



(19) 대한민국특허청(KR)

(12) 등록특허공보(B1)

(45) 공고일자 2024년09월04일

(11) 등록번호 10-2701434

(24) 등록일자 2024년08월28일

(51) 국제특허분류(Int. Cl.)  
H04N 19/513 (2014.01) H04N 19/105 (2014.01)  
H04N 19/13 (2014.01) H04N 19/523 (2014.01)

(52) CPC특허분류  
H04N 19/513 (2015.01)  
H04N 19/105 (2015.01)

(21) 출원번호 10-2023-7027427(분할)

(22) 출원일자(국제) 2019년06월20일

심사청구일자 2023년08월11일

(85) 번역문제출일자 2023년08월11일

(65) 공개번호 10-2023-0122686

(43) 공개일자 2023년08월22일

(62) 원출원 특허 10-2020-7035308

원출원일자(국제) 2019년06월20일

심사청구일자 2020년12월08일

(86) 국제출원번호 PCT/CN2019/092114

(87) 국제공개번호 WO 2020/007199

국제공개일자 2020년01월09일

(30) 우선권주장

201831024666 2018년07월02일 인도(IN)

(56) 선행기술조사문헌

US20180146208 A1\*

Cheng Du, et. al., "PPHPS: a parabolic prediction-based, fast half-pixel search algorithm for very low bit-rate moving-picture coding", IEEE Transactions on Circuits and Systems for Video Technology (Volume: 13, Issue: 6, June 2003), 2003.06.25, Page(s): 514 - 518.\*

Jianle Chen, et. al., "Algorithm Description of Joint Exploration Test Model 7 (JEM 7)", Joint Video Exploration Team (JVET) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 7th Meeting: Torino, IT, 2017.08.19, Document: JVET-G1001-v1.\*

\*는 심사관에 의하여 인용된 문헌

(73) 특허권자

후아웨이 테크놀로지 컴퍼니 리미티드

중국 518129 광둥성 셴젠 룡강 디스트릭트 반티안 후아웨이 어드미니스트레이션 빌딩

(72) 발명자

세투라만, 스리람

인도 560025 카르나타카주 방갈로어 리치몬드 로드 넘버1컨슬레이트-1

에이, 지바 라즈

인도 560025 카르나타카주 방갈로어 리치몬드 로드 넘버1컨슬레이트-1

코테차, 사가

인도 560025 카르나타카주 방갈로어 리치몬드 로드 넘버1컨슬레이트-1

(74) 대리인

양영준, 김성운, 백만기

전체 청구항 수 : 총 3 항

심사관 : 박상철

(54) 발명의 명칭 디코더 측 모션 벡터 정제를 위한 에러 표면 기반 서브 픽셀 정밀 정제 방법

### (57) 요약

디코더 측 모션 벡터 정제/도출이 코딩 시스템의 규범적 양태인 점을 고려하면, 인코더는 인코더의 재구성과 디코더의 재구성 사이에 어떠한 드리프트도 갖지 않기 위해 동일한 에러 표면 기술을 또한 수행해야 할 것이다. 그러므로, 모든 실시예들의 모든 양태들은 인코딩 및 디코딩 시스템들 양자 모두에 적용가능하다. 템플릿 매칭

(뒷면에 계속)

### 대표도



에서, 정제 움직임은 명시적으로 시그널링된 병합 인덱스에 기초하여 또는 비용 평가들을 통해 암시적으로 도출되는 서브 픽셀 정밀 센터로부터 시작하는 참조에서만 발생한다. (평균화된 템플릿이 있는 또는 없는) 쌍방 매칭에서, 명시적으로 시그널링된 병합 인덱스에 기초하여 또는 비용 평가들을 통해 암시적으로 도출되는 각각의 서브 픽셀 정밀 센터들로부터 시작하는 참조 리스트들 L0 및 L1에서 정제들이 시작된다.

(52) CPC특허분류

**H04N 19/13** (2015.01)

**H04N 19/523** (2015.01)

---

## 명세서

### 청구범위

#### 청구항 1

프로세서에 의하여 구현되는 디코더 측 모션 벡터 정제 방법으로서,

하나 이상의 참조 프레임의 각각의 참조 프레임에 대한 정수 거리 정제 모션 벡터를 결정하도록, 상기 프로세서에 의해 비용 함수를 사용하여 정수 1-픽셀 거리 정제 동작들을 반복적으로 수행하여 루프를 착수하는 단계;

상기 프로세서에 의해, 1-픽셀 이웃 위치들의 세트의 비용 함수 값에 대해 현재 동작의 검색 센터의 위치의 비용 함수 값을 결정하는 단계;

현재 반복의 검색 센터의 비용이 가장 낮다고 결정되는 경우:

마지막 검색 센터들의 비용 함수 값들 및 그 주위의 1-픽셀 이웃들의 세트를 사용하여 맞추어진 파라메트릭 에러 표면 상의 최소 값을 갖는 위치를 계산하는 것에 의해 각각의 참조 프레임에서 상기 마지막 검색 센터들 주위의 서브 픽셀 거리 정제 모션 벡터를 결정하는 단계; 및

각각의 참조 프레임에 대해 결정된 정수 거리 정제 모션 벡터 및 결정된 서브 픽셀 거리 정제 모션 벡터의 합으로서 총 정제 모션 벡터를 리턴하는 단계

를 수행하는 단계

를 포함하는, 방법.

#### 청구항 2

제1항에 있어서,

상기 검색 센터의 위치의 상기 비용 함수 값이 가장 낮은 비용 함수 값이 아닌 경우, 현재 동작이 마지막 동작 인지를 결정하는 단계;

상기 현재 동작이 마지막 동작인 경우, 모든 동작들에 걸쳐 가장 낮은 비용 값을 갖는 위치에 대응하는 정제 모션 벡터를 리턴하는 단계;

상기 현재 동작이 마지막 동작이 아닌 경우, 센터를 해당 마지막 동작에서 가장 낮은 비용을 갖는 위치로 업데이트하고, 상기 루프로 다시 가서 착수하는 단계를 반복하는 단계

를 더 포함하는, 방법.

#### 청구항 3

컴퓨터 판독가능 매체로서, 프로세서 상에서 실행될 때, 제1항 또는 제2항에 따른 방법의 단계들을 수행하는 명령어들을 저장하는 컴퓨터 판독가능 매체.

#### 청구항 4

삭제

#### 청구항 5

삭제

#### 청구항 6

삭제

#### 청구항 7

삭제

청구항 8

삭제

청구항 9

삭제

청구항 10

삭제

청구항 11

삭제

청구항 12

삭제

청구항 13

삭제

청구항 14

삭제

청구항 15

삭제

청구항 16

삭제

청구항 17

삭제

청구항 18

삭제

청구항 19

삭제

## 발명의 설명

## 기술 분야

## 배경 기술

[0001] <관련 출원들에 대한 상호 참조들>

[0002] 본 출원은 2018년 7월 2일자로 출원되고 발명의 명칭이 "AN ERROR SURFACE BASED SUB-PIXEL ACCURATE REFINEMENT METHOD FOR DECODER SIDE MOTION VECTOR REFINEMENT"인 인도 임시 특허 출원 제201831024666호의 이점을 청구하며, 이는 그 전체가 본 명세서에 참조로 인용된다.

[0003] <배경기술>

- [0004] H.264/AVC 또는 H.265/HEVC와 같은, 현재의 하이브리드 비디오 코덱들은 예측 코딩을 포함하는 압축을 이용한다. 비디오 시퀀스의 화상은 픽셀들의 블록들로 세분되고 다음으로 이러한 블록들이 코딩된다. 픽셀 단위로 블록을 코딩하는 대신에, 블록의 공간적 또는 시간적 근접성에서 이미 인코딩된 픽셀들을 사용하여 전체 블록이 예측된다. 인코더는 블록과 그 예측 사이의 차이만을 추가로 처리한다. 이러한 추가의 처리는 변환 도메인에서 블록 픽셀을 계수들로 변환하는 것을 통상적으로 포함한다. 다음으로 이러한 계수들은 양자화에 의해 추가로 압축되고 엔트로피 코딩에 의해 추가로 콤팩트화되어 비트스트림을 형성할 수 있다. 이러한 비트스트림은 인코딩된 비디오의 디코딩을 가능하게 하는 임의의 시그널링 정보를 추가로 포함한다. 예를 들어, 이러한 시그널링 정보는 입력 화상의 크기, 프레임 레이트, 양자화 단계 표시, 화상들의 블록들에 적용되는 예측 등과 같은 인코딩에 관한 설정들을 포함할 수 있다. 코딩된 시그널링 정보 및 코딩된 신호는 인코더 및 디코더 양자 모두에 알려진 방식으로 비트스트림 내에서 정렬된다. 이러한 것은 코딩된 시그널링 정보 및 코딩된 신호를 디코더가 파싱하는 것을 가능하게 한다.
- [0005] 시간적 예측은 비디오의, 프레임들이라고 또한 지칭되는, 화상들 사이의 시간적 상관을 활용한다. 시간적 예측은 인터 예측(inter-prediction)이라고 또한 불리며, 그 이유는 이것이 (인터) 상이한 비디오 프레임들 사이의 의존성들을 사용하는 예측이기 때문이다. 따라서, 현재 블록이라고 또한 지칭되는, 인코딩되고 있는 블록은 참조 화상(들)이라고 지칭되는 하나 이상의 이전에 인코딩된 화상(들)으로부터 예측된다. 참조 화상이 반드시 비디오 시퀀스의 디스플레이 순서에서 현재 블록이 위치되는 현재 화상에 선행하는 화상은 아니다. 인코더는 디스플레이 순서와 상이한 코딩 순서로 화상들을 인코딩할 수 있다. 현재 블록의 예측으로서, 참조 화상에서의 공동-위치 블록이 결정될 수 있다. 이러한 공동-위치 블록은 현재 화상에서의 현재 블록과 동일한 위치 상의 참조 화상에 위치되는 블록이다. 이러한 예측은 모션이 없는 화상 영역들, 즉, 하나의 화상으로부터 다른 것으로의 움직임이 없는 화상 영역들에 대해 정밀하다.
- [0006] 움직임을 참작하는 예측기, 즉, 모션 보상 예측기를 획득하기 위해, 현재 블록의 예측을 결정할 때 모션 추정이 통상적으로 이용된다. 따라서, 현재 블록은, 공동-위치 블록의 위치로부터 모션 벡터에 의해 주어지는 거리에 위치되는, 참조 화상에서의 블록에 의해 예측된다. 디코더가 현재 블록의 동일한 예측을 결정하는 것을 가능하게 하기 위해, 모션 벡터가 비트스트림에서 시그널링될 수 있다. 블록들 각각에 대한 모션 벡터를 시그널링하는 것에 의해 야기되는 시그널링 오버헤드를 추가로 감소시키기 위해, 모션 벡터 자체가 추정될 수 있다. 모션 벡터 추정은 공간적 및/또는 시간적 도메인에서 이웃 블록들의 모션 벡터들에 기초하여 수행될 수 있다.
- [0007] 현재 블록의 예측은 하나의 참조 화상을 사용하여 또는 2개 이상의 참조 화상들로부터 획득되는 예측들을 가중하는 것에 의해 계산될 수 있다. 참조 화상은 인접 화상, 즉, 디스플레이 순서에서 현재 화상에 바로 선행하는 화상 및/또는 바로 후속하는 화상일 수 있으며 그 이유는 인접 화상들이 현재 화상과 유사할 가능성이 가장 높기 때문이다. 그러나, 일반적으로, 참조 화상은 또한 디스플레이 순서에서 현재 화상에 선행하는 또는 후속하는 그리고 비트스트림(디코딩 순서)에서 현재 화상에 선행하는 임의의 다른 화상일 수 있다. 이러한 것은 예를 들어 비디오 콘텐츠에서의 폐색 및/또는 비-선형 움직임의 경우에 이점들을 제공할 수 있다. 따라서 참조 화상 식별은 비트스트림에서 또한 시그널링될 수 있다.
- [0008] 인터 예측의 특수 모드는 현재 블록의 예측을 생성함에 있어서 2개의 참조 화상들이 사용되는 소위 쌍방 예측이다. 특히, 각각의 2개의 참조 화상들에서 결정되는 2개의 예측들은 현재 블록의 예측 신호로 조합된다. 쌍방 예측은 단일 예측, 즉, 단일 참조 화상만을 사용하는 예측보다 현재 블록의 더 정밀한 예측을 초래할 수 있다. 더 정밀한 예측은 현재 블록의 픽셀들과 예측 사이의 더 작은 차이들("잔차들(residuals)")이라고 또한 지칭됨)로 이어지며, 이는 더 효율적으로 인코딩될 수 있다, 즉, 더 짧은 비트스트림으로 압축될 수 있다. 일반적으로, 현재 블록을 예측하기 위해 각각의 2개보다 많은 참조 블록들을 발견하기 위해 2개보다 많은 참조 화상들이 사용될 수 있다, 즉, 다중-참조 인터 예측이 적용될 수 있다. 따라서 다중-참조 예측이라는 용어는 2개보다 많은 참조 화상들을 사용하는 예측 뿐만 아니라 쌍방 예측을 포함한다.
- [0009] 더 정밀한 모션 추정을 제공하기 위해, 참조 화상의 해상도는 픽셀들 사이에 샘플들을 보간하는 것에 의해 강화될 수 있다. 가장 가까운 픽셀들의 가중 평균화에 의해 분수 픽셀 보간이 수행될 수 있다. 1/2 픽셀 해상도의 경우에, 예를 들어, 쌍선형 보간이 통상적으로 사용된다. 예측되는 픽셀에 대한 각각의 가장 가까운 픽셀들 사이의 거리의 역에 의해 가중되는 가장 가까운 픽셀들의 평균으로서 다른 분수 픽셀들이 계산된다.
- [0010] 모션 벡터 추정은 현재 블록과 참조 화상에서의 후보 모션 벡터들에 의해 포인팅되는 대응하는 예측 블록들 사이의 유사성이 계산되는 계산적으로 복잡한 작업이다. 통상적으로, 검색 영역은 이미지의  $M \times M$  샘플들을 포함하고  $M \times M$  후보 위치들의 샘플 위치 각각이 테스트된다. 이러한 테스트는, 검색 영역의 테스트된 후보 위치에

위치되는, 블록 R과 N x N 참조 블록 C 사이의 유사성 척도의 계산을 포함한다. 그 단순성을 위해, SAD(sum of absolute differences)는 이러한 목적을 위해 빈번히 사용되는 그리고 다음과 같이 주어지는 척도이다:

$$SAD(x, y) = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} |R_{i,j}(x, y) - C_{i,j}|$$

[0011]

[0012]

위 공식에서, x 및 y는 검색 영역 내의 후보 위치를 정의하고, 한편 인덱스들 i 및 j는 참조 블록 C 및 후보 블록 R 내의 샘플들을 표기한다. 후보 위치는 블록 변위 또는 오프셋이라고 종종 지칭되며, 이는 블록 매칭의 표현을 검색 영역 내의 참조 블록의 시프팅으로서 반영하고 검색 영역의 중첩 부분과 참조 블록 C 사이의 유사성을 계산한다. 복잡성을 감소시키기 위해, 후보 모션 벡터들을 특정 검색 공간으로 제한하는 것에 의해 후보 모션 벡터들의 수가 일반적으로 감소된다. 검색 공간은, 예를 들어, 현재 이미지에서의 현재 블록의 위치에 대응하는 참조 화상에서의 위치 주변의 픽셀들의 수 및/또는 위치들에 의해 정의될 수 있다. 모든 M x M 후보 위치들 x 및 y에 대해 SAD를 계산한 후에, 최상의 매칭 블록 R은, 참조 블록 C와의 가장 큰 유사성에 대응하는, 가장 낮은 SAD를 초래하는 위치 상의 블록이다. 다른 한편, 후보 모션 벡터들은 이웃 블록들의 모션 벡터들에 의해 형성되는 후보 모션 벡터들의 리스트에 의해 정의될 수 있다.

[0013]

모션 벡터들은 일반적으로 적어도 부분적으로 인코더 측에서 결정되고 코딩된 비트스트림 내에서 디코더에 시그널링된다. 그러나, 모션 벡터들은 또한 디코더에서 도출될 수 있다. 이러한 경우에, 현재 블록은 디코더에서 이용가능하지 않으며, 참조 화상에서 후보 모션 벡터들이 포인팅하는 블록들과의 유사성을 계산하기 위해 사용될 수 없다. 따라서, 현재 블록 대신에, 이미 디코딩된 블록들의 픽셀들로부터 구성되는 템플릿이 사용된다. 예를 들어, 현재 블록에 인접하는 이미 디코딩된 픽셀들이 사용될 수 있다. 이러한 모션 추정은 시그널링을 감소시키는 이점을 제공한다: 인코더 및 디코더 양자 모두에서 동일한 방식으로 모션 벡터가 도출되고, 따라서, 시그널링이 필요하지 않다. 다른 한편, 이러한 모션 추정의 정밀도는 더 낮을 수 있다.

[0014]

정밀도와 시그널링 오버헤드 사이의 트레이드오프를 제공하기 위해, 모션 벡터 추정은 2개의 단계들: 모션 벡터 도출 및 모션 벡터 정제로 분할될 수 있다. 예를 들어, 모션 벡터 도출은 후보들의 리스트로부터의 모션 벡터의 선택을 포함할 수 있다. 이러한 선택된 모션 벡터는 예를 들어 검색 공간 내의 검색에 의해 추가로 정제될 수 있다. 검색 공간에서의 검색은 각각의 후보 모션 벡터에 대한, 즉, 후보 모션 벡터가 포인팅하는 블록의 각각의 후보 위치에 대한 비용 함수를 계산하는 것에 기초한다.

[0015]

문헌 JVET-D0029: Decoder-Side Motion Vector Refinement Based on Bilateral Template Matching, X. Chen, J. An, J. Zheng(이러한 문헌은 <http://phenix.it-sudparis.eu/jvet/site>에서 발견될 수 있음)는 정수 픽셀 해상도에서의 제1 모션 벡터가 발견되고 제1 모션 벡터 주위의 검색 공간에서 1/2 픽셀 해상도를 갖는 검색에 의해 추가로 정제되는 모션 벡터 정제를 보여준다. 블록 템플릿에 기초하는 양방향 모션 벡터 검색이 이용된다.

[0016]

모션 벡터 추정은 현대의 비디오 코더들 및 디코더들의 핵심 특징이며 그 이유는 품질, 레이트, 및 복잡성의 면에서 그 효율성이 비디오 코딩 및 디코딩의 효율성에 영향을 미치기 때문이다.

## 발명의 내용

[0017]

본 발명은 비디오의 인코딩 및 디코딩에, 특히 모션 벡터들의 결정에 관련된다.

[0018]

본 출원의 제1 양태에서, 디코더 측 모션 벡터 정제 시스템에서 그들 각각의 초기 서브 픽셀 정밀 정제 센터(들) 주위의 하나 이상의 참조 프레임에서 서브 픽셀 정밀 델타 모션 벡터를 획득하기 위한 방법은 다음 단계들을 포함할 수 있다:

[0019]

비용 함수를 사용하여 복수의 정수 1-픽셀 거리 정제 동작들(반복들)을 반복적으로 수행하여 각각의 참조 프레임에 대한 정수 거리 정제 모션 벡터를 결정하는 단계- 검색 센터는 동작(반복) 후에 해당 동작(반복)에서 비용이 가장 낮은 위치로 업데이트됨 -, 주어진 동작(반복)에서의 센터 위치의 비용이 그 주위의 1-픽셀 이웃들의 세트의 비용보다 더 낮은 것으로 인해 반복 루프로부터의 조기 이탈이 발생한다고 결정하는 단계, 또는 미리 결정된 수의 동작들(반복들)이 도달된다고 결정하는 단계. 조기 이탈이 발생하는 경우에, 이러한 방법은 마지막 검색 센터들 및 그 주위의 1-픽셀 이웃들의 세트의 비용 함수 값들을 사용하여 맞추어진 파라메트릭 에러 표면 상의 값이 최소인 위치를 계산하는 것에 의해 각각의 참조에서의 마지막 검색 센터들 주위의 서브 픽셀 거리 정제 모션 벡터들을 결정하는 단계, 및 각각의 참조 프레임에 대해 결정된 정수 거리 정제 모션 벡터 및 결정된 서브 픽셀 거리 정제 모션 벡터의 합으로서 총 정제 모션 벡터를 리턴하는 단계를 포함할 수 있다. 미리 결정

된 수의 동작들(반복들)이 도달되는 경우에, 이러한 방법은 모든 동작들(반복들)에 걸쳐 가장 작은 비용 함수 값이 있는 위치에 대응하는 정제 모션 벡터들을 리턴하는 단계를 포함할 수 있다.

[0020] 제1 양태의 구현에서, 참조 리스트 L1에서의 변위들이 수평 및 수직 방향들 양자 모두에서 참조 리스트 L0에서의 변위들과 동일한 그리고 반대인 방식으로, 참조 리스트들 L0 및 L1 양자 모두에서 공동 정제를 위해 비용 함수가 수행된다(비용 함수는 SBM\_JOINT라고 지칭됨).

[0021] 본 출원의 제2 양태에서, 디코더 측 모션 벡터 정제 시스템에서 그들 각각의 초기 서브 픽셀 정밀 정제 센터(들) 주위의 하나 이상의 참조 프레임에서 서브 픽셀 정밀 델타 모션 벡터를 획득하기 위한 방법은 다음 단계들을 포함할 수 있다:

[0022] 제1 비용 함수를 사용하여 복수의 정수 1-픽셀 거리 정제 동작들(반복들)을 반복적으로 수행하여 각각의 참조 프레임에 대한 정수 거리 정제 모션 벡터를 결정하는 단계- 검색 센터는 동작(반복) 후에 해당 동작(반복)에서 비용이 가장 낮은 위치로 업데이트됨 -, 주어진 동작(반복)에서의 센터 위치의 비용이 그 주위의 1-픽셀 이웃들의 세트의 비용보다 더 낮은 것으로 인해 반복 루프로부터의 조기 이탈이 발생한다고 결정하는 단계, 또는 미리 결정된 수의 동작들(반복들)이 도달된다고 결정하는 단계;

[0023] 제2 비용 함수를 사용하여 마지막 검색 센터 및 마지막 검색 센터의 1-픽셀 이웃들의 세트에서의 비용 함수 값들을 평가하는 단계: 마지막 검색 센터 위치가 마지막 검색 센터에 대한 1-픽셀 이웃들의 세트에서의 제2 비용 함수 값들에 비교될 때 가장 낮은 제2 비용 함수 값을 가지면, 제2 비용 함수 값들을 사용하여 맞추어진 파라메트릭 에러 표면 상의 값이 최소인 위치를 계산하는 것에 의해 각각의 참조에서의 최상의 정수 거리 정제 위치 주위의 서브 픽셀 거리 정제 모션 벡터들을 결정하는 단계, 및 각각의 참조 프레임에 대해 결정된 정수 거리 정제 모션 벡터 및 결정된 서브 픽셀 거리 정제 모션 벡터의 합으로서 총 정제 모션 벡터를 리턴하는 단계. 마지막 검색 센터 위치가 그 밖의 마지막 검색 센터에 대한 1-픽셀 이웃들의 세트에서의 제2 비용 함수 값들에 비교될 때 가장 낮은 제2 비용 함수 값을 갖지 않으면, 이러한 방법은 각각의 참조 프레임에 대해 최상의 제2 비용 함수 값이 있는 위치에 대응하는 정제 모션 벡터를 리턴하는 단계를 포함한다.

[0024] 제2 양태의 구현에서, 제1 비용 함수는 SBM\_JOINT이고, 제2 비용 함수는 TBM\_INDEPENDENT이며, 이는 공통 쌍방 평균화된 템플릿에 대해 참조 리스트들 L0 및 L1 양자 모두에서 독립적 정제들을 수행하는 것과 연관된 비용 함수로서 정의된다.

[0025] 이러한 방법은 템플릿 매칭 사용-사례 및 쌍방 매칭 사용-사례에 대해 사용될 수 있다.

[0026] 본 발명의 다른 양태에서, 본 명세서에 개시되는 방법들은 비-일시적 컴퓨터 판독가능 매체에 저장되는 명령어 들로서 구현될 수 있으며, 이들은 설명되는 방법들의 단계들을 수행하기 위해 프로세서에 의해 판독되고 실행될 수 있다.

[0027] 본 발명의 일부 양태들에서, 디코더 측 모션 벡터 정제를 위한 방법은 초기 모션 벡터에 관하여 후보 정수 모션 벡터 변위들에 대응하는 정수 거리 비용들을 비교하는 것에 의해 타겟 정수 모션 벡터 변위를 결정하는 단계, 정수 거리 비용들에 대해 계산을 수행하는 것에 의해 서브 픽셀 모션 벡터 변위를 결정하는 단계, 및 타겟 정수 모션 벡터 변위, 서브 픽셀 모션 벡터 변위 및 초기 모션 벡터에 기초하여 정제된 모션 벡터를 결정하는 단계를 포함한다.

[0028] 본 발명에 의해 종래 기술들에 비해 다수의 이점들이 달성된다. 예를 들어, 본 발명의 실시예들은 센터 위치 주위의 1-픽셀 이웃들의 세트의 비용에 대해 주어진 반복 동작에서의 센터 위치의 비용의 체크에 기초하여 반복 루프로부터의 결정된 조기 이탈을 이용한다. 반복 루프를 조기에 이탈하는 것에 의해, 불필요한 계산들이 감소되거나 또는 제거될 수 있다.

[0029] 또한, 위에 설명된 방법들을 구현할 수 있는 장치는 소프트웨어와 하드웨어의 조합일 수 있다. 예를 들어, 인코딩 및/또는 디코딩은 범용 프로세서(CPU), DSP(digital signal processor), ASIC(application specific integrated circuit), 또는 FPGA(field programmable gate array) 등과 같은 집적 회로(반도체 디바이스 또는 칩)에 의해 수행될 수 있다. 그러나, 본 발명의 실시예들이 프로그램가능 하드웨어 상의 구현에 제한되는 것은 아니다. 본 발명의 실시예들은 주문형 집적 회로 상에 또는 CPU, DSP, FPGA 및 ASIC 컴포넌트들 중 하나 이상의 조합에 의해 구현될 수 있다.

## 도면의 간단한 설명



- [0030] 다음에서는, 예시적인 실시예들이 첨부 도면들 및 도해들을 참조하여 더 상세히 설명된다.
- 도 1은 본 개시내용의 실시예에 따른 비디오 신호들을 인코딩하기 위한 인코더의 예시적인 구조를 도시하는 블록도이다.
- 도 2는 본 개시내용의 실시예에 따른 비디오 신호를 디코딩하기 위한 디코더의 예시적인 구조를 도시하는 블록도이다.
- 도 3은 쌍방 예측에 적합한 예시적인 템플릿 매칭을 예시하는 개략도이다.
- 도 4a는 단일 예측 및 쌍방 예측에 적합한 예시적인 템플릿 매칭을 예시하는 개략도이다.
- 도 4b는 단일 예측 및 쌍방 예측에 적합한 예시적인 쌍방 매칭을 예시하는 개략도이다.
- 도 5는 모션 벡터 검색의 가능한 구현을 예시하는 흐름도이다.
- 도 6은 비디오 코딩에서 적용되는 국부적 조명 보상의 예를 예시하는 개략도이다.
- 도 7은 디코더 측 모션 벡터 정제의 예를 예시하는 개략도이다.
- 도 8은 서브 픽셀 위치의 예를 예시하는 개략도이다.
- 도 9는 본 개시내용의 실시예에 따른 하나 이상의 참조 프레임에서 서브 픽셀 정밀 델타 모션 벡터 정제를 획득하기 위한 센터 픽셀 주위의 1 픽셀 이웃 위치들의 세트의 블록도이다.
- 도 10은 본 개시내용의 일부 실시예들에 따른 디코더 측 모션 벡터 정제 시스템에서 하나 이상의 참조 프레임에서 서브 픽셀 정밀 델타 모션 벡터 정제를 획득하기 위한 방법을 예시하는 단순화된 흐름도이다.
- 도 11은 본 개시내용의 실시예를 구현하기 위한 방법을 예시하는 단순화된 흐름도이다.
- 도 12는 본 개시내용의 다양한 실시예들을 구현하기 위해 사용될 수 있는 장치의 블록도이다.

### 발명을 실시하기 위한 구체적인 내용

- [0031] 본 개시내용의 실시예들은 모션 벡터 정제에 적용되는 템플릿 매칭의 개선들에 관련된다. 특히, 템플릿 매칭은 발견된 최상의 매칭 블록들의 평균이 (국부적 조명 제어에 의해) 추가로 조정되지 않더라도 제로 평균 템플릿 및 제로 평균 후보 블록에 적용된다.
- [0032] 템플릿 매칭은 각각 제1 참조 화상 및 제2 참조 화상을 포인팅하는 최상의 제1 및 제2 모션 벡터들을 발견하기 위해 사용된다. 템플릿 매칭은 디코더에 의해 도출되거나 또는 디코더에 시그널링될 수 있는 초기 모션 벡터에 의해 주어지는 위치 상의 미리 결정된 검색 공간에서의 템플릿 매칭에 의해 각각의 참조 화상에 대해 수행된다.
- [0033] 템플릿 매칭은 초기 모션 벡터들에 의해 포인팅되는 블록들로부터 도출되는 블록 템플릿에 기초하여 수행될 수 있다.
- [0034] 현재 블록에 대한 예측기를 획득하기 위해 최상의 매칭 블록들을 발견하기 위한 이러한 템플릿 매칭은, 예를 들어, 하이브리드 비디오 인코더 및/또는 디코더에서 이용될 수 있다. 예를 들어, HEVC 또는 이와 유사한 것과 같은 인코더 및/또는 디코더로의 적용이 유리할 수 있다. 특히 HEVC 또는 새로운 코덱들/표준들의 추가의 개발들이 본 개시내용의 실시예들을 사용할 수 있다.
- [0035] 도 1은 비디오 스트림의 프레임들 또는 화상들의 입력 이미지 샘플들을 수신하기 위한 입력(102) 및 인코딩된 비디오 비트스트림을 생성하기 위한 출력(172)을 포함하는 인코더(100)를 도시한다. 본 명세서에서 사용되는 바와 같이, 본 개시내용에서의 "프레임(frame)"이라는 용어는 화상이라고 또한 지칭된다. 본 개시내용은 비디오의 필드들을 인터레이싱하는 것에 또한 적용가능하다는 점이 주목된다. 일반적으로, 화상은, 이미지 샘플들에 대응하고 하나 이상의 컬러 성분을 포함할 수 있는,  $m \times n$  픽셀들을 포함한다. 본 명세서에서 사용되는 바와 같이, 다음의 설명은 픽셀들을 휘도 샘플들이라고 지칭한다. 그러나, 본 개시내용의 모션 벡터 검색은 RGB 등과 같은 검색 공간의 색차 또는 성분들을 포함하는 임의의 컬러 성분에 적용될 수 있다는 점이 주목된다. 또한, 하나의 성분에 대한 모션 벡터 추정만을 수행하고 추정된 모션 벡터를 더 많은 또는 모든 성분들에 적용하는 것이 유리할 수 있다.
- [0036] 코딩된 입력 블록들이 반드시 동일한 크기를 갖는 것은 아니다. 하나의 화상은 상이한 크기들의 블록들을 포함



할 수 있고 상이한 화상들의 블록 래스터가 또한 상이할 수 있다.

[0037] 예시적인 실시예에서, 인코더(100)는 비디오 스트림에 예측, 변환, 양자화, 및 엔트로피 코딩을 수행하도록 구성된다. 이러한 변환, 양자화, 및 엔트로피 코딩이 변환 유닛(106), 양자화 유닛(108) 및 엔트로피 인코딩 유닛(170)에 의해 각각 수행되어 인코딩된 비디오 비트스트림을 생성한다.

[0038] 이러한 비디오 스트림은 복수의 프레임들을 포함할 수 있으며, 각각의 프레임은 인트라 또는 인터 코딩되는 특정 크기의 블록들로 분할된다. 예를 들어, 비디오 스트림의 제1 프레임의 블록들이 인트라 예측 유닛(154)에 의해 인트라 코딩된다. 인트라 프레임은 동일한 프레임 내의 정보만을 사용하여 코딩되어, 이것은 독립적으로 디코딩될 수 있고 이것은 랜덤 액세스를 위해 비트스트림에서 엔트리 포인트를 제공할 수 있다. 비디오 스트림의 다른 프레임들의 블록들이 인터 예측 유닛(144)에 의해 인터 코딩될 수 있다, 즉, 이전에 코딩된 프레임들(참조 프레임들)로부터의 정보가 시간적 중복성을 감소시키기 위해 사용되어, 인터 코딩된 프레임의 각각의 블록이 참조 프레임에서의 블록으로부터 예측된다. 모드 선택 유닛(160)은 인트라 예측 유닛(154)에 의해 처리되는 프레임의 블록(155) 또는 인터 예측 유닛(144)에 의해 처리되는 프레임의 블록(145) 사이에서 선택하도록 구성된다. 모드 선택 유닛(160)은 인트라 또는 인터 예측의 파라미터들을 또한 제어한다. 이미지 정보의 리프레싱을 가능하게 하기 위해, 인트라 코딩된 블록들이 인터 코딩된 프레임들 내에 제공될 수 있다. 또한, 디코딩을 위한 엔트리 포인트들, 즉, 디코더가 이전에 코딩된 프레임들로부터의 정보를 갖지 않고 디코딩을 시작할 수 있는 포인트들을 제공하기 위해 비디오 시퀀스에 인트라 코딩된 블록들만을 포함하는 인트라 프레임들이 규칙적으로 삽입될 수 있다.

[0039] 인트라 추정 유닛(152) 및 인트라 예측 유닛(154)은 인트라 예측을 수행하도록 구성되는 유닛들이다. 특히, 인트라 추정 유닛(152)은 원본 이미지의 지식에 또한 기초하여 예측 모드를 도출할 수 있는 한편 인트라 예측 유닛(154)은, 차이 코딩을 위해, 대응하는 예측기, 즉, 선택된 예측 모드를 사용하여 예측되는 샘플들을 제공한다. 공간적 또는 시간적 예측을 수행하기 위해, 코딩된 블록들은 역 양자화 유닛(110) 및 역 변환 유닛(112)에 의해 추가로 처리되어 역 변환된 블록(113)을 제공할 수 있다. 재구성 유닛(114)이 역 변환된 블록(113)을 예측 블록(165)과 조합하여 재구성된 블록(115)을 제공하며, 이는 루프 필터링 유닛(120)에 제공되어 디코딩된 이미지의 품질을 추가로 개선한다. 필터링된 블록들은 디코딩된 화상 버퍼(130)에 다음으로 저장되는 참조 프레임들을 다음으로 형성한다. 역 양자화 유닛(110), 역 변환 유닛(112), 재구성 유닛(114), 및 루프 필터(120)는 디코더(디코딩 루프)의 일부분을 형성한다. 인코더 측에서의 이러한 디코딩 루프(디코더)는 디코더 측에서 재구성되는 참조 화상들과 동일한 참조 프레임들을 생성하는 이점을 제공한다. 따라서, 인코더 및 디코더는 대응하는 방식으로 동작한다. 본 명세서에서 "재구성(reconstruction)"이라는 용어는 예측 블록(165)을 역 변환된(디코딩된 잔차) 블록(113)에 추가하는 것에 의해 재구성된 블록(115)을 획득하는 것을 지칭한다.

[0040] 인코더(100)는 인터 추정 유닛(142)을 또한 포함하며, 이는 인터 코딩될 현재 프레임 또는 화상의 화상 블록(101) 및 디코딩된 화상 버퍼(130)로부터의 하나의 또는 몇몇 참조 프레임들을 수신한다. 인터 추정 유닛(142)에 의해 모션 추정이 수행되는 반면 인터 예측 유닛(144)에 의해 모션 보상이 수행된다. 모션 추정은, 예를 들어 코딩될 원본 이미지를 또한 사용하여, 특정 비용 함수에 기초하여 모션 벡터 및 참조 프레임을 획득하기 위해 사용된다. 예를 들어, 모션 추정(인터 추정) 유닛(142)이 초기 모션 벡터 추정을 제공할 수 있다. 다음으로 초기 모션 벡터는 인코더 및 디코더에서 동일한 방식으로 미리 결정된 규칙에 기초하여 구성되는 후보들의 리스트 내의 모션 벡터 후보를 참조하는 인덱스로서 또는 직접 모션 벡터의 형태로 비트스트림 내에서 시그널링(신호로서 송신)될 수 있다. 다음으로 모션 보상은 참조 프레임에서 현재 블록과 공동-위치되는 블록의 참조 프레임에서 참조 블록으로의 변환으로서 현재 블록의 예측기, 즉, 모션 벡터를 도출한다. 인터 예측 유닛(144)은 현재 블록에 대한 예측 블록(145)을 출력하며, 여기서 예측 블록(145)은 비용 함수를 최소화한다. 예를 들어, 비용 함수는 코딩될 현재 블록과 그 예측 블록 사이의 차이일 수 있다, 즉, 비용 함수는 잔차 블록(105)을 최소화한다. 잔차 블록의 최소화는, 예를 들어, 현재 블록과 후보 참조 화상에서의 후보 블록의 모든 픽셀들(샘플들) 사이의 SAD(sum of absolute differences)를 계산하는 것에 기초한다. 일반적으로, MSE(mean square error) 또는 SSIM (structural similarity metric)과 같은, 임의의 다른 유사성 메트릭이 이용될 수 있다.

[0041] 비용 함수는 또한 이러한 인터 블록을 코딩하기 위해 필요한 비트들의 수 및/또는 이러한 코딩으로부터 초래되는 왜곡일 수 있다. 따라서, 모션 벡터 선택에 대해 및/또는 일반적으로 블록에 대한 인터 또는 인트라 예측을 사용할지 및 어떠한 설정들로와 같은 인코딩 파라미터들에 대해 결정하기 위해 레이트 왜곡 최적화 프로시저가 사용될 수 있다.

- [0042] 인트라 추정 유닛(152) 및 인터 예측 유닛(154)은 현재 프레임의 화상 블록(101) 또는 인트라 코딩될 화상 및 현재 프레임의 이미 재구성된 영역으로부터의 하나의 또는 몇몇 참조 샘플(117)을 입력으로서 수신한다. 다음으로 인트라 예측은 현재 프레임의 참조 샘플들의 함수의 면에서 현재 프레임의 현재 블록의 픽셀들을 설명한다. 인트라 예측 유닛(154)은 현재 블록에 대한 예측 블록을 출력하며, 여기서 예측 블록은 코딩될 현재 블록과 그 예측 블록 사이의 차이를 유리하게 최소화한다, 즉, 잔차 블록을 최소화한다. 잔차 블록의 최소화는, 예를 들어, 레이트 왜곡 최적화 프로시저에 기초할 수 있다. 특히, 예측 블록은 참조 샘플들의 방향성 보간으로서 획득된다. 이러한 방향은 인터 예측과 관련하여 위에 설명된 바와 같이 유사성 척도를 계산하는 것에 의해 및/또는 레이트 왜곡 최적화에 의해 결정될 수 있다.
- [0043] 인터 추정 유닛(142)은 인터 코딩될 현재 프레임 또는 화상의 블록 또는 더 보편적으로 형성된 이미지 샘플 및 2개 이상의 이미 디코딩된 화상들(231)을 입력으로서 수신한다. 다음으로 인터 예측은 참조 화상들의 참조 이미지 샘플들에 대한 모션 벡터들의 면에서 현재 프레임의 현재 이미지 샘플을 설명한다. 인터 예측 유닛(144)은 현재 이미지 샘플에 대한 하나 이상의 모션 벡터(145)를 출력하며, 여기서 모션 벡터들에 의해 포인팅되는 참조 이미지 샘플들은 코딩될 현재 이미지 샘플과 그 참조 이미지 샘플들 사이의 차이를 유리하게 최소화한다, 즉, 잔차 이미지 샘플을 최소화한다. 다음으로 현재 블록에 대한 예측기가 차이 코딩을 위해 인터 예측 유닛(144)에 의해 제공된다.
- [0044] 다음으로 현재 블록과 그 예측, 즉, 잔차 블록(105) 사이의 차이가 변환 유닛(106)에 의해 변환되어 변환된 계수들(107)을 생성한다. 변환된 계수들(107)은 양자화 유닛(108)에 의해 양자화되고 엔트로피 인코딩 유닛(170)에 의해 엔트로피 코딩된다. 이렇게 생성된 인코딩된 화상 데이터(171), 즉, 인코딩된 비디오 비트스트림은, 인트라 코딩된 블록들 및 인터 코딩된 블록들 및 (모드 표시, 모션 벡터의 표시, 및/또는 인트라 예측 방향과 같은) 대응하는 시그널링 정보를 포함한다. 변환 유닛(106)은 DFT/FFT 또는 DCT(Discrete Fourier Transformation, Fast Fourier Transformation 또는 Discrete Cosine Transformation)과 같은 선형 변환을 적용할 수 있다. 공간적 주파수 도메인으로의 변환은 결과적인 계수(107)가 더 낮은 주파수에서 더 높은 값들을 통상적으로 갖는다는 이점을 제공한다. 따라서, (지그재그 스캐닝과 같은) 유효 계수 스캐닝, 및 양자화 후에, 결과적인 값들의 시퀀스는 시작에서 일부 더 큰 값들을 통상적으로 갖고 끝에서 제로들의 시퀀스를 갖는다. 이러한 것은 추가의 효율적인 코딩을 가능하게 한다. 양자화 유닛(108)은 계수 값들의 해상도를 감소시키는 것에 의해 실제 손실 압축을 수행한다. 다음으로 엔트로피 코딩 유닛(170)은 계수 값들에 바이너리 코드워드들을 배정하여 비트스트림을 생성한다. 엔트로피 코딩 유닛(170)은 또한 시그널링 정보(도 1에 도시되지 않음)를 코딩한다.
- [0045] 도 2는 비디오 디코더(200)를 도시한다. 비디오 디코더(200)는 디코딩된 화상 버퍼(230), 인터 예측 유닛(244) 및, 블록 예측 유닛인, 인트라 예측 유닛(254)을 포함한다. 디코딩된 화상 버퍼(230)는 인코딩된 비디오 비트스트림으로부터 재구성되는 (단일 예측을 위한) 적어도 하나의 또는 (쌍방 예측을 위한) 적어도 2개의 참조 프레임들을 저장하도록 구성되며, 이러한 참조 프레임들은 인코딩된 비디오 비트스트림의 현재 프레임(현재 디코딩된 프레임)과 상이하다. 인트라 예측 유닛(254)은, 디코딩된 블록의 추정인, 예측 블록을 생성하도록 구성된다. 인트라 예측 유닛(254)은 디코딩된 화상 버퍼(230)로부터 획득되는 참조 샘플들에 기초하여 이러한 예측을 생성하도록 구성된다.
- [0046] 디코더(200)는 비디오 인코더(100)에 의해 생성되는 인코딩된 비디오 비트스트림을 디코딩하도록 구성되고, 바람직하게는 디코더(200) 및 인코더(100) 양자 모두가 인코딩/디코딩될 각각의 블록에 대해 동일한 예측들을 생성한다. 디코딩된 화상 버퍼(230) 및 인트라 예측 유닛(254)의 특징들은 도 1의 디코딩된 화상 버퍼(130) 및 인트라 예측 유닛(154)의 특징들과 유사하다.
- [0047] 비디오 디코더(200)는, 비디오 코더(100)의 역 양자화 유닛(110), 역 변환 유닛(112), 및 루프 필터링 유닛(120)에 각각 대응하는, 예를 들어, 역 양자화 유닛(210), 역 변환 유닛(212), 및 루프 필터링 유닛(220)과 같은, 비디오 인코더(100)에 또한 존재하는 유닛들을 추가로 포함한다.
- [0048] 엔트로피 디코딩 유닛(204)은 수신된 인코딩된 비디오 비트스트림을 디코딩하도록 그리고 양자화된 잔차 변환 계수들(209) 및 시그널링 정보를 대응하여 획득하도록 구성된다. 양자화된 잔차 변환 계수들(209)은 역 양자화 유닛(210) 및 역 변환 유닛(212)에 제공되어 잔차 (역 변환된) 블록을 생성한다. 재구성된 유닛(214)에서 예측 블록(265)에 잔차 블록이 추가되고, 이러한 추가는 디코딩된 비디오를 획득하기 위해 루프 필터링 유닛(220)에 제공된다. 디코딩된 비디오의 프레임들은 디코딩된 화상 버퍼(230)에 저장될 수 있고, 인터 예측을 위한 디코딩된 화상(231)으로서 역할을 할 수 있다.

- [0049] 일반적으로, 도 1 및 도 2의 인트라 예측 유닛들(154 및 254)은 인코딩될 필요가 있는 또는 디코딩될 필요가 있는 블록들에 대한 예측 신호들을 생성하기 위해 이미 인코딩된 영역으로부터의 참조 샘플들을 사용할 수 있다.
- [0050] 엔트로피 디코딩 유닛(204)은 인코딩된 비트스트림(171)을 그 입력으로서 수신한다. 일반적으로, 비트스트림이 먼저 파싱된다, 즉, 시그널링 파라미터들 및 잔차들이 비트스트림으로부터 추출된다. 통상적으로, 비트스트림의 신택스 및 시맨틱은 인코더들 및 디코더들이 상호운용가능한 방식으로 작동할 수 있도록 표준에 의해 정의된다. 위 배경기술 섹션에서 설명되는 바와 같이, 인코딩된 비트스트림이 예측 잔차들만을 포함하는 것은 아니다. 모션 보상된 예측의 경우에, 모션 벡터 표시가 또한 비트스트림에서 코딩되고 디코더에서 그로부터 파싱된다. 모션 벡터 표시는 모션 벡터가 제공되는 참조 화상에 의해 그리고 모션 벡터 좌표들에 의해 주어질 수 있다. 지금까지, 완전한 모션 벡터들을 코딩하는 것이 고려되었다. 그러나, 비트스트림에서의 현재 모션 벡터와 이전 모션 벡터 사이의 차이만이 또한 인코딩될 수 있다. 이러한 접근법은 이웃 블록들의 모션 벡터들 사이의 중복성의 이용을 허용한다.
- [0051] 참조 화상을 효율적으로 코딩하기 위해, H.265 코덱(ITU-T, H265, Series H: Audiovisual and multimedia systems: High Efficient Video Coding)은 각각의 참조 프레임은 리스트 인덱스들에 배정하는 참조 화상들의 리스트를 제공한다. 다음으로 참조 프레임은 대응하는 배정된 리스트 인덱스를 포함하는 것에 의해 비트스트림에서 시그널링된다. 이러한 리스트는 표준에서 정의되거나 또는 비디오의 시작 또는 다수의 프레임들의 세트에서 시그널링될 수 있다. H.265에는, 참조 리스트들 L0 및 L1이라고 지칭되는, 정의된 참조 화상들의 2개의 리스트가 존재한다는 점이 주목된다. 다음으로 참조 화상은 참조 리스트(L0 또는 L1)를 표시하는 것 및 원하는 참조 화상과 연관된 해당 리스트에서의 인덱스를 표시하는 것에 의해 비트스트림에서 시그널링된다. 2개 이상의 리스트들을 제공하는 것은 더 양호한 압축에 대한 이점들을 가질 수 있다. 예를 들어, 참조 리스트 L0은 단일 방향으로 인트라 예측된 슬라이스들 및 양방향으로 인트라 예측된 슬라이스들 양자 모두에 대해 사용될 수 있고 한편 참조 리스트 L1은 양방향으로 인트라 예측된 슬라이스들에 대해서만 사용될 수 있다. 그러나, 일반적으로 본 개시내용은 리스트들 L0 및 L1의 어떠한 콘텐츠에 제한되는 것은 아니다.
- [0052] 참조 리스트들 L0 및 L1은 표준에서 정의되고 고정될 수 있다. 그러나, 비디오 시퀀스의 시작에서 이들을 시그널링하는 것에 의해 코딩/디코딩에서의 더 많은 유연성이 달성될 수 있다. 따라서, 인코더는 인덱스에 따라 정렬되는 특정 참조 화상들로 리스트들 L0 및 L1을 구성할 수 있다. 참조 리스트들 L0 및 L1은 동일한 고정 크기를 가질 수 있다. 일반적으로, 2개보다 많은 리스트들이 존재할 수 있다. 모션 벡터는 참조 화상에서의 좌표에 의해 직접 시그널링될 수 있다. 대안적으로, H.265에서 또한 명시되는 바와 같이, 후보 모션 벡터들의 리스트가 구성될 수 있고, 특정 모션 벡터가 있는 리스트에서 연관되는 인덱스가 송신될 수 있다.
- [0053] 현재 블록의 모션 벡터들은 현재 화상에서의 또는 더 조기에 코딩된 화상들에서의 이웃 블록들의 모션 벡터들과 일반적으로 상관된다. 이러한 것은 이웃 블록들이 유사한 모션이 있는 동일한 움직이는 객체에 대응할 가능성이 있고 이러한 객체의 모션이 시간에 따라 갑자기 변경될 가능성이 없기 때문이다. 결과적으로, 이웃 블록들에서의 모션 벡터들을 예측기들로서 사용하는 것은 시그널링된 모션 벡터 차이의 크기를 감소시킨다. MVP들(Motion Vector Predictors)은 공동-위치 화상에서의 시간적 이웃 블록들로부터의 또는 공간적 이웃 블록들로부터의 이미 인코딩/디코딩된 모션 벡터들로부터 일반적으로 도출된다. H.264/AVC에서, 이러한 것은 3개의 공간적 이웃 모션 벡터들의 성분별 중앙값을 수행하는 것에 의해 행해진다. 이러한 접근법을 사용하면, 예측기의 시그널링이 요구되지 않는다. 공동-위치 화상으로부터의 시간적 MVP들은 H.264/AVC의 소위 시간적 직접 모드에서만 고려된다. H.264/AVC 직접 모드들은 모션 벡터들 이외의 다른 모션 데이터를 도출하기 위해 또한 사용된다. 그러므로, 이들은 HEVC에서의 블록 병합 개념에 더 관련된다. HEVC에서, MVP를 암시적으로 도출하는 접근법은, MVP들의 리스트로부터의 어느 MVP가 모션 벡터 도출에 대해 사용되는지를 명시적으로 시그널링하는, 모션 벡터 경합으로서 알려진 기술로 대체되었다. HEVC에서의 가변 코딩 쿼드-트리 블록 구조는 잠재적 MVP 후보들로서 모션 벡터들이 있는 몇몇 이웃 블록들을 갖는 하나의 블록을 초래할 수 있다. 좌측 이웃을 예로서 취하면, 최악의 경우에 64x64 루마 예측 블록은 64x64 루마 코딩 트리 블록이 추가로 분열되지 않고 좌측의 것이 최대 깊이로 분열될 때 좌측으로 16개의 4x4 루마 예측 블록들을 가질 수 있다.
- [0054] 모션 벡터 경합을 수정하여 이러한 유연한 블록 구조를 설명하기 위해 AMVP(Advanced Motion Vector Prediction)가 도입되었다. HEVC의 개발 동안, 초기 AMVP 설계는 코딩 효율성과 구현 친화적 설계 사이의 양호한 트레이드-오프를 제공하기 위해 상당히 단순화되었다. AMVP의 초기 설계는 3개의 상이한 부류들의 예측기들: 공간적 이웃들로부터의 3개의 모션 벡터들, 3개의 공간적 예측기들의 중앙값 및 공동-위치, 시간적 이웃 블록들로부터의 스케일링된 모션 벡터로부터의 5개의 MVP들을 포함했다. 또한, 예측기들의 리스트는 가장 가능성 있는 모션 예측기를 제1 위치에 배치하도록 재정렬하는 것에 의해 그리고 최소 시그널링 오버헤드를 보



증하도록 중복 후보들을 제거하는 것에 의해 수정되었다. AMVP 후보 리스트 구성의 최종 설계는 다음의 2개의 MVP 후보들: a) 5개의 공간적 이웃 블록들로부터 도출되는 2개까지의 공간적 후보 MVP; b) 공간적 후보 MVP들 양자 모두가 이용가능하지 않거나 또는 이들이 동일할 때의 2개의 시간적, 공동-위치 블록들로부터 도출되는 하나의 시간적 후보 MVP; 및 c) 공간적, 시간적 또는 양자 모두의 후보가 이용가능하지 않을 때의 제로 모션 벡터들을 포함한다. 모션 벡터 결정에 관한 상세들은, 본 명세서에 참조로 인용되는, V. Sze et al (Ed.)이 저술한 책: High Efficiency Video Coding (HEVC): Algorithms and Architectures, Springer, 2014에서, 특히 Chapter 5에서 발견될 수 있다.

[0055] 시그널링 오버헤드를 추가로 증가시키지 않고 모션 벡터 추정을 추가로 개선하기 위해, 인코더 측에서 도출되는 그리고 비트스트림에서 제공되는 모션 벡터들을 추가로 정제하는 것이 유익할 수 있다. 모션 벡터 정제는 인코더로부터의 원조없이 디코더에서 수행될 수 있다. 그 디코더 루프에서의 인코더는 대응하는 모션 벡터들을 획득하기 위해 동일한 정제를 이용할 수 있다. 참조 화상의 정수 픽셀 위치들 및 분수 픽셀 위치들을 포함하는 검색 공간에서 모션 벡터 정제가 수행된다. 예를 들어, 분수 픽셀 위치들은 1/2 픽셀 위치들, 1/4 픽셀 위치들, 또는 다른 분수 위치들일 수 있다. 분수 픽셀 위치들은 쌍방 선형 보간과 같은 보간에 의해 정수 (전체 픽셀) 위치들로부터 획득될 수 있다.

[0056] 현재 블록의 쌍방 예측에서, 참조 리스트 L0의 각각의 제1 모션 벡터 및 참조 리스트 L1의 제2 모션 벡터를 사용하여 획득되는 2개의 예측 블록들은, 단일 예측 신호에 조합되며, 이는 단일 예측보다 원본 신호에 대한 더 양호한 적응을 제공할 수 있어서, 더 적은 잔차 정보 및 가능하게는 더 효율적인 압축을 초래한다.

[0057] 디코더에서, 현재 블록은 디코딩되고 있기 때문에 이용가능하지 않으며, 모션 벡터 정제의 목적을 위해, 템플릿이 사용되며, 이는 현재 블록의 추정이고 이미 처리된(즉, 인코더 측에서 코딩된 그리고 디코더 측에서 디코딩된) 이미지 부분들에 기초하여 구성된다.

[0058] 먼저, 제1 모션 벡터 MV0의 추정 및 제2 모션 벡터 MV1의 추정이 디코더(200)에서 입력으로서 수신된다. 인코더(100)에서, 모션 벡터 추정들 MV0 및 MV1은 블록 매칭에 의해 및/또는 (동일한 화상에서의 또는 인접하는 화상들에서의) 현재 블록에 이웃하는 블록들의 모션 벡터들에 의해 형성되는 (병합 리스트와 같은) 후보들의 리스트에서 검색하는 것에 의해 획득될 수 있다. 다음으로 MV0 및 MV1은 비트스트림 내에서 디코더 측으로 유리하게 시그널링된다. 그러나, 일반적으로, 인코더에서의 제1 결정 스테이지 또한 시그널링 오버헤드를 감소시키는 이점을 제공할 템플릿 매칭에 의해 수행될 수 있다는 점이 주목된다.

[0059] 디코더(200)에서, 모션 벡터들 MV0 및 MV1은 비트스트림에서의 정보에 기초하여 유리하게 획득된다. 모션 벡터들 MV0 및 MV1은 직접 시그널링되거나, 또는 상이하게 시그널링되고, 및/또는 모션 벡터들의 리스트(병합 리스트)에서의 인덱스가 시그널링된다. 그러나, 본 개시내용은 비트스트림에서 모션 벡터들을 시그널링하는 것으로 제한되는 것은 아니다. 오히려, 본 개시내용에 따르면, 모션 벡터는, 인코더의 동작에 대응하여, 모션 벡터 추정의 제1 스테이지에서 이미 템플릿 매칭에 의해 결정될 수 있다. 제1 스테이지의 템플릿 매칭(모션 벡터 도출)은 제2 모션 벡터 정제 스테이지의 검색 공간과는 상이한 검색 공간에 기초하여 수행될 수 있다. 특히, 모션 벡터 정제는 더 높은 해상도(즉, 검색 위치들 사이의 더 짧은 거리)로 검색 공간에 대해 수행될 수 있다.

[0060] 각각의 MV0 및 MV1이 포인팅하는, 2개의 참조 화상들 RefPic0 및 RefPic1의 표시가 마찬가지로 디코더에 제공된다. 참조 화상들은 이전의 처리, 즉, 각각의 인코딩 및 디코딩의 결과로서 인코더 및 디코더에서 디코딩된 화상 버퍼에 저장된다. 이러한 참조 화상들 중 하나는 검색에 의한 모션 벡터 정제를 위해 선택된다. 모션 벡터들의 결정을 위한 장치의 참조 화상 선택 유닛은 MV0이 포인팅하는 제1 참조 화상 및 MV1이 포인팅하는 제2 참조 화상을 선택하도록 구성된다. 선택에 후속하여, 참조 화상 선택 유닛은 제1 참조 화상 또는 제2 참조 화상이 모션 벡터 정제를 수행하기 위해 사용되는지를 결정한다. 모션 벡터 정제를 수행하기 위해, 제1 참조 화상에서의 검색 영역은 모션 벡터 MV0이 포인팅하는 후보 위치 주위에 정의된다. 검색 영역 내의 후보 검색 공간 위치들은 검색 공간 내에서 템플릿 매칭을 수행하는 것 및 SAD(sum of absolute differences)와 같은 유사성 메트릭을 결정하는 것에 의해 템플릿 블록과 가장 유사한 블록을 발견하도록 분석된다. 검색 공간의 위치들은 템플릿의 상부 좌측 코너가 매칭되는 위치들을 표기한다. 위에 설명된 바와 같이, 좌측 상부 코너는 단순한 관계이고 센터 포인트와 같은 검색 공간의 임의의 포인트가 매칭 위치를 표기하기 위해 일반적으로 사용될 수 있다.

[0061] 도 4a는 단일 예측에 또한 적용가능한 대안적 템플릿 매칭을 예시한다. 상세사항들은 문헌 JVET-A1001에서, 특히 Jianle Chen 등이 저술한, "Algorithm Description of Joint Exploration Test Model 1"이라는 명칭의 문헌

JVET-A1001의 Section 2.4.6. "Pattern matched motion vector derivation"에서 발견될 수 있고 <http://phenix.it-sudparis.eu/jvet/>에서 액세스가능하다. 이러한 템플릿 매칭 접근법에서의 템플릿은 현재 프레임에서의 현재 블록에 인접하는 샘플들로서 결정된다. 문헌 JVET-A1001의 도 1에 도시되는 바와 같이, 현재 블록의 상부 및 좌측 경계에 인접하는 이미 재구성된 샘플들이 취해질 수 있고, "L자형 템플릿(L-shaped template)"이라고 지칭된다.

[0062] 참조로 원용되는, 문헌 JVET-D0029에 따르면, DMVR(decoder-side motion vector refinement)은 2개의 각각의 참조 화상들 RefPic0 및 RefPic1을 포인팅하는 초기 모션 벡터들 MV0 및 MV1을 입력으로서 갖는다. 이러한 초기 모션 벡터들은 RefPic0 및 RefPic1에서 각각의 검색 공간들을 결정하기 위해 사용된다. 또한, 모션 벡터들 MV0 및 MV1을 사용하여, 템플릿은 다음과 같이 MV0 및 MV1에 의해 포인팅되는 (샘플들의) 각각의 블록들 A 및 B에 기초하여 구성된다:

[0063]  $Template = function (Block A, Block B).$

[0064] 이러한 함수는 샘플별 가중 합산과 조합되는 샘플 클리핑 동작일 수 있다. 다음으로 이러한 템플릿은 각각의 참조 화상들 Refpic 0 및 Refpic 1에서의 MV0 및 MV1에 기초하여 결정되는 검색 공간들에서 템플릿 매칭을 수행하기 위해 사용된다. 각각의 검색 공간들에서 최상의 템플릿 매칭을 결정하기 위한 비용 함수는 SAD(Template, Block candA')이며, 여기서 블록 candA'는 MV0에 의해 주어지는 위치 상에 걸치는 검색 공간에서 후보 MV에 의해 포인팅되는 후보 코딩 블록이다. 도 3은 최상의 매칭 블록 A' 및 결과적인 정제된 모션 벡터 MV0'의 결정을 예시한다. 대응하여, 도 3에 도시되는 바와 같이, 최상의 매칭 블록 B' 및 블록 B'를 포인팅하는 대응하는 모션 벡터 MV1'을 발견하기 위해 동일한 템플릿이 사용된다. 다시 말해서, 초기 모션 벡터들 MV0 및 MV1에 의해 포인팅되는 블록 A 및 B에 기초하여 템플릿이 구성된 후에, 정제된 모션 벡터들 MV0' 및 MV1'은 템플릿과의 RefPic0 및 RefPic1에 대한 검색을 통해 발견된다.

[0065] 모션 벡터 도출 기술들은 때때로 FRUC(frame rate up-conversion)라고 또한 지칭된다. 초기 모션 벡터들 MV0 및 MV1은 인코더 및 디코더가 모션 벡터 정제를 위해 동일한 초기 포인트를 사용할 수 있다는 점을 보장하기 위해 비트스트림에서 일반적으로 표시될 수 있다. 대안적으로, 초기 모션 벡터들은 하나 이상의 초기 후보를 포함하는 초기 후보들의 리스트를 제공하는 것에 의해 획득될 수 있다. 이들 각각에 대해, 정제된 모션 벡터가 결정되고, 결국, 가장 낮은 비용 함수를 갖는 정제된 모션 벡터가 선택된다.

[0066] 위에 설명된 바와 같이, 템플릿 매칭된 모션 벡터 도출 모드는 FRUC(Frame-Rate Up Conversion) 기술들에 기초하는 특수 병합 모드이다. 이러한 모드로, 디코더 측에서 블록의 모션 정보가 도출된다. 문헌 JVET-A1001(<http://phenix.it-sudparis.eu/jvet/>에서 액세스가능한, "Description of Joint Exploration Test Model 1"에서 설명되는 구체적 구현에 따르면, 병합 플래그가 참일 때 CU 또는 PU에 대해 FRUC 플래그가 시그널링된다. FRUC 플래그가 거짓일 때, 병합 인덱스가 시그널링되고 정규 병합 모드가 사용된다. FRUC 플래그가 참일 때, 추가적인 FRUC 모드 플래그가 시그널링되어 어느 방법(쌍방 매칭 또는 템플릿 매칭)이 블록에 대한 모션 정보를 도출하기 위해 사용될지를 표시한다.

[0067] 요약하면, 모션 벡터 도출 프로세스 동안, 쌍방 매칭 또는 템플릿 매칭에 기초하여 전체 PU(Prediction Unit)에 대해 초기 모션 벡터가 먼저 도출된다. 먼저, 예를 들어, PU의 병합 리스트일 수 있는, MV 후보들의 리스트가 생성된다. 이러한 리스트는 체크되고, 최소 매칭 비용으로 이어지는 후보가 시작 포인트(초기 모션 벡터)로서 선택된다. 다음으로 시작 포인트 주위의 쌍방 매칭 또는 템플릿 매칭에 기초하는 국부적 검색이 수행되고, 최소 매칭 비용을 초래하는 MV(Motion Vector((s)))가 PU에 대한 MV로서 취해진다. 다음으로 모션 정보는 도출된 PU 모션 벡터들로 시작 포인트들로서 추가로 정제된다. PU(prediction unit) 및 CU(coding unit)라는 용어들은 화상(프레임) 내의 샘플들 블록을 설명하기 위해 본 명세서에서 교환가능하게 사용될 수 있다.

[0068] 도 4b에 도시되는 바와 같이, (문헌 JVET-A1001에 설명되는) 쌍방 매칭은 2개의 상이한 참조 화상들에서의 현재 CU의 모션 궤적을 따라 2개의 블록들 사이의 가장 가까운 매칭을 발견하는 것에 의해 현재 CU의 모션 정보를 도출하기 위해 사용된다. 연속 모션 궤적의 가정 하에, 2개의 참조 블록들을 포인팅하는 모션 벡터들 MV0 및 MV1은 현재 화상과 2개의 참조 화상들 사이의 시간적 거리들, 즉, TD0 및 TD1에 비례할 것이다. 따라서, 본 개시 내용의 하나의 실시예에서, 각각의 테스트된 벡터들의 후보 쌍에서, 2개의 각각의 벡터들은 이미지 평면에서의 직선 상에 있다. 특수한 경우로서, 현재 화상이 시간적으로 2개의 참조 화상들 사이에 있고 현재 화상으로부터 2개의 참조 화상들까지의 시간적 거리가 동일할 때, 쌍방 매칭은 미리 기반 양방향 MV가 된다.

[0069] 도 4a에 도시되는 바와 같이, (문헌 JVET-A1001에 설명되는) 템플릿 매칭은 현재 화상에서의 템플릿(현재 CU의

상부 및/또는 좌측 이웃 블록들)과 참조 화상에서의 블록(템플릿과 동일한 크기임) 사이의 가장 가까운 매칭을 발견하는 것에 의해 현재 CU의 모션 정보를 도출하기 위해 사용된다. 문헌 JVET-A1001의 "패턴 패칭된 모션 벡터 도출(Pattern matched motion vector derivation)" 섹션은 템플릿 매칭 및 쌍방 매칭 방법들의 구체적 구현을 설명한다. "블록 병합(block merging)" 동작 모드가 선택되는 점을 표시하는, "병합 플래그(merge flag)"가 참인 경우에만, 쌍방 매칭 동작이 적용되는 예가 개시된다. 여기서 문헌 JVET-A1001의 저자들은 H.265 표준의 "병합 모드(merge mode)"를 참조한다. JVET-A1001에 설명되는 템플릿 매칭 및 쌍방 매칭 방법들은 다른 비디오 코딩 표준들에 또한 적용될 수 있어, 구체적 구현에서의 변동들을 초래한다는 점이 주목된다.

[0070] 도 5는 디코더 측 DMVR(motion vector refinement) 동작을 예시하는 흐름도이다. 문헌 JVET-D0029에 따르면, DMVR은 2개의 조건들: 1) 예측 타입이 스킵 모드 또는 병합 모드로 설정됨, 2) 예측 모드가 쌍방 예측임 하에서 적용된다. 제1 초기 모션 벡터들 MV0(참조 리스트 L0의 것) 및 MV1(참조 리스트 L1의 것)이 도출된다. 도출 프로세스는 각각의 스킵 및 병합 동작들에 따라 수행된다. 여기서 문헌 JVET-D0029의 저자들은 H.265 표준의 스킵 모드 및 병합 모드를 참조한다. 이러한 모드들의 설명은 v. Sze, M. Budagavi 및 G.J. Sullivan (Ed.)이 저술한 책, High Efficiency Video Coding (HEVC), Algorithms and Architectures, 2014의 Section 5.2.2.3 "Motion Data Signaling and Skip Mode"에서 발견될 수 있다. H.265에서 스킵 모드는 블록에 대해 모션 데이터가 명시적으로 시그널링되는 대신에 추론된다는 점 및 예측 잔차가 제로라는, 즉, 어떠한 변환 계수들도 송신되지 않는다는 점을 표시하기 위해 사용된다. 병합 모드가 선택되면, 모션 데이터가 또한 추론되지만, 예측 잔차는 제로가 아니다, 즉, 변환 계수들이 명시적으로 시그널링된다.

[0071] 입력 비디오 스트림으로부터 파싱 인덱스가 파싱된다(510). 파싱된 인덱스는 구성되는 MV 후보 리스트(520)의 최상의 모션 벡터 후보에 포인팅한다. 다음으로 최상의 모션 벡터 후보가 선택되고(530), 가중 평균화에 의해 템플릿이 획득된다(540). DMVR(550)이 다음과 같이 적용된다. 도 3을 참조하여 위에 설명된 바와 같이 MV0 및 MV1에 의해 참조되는 블록들을 함께 추가하는 것에 의해 블록 템플릿이 계산된다. 클리핑이 이후 수행된다. 이러한 템플릿은 초기 모션 벡터 MV0 주위의 정제된 모션 벡터 MV0'을 발견하기 위해 사용된다. 검색 영역은 정수 화소 해상도이다(검색 공간의 포인트들은 정수 샘플 거리만큼 서로로부터 이격됨). SAD(Sum of Absolute Differences) 비용 척도는 템플릿 블록과 MV0'에 의해 포인팅되는 새로운 블록을 비교하기 위해 사용된다. 이러한 템플릿은, MV0' 주위의, 정제된 MV0'를 발견하기 위해 사용된다. 검색 영역은 1/2 화소 해상도이다(검색 공간의 포인트들은 샘플 거리의 1/2만큼 서로로부터 이격됨). "화소(pel)" 및 "픽셀(pixel)"이라는 용어들은 본 명세서에서 교환가능하게 사용된다. 동일한 비용 척도가 사용된다. MV1'을 발견하기 위해 후자의 2개의 단계들이 반복된다. MV0' 및 MV1'에 의해 포인팅되는 블록들을 함께 추가하는 것에 의해 새로운 쌍방 예측된 블록이 형성된다. 다음으로 이러한 정제된 모션 벡터들 MV0'' 및 MV1''에 의해 포인팅되는 블록들 block\_A' 및 block\_B'는 최종 예측을 획득하기 위해 평균화된다, 예를 들어, 가중 평균화된다(560).

[0072] 도 6은 본 개시내용의 실시예에 따른 비디오 코딩 및 디코딩에 이용될 수 있는 LIC(local illumination compensation)의 예를 예시하는 개략도이다. LIC(Local Illumination Compensation)는, 스케일링 인자 "a" 및 오프셋 "b"를 사용하여, 조명 변경들에 대한 선형 모델에 기초한다. LIC는 각각의 인터 모드 코딩된 CU(coding unit)에 대해 적응적으로 인에이블되거나 또는 디스에이블될 수 있다. LIC가 CU에 대해 적용될 때, 현재 CU의 이웃 샘플들 및 그들의 대응하는 참조 샘플들을 사용하여 파라미터들 a 및 b를 도출하기 위해 최소 제곱 에러 방법(least square error method)이 이용될 수 있다. 더 구체적으로, 도 6에 예시되는 바와 같이, CU의 서브 샘플링된 (2:1 서브샘플링) 이웃 샘플들 및 참조 화상에서의 대응하는 샘플들(현재 CU 또는 서브 CU의 모션 정보에 의해 식별됨)이 사용된다. LIC 파라미터들은 각각의 예측 방향에 대해 별도로 도출되고 적용된다. 여기서, 서브샘플링 2:1은 현재 CU 경계 및 참조 블록 상의 모든 두번째 픽셀이 취해진다는 점을 의미한다. 스케일링 인자/곱셈형 가중 인자 및 오프셋의 사용에 대한 더 많은 상세사항들은 문헌 JVET-A1001의 Section "2.4.4. Local illumination compensation"에서 발견될 수 있다.

[0073] 도 7은 참조 화상 RefPic0에 대해 수행되는 DMVR(decoder-side motion vector refinement) 반복을 예시하는 개략도이다. 현재 화상은 RefPic0에서 모션 벡터 MV0에 기초하여 모션 벡터 MV0'이 발견되어야 하는 현재 블록(710)을 포함한다. 5개의 정수 위치들을 포함하는 검색 공간이 결정되고; 후보 위치에 의해 포인팅되는 블록은 Ax라고 지칭된다. 출력은 모션 벡터 MV0'에 의해 포인팅되는 블록들 Ax의 최상의 매칭이다.

[0074] 명시적 병합 모드 인덱스가 시그널링될 때마다, 디코더 측 모션 벡터 정제는 모션 벡터(들) 및 시그널링된 인덱스로부터 규범적으로 추론되는 참조 인덱스들로부터 시작한다. 명시적 병합 모드 인덱스가 시그널링되지 않을 때, 비용 함수를 사용하여 디코더에서 초기 모션 벡터 후보들의 세트가 평가되고, 정제를 위한 시작 포인트로서 비용이 가장 낮은 후보가 선택된다. 따라서, 디코더 측 모션 벡터 도출 방법이 예측된/재구성된 이웃 블록 경



계 샘플들에 기초하는지(보통 TM(Template Matching)이라고 지칭됨, 도 4a 참조) (또는) 참조 리스트 L0 및 참조 리스트 L1에서의 대응하는 패치들 사이의 차이 최소화를 통한 쌍방 매칭에 기초하는지(보통 BM(Bilateral matching) 비용이라고 지칭됨) (또는) 참조 리스트 L0 및 참조 리스트 L1에서의 대응하는 패치들의 평균화 버전과 L0/L1에서의 변위 사이의 차이에 기초하는지(DMVRcost라고 지칭됨)에 무관하게, 서브 픽셀 정밀 모션 벡터들 일 수 있는 시작 포인트들 주위에서 수행될 필요가 있는 정제 검색이 존재한다.

[0075] 비용 함수를 평가하기 위해, 정수 그리드 위치들에서의 참조 프레임들의 값들에 기초하여 서브 픽셀 정밀 센터들에서의 값들을 도출하기 위해 보간이 수행될 필요가 있다. 보간 필터는 쌍선형 보간 필터만큼 단순할 수 있거나 또는 2-D DCT 기반 분리가능 보간 필터들과 같은 더 긴 필터일 수 있다. 블록에 대한 보간된 샘플들을 다시 그리고 정제 동안 고려되는 각각의 위치에서 다시 도출하는 복잡성을 감소시키기 위해, L0 및/또는 L1에서의 서브 픽셀 정밀 위치(들)에 센터를 두는 정제 포인트들의 정수 픽셀 거리 그리드가 다른 발명에서 제안되었다. 이러한 것에 의해, 현재 최상의 비용 위치에 가까운 새로운 위치가 고려됨에 따라 증분적 보간들만이 수행될 필요가 있다. 정수 픽셀 거리 그리드의 정제가 완료된 후에, 병합 MV들에 관하여 최상의 정수 델타 MV들이 획득된다.

[0076] 압축 이득들을 추가로 개선하기 위해, 서브 픽셀 거리 정제가 수행될 수 있다. 1/2 픽셀 거리 정제는 참조 프레임(들)에서의 최상의 정수 거리 MV 위치(들)로부터 1/2 픽셀 거리에 보간된 샘플들을 요구한다. L0에서의 최상의 정수 거리 MV 위치에 관하여 L0에서의 변위들이 무효화되어 L1에서의 최상의 정수 거리 MV 위치에 관하여 L1에서의 대응하는 변위들을 획득하는 경우에 L0과 L1 사이에 서브 픽셀 정밀 정제를 합동으로 수행하는 것이 가능하다. L0 및 L1에서 독립적으로 서브 픽셀 정밀 정제를 수행하는 것이 또한 가능하다.

[0077] 도 8은 본 개시내용의 실시예에 따른 정수 및 분수 샘플 위치들의 예를 예시하는 개념도이다. 도 8을 참조하면, "A"로서 표기되는 픽셀 위치들은 정수 픽셀들이고, 1/2 픽셀 위치들(장소들)은 b, h 및 j로서 표기되며, 모든 1/4 픽셀 위치들(장소들)은 a, c, d, e, f, g, i, k, n, p, q 및 r로서 표기된다. 8-포인트 정사각형 패턴으로 1/2 픽셀 정제를 행하기 위해, 3개의 평면들이 요구되고, 유사하게 1/4 픽셀 정제에 대해 8개의 평면들이 HEVC 표준에 따라 요구된다. 더 새로운 표준들은 1/16의 픽셀 정밀 보간들을 고려하고 있다.

[0078] "A"가 정수 픽셀일 때, 픽셀 장소들 a, b 및 c는 수평 보간만을 요구하고, d, h 및 n은 수직 보간만을 요구하고, 모든 다른 포인트들은 수직 및 수평 보간들 양자 모두를 요구한다.

[0079] 정수 픽셀 거리 정제 및 서브 픽셀 거리 정제 동안 최종 모션 보상된 예측 프로세스에 대해 사용되는 보간과는 상이한 보간 프로시저를 사용하는 것이 가능하다. 예를 들어, 쌍선형 보간은 정제에 대해 사용될 수 있는 더 단순한 보간인 한편 최종 MCP(motion compensated prediction)는 2-D DCT 기반 보간 필터를 요구할 수 있다. 참조 프레임 버퍼들로부터 폐치되는 정수 픽셀 그리드 데이터는 외부 메모리(예를 들어, DDR)로부터 이러한 데이터를 여러 번 다시 폐치하는 것을 회피하기 위해 모든 보간들이 완료될 수 있을 때까지 내부 메모리(예를 들어, SRAM)에서의 제1 버퍼에 유지될 필요가 있다. 정수 픽셀 그리드 샘플들의 보간을 통해 도출되는 그리고 서브 픽셀 정밀 정제 센터들로부터의 정수 픽셀 거리 정제에 대해 요구되는 정수 픽셀 거리 그리드는 모든 정수 픽셀 거리 정제들이 완료될 때까지 내부 메모리에서의 제2 버퍼에 유지될 필요가 있다. (정수 픽셀 그리드에 관하여 서브 픽셀 위치일 수 있는) 최상의 정수 픽셀 거리 위치 주위의 1/2 픽셀 거리 정제에 대해, 센터의 양측 상의 대칭 1/2 픽셀 정제 포인트들 사이의 거리가 1개의 정수 픽셀 거리만큼 분리된다는 점을 고려하면, 2개의 수평 1/2 픽셀 거리 위치들이 보간된 평면을 공유하고, 2개의 수직 1/2 픽셀 거리 위치들이 보간된 평면을 공유하며, 4개의 대각선 1/2 픽셀 거리 위치들이 보간된 평면을 공유한다.

[0080] 정제를 위한 보간이 최종 MCP 보간과 동일할 때, 최상의 정수 픽셀 거리 위치가 서브 픽셀 정제 후 최상의 위치인 것으로 밝혀질 때 최종 MCP 보간 계산이 회피될 수 있으므로 제2 버퍼에 저장되는 정수 픽셀 거리 정제 데이터를 유지하는 것이 바람직할 수 있다. 정제와 최종 MCP 사이에서 보간들이 상이할 때, 제2 버퍼는 1/2 픽셀 거리 정제를 위해 필요한 3개의 평면들 중 하나에 대한 보간들로 덮어쓰기될 수 있다.

[0081] 계산 부하 및 내부 메모리 필요성을 감소시키기 위해, 일부 종래 기술은 4개의 대각선 1/2 픽셀 거리 위치들만을 평가하고 한편 일부 다른 종래 기술은 수평 및 수직 1/2 픽셀 거리 위치들만을 평가한다. 따라서, 요구되는 내부 메모리 버퍼들의 세트는 (대각선 1/2 픽셀 거리 위치들만이 평가되고 보간들이 상이할 때의) 2 내지 (평가들이 모든 3개의 1/2 픽셀 거리 평면들에 대해 수행되고 보간들이 상이하지 않을 때의) 5의 범위에 있다. 일부 종래 기술의 기술들이 추가적 1/4 픽셀 거리 및 1/8 픽셀 거리 정제들을 수행하는 한편, 이러한 정제들에서의 각각의 위치는 개별 보간들을 요구하며, 이들은 계산적으로 엄청나며 추가적 압축 이득들과 부합하지 않는다. 보간이 더 높은 탭 필터들을 사용할 때, 수평 필터링이 먼저 수행되고 필터링된 결과가 임시 버퍼에 저장되고

다음으로 수직 필터링이 수행된다.

- [0082] 본 개시내용의 실시예들은 정수 픽셀 거리 그리드 위치들에서 평가되는 비용들을 이용하는 에러 표면 기술들을 통한 명시적 정제 없이 디코더 측 모션 벡터 정제/도출 동안 최상의 정수 거리 모션 벡터를 주위의 서브 픽셀 거리 델타 모션 벡터들을 도출하기 위한 방법들 및 장치들을 제공한다. 파라메트릭 에러 표면은 이러한 비용들을 사용하여 맞추어지고 파라메트릭 에러 표면의 최소 위치는 연립 방정식의 세트를 해결하는 것에 의해 획득된다. 본 개시내용의 양태들은 (자체가 정수 픽셀 그리드에 관하여 서브 픽셀 위치일 수 있는) 정수 거리 픽셀 위치 주위에서 디코더 측 서브 픽셀 거리 정제를 이용하고, 템플릿 매칭, 쌍방 매칭, 및 템플릿 기반 쌍방 매칭과 같은 디코더 측 모션 벡터 정제의 상이한 변형들을 핸들링한다.
- [0083] 디코더 측 모션 벡터 정제/도출이 코딩 시스템의 규범적 양태인 점을 고려하면, 인코더는 인코더의 재구성과 디코더의 재구성 사이에 어떠한 드리프트도 갖지 않기 위해 동일한 에러 표면 기술을 또한 수행해야 할 것이다. 그러므로, 본 개시내용의 모든 실시예들의 모든 양태들은 인코딩 및 디코딩 시스템들 양자 모두에 적용가능하다.
- [0084] 템플릿 매칭에서, 정제 움직임은 명시적으로 시그널링된 병합 인덱스에 기초하여 또는 비용 평가들을 통해 암시적으로 도출되는 서브 픽셀 정밀 센터로부터 시작하는 참조에서만 발생한다.
- [0085] (평균화된 템플릿이 있는 또는 없는) 쌍방 매칭에서, 명시적으로 시그널링된 병합 인덱스에 기초하여 또는 비용 평가들을 통해 암시적으로 도출되는 각각의 서브 픽셀 정밀 센터들로부터 시작하는 참조 리스트들 L0 및 L1에서 정제들이 시작된다.
- [0086] 참조 리스트 L0에서의 주어진 수평 및 수직 변위에 대해 참조 리스트 L1에서의 동일한 그리고 반대인 수평 및 수직 변위들에 기초하여 쌍방 매칭 비용이 평가될 때, 도면들에 도시되는 위치들은 참조 리스트 L0에 대응하는 것으로 가정되고, 참조 리스트 L1에서의 위치들은 참조 리스트 L1에서의 현재 반복 센터에 관하여 수평 및 수직 방향에서의 변위를 무효화하는 것에 의해 도출된다는 점이 주목되어야 한다.
- [0087] 그러므로, 메모리 크기 및 계산 복잡성을 증가시키지 않고 명시적 서브 픽셀 거리 정제의 코딩 이득들의 대부분을 실현할 필요가 있다.
- [0088] 실시예 1
- [0089] N을 규범적으로 허용되는 정수 1-픽셀 거리 정제 반복들의 최대 수라고 한다. 제1 정수 거리 정제 반복은 이전에 설명된 정제의 서브 픽셀 정밀 센터로부터 시작하고, 센터 위치가 1-픽셀 이웃 위치들의 규정된 세트의 평가된 비용 함수 값들보다 더 낮은 평가된 비용 함수 값을 가질 때까지 또는 N 수의 반복들이 수행될 때까지 진행한다. N 수의 반복들이 도달되고 센터 위치가 그 1-픽셀 이웃 위치들의 세트에 비교될 때 비용 함수 값이 최소인(가장 낮은) 위치가 아닐 때, 에러 표면 기반 서브 픽셀 정제가 수행되지 않고, 모든 반복들에 걸쳐 비용 함수 값이 최소인(가장 낮은) 위치가 디코더 측 모션 벡터 정제 프로세스로부터의 최종 델타 모션 벡터로서 선언된다. 1-픽셀 이웃 위치들의 이탈 반복의 세트에서 최소(가장 낮은) 비용 함수 값을 갖는 센터 위치에 의해 정제가 이탈될 때, 다음의 에러 표면 기반 서브 픽셀 정제 프로시저가 적용된다.
- [0090] 다음에서, 센터 C 주위의 1 픽셀 이웃 위치들의 세트를 그 L(left), T(top), R(right), 및 B(bottom) 정수 1 픽셀 거리라고 한다. 도 9는 본 개시내용의 실시예에 따른 하나 이상의 참조 프레임에서 서브 픽셀 정밀 델타 모션 벡터 정제를 획득하기 위한 센터 픽셀 주위의 1 픽셀 이웃 위치들의 세트의 블록도이다. 도 9를 참조하면, 2-D 좌표계의 원점은 C에 있고, L, T, R, B에서의 좌표들은 (-1,0), (0,1), (1,0), 및 (0,-1)에 대응한다. 이러한 5개의 위치들에서의 평가된 비용 함수 값들은 E(0,0), E(-1,0), E(0,1), E(1,0), 및 E(0,-1)일 것이다. 2-D 포물선 함수를 사용하는 5-파라미터 에러 표면은 다음과 같이 정의된다:
- [0091] 
$$E(x,y) = A*(x-x_0)^2 + B*(y-y_0)^2 + C \quad (1)$$
- [0092] 여기서, E(x,y)는 데카르트 좌표 (x,y)의 평가된 비용 함수이고, (x<sub>0</sub>, y<sub>0</sub>)는 최소(가장 낮은) 에러를 갖는 센터 (0, 0)에 관하여 서브 픽셀 변위와 연관된 (이에 대응하는) 데카르트 좌표이고, C는 이러한 위치에서의 에러에 대응하는 파라미터이고, A 및 B는 에러 표면 모델 파라미터들이다. A, B, 및 C는 E(x, y)를 계산하기 위한 상수 값들이다.
- [0093] 다른 한편, A, B, C, x<sub>0</sub> 및 y<sub>0</sub>이 알려지지 않은 파라미터들인 점을 고려하면, 5개의 위치들 (0, 0), (-1, 0), (0,-1), (1, 0) 및 (0, 1)에서 이용가능한 평가된 비용 함수 값들을 사용하여 5개의 수학식들을 해결하여, (x<sub>0</sub>,

y0)는 다음과 같이 계산(결정)될 수 있다:

$$[0094] \quad x0 = (E(-1,0) - E(1,0)) / (2*(E(-1,0) + E(1,0) - 2* E(0,0))) \quad (2)$$

$$[0095] \quad y0 = (E(0,-1) - E(0,1)) / (2*(E(0,-1) + E(0,1) - 2* E(0,0))) \quad (3)$$

[0096] x0 및 y0은 서브 픽셀 모션 벡터 변위와 연관된 좌표이다. E(-1, 0), E(1, 0), E(0, 0), E(0,-1) 및 E(0, 1)은, 각각, 초기 모션 벡터에 관하여 후보 정수 모션 벡터 변위들 (-1, 0), (1, 0), (0, 0), (0,-1) 및 (0, 1)에 대응하는 정수 거리 비용들이다. 좌표 x0 및 y0은 델타 모션 벡터의 서브 픽셀 정밀도에 따라 스케일 인자 N에 의해 스케일링될 수 있으며, 여기서, 예를 들어, 픽셀 정밀도의 1/2, 1/4, 1/8, 및 1/16에 대해 각각 N=2, 4, 8, 및 16이다.

[0097] 다른 실시예에서, 수학적식들 (2) 및 (3)은 다음과 같이 표현될 수 있다:

$$[0098] \quad x0 = (E(-1,0) - E(1,0)) / (2*N*(E(-1,0) + E(1,0) - 2* E(0,0))) \quad (2')$$

$$[0099] \quad y0 = (E(0,-1) - E(0,1)) / (2*N*(E(0,-1) + E(0,1) - 2* E(0,0))) \quad (3')$$

[0100] 여기서, 예를 들어, 픽셀 정밀도의 1/2, 1/4, 1/8 및 1/16에 대해 각각 N=1, 2, 4 및 8이다.

[0101] 수학적식들 (2) 및 (3)에 기초하여, x0은 위치들 (-1, 0), (1, 0) 및 (0, 0)에 따라서만 계산(결정)될 수 있다는 점이 주목된다. 유사하게, y0은 위치들 (0,-1), (0, 1) 및 (0, 0)에 따라서만 계산(결정)될 수 있다.

[0102] (초기 모션 벡터에 대한 최상의 정수 거리 변위에 대응하는) 센터 (0, 0)에 관련된 모든 4개의 위치들 (-1, 0), (0,-1), (1, 0), 및 (0, 1)에서의 평가된 비용들이 이용가능하지 않을 때 파라메트릭 에러 표면이 맞추어질 수 없다는 점이 또한 주목된다. 이러한 경우들에서, 어떠한 서브 픽셀 정밀 델타 변위도 최상의 정수 거리 변위에 추가되지 않는다.

[0103] 최종 모션 보상에 의해 지원되는 서브 픽셀 정밀도에 의존하여, 스케일 인자 N이 적절하게 선택될 수 있다.

[0104] 서브 픽셀 모션 벡터 변위들은 수학적식들 (1), (2) 및 (3)에 따라 x 및 y 방향들에서 -0.5 및 +0.5에 의해 한정되고, 1/16 픽셀까지의 정밀도가 일반적으로 관심이 있기 때문에, 시프트, 비교, 및 증분 동작들만으로 2개의 분할들이 효율적으로 수행될 수 있다.

[0105] 일부 실시예들에서, 디코더 측 모션 벡터 정제를 위한 방법은 초기 모션 벡터에 관하여 후보 정수 모션 벡터 변위들에 대응하는 정수 거리 비용들을 비교하는 것에 의해 타겟 정수 모션 벡터 변위를 결정하는 단계, 정수 거리 비용들에 대해 계산을 수행하는 것에 의해 서브 픽셀 모션 벡터 변위를 결정하는 단계, 및 타겟 정수 모션 벡터 변위, 서브 픽셀 모션 벡터 변위 및 초기 모션 벡터에 기초하여 정제된 모션 벡터를 결정하는 단계를 포함한다.

[0106] 하나의 실시예에서, 이러한 방법은, 서브 픽셀 모션 벡터 변위를 결정하는 단계 전에, 미리 결정된 모션 벡터 변위가 타겟 정수 모션 벡터 변위를 포함하는지를 결정하는 단계; 미리 결정된 모션 벡터 변위가 타겟 정수 모션 벡터 변위를 포함하는 경우에: 정수 거리 비용들에 대한 계산을 수행하는 것에 의해 서브 픽셀 모션 벡터 변위를 결정하는 단계를 추가로 포함할 수 있다. 다른 실시예에서, 이러한 방법은, 서브 픽셀 모션 벡터 변위를 결정하는 단계 전에, 평가된 비용들이 타겟 정수 모션 벡터 변위에 관련된 위치들 (-1,0), (0,-1), (1,0), 및 (0,1)에서 이용가능함을 결정하는 단계; 평가된 비용들이 타겟 정수 모션 벡터 변위에 관련된 위치들 (-1,0), (0,-1), (1,0) 및 (0,1)에서 이용가능하다고 결정되는 경우에, 정수 거리 비용들에 대해 계산들을 수행하는 것에 의해 서브 픽셀 모션 벡터 변위를 결정하는 단계를 추가로 포함할 수 있다.

[0107] 하나의 실시예에서, 이러한 방법은, 미리 결정된 모션 벡터 변위가 타겟 정수 모션 벡터 변위를 포함하지 않는 경우에: 타겟 정수 모션 벡터 변위 및 초기 모션 벡터에 기초하여 정제된 모션 벡터를 계산하는 단계를 추가로 포함할 수 있다. 다른 실시예에서, 이러한 방법은, 평가된 비용들 중 적어도 하나 이상이 타겟 정수 모션 벡터 변위에 관련된 위치들 (-1,0), (0,-1), (0,1), 및 (1,0)에서 이용가능하지 않다고 결정되는 경우에, 타겟 정수 모션 벡터 변위 및 초기 모션 벡터에 기초하여 정제된 모션 벡터를 계산하는 단계를 추가로 포함할 수 있다.

[0108] 하나의 실시예에서, 타겟 정수 모션 벡터 변위를 결정하는 단계는 각각의 후보 정수 모션 벡터 변위에 대한 정수 거리 비용을 계산하는 단계, 및 가장 낮은 정수 거리 비용에 대응하는 후보 정수 모션 벡터 변위를 타겟 정수 모션 벡터 변위로서 선택하는 단계를 포함할 수 있다.

[0109] 하나의 실시예에서, 타겟 정수 모션 벡터 변위는 참조 화상 리스트 L0에 대응하는 제1 모션 벡터 변위, 및 참조

화상 리스트 L1에 대응하는 제2 모션 벡터 변위를 포함한다. 이러한 방법은, 참조 화상 리스트 L0에 대응하는 후보 정수 모션 벡터 변위들에 대응하는 정수 비용들을 비교하는 것에 의해 제1 모션 벡터 변위를 결정하는 단계, 및 제1 모션 벡터 변위를 무효화하는 것에 의해 제2 모션 벡터 변위를 결정하는 단계를 추가로 포함할 수 있다.

- [0110] 설명된 실시예는 예시적이며 제한적이지는 않고, 정수 거리 정제 프로세스가 1-픽셀 이웃 세트 내에서 이러한 많은 평가된 비용 함수 값들을 초래하면 참조 문헌들에서와 같은 다른 5-포인트, 6-포인트, 및 9-포인트 기반 에러 표면 방법들이 이용될 수 있다는 점이 이해되어야 한다.
- [0111] 실시예에 따른 방법은 2개의 내부 메모리 버퍼들만을 필요로 한다. 제1 버퍼는 재구성된 화상 버퍼로부터의 정수 픽셀 그리드를 저장하도록 구성되고, 제2 버퍼는 (파라메트릭 에러 표면을 획득하기 위해 이용되는) 정수 거리 정제 동안 정수 거리 그리드를 저장하도록 구성되며, 최종 서브 픽셀 정밀 모션 보상된 예측 샘플들로 덮어쓰기하기 위해 사용될 수 있다.
- [0112] 본 개시내용의 실시예들에 따르면, 서브 픽셀 정제 레벨들 중 어떠한 것에서의 명시적 정제 없이 고 정밀 서브 픽셀 정밀 델타 모션 벡터들이 획득된다.
- [0113] 모든 반복들에 걸쳐 동일한 비용 함수가 사용될 때 정수 거리 정제 반복들에 대해 요구되는 계산들 이외의 추가의 보간 또는 비용 함수 평가 계산들이 요구되지 않는다.
- [0114] 도 10은 본 개시내용의 하나의 실시예에 따른 디코더 측 모션 벡터 정제 시스템에서 하나 이상의 참조 프레임에서 서브 픽셀 정밀 델타 모션 벡터를 획득하기 위한 방법(1000)을 예시하는 단순화된 흐름도이다. 이러한 방법(1000)은 다음의 단계들을 포함할 수 있다:
- [0115] 단계 1001: 프로세서를 제공함. 이러한 프로세서는 본 명세서에 설명되는 방법을 수행하기 위해 비디오 인코더 및/또는 디코더에 집적되는 하나 이상의 처리 유닛(CPU, DSP) 또는 비디오 압축 소프트웨어에 집적되는 프로그램 코드일 수 있다.
- [0116] 단계 1003: 프로세서에 의해, 비용 함수를 사용하여 정수 1-픽셀 거리 정제 동작들(반복들)을 반복적으로 수행하여 하나 이상의 참조 프레임의 각각의 참조 프레임에 대한 정수 거리 정제 모션 벡터를 결정하는 것에 의해 루프를 착수함.
- [0117] 단계 1005: 프로세서에 의해, 1-픽셀 이웃 위치들의 세트의 비용 함수 값에 대해 현재 동작의 검색 센터의 위치의 비용 함수 값을 결정함. 현재 반복의 검색 센터의 비용이 가장 낮다고; 즉, 반복 루프로부터 조기 이탈한다고 결정되는 경우에(1005, 예):
- [0118] 단계 1011: 마지막 검색 센터들의 비용 함수 값들 및 그 주위의 1-픽셀 이웃들의 세트를 사용하여 맞추어진 파라메트릭 에러 표면 상의 값이 최소인 위치를 계산하는 것에 의해 각각의 참조 프레임에서 마지막 검색 센터들 주위의 서브 픽셀 거리 정제 모션 벡터들을 결정하는 단계; 및
- [0119] 단계 1013: 각각의 참조 프레임에 대해 결정된 정수 거리 정제 모션 벡터들 및 결정된 서브 픽셀 거리 정제 모션 벡터들의 합으로서 총 정제 모션 벡터를 리턴함.
- [0120] 이러한 방법은, 검색 센터의 위치의 비용 함수 값이 가장 낮은 비용 함수 값이 아닌 경우에(1005, 아니오), 단계 (1007)로 진행되는 것을 추가로 포함한다.
- [0121] 단계 (1007): 현재 동작이 마지막 동작(반복)이라고 결정함. 현재 동작이 마지막 동작인 경우에(1007, 예), 모든 동작들에 걸쳐 비용 값이 가장 낮은 위치에 대응하는 정제 모션 벡터를 리턴함(1015). 현재 동작이 마지막 동작이 아닌 경우에(1007, 아니오): 센터를 해당 마지막 동작에서 비용이 가장 낮은 위치로 업데이트하고, 루프로 다시 가서 및 단계 (1003)을 반복함.
- [0122] 실시예 2
- [0123] 파라메트릭 에러 표면을 도출하기 위해 이용되는 정수 거리 위치에서의 비용 함수 평가가 정수 거리 정제의 더 조기의 반복 동안 수행되는 비용 함수 평가와 상이할 때 실시예 1의 상이한 변동들이 존재한다.
- [0124] 예를 들어, 쌍방 매칭이 사용될 때, 가능한 3개의 타입들의 정제들이 존재한다. 제1 정제 타입은, 참조 리스트 L1에서의 변위들이 참조 리스트 L0에서의 변위들에 대해 수평 및 수직 방향들 양자 모두에서 동일한 그리고 반대인 방식으로, 참조 리스트들 L0 및 L1 양자 모두에서 공동 정제를 수행하는 것이다(제1 정제 타입은



SBM\_JOINT라고 지칭됨). 제2 정제 타입은 공통 쌍방 평균화된 템플릿에 대해 참조 리스트들 L0 및 L1 양자 모두에서 독립적 정제들을 수행하는 것이다(제2 정제 타입은 TBM\_INDEPENDENT 라고 지칭됨). 제3 정제 타입은 쌍방 평균화된 템플릿에 대해 참조 리스트들 L0 또는 L1에서 정제를 수행하고, 다른 참조 리스트(정제를 위해 L0가 사용되었다면 L1, 정제를 위해 L1이 사용되었다면 L0)에서 변위를 얻을 때 추정된 수평 및 수직 변위들을 무효화하는 것이다. 제3 정제 타입은 TBM\_IN\_1REF\_NEG\_IN\_OTHER)라고 지칭된다.

[0125] (센터 위치의 비용이 1-픽셀 이웃 비용들의 주변 세트보다 더 낮은 것으로 인해) 반복 루프로부터의 조기 이탈이 발생할 반복을 예상하는 것이 어렵기 때문에, 파라메트릭 에러 표면을 도출하기 위해 이용되는 정수 거리 위치들에서의 비용 함수 평가들이 정수 거리 정제의 더 조기의 반복들 동안 수행되는 비용 함수 평가들과 상이할 때마다, 조기 이탈 센터 상에 센터를 두는 비용 함수 평가들의 추가 세트가 요구된 비용 함수로 수행된다.

[0126] 다음의 표 1은, 정수 거리 정제 동안의 조기 이탈 후 추가 평가들이 요구될 쌍방 매칭 동안의 일부 상황들을 제공한다.

표 1

정수 거리 정제에 대해 사용되는 정제의 타입	파라메트릭 에러 표면을 도출하기 위해 정수 거리 정제의 조기 이탈에 대해 (또는) 마지막 반복에 대해 사용되는 정제 타입	조기 이탈 후에 요구되는 추가 비용 함수 평가들?
SBM_JOINT	SBM_JOINT	아니오
SBM_JOINT	TBM_INDEPENDENT	예
SBM_JOINT	TBM_IN_1REF_NEG_IN_OTHER	예
TBM_INDEPENDENT	TBM_INDEPENDENT	아니오
TBM_IN_1REF_NEG_N_OTHER	TBM_INDEPENDENT	예

[0127]

[0128] 정수 거리 정제 반복들 동안 정제 타입 SBM\_JOINT가 최상으로 작동한다는 점이 관찰되었다. L0 및 L1에서의 동일한 그리고 반대인 변위들에 비교될 때, 정제 타입 TBM\_INDEPENDENT가 L0 및 L1에서의 독립적 서브 픽셀 델타 모션 벡터들을 얻기 위해 사용될 때 작은 추가적 코딩 이득이 달성된다. 그러나, L0 및 L1에서의 독립적 정제는 L0 및 L1 위치들에서의 비용 평가들을 독립적으로 요구하고 그러므로 공동 동일한 그리고 반대인 변위 추정 옵션보다 계산적으로 더 복잡하다(SBM\_JOINT 정제). SBM\_JOINT 정제 이외에, 조기 이탈은 추가적 비용 함수 평가들을 요구하지 않는다.

[0129] 도 11은 본 개시내용의 하나의 실시예에 따른 디코더 측 모션 벡터 정제 시스템에서 그들 각각의 초기 서브 픽셀 정밀 정제 센터(들) 주위의 하나 이상의 참조 프레임에서 서브 픽셀 정밀 델타 모션 벡터를 획득하기 위한 방법(1100)을 예시하는 단순화된 흐름도이다. 이러한 방법(1100)은 다음의 단계들을 포함할 수 있다:

[0130] 단계 1101: 프로세서를 제공함. 이러한 프로세서는 본 명세서에 설명되는 방법을 수행하기 위해 비디오 인코더 및/또는 디코더에 집적되는 하나 이상의 처리 유닛(CPU, DSP) 또는 비디오 압축 소프트웨어에 집적되는 프로그램 코드일 수 있다.

[0131] 단계 1103: 프로세서에 의해, 비용 함수를 사용하여 정수 1-픽셀 거리 정제 동작들(반복들)을 반복적으로 수행하여 하나 이상의 참조 프레임의 각각의 참조 프레임에 대한 정수 거리 정제 모션 벡터를 결정함.

[0132] 단계 1105: 현재 동작이 마지막 동작인지를 결정함. 현재 동작이 마지막 동작인 경우에(1105, 예):

[0133] 단계 1111: 제2 비용 함수를 사용하여 정수 1-픽셀 거리 정제를 수행하여 하나 이상의 참조 프레임의 각각의 참조 프레임에 대한 독립적 서브 픽셀 델타 모션 벡터들을 획득함.

[0134] 현재 동작이 마지막 동작이 아닌 경우에(1105, 아니오), 단계 1107로 진행한다.

- [0135] 단계 1107: 현재 동작의 검색 센터의 위치의 비용 함수 값이 1-픽셀 이웃 위치들의 세트에서의 가장 낮은 비용 함수 값인지를 결정함. 검색 센터의 위치의 비용 함수 값이 가장 낮은 비용 함수 값인 경우에(1107, 예): 단계 1111로 진행한다(즉, 반복 루프의 조기 이탈). 검색 센터의 위치의 비용 함수 값이 가장 낮은 비용 함수 값이 아닌 경우에(1107, 아니오): 단계 1109로 진행한다.
- [0136] 단계 1109: 센터를 해당 동작에서 비용 값이 가장 낮은 위치로 업데이트하고 다음 정수 1-픽셀 거리 정제 동작을 수행하기 위한 루프로 다시 감.
- [0137] 단계 1111: 제2 비용 함수를 사용하여 정수 1-픽셀 거리 정제를 수행하여 하나 이상의 참조 프레임의 각각의 참조 프레임에 대한 독립적 서브 픽셀 델타 모션 벡터들을 획득함.
- [0138] 단계 1113: 마지막 검색 센터 위치가 마지막 검색 센터에 대한 1-픽셀 이웃들의 세트에서 제2 비용 함수 값들에 대해 가장 낮은 제2 비용 함수 값을 갖는지를 결정함. 마지막 검색 센터 위치가 가장 낮은 제2 비용 함수 값을 갖는 경우에(1113, 예):
- [0139] 단계 1115: 제2 비용 함수 값들을 사용하여 맞추어진 파라메트릭 에리 표면 상의 값이 최소인 위치를 계산하는 것에 의해 각각의 참조 프레임에서의 최상의 정수 거리 정제 위치 주위의 서브 픽셀 거리 정제 모션 벡터들을 결정함.
- [0140] 단계 1117: 각각의 참조 프레임에 대해 결정된 정수 거리 정제 MV 및 결정된 서브 픽셀 거리 정제 MV의 합으로서 총 정제 MV를 리턴함.
- [0141] 본 개시내용의 실시예들은 위에 설명된 방법들을 수행하도록 구성되는 장치들을 또한 제공한다. 이러한 장치들은 소프트웨어 및 하드웨어의 조합일 수 있다. 예를 들어, 인코딩 및/또는 디코딩은 범용 프로세서(CPU), 또는 DSP(digital signal processor), 또는 FPGA(field programmable gate array) 등과 같은 칩에 의해 수행될 수 있다. 그러나, 본 개시내용의 실시예들이 프로그램가능 하드웨어 구현으로 제한되는 것은 아니다. 본 개시내용의 일부 실시예들은 ASIC(application-specific integrated circuit) 또는 위에 언급된 하드웨어 컴포넌트들의 조합을 사용하여 구현될 수 있다.
- [0142] 인코딩 및/또는 디코딩은 컴퓨터 판독가능 매체 상에 저장되는 프로그램 명령어들 또는 프로그램 코드에 의해 또한 구현될 수 있다. 이러한 프로그램 명령어들은, 프로세서 또는 컴퓨터에 의해 실행될 때, 프로세서 또는 컴퓨터로 하여금 위에 설명된 방법들의 단계들을 수행하게 한다. 이러한 컴퓨터 판독가능 매체는 DVD, CD, USB (플래시) 드라이브, 하드 디스크, 네트워크를 통해 이용가능한 서버 스토리지 등과 같이 프로그램 코드가 저장되는 임의의 매체일 수 있다.
- [0143] 도 12는 본 개시내용의 다양한 실시예들을 구현하기 위해 사용될 수 있는 장치(1200)의 블록도이다. 이러한 장치(1200)는 도 1에 도시되는 인코딩 장치(100) 및 도 2에 도시되는 디코딩 장치(200)일 수 있다. 추가적으로, 이러한 장치(1200)는 설명된 엘리먼트들 중 하나 이상을 호스팅할 수 있다. 일부 실시예들에서, 이러한 장치(1200)는, 스피커, 마이크로폰, 마우스, 터치스크린, 키패드, 키보드, 프린터, 디스플레이 등과 같은, 하나 이상의 입력/출력 디바이스들을 구비한다. 이러한 장치(1200)는 버스에 접속되는 하나 이상의 CPU(central processing units)(1210), 메모리(1220), 대용량 스토리지(1230), 비디오 어댑터(1240), 및 I/O 인터페이스(1260)를 포함할 수 있다. 버스는 메모리 버스 또는 메모리 제어기, 주변 버스, 비디오 버스, 등을 포함하는 임의의 타입의 몇몇 버스 아키텍처들 중 하나 이상이다.
- [0144] CPU(1210)는 임의의 타입의 전자 데이터 프로세서를 가질 수 있다. 메모리(1220)는 SRAM(static random access memory), DRAM(dynamic random access memory), SDRAM(synchronous DRAM), ROM (read-only memory), 이들의 조합 등과 같은 임의의 타입의 시스템 메모리를 가질 수 있거나, 또는 이들일 수 있다. 실시예에서, 메모리(1220)는 부트-업 시에 사용하기 위한 ROM, 및 프로그램들을 실행하는 동안 사용하기 위한 프로그램 및 데이터 스토리지를 위한 DRAM을 포함할 수 있다. 일부 실시예들에서, 메모리(1220)는 비-일시적일 수 있다. 대용량 스토리지(1230)는 데이터, 프로그램들, 및 다른 정보를 저장하는 그리고 이러한 데이터, 프로그램들, 및 다른 정보를 버스를 통해 액세스가능하게 하는 임의의 타입의 스토리지 디바이스를 포함할 수 있다. 대용량 스토리지(1230)는, 예를 들어, CPU(1210)에 의해 실행될 때, CPU로 하여금 본 명세서에 설명되는 방법들을 수행하게 하는, 프로그램 코드를 저장하도록 구성되는 솔리드 스테이트 드라이브, 하드 디스크 드라이브, 자기 디스크 드라이브, 광 디스크 드라이브 등 중 하나 이상을 포함한다. CPU(1210)는 도 10 및 도 11과 관련하여 설명되는 단계들의 다수의 동작들에 대해 반복적으로 구성될 수 있다. CPU(1210)는 MV0이 포인팅하는 제1 참조 화상 및 MV1이 포인팅하는 제2 참조 화상을 선택하도록 구성되는 참조 선택 유닛을 포함할 수 있다. 참조 화상의 선택



후에, 참조 화상 선택 유닛은 제1 참조 화상 또는 제2 참조 화상이 모션 벡터 정제를 수행하기 위해 사용되는지를 결정할 수 있다.

[0145] 비디오 어댑터(1240) 및 I/O 인터페이스(1260)는 외부 입력 및 출력 디바이스들을 장치(1200)에 연결하는 인터페이스들을 제공한다. 예를 들어, 이러한 장치(1200)는 SQL 커맨드 인터페이스를 클라이언트들에 제공할 수 있다. 예시되는 바와 같이, 입력 및 출력 디바이스들의 예들은 비디오 어댑터(1240)에 연결되는 디스플레이(1290) 및 I/O 인터페이스(1260)에 연결되는 마우스/키보드/프린터(1270)의 임의의 조합을 포함한다. 다른 디바이스들이 장치(1200)에 연결될 수 있고, 추가적인 또는 더 적은 인터페이스 카드들이 이용될 수 있다. 예를 들어, 프린터에 대한 직렬 인터페이스를 제공하기 위해 직렬 인터페이스 카드(도시되지 않음)가 사용될 수 있다.

[0146] 이러한 장치(1200)는, 이더넷 케이블 등과 같은, 유선 링크들, 및/또는 액세스 노드들 또는 하나 이상의 네트워크(1280)로의 무선 링크들을 포함하는, 하나 이상의 네트워크 인터페이스(1250)를 또한 포함할 수 있다. 네트워크 인터페이스(1250)는 장치(1200)가 네트워크들(1280)을 통해 원격 유닛들과 통신하는 것을 허용한다. 예를 들어, 네트워크 인터페이스(1250)는 데이터베이스에 통신을 제공할 수 있다. 실시예에서, 장치(1200)는, 다른 처리 유닛들, 인터넷, 또는 원격 스토리지 설비들 등과 같은, 원격 디바이스들과의 통신 및 데이터 처리를 위해 근거리 네트워크 또는 광역 네트워크에 연결된다. 이러한 장치(1200)는 입력(예를 들어, 네트워크 인터페이스)으로부터 수신되는 하나 이상의 화상 블록을 인코딩하고 및/또는 비트스트림으로부터 비디오 화상을 디코딩하기 위해 사용될 수 있다. 이러한 장치(1200)는 비트스트림으로부터 압축된 화상 블록들을 추출하도록 구성되는 비트스트림 파서, 서브 픽셀 정밀 델타 모션 벡터 정제를 획득하도록 구성되는 모션 벡터 정제 유닛, 및 획득된 모션 벡터들에 기초하여 블록 재구성을 수행하도록 구성되는 재구성 유닛을 포함할 수 있다.

[0147] 본 개시내용의 특정 특징 또는 양태가 몇몇 구현들 또는 실시예들 중 하나에 관하여만 개시되었을 수 있지만, 이러한 특징 또는 양태는 임의의 주어진 또는 특정 적용을 위해 요망되는 그리고 유리할 수 있는 다른 구현들 또는 실시예들의 하나 이상의 다른 특징 또는 양태들과 조합될 수 있다. 또한, "포함하다(include)", "갖다(have)", "함께(with)"이라는 용어들, 또는 이들의 다른 변형들이 상세한 설명 또는 청구항들 중 어느 하나에서 사용되는 정도까지, 이러한 용어들은 "포함하다(comprise)"는 용어와 유사한 방식으로 포괄적인 것으로 의도된다. 또한, "예시적인(exemplary)", "예를 들어(for example)" 및 "예를 들어(e.g.)"라는 용어들은, 최상 또는 최적의 것보다는 오히려, 단지 예로서 의미되는 것이다. "연결된(coupled)" 및 "접속된(connected)"이라는 용어들이 파생어들과 함께 사용되었을 수 있다. 이러한 용어들은 그것들이 직접적인 물리적 또는 전기적 접촉인지, 또는 그것들이 서로 직접 접촉하지 않는지에 무관하게 2개의 엘리먼트들이 서로 협력하거나 또는 상호작용한다는 점을 표시하기 위해 사용될 수 있다는 점이 이해되어야 한다.

[0148] 본 명세서에서 구체적 양태들이 예시되고 설명되었더라도, 본 개시내용의 범위로부터 벗어나지 않고 도시되고 설명되는 구체적 양태들에 대해 다양한 대안의 및/또는 동등한 구현들이 대체될 수 있다는 점이 해당 분야에서의 통상의 기술자에 의해 이해될 것이다. 본 출원은 본 명세서에서 논의되는 구체적 양태들의 임의의 적응들 또는 변동들을 커버하도록 의도된다.

[0149] 다음의 청구항들에서의 엘리먼트들은 대응하는 라벨링이 있는 특정 시퀀스로 열거되더라도, 청구항들 열거들이 이러한 엘리먼트들의 일부 또는 전부를 구현하기 위한 특정 시퀀스를 달리 암시하지 않는 한, 이러한 엘리먼트들은 반드시 해당 특정 시퀀스로 구현되는 것으로 제한되도록 의도되는 것은 아니다.

[0150] 위 교시사항들에 비추어 많은 대안들, 수정들, 및 변동들이 해당 분야에서의 기술자들에게 명백할 것이다. 물론, 해당 분야에서의 기술자들은 본 명세서에서 설명되는 것들 이외의 본 발명의 수많은 적응들이 존재한다는 점을 쉽게 인식한다. 본 발명은 하나 이상의 특정 실시예를 참조하여 설명되었지만, 해당 분야에서의 기술자들은 본 발명의 범위로부터 벗어나지 않고 많은 변형들이 이루어질 수 있다는 점을 인식한다. 따라서 첨부된 청구항들 및 그들의 균등물들의 범위 내에서, 본 발명은 본 명세서에서 구체적으로 설명되는 것과는 달리 실시될 수 있다는 점이 이해되어야 한다.

[0151] 위에 언급된 회로는 또한 단일 칩에 집적될 수 있다. 그러나, 본 발명이 이에 제한되는 것은 아니며, 이러한 회로는 대응하는 코드로 프로그램되는 범용 프로세서 또는 DSP와 같은 하드웨어 및 소프트웨어의 조합 또는 상이한 피스들 또는 하드웨어를 포함할 수 있다.

[0152] 위에 설명된 흐름도들은 디코더 측 모션 벡터 정제 기술들의 예들을 예시하도록 의도된다. 해당 분야에서의 기술자는 본 개시내용의 범위를 벗어나지 않고 본 개시내용을 구현하기 위해 단계들을 수정하거나 또는 단계들을

조합할 수 있다.

- [0153] 본 개시내용의 하나의 양태에서는, 디코더 측 모션 벡터 정제 시스템에서 하나 이상의 참조 프레임에서 서브 픽셀 정밀 델타 모션 벡터 정제를 획득하기 위한 방법이 제공된다. 이러한 방법은,
- [0154] 프로세서(1001)를 제공하는 단계;
- [0155] 프로세서에 의해, 비용 함수를 사용하여 정수 1-픽셀 거리 정제 동작을 반복적으로 수행하여 하나 이상의 참조 프레임의 각각의 참조 프레임에 대한 정수 거리 정제 모션 벡터를 결정하는 것에 의해 루프 동작을 시작하는 단계(1003);
- [0156] 프로세서에 의해, 검색 센터의 위치의 비용 함수 값이 1-픽셀 이웃 위치들의 세트에서의 가장 낮은 함수 값인지를 결정하는 단계(1005);
- [0157] 검색 센터의 위치의 비용 함수 값이 가장 낮은 비용 함수 값인 경우에(1005, 예): 루프 동작을 이탈하는 단계;
- [0158] 검색 센터의 비용 함수 값 및 1-픽셀 이웃 위치들의 세트를 사용하여 맞추어지는 파라메트릭 에러 표면 상의 값이 최소인 위치를 계산하는 것에 의해 각각의 참조 프레임에서 검색 센터 주위의 서브 픽셀 거리 정제 모션 벡터들을 결정하는 단계(1011); 및
- [0159] 각각의 참조 프레임에 대해 파라메트릭 에러 표면을 사용하여 획득되는 서브 픽셀 위치를 리턴하는 단계(1013)를 포함할 수 있다.
- [0160] 하나의 실시예에서, 이러한 방법은, 검색 센터의 위치의 비용 함수 값이 가장 낮은 비용 함수 값이 아닌 경우에(1005, 아니오); 현재 동작이 마지막 동작인지를 결정하는 단계(1007);
- [0161] 현재 동작이 마지막 동작인 경우에(1007, 예):
- [0162] 비용 함수가 가장 낮은 각각의 참조 프레임에서의 위치들을 리턴하는 단계(1015); 및
- [0163] 현재 동작이 마지막 동작이 아닌 경우에(1007, 아니오):
- [0164] 프로세서에 의해, 현재 정수 1-픽셀 거리 정제 동작의 검색 센터의 위치를 현재 동작에서 가장 낮은 비용을 갖는 위치에 의해 업데이트하는 단계(1009); 및
- [0165] 루프 동작(1003, 1005, 1007, 1009)을 반복하는 단계를 추가로 포함할 수 있다.
- [0166] 하나의 실시예에서, 파라메트릭 에러 표면은 동일한 거리로 센터 픽셀로부터 이격되는 4개의 주변 픽셀들에 의해 둘러싸이는 센터 픽셀에 배열되는 5개의 픽셀들을 포함한다.
- [0167] 하나의 실시예에서, 비용 함수는 다음 표현에 의해 계산된다:
- [0168]  $E(x,y) = A*(x-x_0)^2 + B*(y-y_0)^2 + C$ ;
- [0169] 여기서  $E(x, y)$ 는 좌표들  $(x, y)$ 의 평가된 비용 함수 값이고,  $x_0$  및  $y_0$ 은 에러가 최소인 센터  $(0, 0)$ 에 관하여 서브 픽셀 변위와 연관된 좌표들이고,  $C$ 는 좌표들  $(x_0, y_0)$ 에서의 에러와 연관된 파라미터이고,  $A$  및  $B$ 는 상수 값들이고,  $x$  및  $y$ 는 이웃 위치들과 연관된 좌표들이고,  $x$  및  $y$  각각은 -1, 0, 및 1을 포함하는 정수이다.
- [0170] 하나의 실시예에서, 서브 픽셀 위치는 다음의 표현들에 의해 획득된다:
- [0171]  $x_0 = (E(-1,0) - E(1,0)) / (2*N*(E(-1,0) + E(1,0) - 2* E(0,0)))$ ; 및
- [0172]  $y_0 = (E(0,-1) - E(0,1)) / (2*N*(E(0,-1) + E(0,1) - 2* E(0,0)))$ ,
- [0173] 여기서,
- [0174]  $E(x,y) = A*(x-x_0)^2 + B*(y-y_0)^2 + C$ ;
- [0175]  $E(x, y)$ 는 평가된 비용 함수 값이고,  $x_0$  및  $y_0$ 은 에러가 최소인 센터  $(0, 0)$ 에 관하여 서브 픽셀 변위와 연관된 좌표들이고,  $C$ 는 좌표들  $(x_0, y_0)$ 에서의 에러와 연관된 파라미터이고,  $A$  및  $B$ 는 상수 값들이고,  $x$  및  $y$ 는 이웃 위치들과 연관된 좌표들이고,  $x$  및  $y$  각각은 -1, 0, 및 1을 포함하는 정수이고,  $N$ 은 서브 픽셀 정밀도의 1/2, 1/4, 1/8, 또는 1/16에 대해 1, 2, 4, 또는 8의 정수이다.
- [0176] 하나의 실시예에서, 서브 픽셀 정밀 델타 모션 벡터 정제는 템플릿 매칭에 의해 획득된다.

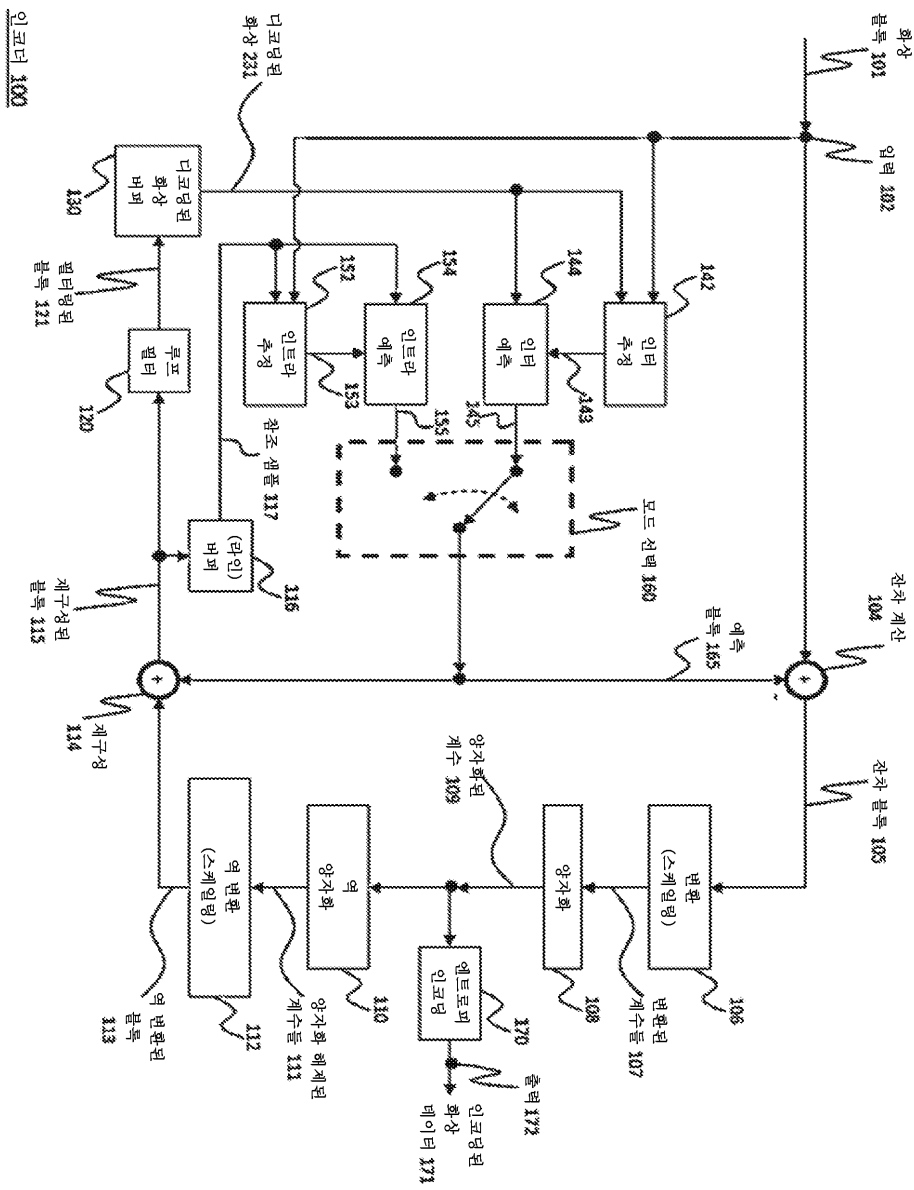
- [0177] 본 개시내용의 다른 양태에서는, 디코더 측 모션 벡터 정제 시스템에서 하나 이상의 참조 프레임에서 서브 픽셀 정밀 델타 모션 벡터를 획득하기 위한 장치가 제공된다. 이러한 장치는 처리 유닛 및 컴퓨터 판독가능 명령어들을 내부에 갖는 비-일시적 컴퓨터 판독가능 매체를 포함하며, 처리 유닛으로 하여금:
- [0178] 비용 함수를 사용하여 수 N의 정수 1-픽셀 거리 정제 동작들을 반복적으로 수행하여 각각의 참조 프레임에 대한 정수 거리 정제 모션 벡터를 결정하는 것에 의해 루프 동작을 시작하는 단계- 현재 반복의 검색 센터의 위치는 이전 동작에서 비용이 가장 낮은 위치에 의해 업데이트됨 -;
- [0179] 검색 센터의 위치의 비용이 1-픽셀 이웃 위치들의 세트의 임의의 비용보다 더 낮은지를 결정하는 단계;
- [0180] 검색 센터의 위치의 비용이 가장 낮은 비용인 경우에;
- [0181] 루프 동작을 이탈하는 단계;
- [0182] 검색 센터의 비용 함수 값 및 1-픽셀 이웃 위치들의 세트를 사용하여 맞추어지는 파라메트릭 에러 표면 상의 값이 최소인 위치를 계산하는 것에 의해 각각의 참조 프레임에서 검색 센터 주위의 서브 픽셀 거리 정제 모션 벡터들을 결정하는 단계; 및
- [0183] 각각의 참조 프레임에 대해 파라메트릭 에러 표면을 사용하여 획득되는 서브 픽셀 위치를 리턴하는 단계를 수행하게 한다.
- [0184] 하나의 실시예에서, 컴퓨터 판독가능 명령어들을 내부에 갖는 비-일시적 컴퓨터 판독가능 매체는 추가로 처리 유닛으로 하여금:
- [0185] 검색 센터의 위치의 비용 함수 값이 가장 낮은 비용 함수 값이 아닌 경우에(1005, 아니오): 현재 동작이 마지막 동작인지를 결정하는 단계;
- [0186] 현재 동작이 마지막 동작인 경우에: 비용 함수가 가장 낮은 각각의 참조 프레임에서의 위치들을 리턴하는 단계; 및
- [0187] 현재 동작이 마지막 동작이 아닌 경우에: 프로세서에 의해, 현재 정수 1-픽셀 거리 정제 동작의 검색 센터의 위치를 현재 정제 동작에서 가장 낮은 비용을 갖는 위치에 의해 업데이트하는 단계(1009); 루프 동작을 반복하는 단계를 수행하게 한다.
- [0188] 하나의 실시예에서, 비용 함수는 다음 표현에 의해 계산된다:
- [0189]  $E(x,y) = A*(x-x_0)^2 + B*(y-y_0)^2 + C$ ;
- [0190] 여기서  $E(x, y)$ 는 평가된 비용 함수 값이고,  $x_0$  및  $y_0$ 은 에러가 최소인 센터 (0, 0)에 관하여 서브 픽셀 변위와 연관된 좌표들이고, C는 좌표들 ( $x_0, y_0$ )에서의 에러와 연관된 파라미터이고, A 및 B는 상수 값들이고, x 및 y는 이웃 위치들과 연관된 좌표들이고, x 및 y 각각은 -1, 0, 및 1을 포함하는 정수이다.
- [0191] 하나의 실시예에서, 서브 픽셀 위치는 다음의 표현에 의해 획득된다:
- [0192]  $x_0 = (E(-1,0) - E(1,0)) / (2*N*(E(-1,0) + E(1,0) - 2* E(0,0)))$ ; 및
- [0193]  $y_0 = (E(0,-1) - E(0,1)) / (2*N*(E(0,-1) + E(0,1) - 2* E(0,0)))$ ,
- [0194] 여기서,
- [0195]  $E(x,y) = A*(x-x_0)^2 + B*(y-y_0)^2 + C$ ;
- [0196]  $E(x, y)$ 는 평가된 비용 함수 값이고,  $x_0$  및  $y_0$ 은 에러가 최소인 센터 (0, 0)에 관하여 서브 픽셀 변위와 연관된 좌표들이고, C는 좌표들 ( $x_0, y_0$ )에서의 에러와 연관된 파라미터이고, A 및 B는 상수 값들이고, x 및 y는 이웃 위치들과 연관된 좌표들이고, x 및 y 각각은 -1, 0, 및 1을 포함하는 정수이고, N은 서브 픽셀 정밀도의 1/2, 1/4, 1/8, 또는 1/16에 대해 1, 2, 4, 또는 8의 정수이다.
- [0197] 본 개시내용의 다른 양태는 디코더 측 모션 벡터 정제 시스템에서 참조 리스트 L0에서의 및 참조 리스트 1에서의 하나 이상의 참조 프레임에서 서브 픽셀 정밀 델타 모션 벡터 정제를 획득하기 위한 방법을 제공한다. 이러한 방법은,
- [0198] 프로세서(1101)를 제공하는 단계;

- [0199] 프로세서에 의해, 제1 비용 함수를 사용하여 정수 1-픽셀 거리 정제 동작을 반복적으로 수행하여 하나 이상의 참조 프레임의 각각의 참조 프레임에 대한 정수 거리 정제 모션 벡터를 결정하는 것에 의해 루프 동작을 시작하는 단계(1103);
- [0200] 현재 동작이 마지막 동작인지를 결정하는 단계(1105);
- [0201] 현재 동작이 마지막 동작인 경우에(1105, 예):
- [0202] 제2 비용 함수를 사용하여 정수 1-픽셀 거리 정제를 수행하여 각각의 참조 프레임에 대한 독립적 서브 픽셀 델타 모션 벡터들을 획득하는 단계(1111);
- [0203] 현재 동작이 마지막 동작이 아닌 경우에(1105, 아니오);
- [0204] 프로세서에 의해, 현재 동작의 검색 센터의 위치의 비용 함수 값이 1-픽셀 이웃 위치들의 세트에서의 첫번째 가장 낮은 함수 값인지를 결정하는 단계(1107);
- [0205] 검색 센터의 위치의 비용 함수 값이 첫번째 가장 낮은 함수 값인 경우에(1107, 예):
- [0206] 루프 동작을 이탈하는 단계;
- [0207] 제2 비용 함수를 사용하여 정수 1-픽셀 거리 정제를 수행하여 각각의 참조 프레임에 대한 독립적 서브 픽셀 델타 모션 벡터들을 획득하는 단계(1111);
- [0208] 현재 동작의 검색 센터의 위치의 비용 함수 값이 첫번째 가장 낮은 함수 값이 아닌 경우에(1107, 아니오):
- [0209] 현재 동작의 비용이 첫번째 가장 낮은 위치로 센터를 업데이트하는 단계; 및
- [0210] 루프 동작(1103, 1105, 1107, 1109)을 반복하는 단계를 포함할 수 있다.
- [0211] 하나의 실시예에서, 이러한 방법은,
- [0212] 프로세서에 의해, 획득된 독립적 서브 픽셀 델타 모션 벡터들의 검색 센터의 위치의 제2 비용 함수 값이 두번째 가장 낮은 함수 값인지를 결정하는 단계(1113);
- [0213] 제2 비용 함수 값이 가장 낮은 비용 함수 값인 경우에(1113, 예);
- [0214] 검색 센터의 비용 함수 값 및 1-픽셀 이웃 위치들의 세트를 사용하여 맞추어지는 파라메트릭 에러 표면 상의 값이 최소인 위치를 계산하는 것에 의해 각각의 참조 프레임에서 검색 센터 주위의 서브 픽셀 거리 정제 모션 벡터들을 결정하는 단계(1115); 및
- [0215] 각각의 참조 프레임에 대한 파라메트릭 에러 표면을 사용하여 획득되는 각각의 참조 프레임에 대해 결정된 정수 거리 정제 모션 벡터 및 결정된 서브 픽셀 거리 정제 모션 벡터의 합으로서 총 정제 모션 벡터를 리턴하는 단계(1117);
- [0216] 제2 비용 함수 값이 두번째 가장 낮은 비용 함수 값이 아닌 경우에(1113, 아니오);
- [0217] 비용 함수 값이 두번째 가장 낮은 각각의 참조 프레임에서의 위치들에 대응하는 정제 모션 벡터를 리턴하는 단계(1119)를 추가로 포함할 수 있다.
- [0218] 하나의 실시예에서, 서브 픽셀 위치는 다음에 의해 도출된다:
- [0219] 
$$x_0 = (E(-1,0) - E(1,0)) / (2*N*(E(-1,0) + E(1,0) - 2* E(0,0)))$$
- [0220] 
$$y_0 = (E(0,-1) - E(0,1)) / (2*N*(E(0,-1) + E(0,1) - 2* E(0,0)))$$
- [0221] 여기서,
- [0222] 
$$E(x,y) = A*(x-x_0)^2 + B*(y-y_0)^2 + C;$$
- [0223]  $(x_0, y_0)$ 은 에러가 최소인  $(0, 0)$ 에 관하여 서브 픽셀 변위에 대응하고  $C$ 는  $(x_0, y_0)$ 에서의 에러에 대응하고,  $A$  및  $B$ 는 상수 수이고,  $(x, y)$ 는 이웃 위치들에 대응하고,  $x$ 는  $-1,0,1$ 을 포함하고,  $y$ 는  $-1,0,1$ 을 포함한다.
- [0224] 하나의 실시예에서, 제1 비용 함수 및 제2 비용 함수 각각은 다음 표현에 의해 계산된다:
- [0225] 
$$E(x,y) = A*(x-x_0)^2 + B*(y-y_0)^2 + C;$$

- [0226] 여기서  $E(x, y)$ 는 평가된 비용 함수 값이고,  $x_0$  및  $y_0$ 은 에러가 최소인 센터  $(0, 0)$ 에 관하여 서브 픽셀 변위와 연관된 데카르트 좌표들이고,  $C$ 는 데카르트 좌표들  $(x_0, y_0)$ 에서의 에러와 연관된 파라미터이고,  $A$  및  $B$ 는 상수 값들이고,  $x$  및  $y$ 는 이웃 위치들과 연관된 데카르트 좌표들이고,  $x$  및  $y$  각각은  $-1, 0$ , 및  $1$ 을 포함하는 정수이다.
- [0227] 하나의 실시예에서, 서브 픽셀 정밀 델타 모션 벡터 정제는 쌍방 매칭에 의해 획득된다.
- [0228] 하나의 실시예에서, 이러한 쌍방 매칭은 참조 리스트  $L_0$  및 참조 리스트  $L_1$  양자 모두에서 공동 정제를 수행하는 것을 포함한다.
- [0229] 하나의 실시예에서, 이러한 쌍방 매칭은 공통 쌍방 평균화된 템플릿에 관련하여 참조 리스트  $L_0$  및 참조 리스트  $L_1$  양자 모두에서 독립적 공동 정제를 수행하는 것을 포함한다.
- [0230] 하나의 실시예에서, 이러한 쌍방 매칭은, 공통 쌍방 평균화된 템플릿에 관련하여 참조 리스트  $L_0$  또는 참조 리스트  $L_1$ 에서  $s$  정제를 수행하는 것, 및 다른 참조 리스트에서 변위를 획득할 때 추정된 수평 및 수직 변위들을 무효화하는 것을 포함한다.
- [0231] 요약하면, 본 개시내용의 실시예들은 양방향 모션 벡터 추정을 위한 템플릿 매칭에 기초하는 모션 벡터 결정에 관련된다. 특히, 정제될 초기 모션 벡터들에 의해 포인팅되는 블록들의 평균으로서 블록 템플릿이 구성된다. 다음으로, 2개의 상이한 참조 화상들에서의 템플릿 매칭에 의해 모션 벡터 정제가 수행된다. 이러한 매칭은 2개의 참조 화상들 각각에서 최상의 매칭 블록에 대응하는 매칭 함수의 최적(함수에 의존하여, 최소 또는 최대)을 발견하는 것에 의해 수행된다. 이러한 최적은 (검색 공간의 모션 벡터 후보들에 의해 포인팅되는 블록 위치들 중에서) 제로 평균 템플릿 및 제로 평균 후보 블록에 대해 검색된다. 다시 말해서, 함수 최적화를 수행하기 전에, 템플릿의 평균이 템플릿으로부터 감산되고 후보 블록의 평균이 후보 블록으로부터 감산된다. 다음으로 현재 블록의 예측기는 각각의 참조 화상들에서 최상의 매칭 블록들의 가중 평균으로서 계산된다.

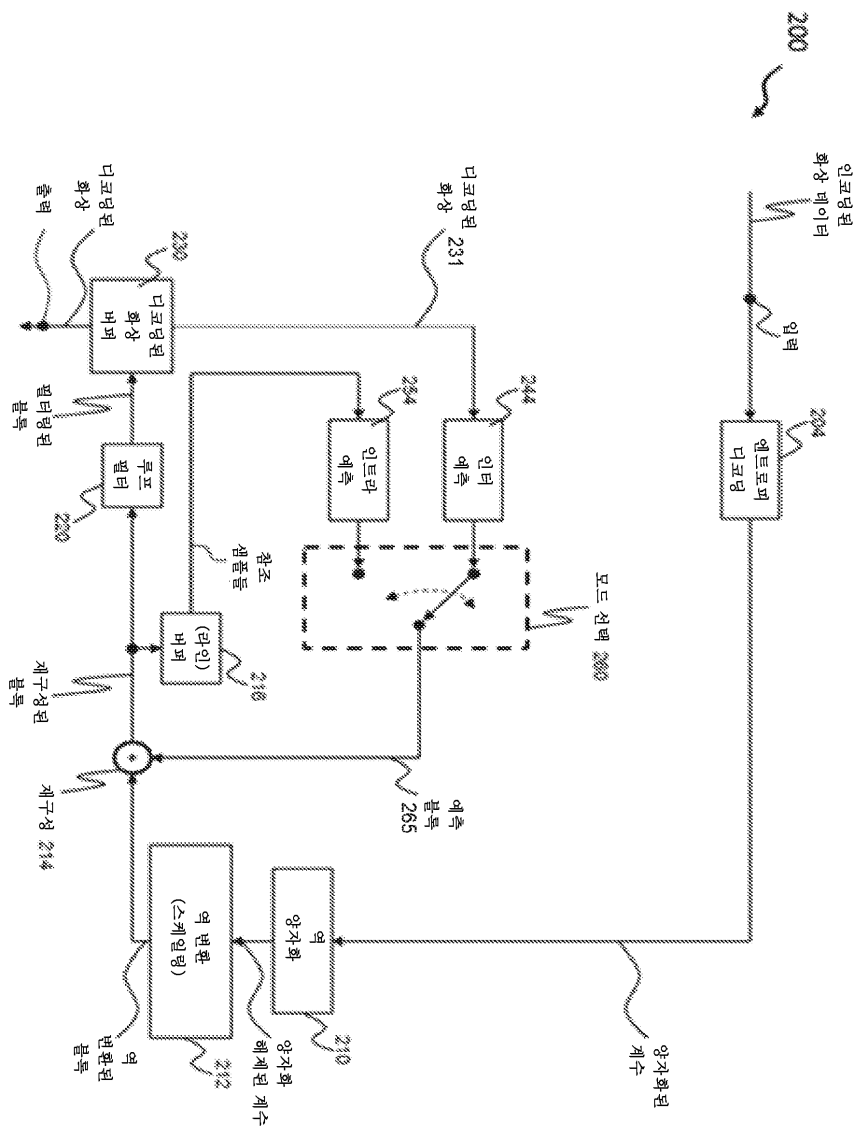
도면

도면1

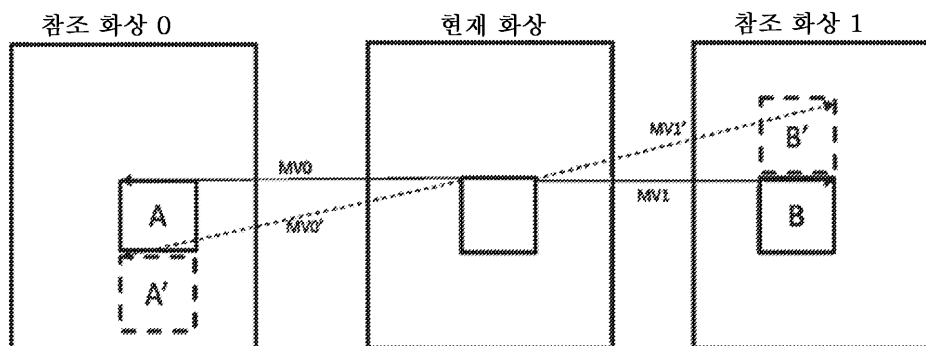




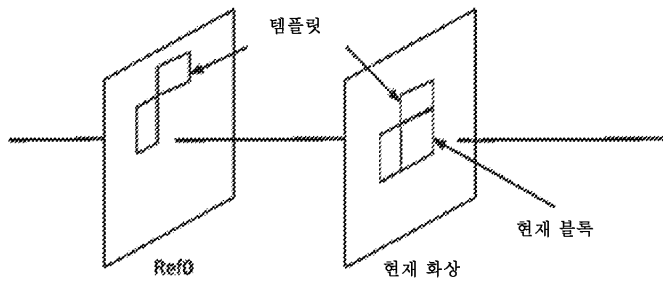
도면2



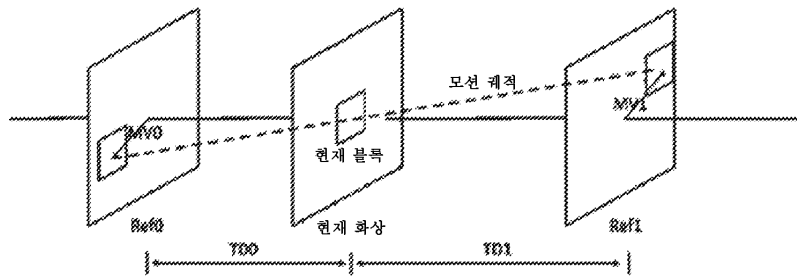
도면3



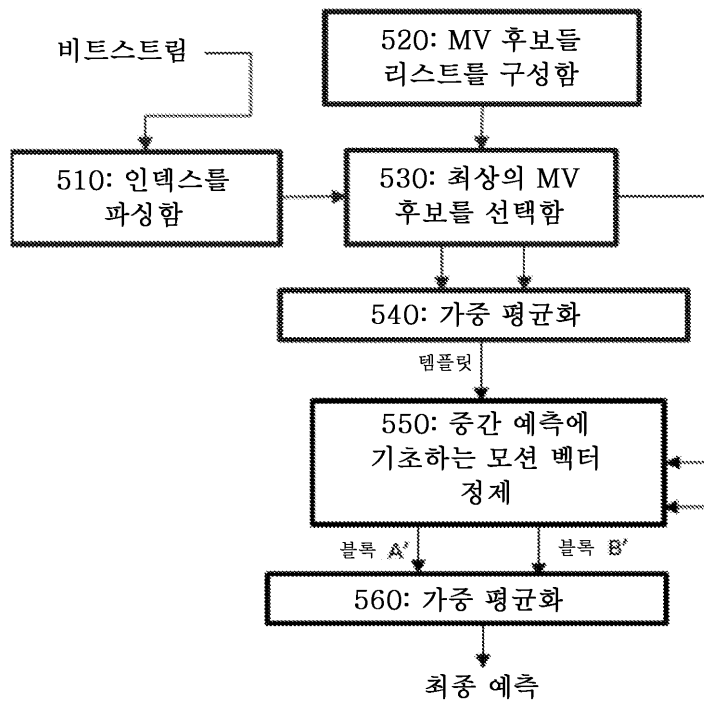
도면4a



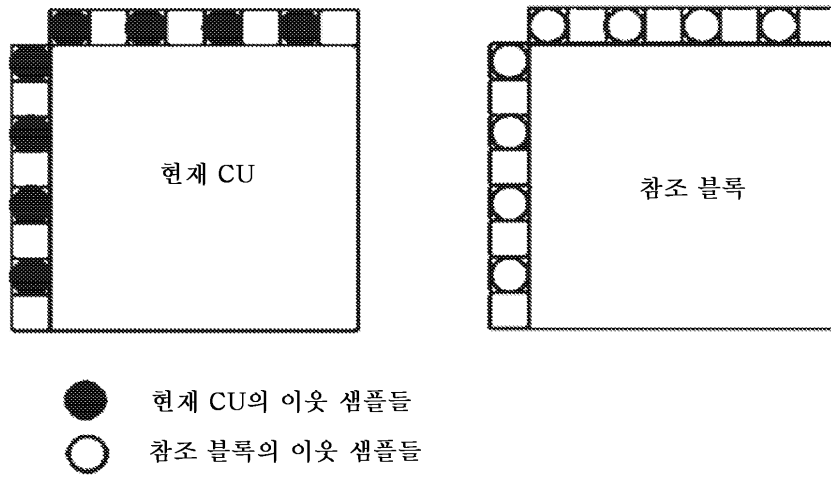
도면4b



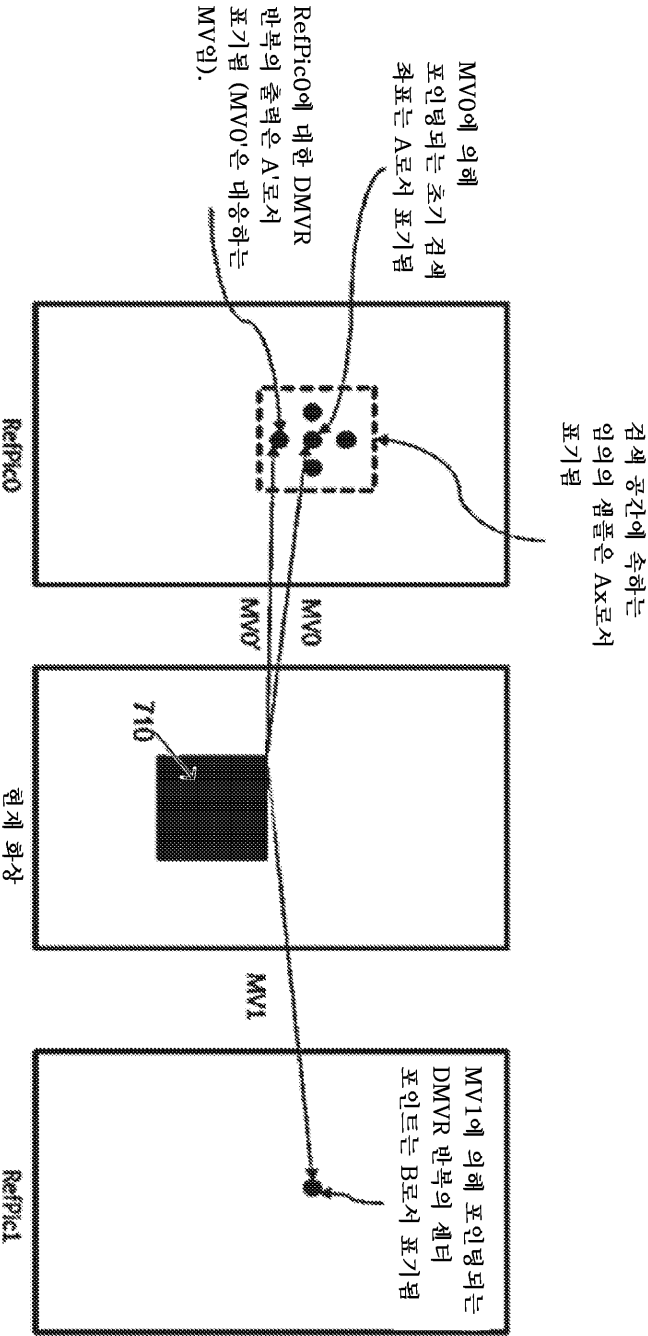
도면5



도면6



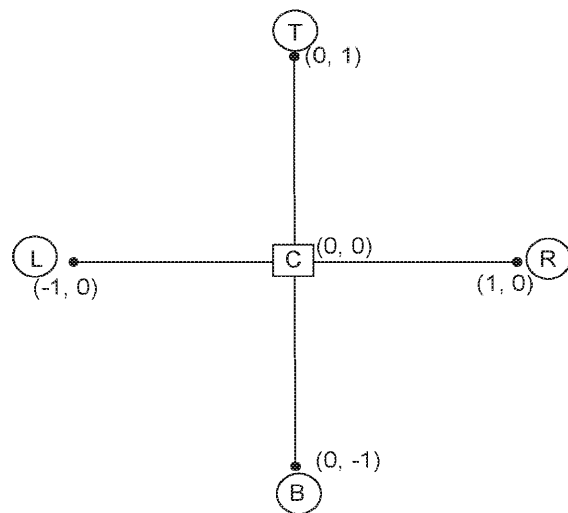
도면7



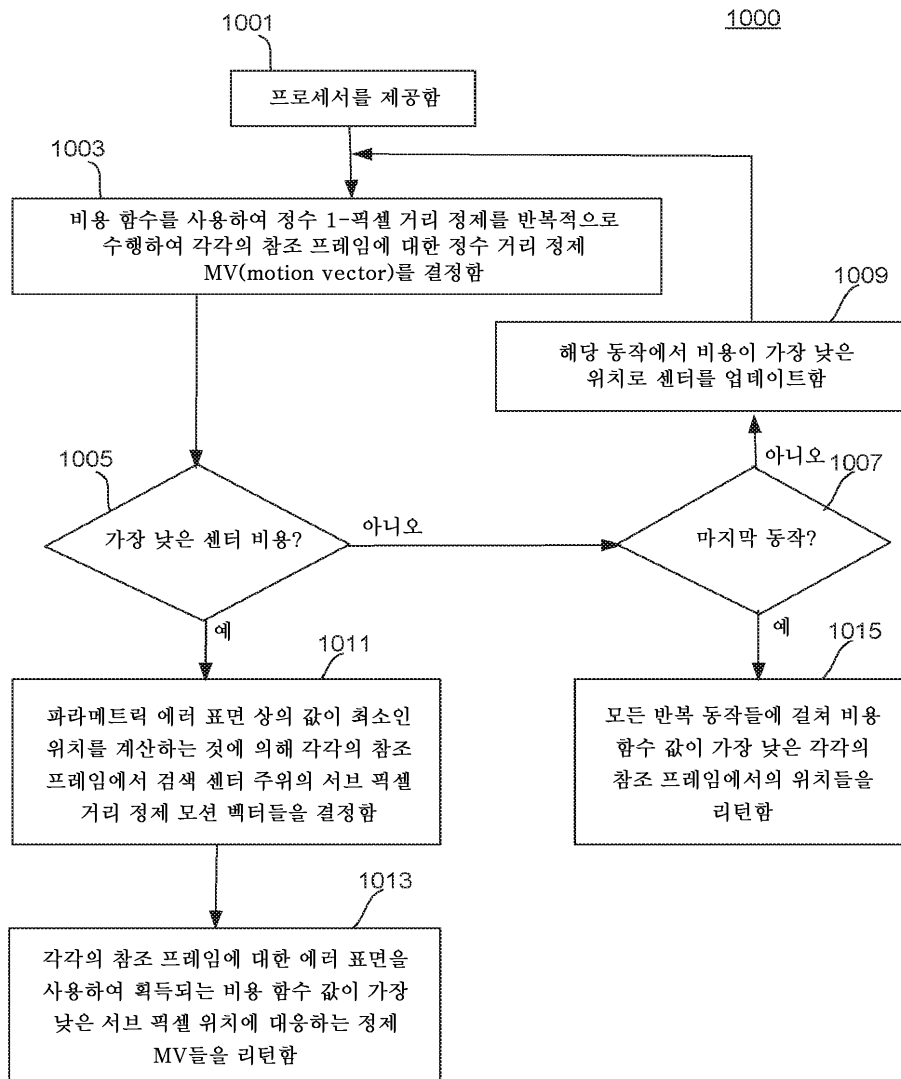
도면8

$A_{1,1}$				$A_{0,1}$	$a_{0,1}$	$b_{0,1}$	$c_{0,1}$	$A_{1,1}$				$A_{2,1}$
$A_{1,0}$				$A_{0,0}$	$a_{0,0}$	$b_{0,0}$	$c_{0,0}$	$A_{1,0}$				$A_{2,0}$
$d_{1,0}$				$d_{0,0}$	$e_{0,0}$	$f_{0,0}$	$g_{0,0}$	$d_{1,0}$				$d_{2,0}$
$h_{1,0}$				$h_{0,0}$	$i_{0,0}$	$j_{0,0}$	$k_{0,0}$	$h_{1,0}$				$h_{2,0}$
$n_{1,0}$				$n_{0,0}$	$p_{0,0}$	$q_{0,0}$	$r_{0,0}$	$n_{1,0}$				$n_{2,0}$
$A_{1,1}$				$A_{0,1}$	$a_{0,1}$	$b_{0,1}$	$c_{0,1}$	$A_{1,1}$				$A_{2,1}$
$A_{1,2}$				$A_{0,2}$	$a_{0,2}$	$b_{0,2}$	$c_{0,2}$	$A_{1,2}$				$A_{2,2}$

도면9

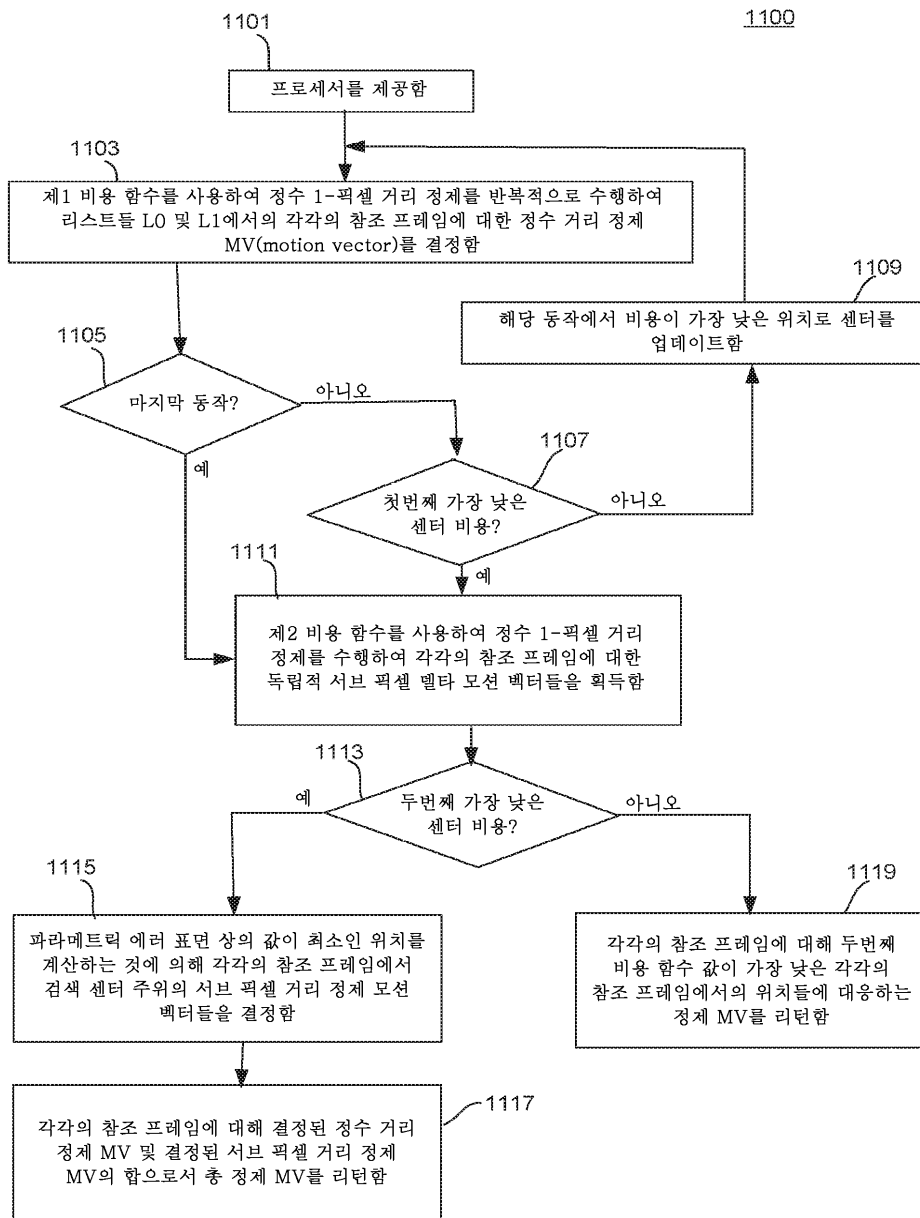


도면10





도면11



도면12

