대해, 코딩 블록들의 잔차 신호를 변환하기 위한 변환은 개별적으로 수행될 수 있다.

[0055]

이하에서는 본 발명의 추가적인 실시예들을 설명한다. 위 실시예들은 한편으로 블록 병합과 다른 한편으로 블록 파티션화간의 관계에 초점을 맞췄던 반면에, 이하의 설명은 또한 SKIP/DIRECT 모드들과 같은, 현재 코덱들에서 알려진 다른 코딩 원리들에 관한 본 출원의 양태들을 포함한다. 그럼에도 불구하고, 후속하는 설명은 단지 별개의 실시예들, 즉 위에서 설명된 것과는 별개의 실시예들만을 설명하는 것으로서 간주되어서는 안된다. 이보다는, 아래의 설명은 또한 위에서 설명한 실시예들에 대한 잠재적인 구현 상세사항들을 공개하는 것이다. 따라서, 아래의 설명은 위에서 이미 설명한 도면들의 참조 부호들을 이용하며, 이로써 아래에서 설명한 각각의

잠재적인 구현은 또한, 위에서 설명한 실시예들의 잠재적인 변형들을 정의한 것이다. 이러한 변형들 대부분은 위 실시예들로 개별적으로 이전될 수 있다.

[0056] 다시 말하면, 본 출원의 실시예들은 샘플들의 세트들에 대한 잔차 데이터의 부재와 병합의 결합된 신호전달에 의해 이미지 및 비디오 코딩 응용들에서 보조 정보 레이트를 감소시키기 위한 방법들을 설명한다. 다시 말하면, 이미지 및 비디오 코딩 응용들에서의 보조 정보 레이트는 잔차 데이터의 부재를 표시하는 구문 엘리먼트들 및 병합 방식들의 활용을 표시하는 구문 엘리먼트들을 결합함으로써 감소된다.

[0057] 더 나아가, 이러한 변형들과 추가적인 상세사항들을 설명하기에 앞서, 화상 및 비디오 코덱들에 대한 개관이 제공된다.

[0058] 이미지 및 비디오 코딩 응용들에서, 화상과 연관된 샘플 어레이는 일반적으로 샘플들의 특정 세트들(또는 샘플 세트들)로 파티션화되며, 이러한 샘플 세트들은 직사각형 또는 정방형 블록들 또는 임의적인 형상의 영역들, 삼각형들, 또는 임의의 다른 형상들을 포함한 임의의 다른 샘플들의 집합체를 나타낼 수 있다. 샘플 어레이들의 하위분할은 구문에 의해 고정될 수 있거나 또는 하위분할은 (적어도 부분적으로) 비트스트림 내부에서 신호전달된다. 하위분할 정보를 신호전달하기 위한 보조 정보 레이트를 작게 유지하기 위해, 구문은 일반적으로 블록들을 보다 작은 블록들로 하위분할하는 것과 같은 단순한 파티션화를 초래시키는 제한된 횟수의 선택들만을 허용한다. 종종 이용되는 파티션화 방식은 사각형 블록을 네 개의 보다 작은 사각형 블록들로 파티션화하는 것이거나, 또는 동일한 크기의 두 개의 직사각형 블록들로 파티션화하는 것이거나, 또는 상이한 크기들의 두 개의 직사각형 블록들로 파티션화하는 것이며, 여기서 실제로 채워지는 파티션화는 비트스트림 내에서 신호전달된다. 샘플 세트들은 잔차 코딩 모드들 또는 예측 정보 등을 규정할 수 있는 특정 코딩 파라미터들과 연관이 있다. 비디오 코딩 응용들에서, 파티션화는 종종 모션 표현을 목적으로 행해진다. (파티션화 패턴 내부의) 블록의 모든 샘플들은 모션 파라미터들의 샘플 세트와 연관이 있으며, 이러한 파라미터들은 예측의 유형(예컨대, 리스트 0, 리스트 1, 또는 양방향 예측; 및/또는 병진 또는 아핀(affine) 예측 또는 상이한 모션 모델을 갖는 예측)을 규정하는 파라미터들, 활용된 참조 화상들을 규정하는 파라미터들, 차이(difference)로서 일반적으로 예측기에 송신되는, 참조 화상들에 관한 모션(예컨대, 변위 벡터들, 아핀 모션 파라미터 벡터들, 또는 임의의 다른 모션 모델을 위한 모션 파라미터 벡터들)을 규정하는 파라미터들, 모션 파라미터들의 정확도(예컨대, 하프 샘플 또는 쿼터 샘플 정확도)를 규정하는 파라미터들, (예컨대, 조명 보정을 목적으로 하는) 참조 샘플 신호의 가중치를 규정하는 파라미터들, 또는 현재 블록의 모션 보정된 예측 신호를 유도하기 위해 활용되는 보간 필터를 규정하는 파라미터들을 포함할 수 있다. 각각의 샘플 세트에 대해, (예컨대, 예측 및/또는 잔차 코딩을 규정하기 위한) 개별적인 코딩 파라미터들이 송신된다. 개선된 코딩 효율을 달성하기 위해, 본 발명은 두 개 이상의 샘플 세트들을 소위 말하는 샘플 세트들의 그룹들로 병합하기 위한 특정 실시예들 및 방법을 제공한다. 이러한 그룹의 모든 샘플 세트들은 샘플 세트들 중의 하나의 샘플 세트와 함께 그룹으로 송신될 수 있는 동일한 코딩 파라미터들을 공유한다. 이렇게 함으로써, 코딩 파라미터들은 샘플 세트들의 그룹의 각각의 샘플 세트마다 개별적으로 송신될 필요가 없으며, 이 대신에 코딩 파라미터들은 샘플 세트들의 전체 그룹에 대해서 단 한번 송신된다.

[0059] 그 결과 코딩 파라미터들을 송신하기 위한 보조 정보 레이트는 감소되고 전체적인 코딩 효율성은 개선된다. 대안적인 접근법으로서, 하나 이상의 코딩 파라미터들을 위한 추가적인 정제는 샘플 세트들의 그룹 중의 하나 이상의 샘플 세트들을 위해 송신될 수 있다. 이러한 정제는 모든 샘플 세트들의 그룹에 적용될 수 있거나 또는 송신되는 샘플 세트에만 적용될 수 있다.

[0060] 본 발명의 몇몇의 실시예들은 (상술한) 다양한 하위블록들(50, 60)로의 블록의 파티션화를 병합 프로세스와 결합시킨다. 일반적으로, 이미지 또는 비디오 코딩 시스템들은 블록(40)을 위한 다양한 파티션화 패턴들을 지원한다. 예시로서, 정사각형 블록은 파티션화되지 않을 수 있거나 또는 동일한 크기의 네 개의 정사각형 블록들로 파티션화될 수 있거나, 또는 동일한 크기의 두 개의 직사각형 블록들로 파티션화될 수 있거나(이 경우 정사각형 블록은 수직하게 또는 수평하게 분할될 수 있다), 또는 상이한 크기들의 직사각형 블록들로 (수평하게 또는 수직하게) 파티션화될 수 있다. 설명한 예시적인 파티션 패턴들이 도 8에서 도시된다. 위 설명에 더하여, 파티션화는 심지어 하나 이상의 레벨의 파티션화를 수반할 수 있다. 예를 들어, 정사각형 하위블록들은 택일적 사항으로서 동일한 파티션화 패턴들을 이용하여 추가적으로 파티션화될 수 있다. 이러한 파티션화 프로세스가, 예컨대 (정사각형 또는 직사각형) 블록이 그 이웃 블록들 중 하나와 병합하도록 해주는 병합 프로세스와 결합될 때 일어나는 문제는 파티션화 패턴들과 병합 신호들의 상이한 결합들에 의해 동일한 결과적인 파티션화가 달성될 수 있다는 점이다. 이에 따라, 상이한 코드워드들을 이용하여 비트스트림 내에서 동일한 정보가 송신될 수 있는데, 이것은 코딩 효율 측면에서 명백히 차선책이다. 단순한 예시로서, (도 8의 좌측 상단 모서리에서 도시된) 더 이상 파티션화되지 않는 정사각형 블록을 고려해본다. 이러한 파티션화는 이 블록(40)이 하위분할되지 않는 구문

엘리먼트를 보냄으로써 곧바로 신호전달될 수 있다. 하지만, 예컨대 이러한 블록이 두 개의 수직하게(또는 수평하게) 정렬된 직사각형 블록들(50, 60)로 하위분할되는 것을 규정하는 구문 엘리먼트를 보냄으로써 동일한 패턴이 또한 신호전달될 수 있다. 그런 후 이러한 직사각형 블록들 중 두번째가 첫번째 직사각형 블록과 병합되는 것을 규정하는 병합 정보를 송신할 수 있고, 그 결과 블록이 더 이상 분할되지 않는다는 것을 신호전달할 때와 아주 똑같은 파티션화가 초래된다. 위 내용은 또한, 블록이 네 개의 정사각형 하위블록들로 하위분할되고 이어서 이러한 네 개의 블록들 모두를 효과적으로 병합하는 병합 정보를 송신하는 것을 제일먼저 규정함으로써 달성될 수 있다. (여기서는 동일한 것을 신호전달하기 위해 상이한 코드워드들을 갖기 때문에) 이러한 개념은 명백히 차선책이다.

[0061]

본 발명의 몇몇의 실시예들은 보조 정보 레이트를 감소시키며 이에 따라 병합 개념과 블록을 위한 상이한 파티션화 패턴들을 제공하는 개념과의 결합을 위한 코딩 효율성을 증가시킨다. 만약 도 8에서의 예시적인 파티션화 패턴들을 고찰해보면, 임의의 파티션화 패턴들에 의해 두 개의 직사각형 블록들로 더 이상 분할되지 않는 블록의 "시뮬레이션"은 직사각형 블록이 첫번째 직사각형 블록과 병합되는 경우를 금지(즉, 비트스트림으로부터 구문 규정을 배제시킴)시킬 때 회피될 수 있다. 이 문제를 보다 깊게 고찰해보면, 두번째 직사각형 블록을 첫번째 직사각형 블록과 동일한 파라미터들(예컨대, 예측을 규정하는 정보)과 연관된 임의의 다른 이웃(즉, 첫번째 직사각형 블록이 아닌 블록)과 병합시킴으로써 하위분할되지 않는 패턴을 "시뮬레이션"하는 것도 가능하다. 지원된 파티션화 패턴들 중 하나를 신호전달함으로써 또한 달성될 수 있는 패턴을 이러한 병합 파라미터들이 초래시킬 때 비트스트림 구문으로부터 배제되는 특정한 병합 파라미터들을 송신하는 방식으로 병합 정보를 송신하는 것을 조정함으로써 리던던시는 회피될 수 있다. 예로서, 두번째 블록(즉, 도 1과 도 3의 경우, 참조번호 60)을 위한 병합 정보를 보내기 전에, 예컨대 도 1과 도 3에서 도시된, 두 개의 직사각형 블록들로의 하위분할을 현재의 파티션화 패턴이 규정하는 경우, 잠재적인 병합 후보들 중 어느 것이 도 1과 도 3의 경우에서의 첫번째 직사각형 블록, 즉 참조번호 50과 동일한 파라미터들(예컨대, 예측 신호를 규정하기 위한 파라미터들)을 갖는지가 체크될 수 있다. (첫번째 직사각형 블록 자체를 포함하여) 동일한 모션 파라미터들을 갖는 모든 후보들은 병합 후보들의 세트로부터 제거된다. 병합 정보를 신호전달하기 위해 송신되는 코드워드들 또는 플래그들은 결과적인 후보 세트에 적용된다. 후보 세트가 파라미터 체크로 인해 비어지는 경우, 어떠한 병합 정보도 송신되지 않을 수 있다. 후보 세트가 단하나의 엔트리로만 구성되는 경우, 블록이 병합되어 있는지 아닌지 여부만이 신호전달되되, 후보는 디코더측 등에서 유도될 수 있기 때문에 신호전달될 필요는 없다. 위 예시의 경우, 동일한 개념이 또한 정사각형 블록을 보다 작은 네 개의 정사각형 블록들로 분할하는 파티션화 패턴에 활용된다. 여기서, 병합 플래그들의 송신은 하위분할을 규정하지 않는 파티션화 패턴이나 또는 동일 크기의 두 개의 직사각형 블록들로의 하위분할을 규정하는 두 개의 파티션화 패턴들 중 어느 것도 병합 플래그들의 결합에 의해 달성될 수 없는 방식으로 적용된다. 비록, 여기서는 특정 파티션화 패턴들을 갖는 위 예시에 대한 개념을 대부분 설명하였지만, (또다른 파티션화 패턴과 대응하는 병합 정보의 결합에 의한 특정 파티션화 패턴의 규정을 회피하는) 이와 동일한 개념은 임의의 다른 파티션화 패턴들의 세트에 대해서도 활용될 수 있다는 것은 분명해져야 한다.

[0062]

고려될 필요가 있는 또다른 양태는 병합 개념은 비디오 코딩 설계들에서 발견되는 SKIP 또는 DIRECT 모드들과 어느 정도 유사하다는 점이다. SKIP/DIRECT 모드들에서는, 기본적으로 현재 블록을 위해 어떠한 모션 파라미터들도 송신되지 않되, 이 파라미터들은 공간적 및/또는 시간적 이웃으로부터 추론된다. SKIP/DIRECT 모드들의 특정한 효율적인 개념에서, 모션 파라미터 후보들(참조 프레임 인덱스들, 변위 벡터들 등)의 리스트는 공간적 및/또는 시간적 이웃으로부터 생성되며 후보 파라미터들 중 어느 것이 선택되는지를 규정하는 인덱스가 이 리스트 내에서 송신된다. 양방향 예측 블록들(또는 멀티 추정 프레임들)의 경우, 별개의 후보들이 각각의 참조 리스트마다 신호전달될 수 있다. 잠재적 후보들은 현재 블록의 최상단쪽에 있는 블록, 현재 블록의 좌측쪽에 있는 블록, 현재 블록의 좌측 최상단쪽에 있는 블록, 현재 블록의 우측 최상단쪽에 있는 블록, 이러한 다양한 후보들의 중간값 예측기, 하나 이상의 이전 참조 프레임들에서 공동 위치한 블록(또는 이미 코딩된 임의의 다른 블록, 또는 이미 코딩된 블록들로부터 획득된 조합)을 포함할 수 있다.

[0063]

SKIP/DIRECT를 병합 개념과 결합시키는 것은 블록이 SKIP/DIRECT 또는 병합 모드 중 어느 하나를 이용하여 코딩될 수 있다는 것을 의미한다. SKIP/DIRECT 및 병합 개념들은 유사하지만, 이 두 개의 개념들간에는 차이점들이 존재하는데, 이것은 섹션 1에서 보다 자세하게 설명한다. SKIP와 DIRECT간의 주요한 차이점은 SKIP 모드는 어떠한 잔차 신호도 송신되지 않는다는 것을 추가로 신호전달한다는 점이다. 병합 개념이 이용될 때, 일반적으로 블록이 비제로(non-zero) 변환 계수 레벨들을 포함하는지 여부를 신호전달하는 플래그가 송신된다.

[0064]

개선된 코딩 효율을 달성하기 위해, 위와 아래에서 설명하는 실시예들은 샘플 세트가 또다른 샘플 세트의 코딩 파라미터들을 이용하는지 여부에 대한 신호전달과 어떠한 잔차 신호도 블록을 위해 송신되지 않는지 여부에 대

한 신호전달을 결합시킨다. 결합된 플래그는 샘플 세트가 또다른 샘플 세트의 코딩 파라미터들을 이용한다는 것과 어떠한 잔차 데이터도 송신되지 않는다는 것을 표시한다. 이 경우, 두 개 대신 단하나의 플래그만이 송신될 필요가 있다.

[0065]

위에서 언급한 바와 같이, 본 발명의 몇몇의 실시예들은 또한 비트스트림을 생성하기 위한 보다 큰 자유도를 인코더에 제공하는데, 그 이유는 이러한 병합 접근법은 비트스트림에서 리던던시를 도입시키지 않고 화상의 샘플 어레이들을 위한 파티션화를 선택하는 횟수 가능성들을 상당히 증가시키기 때문이다. 인코더는 예컨대 특정한 레이트 왜곡 수치를 최소화하기 위해 보다 많은 옵션들 중에서 선택할 수 있기 때문에, 코딩 효율성은 개선될 수 있다. 예로서, 하위파티션화와 병합의 결합에 의해 표현될 수 있는 추가적인 패턴들 중 몇몇의 패턴(예컨대, 도 9에서의 패턴들)이 (모션 추정과 모드 결정을 위해 대응하는 블록 크기들을 이용하여) 추가적으로 테스트될 수 있고, 순수한 파티션화(도 8) 및 파티션화와 병합(도 9)에 의해 제공된 패턴들 중 최상의 것이 특정한 레이트 왜곡 수치에 기초하여 선택될 수 있다. 게다가 각각의 블록에 대해, 이미 코딩된 후보 세트들 중 임의의 것과의 병합이 특정한 레이트 왜곡 수치의 감소를 야기시키는지 여부가 테스트될 수 있고 그런 후 대응하는 병합 플래그들이 인코딩 프로세스 동안에 설정된다. 요약하면, 인코더를 동작시키기 위한 가능성들은 여러가지가 있다. 단순한 접근법에서, 인코더는 제일먼저 (최신의 코딩 방식들에서와 같은) 샘플 어레이들의 최상의 하위분할을 결정할 수 있다. 그런 후 각각의 샘플 세트에 대해, 또다른 샘플 세트 또는 또다른 샘플 세트들의 그룹과의 병합이 특정한 레이트 왜곡 비용을 감소시키는지 여부에 대해 체크될 수 있다. 여기서, 병합된 샘플 세트들의 그룹과 연관된 예측 파라미터들은 (예컨대 새로운 모션 검색을 수행함으로써) 재추정될 수 있거나 또는 병합을 위한 후보 샘플 세트(또는 샘플 세트들의 그룹)와 현재 샘플 세트를 위해 이미 결정되었던 예측 파라미터들은 고려된 샘플 세트들의 그룹에 대해 평가될 수 있다. 보다 확장적인 접근법에서, 샘플 세트들의 추가적인 후보 그룹들에 대해 특정한 레이트 왜곡 비용 수치가 평가될 수 있다. 특정 예시로서, 다양한 잠재적인 파티션화 패턴들(예컨대 도 8 참조)을 테스트할 때, 파티션화와 병합의 결합에 의해 표현될 수 있는 모든 패턴 또는 일부 패턴(예컨대 도 9 참조)이 추가적으로 테스트될 수 있다. 즉, 모든 패턴들에 대해, 특정 모션 추정 및 모드 결정 프로세스가 수행되며 가장작은 레이트 왜곡 수치를 산출시키는 패턴이 선택된다. 이 프로세스는 또한 상술한 낮은 복잡도 프로세스와 결합될 수 있으며, 이로써 결과적인 블록들에 대해 (예컨대, 도 8과 도 9의 패턴들 외부에 있는) 이미 코딩된 블록들과의 병합이 레이트 왜곡 수치의 감소를 산출시키는지 여부가 추가적으로 테스트된다.

[0066]

아래에서는, 도 1, 도 2 및 도 5에서의 인코더 및 도 3, 도 4 및 도 6의 디코더와 같은, 위에서 개술된 실시예들을 위한 몇몇의 잠재적인 상세한 구현을 설명한다. 위에서 이미 언급한 바와 같이, 위 것들은 이미지 및 비디오 코딩에서 이용가능하다. 상술한 바와 같이, 화상들 또는 화상들을 위한 샘플 어레이들의 특정 세트들은 블록들로 분해될 수 있고, 이것은 특정 코딩 파라미터들과 연관된다. 화상들은 보통 다수의 샘플 어레이들로 구성된다. 게다가, 화상은 또한 예컨대 투명도 정보 또는 깊이 맵들을 규정할 수 있는 추가적인 보조 샘플 어레이들과 연관될 수 있다. (보조 샘플 어레이들을 포함한) 화상의 샘플 어레이들은 소위 말하는 하나 이상의 평면 그룹들로 그룹화될 수 있고, 각각의 평면 그룹은 하나 이상의 샘플 어레이들로 구성된다. 화상의 평면 그룹들은 독립적으로 코딩될 수 있거나, 또는 화상이 하나보다 많은 평면 그룹과 연관된 경우, 동일 화상의 다른 평면 그룹들로부터의 예측으로 코딩될 수 있다. 각각의 평면 그룹은 보통 블록들로 분해된다. 블록들(또는 샘플 어레이들의 대응 블록들)은 인터 화상 예측 또는 인트라 화상 예측에 의해 예측이 된다. 블록들은 상이한 크기들을 가질 수 있으며, 정방형 또는 직사각형일 수 있다. 블록들로의 화상의 파티션화는 구문에 의해 고정될 수 있거나, 또는 비트스트림 내에서 (적어도 부분적으로) 신호전달될 수 있다. 종종 미리정의된 크기들의 블록들에 대한 하위분할을 신호전달하는 구문 엘리먼트들이 송신된다. 이러한 구문 엘리먼트들은 블록이 보다 작은 블록들로 하위분할되고 예컨대 예측을 목적으로 코딩 파라미터들과 연관되어 있는지 여부 및 그 방법을 규정할 수 있다. 잠재적인 파티션화 패턴들의 예시가 도 8에서 도시된다. 블록의 모든 샘플들 (또는 샘플 어레이들의 대응 블록들)에 대해, 연관된 코딩 파라미터들의 디코딩이 일정한 방식으로 규정된다. 예시에서, 블록 내 모든 샘플들은 (이미 코딩된 화상들의 세트에서 참조 화상을 식별하는) 참조 인덱스들, (현재 화상과 참조 화상간의 블록들의 이동에 대한 수치를 규정하는) 모션 파라미터들, 보간 필터, 인트라 예측 모드들 등을 규정하기 위한 파라미터와 같은, 예측 파라미터들의 샘플 세트를 이용하여 예측된다. 모션 파라미터들은 수평적 및 수직적 성분으로 변위 벡터들에 의해 표현될 수 있거나 또는 여섯 개의 성분들로 구성된 아핀 모션 파라미터들과 같은 보다 높은 차수의 모션 파라미터들에 의해 표현될 수 있다. (참조 인덱스들 및 모션 파라미터들과 같은) 특정 예측 파라미터들의 하나보다 많은 세트가 단일 블록과 연관되는 것도 가능하다. 이 경우, 이러한 특정 예측 파라미터들의 각 세트에 대해, 블록 (또는 샘플 어레이들의 대응 블록들)을 위한 단일 중간 예측 신호가 생성되고, 최종적인 예측 신호가 중간 예측 신호들을 중첩시키는 것을 포함한 결합에 의해 구축된다. 대응하는 가중화 파라미터들 및 (가중화

함에 가산되는) 잠재적으로 또한 일정한 오프셋은 화상, 또는 참조 화상, 또는 참조 화상들의 세트에 대해 고정될 수 있거나, 또는 이것들은 대응 블록을 위한 예측 파라미터들의 세트에 포함될 수 있다. 원래 블록들(또는 샘플 어레이들의 대응 블록들)과 이러한 블록들의 예측 신호들간의 차이(이것은 또한 잔차 신호라고도 불리어진다)는 보통 변환되고 양자화된다. 종종, 이차원 변환이 잔차 신호(또는 잔차 블록에 대한 대응하는 샘플 어레이들)에 적용된다. 변환 코딩의 경우, 예측 파라미터들의 특정 세트가 이용되는 블록들(또는 샘플 어레이들의 대응 블록들)은 변환을 적용하기 전에 추가적으로 나뉘질 수 있다. 변환 블록들은 예측을 위해 이용되는 블록들과 동등하거나 또는 이보다 작을 수 있다. 변환 블록은 예측을 위해 이용되는 블록을 하나보다 많이 포함하는 것도 가능하다. 상이한 변환 블록들은 상이한 크기들을 가질 수 있으며, 변환 블록들은 정방형 또는 직사각형 블록들을 나타낼 수 있다. 도 1 내지 도 7에 대한 위 예시에서, 첫번째 하위분할의 리프 노드들, 즉 코딩 블록들(40)은 한편으로는 코딩 파라미터들의 입도를 정의하는 파티션으로 추가적으로 파티션화될 수 있고, 다른 한편으로는 이차원 변환이 개별적으로 적용된 변환 블록들로 이와 병렬적으로 추가적으로 파티션화될 수 있는 것이 가능하다는 점을 유념해야 한다. 변환 후, 결과적인 변환 계수들은 양자화되고 소위 말하는 변환 계수 레벨들이 획득된다. 변환 계수 레벨들 뿐만이 아니라 예측 파라미터들 및, (존재하는 경우) 하위분할 정보는 엔트로피 코딩된다. 특히, 변환 블록들을 위한 코딩 파라미터들을 잔차 파라미터들이라고 부른다. 잔차 파라미터들 뿐만이 아니라 예측 파라미터들 및, (존재하는 경우) 하위분할 정보는 엔트로피 코딩될 수 있다. H.264와 같은 최신의 비디오 코딩 표준들에서, 코딩된 블록 플래그(coded block flag; CBF)라고 불리우는 플래그는 모든 변환 계수 레벨들이 제로이고 이에 따라 어떠한 잔차 파라미터들도 코딩되지 않는다는 것을 신호전달할 수 있다. 본 발명에 따르면, 이러한 신호전달은 병합 활성화 신호전달 내로 결합된다.

[0067]

최신의 이미지 및 비디오 코딩 표준들에서, 구문에 의해 제공되는 블록들로 화상(또는 평면 그룹)을 하위분할하는 가능성들은 극히 제한적이다. 보통은, 미리정의된 크기의 블록이 보다 작은 블록들로 하위분할될 수 있는지 여부(및 잠재적으로는 그 방법)만이 규정될 수 있다. 예로서, H.264에서 가장 큰 블록 크기는 16x16이다. 16x16 블록들을 또한 매크로블록들이라고 부르며 각각의 화상은 제1 단계에서 매크로블록들로 파티션화된다. 각각의 16x16 매크로블록에 대해, 16x16 블록으로서 코딩되는지, 또는 두 개의 8x8 블록들로서 코딩되는지, 또는 두 개의 8x16 블록들로서 코딩되는지, 또는 네 개의 8x8 블록들로서 코딩되는지 여부가 신호전달될 수 있다. 만약 16x16 블록이 네 개의 8x8 블록으로 하위분할되는 경우, 이러한 8x8 블록들 각각은 하나의 8x8 블록으로서 코딩될 수 있거나, 또는 두 개의 8x4 블록들로서 코딩될 수 있거나, 또는 두 개의 4x8 블록들로서 코딩될 수 있거나, 또는 네 개의 4x4 블록들로서 코딩될 수 있다. 최신의 이미지 및 비디오 코딩 표준들에서 블록들로의 파티션화를 규정하는 작은 가능성들의 세트는 하위분할 정보를 신호전달하기 위한 보조 정보 레이트가 작게 유지될 수 있다는 장점을 갖지만, 블록들을 위한 예측 파라미터들을 송신하는데 필요한 비트 레이트는 아래에서 설명하는 바와 같이 상당해질 수 있다는 단점을 갖는다. 예측 정보를 신호전달하기 위한 보조 정보 레이트는 일반적으로 블록에 대한 상당한 양의 전체 비트 레이트를 나타낸다. 코딩 효율성은 이러한 보조 정보가 감소될 때 증가될 수 있으며, 이것은 예컨대 보다 큰 블록 크기들을 이용함으로써 달성될 수 있다. H.264와 비교하여 지원된 파티션화 패턴들의 세트를 증가시키는 것도 가능하다. 예를 들어, 도 8에서 도시된 파티션화 패턴들은 모든 크기들(또는 선택된 크기들)의 정사각형 블록들을 위해 제공될 수 있다. 비디오 시퀀스의 실제 이미지들 또는 화상들은 특정한 특성들을 갖는 임의적으로 형상화된 객체들로 구성된다. 예로서, 이러한 객체들 또는 객체들의 부분들은 고유한 텍스처 또는 고유한 모션에 의해 특징지어진다. 일반적으로, 예측 파라미터들의 동일 세트가 이러한 객체 또는 객체의 부분에 대해 적용될 수 있다. 하지만 객체 경계들은 일반적으로 커다란 예측 블록들(예컨대 H.264에서 16x16 매크로블록들)에 대한 잠재적인 블록 경계들과 일치하지 않는다. 인코더는 일반적으로 특정 레이트 왜곡 비용 수치의 최소값을 초래시키는 (제한된 가능성들의 세트 중의) 하위분할을 결정한다. 임의적으로 형상화된 객체들의 경우, 이것은 많은 수의 작은 블록들을 초래시킨다. 이러한 진술은 위에서 (언급된) 보다 많은 파티션화 패턴들이 제공될 때에도 들어맞는다. 파티션화 패턴들의 양은 너무 커지지 않아야 한다는 것을 유념해야 하는데, 그 이유는 이러한 패턴들을 신호전달하고 프로세싱하기 위해 많은 보조 정보 및/또는 인코더/디코더 복잡도가 요구되기 때문이다. 따라서, 임의적으로 형상화된 객체들은 파티션화로 인해 방대한 수의 작은 블록들을 종종 초래시킨다. 이러한 작은 블록들 각각은 송신될 필요가 있는 예측 파라미터들의 세트와 연관되기 때문에, 보조 정보 레이트는 전체 비트 레이트의 중요 부분이 될 수 있다. 여러 개의 작은 블록들은 여전히 동일 객체 또는 객체의 부분의 영역들을 나타내기 때문에, 획득된 복수의 블록들에 대한 예측 파라미터들은 동일하거나 또는 매우 유사하다. 직관적으로, 블록을 하위분할하도록 하는 것뿐만이 아니라, 하위분할 이후에 획득되는 두 개 이상의 블록들을 병합하도록 하는 방식으로 구문이 확장될 때 코딩 효율성은 증가될 수 있다. 그 결과로, 동일한 예측 파라미터들로 코딩되는 블록들의 그룹을 획득할 것이다. 이러한 블록들의 그룹을 위한 예측 파라미터들은 단 한번 코딩될 필요만 있다. 예컨대 도 1 내지 도 7의 위 예시들에서, 현재 블록(40)을

위한 코딩 파라미터들은 병합이 발생하는 경우에는 송신되지 않는다. 즉, 인코더는 현재 블록과 연관된 코딩 파라미터들을 송신하지 않으며, 디코더는 비트스트림(30)이 현재 블록(40)을 위한 코딩 파라미터들을 포함한다고 예측하지 않는다. 이보다는, 본 발명의 특정 실시예들에 따라, 병합된 현재 블록(40)을 위해 단지 정제 정보가 운송될 수 있다. 화상(20)의 다른 코딩 블록들(40)에 대해 후보 세트 및 그 감소의 결정뿐만이 아니라 병합 등이 또한 수행된다. 코딩 블록들은 다소간에 코딩 체인을 따라 코딩 블록들의 그룹들을 형성하며, 이러한 그룹들을 위한 코딩 파라미터들은 단 한번에 전체가 비트스트림 내에서 송신된다.

[0068]

코딩된 예측 파라미터들의 갯수를 감소시킴으로써 절감되는 비트 레이트가 병합 정보를 코딩하는데 추가적으로 소비되는 비트 레이트보다 큰 경우, 설명된 병합은 증가된 코딩 효율을 초래시킨다. (병합을 위한) 설명한 구문 확장은 리던던시를 도입시키지 않고서 화상 또는 평면 그룹의 블록들로의 파티션화를 선택할 때의 추가적인 자유도를 인코더에 제공한다는 것이 추가적으로 언급되어야 한다. 인코더는 제일먼저 하위분할을 행하고 그런 후 결과적인 블록들 중의 몇몇이 예측 파라미터들의 동일 세트를 갖는지 여부를 체크하는 것으로 제한되지 않는다. 하나의 단순한 대안구성으로서, 인코더는 최신의 코딩 기술들에서와 같이 제일먼저 하위분할을 결정할 수 있다. 그런 후 각각의 블록에 대해, 이웃 블록들 중 하나(또는 블록들의 이미 결정된 연관 그룹)과의 병합이 레이트 왜곡 비용 수치를 감소시키는지 여부를 체크할 수 있다. 여기서, 새로운 블록들의 그룹과 연관된 예측 파라미터들은 (예컨대 새로운 모션 검색을 수행함으로써) 채추정될 수 있거나 또는 이웃하는 블록 또는 블록들의 그룹과 현재 블록을 위해 이미 결정되었던 예측 파라미터들이 새로운 블록들의 그룹에 대해 평가될 수 있다. 인코더는 또한 분할 및 병합의 결합에 의해 제공되는 패턴들(의 하위세트)을 직접적으로 체크할 수 있는데, 즉 이미 위에서 언급한 바와 같은 결과적인 형상들을 갖도록 모션 추정과 모드 결정이 행해질 수 있다. 병합 정보는 블록별로 신호전달될 수 있다. 사실상, 병합은 또한 현재 블록에 대한 예측 파라미터들의 추론으로서 해석될 수 있으며, 추론된 예측 파라미터들은 이웃 블록들 중 하나의 블록의 예측 파라미터들과 동일하게 설정된다.

[0069]

SKIP 이외의 다른 모드들의 경우, CBF와 같은 추가적인 플래그들은 어떠한 잔차 신호도 송신되지 않는다는 것을 신호전달할 필요가 있다. H.264의 최신의 비디오 코딩 표준에서는 SKIP/DIRECT 모드들의 두가지 변형들이 존재하는데, 이것들은 화상 레벨, 즉 시간적 직접 모드와 공간적 직접 모드로 선택된다. 양측 직접 모드들은 B 화상들에만 적용가능하다. 시간적 직접 모드에서, 참조 화상 리스트 0에 대한 참조 인덱스는 0과 동일하게 설정되고 참조 화상 리스트 1에 대한 참조 인덱스뿐만이 아니라 양쪽 참조 리스트들에 대한 모션 벡터들은 참조 화상 리스트 1에서의 제1 참조 화상에서 공동 위치한(co-located) 매크로블록의 모션 데이터에 기초하여 유도된다. 시간적 DIRECT 모드는 시간적 공동위치 블록으로부터의 모션 벡터를 이용하고 현재 블록과 공동위치 블록간의 시간적 거리에 따라 모션 벡터를 스케일링한다. 공간적 직접 모드에서, 양쪽 참조 화상 리스트들을 위한 참조 인덱스들 및 모션 벡터들은 기본적으로 공간적 이웃에서의 모션 데이터에 기초하여 추론된다. 참조 인덱스들은 공간적 이웃에서의 대응하는 참조 인덱스들의 최소값으로서 선택되며 각각의 모션 벡터 성분은 공간적 이웃에서의 대응하는 모션 벡터 성분들의 중간값과 동일하게 설정된다. SKIP 모드는 H.264에서 16x16 매크로블록들(P 및 B 화상들)만을 코딩하기 위해 이용될 수 있으며 DIRECT 모드는 16x16 매크로블록들 또는 8x8 하위매크로블록들을 코딩하기 위해 이용될 수 있다. DIRECT 모드와는 대조적으로, 현재 블록에 병합이 적용되는 경우, 모든 예측 파라미터들은 현재 블록과 병합된 블록으로부터 복사될 수 있다. 병합은 또한 앞서 언급한 보다 많은 유연한 파티션화 패턴들을 초래하는 임의적인 블록 크기들에 적용될 수 있고, 여기서는 하나의 패턴의 모든 샘플들이 동일한 예측 파라미터들을 이용하여 예측되고 있다.

[0070]

위와 아래에서 개술한 실시예들의 기본적 아이디어는 병합 플래그와 CBF 플래그를 결합시킴으로써 CBF 플래그들을 송신하는데 필요한 비트 레이트를 감소시키는 것이다. 샘플 세트가 병합을 이용하고 어떠한 잔차 데이터도 송신되지 않는 경우, 하나의 플래그가 이것들을 신호전달하도록 송신된다.

[0071]

이미지 및 비디오 코딩 응용들에서의 보조 정보 레이트를 감소시키기 위해, (직사각형 블록 또는 정방형 블록 또는 임의적으로 형상화된 영역들 또는 임의의 다른 샘플들의 집합체를 나타낼 수 있는) 특정 샘플들의 세트들이 일반적으로 특정 코딩 파라미터들의 세트와 연관된다. 이러한 샘플 세트들 각각에 대해, 코딩 파라미터들은 비트스트림 내에 포함된다. 코딩 파라미터들은 예측 파라미터들을 나타낼 수 있으며, 이 예측 파라미터들은 대응하는 샘플들의 세트가 이미 코딩된 샘플들을 이용하여 어떻게 예측되는지를 규정한다. 샘플 세트들로의 화상의 샘플 어레이들의 파티션화는 구문에 의해 고정될 수 있거나 또는 비트스트림 내의 대응하는 하위분할 정보에 의해 신호전달될 수 있다. 블록에 대한 다수의 파티션화 패턴들이 허용될 수 있다. 샘플 세트들에 대한 코딩 파라미터들은 구문에 의해 주어지는 미리정의된 순서로 송신된다. 이것은 현재 샘플 세트들에 대해 샘플 세트들의 그룹으로 (예컨대, 예측을 목적으로) 하나 이상의 다른 샘플 세트들과 병합된다는 것이 신호전달될 수 있다. 특정 파티션화 패턴들이 다른 파티션화 패턴들 및 대응하는 병합 데이터의 결합에 의해 표현될 수 없도록 하는 방

식으로, 대응하는 병합 정보에 대한 잠재적인 값들의 세트는 활용된 파티션화 패턴에 적응될 수 있다. 샘플 세트들의 그룹에 대한 코딩 파라미터들은 단 한번 송신될 필요가 있다. 예측 파라미터들에 추가적으로, (변환 및 양자화 보조 정보 및 변환 계수 레벨들과 같은) 잔차 파라미터들이 송신될 수 있다. 현재 샘플 세트가 병합되면, 병합 프로세스를 기술하는 보조 정보가 송신된다. 이 보조 정보는 병합 정보라고도 칭해질 것이다. 위와 아래에서 설명한 실시예들은 병합 정보의 신호전달이 (블록에 대해 잔차 데이터가 존재하는지 여부를 규정하는) 코딩된 블록 플래그의 신호전달과 결합되는 개념을 설명한다.

- [0072] 특정 실시예에서, 병합 정보는 소위말하는 결합 플래그, 즉 `mrg_cbf`를 포함하는데, 이것은 현재 샘플 세트가 병합되고 어떠한 잔차 데이터도 송신되지 않는 경우에 1과 동일하다. 이 경우 더 이상 코딩 파라미터들 및 잔차 파라미터들은 송신되지 않는다. 결합 `mrg_cbf` 플래그가 0과 동일하면, 병합이 적용되는지 아닌지의 여부를 표시하는 또다른 플래그가 코딩된다. 뿐만 아니라 어떠한 잔차 파라미터들도 송신되지 않는다는 것을 표시하는 플래그들이 코딩된다. CABAC 및 컨텍스트 적응적 VLC에서, 병합 정보와 관련된 구문 엘리먼트들에 대한 확률 유도 (및 VLC 테이블 스위칭)를 위한 컨텍스트들이 (결합 `mrg_cbf` 플래그와 같은) 이미 송신된 구문 엘리먼트들 및/또는 디코딩된 파라미터들의 함수로서 선택될 수 있다.
- [0073] 바람직한 실시예에서, 결합 `mrg_cbf` 플래그를 포함하는 병합 정보는 코딩 파라미터들(예컨대, 예측 정보 및 하위분할 정보) 이전에 코딩된다.
- [0074] 바람직한 실시예에서, 결합 `mrg_cbf` 플래그를 포함하는 병합 정보는 코딩 파라미터들의 하위세트(예컨대, 예측 정보 및 하위분할 정보) 이후에 코딩된다. 하위분할 정보로부터 초래된, 매 샘플 세트마다, 병합 정보는 코딩될 수 있다.
- [0075] 도 11 내지 도 13과 관련하여 아래에서 추가적으로 설명되는 실시예들에서, `mrg_cbf`는 `skip_flag`이라고 칭해진다. 일반적으로 `mrg_cbf`는 블록 병합과 관련된 스킵의 또다른 버전인 것을 보여주기 위해 `merge_skip`이라고 칭해질 수 있다.
- [0076] 아래의 바람직한 실시예들은 직사각형 및 정방형 블록들을 나타내는 샘플들의 세트에 대해 설명되지만, 단순 방식으로 임의적인 형상의 영역들 또는 다른 샘플들의 집합체들로 확장될 수 있다. 바람직한 실시예들은 병합 방식과 관련된 구문 엘리먼트들과 잔차 데이터의 부재를 표시하는 구문 엘리먼트들의 결합들을 설명한다. 잔차 데이터는 잔차 보조 정보뿐만 아니라 변환 계수 레벨들을 포함할 수 있다. 모든 바람직한 실시예들에 대해, 잔차 데이터의 부재는 코딩된 블록 플래그(coded block flag; CBF)에 의해 규정되지만 다른 수단들 또는 플래그들에 의해서도 표현될 수 있다. CBF가 0과 같다는 것은 어떠한 잔차 데이터도 송신되지 않는 경우와 관련이 있다.
- [0077] 1. 병합 플래그 및 CBF 플래그의 결합
- [0078] 이하에서는, 보조 병합 활성화 플래그는 `mrg`이라고 칭해지지만, 나중에는, 도 11 내지 도 13과 관련하여 `merge_flag`이라고 칭해진다. 마찬가지로 병합 인덱스는 이제 `mrg_idx`이라고 칭해지는 반면에 그 후에는 `merge_idx`가 이용된다.
- [0079] 하나의 구문 엘리먼트를 이용하는 것에 의한 병합 플래그와 CBF 플래그의 잠재적인 결합을 본 섹션에서 설명한다. 아래에서 개술되는 이러한 잠재적인 결합의 설명은 도 1 내지 도 6에서 도시된 임의의 상술한 내용들로 이전될 수 있다.
- [0080] 바람직한 실시예에서는, 세 개까지의 구문 엘리먼트들이 병합 정보 및 CBF를 규정하기 위해 송신된다.
- [0081] 아래에서 `mrg_cbf`이라고 칭해지는 제1 구문 엘리먼트는 현재 샘플 세트가 또다른 샘플 세트와 병합되는지 여부 및 대응하는 모든 CBF들이 0과 동일한지 여부를 규정한다. `mrg_cbf` 구문 엘리먼트는 (병합 없이 상이한 파티션화 패턴에 의해 신호전달될 수 있는 파티션화를 생성시킬 후보들의 잠재적인 제거 이후) 후보 샘플 세트들 중의 유도된 세트가 비어있지 않은 경우에만 코딩될 수 있다. 하지만, 병합 후보들의 리스트는 소실되지 않는 것이 디폴트에 의해 보장될 수 있으며, 적어도 하나 또는 심지어 적어도 두 개의 이용가능한 병합 후보들이 존재가능하다. 본 발명의 바람직한 실시예에서, 후보 샘플 세트들 중의 유도된 세트가 비어있지 않은 경우, `mrg_cbf` 구문 엘리먼트는 다음과 같이 코딩된다.
- [0082] - 현재 블록이 병합되고 모든 성분들(예컨대, 루마 및 두 개의 크로마 성분들)에 대해 CBF가 0과 동일한 경우, `mrg_cbf` 구문 엘리먼트는 1로 설정되어 코딩된다.
- [0083] - 그렇지 않은 경우, `mrg_cbf` 구문 엘리먼트는 0과 동일하게 설정되어 코딩된다.

- [0084] mrg_cbf 구문 엘리먼트들의 값 0과 값 1은 또한 전환될 수 있다.
- [0085] mrg라고 추가적으로 칭해지는 제2 구문 엘리먼트는 현재 샘플 세트가 또다른 샘플 세트와 병합되는지 여부를 규정한다. mrg_cbf 구문 엘리먼트가 1과 동일하면 mrg 구문 엘리먼트는 코딩되지 않고 이 대신에 1과 동일한 것으로 추론된다. (후보 샘플들의 유도된 세트가 비어있기 때문에) mrg_cbf 구문 엘리먼트가 존재하지 않은 경우, mrg 구문 엘리먼트도 존재하지 않지만, 이것은 0과 동일한 것으로 추론된다. 하지만, 병합 후보들의 리스트는 소실되지 않는 것이 디폴트에 의해 보장될 수 있으며, 적어도 하나 또는 심지어 적어도 두 개의 이용가능한 병합 후보들이 존재가능하다.
- [0086] mrg_idx라고 추가적으로 칭해지며, mrg 구문 엘리먼트가 1과 동일한 경우(또는 1과 동일한 것으로 추론되는 경우)에만 코딩되는, 제3 구문 엘리먼트는 후보 샘플 세트들 중 어느 세트들이 병합을 위해 활용되는지를 규정한다. 바람직한 실시예에서, mrg_idx 구문 엘리먼트는 후보 샘플 세트들 중의 유도된 세트가 하나보다 많은 후보 샘플 세트를 포함하는 경우에만 코딩된다. 추가적인 바람직한 실시예에서, mrg_idx 구문 엘리먼트는 후보 샘플 세트들 중의 유도된 세트의 적어도 두 개의 샘플 세트들이 상이한 코딩 파라미터들과 연관되어 있는 경우에만 코딩된다.
- [0087] 파싱(parsing)과 재구축을 디커플링하여 파싱 쓰루풋을 개선시키고 정보 손실 측면에서 보다 견고해지도록 하기 위해 병합 후보 리스트는 심지어 고정될 수 있다는 것이 언급되어야 한다. 보다 정확히 하기 위해, 디커플링은 리스트 엔트리들 및 코드워드들의 고정된 할당을 이용함으로써 보장될 수 있다. 이것은 리스트의 길이를 고정시킬 필요가 없을 것이다. 하지만, 추가적인 후보들을 추가시킴으로써 리스트의 길이를 동시적으로 고정시키는 것은 고정된 (보다 긴) 코드워드들의 코딩 효율성 손실을 보상하게 해준다. 따라서, 이전에 설명한 바와 같이, 병합 인덱스 구문 엘리먼트는 후보들의 리스트가 하나보다 많은 후보를 포함하는 경우에만 송신될 수 있다. 하지만, 이것은 파싱 병합 인덱스 이전에 리스트를 유도하여 이러한 두 개의 프로세스들을 병렬적으로 수행하지 못하도록 할 것을 필요로 할 것이다. 증가된 파싱 쓰루풋을 가능하게 하고 송신 에러들의 관점에서 파싱 프로세스가 보다 견고해지도록 하기 위해, 각각의 인덱스 값을 위한 고정된 코드워드 및 고정된 갯수의 후보들을 이용함으로써 이러한 의존성은 제거되는 것이 가능하다. 이 갯수가 후보 선택에 의해 도달될 수 없는 경우, 보조적인 후보들을 유도하여 리스트를 완성시키는 것이 가능하다. 이러한 추가적인 후보들은 소위말하는 결함 후보들을 포함할 수 있는데, 이 후보들은 리스트에 이미 있는 잠재적으로 상이한 후보들의 모션 파라미터들, 또는 제로 모션 벡터들로부터 구축된다.
- [0088] 바람직한 실시예에서, 샘플들의 세트에 대한 병합 정보는 예측 파라미터들(또는, 보다 일반적으로, 샘플 세트들과 연관된 특정한 코딩 파라미터들)의 하위세트가 송신된 후에 코딩된다. 예측 파라미터들의 하위세트는 모션 파라미터 벡터의 하나 이상의 성분들 또는 하나 이상의 참조 화상 인덱스들 또는 모션 파라미터 벡터의 하나 이상의 성분들 또는 참조 화상 인덱스 등으로 구성될 수 있다.
- [0089] 바람직한 실시예에서, 병합 정보의 mrg_cbf 구문 엘리먼트는 감소된 파티션화 모드들의 세트에 대해서만 코딩된다. 잠재적인 파티션화 모드들의 세트가 도 8에서 제공된다. 바람직한 실시예에서, 이러한 감소된 파티션화 모드들의 세트는 하나로 제한되며 제1 파티션화 모드(도 8에서의 리스트의 좌측 최상단)에 대응한다. 예로서, 블록이 더 이상 파티션화되지 않는 경우에만 mrg_cbf가 코딩된다. 추가적인 예로서, mrg_cbf는 정사각형 블록들에 대해서만 코딩될 수 있다.
- [0090] 또다른 바람직한 실시예에서 병합 정보의 mrg_cbf 구문 엘리먼트는 하나의 파티션화 블록에 대해서만 코딩되며, 이러한 파티션화는 도 8에서 도시된 잠재적인 파티션화 모드들 중 하나, 예컨대 좌측 하단의 네 개의 블록들을 갖는 파티션화 모드이다. 바람직한 실시예에서, 이러한 파티션화 모드들 중 하나의 모드에서 병합된 블록이 하나 이상이 존재하는 경우, (디코딩 순서로) 첫번째로 병합된 블록의 병합 정보는 완성 파티션에 대한 mrg_cbf 구문 엘리먼트를 포함한다. 나중에 디코딩되는 동일 파티션화 모드의 다른 모든 블록들에 대해, 병합 정보는 현재 샘플들의 세트가 또다른 샘플 세트와 병합되는지 아닌지의 여부를 규정하는 mrg 구문 엘리먼트만을 포함한다. 잔차 데이터가 존재하는지 아닌지의 여부의 정보는 첫번째 블록에서 코딩된 mrg_cbf 구문 엘리먼트로부터 추론된다.
- [0091] 본 발명의 추가적인 바람직한 실시예에서, 샘플들의 세트에 대한 병합 정보는 예측 파라미터들(또는, 보다 일반적으로, 샘플 세트들과 연관된 특정한 코딩 파라미터들) 이전에 코딩된다. mrg_cbf, mrg 및 mrg_idx 구문 엘리먼트를 포함한 병합 정보는 위의 첫번째 바람직한 실시예에서 설명된 방식대로 코딩된다. 예측 또는 코딩 파라미터들 및 잔차 파라미터들은 현재 샘플들의 세트가 또다른 샘플들의 세트와 병합되지 않고 성분들 중 적어도 하나에 대한 CBF가 1과 동일하다는 것을 병합 정보가 신호전달하는 경우에만 송신된다. 바람직한 실시예에서,

모든 성분들에 대한 CBF들이 0과 동일하고 현재 블록이 병합되는 것을 mrg_cbf 구문 엘리먼트가 규정하는 경우, 이러한 현재 블록에 대해 병합 정보 이후에 더 이상 신호전달은 필요하지 않을 것이다.

[0092]

본 발명의 또다른 바람직한 실시예에서, 구문 엘리먼트들 mrg_cbf, mrg, 및 mrg_idx는 하나 또는 두 개의 구문 엘리먼트들로서 결합되고 코딩된다. 하나의 바람직한 실시예에서, mrg_cbf 및 mrg는 아래의 경우들 중 임의의 경우를 규정하는 하나의 구문 엘리먼트로 결합된다. (a) 블록이 병합되고 잔차 데이터를 포함하지 않은 경우, (b) 블록이 병합되고 잔차 데이터를 포함하는(또는 잔차 데이터를 포함할 수 있는) 경우, (c) 블록이 병합되지 않은 경우. 또다른 바람직한 실시예에서, 구문 엘리먼트들 mrg 및 mrg_idx는 하나의 구문 엘리먼트로 결합된다. N은 병합 후보들의 갯수이고, 결합된 구문 엘리먼트는 다음의 경우들 중 하나, 즉 블록이 병합되지 않은 경우, 블록이 후보 1과 병합된 경우, 블록이 후보 2와 병합된 경우, ..., 블록이 후보 N과 병합된 경우를 규정한다. 본 발명의 추가적인 바람직한 실시예에서, 구문 엘리먼트들 mrg_cbf, mrg, 및 mrg_idx는 하나의 구문 엘리먼트로 결합되며, 이 구문 엘리먼트는 다음의 경우들 중 하나, 즉 블록이 병합되지 않은 경우, 블록이 후보 1과 병합되되 잔차 데이터를 포함하지 않은 경우, 블록이 후보 2와 병합되되 잔차 데이터를 포함하지 않은 경우, ..., 블록이 후보 N과 병합되되 잔차 데이터를 포함하지 않은 경우, 블록이 후보 1과 병합되되 잔차 데이터를 포함한(또는 포함할 수 있는) 경우, 블록이 후보 2와 병합되되 잔차 데이터를 포함한(또는 포함할 수 있는) 경우, ..., 블록이 후보 N과 병합되되 잔차 데이터를 포함한(또는 포함할 수 있는) 경우를 규정한다(여기서 N은 후보들의 갯수이다). 결합된 구문 엘리먼트들은 가변 길이 코드로 송신될 수 있거나 또는 산술 코딩에 의해 송신될 수 있거나 또는 임의의 특정 바이너리 방식을 이용하여 바이너리 산술 코딩에 의해 송신될 수 있다.

[0093]

2. 병합 플래그 및 CBF 플래그 및 SKIP/DIRECT 모드들의 결합

[0094]

SKIP/DIRECT 모드들은 단지 특정의 또는 모든 블록 크기들 및/또는 블록 형상들에 대해 지원될 수 있다. H.264의 최신의 비디오 코딩 표준에서 규정된 SKIP/DIRECT 모드들의 확장에서, 후보 블록들의 세트가 SKIP/DIRECT 모드들을 위해 이용된다. SKIP와 DIRECT간의 차이는 잔차 파라미터들이 보내지는지 아닌지의 여부이다. SKIP 및 DIRECT의 파라미터들(예컨대, 예측용)은 임의의 대응 후보들로부터 추론될 수 있다. 코딩 파라미터들을 추론하기 위해 어느 후보가 이용되는지를 신호전달하는 후보 인덱스가 코딩된다. (H.264 B-프레임들에서 이용되는 양방향 예측 블록들에서와 같이) 다수의 예측들이 결합되어 현재 블록에 대한 최종적인 예측 신호를 형성하는 경우, 모든 예측은 상이한 후보에 대해 추론될 수 있다. 따라서 모든 예측에 대해 후보 인덱스가 코딩될 수 있다.

[0095]

본 발명의 바람직한 실시예에서, SKIP/DIRECT에 대한 후보 리스트는 병합 모드들을 위한 후보 리스트와는 상이한 후보 블록들을 포함할 수 있다. 도 10에서 예시가 도시된다. 후보 리스트는 다음의 블록들을 포함할 수 있다 (현재 블록은 Xi로 표기된다):

[0096]

- 모션 벡터 (0,0)

[0097]

- (좌측, 위, 모서리간의) 중간값

[0098]

- 좌측 블록 (Li)

[0099]

- 위 블록 (Ai)

[0100]

- 모서리 블록들 (순서에 따라: 우측 위(Ci1), 좌측 아래(Ci2), 좌측 위(Ci3))

[0101]

- 상이하지만 이미 코딩된 화상에 있는 공동위치 블록

[0102]

다음의 실시예들을 설명하기 위해 아래의 표기가 이용된다:

[0103]

- set_mvp_ori는 SKIP/DIRECT 모드를 위해 이용되는 후보들의 세트이다. 이러한 세트는 {중간값, 좌측, 위, 모서리, 공동위치}로 구성되고, 여기서 중간값은 중간값(좌측, 위 및 모서리의 순서화된 세트의 중앙값) 및, 공동위치는 가장 가까이에 있는 참조 프레임(또는 참조 화상 리스트들 중 하나의 리스트에서의 첫번째 참조 화상)에 의해 주어지며, 대응 모션 벡터들은 시간적 거리에 따라 스케일링된다. 예컨대 좌측 블록도 없고, 위 블록도 없고, 모서리 블록도 없는 경우, 0과 동일한 양쪽 성분들을 갖는 모션 벡터가 추가적으로 후보들의 리스트 내로 삽입될 수 있다.

[0104]

- set_mvp_comb는 set_mvp_ori의 하위세트이다.

- [0105] 바람직한 실시예에서, SKIP/DIRECT 모드들 모두와 블록 병합 모드들이 지원된다. SKIP/DIRECT 모드들은 원래의 후보들의 세트, `set_mvp_ori`를 이용한다. 블록 병합 모드와 관련된 병합 정보는 결합된 `mrg_cbf` 구문 엘리먼트를 포함할 수 있다.
- [0106] 또다른 실시예에서, SKIP/DIRECT 모드들 모두와 블록 병합 모드들이 지원되지만, SKIP/DIRECT 모드들은 수정된 후보들의 세트, `set_mvp_comb`를 이용한다. 이러한 수정된 후보들의 세트는 원래 세트 `set_mvp_ori`의 특정 하위 세트일 수 있다. 바람직한 실시예에서, 수정된 후보들의 세트는 모서리 블록들과 공동위치 블록으로 구성된다. 또다른 실시예에서, 수정된 후보들의 세트는 공동위치 블록으로만 구성된다. 추가적인 하위세트들이 가능하다.
- [0107] 또다른 실시예에서, `mrg_cbf` 구문 엘리먼트를 포함한 병합 정보는 SKIP 모드 관련 파라미터들 이전에 코딩된다.
- [0108] 또다른 실시예에서, SKIP 모드 관련 파라미터들은 `mrg_cbf` 구문 엘리먼트를 포함한 병합 정보 이전에 코딩된다.
- [0109] 또다른 실시예에 따르면, DIRECT 모드는 활성화되지 않을 수 있고(심지어 존재하지 않음), 블록 병합은 `mrg_cbf`에 의해 SKIP 모드로 대체된 확장된 후보들의 세트를 갖는다.
- [0110] 바람직한 실시예에서, 블록 병합을 위한 후보 리스트는 상이한 후보 블록들을 포함할 수 있다. 도 10에서 예시가 도시된다. 후보 리스트는 다음의 블록들을 포함할 수 있다(현재 블록은 Xi로 표기된다):
- [0111] • 모션 벡터 (0,0)
 - [0112] • 좌측 블록 (Li)
 - [0113] • 위 블록 (Ai)
 - [0114] • 상이하지만 이미 코딩된 화상에 있는 공동위치 블록
 - [0115] • 모서리 블록들 (순서에 따라: 우측 위(Ci1), 좌측 아래(Ci2), 좌측 위(Ci3))
 - [0116] • 결합된 양방향 예측 후보들
 - [0117] • 스케일링되지 않은 양방향 예측 후보들
- [0118] 메모리 액세스를 절감하기 위해 블록 병합을 위한 후보들의 위치는 인터 예측에서 MVP의 리스트와 동일할 수 있다는 것이 언급되어야 한다.
- [0119] 더 나아가, 리스트는 파싱(parsing)과 재구축을 디커플링하여 파싱 쓰루풋을 개선시키고 정보 손실 측면에서 보다 견고해지도록 하기 위해 위에서 개술된 방식으로 "고정"될 수 있다.
- [0120] 3. CBF의 코딩
- [0121] 바람직한 실시예에서, `mrg_cbf` 구문 엘리먼트가 0과 동일하면(블록이 병합되지 않거나 또는 비제로 잔차 데이터를 포함한다는 것을 신호전달함), 잔차 데이터의 모든 성분들(예컨대, 루마 및 두 개의 크로마 성분들)이 제로인지 아닌지의 여부를 신호전달하는 플래그가 송신된다. `mrg_cbf`가 1과 동일한 경우 이러한 플래그는 송신되지 않는다. 특정 구성에서, 이러한 플래그는 `mrg_cbf`가 0과 동일하고 구문 엘리먼트 `mrg`가 블록이 병합되어 있다는 것을 규정하는 경우에는 송신되지 않는다.
- [0122] 또다른 바람직한 실시예에서, `mrg_cbf` 구문 엘리먼트가 0과 동일하면(블록이 병합되지 않거나 또는 비제로 잔차 데이터를 포함한다는 것을 신호전달함), 해당 성분을 위한 잔차 데이터가 제로인지 아닌지의 여부를 신호전달하는 각각의 성분에 대한 개별적인 구문 엘리먼트가 송신된다.
- [0123] `mrg_cbf`에 대해 상이한 컨텍스트 모델들이 이용될 수 있다.
- [0124] 따라서, 위 실시예들에서는 그 중에서, 화상을 인코딩하기 위한 장치를 설명하며, 본 장치는,
- [0125] 화상을 샘플들의 샘플 세트들로 하위분할하도록 구성된 하위분할기;
- [0126] 샘플 세트들을 하나 이상의 샘플 세트들 각각의 비결합(disjoint) 세트들로 병합하도록 구성된 병합기;
- [0127] 샘플 세트들의 비결합 세트들의 유닛들에서 상기 화상에 걸쳐 변하는 코딩 파라미터들을 이용하여 화상을 인코딩하도록 구성된 인코더로서, 상기 인코더는 화상을 예측하고 미리결정된 샘플 세트들을 위한 예측 잔차

를 인코딩함으로써 화상을 인코딩하도록 구성된 것인, 상기 인코더; 및

- [0128] 각각의 샘플 세트가 또다른 샘플 세트와 함께 비결합 세트들 중의 하나의 비결합 세트로 병합되는지 아닌지의 여부에 대해 신호전달하는, 적어도 샘플 세트들의 하위세트 각각에 대한 하나 이상의 구문 엘리먼트들과 함께, 예측 잔차와 코딩 파라미터들을 비트스트림 내로 삽입하도록 구성된 스트림 생성기를 포함한다.
- [0129] 또한, 화상이 인코딩되어 있는 비트스트림을 디코딩하기 위한 장치가 설명되며, 본 장치는,
- [0130] 화상을 샘플 세트들로 하위분할하도록 구성된 하위분할기;
- [0131] 샘플 세트들을 하나 이상의 샘플 세트들 각각의 비결합(disjoint) 세트들로 병합하도록 구성된 병합기;
- [0132] 샘플 세트들의 비결합 세트들의 유닛들에서 상기 화상에 걸쳐 변하는 코딩 파라미터들을 이용하여 화상을 디코딩하도록 구성된 디코더로서, 상기 디코더는 화상을 예측하고, 미리결정된 샘플 세트들을 위한 예측 잔차를 디코딩하며, 미리결정된 샘플 세트들에 대해, 화상을 예측함으로써 초래된 예측치와 예측 잔차를 결합함으로써 화상을 디코딩하도록 구성된 것인, 상기 디코더; 및
- [0133] 각각의 샘플 세트가 또다른 샘플 세트와 함께 비결합 세트들 중의 하나의 비결합 세트로 병합되는지 아닌지의 여부에 대해 신호전달하는, 적어도 샘플 세트들의 하위세트 각각에 대한 하나 이상의 구문 엘리먼트들과 함께, 비트스트림으로부터 예측 잔차와 코딩 파라미터들을 추출하도록 구성된 추출기를 포함하며, 상기 병합기는 구문 엘리먼트들에 응답하여 병합을 수행하도록 구성된다.
- [0134] 하나 이상의 구문 엘리먼트들의 잠재적인 상태들 중 하나의 상태는 각각의 샘플 세트가 또다른 샘플 세트와 함께 비결합 세트들 중 하나의 비결합 세트로 병합될 것이며 비트스트림 내로 인코딩되고 삽입되는 예측 잔차를 갖지 않는다는 것을 신호전달한다.
- [0135] 추출기는 또한 비트스트림으로부터 하위분할 정보를 추출하도록 구성될 수 있고, 하위분할기는 하위분할 정보에 응답하여 화상을 샘플 세트들로 하위분할하도록 구성된다.
- [0136] 추출기와 병합기는, 예컨대, 샘플 세트 스캔 순서에 따라 샘플 세트들을 거쳐서, 현재 샘플 세트로 순차적으로 진행하고;
- [0137] 비트스트림으로부터 제1 바이너리 구문 엘리먼트(mrg_cbf)를 추출하고;
- [0138] 제1 바이너리 구문 엘리먼트가 제1 바이너리 상태를 취하는 경우, 현재 샘플 세트에 대한 코딩 파라미터들이 비결합 세트들 중 하나의 비결합 세트와 연관된 코딩 파라미터들과 동일한 것으로 추론함으로써 현재 샘플 세트를 이러한 비결합 세트로 병합시키고, 현재 샘플 세트에 대한 예측 잔차의 추출을 스킵하고, 샘플 세트 스캔 순서에서의 다음 샘플 세트로 진행하고;
- [0139] 제1 바이너리 구문 엘리먼트가 제2 바이너리 상태를 취하는 경우, 비트스트림으로부터 제2 구문 엘리먼트(mrg, mrg_idx)를 추출하며;
- [0140] 제2 구문 엘리먼트에 의존하여, 현재 샘플 세트에 대한 코딩 파라미터들이 비결합 세트들 중 하나의 비결합 세트와 연관된 코딩 파라미터들과 동일한 것으로 추론함으로써 현재 샘플 세트를 이러한 비결합 세트로 병합시키거나, 또는 현재 샘플 세트에 대한 예측 잔차와 관련된 적어도 하나의 추가적인 구문 엘리먼트를 추출함과 더불어, 현재 샘플 세트에 대한 코딩 파라미터들의 추출을 수행하도록 구성될 수 있다.
- [0141] 적어도 샘플 세트들의 하위세트 각각에 대한 하나 이상의 구문 엘리먼트들은 또한, 각각의 샘플 세트가 또다른 샘플 세트와 함께 비결합 세트들 중의 하나로 병합되는 경우, 각각의 샘플 세트에 이웃해 있는 미리결정된 후보 샘플 세트들 중의 어느 샘플 세트와 함께, 각각의 샘플 세트가 병합될 것인지를 신호전달할 수 있다.
- [0142] 추출기는, 각각의 샘플 세트가 또다른 샘플 세트와 함께 비결합 세트들 중 하나로 병합될 것이라는 것을 하나 이상의 구문 엘리먼트들이 신호전달하지 않는 경우,
- [0143] 각각의 샘플 세트에 이웃해 있는 미리결정된 후보 샘플 세트들의 추가적인 세트 중 어느 세트로부터, 각각의 샘플 세트에 대한 코딩 파라미터들의 적어도 일부분이 예측될 것인지의 여부에 대해 신호전달하는 하나 이상의 추가적인 구문 엘리먼트들(SKIP/DIRECT 모드)을 비트스트림으로부터 추출하도록 구성될 수 있다.
- [0144] 이 경우, 미리결정된 후보 샘플 세트들의 세트와 미리결정된 후보 샘플 세트들의 추가적인 세트는 미리결정된 후보 샘플 세트들의 세트와 미리결정된 후보 샘플 세트들의 추가적인 세트 중의 소수의 미리결정된 후보 샘플

세트들 각각에 대해 비결합되거나 또는 서로 교차할 수 있다.

- [0145] 추출기는 또한 비트스트림으로부터 하위분할 정보를 추출하도록 구성될 수 있고, 하위분할기는 하위분할 정보에 응답하여 화상을 샘플 세트들로 계층적으로 하위분할하도록 구성되고, 추출기는 화상이 하위분할된 샘플 세트들에 의해 구성된, 부모 샘플 세트의 자손 샘플 세트들을 거쳐서, 현재 자손 샘플 세트에 대해 순차적으로 진행하도록 구성되고,
- [0146] 비트스트림으로부터 제1 바이너리 구문 엘리먼트(mrg_cbf)를 추출하고;
- [0147] 제1 바이너리 구문 엘리먼트가 제1 바이너리 상태를 취하는 경우, 현재 자손 샘플 세트에 대한 코딩 파라미터들이 비결합 세트들 중의 하나의 비결합 세트와 연관된 코딩 파라미터들과 동일한 것으로 추론함으로써 현재 자손 샘플 세트를 이러한 비결합 세트로 병합시키고, 현재 자손 샘플 세트에 대한 예측 잔차의 추출을 스킵하고, 다음 자손 샘플 세트들로 진행하고;
- [0148] 제1 바이너리 구문 엘리먼트가 제2 바이너리 상태를 취하는 경우, 비트스트림으로부터 제2 구문 엘리먼트(mrg, mrg_idx)를 추출하며;
- [0149] 제2 구문 엘리먼트에 의존하여, 현재 자손 샘플 세트에 대한 코딩 파라미터들이 비결합 세트들 중의 하나의 비결합 세트와 연관된 코딩 파라미터들과 동일한 것으로 추론함으로써 현재 자손 샘플 세트를 이러한 비결합 세트로 병합시키거나, 또는 현재 자손 샘플 세트에 대한 예측 잔차와 관련된 적어도 하나의 추가적인 구문 엘리먼트를 추출함과 더불어, 현재 자손 샘플 세트에 대한 코딩 파라미터들의 추출을 수행하며, 그런 후 다음 자손 샘플 세트들로 진행하며,
- [0150] *다음 자손 샘플 세트들에 대해, 현재 자손 샘플 세트의 제1 바이너리 구문 엘리먼트가 제1 바이너리 상태를 취하는 경우 제1 바이너리 구문 엘리먼트의 추출을 스킵하고 이 대신에 제2 구문 엘리먼트를 추출하는 것을 시작하고, 현재 자손 샘플 세트의 제1 바이너리 구문 엘리먼트가 제2 바이너리 상태를 취하는 경우 제1 바이너리 구문 엘리먼트를 추출하도록 구성될 수 있다.
- [0151] 예를 들어, 부모 샘플 세트(CU)가 두 개의 자손 샘플 세트들(PU)로 분할되는 것을 가정하자. 그런 후, 제1 바이너리 구문 엘리먼트(merge_cbf)가 제1 바이너리 상태를 갖는 경우, 1) 제1 PU에 대해, 제1 PU는 병합을 이용하고 제1 및 제2 PU(전체 CU)는 비트스트림에서 잔차 데이터를 갖지 않으며, 2) 제2 PU에 대해 제2 바이너리 구문 엘리먼트(merge_flag, merge_idx)가 신호전달된다. 하지만, 제1 PU에 대한 제1 바이너리 구문 엘리먼트가 제2 바이너리 상태를 갖는 경우, 1) 제1 PU에 대해, 제2 바이너리 구문 엘리먼트(merge_flag, merge_idx)가 신호전달되고, 잔차 데이터는 비트스트림 내에도 존재하는 반면에, 2) 제2 PU에 대해, 제1 바이너리 구문 엘리먼트(merge_cbf)가 신호전달된다. 따라서, merge_cbf가 이전의 모든 자손 샘플 세트들에 대한 제2 바이너리 상태에 있는 경우, merge_cbf가 또한 PU 레벨로, 즉 계승한 자손 샘플 세트들에 대해 신호전달될 수 있다. merge_cbf가 계승한 자손 샘플 세트에 대한 제1 바이너리 상태에 있는 경우, 이 자손 샘플 세트를 뒤따르는 모든 자손 샘플 세트들은 비트스트림에서 잔차 데이터를 갖지 않는다. 예를 들어, CU가 예컨대 4개의 PU들로 분할되는 경우, 코딩 순서로 세번째의 PU와 네번째의 PU는 비트스트림에서 잔차 데이터를 갖지 않지만 첫번째 PU는 잔차 데이터를 갖거나 또는 가질 수 있다는 것을 의미하는 제2 PU에 대한 제1 바이너리 상태에 merge_cbf가 있는 것이 가능하다.
- [0152] 제1 및 제2 바이너리 구문 엘리먼트들은 컨텍스트 적응 가변 길이 코딩 또는 컨텍스트 적응(바이너리) 산술 코딩을 이용하여 코딩될 수 있고, 구문 엘리먼트들을 코딩하기 위한 컨텍스트들은 이미 코딩된 블록들에서의 이러한 구문 엘리먼트들에 대한 값들에 기초하여 유도된다.
- [0153] 다른 바람직한 실시예들에서 설명된 바와 같이, 구문 엘리먼트 merge_idx는 후보들의 리스트가 하나보다 많은 후보를 포함하는 경우에만 송신될 수 있다. 이것은 파싱 병합 인덱스 이전에 리스트를 유도하여 이러한 두 개의 프로세스들을 병렬적으로 수행하지 못하도록 할 것을 필요로 한다. 증가된 파싱 쓰루풋을 가능하게 하고 송신 에러들의 관점에서 파싱 프로세스가 보다 견고해지도록 하기 위해, 각각의 인덱스 값을 위한 고정된 코드워드 및 고정된 갯수의 후보들을 이용함으로써 이러한 의존성은 제거되는 것이 가능하다. 이 갯수가 후보 선택에 의해 도달될 수 없는 경우, 보조적인 후보들을 유도하여 리스트를 완성시키는 것이 가능하다. 이러한 추가적인 후보들은 소위말하는 결함 후보들을 포함할 수 있는데, 이 후보들은 리스트에 이미 있는 잠재적으로 상이한 후보들에 대한 모션 파라미터들, 또는 제로 모션 벡터들로부터 구축된다.

- [0154] 또다른 바람직한 실시예에서, 후보 세트의 블록들 중 어느 블록을 신호전달할지를 위한 구문은 인코더와 디코더에서 동시에적으로 적용된다. 예를 들어, 병합을 위한 블록들의 세가지 선택이 주어지는 경우, 이러한 세가지 선택들은 구문에서만 제공되며 엔트로피 코딩용으로 간주된다. 다른 모든 선택들에 대한 가능성들은 0인 것으로 간주되고 엔트로피 코딩은 인코더와 디코더에서 동시에적으로 조정된다.
- [0155] 병합 프로세스의 결과로서 추론되는 예측 파라미터들은 블록과 연관된 예측 파라미터들의 완전한 세트를 나타낼 수 있거나 또는 이것들은 이러한 예측 파라미터들의 하위세트(예컨대, 멀티 추정 예측이 이용되는 블록의 하나의 추정을 위한 예측 파라미터들)를 나타낼 수 있다.
- [0156] 바람직한 실시예에서, 병합 정보와 관련된 구문 엘리먼트들은 컨텍스트 모델링을 이용하여 엔트로피 코딩된다.
- [0157] 위에서 개술한 실시예들을 특정 구문으로 이전시키는 한가지 방법을 다음 도면들을 참조하여 아래에서 설명한다. 특히, 도 11 내지 도 13은 위에서 개술한 실시예들의 장점을 취하는 구문의 상이한 부분들을 도시한다. 특히, 아래에서 개술되는 실시예에 따르면, 화상(20)은 제일먼저 코딩 트리 블록들로 분할되며, 그 화상 콘텐츠는 도 11에서 도시된 구문 `coding_tree`를 이용하여 코딩된다. 도시된 바와 같이, 예컨대 컨텍스트 적응 바이너리 산술 코딩 또는 또다른 특정 엔트로피 코딩 모드와 관련이 있는 `entropy_coding_mode_flag=1`인 경우, 참조부호(400)에서 현재 코딩 트리 블록의 쿼드 트리 하위분할이 `split_coding_unit_flag`라고 칭해지는 플래그들에 의해 구문 부분 `coding_tree` 내에서 신호전달된다. 도 11에서 도시된 바와 같이, 이후에 설명하는 실시예에 따라, 트리 루트 블록은 `split_coding_unit_flag`에 의해 신호전달된 바와 같이 도 7a에서 도시된 깊이 우선 탐색 순서로 하위분할된다. 리프 노드에 도달할 때 마다, 리프 노드는 구문 함수 `coding_unit`을 이용하여 곧바로 코딩되는 코딩 유닛을 나타낸다. 이것은 현재 `split_coding_unit_flag`가 설정되어 있는지 아닌지의 여부에 대해 체크하는 참조부호 402에서의 if절을 검토할 때 도 11로부터 살펴볼 수 있다. `split_coding_unit_flag`가 설정된 경우, 함수 `coding_tree`는 회귀적으로 호출되어, 인코더와 디코더에서 각각 추가적인 `split_coding_unit_flag`의 추가적인 송신/추출을 야기시킨다. `split_coding_unit_flag`가 설정되지 않은 경우, 즉 `split_coding_unit_flag=0`인 경우, 도 7a의 트리 루트 블록(200)의 현재 하위블록은 리프 블록이고, 이러한 코딩 유닛을 코딩하기 위해, 도 10의 함수 `coding_unit`이 참조부호 404에서 호출된다.
- [0158] 현재 설명하는 실시예에서, 인터 예측 모드가 이용가능한 화상들에 대해 병합이 단순히 이용가능하도록 해주는 위에서 언급한 옵션이 이용된다. 즉, 인트라 코딩된 슬라이스들/화상들은 병합을 더 이상 이용하지 않는다. 이것은 도 12로부터 살펴볼 수 있는데, 도 12에서는 슬라이스 유형이 인트라 화상 슬라이스 유형과 동일하지 않는 경우, 즉 현재 코딩 유닛이 속해 있는 현재 슬라이스가 파티션들이 인트라코딩되도록 해주는 경우에만 플래그 `skip_flag`가 참조부호 406에서 송신된다. 본 실시예에 따르면, 병합은 인터 예측과 관련된 예측 파라미터들과 단순히 관련이 있다. 본 실시예에 따르면, `skip_flag`가 전체 코딩 유닛(40)에 대해 신호전달되고, `skip_flag`가 1과 동일하면, 이 플래그 값은 동시에적으로 디코더에게,
- [0159] 1) 현재 코딩 유닛에 대한 파티션화 모드가 현재 코딩 유닛이 파티션화되지 않도록 해주는 비파티션화 모드이고 해당 코딩 유닛의 파티션만을 나타낸다는 것,
- [0160] 2) 현재 코딩 유닛/파티션이 인트라 코딩되고, 즉 인트라 코딩 모드에 할당되는 것,
- [0161] 3) 현재 코딩 유닛/파티션이 병합처리되는 것, 및
- [0162] 4) 현재 코딩 유닛/파티션이 스킵 모드처리되고, 즉 스킵 모드를 활성화시킨다는 것을 신호전달한다.
- [0163] 따라서, `skip_flag`가 설정되면, 현재 코딩 유닛이 예측 유닛이 되는 것을 표시하는 함수 `prediction_unit`이 참조부호 408에서 호출된다. 하지만, 이것은 병합 옵션에 대한 스위칭을 위한 가능성만은 아니다. 이보다는, 전체 코딩 유닛과 관련된 `skip_flag`가 참조부호 406에서 설정되지 않으면, 비(non)인트라 화상 슬라이스의 코딩 유닛의 예측 유형은 구문 엘리먼트 `pred_type`에 의해 참조부호 410에서 신호전달되며, 이에 의존하여, 현재 코딩 유닛이 더이상 파티션화되지 않는 경우 예컨대 참조부호 412에서 현재 코딩 유닛의 임의의 파티션을 위한 함수 `prediction_unit`을 호출한다. 도 12에서는 단지 네 개의 상이한 파티션화 옵션들이 이용가능한 것으로 도시되지만, 도 8에서 도시된 다른 파티션화 옵션들도 이용가능할 수 있다. 또다른 가능성은 파티션화 옵션 `PART_NxN`이 이용가능하지 않지만, 다른 것들은 이용가능하다는 것일 것이다. 도 8에서 도시된 파티션화 옵션들에 대해 도 12에서 이용된 파티션화 모드들을 위한 명칭들간의 연관은 개별적인 파티션화 옵션들 아래의 각각의 아래첨자들에 의해 도 8에서 표시된다. 예측 유형 구문 엘리먼트 `pred_type`는 예측 모드, 즉 인트라 또는 인트라 코딩을 신호전달하는 것뿐만이 아니라, 인트라 코딩 모드의 경우에서의 파티션화를 신호전달한다는 것을 유념하길 바란다. 인트라 코딩 모드 경우를 추가적으로 설명한다. 함수 `prediction_unit`이 파티션들(50, 60)과 같은 각각의 파티션

마다 위에서 언급한 코딩 순서로 호출된다. 함수 prediction_unit은 참조부호 414에서 skip_flag를 체크하는 것으로 시작한다. skip_flag가 설정되면, 참조부호 416에서 merge_idx가 필연적으로 뒤따른다. 단계 414에서의 체크는, 참조부호 406에서 신호전달된 전체 코딩 유닛과 관련된 skip_flag가 설정되었는지 아닌지의 여부에 대한 체크를 위한 것이다. skip_flag가 설정되지 않은 경우, 참조부호 418에서 merge_flag가 다시 신호전달되고, merge_flag가 설정되면, 참조부호 420에서, 현재 파티션을 위한 병합 후보를 나타내는 merge_idx가 뒤따른다. 다시, 현재 코딩 유닛의 현재 예측 모드가 인터 예측 모드(참조부호 422 참조)인 경우에서만 merge_flag는 참조부호 418에서 현재 파티션을 위해 신호전달된다. 즉, skip_flag가 설정되지 않은 경우, 예측 모드는 참조부호 410에서 pred_type을 통해 신호전달되고, 이에 따라, 각각의 예측 유닛에 대해, 인터 코딩 모드가 활성화되어 있다(참조부호 422 참조)라는 것을 pred_type이 신호전달하는 경우, 병합이 병합 인덱스 merge_idx에 의해 각각의 파티션을 위해 활성화되면, 병합 특유적 플래그, 즉 merge_flag가 각각의 뒤따르는 파티션에 대해 개별적으로 송신된다.

[0164] 도 13으로부터 살펴볼 수 있는 바와 같이, 참조부호 424에서의 현재 예측 유닛용으로 이용하는 예측 파라미터들의 송신은, 본 실시예에 따라, 병합이 본 예측 유닛을 위해 이용되지 않은 경우에만 수행되는데, 즉 그 이유는 병합은 각각의 파티션의 각각의 merge_flag에 의해서도 skip_flag에 의해서도 활성화되지 않기 때문이다.

[0165] 이미 위에서 언급한 바와 같이, skip_flag = 1은 잔차 데이터가 송신되지 않는다는 것을 동시에 신호전달한다. 이것은 현재 코딩 유닛에 대한 도 12에서의 참조부호 426에서 잔차 데이터의 송신은, 그 송신 직후의 skip_flag의 상태를 체크하는 if절(428)의 else 옵션 내에서 이러한 잔차 데이터 송신이라는 사실로부터 유도될 수 있는, skip_flag가 0과 동일한 경우에만 발생한다라는 사실로부터 유도될 수 있다.

[0166] 지금까지, 도 11 내지 도 13의 실시예는 entropy_coding_mode_flag가 1과 동일하다는 가정하에서만 설명되어 왔다. 하지만, 도 11 내지 도 13의 실시예들은 또한 entropy_coding_mode_flag = 0인 경우의 위에서 개술된 실시예들의 실시예를 포함하며, 이 경우 예컨대 가변 길이 코딩, 및 보다 정확히 하기 위해, 예컨대 콘텍스트 적응 가변 길이 코딩과 같은 구문 엘리먼트들을 엔트로피 인코딩하기 위해 또다른 엔트로피 코딩 모드가 이용된다. 특히, 한편으로 병합과 다른 한편으로 스킵 모드의 활성화를 동시에 신호전달하는 가능성은 위에서 개술된 대안구성을 따르며, 이 대안구성에 따르면, 공동의 신호전달 상태는 각각의 구문 엘리먼트의 두 개보다 많은 상태들 중의 단하나의 상태이다. 이것을 이제부터 보다 자세하게 설명한다. 하지만, 엔트로피 코딩 모드들간의 스위칭하는 가능성은 택일적이며, 이에 따라 대안적인 실시예들은 두 개의 엔트로피 코딩 모드들 중의 하나를 단순히 허용함으로써 도 11 내지 도 13으로부터 손쉽게 유도될 수 있다는 것을 강조해둔다.

[0167] 예컨대, 도 11을 참조하라. entropy_coding_mode_flag가 0과 동일하고 slice_type 구문 엘리먼트가 현재 트리 루트 블록이 인터 코딩 슬라이스에 속한다는 것, 즉 인터 코딩 모드가 이용가능하다는 것을 신호전달하는 경우, 구문 엘리먼트 cu_split_pred_part_mode가 참조부호 430에서 송신되고, 이 구문 엘리먼트는 그 명칭에서 나타난 바와 같이, 각각의 파티션화 정보와 함께 현재 코딩 유닛의 추가적인 하위분할, 스킵 모드의 활성화 또는 비활성화, 병합의 활성화 또는 비활성화 및 예측 모드에 대한 정보를 신호전달한다. [표 1]을 참조하라.

[0168] [표 1]

cu_split_pred_part_mode	split_coding_unit_flag	skip_flag	merge_flag	PredMode	PartMode
0	1	-	-	-	-
1	0	1	-	MODE_SKIP	PART_2Nx2N
2	0	0	1	MODE_INTER	PART_2Nx2N
3	0	0	0	MODE_INTER	PART_2Nx2N
4	0	-	-	MODE_INTER	PART_2NxN
5	0	-	-	MODE_INTER	PART_Nx2N
6	0	-	-	MODE_INTRA	PART_2Nx2N

[0169] [표 1]은 현재 코딩 유닛이 현재 트리 루트 블록의 쿼드 트리 하위분할에서의 가장작은 크기가 아닌 크기를 갖는 경우, 구문 엘리먼트들 cu_split_pred_part_mode의 잠재적인 상태들의 시그니피칸스(significance)를 규정한다. 잠재적 상태들은 [표 1]의 좌측끝에 있는 열에서 나열된다. [표 1]은 현재 코딩 유닛이 가장 작은 크기를 갖지 않는 경우를 가리키므로, cu_split_pred_part_mode의 상태, 즉 상태 0이 존재하는데, 이것은 현재 코딩 유닛이 실제 코딩 유닛은 아니지만 네 개의 추가적인 유닛들로 하위분할되어야 한다는 것을 신호전달하며, 그 후

네 개의 유닛들은 참조부호 432에서 다시 함수 coding_tree를 호출함으로써 개술된 바와 같은 깊이 우선 탐색 순서로 탐색된다. 즉, cu_split_pred_part_mode = 0은 현재 트리 루트 블록의 현재 쿼드 트리 하위분할 유닛이 네 개의 추가적인 보다 작은 유닛들로 다시 하위분할될 것, 즉 split_coding_unit_flag = 1을 신호전달한다. 하지만, cu_split_pred_part_mode가 임의의 다른 잠재적 상태를 취하는 경우, split_coding_unit_flag = 0이고, 현재 유닛은 현재 트리 루트 블록의 리프 블록, 즉 코딩 유닛을 형성한다. 이 경우, cu_split_pred_part_mode의 나머지 잠재적 상태들 중 하나는, 현재 코딩 유닛의 추가적인 파티션이 발생하지 않는다는 것, 즉 PART_2Nx2N이 파티션화 모드로서 선택되는 것을 동시에 신호전달하면서, 현재 코딩 유닛이 병합되고 스킵 모드를 활성화시키는 것([표 1]의 세번째 열에서 skip_flag를 1과 동일하게 한 것으로서 표시됨)을 동시에 신호전달하는, 위에서 설명한 공동의 신호전달 상태를 표현한다. cu_split_pred_part_mode는 또한 스킵 모드가 비활성화되는 것과 함께 병합의 활성화를 신호전달하는 잠재적 상태를 갖는다. 이것은 skip_flag = 0이면서 merge_flag = 1이고 어떠한 파티션화 모드도 활성화되지 않은 것, 즉 PART_2Nx2N에 대응하는 잠재적 상태 2이다. 즉, 이 경우 merge_flag는 prediction_unit 구문 내에서보다는 사전에 신호전달된다. cu_split_pred_part_mode의 나머지 잠재적 상태들에서, 다른 파티션화 모드들과 함께 인터 예측 모드가 신호전달되고, 이러한 파티션화 모드들은 하나보다 많은 파티션으로 현재 코딩 유닛을 파티션화한다.

[표 2]

cu_split_pred_part_mode	split_coding_unit_flag	skip_flag	merge_flag	PredMode	PartMode
0	0	1	-	MODE_SKIP	PART_2Nx2N
1	0	0	1	MODE_INTER	PART_2Nx2N
2	0	0	0	MODE_INTER	PART_2Nx2N
3	0	-	-	MODE_INTER	PART_2NxN
4	0	-	-	MODE_INTER	PART_Nx2N
5 (탈출 심볼)	0	-	-	MODE_INTRA	PART_2Nx2N
				MODE_INTRA	PART_NxN
				MODE_INTER	PART_NxN

[표 2]는 현재 코딩 유닛이 현재 트리 루트 블록의 쿼드 트리 하위분할에 따라 가능할 수 있는 가장 작은 크기를 갖는 경우, cu_split_pred_part_mode의 잠재적인 상태들의 시그니피칸스(significance) 또는 시맨틱스(semantic)를 도시한다. 이 경우, cu_split_pred_part_mode의 모든 잠재적 상태들은 split_coding_unit_flag = 0에 따라 추가적인 하위분할이 없다는 것에 대응한다. 하지만, 잠재적 상태 0은 skip_flag = 1을 신호전달하는데, 즉 병합이 활성화되고 스킵 모드가 활성화되는 것을 동시에 신호전달한다. 게다가, 이것은 어떠한 파티션화도 발생하지 않는다는 것, 즉 파티션화 모드 PART_2Nx2N을 신호전달한다. 잠재적 상태 1은 [표 1]의 잠재적 상태 2에 대응하고, [표 1]의 잠재적 상태 3에 대응하는 [표 2]의 잠재적 상태 2에도 동일 내용이 적용된다.

도 11 내지 도 13의 실시예의 위 설명은 대부분의 기능 및 시맨틱스를 이미 설명하였지만, 아래에서는 몇몇의 추가적인 정보가 제공된다.

skip_flag[x0][y0]가 1과 동일하다는 것은 현재 코딩 유닛(도면들에서 참조부호 40 참조)에 대해, P 또는 B 슬라이스를 디코딩할 때, 모션 벡터 예측기 인덱스들(merge_idx)을 제외한 구문 엘리먼트들은 더 이상 skip_flag[x0][y0] 이후에 파싱되지 않는다는 것을 규정한다. skip_flag[x0][y0]가 0과 동일하다는 것은 코딩 유닛이 스킵되지 않은 것을 규정한다. 어레이 인덱스들 x0, y0은 화상(도면들에서 참조부호 20 참조)의 좌측 상단 루마 샘플에 대한 고려된 코딩 유닛의 좌측 상단 루마 샘플의 위치(x0, y0)를 규정한다.

skip_flag[x0][y0]가 존재하지 않은 경우, 이것은 0과 동일한 것으로 추론된다.

상술한 바와 같이 skip_flag[x0][y0]가 1과 동일하면,

- PredMode는 MODE_SKIP와 동일한 것으로 추론된다

- PartMode는 PART_2Nx2N와 동일한 것으로 추론된다

cu_split_pred_part_mode[x0][y0]는 split_coding_unit_flag를 규정하며 코딩 유닛이 분할되지 않을 때 skip_flag[x0][y0], 코딩 유닛의 PredMode, PartMode, merge_flag[x0][y0]을 규정한다. 어레이 인덱스들 x0, y0은 화상의 좌측 상단 루마 샘플에 대한 코딩 유닛의 좌측 상단 루마 샘플의 위치(x0, y0)를 규정한다.

다.

- [0181] **merge_flag**[x0][y0]는 현재 예측 유닛(도면들에서 참조번호 50 및 60 참조, 즉 코딩 유닛(40) 내의 파티션)에 대한 인터 예측 파라미터들이 이웃해 있는 인터 예측된 파티션으로부터 추론되는지 여부를 규정한다. 어레이 인덱스들 x0, y0은 화상의 좌측 상단 루마 샘플에 대한 고려된 예측 블록의 좌측 상단 루마 샘플의 위치 (x0, y0)를 규정한다.
- [0182] **merge_idx**[x0][y0]는 병합 후보 리스트의 병합 후보 인덱스를 규정하며, 여기서 x0, y0은 화상의 좌측 상단 루마 샘플에 대한 고려된 예측 블록의 좌측 상단 루마 샘플의 위치 (x0, y0)를 규정한다.
- [0183] 도 11 내지 도 13의 위 설명에서는 구체적으로 나타나지 않았지만, 병합 후보들 또는 병합 후보들의 리스트는 이 실시예에서 공간적으로 이웃하는 예측 유닛/파티션들의 예측 파라미터들 또는 코딩 파라미터들을 예시적으로 이용하여 결정될 뿐만이 아니라, 후보들의 리스트는 또한 시간적으로 이웃해 있고 이전에 코딩된 화상들의 시간적으로 이웃해 있는 파티션들의 예측 파라미터들을 이용하여 형성된다. 게다가, 공간적으로 및/또는 시간적으로 이웃해 있는 예측 유닛들/파티션들의 예측 파라미터들의 결합들은 병합 후보들의 리스트 내에 포함되어 이용된다. 당연히, 그 하위세트도 이용될 수 있다. 특히, 도 14는 공간적 이웃들, 즉 공간적으로 이웃해 있는 파티션들 또는 예측 유닛들을 결정하는 한가지 가능성을 도시한다. 도 14는 예시적으로 예측 유닛 또는 파티션(60)과 픽셀들 B₀ 내지 B₂, A₀ 및 A₁을 도시하며, 이 픽셀들은 파티션(60)의 경계부(500)에 바로 인접하여 위치하는데, 즉 B₂는 파티션(60)의 좌측 상단 픽셀에 대각선 방향으로 인접해 있고, B₁은 파티션(60)의 우측 상단 픽셀의 수직방향 윗쪽에서 픽셀에 인접하여 위치하고, B₀은 파티션(60)의 우측 상단 픽셀에 대각선 방향으로 위치해 있고, A₁은 파티션(60)의 좌측 바닥 픽셀의 좌측에 수평한 방향으로 픽셀에 인접하여 위치하며, A₀은 파티션(60)의 좌측 바닥 픽셀에 대각선 방향으로 위치해 있다. 픽셀들 B₀ 내지 B₂, A₀ 및 A₁ 중 적어도 하나를 포함하는 파티션은 공간적 이웃을 형성하며 그 이웃의 예측 파라미터들은 병합 후보를 형성한다.
- [0184] 또한 이용가능한 것인 또다른 파티션화 모드를 야기시킬 이러한 후보들의 앞서 언급한 택일적 제거를 수행하기 위해, 다음의 함수들이 이용될 수 있다:
- [0185] 특히, 후보 N, 즉 픽셀 N=(B₀, B₁, B₂, A₀, A₁), 즉 위치(xN, yN)를 커버하는 예측 유닛/파티션으로부터 유래된 코딩/예측 파라미터들은 아래의 조건들 중 임의의 것이 참인 경우 후보 리스트로부터 제거된다(파티션화 모드 PartMode 및 코딩 유닛 내의 각각의 파티션을 인덱싱하는 대응하는 파티션화 인덱스 PartIdx에 대해서는 도 8을 참조하라):
- [0186] - 현재 예측 유닛의 PartMode는 PART_2NxN이고 PartIdx는 1과 동일하며 루마 위치(xP, yP - 1)(PartIdx = 0) 및 루마 위치(xN, yN)(Cand. N)를 커버하는 예측 유닛들은 동일한 모션 파라미터들을 갖는다:
- [0187]
$$mvLX[xP, yP - 1] == mvLX[xN, yN]$$
- [0188]
$$*refIdxLX[xP, yP - 1] == refIdxLX[xN, yN]$$
- [0189]
$$predFlagLX[xP, yP - 1] == predFlagLX[xN, yN]$$
- [0190] - 현재 예측 유닛의 PartMode는 PART_Nx2N이고 PartIdx는 1과 동일하며 루마 위치(xP - 1, yP)(PartIdx = 0) 및 루마 위치(xN, yN)(Cand. N)를 커버하는 예측 유닛들은 동일한 모션 파라미터들을 갖는다:
- [0191]
$$mvLX[xP - 1, yP] == mvLX[xN, yN]$$
- [0192]
$$refIdxLX[xP - 1, yP] == refIdxLX[xN, yN]$$
- [0193]
$$predFlagLX[xP - 1, yP] == predFlagLX[xN, yN]$$
- [0194] - 현재 예측 유닛의 PartMode는 PART_NxN이고 PartIdx는 3과 동일하며 루마 위치(xP - 1, yP)(PartIdx = 2) 및 루마 위치(xP - 1, yP - 1)(PartIdx = 0)를 커버하는 예측 유닛들은 동일한 모션 파라미터들을 갖는다:
- [0195]
$$mvLX[xP - 1, yP] == mvLX[xP - 1, yP - 1]$$
- [0196]
$$refIdxLX[xP - 1, yP] == refIdxLX[xP - 1, yP - 1]$$

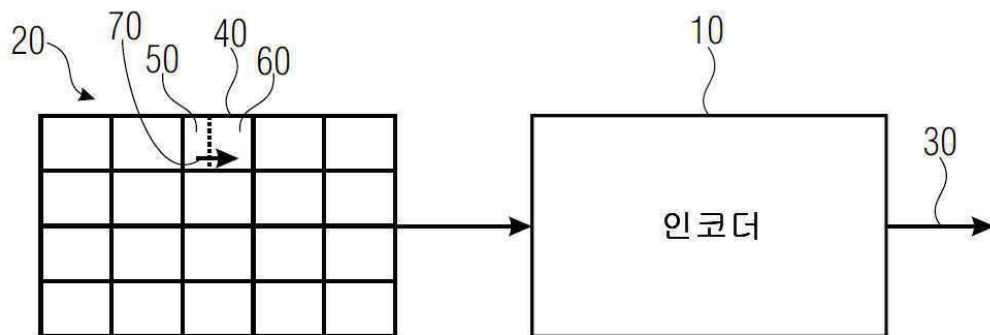
- [0197] $\text{predFlagLX}[xP - 1, yP] == \text{predFlagLX}[xP - 1, yP - 1]$
- [0198] 그리고, 루마 위치($xP, yP - 1$)($\text{PartIdx} = 1$) 및 루마 위치(xN, yN)(Cand. N)를 커버하는 예측 유닛들은 동일한 모션 파라미터들을 갖는다:
- [0199] $\text{mvLX}[xP, yP - 1] == \text{mvLX}[xN, yN]$
- [0200] $\text{refIdxLX}[xP, yP - 1] == \text{refIdxLX}[xN, yN]$
- [0201] $\text{predFlagLX}[xP, yP - 1] == \text{predFlagLX}[xN, yN]$
- [0202] - 현재 예측 유닛의 PartMode 는 PART_NxN 이고 PartIdx 는 3과 동일하며 루마 위치($xP, yP - 1$)($\text{PartIdx} = 1$) 및 루마 위치($xP - 1, yP - 1$)($\text{PartIdx} = 0$)를 커버하는 예측 유닛들은 동일한 모션 파라미터들을 갖는다:
- [0203] $\text{mvLX}[xP, yP - 1] == \text{mvLX}[xP - 1, yP - 1]$
- [0204] $\text{refIdxLX}[xP, yP - 1] == \text{refIdxLX}[xP - 1, yP - 1]$
- [0205] $\text{predFlagLX}[xP, yP - 1] == \text{predFlagLX}[xP - 1, yP - 1]$
- [0206] 그리고, 루마 위치($xP - 1, yP$)($\text{PartIdx} = 2$) 및 루마 위치(xN, yN)(Cand. N)를 커버하는 예측 유닛들은 동일한 모션 파라미터들을 갖는다:
- [0207] $\text{mvLX}[xP - 1, yP] == \text{mvLX}[xN, yN]$
- [0208] $\text{refIdxLX}[xP - 1, yP] == \text{refIdxLX}[xN, yN]$
- [0209] 이와 관련하여, 포지션 또는 위치(xP, yP)는 현재 파티션/예측 유닛의 최상위 픽셀을 표시한다는 것을 유념하라. 즉, 첫번째 항목에 따르면, 이웃해 있는 예측 유닛들, 즉 예측 유닛 N의 각각의 코딩 파라미터들을 직접 채용함으로써 유도되었던 모든 코딩 파라미터 후보들이 체크된다. 하지만, 다른 추가적인 코딩 파라미터 후보들이 구문에 의해 또한 지원되는 또다른 파티션화 패턴을 획득하는 것을 초래시킬 때 출현하는 각각의 예측 유닛의 코딩 파라미터들과 동일한지 여부에 대해 이와 동일한 방식으로 이러한 다른 추가적인 코딩 파라미터 후보들이 체크될 수 있다. 방금 설명한 실시예들에 따르면, 코딩 파라미터들의 동등성은 모션 벡터, 즉 mvLX , 참조 인덱스, 즉 refIdxLX , 및 예측 플래그 predFlagLX 의 동등성의 체크를 포함하며, 예측 플래그 predFlagLX 는 참조 리스트 X(X 는 0 또는 1임)와 연관된 파라미터들, 즉 모션 벡터와 참조 인덱스가 인터 예측에서 이용되는지를 표시한다.
- [0210] 방금 설명한 이웃 예측 유닛들/파티션들의 코딩 파라미터 후보들의 제거를 위한 가능성은 또한 도 8의 우측 절반에서 도시된 비대칭적 파티션화 모드를 지원하는 경우에도 적용가능할 것이라는 것을 유념하라. 이 경우, 모드 PART_2NxN 은 모든 수평적 하위분할 모드들을 표현할 수 있고, PART_Nx2N 은 모든 수직적 하위분할 모드들에 대응할 수 있다. 더 나아가, 모드 PART_NxN 은 지원된 파티션화 모드들 또는 파티션화 패턴들로부터 배제될 수 있고, 이 경우, 단지 첫번째 두 개의 제거 체크들만이 수행되어야 할 것이다.
- [0211] 도 11 내지 도 14의 실시예와 관련하여, 후보들의 리스트로부터 인트라 예측 파티션들이 배제되는 것이 가능하다는 것, 즉 이러한 파티션들의 코딩 파라미터들이 당연히 후보들의 리스트 내에 포함되지 않는 것이 가능하다는 것을 또한 유념해야 한다.
- [0212] 더 나아가, skip_flag , merge_flag 및 merge_idx 각각에 대해 세 개의 콘텍스트들이 이용될 수 있다는 것을 유념해둔다.
- [0213] 비록 몇몇 양태들은 장치의 관점에서 설명되었지만, 이러한 양태들은 또한 대응 방법의 설명을 나타낸다는 것이 명백하며, 여기서 블록 또는 디바이스는 방법 단계 또는 방법 단계의 특징에 대응한다. 마찬가지로, 방법 단계의 관점에서 설명된 양태들은 또한 대응하는 장치의 대응하는 블록 또는 아이템 또는 특징의 설명을 나타낸다. 방법 단계들 모두 또는 그 일부는 예컨대, 마이크로프로세서, 프로그램가능 컴퓨터 또는 전자 회로와 같은 하드웨어 장치에 의해(또는 이것을 이용하여) 실행될 수 있다. 몇몇 실시예들에서, 가장 중요한 방법 단계들 중의 몇몇의 하나 이상의 방법 단계들은 이러한 장치에 의해 실행될 수 있다.
- [0214] 일정한 구현 요건에 따라, 본 발명의 실시예들은 하드웨어나 소프트웨어로 구현될 수 있다. 이러한 구현은 전자적으로 판독가능한 제어 신호들이 저장되어 있으며, 각각의 방법이 수행되도록 프로그램가능한 컴퓨터 시스템과

협동하는(또는 이와 협동가능한) 디지털 저장 매체, 예컨대 플로피 디스크, DVD, 블루레이, CD, ROM, PROM, EPROM, EEPROM 또는 FLASH 메모리를 이용하여 수행될 수 있다. 그러므로, 디지털 저장 매체는 컴퓨터로 판독가능할 수 있다.

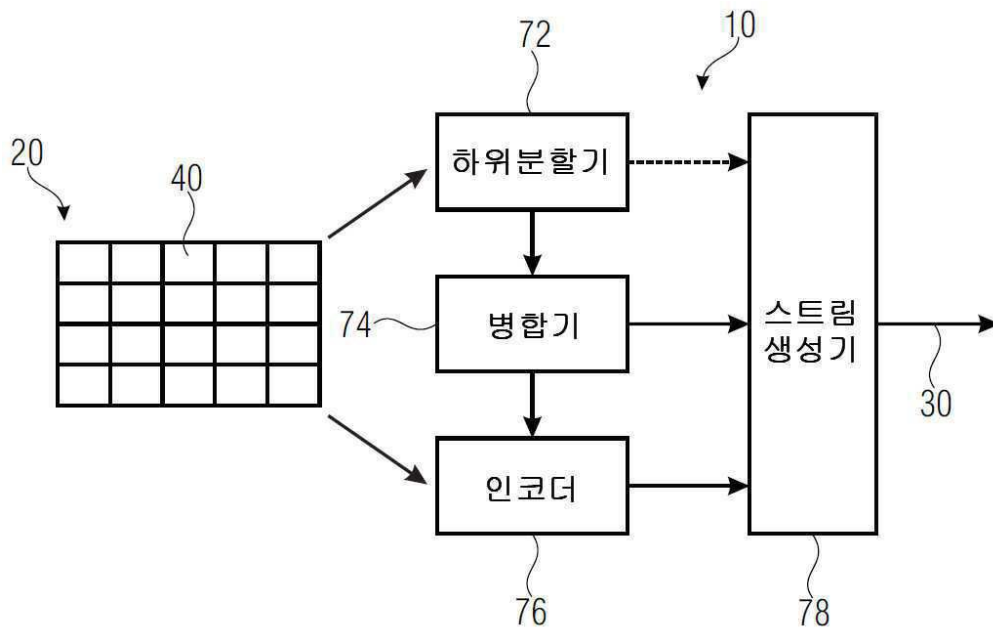
- [0215] 본 발명에 따른 몇몇의 실시예들은 여기서 설명된 방법들 중 하나의 방법이 수행되도록, 프로그램가능한 컴퓨터 시스템과 협동할 수 있는 전자적으로 판독가능한 제어 신호들을 갖는 데이터 캐리어를 포함한다.
- [0216] 일반적으로, 본 발명의 실시예들은 컴퓨터 프로그램 제품이 컴퓨터 상에서 구동될 때 본 방법들 중 하나의 방법을 수행하기 위해 동작되는 프로그램 코드를 갖는 컴퓨터 프로그램 제품으로서 구현될 수 있다. 프로그램 코드는 예컨대 머신 판독가능한 캐리어 상에 저장될 수 있다.
- [0217] 다른 실시예들은 머신 판독가능한 캐리어 상에서 저장되는, 여기서 설명된 방법들 중 하나의 방법을 수행하기 위한 컴퓨터 프로그램을 포함한다.
- [0218] 다시 말하면, 본 발명의 방법의 실시예는, 따라서, 컴퓨터 상에서 컴퓨터 프로그램이 구동될 때, 여기서 설명된 방법들 중 하나의 방법을 수행하기 위한 프로그램 코드를 갖는 컴퓨터 프로그램이다.
- [0219] 본 발명의 방법들의 추가적인 실시예는, 이에 따라 여기서 설명된 방법들 중 하나의 방법을 수행하기 위한 컴퓨터 프로그램이 기록되어 있는 데이터 캐리어(또는 디지털 저장 매체, 또는 컴퓨터 판독가능한 매체)이다. 데이터 캐리어, 디지털 저장 매체 또는 기록 매체는 일반적으로 유형적이며 및/또는 비일시적이다.
- [0220] 본 발명의 방법의 추가적인 실시예는, 이에 따라 여기서 설명된 방법들 중 하나의 방법을 수행하기 위한 컴퓨터 프로그램을 표현한 신호들의 시퀀스 또는 데이터 스트림이다. 신호들의 시퀀스 또는 데이터 스트림은 데이터 통신 접속, 예컨대 인터넷을 통해 전송되도록 구성될 수 있다.
- [0221] 추가적인 실시예는 여기서 설명된 방법들 중 하나의 방법을 수행하도록 구성되거나 적용된 처리수단, 예컨대 컴퓨터, 또는 프로그램가능 논리 디바이스를 포함한다.
- [0222] 추가적인 실시예는 여기서 설명된 방법들 중 하나의 방법을 수행하기 위한 컴퓨터 프로그램이 설치된 컴퓨터를 포함한다.
- [0223] 본 발명에 따른 추가적인 실시예는 여기서 설명된 방법들 중 하나의 방법을 수행하기 위한 컴퓨터 프로그램을 (예컨대, 전자적으로 또는 광학적으로) 수신기에 전송하도록 구성된 장치 또는 시스템을 포함한다. 수신기는, 예컨대 컴퓨터, 모바일 디바이스, 메모리 디바이스 등일 수 있다. 장치 또는 시스템은, 예컨대 컴퓨터 프로그램을 수신기에 전송하기 위한 파일 서버를 포함할 수 있다.
- [0224] 몇몇의 실시예들에서, 프로그램가능한 논리 디바이스(예컨대 필드 프로그램가능한 게이트 어레이)는 여기서 설명된 방법들의 기능들 모두 또는 그 일부를 수행하기 위해 이용될 수 있다. 몇몇의 실시예들에서, 여기서 설명된 방법들 중 하나의 방법을 수행하기 위해 필드 프로그램가능한 게이트 어레이가 마이크로프로세서와 협동할 수 있다. 일반적으로, 본 방법들은 바람직하게는 임의의 하드웨어 장치에 의해 수행된다.
- [0225] 상술한 실시예들은 본 발명의 원리들에 대한 일례에 불과하다. 여기서 설명된 구성 및 상세사항의 수정 및 변형은 본 발명분야의 당업자에게 자명할 것으로 이해된다. 그러므로, 본 발명은 계류중인 본 특허 청구항들의 범위에 의해서만 제한이 되며 여기서의 실시예들의 설명 및 해설을 통해 제시된 특정한 세부사항들에 의해서는 제한되지 않는다는 것이 본 취지이다.

도면

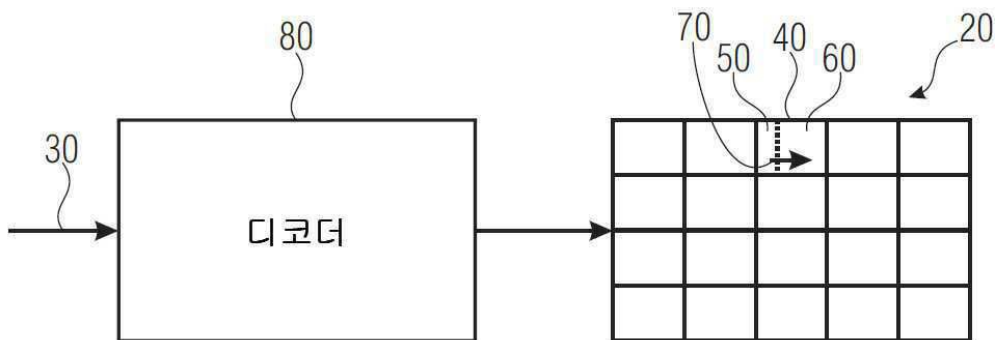
도면1



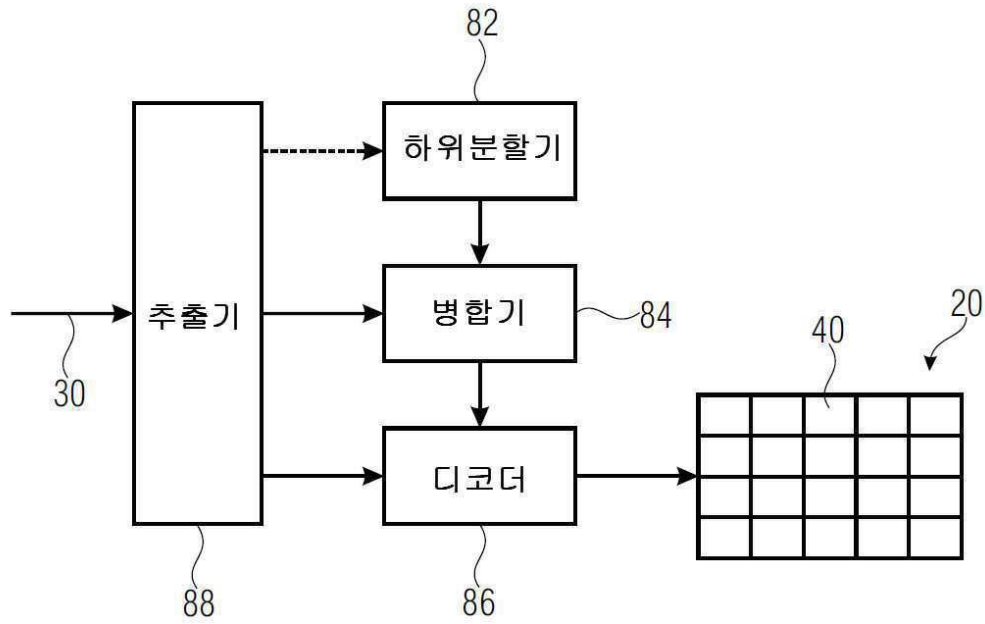
도면2



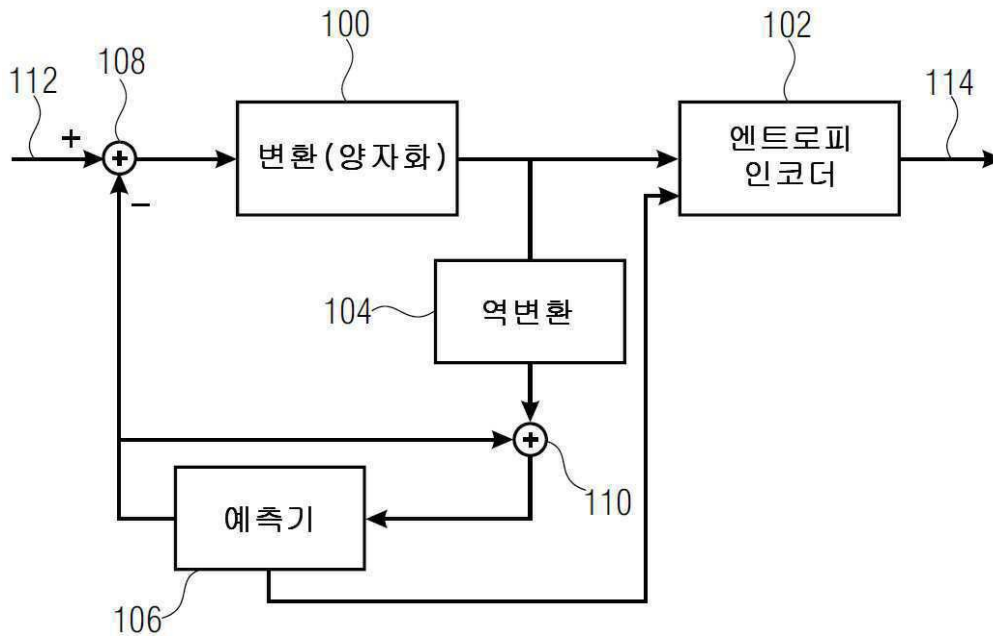
도면3



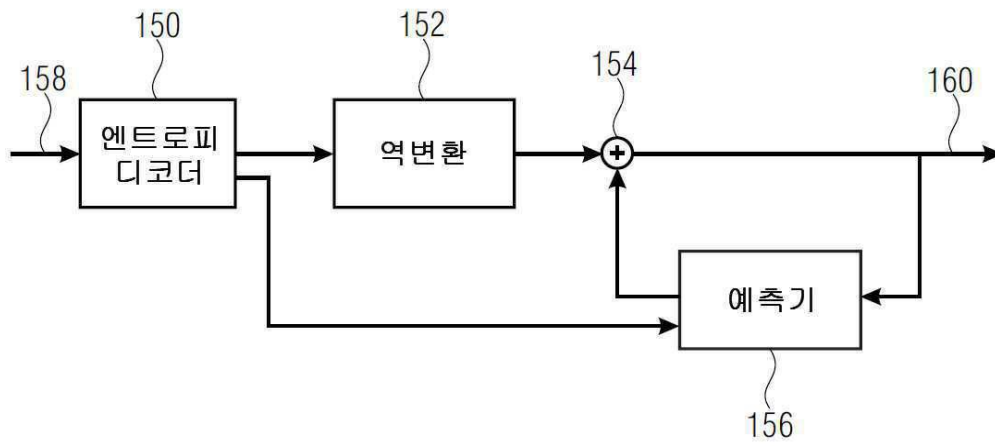
도면4



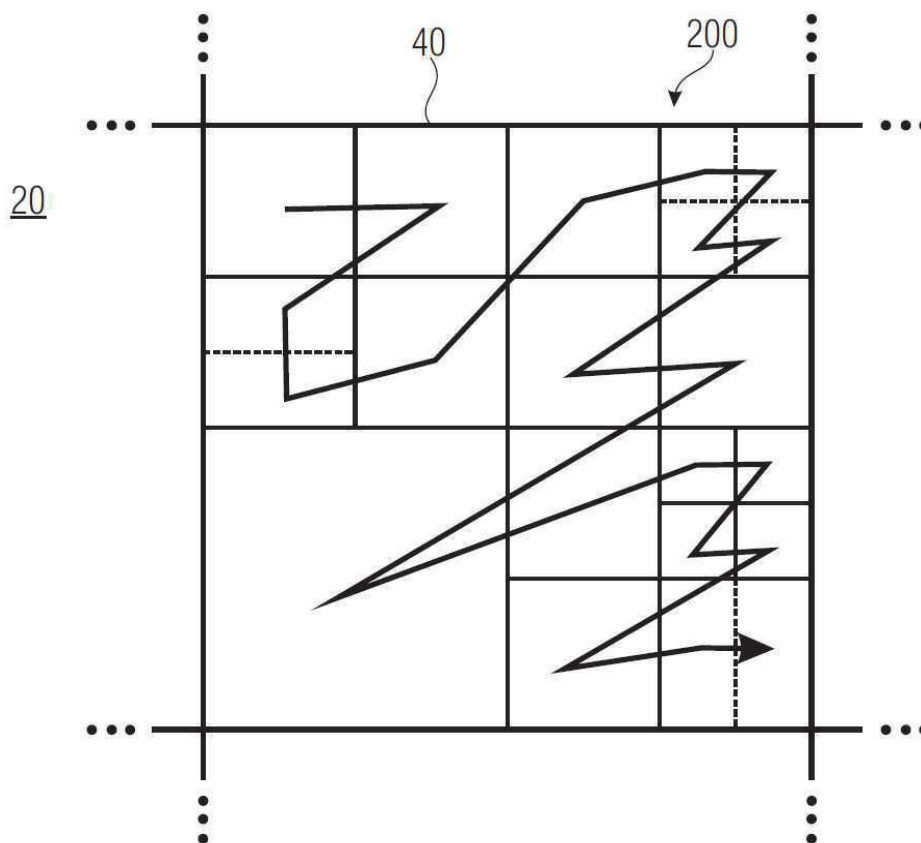
도면5



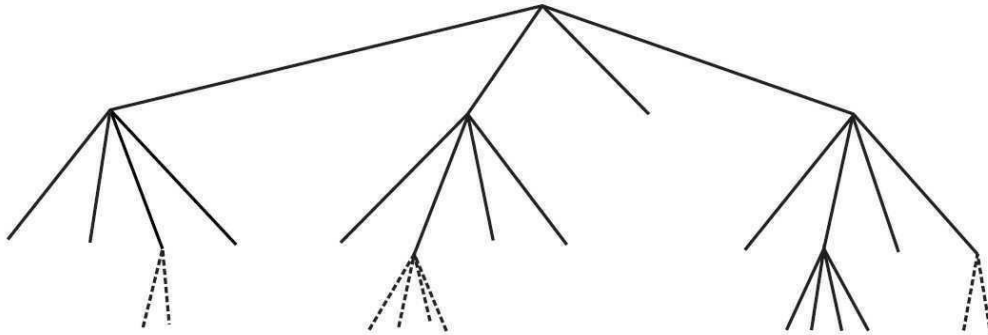
도면6



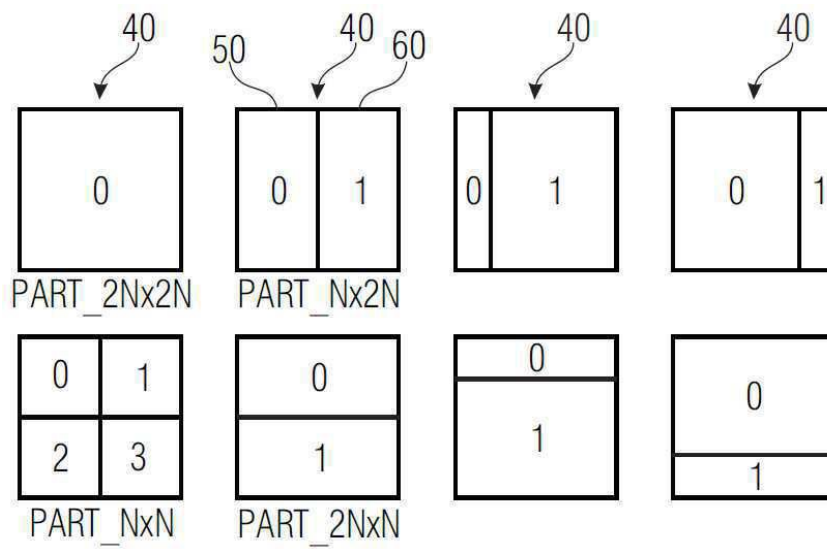
도면7a



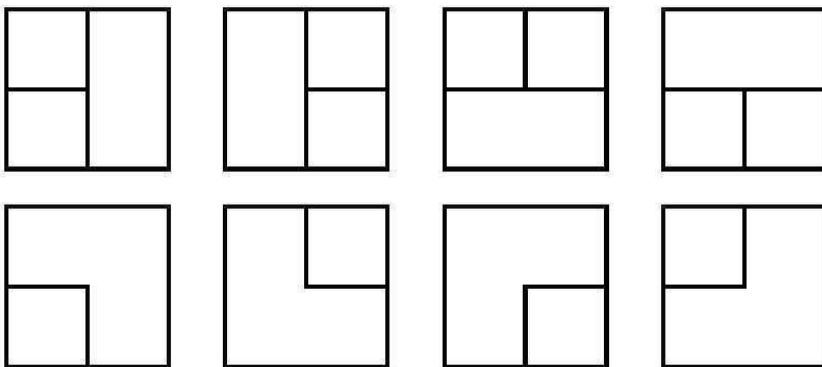
도면7b



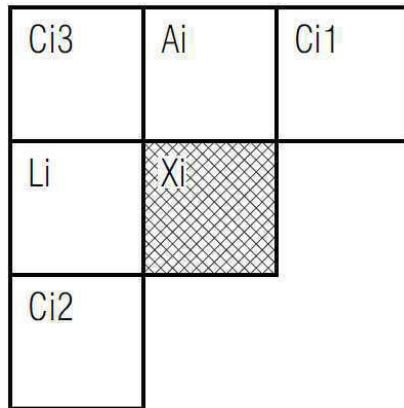
도면8



도면9



도면10



도면11

coding_tree(x0, y0, log2CUSize) {	기술어
<pre> if(x0 + (1 << log2CUSize) <= PicWidthInSamplesL && y0 + (1 << log2CUSize) <= PicHeightInSamplesL && cuAddress(x0, y0) >= SliceAddress) { 430 if(!entropy_coding_mode_flag && slice_type != I) cu_split_pred_part_mode[x0][y0] 400 else if(log2CUSize > Log2MinCUSize) split_coding_unit_flag[x0][y0] </pre>	ce(v)
<pre> } if(adaptive_loop_filter_flag && alf_cu_control_flag) { cuDepth = Log2MaxCUSize - log2CUSize if(cuDepth <= alf_cu_control_max_depth) if(cuDepth == alf_cu_control_max_depth split_coding_unit_flag[x0][y0] == 0) AlfCuFlagIdx++ 402 } if(split_coding_unit_flag[x0][y0]) { if(cu_qp_delta_enabled_flag && log2CUSize == log2MinCUDQPSize) IsCuQpDeltaCoded = 0 x1 = x0 + ((1 << log2CUSize) >> 1) y1 = y0 + ((1 << log2CUSize) >> 1) if(cuAddress(x1, y0) > SliceAddress) moreDataFlag = coding_tree(x0, y0, log2CUSize - 1) if(cuAddress(x0, y1) > SliceAddress && moreDataFlag && x1 < PicWidthInSamplesL) moreDataFlag = coding_tree(x1, y0, log2CUSize - 1) 432 if(cuAddress(x1, y1) > SliceAddress && moreDataFlag && y1 < PicHeightInSamplesL) moreDataFlag = coding_tree(x0, y1, log2CUSize - 1) if(moreDataFlag && x1 < PicWidthInSamplesL && y1 < PicHeightInSamplesL) moreDataFlag = coding_tree(x1, y1, log2CUSize - 1) } else { if(adaptive_loop_filter_flag && alf_cu_control_flag) AlfCuFlag[x0][y0] = alf_cu_flag[AlfCuFlagIdx] coding_unit(x0, y0, log2CUSize) if(!entropy_coding_mode_flag) moreDataFlag = more_rbsp_data() 404 else { if(granularity_block_boundary(x0, y0, log2CUSize)) { end_of_slice_flag moreDataFlag = !end_of_slice_flag } else moreDataFlag = 1 } } } return moreDataFlag } } } </pre>	u(1) ae(v)
<pre> } return moreDataFlag } </pre>	ae(v)

도면12

	기술어
<pre> coding_unit(x0, y0, log2CUSize) { 406 if(entropy_coding_mode_flag && slice_type != 1) skip_flag[x0][y0] if(skip_flag[x0][y0]) 428 prediction_unit(x0, y0, log2CUSize, log2CUSize, 0, 0) 408 else { if(!entropy_coding_mode_flag) { if(slice_type == 1 && log2CUSize == Log2MinCUSize) intra_part_mode 410 } else if(slice_type != 1 log2CUSize == Log2MinCUSize) pred_type x1 = x0 + ((1 << log2CUSize) >> 1) y1 = y0 + ((1 << log2CUSize) >> 1) if(PartMode == PART_2Nx2N) { prediction_unit(x0, y0, log2CUSize, log2CUSize, 0) 412 } else if(PartMode == PART_2NxN) { prediction_unit(x0, y0, log2CUSize, log2CUSize - 1, 0) prediction_unit(x0, y1, log2CUSize, log2CUSize - 1, 1) } else if(partmode == PART_Nx2N) { prediction_unit(x0, y0, log2CUSize - 1, log2CUSize, 0) prediction_unit(x1, y0, log2CUSize - 1, log2CUSize, 1) } else { /* PART_NxN */ prediction_unit(x0, y0, log2CUSize - 1, log2CUSize - 1, 0) prediction_unit(x1, y0, log2CUSize - 1, log2CUSize - 1, 1) prediction_unit(x0, y1, log2CUSize - 1, log2CUSize - 1, 2) prediction_unit(x1, y1, log2CUSize - 1, log2CUSize - 1, 3) } if(!pcm_flag) { 426 { transform_tree(x0, y0, log2CUSize, 0, 0) transform_coeff(x0, y0, log2CUSize, 0, 0) transform_coeff(x0, y0, log2CUSize, 0, 1) transform_coeff(x0, y0, log2CUSize, 0, 2) } } } </pre>	u(1) ae(v) u(1) u(v) ae(v)

도면13a

prediction_unit(x0, y0, log2PUWidth, log2PUHeight, PartIdx) {	기술어
if(skip_flag[x0][y0]) {	
414 merge_idx[x0][y0]	ue(v) ae(v)
} else if(PredMode == MODE_INTRA) {	
416 if(PartMode == PART_2Nx2N && log2PUWidth >= Log2IPCMCSize)	u(1) ae(v)
pcm_flag	
if(pcm_flag) {	
while (!byte_aligned())	
pcm_alignment_zero_bit	u(v)
for(i = 0; i < 1 << (log2CUSize << 1); i++)	
pcm_sample_luma[i]	u(v)
for(i = 0; i < (1 << (log2CUSize << 1)) >> 1; i++)	
pcm_sample_chroma[i]	u(v)
} else {	
prev_intra_luma_pred_flag[x0][y0]	u(1) ae(v)
if(prev_intra_luma_pred_flag[x0][y0])	
if(NumMPMCand < 1)	
mpm_idx[x0][y0]	u(1) ae(v)
else	
rem_intra_luma_pred_mode[x0][y0]	ce(v) ae(v)
if(IntraPredMode[x0][y0] == 2)	
planar_flag_luma[x0][y0]	u(1) ae(v)
intra_chroma_pred_mode[x0][y0]	ue(v) ae(v)
SignaledAsChromaDC =	
(chroma_pred_from_luma_enabled_flag ?	
intra_chroma_pred_mode[x0][y0] == 3 :	
intra_chroma_pred_mode[x0][y0] == 2)	
if(IntraPredMode[x0][y0] != 2 && IntraPredMode[x0][y0] != 34 && SignaledAsChromaDC)	
422 planar_flag_chroma[x0][y0]	u(1) ae(v)
}	
} else { /* MODE_INTER */	
if(entropy_coding_mode_flag PartMode != PART_2Nx2N)	
418 merge_flag[x0][y0]	u(1) ae(v)
if(merge_flag[x0][y0]) {	
420 merge_idx[x0][y0]	ue(v) ae(v)
} else {	
if(slice_type == B) {	
if(!entropy_coding_mode_flag) {	
combined_inter_pred_ref_idx	ue(v)
if(combined_inter_pred_ref_idx == MaxPredRef)	
inter_pred_flag[x0][y0]	ue(v)
...	
...	
...	

도면13b

⋮	} else inter_pred_flag[x0][y0]	ue(v) ae(v)
	} if(inter_pred_flag[x0][y0] == Pred_LC) { if(num_ref_idx_lc_active_minus1 > 0) { if(!entropy_coding_mode_flag) { if(combined_inter_pred_ref_idx == MaxPredRef) ref_idx_lc_minus4[x0][y0]	ue(v)
	} else ref_idx_lc[x0][y0]	ae(v)
	}	
	mvd_lc[x0][y0][0]	se(v) ae(v)
	mvd_lc[x0][y0][1]	se(v) ae(v)
	mvp_idx_lc[x0][y0]	ue(v) ae(v)
424	} else { /* Pred_L0 or Pred_BI */ if(num_ref_idx_10_active_minus1 > 0) { if(!entropy_coding_mode_flag) { if(combined_inter_pred_ref_idx == MaxPredRef) ref_idx_10_minusX[x0][y0]	ue(v)
	} else ref_idx_10_minusX[x0][y0]	ue(v) ae(v)
	}	
	mvd_10[x0][y0][0]	se(v) ae(v)
	mvd_10[x0][y0][1]	se(v) ae(v)
	mvp_idx_10[x0][y0]	ue(v) ae(v)
	}	
	if(inter_pred_flag[x0][y0] == Pred_BI) { if(num_ref_idx_11_active_minus1 > 0) { if(!entropy_coding_mode_flag) { if(combined_inter_pred_ref_idx == MaxPredRef) ref_idx_11_minusX[x0][y0]	ue(v)
	} else ref_idx_11[x0][y0]	ue(v) ae(v)
	}	
	mvd_11[x0][y0][0]	se(v) ae(v)
	mvd_11[x0][y0][1]	se(v) ae(v)
	mvp_idx_11[x0][y0]	ue(v) ae(v)
	}	
	}	
	}	

도면14

