(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2024-0050408
(43) 공개일자 2024년04월18일

(51) 국제특허분류(Int. Cl.)
H04N 19/513 (2014.01) *H04N 19/176* (2014.01)
H04N 19/192 (2014.01) *H04N 19/573* (2014.01)
(52) CPC특허분류
H04N 19/513 (2015.01)
H04N 19/176 (2015.01)
(21) 출원번호 10-2024-7010188
(22) 출원일자(국제) 2022년08월29일
심사청구일자 없음
(85) 번역문제출일자 2024년03월26일
(86) 국제출원번호 PCT/KR2022/012897
(87) 국제공개번호 WO 2023/027564
국제공개일자 2023년03월02일
(30) 우선권주장
1020210114039 2021년08월27일 대한민국(KR)
1020210124790 2021년09월17일 대한민국(KR)

(71) 출원인
주식회사 윌러스표준기술연구소
경기도 성남시 분당구 황새울로 216 , 5층(수내동)
(72) 발명자
김경용
경기도 성남시 분당구 황새울로 216, 5층 주식회사 윌러스표준기술연구소
김동철
경기도 성남시 분당구 황새울로 216, 5층 주식회사 윌러스표준기술연구소
(뒷면에 계속)
(74) 대리인
홍성진

전체 청구항 수 : 총 20 항

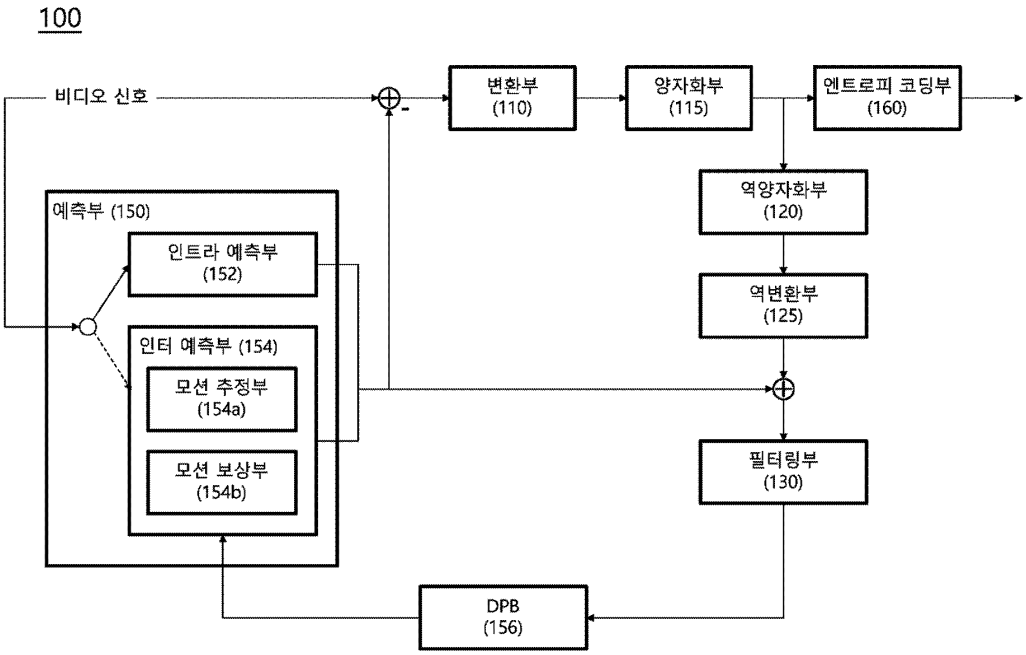
(54) 발명의 명칭 **움직임 정보를 보정하는 방법 및 이를 위한 장치**

(57) 요 약

비디오 신호 디코딩 장치는 프로세서를 포함하고, 상기 프로세서는 현재 블록과 관련된 하나 이상의 제1 움직임 정보들을 포함하는 제1 움직임 정보 리스트를 획득하고, 상기 하나 이상의 제1 움직임 정보들을 보정하여 하나 이상의 제1 보정된 움직임 정보들을 획득하고, 상기 하나 이상의 제1 보정된 움직임 정보들 각각에 대한 하나 이

(뒷면에 계속)

대 표 도 - 도1



상의 제1 코스트 값들을 획득하고, 상기 하나 이상의 제1 보정된 움직임 정보들을 상기 제1 코스트 값들에 기초하여 재정렬하여 제2 움직임 정보 리스트를 획득하고, 상기 제2 움직임 정보 리스트에 포함되는 제1 움직임 정보에 기초하여 하나 이상의 제2 움직임 정보들을 획득하고, 상기 하나 이상의 제2 움직임 정보들을 보정하여 하나 이상의 제2 보정된 움직임 정보들을 획득하고, 상기 제2 보정된 움직임 정보들 각각에 대한 하나 이상의 제2 코스트 값들을 획득하고, 상기 제2 코스트 값들에 기초하여 결정되는 제2 움직임 정보에 기초하여 상기 현재 블록을 예측한다.

(52) CPC특허분류
H04N 19/192 (2015.01)
H04N 19/573 (2015.01)

(72) 발명자
손주형
경기도 성남시 분당구 황새울로 216, 5층 주식회사 윌러스표준기술연구소

곽진삼
경기도 성남시 분당구 황새울로 216, 5층 주식회사 윌러스표준기술연구소

명 세 서

청구범위

청구항 1

비디오 신호 디코딩 장치에 있어서,

프로세서를 포함하며,

상기 프로세서는,

현재 블록과 관련된 하나 이상의 제1 움직임 정보들을 포함하는 제1 움직임 정보 리스트를 획득하고,

상기 하나 이상의 제1 움직임 정보들을 보정하여 하나 이상의 제1 보정된 움직임 정보들을 획득하고,

상기 하나 이상의 제1 보정된 움직임 정보들 각각에 대한 하나 이상의 제1 코스트 값들을 획득하고,

상기 하나 이상의 제1 보정된 움직임 정보들을 상기 제1 코스트 값들에 기초하여 재정렬하여 제2 움직임 정보 리스트를 획득하고,

상기 제2 움직임 정보 리스트에 포함되는 제1 움직임 정보에 기초하여 하나 이상의 제2 움직임 정보들을 획득하고,

상기 하나 이상의 제2 움직임 정보들을 보정하여 하나 이상의 제2 보정된 움직임 정보들을 획득하고,

상기 제2 보정된 움직임 정보들 각각에 대한 하나 이상의 제2 코스트 값들을 획득하고,

상기 제2 코스트 값들에 기초하여 결정되는 제2 움직임 정보에 기초하여 상기 현재 블록을 예측하는 것을 특징으로 하는 비디오 신호 디코딩 장치.

청구항 2

제 1항에 있어서,

상기 하나 이상의 제1 코스트 값들은 상기 현재 블록과 상기 하나 이상의 제1 보정된 움직임 정보들 각각에 대응되는 참조 블록간의 유사도와 관련된 값이고,

상기 하나 이상의 제2 코스트 값들은 상기 현재 블록과 상기 하나 이상의 제2 보정된 움직임 정보들 각각에 대응되는 참조 블록간의 유사도와 관련된 값인 것을 특징으로 하는 비디오 신호 디코딩 장치.

청구항 3

제 1항에 있어서,

상기 제2 움직임 정보 리스트는 상기 하나 이상의 제1 보정된 움직임 정보들에 각각 대응되는 코스트 값이 오름차순으로 정렬되는 것을 특징으로 하는 비디오 신호 디코딩 장치.

청구항 4

제 1항에 있어서,

상기 제1 움직임 정보는 상기 제1 코스트 값들 중 가장 작은 코스트 값에 대응하는 보정된 움직임 정보인 것을 특징으로 하는 비디오 신호 디코딩 장치.

청구항 5

제 1항에 있어서,

상기 제2 움직임 정보는 상기 제2 코스트 값들 중 가장 작은 코스트 값에 대응하는 보정된 움직임 정보인 것을 특징으로 하는 비디오 신호 디코딩 장치.

청구항 6

제 1항에 있어서,

상기 하나 이상의 제1 움직임 정보들은 각각 서로 다른 픽쳐에 포함되고,

상기 하나 이상의 제2 움직임 정보들은 각각 서로 다른 픽쳐에 포함되는 것을 특징으로 하는 비디오 신호 디코딩 장치.

청구항 7

제 1항에 있어서,

상기 하나 이상의 제1 보정된 움직임 정보들 및 상기 하나 이상의 제2 보정된 움직임 정보들은,

MVD(Motion Vector Difference), TM(Template Matching), BM(Bilateral Matching), MMVD(Merge mode with MVD), MMVD(Merge mode with MVD) 기반의 TM, 광 흐름(Optical flow)에 기초한 기반 TM, 멀티 패스 DMVR (Multi pass Decoder-side Motion Vector Refinement) 중 적어도 어느 하나 이상에 기초하여 보정되는 것을 특징으로 하는 비디오 신호 디코딩 장치.

청구항 8

제 1항에 있어서,

상기 현재 블록의 부호화 모드가 머지(merge) 모드인 경우,

상기 하나 이상의 제1 움직임 정보들 및 상기 하나 이상의 제2 움직임 정보들은 MMVD(Merge mode with MVD)에 의해 유도되는 움직임 정보인 것을 특징으로 하는 비디오 신호 디코딩 장치.

청구항 9

제 1항에 있어서,

상기 하나 이상의 제1 움직임 정보들 각각에 대응되는 블록들은 제1 탐색 영역 내에 위치하는 블록이고,

상기 하나 이상의 제2 움직임 정보들 각각에 대응되는 블록들은 제2 탐색 영역 내에 위치하는 블록이고,

상기 제1 탐색 영역과 상기 제2 탐색 영역은 서로 상이한 것을 특징으로 하는 비디오 신호 디코딩 장치.

청구항 10

제 1항에 있어서,

상기 제1 코스트 값들은 상기 하나 이상의 제1 보정된 움직임 정보들 각각에 대해 순차적으로 계산되고,

상기 제2 코스트 값들은 상기 하나 이상의 제2 보정된 움직임 정보들 각각에 대해 순차적으로 계산되는 것을 특징으로 하는 비디오 신호 디코딩 장치.

청구항 11

제 10항에 있어서,

상기 제1 코스트 값들이 순차적으로 계산되는 중 기 설정된 제1 값보다 작은 코스트 값이 계산되는 경우, 나머지 보정된 움직임 정보들에 대한 코스트 값들은 계산되지 않고,

상기 제2 코스트 값들이 순차적으로 계산되는 중 기 설정된 제2 값보다 작은 코스트 값이 계산되는 경우, 나머지 보정된 움직임 정보들에 대한 코스트 값들은 계산되지 않는 것을 특징으로 하는 비디오 신호 디코딩 장치.

청구항 12

비디오 신호 인코딩 장치에 있어서,

프로세서를 포함하며,

상기 프로세서는,

디코딩 방법에 의해 디코딩되는 비트스트림을 획득하고,

상기 디코딩 방법은,

현재 블록과 관련된 하나 이상의 제1 움직임 정보들을 포함하는 제1 움직임 정보 리스트를 획득하는 단계;

상기 하나 이상의 제1 움직임 정보들을 보정하여 하나 이상의 제1 보정된 움직임 정보들을 획득하는 단계;

상기 하나 이상의 제1 보정된 움직임 정보들 각각에 대한 하나 이상의 제1 코스트 값들을 획득하는 단계;

상기 하나 이상의 제1 보정된 움직임 정보들을 상기 제1 코스트 값들에 기초하여 재정렬하여 제2 움직임 정보 리스트를 획득하는 단계;

상기 제2 움직임 정보 리스트에 포함되는 제1 움직임 정보에 기초하여 하나 이상의 제2 움직임 정보들을 획득하는 단계;

상기 하나 이상의 제2 움직임 정보들을 보정하여 하나 이상의 제2 보정된 움직임 정보들을 획득하는 단계;

상기 제2 보정된 움직임 정보들 각각에 대한 하나 이상의 제2 코스트 값들을 획득하는 단계; 및

상기 제2 코스트 값들에 기초하여 결정되는 제2 움직임 정보에 기초하여 상기 현재 블록을 예측하는 단계를 포함하는 것을 특징으로 하는 비디오 신호 인코딩 장치.

청구항 13

제 12항에 있어서,

상기 제2 움직임 정보 리스트는 상기 하나 이상의 제1 보정된 움직임 정보들에 각각 대응되는 코스트 값이 오름차순으로 정렬되는 것을 특징으로 하는 비디오 신호 인코딩 장치.

청구항 14

제 12항에 있어서,

상기 제1 움직임 정보는 상기 제1 코스트 값들 중 가장 작은 코스트 값에 대응하는 보정된 움직임 정보인 것을 특징으로 하는 비디오 신호 인코딩 장치.

청구항 15

제 12항에 있어서,

상기 제2 움직임 정보는 상기 제2 코스트 값들 중 가장 작은 코스트 값에 대응하는 보정된 움직임 정보인 것을 특징으로 하는 비디오 신호 인코딩 장치.

청구항 16

제 12항에 있어서,

상기 하나 이상의 제1 움직임 정보들은 각각 서로 다른 픽처에 포함되고,

상기 하나 이상의 제2 움직임 정보들은 각각 서로 다른 픽처에 포함되는 것을 특징으로 하는 비디오 신호 인코딩 장치.

청구항 17

제 12항에 있어서,

상기 현재 블록의 부호화 모드가 머지(merge) 모드인 경우,

상기 하나 이상의 제1 움직임 정보들 및 상기 하나 이상의 제2 움직임 정보는 MMVD(Merge mode with MVD)에 의해 유도되는 움직임 정보인 것을 특징으로 하는 비디오 신호 인코딩 장치.

청구항 18

제 12항에 있어서,

상기 제1 코스트 값들은 상기 하나 이상의 제1 보정된 움직임 정보들 각각에 대해 순차적으로 계산되고,

상기 제2 코스트 값들은 상기 하나 이상의 제2 보정된 움직임 정보들 각각에 대해 순차적으로 계산되는 것을 특징으로 하는 비디오 신호 인코딩 장치.

청구항 19

제 12항에 있어서,

상기 제1 코스트 값들이 순차적으로 계산되는 중 기 설정된 제1 값보다 작은 코스트 값이 계산되는 경우, 나머지 보정된 움직임 정보들에 대한 코스트 값들은 계산되지 않고,

상기 제2 코스트 값들이 순차적으로 계산되는 중 기 설정된 제2 값보다 작은 코스트 값이 계산되는 경우, 나머지 보정된 움직임 정보들에 대한 코스트 값들은 계산되지 않는 것을 특징으로 하는 비디오 신호 인코딩 장치.

청구항 20

비트스트림을 저장하는 컴퓨터 판독 가능한 비 일시적 저장 매체에 있어서, 상기 비트스트림은 디코딩 방법에 의해 디코딩되고,

상기 디코딩 방법은,

현재 블록과 관련된 하나 이상의 제1 움직임 정보들을 포함하는 제1 움직임 정보 리스트를 획득하는 단계;

상기 하나 이상의 제1 움직임 정보들을 보정하여 하나 이상의 제1 보정된 움직임 정보들을 획득하는 단계;

상기 하나 이상의 제1 보정된 움직임 정보들 각각에 대한 하나 이상의 제1 코스트 값들을 획득하는 단계;

상기 하나 이상의 제1 보정된 움직임 정보들을 상기 제1 코스트 값들에 기초하여 재정렬하여 제2 움직임 정보 리스트를 획득하는 단계;

상기 제2 움직임 정보 리스트에 포함되는 제1 움직임 정보에 기초하여 하나 이상의 제2 움직임 정보들을 획득하는 단계;

상기 하나 이상의 제2 움직임 정보들을 보정하여 하나 이상의 제2 보정된 움직임 정보들을 획득하는 단계;

상기 제2 보정된 움직임 정보들 각각에 대한 하나 이상의 제2 코스트 값들을 획득하는 단계; 및

상기 제2 코스트 값들에 기초하여 결정되는 제2 움직임 정보에 기초하여 상기 현재 블록을 예측하는 단계를 포함하는 것을 특징으로 하는 비 일시적 저장 매체.

발명의 설명

기 술 분 야

[0001] 본 발명은 비디오 신호의 처리 방법 및 장치에 관한 것으로, 보다 상세하게는 비디오 신호를 인코딩하거나 디코딩하는 비디오 신호 처리 방법 및 장치에 관한 것이다.

배 경 기 술

[0002] 압축 부호화란 디지털화한 정보를 통신 회선을 통해 전송하거나, 저장 매체에 적합한 형태로 저장하기 위한 일련의 신호 처리 기술을 의미한다. 압축 부호화의 대상에는 음성, 영상, 문자 등의 대상이 존재하며, 특히 영상을 대상으로 압축 부호화를 수행하는 기술을 비디오 영상 압축이라고 일컫는다. 비디오 신호에 대한 압축 부호화는 공간적인 상관관계, 시간적인 상관관계, 확률적인 상관관계 등을 고려하여 잉여 정보를 제거함으로써 이루어진다. 그러나 최근의 다양한 미디어 및 데이터 전송 매체의 발전으로 인해, 더욱 고효율의 비디오 신호 처리 방법 및 장치가 요구되고 있다.

발명의 내용

해결하려는 과제

[0003] 본 명세서는 비디오 신호 처리 방법 및 이를 위한 장치를 제공하여 비디오 신호의 코딩 효율을 높이기 위한 목

적이 있다.

과제의 해결 수단

[0004] 본 명세서는 비디오 신호 처리 방법 및 이를 위한 장치를 제공한다.

[0005] 본 명세서에 있어서, 비디오 신호 디코딩 장치는 프로세서를 포함하며, 상기 프로세서는, 현재 블록과 관련된 하나 이상의 제1 움직임 정보들을 포함하는 제1 움직임 정보 리스트를 획득하고, 상기 하나 이상의 제1 움직임 정보들을 보정하여 하나 이상의 제1 보정된 움직임 정보들을 획득하고, 상기 하나 이상의 제1 보정된 움직임 정보들 각각에 대한 하나 이상의 제1 코스트 값들을 획득하고, 상기 하나 이상의 제1 보정된 움직임 정보들을 상기 제1 코스트 값들에 기초하여 재정렬하여 제2 움직임 정보 리스트를 획득하고, 상기 제2 움직임 정보 리스트에 포함되는 제1 움직임 정보에 기초하여 하나 이상의 제2 움직임 정보들을 획득하고, 상기 하나 이상의 제2 움직임 정보들을 보정하여 하나 이상의 제2 보정된 움직임 정보들을 획득하고, 상기 제2 보정된 움직임 정보들 각각에 대한 하나 이상의 제2 코스트 값들을 획득하고, 상기 제2 코스트 값들에 기초하여 결정되는 제2 움직임 정보에 기초하여 상기 현재 블록을 예측하는 것을 특징으로 한다.

[0006] 본 명세서에 있어서, 비디오 신호 인코딩 장치는 프로세서를 포함하며 상기 프로세서는, 디코딩 방법에 의해 디코딩되는 비트스트림을 획득하고, 상기 디코딩 방법은, 현재 블록과 관련된 하나 이상의 제1 움직임 정보들을 포함하는 제1 움직임 정보 리스트를 획득하는 단계; 상기 하나 이상의 제1 움직임 정보들을 보정하여 하나 이상의 제1 보정된 움직임 정보들을 획득하는 단계; 상기 하나 이상의 제1 보정된 움직임 정보들 각각에 대한 하나 이상의 제1 코스트 값들을 획득하는 단계; 상기 하나 이상의 제1 보정된 움직임 정보들을 상기 제1 코스트 값들에 기초하여 재정렬하여 제2 움직임 정보 리스트를 획득하는 단계; 상기 제2 움직임 정보 리스트에 포함되는 제1 움직임 정보에 기초하여 하나 이상의 제2 움직임 정보들을 획득하는 단계; 상기 하나 이상의 제2 움직임 정보들을 보정하여 하나 이상의 제2 보정된 움직임 정보들을 획득하는 단계; 상기 제2 보정된 움직임 정보들 각각에 대한 하나 이상의 제2 코스트 값들을 획득하는 단계; 및 상기 제2 코스트 값들에 기초하여 결정되는 제2 움직임 정보에 기초하여 상기 현재 블록을 예측하는 단계를 포함하는 것을 특징으로 한다.

[0007] 본 명세서에 있어서, 비트스트림을 저장하는 컴퓨터 판독 가능한 비 일시적 저장 매체에 있어서, 상기 비트스트림은 디코딩 방법에 의해 디코딩되고, 상기 디코딩 방법은, 현재 블록과 관련된 하나 이상의 제1 움직임 정보들을 포함하는 제1 움직임 정보 리스트를 획득하는 단계; 상기 하나 이상의 제1 움직임 정보들을 보정하여 하나 이상의 제1 보정된 움직임 정보들을 획득하는 단계; 상기 하나 이상의 제1 보정된 움직임 정보들 각각에 대한 하나 이상의 제1 코스트 값들을 획득하는 단계; 상기 하나 이상의 제1 보정된 움직임 정보들을 상기 제1 코스트 값들에 기초하여 재정렬하여 제2 움직임 정보 리스트를 획득하는 단계; 상기 제2 움직임 정보 리스트에 포함되는 제1 움직임 정보에 기초하여 하나 이상의 제2 움직임 정보들을 획득하는 단계; 상기 하나 이상의 제2 움직임 정보들을 보정하여 하나 이상의 제2 보정된 움직임 정보들을 획득하는 단계; 상기 제2 보정된 움직임 정보들 각각에 대한 하나 이상의 제2 코스트 값들을 획득하는 단계; 및 상기 제2 코스트 값들에 기초하여 결정되는 제2 움직임 정보에 기초하여 상기 현재 블록을 예측하는 단계를 포함하는 것을 특징으로 한다.

[0008] 상기 하나 이상의 제1 코스트 값들은 상기 현재 블록과 상기 하나 이상의 제1 보정된 움직임 정보들 각각에 대응되는 참조 블록간의 유사도와 관련된 값이고, 상기 하나 이상의 제2 코스트 값들은 상기 현재 블록과 상기 하나 이상의 제2 보정된 움직임 정보들 각각에 대응되는 참조 블록간의 유사도와 관련된 값인 것을 특징으로 한다.

[0009] 상기 제2 움직임 정보 리스트는 상기 하나 이상의 제1 보정된 움직임 정보들에 각각 대응되는 코스트 값이 오름차순으로 정렬되는 것을 특징으로 한다.

[0010] 상기 제1 움직임 정보는 상기 제1 코스트 값들 중 가장 작은 코스트 값에 대응하는 보정된 움직임 정보인 것을 특징으로 한다.

[0011] 상기 제2 움직임 정보는 상기 제2 코스트 값들 중 가장 작은 코스트 값에 대응하는 보정된 움직임 정보인 것을 특징으로 한다.

[0012] 상기 하나 이상의 제1 움직임 정보들은 각각 서로 다른 픽처에 포함되고, 상기 하나 이상의 제2 움직임 정보들은 각각 서로 다른 픽처에 포함되는 것을 특징으로 한다.

[0013] 상기 하나 이상의 제1 보정된 움직임 정보들 및 상기 하나 이상의 제2 보정된 움직임 정보들은, MVD(Motion Vector Difference), TM(Template Matching), BM(Bilateral Matching), MMVD(Merge mode with MVD),

MMVD(Merge mode with MVD) 기반의 TM, 광 흐름(Optical flow)에 기초한 기반 TM, 멀티 패스 DMVR (Multi pass Decoder-side Motion Vector Refinement) 중 적어도 어느 하나 이상에 기초하여 보정되는 것을 특징으로 한다.

[0014] 상기 현재 블록의 부호화 모드가 머지(merge) 모드인 경우, 상기 하나 이상의 제1 움직임 정보들 및 상기 하나 이상의 제2 움직임 정보들은 MMVD(Merge mode with MVD)에 의해 유도되는 움직임 정보인 것을 특징으로 한다.

[0015] 상기 하나 이상의 제1 움직임 정보들 각각에 대응되는 블록들은 제1 탐색 영역 내에 위치하는 블록이고, 상기 하나 이상의 제2 움직임 정보들 각각에 대응되는 블록들은 제2 탐색 영역 내에 위치하는 블록이고, 상기 제1 탐색 영역과 상기 제2 탐색 영역은 서로 상이한 것을 특징으로 한다.

[0016] 상기 제1 코스트 값들은 상기 하나 이상의 제1 보정된 움직임 정보들 각각에 대해 순차적으로 계산되고, 상기 제2 코스트 값들은 상기 하나 이상의 제2 보정된 움직임 정보들 각각에 대해 순차적으로 계산되는 것을 특징으로 한다.

[0017] 상기 제1 코스트 값들이 순차적으로 계산되는 중 기 설정된 제1 값보다 작은 코스트 값이 계산되는 경우, 나머지 보정된 움직임 정보들에 대한 코스트 값들은 계산되지 않고, 상기 제2 코스트 값들이 순차적으로 계산되는 중 기 설정된 제2 값보다 작은 코스트 값이 계산되는 경우, 나머지 보정된 움직임 정보들에 대한 코스트 값들은 계산되지 않는 것을 특징으로 한다.

발명의 효과

[0018] 본 명세서는 효율적으로 비디오 신호를 처리하기 위한 방법을 제공한다.

[0019] 본 명세서에서 얻을 수 있는 효과는 이상에서 언급한 효과들로 제한되지 않으며, 언급하지 않은 또 다른 효과들은 아래의 기재로부터 본 발명이 속하는 기술분야에서 통상의 지식을 가진 자에게 명확하게 이해될 수 있을 것이다.

도면의 간단한 설명

[0020] 도 1은 본 발명의 일 실시예에 따른 비디오 신호 인코딩 장치의 개략적인 블록도이다.

도 2는 본 발명의 일 실시예에 따른 비디오 신호 디코딩 장치의 개략적인 블록도이다.

도 3은 픽쳐 내에서 코딩 트리 유닛이 코딩 유닛들로 분할되는 실시예를 도시한다.

도 4는 쿼드 트리 및 멀티-타입 트리의 분할을 시그널링하는 방법의 일 실시예를 도시한다.

도 5 및 도 6은 본 발명의 실시예에 따른 인트라 예측 방법을 더욱 구체적으로 도시한다.

도 7은 인터 예측에서 움직임 후보 리스트를 구성하기 위해 사용되는 주변 블록들의 위치를 나타낸 도면이다.

도 8은 본 발명의 일 실시예에 따른 움직임 정보를 보정하는 방법을 나타내는 도면이다.

도 9는 본 발명의 일 실시예에 따른 움직임 보정 방법을 재귀적으로 수행하여 현재 블록에 대한 움직임 정보를 보정하는 방법을 나타내는 도면이다.

도 10은 본 발명의 일 실시예에 따른 TM 방법이 수행되는 순서를 나타내는 도면이다.

도 11은 본 발명의 일 실시예에 따른 초기 움직임 정보에 기초하여 TM 방법을 위한 탐색 영역을 설정하는 방법을 나타내는 도면이다.

도 12는 본 발명의 일 실시예에 따른 탐색 영역 내에서 탐색되는 움직임 후보의 위치를 나타내는 도면이다.

도 13은 본 발명의 일 실시예에 따른 움직임 후보의 위치를 탐색하는 과정을 나타내는 도면이다.

도 14는 본 발명의 일 실시예 따른 탐색 후보를 평가하는 과정을 나타내는 도면이다.

도 15는 본 발명의 일 실시예에 따른 현재 픽쳐 내 객체의 경계 부분에 대한 움직임 특성을 나타내는 도면이다.

도 16은 본 발명의 일 실시예에 따른 템플릿 매칭을 이용하여 참조 픽쳐를 결정하는 방법을 나타내는 도면이다.

도 17, 도 18은 본 발명의 일 실시예에 따른 템플릿 매칭을 수행하는 과정을 나타내는 도면이다.

도 19는 본 발명의 일 실시예에 따른 현재 블록의 주변 블록으로부터 유도되는 초기 움직임 정보들이 서로 붙어

있는 경우를 나타내는 도면이다.

도 20, 21은 본 발명의 일 실시예에 따른 초기 움직임 정보를 변경하는 방법을 나타내는 도면이다.

도 22는 본 발명의 일 실시예에 따른 초기 움직임 정보를 통해 유도되는 움직임 후보가 탐색되는 위치를 나타내는 도면이다.

도 23은 본 발명의 일 실시예에 따른 움직임 후보가 탐색되는 탐색 위치를 변경하는 방법을 나타내는 도면이다.

도 24는 본 발명의 일 실시예에 따른 움직임 벡터의 코스트 값에 기초한 TM을 나타내는 도면이다.

도 25, 도 26은 본 발명의 일 실시예에 따른 DMVR을 사용하여 움직임 정보를 보정하는 방법을 나타내는 도면이다.

도 27은 본 발명의 일 실시예에 따른 다중 DMVR이 수행되는 과정을 나타내는 도면이다.

도 28은 본 발명의 일 실시예에 따른 코딩 블록의 보정된 움직임 정보와 관련된 코스트 값을 획득하기 위한 탐색 방법을 나타내는 도면이다.

도 29는 본 발명의 일 실시예에 따른 3x3 Square 탐색 방법을 나타내는 도면이다.

도 30은 본 발명의 일 실시예에 따른 정수 단위 전역 탐색 과정에서 탐색 영역을 구역별로 나누어 진 것을 나타낸 도면이다.

도 31, 도 32는 본 발명의 일 실시예에 따른 정수 단위의 전역 탐색 과정을 나타내는 도면이다.

도 33, 34는 본 발명의 일 실시예에 따른 BDOF에 기반한 움직임 정보의 보정을 수행하는 방법을 나타내는 도면이다.

도 35, 도 36은 본 발명의 일 실시예에 따른 BDOF에 기반한 현재 블록에 대한 예측 블록을 생성하는 방법을 나타내는 도면이다.

도 37은 본 발명의 일 실시예에 따른 DMVR이 적용되는 서브블록을 나타내는 도면이다.

도 38은 본 발명의 일 실시예에 따른 다중 DMVR을 수행하면서 현재 블록에 대한 예측 블록을 생성하는 방법을 나타내는 도면이다.

발명을 실시하기 위한 구체적인 내용

[0021] 본 명세서에서 사용되는 용어는 본 발명에서의 기능을 고려하면서 가능한 현재 널리 사용되는 일반적인 용어를 선택하였으나, 이는 당 분야에 종사하는 기술자의 의도, 관례 또는 새로운 기술의 출현 등에 따라 달라질 수 있다. 또한 특정 경우는 출원인이 임의로 선정한 용어도 있으며, 이 경우 해당되는 발명의 설명 부분에서 그 의미를 기재할 것이다. 따라서 본 명세서에서 사용되는 용어는, 단순한 용어의 명칭이 아닌 그 용어가 가진 실질적인 의미와 본 명세서의 전반에 걸친 내용을 토대로 해석되어야 함을 밝혀두고자 한다.

[0022] 본 명세서에서 'A 및/또는 B'는 'A 또는 B 중 적어도 하나를 포함하는'과 같은 의미로 해석될 수 있다.

[0023] 본 명세서에서 일부 용어들은 다음과 같이 해석될 수 있다. 코딩은 경우에 따라 인코딩 또는 디코딩으로 해석될 수 있다. 본 명세서에서 비디오 신호의 인코딩(부호화)을 수행하여 비디오 신호 비트스트림을 생성하는 장치는 인코딩 장치 또는 인코더로 지칭되며, 비디오 신호 비트스트림의 디코딩(복호화)을 수행하여 비디오 신호를 복원하는 장치는 디코딩 장치 또는 디코더로 지칭된다. 또한, 본 명세서에서 비디오 신호 처리 장치는 인코더 및 디코더를 모두 포함하는 개념의 용어로 사용된다. 정보(information)는 값(values), 파라미터(parameter), 계수(coefficients), 성분(elements) 등을 모두 포함하는 용어로서, 경우에 따라 의미는 달리 해석될 수 있으므로 본 발명은 이에 한정되지 아니한다. '유닛'은 영상 처리의 기본 단위 또는 픽처의 특정 위치를 지칭하는 의미로 사용되며, 휘도(luma) 성분 및 색차(chroma) 성분 중 적어도 하나를 포함하는 이미지 영역을 가리킨다. 또한, '블록'은 휘도 성분 및 색차 성분들(즉, Cb 및 Cr) 중 특정 성분을 포함하는 이미지 영역을 가리킨다. 다만, 실시예에 따라서 '유닛', '블록', '파티션', '신호' 및 '영역' 등의 용어는 서로 혼용하여 사용될 수 있다. 또한, 본 명세서에서 '현재 블록'은 현재 부호화를 진행할 예정인 블록을 의미하며, '참조 블록'은 이미 부호화 또는 복호화가 완료된 블록으로 현재 블록에서 참조로 사용되는 블록을 의미한다. 또한, 본 명세서에서 '루마', 'luma', '휘도', 'Y' 등의 용어는 서로 혼용하여 사용될 수 있다. 또한, 본 명세서에서 '크로마', 'chroma' '색차', 'Cb 또는 Cr' 등의 용어는 서로 혼용하여 사용될 수 있으며, 색차 성분은 Cb와 Cr 2가지로 나누어지므로

각 색차 성분은 구분되어 사용될 수 있다. 또한, 본 명세서에서 유닛은 코딩 유닛, 예측 유닛, 변환 유닛을 모두 포함하는 개념으로 사용될 수 있다. 픽쳐는 필드 또는 프레임을 가리키며, 실시예에 따라 상기 용어들은 서로 혼용하여 사용될 수 있다. 구체적으로 촬영된 영상이 비월주사식(interlace) 영상일 경우, 하나의 프레임은 홀수(또는 기수, top) 필드와 짝수(또는 우수, bottom) 필드로 분리되어, 각 필드는 하나의 픽쳐 단위로 구성되어 부호화 또는 복호화 될 수 있다. 만일 촬영된 영상이 순차주사(progressive) 영상일 경우, 하나의 프레임이 픽쳐로서 구성되어 부호화 또는 복호화 될 수 있다. 또한, 본 명세서에서 '오차 신호', '레지듀얼 신호', '잔차 신호', '잔여 신호' 및 '차분 신호' 등의 용어는 서로 혼용하여 사용될 수 있다. 또한, 본 명세서에서 '인트라 예측 모드', '인트라 예측 방향성 모드', '화면 내 예측 모드' 및 '화면 내 예측 방향성 모드' 등의 용어는 서로 혼용하여 사용될 수 있다. 또한, 본 명세서에서 '모션', '움직임' 등의 용어는 서로 혼용하여 사용될 수 있다. 또한, 본 명세서에서 '좌측', '좌상측', '상측', '우상측', '우측', '우하측', '하측', '좌하측'은 '좌단', '좌상단', '상단', '우상단', '우단', '우하단', '하단', '좌하단'와 서로 혼용하여 사용될 수 있다. 또한, 원소(element), 멤버(member)는 서로 혼용하여 사용될 수 있다. POC(Picture Order Count)는 픽쳐(또는 프레임)의 시간적 위치 정보를 나타내며, 화면에 출력되는 재생 순서가 될 수 있으며, 픽쳐마다 고유의 POC를 가질 수 있다.

[0024] 도 1은 본 발명의 일 실시예에 따른 비디오 신호 인코딩 장치(100)의 개략적인 블록도이다. 도 1을 참조하면, 본 발명의 인코딩 장치(100)는 변환부(110), 양자화부(115), 역양자화부(120), 역변환부(125), 필터링부(130), 예측부(150) 및 엔트로피 코딩부(160)를 포함한다.

[0025] 변환부(110)는 입력 받은 비디오 신호와 예측부(150)에서 생성된 예측 신호의 차이인 레지듀얼 신호를 변환하여 변환 계수 값을 획득한다. 예를 들어, 이산 코사인 변환(Discrete Cosine Transform, DCT), 이산 사인 변환(Discrete Sine Transform, DST) 또는 웨이블릿 변환(Wavelet Transform) 등이 사용될 수 있다. 이산 코사인 변환 및 이산 사인 변환은 입력된 픽쳐 신호를 블록 형태로 나누어 변환을 수행하게 된다. 변환에 있어서 변환 영역 내의 값들의 분포와 특성에 따라서 코딩 효율이 달라질 수 있다. 레지듀얼 블록에 대한 변환에 사용되는 변환 커널은 수직 변환 및 수평 변환의 분리 가능한 특성을 가지는 변환 커널일 수 있다. 이 경우, 레지듀얼 블록에 대한 변환은 수직 변환 및 수평 변환으로 분리되어 수행될 수 있다. 예를 들어, 인코더는 레지듀얼 블록의 수직 방향으로 변환 커널을 적용하여 수직 변환을 수행할 수 있다. 또한, 인코더는 레지듀얼 블록의 수평 방향으로 변환 커널을 적용하여 수평 변환을 수행할 수 있다. 본 개시에서, 변환 커널은 변환 매트릭스, 변환 어레이, 변환 함수, 변환과 같이 레지듀얼 신호의 변환에 사용되는 파라미터 세트를 지칭하는 용어로 사용될 수 있다. 예를 들어, 변환 커널은 복수의 사용 가능한 커널들 중 어느 하나일 수 있다. 또한, 수직 변환 및 수평 변환 각각에 대해 서로 다른 변환 타입에 기반한 변환 커널이 사용될 수도 있다.

[0026] 변환계수는 블록의 좌상단으로 갈수록 높은 계수가 분포하고, 블록의 우하단으로 갈수록 '0'에 가까운 계수가 분포한다. 현재 블록의 크기가 커질수록 우하단 영역에서 계수 '0'이 많이 존재할 가능성이 있다. 크기가 큰 블록의 변환 복잡도를 감소시키기 위해서, 임의의 좌상단 영역만을 남기고 나머지 영역은 '0'으로 재설정될 수 있다.

[0027] 또한, 코딩 블록에서 일부 영역에만 오차 신호가 존재할 수 있다. 이 경우, 임의의 일부 영역에 대해서만 변환 과정이 수행될 수 있다. 실시 일 예로, 2Nx2N 크기의 블록에서 첫번째 2NxN 블록에만 오차 신호가 존재할 수 있으며, 첫번째 2NxN블록에만 변환과정이 수행되지만 두번째 2NxN 블록은 변환과정이 수행되지 않고 인코딩 또는 디코딩되지 않을 수 있다. 여기서 N은 임의의 양의 정수가 될 수 있다.

[0028] 인코더는 변환 계수가 양자화되기 전에 추가적인 변환을 수행할 수 있다. 전술한 변환 방법은 1차 변환(primary transform)으로 지칭되고, 추가적인 변환은 2차 변환(secondary transform)으로 지칭될 수 있다. 2차 변환은 레지듀얼 블록 별로 선택적일 수 있다. 일 실시예에 따라, 인코더는 1차 변환만으로 저주파 영역에 에너지를 집중시키기 어려운 영역에 대해 2차 변환을 수행하여 코딩 효율을 향상시킬 수 있다. 예를 들어, 레지듀얼 값들이 레지듀얼 블록의 수평 또는 수직 방향 이외의 방향에서 크게 나타나는 블록에 대해 2차 변환이 추가로 수행될 수 있다. 2차 변환은 1차 변환과 달리 수직 변환 및 수평 변환으로 분리되어 수행되지 않을 수 있다. 이러한 2차 변환은 저대역 비-분리 변환(Low Frequency Non-Separable Transform, LFNST)으로 지칭될 수 있다.

[0029] 양자화부(115)는 변환부(110)에서 출력된 변환 계수 값을 양자화한다.

[0030] 코딩 효율을 높이기 위하여 픽쳐 신호를 그대로 코딩하는 것이 아니라, 예측부(150)를 통해 이미 코딩된 영역을 이용하여 픽쳐를 예측하고, 예측된 픽쳐에 원본 픽쳐와 예측 픽쳐 간의 레지듀얼 값을 더하여 복원 픽쳐를 획득하는 방법이 사용된다. 인코더와 디코더에서 미스매치가 발생되지 않도록 하기 위해, 인코더에서 예측을 수행할

때에는 디코더에서도 사용 가능한 정보를 사용해야 한다. 이를 위해, 인코더에서는 부호화한 현재 블록을 다시 복원하는 과정을 수행한다. 역양자화부(120)에서는 변환 계수 값을 역양자화하고, 역변환부(125)에서는 역양자화된 변환 계수값을 이용하여 레지듀얼 값을 복원한다. 한편, 필터링부(130)는 복원된 픽처의 품질 개선 및 부호화 효율 향상을 위한 필터링 연산을 수행한다. 예를 들어, 디블록킹 필터, 샘플 적응적 오프셋(Sample Adaptive Offset, SAO) 및 적응적 루프 필터 등이 포함될 수 있다. 필터링을 거친 픽처는 출력되거나 참조 픽처로 이용하기 위하여 복호 픽처 버퍼(Decoded Picture Buffer, DPB, 156)에 저장된다.

[0031] 디블록킹 필터(deblocking filter)는 복원된 픽처에서 블록 간의 경계에 생성된 블록 내의 왜곡을 제거하기 위한 필터이다. 인코더는 블록 내의 임의 경계(edge)를 기준으로 몇 개의 열 또는 행에 포함된 픽셀들의 분포를 통해, 해당 경계에 디블록킹 필터를 적용할지 여부를 판단할 수 있다. 블록에 디블록킹 필터를 적용되는 경우, 인코더는 디블록킹 필터링 강도에 따라 긴 필터(Long Filter), 강한 필터(Strong Filter) 또는 약한 필터(Weak Filter)를 적용할 수 있다. 또한, 수평 방향 필터링 및 수직 방향 필터링이 병렬적으로 처리될 수 있다. 샘플 적응적 오프셋(SAO)은 디블록킹 필터가 적용된 레지듀얼 블록에 대하여, 픽셀 단위로 원본 영상과의 오프셋을 보정하는데 사용될 수 있다. 인코더는 특정 픽처에 대한 오프셋을 보정하기 위하여 영상에 포함된 픽셀을 일정한 수의 영역으로 구분한 후, 오프셋 보정을 수행할 영역을 결정하고, 해당 영역에 오프셋을 적용하는 방법(Band Offset)을 사용할 수 있다. 또는 인코더는 각 픽셀의 에지 정보를 고려하여 오프셋을 적용하는 방법(Edge Offset)을 사용할 수 있다. 적응적 루프 필터(Adaptive Loop Filter, ALF)는 영상에 포함된 픽셀을 소정의 그룹으로 나눈 후, 해당 그룹에 적용될 하나의 필터를 결정하여 그룹마다 차별적으로 필터링을 수행하는 방법이다. ALF를 적용할지 여부에 관련된 정보는 코딩 유닛 단위로 시그널링될 수 있고, 각각의 블록에 따라 적용될 ALF 필터의 모양 및 필터 계수가 달라질 수 있다. 또한, 적용할 대상 블록의 특성에 관계없이 동일한 형태(고정된 형태)의 ALF 필터가 적용될 수도 있다.

[0032] 예측부(150)는 인트라 예측부(152)와 인터 예측부(154)를 포함한다. 인트라 예측부(152)에서는 현재 픽처 내에서 인트라(intra) 예측을 수행하며, 인터 예측부(154)에서는 복호 픽처 버퍼(156)에 저장된 참조 픽처를 이용하여 현재 픽처를 예측하는 인터(inter) 예측을 수행한다. 인트라 예측부(152)는 현재 픽처 내의 복원된 영역들로부터 인트라 예측을 수행하여, 인트라 부호화 정보를 엔트로피 코딩부(160)에 전달한다. 인트라 부호화 정보는 인트라 예측 모드, MPM(Most Probable Mode) 플래그, MPM 인덱스, 참조 샘플에 관한 정보 중 적어도 하나를 포함할 수 있다. 인터 예측부(154)는 다시 모션 추정부(154a) 및 모션 보상부(154b)를 포함하여 구성될 수 있다. 모션 추정부(154a)에서는 복원된 참조 픽처의 특정 영역을 참조하여 현재 영역과 가장 유사한 부분을 찾고 영역 간의 거리인 모션 벡터 값을 획득한다. 모션 추정부(154a)에서 획득한 참조 영역에 대한 모션 정보(참조 방향 지시 정보(L0 예측, L1 예측, 양방향 예측), 참조 픽처 인덱스, 모션 벡터 정보 등) 등을 엔트로피 코딩부(160)로 전달하여 비트스트림에 포함될 수 있도록 한다. 모션 추정부(154a)에서 전달된 모션 정보를 이용하여 모션 보상부(154b)에서는 인터 모션 보상을 수행하여 현재 블록을 위한 예측 블록을 생성한다. 인터 예측부(154)는 참조 영역에 대한 모션 정보를 포함하는 인터 부호화 정보를 엔트로피 코딩부(160)에 전달한다.

[0033] 추가적인 실시예에 따라, 예측부(150)는 인트라 블록 카피(Intra block copy, IBC) 예측부(미도시)를 포함할 수 있다. IBC 예측부는 현재 픽처 내의 복원된 샘플들로부터 IBC 예측을 수행하여, IBC 부호화 정보를 엔트로피 코딩부(160)에 전달한다. IBC 예측부는 현재 픽처 내의 특정 영역을 참조하여 현재 영역의 예측에 이용되는 참조 영역을 지시하는 블록 벡터값을 획득한다. IBC 예측부는 획득된 블록 벡터값을 이용하여 IBC 예측을 수행할 수 있다. IBC 예측부는 IBC 부호화 정보를 엔트로피 코딩부(160)로 전달한다. IBC 부호화 정보는 참조 영역의 크기 정보, 블록 벡터 정보(움직임 후보 리스트 내에서 현재 블록의 블록 벡터 예측을 위한 인덱스 정보, 블록 벡터 차분 정보) 중에서 적어도 하나를 포함할 수 있다.

[0034] 위와 같은 픽처 예측이 수행될 경우, 변환부(110)는 원본 픽처와 예측 픽처 간의 레지듀얼 값을 변환하여 변환 계수 값을 획득한다. 이때, 변환은 픽처 내에서 특정 블록 단위로 수행될 수 있으며, 특정 블록의 크기는 기 설정된 범위 내에서 가변할 수 있다. 양자화부(115)는 변환부(110)에서 생성된 변환 계수 값을 양자화하여 양자화된 변환 계수를 엔트로피 코딩부(160)로 전달한다.

[0035] 상기 2차원 배열 형태의 양자화된 변환 계수는 엔트로피 코딩을 위해 1차원의 배열 형태로 재정렬될 수 있다. 양자화된 변환 계수를 스캐닝하는 방법은 변환 블록의 크기 및 화면 내 예측 모드에 따라 어떠한 스캔 방법이 사용될지 여부가 결정될 수 있다. 실시 일 예로, 대각(Diagonal), 수직(vertical), 수평(horizontal) 스캔이 적용될 수 있다. 이러한 스캔 정보는 블록 단위로 시그널링될 수 있으며, 이미 정해진 규칙에 따라 유도될 수 있다.

[0036] 엔트로피 코딩부(160)는 양자화된 변환 계수를 나타내는 정보, 인트라 부호화 정보, 및 인터 부호화 정보 등을 엔트로피 코딩하여 비디오 신호 비트스트림을 생성한다. 엔트로피 코딩부(160)에서는 가변 길이 코딩(Variable Length Coding, VLC) 방식과 산술 코딩(arithmetic coding) 방식 등이 사용될 수 있다. 가변 길이 코딩(VLC) 방식은 입력되는 심볼들을 연속적인 코드워드로 변환하는데, 코드워드의 길이는 가변적일 수 있다. 예를 들어, 자주 발생하는 심볼들을 짧은 코드워드로, 자주 발생하지 않은 심볼들은 긴 코드워드로 표현하는 것이다. 가변 길이 코딩 방식으로서 컨텍스트 기반 적응형 가변 길이 코딩(Context-based Adaptive Variable Length Coding, CAVLC) 방식이 사용될 수 있다. 산술 코딩은 각 데이터 심볼들의 확률 분포를 이용하여 연속적인 데이터 심볼들을 하나의 소수로 변환하는데, 산술 코딩은 각 심볼을 표현하기 위하여 필요한 최적의 소수 비트를 얻을 수 있다. 산술 코딩으로서 컨텍스트 기반 적응형 산술 부호화(Context-based Adaptive Binary Arithmetic Code, CABAC)가 이용될 수 있다.

[0037] CABAC은 실험을 통해 얻은 확률을 기반으로 생성된 여러 개의 문맥 모델(context model)을 통해 이진 산술 부호화하는 방법이다. 먼저, 심볼이 이진 형태가 아닐 경우, 인코더는 exp-Golomb 등을 사용하여 각 심볼을 이진화한다. 이진화된 0 또는 1은 빈(bin)으로 기술될 수 있다. CABAC 초기화 과정은 문맥 초기화와 산술 코딩 초기화로 구분된다. 문맥 초기화는 각 심볼의 발생 확률을 초기화하는 과정으로, 심볼의 종류, 양자화 파라미터(QP), 슬라이스 타입(I, P, B 인지)에 따라 결정된다. 이러한 초기화 정보를 가지는 문맥 모델은 실험을 통해 얻은 확률 기반 값을 사용할 수 있다. 문맥 모델은 현재 코딩하려는 심볼에 대한 LPS(Least Probable Symbol) 또는 MPS(Most Probable Symbol)의 발생 확률과 0과 1중에서 어떤 빈 값이 MPS에 해당되는지에 대한 정보(valMPS)를 제공한다. 문맥 인덱스(Context index, ctxIdx)를 통해 여러 개의 문맥 모델 중에서 하나가 선택되며, 문맥 인덱스는 현재 부호화할 블록의 정보 또는 주변 블록의 정보를 통해 유도될 수 있다. 문맥 모델에서 선택된 확률 모델을 기반으로 이진 산술 코딩을 위한 초기화가 수행된다. 이진 산술 부호화는 0과 1의 발생 확률을 통해 확률 구간으로 분할한 후, 처리할 빈에 해당하는 확률 구간이 다음에 처리될 빈에 대한 전체 확률 구간이 되는 과정을 통해 부호화가 진행된다. 마지막 빈이 처리된 확률 구간 안의 위치 정보가 출력된다. 단, 확률 구간이 무한정 분할될 수 없으므로, 일정 크기 이내로 줄어들 경우에는 재규격화(renormalization)과정이 수행되어 확률 구간이 넓어지고 해당 위치 정보가 출력된다. 또한, 각 빈이 처리된 후, 처리된 빈의 정보를 통해 다음 처리될 빈에 대한 확률이 새롭게 설정되는 확률 업데이트 과정이 수행될 수 있다.

[0038] 상기 생성된 비트스트림은 NAL(Network Abstraction Layer) 유닛을 기본 단위로 캡슐화 된다. NAL 유닛은 영상 데이터를 포함하는 VCL(Video Coding Layer) NAL 유닛과 영상 데이터를 디코딩하기 위한 파라미터 정보를 포함하는 non-VCL NAL 유닛으로 구분되며, 다양한 종류의 VCL 또는 non-VCL NAL 유닛이 존재한다. NAL 유닛은 NAL 헤더 정보와 데이터인 RBSP(Raw Byte Sequence Payload)로 구성되며, NAL 헤더 정보에는 RBSP에 대한 요약 정보가 포함된다. VCL NAL 유닛의 RBSP에는 부호화된 정수 개의 코딩 트리 유닛(coding tree unit)을 포함한다. 비디오 디코더에서 비트스트림을 복호화하기 위해서는 먼저 비트스트림을 NAL 유닛 단위로 분리한 후, 분리된 각각의 NAL 유닛을 복호화해야 한다. 한편, 비디오 신호 비트스트림의 복호화를 위해 필요한 정보들은 픽처 파라미터 세트(Picture Parameter Set, PPS), 시퀀스 파라미터 세트(Sequence Parameter Set, SPS), 비디오 파라미터 세트(Video Parameter Set, VPS) 등에 포함되어 전송될 수 있다.

[0039] 한편, 도 1의 블록도는 본 발명의 일 실시예에 따른 인코딩 장치(100)를 나타낸 것으로서, 분리하여 표시된 블록들은 인코딩 장치(100)의 엘리먼트들을 논리적으로 구별하여 도시한 것이다. 따라서 전술한 인코딩 장치(100)의 엘리먼트들은 디바이스의 설계에 따라 하나의 칩으로 또는 복수의 칩으로 장착될 수 있다. 일 실시예에 따르면, 전술한 인코딩 장치(100)의 각 엘리먼트의 동작은 프로세서(미도시)에 의해 수행될 수 있다.

[0040] 도 2는 본 발명의 일 실시예에 따른 비디오 신호 디코딩 장치(200)의 개략적인 블록도이다. 도 2를 참조하면 본 발명의 디코딩 장치(200)는 엔트로피 디코딩부(210), 역양자화부(220), 역변환부(225), 필터링부(230) 및 예측부(250)를 포함한다.

[0041] 엔트로피 디코딩부(210)는 비디오 신호 비트스트림을 엔트로피 디코딩하여, 각 영역에 대한 변환 계수 정보, 인트라 부호화 정보, 인터 부호화 정보 등을 추출한다. 예를 들어, 엔트로피 디코딩부(210)는 비디오 신호 비트스트림으로부터 특정 영역의 변환 계수 정보에 대한 이진화 코드를 획득할 수 있다. 또한, 엔트로피 디코딩부(210)는 이진화 코드를 역 이진화하여 양자화된 변환 계수를 획득한다. 역양자화부(220)는 양자화된 변환 계수를 역양자화하고, 역변환부(225)는 역양자화된 변환 계수를 이용하여 레지듀얼 값을 복원한다. 비디오 신호 처리 장치(200)는 역변환부(225)에서 획득된 레지듀얼 값을 예측부(250)에서 획득된 예측 값과 합산하여 원래의 화소값을 복원한다.

[0042] 한편, 필터링부(230)는 픽처에 대한 필터링을 수행하여 화질을 향상시킨다. 여기에는 블록 왜곡 현상을 감소시키기 위한 디블록킹 필터 및/또는 픽처 전체의 왜곡 제거를 위한 적응적 루프 필터 등이 포함될 수 있다. 필터링을 거친 픽처는 출력되거나 다음 픽처에 대한 참조 픽처로 이용하기 위하여 복호 픽처 버퍼(DPB, 256)에 저장된다.

[0043] 예측부(250)는 인트라 예측부(252) 및 인터 예측부(254)를 포함한다. 예측부(250)는 전술한 엔트로피 디코딩부(210)를 통해 복호화된 부호화 타입, 각 영역에 대한 변환 계수, 인트라/인터 부호화 정보 등을 활용하여 예측 픽처를 생성한다. 복호화가 수행되는 현재 블록을 복원하기 위해서, 현재 블록이 포함된 현재 픽처 또는 다른 픽처들의 복호화된 영역이 이용될 수 있다. 복원에 현재 픽처만을 이용하는, 즉 인트라 예측 또는 인트라 BC 예측을 수행하는 픽처(또는, 타일/슬라이스)를 인트라 픽처 또는 I 픽처(또는, 타일/슬라이스), 인트라 예측, 인터 예측 및 인트라 BC 예측을 모두 수행할 수 있는 픽처(또는, 타일/슬라이스)를 인터 픽처(또는, 타일/슬라이스)라고 한다. 인터 픽처(또는, 타일/슬라이스) 중 각 블록의 샘플값들을 예측하기 위하여 최대 하나의 모션 벡터 및 참조 픽처 인덱스를 이용하는 픽처(또는, 타일/슬라이스)를 예측 픽처(predictive picture) 또는 P 픽처(또는, 타일/슬라이스)라고 하며, 최대 두 개의 모션 벡터 및 참조 픽처 인덱스를 이용하는 픽처(또는, 타일/슬라이스)를 쌍예측 픽처(Bi-predictive picture) 또는 B 픽처(또는, 타일/슬라이스) 라고 한다. 다시 말해서, P 픽처(또는, 타일/슬라이스)는 각 블록을 예측하기 위해 최대 하나의 모션 정보 세트를 이용하고, B 픽처(또는, 타일/슬라이스)는 각 블록을 예측하기 위해 최대 두 개의 모션 정보 세트를 이용한다. 여기서, 모션 정보 세트는 하나 이상의 모션 벡터와 하나의 참조 픽처 인덱스를 포함한다.

[0044] 인트라 예측부(252)는 인트라 부호화 정보 및 현재 픽처 내의 복원된 샘플들을 이용하여 예측 블록을 생성한다. 전술한 바와 같이, 인트라 부호화 정보는 인트라 예측 모드, MPM(Most Probable Mode) 플래그, MPM 인덱스 중 적어도 하나를 포함할 수 있다. 인트라 예측부(252)는 현재 블록의 좌측 및/또는 상측에 위치한 복원된 샘플들을 참조 샘플들로 이용하여 현재 블록의 샘플 값들을 예측한다. 본 개시에서, 복원된 샘플들, 참조 샘플들 및 현재 블록의 샘플들은 픽셀들을 나타낼 수 있다. 또한, 샘플 값(sample value)들은 픽셀 값들을 나타낼 수 있다.

[0045] 일 실시예에 따르면, 참조 샘플들은 현재 블록의 주변 블록에 포함된 샘플들일 수 있다. 예를 들어, 참조 샘플들은 현재 블록의 좌측 경계에 인접한 샘플들 및/또는 상측 경계에 인접한 샘플들일 수 있다. 또한, 참조 샘플들은 현재 블록의 주변 블록의 샘플들 중 현재 블록의 좌측 경계로부터 기 설정된 거리 이내의 라인 상에 위치하는 샘플들 및/또는 현재 블록의 상측 경계로부터 기 설정된 거리 이내의 라인 상에 위치하는 샘플들일 수 있다. 이때, 현재 블록의 주변 블록은 현재 블록에 인접한 좌측(L) 블록, 상측(A) 블록, 하좌측(Below Left, BL) 블록, 상우측(Above Right, AR) 블록 또는 상좌측(Above Left, AL) 블록 중 적어도 하나를 포함할 수 있다.

[0046] 인터 예측부(254)는 복호 픽처 버퍼(256)에 저장된 참조 픽처 및 인터 부호화 정보를 이용하여 예측 블록을 생성한다. 인터 부호화 정보는 참조 블록에 대한 현재 블록의 모션 정보 세트(참조 픽처 인덱스, 모션 벡터 정보 등)를 포함할 수 있다. 인터 예측에는 L0 예측, L1 예측 및 쌍예측(Bi-prediction)이 있을 수 있다. L0 예측은 L0 픽처 리스트에 포함된 1개의 참조 픽처를 이용한 예측이고, L1 예측은 L1 픽처 리스트에 포함된 1개의 참조 픽처를 이용한 예측을 의미한다. 이를 위해서는 1세트의 모션 정보(예를 들어, 모션 벡터 및 참조 픽처 인덱스)가 필요할 수 있다. 쌍예측 방식에서는 최대 2개의 참조 영역을 이용할 수 있는데, 이 2개의 참조 영역은 동일한 참조 픽처에 존재할 수도 있고, 서로 다른 픽처에 각각 존재할 수도 있다. 즉, 쌍예측 방식에서는 최대 2세트의 모션 정보(예를 들어, 모션 벡터 및 참조 픽처 인덱스)가 이용될 수 있는데, 2개의 모션 벡터가 동일한 참조 픽처 인덱스에 대응될 수도 있고 서로 다른 참조 픽처 인덱스에 대응될 수도 있다. 이때, 참조 픽처들은 현재 픽처를 기준으로 시간적으로 이전 또는 이후에 위치하는 픽처로서, 이미 복원된 완료된 픽처가 될 수 있다. 일 실시예에 따라, 쌍예측 방식에서는 사용되는 2개의 참조 영역은 L0 픽처 리스트 및 L1 픽처 리스트 각각에서 선택된 영역일 수 있다.

[0047] 인터 예측부(254)는 모션 벡터 및 참조 픽처 인덱스를 이용하여 현재 블록의 참조 블록을 획득할 수 있다. 상기 참조 블록은 참조 픽처 인덱스에 대응하는 참조 픽처 내에 존재한다. 또한, 모션 벡터에 의해서 특정된 블록의 샘플 값 또는 이의 보간(interpolation)된 값이 현재 블록의 예측자(predictor)로 이용될 수 있다. 서브펠(sub-pel) 단위의 픽셀 정확도를 갖는 모션 예측을 위하여 이를 테면, 휘도 신호에 대하여 8-탭 보간 필터가, 색차 신호에 대하여 4-탭 보간 필터가 사용될 수 있다. 다만, 서브펠 단위의 모션 예측을 위한 보간 필터는 이에 한정되지 않는다. 이와 같이 인터 예측부(254)는 이전에 복원된 픽처로부터 현재 유닛의 텍스처를 예측하는 모션 보상(motion compensation)을 수행한다. 이때, 인터 예측부는 모션 정보 세트를 이용할 수 있다.

[0048] 추가적인 실시예에 따라, 예측부(250)는 IBC 예측부(미도시)를 포함할 수 있다. IBC 예측부는 현재 픽처 내의 복원된 샘플들을 포함하는 특정 영역을 참조하여 현재 영역을 복원할 수 있다. IBC 예측부는 엔트로피 디코딩부(210)로부터 획득된 IBC 부호화 정보를 이용하여 IBC 예측을 수행할 수 있다. IBC 부호화 정보는 블록 벡터 정보를 포함할 수 있다.

[0049] 상기 인트라 예측부(252) 또는 인터 예측부(254)로부터 출력된 예측값, 및 역변환부(225)로부터 출력된 레지듀얼 값이 더해져서 복원된 비디오 픽처가 생성된다. 즉, 비디오 신호 디코딩 장치(200)는 예측부(250)에서 생성된 예측 블록과 역변환부(225)로부터 획득된 레지듀얼을 이용하여 현재 블록을 복원한다.

[0050] 한편, 도 2의 블록도는 본 발명의 일 실시예에 따른 디코딩 장치(200)를 나타낸 것으로서, 분리하여 표시된 블록들은 디코딩 장치(200)의 엘리먼트들을 논리적으로 구별하여 도시한 것이다. 따라서 전술한 디코딩 장치(200)의 엘리먼트들은 디바이스의 설계에 따라 하나의 칩으로 또는 복수의 칩으로 장착될 수 있다. 일 실시예에 따르면, 전술한 디코딩 장치(200)의 각 엘리먼트의 동작은 프로세서(미도시)에 의해 수행될 수 있다.

[0051] 한편, 본 명세서에서 제안된 기술은 인코더와 디코더의 방법 및 장치에 모두 적용 가능한 기술이며, 시그널링과 파싱으로 기술된 부분은 설명의 편의를 위해 기술한 것일 수 있다. 일반적으로 시그널링은 인코더 관점에서 각 신택스(syntax)를 부호화하기 위한 것이고, 파싱은 디코더 관점에서 각 신택스의 해석을 위한 것으로 설명될 수 있다. 즉, 각 신택스는 인코더로부터 비트스트림에 포함되어 시그널링될 수 있으며, 디코더에서는 신택스를 파싱하여 복원과정에서 사용할 수 있다. 이때, 규정된 계층적 구성대로 나열한 각 신택스에 대한 비트의 시퀀스를 비트스트림이라고 할 수 있다.

[0052] 하나의 픽처는 서브 픽처(sub-picture), 슬라이스(slice), 타일(tile) 등으로 분할되어 부호화될 수 있다. 서브 픽처는 하나 이상의 슬라이스 또는 타일을 포함할 수 있다. 하나의 픽처가 여러 개의 슬라이스 또는 타일로 분할되어 부호화되었을 경우, 픽처 내의 모든 슬라이스 또는 타일이 디코딩이 완료되어야만 화면에 출력이 가능하다. 반면에, 하나의 픽처가 여러 개의 서브 픽처로 부호화되었을 경우, 임의의 서브 픽처만 디코딩되어 화면에 출력될 수 있다. 슬라이스는 여러 개의 타일 또는 서브 픽처를 포함할 수 있다. 또는 타일은 여러 개의 서브 픽처 또는 슬라이스를 포함할 수 있다. 서브 픽처, 슬라이스, 타일은 서로 독립적으로 인코딩 또는 디코딩이 가능하므로 병렬처리 및 처리 속도 향상에 효과적이다. 하지만, 인접한 다른 서브 픽처, 다른 슬라이스, 다른 타일의 부호화된 정보를 이용할 수 없으므로 비트량이 증가되는 단점이 있다. 서브 픽처, 슬라이스, 타일은 여러 개의 코딩 트리 유닛(Coding Tree Unit, CTU)으로 분할되어 부호화될 수 있다.

[0053] 도 3은 픽처 내에서 코딩 트리 유닛(Coding Tree Unit, CTU)이 코딩 유닛들(Coding Units, CUs)로 분할되는 실시예를 도시한다. 비디오 신호의 코딩 과정에서, 픽처는 코딩 트리 유닛(CTU)들의 시퀀스로 분할될 수 있다. 코딩 트리 유닛은 휘도(luma) 코딩 트리 블록(Coding Tree Block, CTB)와 2개의 색차(chroma) 코딩 트리 블록들, 그리고 그것의 부호화된 신택스(syntax) 정보로 구성될 수 있다. 하나의 코딩 트리 유닛은 하나의 코딩 유닛으로 구성될 수 있으며, 또는 하나의 코딩 트리 유닛은 여러 개의 코딩 유닛으로 분할될 수 있다. 하나의 코딩 유닛은 휘도 코딩 블록(Coding Block, CB)과 2개의 색차 코딩 블록들, 그리고 그것의 부호화된 신택스 정보로 구성될 수 있다. 하나의 코딩 블록은 여러 개의 서브 코딩 블록으로 분할될 수 있다. 하나의 코딩 유닛은 하나의 변환 유닛(Transform Unit, TU)으로 구성될 수 있으며, 또는 하나의 코딩 유닛은 여러 개의 변환 유닛으로 분할될 수 있다. 하나의 변환 유닛은 휘도 변환 블록(Transform Block, TB)과 2개의 색차 변환 블록들, 그리고 그것의 부호화된 신택스 정보로 구성될 수 있다. 코딩 트리 유닛은 복수의 코딩 유닛들로 분할될 수 있다. 코딩 트리 유닛은 분할되지 않고 리프 노드가 될 수도 있다. 이 경우, 코딩 트리 유닛 자체가 코딩 유닛이 될 수 있다.

[0054] 코딩 유닛은 상기에서 설명한 비디오 신호의 처리 과정, 즉 인트라/인터 예측, 변환, 양자화 및/또는 엔트로피 코딩 등의 과정에서 픽처를 처리하기 위한 기본 단위를 가리킨다. 하나의 픽처 내에서 코딩 유닛의 크기 및 모양은 일정하지 않을 수 있다. 코딩 유닛은 정사각형 또는 직사각형의 모양을 가질 수 있다. 직사각형 코딩 유닛(또는, 직사각형 블록)은 수직 코딩 유닛(또는, 수직 블록)과 수평 코딩 유닛(또는, 수평 블록)을 포함한다. 본 명세서에서, 수직 블록은 높이가 너비보다 큰 블록이며, 수평 블록은 너비가 높이보다 큰 블록이다. 또한, 본 명세서에서 정사각형이 아닌(non-square) 블록은 직사각형 블록을 가리킬 수 있지만, 본 발명은 이에 한정되지 않는다.

[0055] 도 3을 참조하면, 코딩 트리 유닛은 먼저 쿼드 트리(Quad Tree, QT) 구조로 분할된다. 즉, 쿼드 트리 구조에서 2NX2N 크기를 가지는 하나의 노드는 NXN 크기를 가지는 네 개의 노드들로 분할될 수 있다. 본 명세서에서 쿼드 트리는 4진(quaternary) 트리로도 지칭될 수 있다. 쿼드 트리 분할은 재귀적으로 수행될 수 있으며, 모든 노드

들이 동일한 깊이로 분할될 필요는 없다.

[0056] 한편, 전술한 쿼드 트리의 리프 노드(leaf node)는 멀티-타입 트리(Multi-Type Tree, MTT) 구조로 더욱 분할될 수 있다. 본 발명의 실시예에 따르면, 멀티 타입 트리 구조에서는 하나의 노드가 수평 또는 수직 분할의 2진(binary, 바이너리) 또는 3진(ternary, 터너리) 트리 구조로 분할될 수 있다. 즉, 멀티-타입 트리 구조에는 수직 바이너리 분할, 수평 바이너리 분할, 수직 터너리 분할 및 수평 터너리 분할의 4가지 분할 구조가 존재한다. 본 발명의 실시예에 따르면, 상기 각 트리 구조에서 노드의 너비 및 높이는 모두 2의 거듭제곱 값을 가질 수 있다. 예를 들어, 바이너리 트리(Binary Tree, BT) 구조에서, 2NX2N 크기의 노드는 수직 바이너리 분할에 의해 2개의 NX2N 노드들로 분할되고, 수평 바이너리 분할에 의해 2개의 2NXN 노드들로 분할될 수 있다. 또한, 터너리 트리(Ternary Tree, TT) 구조에서, 2NX2N 크기의 노드는 수직 터너리 분할에 의해 (N/2)X2N, NX2N 및 (N/2)X2N의 노드들로 분할되고, 수평 터너리 분할에 의해 2NX(N/2), 2NXN 및 2NX(N/2)의 노드들로 분할될 수 있다. 이러한 멀티-타입 트리 분할은 재귀적으로 수행될 수 있다.

[0057] 멀티-타입 트리의 리프 노드는 코딩 유닛이 될 수 있다. 코딩 유닛이 최대 변환 길이에 비해 크지 않은 경우, 해당 코딩 유닛은 더 이상의 분할 없이 예측 및/또는 변환의 단위로 사용될 수 있다. 일 실시예로서, 현재 코딩 유닛의 너비 또는 높이가 최대 변환 길이보다 큰 경우, 현재 코딩 유닛은 분할에 관한 명시적 시그널링 없이 복수의 변환 유닛으로 분할될 수 있다. 한편, 전술한 쿼드 트리 및 멀티-타입 트리에서 다음의 파라메터들 중 적어도 하나가 사전에 정의되거나 PPS, SPS, VPS 등과 같은 상위 레벨 세트의 RBSP를 통해 전송될 수 있다. 1) CTU 크기: 쿼드 트리의 루트 노드(root node) 크기, 2) 최소 QT 크기(MinQtSize): 허용된 최소 QT 리프 노드 크기, 3) 최대 BT 크기(MaxBtSize): 허용된 최대 BT 루트 노드 크기, 4) 최대 TT 크기(MaxTtSize): 허용된 최대 TT 루트 노드 크기, 5) 최대 MTT 깊이(MaxMttDepth): QT의 리프 노드로부터의 MTT 분할의 최대 허용 깊이, 6) 최소 BT 크기(MinBtSize): 허용된 최소 BT 리프 노드 크기, 7) 최소 TT 크기(MinTtSize): 허용된 최소 TT 리프 노드 크기.

[0058] 도 4는 쿼드 트리 및 멀티-타입 트리의 분할을 시그널링하는 방법의 일 실시예를 도시한다. 전술한 쿼드 트리 및 멀티-타입 트리의 분할을 시그널링하기 위해 기 설정된 플래그들이 사용될 수 있다. 도 4를 참조하면, 노드의 분할 여부를 지시하는 플래그 'split_cu_flag', 쿼드 트리 노드의 분할 여부를 지시하는 플래그 'split_qt_flag', 멀티-타입 트리 노드의 분할 방향을 지시하는 플래그 'mtt_split_cu_vertical_flag' 또는 멀티-타입 트리 노드의 분할 모양을 지시하는 플래그 'mtt_split_cu_binary_flag' 중 적어도 하나가 사용될 수 있다.

[0059] 본 발명의 실시예에 따르면, 현재 노드의 분할 여부를 지시하는 플래그인 'split_cu_flag'가 먼저 시그널링될 수 있다. 'split_cu_flag'의 값이 0인 경우, 현재 노드가 분할되지 않는 것을 나타내며, 현재 노드는 코딩 유닛이 된다. 현재 노드가 코팅 트리 유닛인 경우, 코딩 트리 유닛은 분할되지 않은 하나의 코딩 유닛을 포함한다. 현재 노드가 쿼드 트리 노드 'QT node'인 경우, 현재 노드는 쿼드 트리의 리프 노드 'QT leaf node'이며 코딩 유닛이 된다. 현재 노드가 멀티-타입 트리 노드 'MTT node'인 경우, 현재 노드는 멀티-타입 트리의 리프 노드 'MTT leaf node'이며 코딩 유닛이 된다.

[0060] 'split_cu_flag'의 값이 1인 경우, 현재 노드는 'split_qt_flag'의 값에 따라 쿼드 트리 또는 멀티-타입 트리의 노드들로 분할될 수 있다. 코딩 트리 유닛은 쿼드 트리의 루트 노드이며, 쿼드 트리 구조로 우선 분할될 수 있다. 쿼드 트리 구조에서는 각각의 노드 'QT node' 별로 'split_qt_flag'가 시그널링된다. 'split_qt_flag'의 값이 1인 경우 해당 노드는 4개의 정사각형 노드들로 분할되며, 'split_qt_flag'의 값이 0인 경우 해당 노드는 쿼드 트리의 리프 노드 'QT leaf node'가 되며, 해당 노드는 멀티-타입 노드들로 분할된다. 본 발명의 실시예에 따르면, 현재 노드의 종류에 따라서 쿼드 트리 분할은 제한될 수 있다. 현재 노드가 코딩 트리 유닛(쿼트 트리의 루트 노드) 또는 쿼트 트리 노드인 경우에 쿼드 트리 분할이 허용될 수 있으며, 현재 노드가 멀티-타입 트리 노드인 경우 쿼트 트리 분할은 허용되지 않을 수 있다. 각각의 쿼드 트리 리프 노드 'QT leaf node'는 멀티-타입 트리 구조로 더 분할될 수 있다. 상술한 바와 같이, 'split_qt_flag'가 0인 경우 현재 노드는 멀티-타입 노드들로 분할될 수 있다. 분할 방향 및 분할 모양을 지시하기 위하여, 'mtt_split_cu_vertical_flag' 및 'mtt_split_cu_binary_flag'가 시그널링될 수 있다. 'mtt_split_cu_vertical_flag'의 값이 1인 경우 노드 'MTT node'의 수직 분할이 지시되며, 'mtt_split_cu_vertical_flag'의 값이 0인 경우 노드 'MTT node'의 수평 분할이 지시된다. 또한, 'mtt_split_cu_binary_flag'의 값이 1인 경우 노드 'MTT node'는 2개의 직사각형 노드들로 분할되며, 'mtt_split_cu_binary_flag'의 값이 0인 경우 노드 'MTT node'는 3개의 직사각형 노드들로 분할된다.

[0061] 트리 분할 구조는 휘도 블록과 색차 블록이 동일한 형태로 분할될 수 있다. 즉, 색차 블록은 휘도 블록의 분할 형태를 참조하여 색차 블록을 분할할 수 있다. 현재 색차 블록이 임의의 정해진 크기보다 적다면, 휘도 블록이 분할되었더라도 색차 블록은 분할되지 않을 수 있다.

[0062] 트리 분할 구조는 휘도 블록과 색차 블록이 서로 다른 형태를 가질 수 있다. 이때, 휘도 블록에 대한 분할 정보와 색차 블록에 대한 분할 정보가 각각 시그널링될 수 있다. 또한, 분할 정보 뿐만 아니라 휘도 블록과 색차 블록의 부호화 정보도 다를 수 있다. 실시 일 예로, 휘도 블록과 색차 블록의 인트라 부호화 모드, 움직임 정보에 대한 부호화 정보 등이 적어도 하나 이상 다를 수 있다.

[0063] 가장 작은 단위로 분할될 노드는 하나의 코딩 블록으로 처리될 수 있다. 현재 블록이 코딩 블록일 경우, 코딩 블록은 여러 개의 서브 블록(서브 코딩 블록)으로 분할될 수 있으며, 각 서브 블록의 예측 정보는 서로 같거나 또는 다를 수 있다. 실시 일 예로, 코딩 유닛이 인트라 모드일 경우, 각 서브 블록의 인트라 예측 모드는 서로 같거나 또는 다를 수 있다. 또한, 코딩 유닛이 인터 모드일 경우, 각 서브 블록의 움직임 정보는 서로 같거나 또는 다를 수 있다. 또한, 각 서브 블록은 서로 독립적으로 인코딩 또는 디코딩이 가능할 수 있다. 각각의 서브 블록은 서브 블록 인덱스(sub-block index, sbIdx)를 통해 구분될 수 있다. 또한 코딩 유닛이 서브 블록으로 분할될 때, 수평 또는 수직 방향으로 분할되거나 사선으로 분할될 수 있다. 인트라 모드에서 현재 코딩 유닛을 수평 또는 수직 방향으로 2개 또는 4개의 서브 블록으로 분할하는 모드를 ISP(Intra Sub Partitions)이라 한다. 인터 모드에서 현재 코딩 블록을 사선으로 분할하는 모드를 GPM(Geometric partitioning mode)이라 한다. GPM 모드에서 사선의 위치와 방향은 미리 정해진 각도 테이블을 사용하여 유도하고, 각도 테이블의 인덱스 정보가 시그널링된다.

[0064] 코딩을 위한 픽쳐 예측(모션 보상)은 더 이상 나누어지지 않는 코딩 유닛(즉 코딩 트리 유닛의 리프 노드)을 대상으로 이루어진다. 이러한 예측을 수행하는 기본 단위를 이하에서는 예측 유닛(prediction unit) 또는 예측 블록(prediction block)이라고 한다.

[0065] 이하, 본 명세서에서 사용되는 유닛이라는 용어는 예측을 수행하는 기본 단위인 상기 예측 유닛을 대체하는 용어로 사용될 수 있다. 다만, 본 발명이 이에 한정되는 것은 아니며, 더욱 광의적으로는 상기 코딩 유닛을 포함하는 개념으로 이해될 수 있다.

[0066] 도 5 및 도 6은 본 발명의 실시예에 따른 인트라 예측 방법을 더욱 구체적으로 도시한다. 전술한 바와 같이, 인트라 예측부는 현재 블록의 좌측 및/또는 상측에 위치한 복원된 샘플들을 참조 샘플들로 이용하여 현재 블록의 샘플 값들을 예측한다.

[0067] 먼저, 도 5는 인트라 예측 모드에서 현재 블록의 예측을 위해 사용되는 참조 샘플들의 일 실시예를 도시한다. 일 실시예에 따르면, 참조 샘플들은 현재 블록의 좌측 경계에 인접한 샘플들 및/또는 상측 경계에 인접한 샘플들일 수 있다. 도 5에 도시된 바와 같이, 현재 블록의 크기가 WXH이고 현재 블록에 인접한 단일 참조 라인(line)의 샘플들이 인트라 예측에 사용될 경우, 현재 블록의 좌측 및/또는 상측에 위치한 최대 2W+2H+1개의 주변 샘플들을 사용하여 참조 샘플들이 설정될 수 있다.

[0068] 한편, 현재 블록의 인트라 예측을 위해 다중 참조 라인의 픽셀들이 사용될 수 있다. 다중 참조 라인은 현재 블록으로부터 기 설정된 범위 이내에 위치한 n개의 라인들로 구성될 수 있다. 일 실시예에 따르면, 인트라 예측을 위해 다중 참조 라인의 픽셀들이 사용될 경우, 참조 픽셀들로 설정될 라인들을 지시하는 별도의 인덱스 정보가 시그널링될 수 있으며, 이를 참조 라인 인덱스라고 명명할 수 있다.

[0069] 또한, 참조 샘플로 사용될 적어도 일부의 샘플이 아직 복원되지 않은 경우, 인트라 예측부는 참조 샘플 패딩 과정을 수행하여 참조 샘플을 획득할 수 있다. 또한, 인트라 예측부는 인트라 예측의 오차를 줄이기 위해 참조 샘플 필터링 과정을 수행할 수 있다. 즉, 주변 샘플들 및/또는 참조 샘플 패딩 과정에 의해 획득된 참조 샘플들에 필터링을 수행하여 필터링된 참조 샘플들을 획득할 수 있다. 인트라 예측부는 이와 같이 획득된 참조 샘플들을 이용하여 현재 블록의 샘플들을 예측한다. 인트라 예측부는 필터링되지 않은 참조 샘플들 또는 필터링된 참조 샘플들을 이용하여 현재 블록의 샘플들을 예측한다. 본 개시에서, 주변 샘플들은 적어도 하나의 참조 라인 상의 샘플들을 포함할 수 있다. 예를 들어, 주변 샘플들은 현재 블록의 경계에 인접한 라인 상의 인접 샘플들을 포함할 수 있다.

[0070] 다음으로, 도 6은 인트라 예측에 사용되는 예측 모드들의 일 실시예를 도시한다. 인트라 예측을 위해, 인트라 예측 방향을 지시하는 인트라 예측 모드 정보가 시그널링될 수 있다. 인트라 예측 모드 정보는 인트라 예측 모드 세트를 구성하는 복수의 인트라 예측 모드들 중 어느 하나를 지시한다. 현재 블록이 인트라 예측 블록일 경

우, 디코더는 비트스트림으로부터 현재 블록의 인트라 예측 모드 정보를 수신한다. 디코더의 인트라 예측부는 추출된 인트라 예측 모드 정보에 기초하여 현재 블록에 대한 인트라 예측을 수행한다.

[0071] 본 발명의 실시예에 따르면, 인트라 예측 모드 세트는 인트라 예측에 사용되는 모든 인트라 예측 모드들(예, 총 67개의 인트라 예측 모드들)을 포함할 수 있다. 더욱 구체적으로, 인트라 예측 모드 세트는 평면 모드, DC 모드 및 복수의(예, 65개의) 각도 모드들(즉, 방향 모드들)을 포함할 수 있다. 각각의 인트라 예측 모드는 기 설정된 인덱스(즉, 인트라 예측 모드 인덱스)를 통해 지시될 수 있다. 예를 들어, 도 6에 도시된 바와 같이 인트라 예측 모드 인덱스 0은 평면(planar) 모드를 지시하고, 인트라 예측 모드 인덱스 1은 DC 모드를 지시한다. 또한, 인트라 예측 모드 인덱스 2 내지 66은 서로 다른 각도 모드들을 각각 지시할 수 있다. 각도 모드들은 기 설정된 각도 범위 이내의 서로 다른 각도들을 각각 지시한다. 예를 들어, 각도 모드는 시계 방향으로 45도에서 -135도 사이의 각도 범위(즉, 제1 각도 범위) 이내의 각도를 지시할 수 있다. 상기 각도 모드는 12시 방향을 기준으로 정의될 수 있다. 이때, 인트라 예측 모드 인덱스 2는 수평 대각(Horizontal Diagonal, HDIA) 모드를 지시하고, 인트라 예측 모드 인덱스 18은 수평(Horizontal, HOR) 모드를 지시하고, 인트라 예측 모드 인덱스 34는 대각(Diagonal, DIA) 모드를 지시하고, 인트라 예측 모드 인덱스 50은 수직(Vertical, VER) 모드를 지시하며, 인트라 예측 모드 인덱스 66은 수직 대각(Vertical Diagonal, VDIA) 모드를 지시한다.

[0072] 한편, 기 설정된 각도 범위는 현재 블록의 모양에 따라 서로 다르게 설정될 수 있다. 예를 들어, 현재 블록이 직사각형 블록일 경우 시계 방향으로 45도를 초과하거나 -135도 미만 각도를 지시하는 광각 모드가 추가적으로 사용될 수 있다. 현재 블록이 수평 블록일 경우, 각도 모드는 시계 방향으로 (45+offset1)도에서 (-135+offset1)도 사이의 각도 범위(즉, 제2 각도 범위) 이내의 각도를 지시할 수 있다. 이때, 제1 각도 범위를 벗어나는 각도 모드 67 내지 76이 추가적으로 사용될 수 있다. 또한, 현재 블록이 수직 블록일 경우, 각도 모드는 시계 방향으로 (45-offset2)도에서 (-135-offset2)도 사이의 각도 범위(즉, 제3 각도 범위) 이내의 각도를 지시할 수 있다. 이때, 제1 각도 범위를 벗어나는 각도 모드 -10 내지 -1이 추가적으로 사용될 수 있다. 본 발명의 실시예에 따르면, offset1 및 offset2의 값은 직사각형 블록의 너비와 높이 간의 비율에 따라 서로 다르게 결정될 수 있다. 또한, offset1 및 offset2는 양수일 수 있다.

[0073] 본 발명의 추가적인 실시예에 따르면, 인트라 예측 모드 세트를 구성하는 복수의 각도 모드들은 기본 각도 모드와 확장 각도 모드를 포함할 수 있다. 이때, 확장 각도 모드는 기본 각도 모드에 기초하여 결정될 수 있다.

[0074] 일 실시예에 따르면, 기본 각도 모드는 기존 HEVC(High Efficiency Video Coding) 표준의 인트라 예측에서 사용되는 각도에 대응하는 모드이고, 확장 각도 모드는 차세대 비디오 코덱 표준의 인트라 예측에서 새롭게 추가되는 각도에 대응하는 모드일 수 있다. 더욱 구체적으로, 기본 각도 모드는 인트라 예측 모드 {2, 4, 6, …, 66} 중 어느 하나에 대응하는 각도 모드이고, 확장 각도 모드는 인트라 예측 모드 {3, 5, 7, …, 65} 중 어느 하나에 대응하는 각도 모드일 수 있다. 즉, 확장 각도 모드는 제1 각도 범위 내에서 기본 각도 모드들 사이의 각도 모드일 수 있다. 따라서, 확장 각도 모드가 지시하는 각도는 기본 각도 모드가 지시하는 각도에 기초하여 결정될 수 있다.

[0075] 다른 실시예에 따르면, 기본 각도 모드는 기 설정된 제1 각도 범위 이내의 각도에 대응하는 모드이고, 확장 각도 모드는 상기 제1 각도 범위를 벗어나는 광각 모드일 수 있다. 즉, 기본 각도 모드는 인트라 예측 모드 {2, 3, 4, …, 66} 중 어느 하나에 대응하는 각도 모드이고, 확장 각도 모드는 인트라 예측 모드 {-14, -13, -12, …, -1} 및 {67, 68, …, 80} 중 어느 하나에 대응하는 각도 모드일 수 있다. 확장 각도 모드가 지시하는 각도는 대응하는 기본 각도 모드가 지시하는 각도의 반대편 각도로 결정될 수 있다. 따라서, 확장 각도 모드가 지시하는 각도는 기본 각도 모드가 지시하는 각도에 기초하여 결정될 수 있다. 한편, 확장 각도 모드들의 개수는 이에 한정되지 않으며, 현재 블록의 크기 및/또는 모양에 따라 추가적인 확장 각도들이 정의될 수 있다. 한편, 인트라 예측 모드 세트에 포함되는 인트라 예측 모드들의 총 개수는 전술한 기본 각도 모드와 확장 각도 모드의 구성에 따라 가변할 수 있다.

[0076] 상기 실시예에서, 확장 각도 모드들 간의 간격은 대응하는 기본 각도 모드들 간의 간격에 기초하여 설정될 수 있다. 예를 들어, 확장 각도 모드들 {3, 5, 7, …, 65} 간의 간격은 대응하는 기본 각도 모드들 {2, 4, 6, …, 66} 간의 간격에 기초하여 결정될 수 있다. 또한, 확장 각도 모드들 {-14, -13, …, -1} 간의 간격은 대응하는 반대편의 기본 각도 모드들 {53, 53, …, 66} 간의 간격에 기초하여 결정되고, 확장 각도 모드들 {67, 68, …, 80} 간의 간격은 대응하는 반대편의 기본 각도 모드들 {2, 3, 4, …, 15} 간의 간격에 기초하여 결정될 수 있다. 확장 각도 모드들 간의 각도 간격은 대응하는 기본 각도 모드들 간의 각도 간격과 동일하도록 설정될 수 있다. 또한, 인트라 예측 모드 세트에서 확장 각도 모드들의 개수는 기본 각도 모드들의 개수 이하로 설정될 수

있다.

[0077] 본 발명의 실시예에 따르면, 확장 각도 모드는 기본 각도 모드를 기초로 시그널링될 수 있다. 예를 들어, 광각 모드(즉, 확장 각도 모드)는 제1 각도 범위 이내의 적어도 하나의 각도 모드(즉, 기본 각도 모드)를 대체할 수 있다. 대체되는 기본 각도 모드는 광각 모드의 반대편에 대응하는 각도 모드일 수 있다. 즉, 대체되는 기본 각도 모드는 광각 모드가 지시하는 각도의 반대 방향의 각도에 대응하거나 또는 상기 반대 방향의 각도로부터 기 설정된 오프셋 인덱스만큼 차이 나는 각도에 대응하는 각도 모드이다. 본 발명의 실시예에 따르면, 기 설정된 오프셋 인덱스는 1이다. 대체되는 기본 각도 모드에 대응하는 인트라 예측 모드 인덱스는 광각 모드에 다시 매핑되어 해당 광각 모드를 시그널링할 수 있다. 예를 들어, 광각 모드 {-14, -13, …, -1}은 인트라 예측 모드 인덱스 {52, 53, …, 66}에 의해 각각 시그널링될 수 있고, 광각 모드 {67, 68, …, 80}은 인트라 예측 모드 인덱스 {2, 3, …, 15}에 의해 각각 시그널링될 수 있다. 이와 같이 기본 각도 모드를 위한 인트라 예측 모드 인덱스가 확장 각도 모드를 시그널링하도록 함으로, 각 블록의 인트라 예측에 사용되는 각도 모드들의 구성이 서로 다르더라도 동일한 세트의 인트라 예측 모드 인덱스들이 인트라 예측 모드의 시그널링에 사용될 수 있다. 따라서, 인트라 예측 모드 구성의 변화에 따른 시그널링 오버헤드가 최소화될 수 있다.

[0078] 한편, 확장 각도 모드의 사용 여부는 현재 블록의 모양 및 크기 중 적어도 하나에 기초하여 결정될 수 있다. 일 실시예에 따르면, 현재 블록의 크기가 기 설정된 크기보다 클 경우 확장 각도 모드가 현재 블록의 인트라 예측을 위해 사용되고, 그렇지 않을 경우 기본 각도 모드만 현재 블록의 인트라 예측을 위해 사용될 수 있다. 다른 실시예에 따르면, 현재 블록이 정사각형이 아닌 블록인 경우 확장 각도 모드가 현재 블록의 인트라 예측을 위해 사용되고, 현재 블록이 정사각형 블록인 경우 기본 각도 모드만 현재 블록의 인트라 예측을 위해 사용될 수 있다.

[0079] 인트라 예측부는 현재 블록의 인트라 예측 모드 정보에 기초하여, 현재 블록의 인트라 예측에 사용될 참조 샘플들 및/또는 보간된 참조 샘플들을 결정한다. 인트라 예측 모드 인덱스가 특정 각도 모드를 지시할 경우, 현재 블록의 현재 샘플로부터 상기 특정 각도에 대응하는 참조 샘플 또는 보간된 참조 샘플이 현재 픽셀의 예측에 사용된다. 따라서, 인트라 예측 모드에 따라 서로 다른 세트의 참조 샘플들 및/또는 보간된 참조 샘플들이 인트라 예측에 사용될 수 있다. 참조 샘플들 및 인트라 예측 모드 정보를 이용하여 현재 블록의 인트라 예측이 수행되고 나면, 디코더는 역변환부로부터 획득된 현재 블록의 잔차 신호를 현재 블록의 인트라 예측 값과 더하여 현재 블록의 샘플 값들을 복원한다.

[0080] 인터 예측에 사용되는 움직임(모션) 정보에는 참조 방향 지시 정보(inter_pred_idc), 참조 픽처 인덱스(ref_idx_l0, ref_idx_l1), 움직임(모션) 벡터(mvL0, mvL1)이 포함될 수 있다. 참조 방향 지시 정보에 따라 참조 픽처 리스트 활용 정보(predFlagL0, predFlagL1)가 설정될 수 있다. 실시 일 예로, L0 참조 픽처를 사용하는 단방향 예측인 경우, predFlagL0=1, predFlagL1=0로 설정될 수 있다. L1 참조 픽처를 사용하는 단방향 예측인 경우, predFlagL0=0, predFlagL1=1로 설정될 수 있다. L0와 L1 참조 픽처를 모두 사용하는 양방향 예측인 경우, predFlagL0=1, predFlagL1=1로 설정될 수 있다.

[0081] 현재 블록이 코딩 유닛일 경우, 코딩 유닛은 여러 개의 서브 블록으로 분할될 수 있으며, 각 서브 블록의 예측 정보는 서로 같거나 또는 다를 수 있다. 실시 일 예로, 코딩 유닛이 인트라 모드일 경우, 각 서브 블록의 인트라 예측 모드는 서로 같거나 또는 다를 수 있다. 또한, 코딩 유닛이 인터 모드일 경우, 각 서브 블록의 움직임 정보는 서로 같거나 또는 다를 수 있다. 또한, 각 서브 블록은 서로 독립적으로 인코딩 또는 디코딩이 가능할 수 있다. 각각의 서브 블록은 서브 블록 인덱스(sub-block index, sbIdx)를 통해 구분될 수 있다.

[0082] 현재 블록의 움직임 벡터는 주변 블록의 움직임 벡터와 유사할 가능성이 높다. 따라서, 주변 블록의 움직임 벡터는 움직임 예측 값(motion vector predictor, mvp)으로 사용될 수 있고, 현재 블록의 움직임 벡터는 주변 블록의 움직임 벡터를 이용하여 유도될 수 있다. 또한, 움직임 벡터의 정확성을 높이기 위해서, 인코더에서 원본 영상으로 찾은 현재 블록의 최적의 움직임 벡터와 움직임 예측 값 간의 움직임 벡터의 차이(motion vector difference, mvd)가 시그널링될 수 있다.

[0083] 움직임 벡터는 다양한 해상도를 가질 수 있으며, 블록 단위로 움직임 벡터의 해상도가 달라질 수 있다. 움직임 벡터 해상도는 정수 단위, 반화소 단위, 1/4 화소 단위, 1/16 화소 단위, 4의 정수 화소 단위 등으로 표현될 수 있다. 스크린 콘텐츠와 같은 영상은 문자와 같은 단순한 그래픽 형태이므로 보간(interpolation) 필터를 적용하지 않아도 되므로, 정수 단위와 4의 정수 화소 단위가 블록 단위 선택적으로 적용될 수 있다. 회전 및 스케일을 표현할 수 있는 어파인(Affine) 모드로 부호화된 블록은 형태의 변화가 심하므로, 정수 단위, 1/4 화소 단위, 1/16 화소 단위가 블록 기반 선택적으로 적용될 수 있다. 블록 단위로 움직임 벡터 해상도를 선택적으로 적용할

지에 대한 여부 정보는 amvr_flag으로 시그널링된다. 만일 적용되는 경우, 어떠한 움직임 벡터 해상도를 현재 블록에 적용할지는 amvr_precision_idx으로 시그널링된다.

[0084] 양방향 예측이 적용되는 블록의 경우, 가중치 평균을 적용할 때 2개의 예측 블록 간의 가중치를 같거나 또는 다르게 적용할 수 있으며, 가중치에 대한 정보는 bcw_idx를 통해 시그널링된다.

[0085] 움직임 예측 값의 정확도를 높이기 위해서, 머지(Merge) 또는 AMVP(dvanced motion vector prediction) 방법이 블록 단위 선택적으로 사용될 수 있다. Merge 방법은 현재 블록의 움직임 정보를 현재 블록에 인접한 주변 블록의 움직임 정보와 동일하게 구성하는 방법으로, 동질성을 갖는 움직임 영역에서 움직임 정보가 변화없이 공간적으로 전파됨으로써 움직임 정보의 부호화 효율을 증가시키는 장점이 있다. 반면에 AMVP 방법은 정확한 움직임 정보를 표현하기 위해 L0 및 L1 예측 방향으로 각각 움직임 정보를 예측하고 가장 최적의 움직임 정보를 시그널링하는 방법이다. 디코더는 AMVP 또는 Merge 방법을 통해 현재 블록에 대한 움직임 정보를 유도한 후, 참조 픽처(reference picture)에서 유도한 움직임 정보에 위치한 참조 블록을 현재 블록을 위한 예측 블록으로 사용한다.

[0086] Merge 또는 AMVP에서 움직임 정보를 유도하는 방법은 현재 블록의 주변 블록으로부터 유도된 움직임 예측 값을 사용하여 움직임 후보 리스트를 구성한 후, 최적의 움직임 후보에 대한 인덱스 정보가 시그널링되는 방법일 수 있다. AMVP의 경우, L0와 L1 각각 움직임 후보 리스트가 유도되므로, L0와 L1 각각에 대한 최적의 움직임 후보 인덱스(mvp_l0_flag, mvp_l1_flag)가 시그널링된다. Merge의 경우, 하나의 움직임 후보 리스트가 유도되므로, 하나의 머지 인덱스(merge_idx)가 시그널링된다. 하나의 코딩 유닛에서 유도되는 움직임 후보 리스트는 다양할 수 있으며, 각 움직임 후보 리스트마다 움직임 후보 인덱스 또는 머지 인덱스가 시그널링될 수 있다. 이때, Merge 모드로 부호화된 블록에서 잔여 블록에 대한 정보가 없는 모드를 머지 스킵(MergeSkip) 모드라고 할 수 있다.

[0087] SMVD(Symmetric MVD)는 양방향 예측(bi-directional prediction)의 경우에, L0 방향과 L1 방향의 MVD(Motion Vector Difference) 값이 대칭을 이루도록 하여 전송되는 움직임 정보의 비트량을 줄이는 방법이다. L0 방향과 대칭을 이루는 L1 방향의 MVD 정보는 전송하지 않으며, 더불어 L0 및 L1 방향의 참조 픽처 정보도 전송하지 않고 복호화 과정에서 유도한다.

[0088] OBMC(Overlapped Block Motion Compensation)는 블록 간의 움직임 정보가 서로 다른 경우, 주변 블록들의 움직임 정보를 사용하여 현재 블록에 대한 예측 블록들을 생성한 후, 예측 블록들을 가중치 평균하여 현재 블록에 대한 최종 예측 블록을 생성하는 방법이다. 이는 움직임 보상된 영상의 블록 경계에서 발생되는 블록킹 현상을 줄여주는 효과가 있다.

[0089] 일반적으로 머지 움직임 후보는 움직임의 정확도가 낮다. 이러한 머지 움직임 후보의 정확도를 높이기 위해서, MMVD(Merge mode with MVD) 방법이 사용될 수 있다. MMVD 방법은 몇 개의 움직임 차분 값 후보들 중에서 선택된 하나의 후보를 이용하여 움직임 정보를 보정하는 방법이다. MMVD 방법을 통해 획득되는 움직임 정보의 보정 값에 대한 정보(예를 들어, 움직임 차분 값 후보들 중에서 선택된 하나의 후보를 지시하는 인덱스 등)은 비트스트림에 포함되어 디코더로 전송될 수 있다. 기존의 움직임 정보 차분 값을 비트스트림에 포함하는 것에 비해 움직임 정보의 보정 값에 대한 정보를 비트스트림에 포함함으로써 비트량을 절약할 수 있다.

[0090] TM(Template Matching) 방법은 현재 블록의 주변 화소를 통해 템플릿을 구성하여 템플릿과 가장 유사도가 높은 매칭 영역을 찾아서 움직임 정보를 보정하는 방법이다. TM(Template matching)은 부호화되는 비트스트림의 크기를 줄이기 위해서, 움직임 정보를 비트스트림에 포함하지 않고 디코더에서 움직임 예측을 수행하는 방법이다. 이때, 디코더는 원본 영상이 없으므로 이미 복원된 주변 블록을 사용하여 현재 블록에 대한 움직임 정보를 개략적으로 유도할 수 있다.

[0091] DMVR(Decoder-side Motion Vector Refinement) 방법은 조금 더 정확한 움직임 정보를 찾기 위해 이미 복원된 참조 영상들의 상관성을 통해 움직임 정보를 보정하는 방법으로써, 현재 블록의 양방향 움직임 정보를 사용하여 2개의 참조 픽처의 임의의 정해진 영역 내에서 참조 픽처 내의 참조 블록 간의 가장 매칭이 잘되는 지점을 새로운 양방향 움직임으로 사용하는 방법이다. 이러한 DMVR이 수행될 때, 인코더는 하나의 블록 단위에서 DMVR을 수행하여 움직임 정보를 보정한 후, 다시 블록을 서브 블록으로 분할하여 각 서브 블록 단위에서 DMVR을 수행하여 서브 블록의 움직임 정보를 다시 보정할 수 있으며, 이를 MP-DMVR(Multi-pass DMVR)이라 할 수 있다.

[0092] LIC(Local Illumination Compensation) 방법은 블록 간의 휘도 변화를 보상하는 방법으로, 현재 블록에 인접한 주변 화소들을 사용하여 선형 모델을 유도한 후, 선형 모델을 통해 현재 블록의 휘도 정보를 보상하는

방법이다.

[0093] 기존 비디오 부호화 방법들은 상하좌우의 평행 이동만을 고려한 움직임 보상을 수행하기 때문에, 현실에서 일반적으로 접하는 확대, 축소, 회전 등과 같은 움직임 포함하고 있는 비디오들의 부호화 시 부호화 효율이 저하된다. 이러한 확대, 축소, 회전에 대한 움직임을 표현하기 위하여, 4개(회전) 또는 6개(확대, 축소, 회전) 파라미터 모델을 이용하는 Affine 모델 기반 움직임 예측 기술이 적용될 수 있다.

[0094] BDOF(Bi-Directional Optical Flow)는 양방향 움직임으로 구성된 블록의 참조 블록으로부터 광-흐름(optical-flow) 기반으로 화소의 변화량을 추정하여 예측 블록을 보정하는데 사용된다. 이러한 VVC의 BDOF에서 유도된 움직임 정보를 이용하여 현재 블록의 움직임을 보정할 수 있다.

[0095] PROF(Prediction refinement with optical flow)는 서브 블록 단위 Affine 움직임 예측의 정확도를 픽셀 단위 움직임 예측의 정확도와 유사하도록 개선하기 위한 기술이다. PROF는 BDOF와 유사하게 광-흐름(optical-flow)에 기반하여 서브 블록 단위로 Affine 움직임 보상된 픽셀 값들에 대해 픽셀 단위로 보정 값을 계산하여 최종 예측 신호를 획득하는 기술이다.

[0096] CIIP(Combined Inter-/Intra-picture Prediction)방법은 현재 블록에 대한 예측 블록을 생성할 때, 화면 내 예측 방법으로 생성한 예측 블록과 화면 간 예측 방법으로 생성한 예측 블록들을 가중치 평균하여 최종 예측 블록을 생성하는 방법이다.

[0097] IBC(Intra Block Copy) 방법은 현재 블록과 가장 유사한 부분을 현재 픽처 내의 이미 복원된 영역에서 찾아서, 해당 참조 블록을 현재 블록에 대한 예측 블록으로 사용하는 방법이다. 이때, 현재 블록과 참조 블록 간의 거리인 블록 벡터(Block Vector)와 관련된 정보는 비트스트림에 포함될 수 있다. 디코더는 비스트스림에 포함된 블록 벡터와 관련된 정보를 파싱하여 현재 블록을 위한 블록 벡터를 계산하거나 설정할 수 있다.

[0098] BCW(Bi-prediction with CU-level Weights) 방법은 서로 다른 참조 픽쳐로 부터 움직임 보상된 두 개의 예측 블록에 대하여, 평균으로 예측 블록을 생성하지 않고, 블록 단위로 적응적으로 가중치를 적용하여 움직임 보상된 두 개의 예측 블록에 가중치 평균을 수행하는 방법이다.

[0099] MHP(Multi-hypothesis prediction) 방법은 화면 간 예측 시 단방향 및 양방향 움직임 정보에 추가적인 움직임 정보를 전송함으로써, 다양한 예측 신호를 통한 가중치 예측을 수행하는 방법이다.

[0100] CCLM(Cross-component linear model)은 휘도 신호와 해당 휘도 신호와 동일한 위치에 있는 색차 신호 간의 높은 상관성을 이용하여 선형 모델을 구성한 후, 해당 선형 모델을 통해 색차 신호를 예측하는 방법이다. 현재 블록에 인접한 주변 블록 중에서 복원이 완료된 블록을 사용하여 템플릿을 구성한 후, 템플릿을 통해 선형 모델에 대한 파라미터가 유도된다. 다음으로, 영상 포맷에 따라 선택적으로 색차 블록의 크기에 맞게 복원된 현재 휘도 블록이 다운 샘플링된다. 마지막으로, 다운 샘플링된 휘도 블록과 해당 선형 모델을 이용하여 현재 블록의 색차 블록을 예측한다. 이때, 2개 이상의 선형 모델을 사용하는 방법을 MMLM(Multi-model Linear mode)이라고 한다.

[0101] 독립 스칼라 양자화(independent scalar quantization)에서, 입력된 계수 t_k에 대한 복원된 계수 t'_k는 관련된 양자화 인덱스(quantization index) q_k에만 의존적이다. 즉, 임의의 복원된 계수에 대한 양자화 인덱스(quantization index)는 다른 복원된 계수들에 대한 양자화 인덱스들과는 다른 값을 가진다. 이때 t'_k는 t_k에서 양자화 오차가 포함된 값일 수 있으며, 양자화 파라미터에 따라 서로 다르거나 또는 같을 수 있다. 여기서, t'_k는 복원된 변환 계수 또는 역양자화된 변환계수라고 명명할 수 있으며, 양자화 인덱스를 양자화된 변환 계수라고 명명할 수도 있다.

[0102] 균일 복원 양자화(URQ; Uniform Reconstruction Quantizers)에서, 복원된 계수들은 동일한 간격으로 배치되는 특성을 지닌다. 이때 인접하는 두 복원 값들 사이의 거리를 양자화 단계 크기(quantization step size)라고 할 수 있다. 복원된 값 중에는 0을 포함할 수 있으며, 사용 가능한 복원 값들의 전체 집합(set)은 양자화 단계 크기에 따라 고유하게 정의될 수 있다. 양자화 단계 크기는 양자화 파라미터에 따라 달라질 수 있다.

[0103] 기존 방법에서는 양자화로 인해 허용 가능한 복원된 변환 계수들의 집합(세트)이 감소하며, 이러한 집합의 원소(element)는 유한 개일 수 있다. 이로 인해, 원본 영상과 복원된 영상 간의 평균적인 오차를 최소화하는데 한계가 존재한다. 이러한 평균적인 오차를 최소화하기 위한 방법으로 벡터 양자화(Vector Quantization)를 사용할 수 있다.

[0104] 비디오 부호화에서 사용되는 간단한 형태의 벡터 양자화 방법에는 부호 데이터 은닉(sign data hiding)이 있다.

이는 인코더에서 0이 아닌 하나의 계수에 대한 부호를 부호화하지 않고, 디코더에서는 해당 계수에 대한 부호를 모든 계수들에 대한 절대값의 합이 짝수인지 또는 홀수인지에 따라 결정하는 방법이다. 이를 위해 인코더에서는 적어도 하나의 계수가 '1'이 증가되거나 또는 감소될 수 있으며, 이는 율-왜곡(rate-distortion)에 대한 비용(Cost) 관점에서 최적이 되도록 적어도 하나의 계수가 선택되어 값이 조정될 수 있다. 실시 일 예로, 양자화 간격의 경계에 가까운 값을 가지는 계수가 선택될 수 있다.

[0105] 또 다른 벡터 양자화 방법에는 트렐리스 부호화된 양자화(Trellis-Coded Quantization)가 있으며, 비디오 부호화에서는 종속 양자화(dependent quantization)에서 최적화된 양자화 값을 얻기 위한 최적의 경로 탐색 기법으로 활용된다. 블록 단위로, 블록 내 모든 계수들에 대한 양자화 후보들을 트렐리스 그래프에 배치하고, 최적화된 양자화 후보들 간의 최적의 트렐리스 경로를 율-왜곡(rate-distortion)에 대한 비용(Cost)을 고려하여 탐색한다. 구체적으로, 비디오 부호화에 적용된 종속 양자화는 변환 계수에 대한 허용 가능한 복원된 변환 계수들의 집합이 복원 순서에서 현재 변환 계수에 선행하는 변환 계수의 값에 의존하도록 설계될 수 있다. 이때, 여러 개의 양자화기를 변환 계수에 따라 선택적으로 사용하게 함으로써, 원본 영상과 복원된 영상 간의 평균적인 오차를 최소화하여 부호화 효율을 높이는 효과가 있다.

[0106] 인트라 예측 부호화 기술 중에서 MIP(Matrix Intra Prediction) 방법은 행렬 기반 인트라 예측 방법으로 현재 블록에 인접한 주변 블록의 픽셀로부터 방향성을 가지는 예측 방법과 달리, 주변 블록의 좌측 및 상단의 픽셀들을 미리 정의된 행렬 매트릭스와 오프셋 값을 이용하여 예측 신호를 구하는 방법이다.

[0107] 현재 블록의 인트라 예측 모드를 유도하기 위해서, 현재 블록에 인접하면서 복원된 임의의 영역인 템플릿(template)을 기반으로, 해당 템플릿의 주변 픽셀을 통해 유도된 템플릿에 대한 인트라 예측 모드가 현재 블록의 복원을 위해 이용할 수 있다. 우선, 디코더는 템플릿에 인접한 주변 픽셀(reference)을 이용하여 템플릿에 대한 예측 템플릿을 생성하고, 이미 복원된 템플릿과 가장 유사한 예측 템플릿을 생성한 인트라 예측 모드를 현재 블록의 복원을 위해 사용할 수 있다. 이러한 방법을 TIMD(Template intra mode derivation)이라고 할 수 있다.

[0108] 일반적으로 인코더는 예측 블록을 생성하기 위한 예측 모드를 결정하여 결정된 예측 모드에 대한 정보가 포함된 비트스트림을 생성할 수 있다. 디코더는 수신한 비트스트림을 파싱하여 인트라 예측 모드를 설정할 수 있다. 이때, 예측 모드에 대한 정보의 비트량은 전체 비트스트림 크기의 10% 정도일 수 있다. 예측 모드에 대한 정보의 비트량을 감소시키기 위해 인코더는 비트스트림에 인트라 예측 모드에 대한 정보를 포함하지 않을 수 있다. 이에, 디코더는 주변 블록의 특성을 이용하여 현재 블록의 복원을 위한 인트라 예측 모드를 유도(결정)할 수 있고, 유도된 인트라 예측 모드를 이용하여 현재 블록을 복원할 수 있다. 이때, 디코더는 인트라 예측 모드를 유도하기 위해 현재 블록에 인접한 주변 화소(픽셀)들마다 소벨(Sobel) 필터를 가로 및 세로 방향으로 적용하여 방향성 정보를 유추한 후, 해당 방향성 정보를 인트라 예측 모드로 매핑하는 방법을 이용할 수 있다. 디코더가 주변 블록을 이용하여 인트라 예측 모드를 유도하는 방법은 DIMD(Decoder side intra mode derivation)로 기술될 수 있다.

[0109] 도 7은 인터 예측에서 움직임 후보 리스트를 구성하기 위해 사용되는 주변 블록들의 위치를 나타낸 도면이다.

[0110] 주변 블록들은 공간적인 위치의 블록이거나 시간적인 위치의 블록일 수 있다. 현재 블록에 공간적으로 인접한 주변 블록은 좌측(Left, A1) 블록, 좌하측(Left Below, A0) 블록, 상측(Above, B1) 블록, 상우측(Above Right, B0) 블록 또는 상좌측(Above Left, B2) 블록 중 적어도 하나가 될 수 있다. 현재 블록에 시간적으로 인접한 주변 블록은 대응되는 픽처(Collocated picture)에서 현재 블록의 하우측(bottom Right, BR) 블록의 좌상단 픽셀 위치를 포함하는 블록이 될 수 있다. 상기 현재 블록에 시간적으로 인접한 주변 블록이 인트라 모드로 부호화되거나 상기 현재 블록에 시간적으로 인접한 주변 블록이 사용할 수 없는 위치에 존재하면, 현재 픽처에 대응되는 픽처(Collocated picture)에서 현재 블록의 가로 및 세로의 중앙(Center, Ctr) 픽셀 위치를 포함하는 블록이 시간적 주변 블록으로 사용될 수 있다. 대응되는 픽처에서 유도된 움직임 후보 정보는 TMVP(Temporal Motion Vector Predictor)라 지칭될 수 있다. TMVP는 하나의 블록에서 하나만 유도될 수 있고, 하나의 블록을 여러 개의 서브 블록으로 분할한 후, 각 서브 블록마다 각각의 TMVP 후보가 유도될 수 있다. 서브 블록 단위의 TMVP 유도 방법은 sbTMVP(sub-block Temporal Motion Vector Predictor)로 지칭될 수 있다.

[0111] 본 명세서에서 설명하는 방법들이 적용될 것인지 여부는 슬라이스 타입 정보 (예, I 슬라이스, P 슬라이스, B 슬라이스 인지 여부), 타일인지 여부, 서브 픽쳐인지 여부, 현재 블록의 크기, 코딩 유닛의 깊이, 현재 블록이 휘도 블록인지 색차 블록인지 여부, 참조 프레임인지 비참조 프레임인지 여부, 참조 순서 및 계층에 따른 시간적인 계층 등에 대한 정보들 중 적어도 어느 하나에 기초하여 결정될 수 있다. 본 명세서에서 설명하는 방법들

이 적용될 것인지 여부를 결정하기 위해 사용되는 정보들은 디코더 및 인코더 간 미리 약속된 정보일 수 있다. 또한, 이러한 정보들은 프로파일 및 레벨에 따라 결정되어 있을 수 있다. 이러한 정보들은 변수 값으로 표현될 수 있고, 비트스트림에는 변수 값에 대한 정보가 포함될 수 있다. 즉, 디코더는 비트스트림에 포함된 변수 값에 대한 정보를 파싱하여 상술한 방법들이 적용되는지 여부를 결정할 수 있다. 예를 들어, 코딩 유닛의 가로의 길이 또는 세로의 길이에 기초하여 상술한 방법들이 적용될 것인지 여부가 결정될 수 있다. 가로의 길이 또는 세로의 길이가 32 이상(예, 32, 64, 128 등)이면 상술한 방법들은 적용될 수 있다. 또한 가로의 길이 또는 세로의 길이가 32 보다 작은 경우(예, 2, 4, 8, 16)에 상술한 방법들은 적용될 수 있다. 또한 가로의 길이 또는 세로의 길이가 4 또는 8인 경우 상술한 방법들은 적용될 수 있다.

[0112] 도 8은 본 발명의 일 실시예에 따른 움직임 정보를 보정하는 방법을 나타내는 도면이다.

[0113] 도 8(a)는 현재 블록의 주변 블록으로부터 유도된 움직임 정보를 보정(revision)하여 새로운 움직임 정보를 출력하는 과정을 나타낸다. 도 8(a)를 참조하면 디코더는 현재 블록의 주변 블록에 대한 움직임 정보를 다양한 움직임 보정 방법으로 보정하여 보정된 움직임 정보를 획득할 수 있다. 도 8(b)를 참조하면 디코더는 현재 블록의 주변 블록으로부터 움직임 후보 리스트를 유도한 후, 유도된 움직임 후보 리스트 내의 하나 이상의 움직임 후보들을 다양한 움직임 보정 방법으로 보정하여 보정된 움직임 후보 리스트를 획득할 수 있다. 움직임 후보 리스트는 현재 블록의 주변 블록으로부터 유도된 움직임 정보를 이용하여 구성될 수 있다. 디코더는 움직임 후보 리스트 내의 하나 이상의 움직임 후보들을 각각 또는 전부에 대해 움직임 보정 과정을 수행하여 보정된 하나 이상의 움직임 후보들을 포함하는 보정된 움직임 후보 리스트를 획득할 수 있다. 도 8(c)를 참조하면 디코더는 현재 블록에 대한 초기 움직임 정보를 다양한 움직임 보정 방법으로 보정하여 보정된 움직임 정보를 획득할 수 있다.

[0114] 도 9는 본 발명의 일 실시예에 따른 움직임 보정 방법을 재귀적으로 수행하여 현재 블록에 대한 움직임 정보를 보정하는 방법을 나타내는 도면이다.

[0115] 디코더는 움직임 보정 방법을 재귀적으로 수행하여 현재 블록에 대한 초기 움직임 정보를 보정할 수 있다. 디코더는 현재 블록의 주변 블록들을 이용하여 현재 블록에 대한 움직임 후보 리스트를 구성하고, 하나 이상의 움직임 보정 방법을 재귀적으로 수행하여 움직임 정보를 보정할 수 있다. 이때, 하나 이상의 움직임 보정 방법은 MVD(Motion Vector Difference), TM(Template Matching), BM(Bilateral Matching), MMVD(Merge mode with MVD), MMVD(Merge mode with MVD) 기반의 TM, Optical flow 기반 TM, Multi pass DMVR 등일 수 있다. MVD는 인코더가 움직임 정보에 대한 보정 값을 비트스트림에 포함하여 생성하고, 디코더는 비트스트림을 통해 움직임 정보에 대한 보정 값을 획득하여 움직임 정보를 보정하는 방법일 수 있다(도 9의 MV 차분 값 보정). TM은 디코더가 현재 블록의 주변 화소에 기초하여 템플릿을 구성하고, 구성된 템플릿과 가장 유사도가 높은 매칭 영역을 찾아 움직임 정보를 보정하는 방법일 수 있다. BM은 디코더가 현재 블록의 움직임 정보를 기반으로 유도된 L0 픽쳐 리스트에 포함된 픽쳐 내의 참조 블록과 L1 픽쳐 리스트에 포함된 픽쳐 내의 참조 블록 간의 유사도에 기초하여 움직임 정보를 보정하는 방법일 수 있다. MMVD 방법은 하나 이상의 움직임 차분 값 후보들 중에서 하나를 이용하여 움직임 정보를 보정하는 방법이다. 인코더는 하나 이상의 움직임 차분 값 후보들 중에서 어느 하나를 나타내는 인덱스에 대한 정보를 포함하는 비트스트림을 생성할 수 있다. 디코더는 비트스트림에 포함된 인덱스에 대한 정보를 파싱하여 인덱스가 지시하는 차분 값 후보를 획득하고, 획득한 차분 값 후보에 기초하여 움직임 정보를 보정할 수 있다. MMVD 기반의 TM 방법은 움직임 후보 리스트와 하나 이상의 움직임 차분 값 후보들을 이용하여 구성되는 확장된 움직임 후보 리스트가 TM 코스트 값에 기초하여 재정렬되고, 재정렬된 리스트 내의 움직임 후보를 이용하여 현재 블록의 움직임 정보를 보정하는 방법이다. 인코더는 재정렬된 리스트 내의 후보들 중 어느 하나를 나타내는 인덱스에 대한 정보를 포함하는 비트스트림을 생성할 수 있다. 디코더는 비트스트림에 포함된 인덱스에 대한 정보를 파싱하여 해당 인덱스가 지시하는 움직임 보정 후보를 현재 블록의 움직임 정보의 보정 값으로 사용할 수 있다. Optical flow 기반 TM 방법은 디코더가 현재 블록에 인접한 영역의 템플릿을 Optical flow 맵으로 구성하여 참조 픽쳐의 Optical flow 맵과 유사한 영역을 찾아서 움직임 정보를 보정하는 방법일 수 있다.

[0116] 하나 이상의 움직임 보정 방법은 머지 또는 AMVP 모드에 적용될 수 있다. 도 9를 참조하면 디코더는 현재 블록에 대한 움직임 후보 리스트(예, 머지 후보 리스트)를 유도(구성)할 수 있다. 그리고 디코더는 하나 이상의 움직임 보정 방법을 이용하여 움직임 후보 리스트 내 움직임 정보(예, 머지 후보)를 각각 또는 전부 보정할 수 있다. 디코더는 보정된 움직임 정보에 대한 코스트(Cost) 값에 기초하여 움직임 후보 리스트를 재정렬할 수 있다. 상술한 바와 같이 디코더는 재정렬된 움직임 후보 리스트 내의 움직임 후보들 각각 또는 전부에 대해 상술한 보정 방법을 수행하고, 움직임 후보 리스트를 재정렬할 수 있다. 즉, 디코더는 상술한 움직임 후보들에 대한 보정 및 움직임 후보 리스트의 재정렬을 재귀적으로 수행할 수 있다. 이러한 방법이 적용되면 움직임 후보 리스트 내

움직임 후보에 대한 움직임 정보의 정확도가 높아지고, 이로 인해, 잔여 신호가 줄어들 수 있어, 잔여 신호에 대한 비트량이 감소되는 효과가 있을 수 있다. 디코더는 재정렬된 움직임 후보 리스트 내의 움직임 후보들 중 어느 하나를 지시하는 인덱스를 별도로 시그널링 받고 인덱스가 지시하는 움직임 후보에 기초하여 현재 블록을 예측(복원) 할 수 있다. 또는 디코더는 코스트 값이 가장 낮은 움직임 후보를 선택하여 코스트 값이 가장 낮은 움직임 후보에 기초하여 현재 블록을 예측(복원)할 수 있다.

[0117] 머지 후보의 정확도를 높이기 위해서, 인코더는 MMVD(Merge mode with MVD) 방법을 사용하여 획득되는 움직임 정보의 보정 값들 중 어느 하나를 나타내는 인덱스에 대한 정보를 포함하여 비트스트림을 생성할 수 있고, 디코더는 비트스트림에 포함된 인덱스에 대한 정보를 파싱하여 획득되는 인덱스를 통해 움직임 정보의 보정 값을 획득하고, 움직임 종보의 보정 값을 현재 블록을 예측(복원)하는데 사용할 수 있다. MMVD 방법은 복수 개의 움직임 차분 값 후보들 중에서 어느 하나를 선택하는 방법으로 기존의 움직임 정보 차분 값을 정확하게 보내는 것 대비 정확도는 다소 부족하나 많은 비트량을 절약할 수 있다는 효과가 있다. 정확도를 조금 더 높이기 위해 디코더는 MMVD 방법을 사용하여 획득되는 움직임 정보의 보정 값에 기초하여 보정되는 제1 보정 움직임 정보에 TM, BM, MMVD(Merge mode with MVD), MMVD(Merge mode with MVD) 기반의 TM, Optical flow기반 TM, Multi pass DMVR 방법 중 적어도 어느 하나의 방법을 추가적으로 적용하여 제2 보정 움직임 정보를 획득할 수 있다. 또는, 인코더는 TM, BM, MMVD(Merge mode with MVD), MMVD(Merge mode with MVD) 기반의 TM, Optical flow 기반 TM, Multi pass DMVR 방법 중 적어도 어느 하나의 방법을 적용하여 움직임 정보를 보정한 후, 상기 적어도 어느 하나의 방법을 적용하여 획득되는 움직임 정보에 대한 보정 값에 대한 정보를 추가적으로 포함하여 비트스트림을 생성할 수 있다.

[0118] AMVP를 위해서는 움직임 정보의 차분 값이 비스트스트림에 포함될 수 있다. 디코더는 비트스트림에 포함된 움직임 정보의 차분 값을 사용하여 현재 블록에 대한 예측 블록을 생성할 수 있다. 움직임 정보의 차분 값이 비트스트림에 포함되기 때문에, 비트량이 증가되는 문제가 있다. 이를 위해 상술한 방법은 AMVP 후보 리스트에도 적용될 수 있다. 즉, AMVP를 이용하여 획득된 후보 리스트 내 하나 이상의 후보들 각각 또는 전부는 TM, BM, MMVD(Merge mode with MVD), MMVD(Merge mode with MVD) 기반의 TM, Optical flow기반 TM, Multi pass DMVR 중 적어도 어느 하나의 방법에 기초하여 보정될 수 있다. 보정된 움직임 정보가 사용되기 때문에 실제 비트스트림에 포함되는 움직임 정보의 차분 값에 대한 비트량이 줄어드는 효과가 있다.

[0119] MVD 방법이 먼저 수행되고, TM, BM, MMVD(Merge mode with MVD), MMVD(Merge mode with MVD) 기반의 TM, Optical flow기반 TM, Multi pass DMVR 중 적어도 어느 하나를 이용한 보정 방법이 수행될 수 있다. 또는 인코더는 MVD를 사용하여 초기 움직임 정보에 대한 보정 값을 포함한 비트스트림을 생성할 수 있다. 디코더는 비트스트림에 포함된 초기 움직임 정보에 대한 보정 값에 기초하여 상술한 움직임 보정 방법을 수행할 수 있다.

[0120] 이하에서 MVD(Motion Vector Difference), TM(Template Matching), BM(Bilateral Matching), MMVD(Merge mode with MVD), MMVD(Merge mode with MVD) 기반의 TM, Optical flow 기반 TM, Multi pass DMVR 등을 적용하는 방법에 대해 설명한다.

[0121] 현재 블록의 부호화 모드(예측 모드)에 기초하여 사용되는 움직임 보정 방법이 결정될 수 있다. 예를 들어, 현재 블록이 GPM 모드로 부호화된 경우, 먼저 MMVD를 이용하여 보정된 MV 차분 값에 대해 TM, BM, Optical flow 기반 TM 방법 중에서 적어도 어느 하나가 수행될 수 있다. 예를 들어, 현재 블록이 AMVP 모드로 부호화된 경우, 머지 모드의 MMVD를 이용하여 보정된 MV 차분 값에 대해 TM, BM, Optical flow 기반 TM 방법 중에서 적어도 어느 하나가 수행될 수 있다. 이때, AMVP 모드의 MVD는 적용되지 않을 수 있다.

[0122] 예를 들어, 현재 블록에 인접한 주변 블록들의 움직임 정보가 서로 같거나 비슷한 경우, MV 차분 값에 대한 보정은 수행되지 않고, TM, BM, Optical flow 기반 TM 방법 중에서 적어도 어느 하나가 수행될 수 있다. 현재 블록의 움직임도 주변 블록의 움직임과 비슷할 가능성이 크기 때문이다. 한편 현재 블록에 인접한 주변 블록들의 움직임 정보들의 분포가 서로 비슷하지 않는 경우, MVD 또는 MMVD를 이용하여 보정된 MV 차분 값에 대해 TM, BM, Optical flow 기반 TM 방법 중에서 적어도 어느 하나가 수행될 수 있다. 현재 블록의 움직임이 주변 블록과 상이할 수 있기 때문이다.

[0123] 예를 들어, 현재 블록의 크기, 현재 블록이 휘도 성분 블록인지 색차 성분 블록인지 여부, 현재 블록에 대한 양자화 파라미터 정보, 현재 블록의 움직임 해상도 정보들, 현재 블록에 차분 신호가 존재하는지 여부, 현재 블록에 대한 차분 신호에서 0이 아닌 양자화 인덱스들의 절대값의 합 혹은 개수 중 적어도 어느 하나에 기초하여 움직임 보정 방법(예, MVD(Motion Vector Difference), TM(Template Matching), BM(Bilateral Matching), Optical flow 기반 TM, Multi pass DMVR 등)이 선택될 수 있다. 현재 블록의 크기가 임의의 크기 이상이거나

현재 블록의 움직임 해상도가 1/16 화소 단위인 경우, TM 방법은 선택되지 않을 수 있다. TM 방법은 복잡도가 높기 때문이다. 현재 블록이 색차 성분 블록인 경우, TM 방법이 수행되지 않고, 현재 블록의 휘도 성분 블록에서 TM 방법을 통해 보정된 움직임 정보가 사용될 수 있다. 예를 들어, 현재 블록의 휘도 성분 블록에서 TM 방법을 통해 보정된 움직임 정보는 스케일(scale)되어 현재 블록의 색차 블록에 사용될 수 있다.

[0124] 예를 들어, 현재 블록의 특성에 따라 움직임 보정 방법(예, MVD(Motion Vector Difference), TM(Template Matching), BM(Bilateral Matching), MMVD(Merge mode with MVD), MMVD(Merge mode with MVD) 기반의 TM, Optical flow 기반 TM, Multi pass DMVR 등)은 선택될 수 있다. 각 움직임 보정 방법마다 복잡도 및 정확도 간의 상충관계(trade-off)가 존재하기 때문이다. 예를 들어, TM은 복잡도가 높고 병렬처리가 불가한 반면에 가장 높은 성능을 보이며, BM은 TM보다 성능은 낮지만 병렬처리가 가능하며, Optical flow는 복잡도는 낮고 병렬처리도 가능하지만, 성능이 낮다는 단점이 있다. 이때, 선택되는 움직임 보정 방법은 별도로 시그널링될 수 있다. 예를 들어, 디코더는 비트스트림에 포함된 신택스 요소에 의해 움직임 보정 방법을 결정할 수 있다. 이때, 신택스 요소는 SPS 레벨, PPS 레벨, 픽쳐 레벨, 또는 슬라이스 레벨, CU(Coding Unit) 레벨에서 시그널링될 수 있다.

[0125] 도 10은 본 발명의 일 실시예에 따른 TM 방법이 수행되는 순서를 나타내는 도면이다.

[0126] 도 10을 참조하면 디코더는 주변 블록으로부터 유도된 초기 움직임 정보(initial MV, reference index)를 획득할 수 있다. 디코더는 초기 움직임 정보에 기초하여 참조 영상 내에서 탐색 영역을 설정할 수 있다. 디코더는 미리 정의된 탐색 패턴에 따라, 탐색 영역 내에서 몇 개의 후보 위치를 선정할 수 있다. 디코더는 현재 블록의 주변 블록을 사용하여 현재 블록에 대한 템플릿을 구성하고, 후보 위치를 기준으로 현재 블록에 대한 템플릿과 동일한 크기의 참조 영상 템플릿을 구성할 수 있다. 디코더는 현재 블록에 대한 템플릿과 참조 영상 템플릿 간의 코스트(cost) 값을 획득할 수 있다. 코스트 값은 SAD(Sum of Absolute Differences) 또는 MRSAD(Mean-Removed SAD)를 통해 획득될 수 있다. 디코더는 탐색 영역 내의 모든 후보 위치들에 대하여, 코스트 값을 획득하고, 최소의 코스트 값에 대응되는 위치의 움직임 후보의 정보를 최종 움직임 정보(도 10의 개선된 움직임 정보)로 사용할 수 있다. 본 명세서에서 코스트 값을 획득한다는 의미는 디코더가 코스트 값을 계산한다는 의미와 동일할 수 있다.

[0127] 도 11은 본 발명의 일 실시예에 따른 초기 움직임 정보에 기초하여 TM 방법을 위한 탐색 영역을 설정하는 방법을 나타내는 도면이다.

[0128] 도 11을 참조하면 디코더는 참조 픽쳐에서 현재 블록에 대응되는 참조 블록을 찾기 위해, 현재 블록의 좌상단 위치를 기준으로 초기 움직임 정보(initial MV)만큼 이동한 위치를 참조 블록의 위치로 설정할 수 있다. 디코더는 참조 블록의 좌상단 위치를 기준으로 임의의 (m x n) 크기만큼 탐색 영역(Search range)을 설정할 수 있다. 이때, m x n 은 16 x 16일 수 있다. 예를 들어, 탐색 영역은 초기 움직임 정보의 위치를 기준으로 수평 방향으로 -8에서 8만큼의 범위를 가질 수 있고, 수직 방향으로 -8에서 8까지의 범위를 가질 수 있다. 구체적으로 초기 움직임 정보가 나타내는 위치를 수평 방향, 수직 방향의 좌표 형태로 표현하면 (x, y)일 수 있다. 이때, 탐색 영역의 수평 방향의 좌표는 x-8에서 x+8 까지의 범위일 수 있고, 수직 방향의 좌표는 y-8에서 y+8 까지의 범위일 수 있다. 이후 디코더는 현재 블록에 인접한 블록을 이용하여 현재 블록의 좌측 템플릿과 현재 블록의 상측 템플릿을 구성할 수 있다. 또한, 디코더는 참조 픽쳐에서도 설정한 참조 블록의 위치를 기준으로 참조 블록의 좌측 템플릿과 참조 블록의 상측 템플릿을 구성한다. 이때, 현재 블록의 좌측 템플릿과 참조 블록의 좌측 템플릿의 크기는 동일할 수 있고, 현재 블록의 상측 템플릿과 참조 블록의 상측 템플릿의 크기는 동일할 수 있다.

[0129] 도 12는 본 발명의 일 실시예에 따른 탐색 영역 내에서 탐색되는 움직임 후보의 위치를 나타내는 도면이다.

[0130] 도 12를 참조하면 TM 방법을 위한 탐색 영역 내에서 탐색되는 움직임 후보의 위치는 초기 움직임 정보에 대한 위치(도 12의 가운데 점)를 기준으로 설정될 수 있다. 움직임 후보가 탐색되는 위치는 탐색 패턴에 따라 달라질 수 있다. 탐색 패턴은 다이아몬드(DIAMOND) 패턴, 크로스(CROSS) 패턴 등이 있을 수 있다. 도 12에서 ◇는 다이아몬드 패턴에 따라 움직임 후보가 탐색되는 위치를 나타내고, +는 크로스 패턴에 따라 움직임 후보가 탐색되는 위치를 나타낼 수 있다. 패턴의 간격은 초기 움직임 정보에서 멀어질수록 간격이 더 넓어지도록 설정되거나 좁아지도록 설정될 수 있다.

[0131] 도 13은 본 발명의 일 실시예에 따른 움직임 후보의 위치를 탐색하는 과정을 나타내는 도면이다.

[0132] 구체적으로, 도 13은 TM 방법을 위한 움직임 후보가 탐색되는 위치에 대한 탐색 수행 과정을 나타낸 순서도이다. 도 13을 참조하면 디코더는 초기 움직임 정보에 대한 화소(픽셀) 기반 코스트 값을 획득(계산)할

수 있다. 디코더는 탐색 수행의 반복횟수가 '0'이라면, TM 방법을 수행하지 않고 종료할 수 있다. 그렇지 않다면(즉, 탐색 수행의 반복횟수가 '0'이 아니라면), 디코더는 TM 방법을 수행할 수 있다. 이때, 반복횟수는 1 이상의 임의의 정수 값일 수 있다. 또한, 반복횟수, 초기 탐색 패턴, 초기 탐색 간격은 도 13을 통해 설명하는 탐색 수행 과정이 수행되기 전에 설정될 수 있다. 예를 들어, 반복횟수는 '375', 초기 탐색 패턴은 '다이아몬드(DIAMOND)', 초기 탐색 간격은 '6'으로 설정될 수 있다.

[0133] 다음으로, 탐색 패턴과 탐색 간격이 재설정될 수 있다. 탐색 패턴과 탐색 간격은 현재 블록의 크기, 현재 블록이 휘도 성분 블록인지 색차 성분 블록인지 여부, 현재 움직임 해상도, 반복횟수, 이전 반복 단계에서 계산된 움직임 후보 위치에 대한 코스트 값의 분포 중에서 적어도 하나 이상에 기초하여 결정될 수 있다. 이하에서 탐색 패턴과 탐색 간격을 설정하는 방법에 대해 설명한다.

[0134] 탐색 간격은 현재 블록의 움직임 정보 해상도에 따라 결정될 수 있다. 움직임 정보 해상도는 1의 정수 화소, 4의 정수 화소, 1/2 정수 화소, 1/4 정수 화소, 1/16 정수 화소 단위가 될 수 있다. 움직임 정보 해상도가 1/4 정수 화소인 경우 초기 탐색 간격은 6으로 설정될 수 있고 그 외의 경우에 초기 탐색 간격은 4로 설정될 수 있다.

[0135] 탐색 패턴은 다이아몬드 패턴 또는 크로스 패턴으로 결정될 수 있다. 탐색 간격은 초기 탐색 간격에서 임의의 간격만큼 감소되거나 증가되면서 조정될 수 있다. 예를 들어, 탐색 패턴과 탐색 간격은 반복 단계에 다라 달라질 수 있다. 반복 단계는 반복횟수가 0이 아닌 경우 탐색 패턴과 탐색 간격의 재설정이 몇 번째로 반복되는지를 나타낼 수 있다. 즉, 몇 번째 반복 단계인지에 따라 탐색 패턴과 탐색 간격이 달라질 수 있다. 예를 들어 첫 번째 반복 단계에서 탐색 패턴과 탐색 간격은 다이아몬드 탐색 패턴과 초기 탐색 간격으로 설정될 수 있다. 두 번째 단계에서 탐색 패턴과 탐색 간격은 크로스 패턴과 초기 탐색 간격에서 1만큼 감소시킨 탐색 간격으로 설정될 수 있다. 두 번째 단계 이후 단계에서 탐색 패턴과 탐색 간격은 크로스 패턴과 이전 단계보다 1만큼 감소된 탐색 간격으로 설정될 수 있다.

[0136] 탐색 패턴과 탐색 간격은 현재 블록의 컬러 성분에 따라 설정될 수 있다. 색차 성분에 대한 탐색 패턴과 탐색 간격은 휘도 성분에 대한 탐색 패턴과 탐색 간격보다 넓게 설정될 수 있다. 휘도 성분(신호) 대비 색차 성분(신호)는 공간적인 상관도가 높기 때문이다. 또는, 성능을 높이기 위해서 색차 성분에 대한 탐색 패턴과 탐색 간격은 휘도 성분에 대한 탐색 패턴과 탐색 간격보다 더 짧게 설정될 수 있다.

[0137] 탐색 패턴과 탐색 간격은 현재 블록의 크기에 기초하여 설정될 수 있다. 현재 블록의 크기가 임의의 정해진 값보다 큰 경우, 탐색 패턴은 크로스 패턴이 사용되고 탐색 간격은 초기 탐색 간격보다 넓게 설정될 수 있다. 예를 들어 탐색 간격은 '7'일 수 있다. 또는, 성능을 높이기 위해 현재 블록의 크기가 임의의 정해진 수보다 큰 경우, 탐색 패턴은 다이아몬드 패턴이 사용되고 탐색 간격은 초기 탐색 간격보다 짧게 설정될 수 있다. 예를 들어 탐색 간격은 '5'일 수 있다. 현재 블록의 크기는 16x16, 32x32가 될 수 있으며, 탐색 패턴과 탐색 간격은 현재 블록의 가로의 크기 및 세로의 크기의 합에 기초하여 설정될 수 있다.

[0138] 다음으로, 디코더는 탐색 패턴과 탐색 간격을 이용하여 탐색되는 움직임 후보의 위치에 대한 보정 값(offset)을 설정하고, 탐색되는 움직임 후보들에 대한 평가를 수행할 수 있다. 본 명세서에서의 평가는 코스트 값을 획득하는 것을 의미할 수 있다. 움직임 후보가 탐색되는 위치는 탐색 패턴에 따라 달라질 수 있다. 예를 들어, 탐색 패턴이 크로스 패턴인 경우, 보정 값은 (0, 1), (1, 0), (0, -1), (-1, 0)이며, 탐색 패턴이 다이아몬드 패턴인 경우, 보정 값은 (0, 2), (1, 1), (2, 0), (1, -1), (0, -2), (-1, -1), (-2, 0), (-1, 1)이다. 이때 보정 값 (x, y)는 (수평, 수직)으로 x는 수평 방향에 대한 보정 값이고 y는 수직 방향에 대한 보정 값일 수 있다.

[0139] 도 13을 통해 설명한 방법은 임의의 정해진 반복횟수만큼 재귀적으로 수행될 수 있다. 예를 들어 반복횟수가 1이면 1회 이상 수행될 수 있다. 디코더는 움직임 후보들을 탐색하여 탐색되는 움직임 후보들 전부에 대한 평가를 수행하였다면 가장 작은 코스트 값에 대응되는 움직임 후보의 움직임 정보를 최종 움직임 정보로 사용할 수 있다.

[0140] 도 13의 초기 움직임 후보는 이전 반복 단계에서 가장 작은 코스트 값에 기초하여 재설정될 수 있다. 첫 번째 단계에서 가장 작은 코스트 값에 대응되는 움직임 후보를 기준으로 다음 단계의 초기 움직임 후보가 재설정될 수 있다. 예를 들어, 첫 번째 단계에서 가장 작은 코스트 값에 대응되는 움직임 후보가 좌상단에 위치한 움직임 후보인 경우, 다음 반복 단계에서는 좌상단에 위치한 움직임 후보에 인접한 움직임 후보에 대해서 디코더는 평가를 수행할 수 있다.

[0141] 현재 블록이 코딩 유닛(블록)인지 서브 블록(서브 코딩 블록)인지에 따라 도 13을 참조하여 설명한 탐색 수행

과정은 다르게 진행될 수 있다.

[0142] 현재 블록이 코딩 블록이고 현재 블록에 AMVP 모드가 적용되는 경우, 현재 블록의 L0 예측을 위한 L0 움직임 후보 리스트와 L1 예측을 위한 L1 움직임 후보 리스트가 유도될 수 있다. 유도된 후보 리스트의 일부 또는 모든 움직임 후보에 대해 탐색 수행 과정이 진행되어 보정된 움직임 정보가 유도될 수 있다. 한편, 현재 블록이 코딩 블록이고 현재 블록에 Merge 모드가 적용되는 경우, 현재 블록의 L0 및 L1 예측을 위한 하나의 움직임 후보 리스트가 유도될 수 있다. 유도된 하나의 움직임 후보 리스트 내의 일부 또는 모든 움직임 후보들에 대해 탐색 수행 과정이 진행될 수 있다. 본 명세서에서 기술하는 L0는 L0 예측, L1은 L1 예측을 의미할 수 있다.

[0143] 현재 블록에는 L0 단방향 예측, L1 단방향 예측 또는 양방향 예측이 적용될 수 있다. 현재 블록에는 L0 단방향 예측, L1 단방향 예측 또는 양방향 예측 중 어느 것이 적용되는지 여부는 참조 방향 지시 정보에 의해 지시될 수 있다. 참조 방향 지시 정보는 코스트 값에 기초하여 재설정될 수 있다. 예를 들어, L0의 초기 움직임 정보를 이용하여 생성되는 예측 블록의 코스트 값, L1의 초기 움직임 정보를 이용하여 생성되는 예측 블록의 코스트 값, L0와 L1의 초기 움직임 정보를 통해 양방향 예측을 수행하여 2개의 예측 블록을 가중치 평균하여 생성되는 예측 블록의 코스트 값, L0의 보정된 움직임 정보를 이용하여 생성되는 예측 블록의 코스트 값, L1의 보정된 움직임 정보를 이용하여 생성되는 예측 블록의 코스트 값, L0와 L1의 보정된 움직임을 통해 양방향 예측을 수행하여 2개의 예측 블록을 가중치 평균하여 생성되는 예측 블록의 코스트 값 중에서 가장 작은 코스트 값에 대응되는 움직임 정보 및 참조 방향 지시 정보가 현재 블록에 재설정될 수 있다.

[0144] 현재 코딩 블록은 여러 개의 서브 블록들로 분할될 수 있다. 각 서브 블록의 초기 움직임 정보는 탐색 수행 과정에 따라 보정된 움직임 정보로 재설정될 수 있다. 각 서브 블록마다 템플릿은 상이할 수 있고, 인접한 서브 블록의 화소(픽셀)가 템플릿으로 사용될 수 있다. 다만, 디코더는 인접한 서브 블록이 복원된 경우에만 다음 서브 블록에 대한 탐색을 수행할 수 있으므로, 탐색 수행 과정이 각 서브 블록 별로 병렬적으로 처리되지 않는 문제가 있다. 이를 위해 현재 블록의 경계에 위치하는 서브 블록에 대해서만 탐색 수행 과정이 진행될 수 있다. 또는, 현재 블록의 경계에 위치한 서브 블록에 대해 디코더는 TM 방법으로 보정된 움직임 정보를 유도하고, 현재 블록의 경계에 위치하지 않는 서브 블록에 대해 디코더는 BM, Optical flow 기반 TM, Multi pass DMVR 방법 중에서 어느 하나 이상을 이용하여 보정된 움직임 정보를 유도할 수 있다.

[0145] 현재 블록이 코딩 블록으로 처리되는 경우, 디코더는 L0와 L1에 대한 초기 움직임 정보를 사용하여 현재 블록 전체에 대한 코스트 값을 계산하고, 계산된 코스트 값에 기초하여 보정된 움직임 정보를 유도할 수 있다. 이때, 코딩 블록으로 처리되는 블록 내의 우하단 일부 영역의 움직임은 코딩 블록의 전체 움직임과 조금 상이할 수 있다. 템플릿이 어떻게 구성되는지에 따라 탐색 수행 과정은 달라질 수 있고, 보정된 움직임 정보도 달라질 수 있다. 따라서, 현재 블록이 코딩 블록으로 처리되더라도 보정된 움직임 정보는 서브 블록 기반으로 구성된 템플릿에 대한 코스트 값에 기초하여 서브 블록 단위로 보정된 움직임 정보가 유도될 수 있다.

[0146] 도 14는 본 발명의 일 실시예 따른 탐색 후보를 평가하는 과정을 나타내는 도면이다.

[0147] 구체적으로 도 14는 도 13을 참조하여 설명한 탐색 패턴과 탐색 간격에 따라 선정된 후보 위치에 대한 보정 값들을 사용하여 탐색 후보를 평가하는 과정을 나타내는 순서도이다. 디코더는 초기 움직임 정보를 최종 움직임 정보로 저장할 수 있다. 그리고 디코더는 탐색 후보 전부에 대한 보정 값들에 대해 후술하는 과정을 수행할 수 있다.

[0148] 디코더는 탐색할 후보 위치에 대한 보정 값들 중에서 하나를 선택할 수 있다. 보정 값은 현재 블록에서 사용될 움직임 해상도(resolution)에 적합하도록 재설정될 수 있다. 디코더는 재설정된 보정 값을 초기 움직임 정보에 더하여 평가할 움직임 정보를 재구성할 수 있다. 재구성된 움직임 정보에 기초하여 코스트 값을 획득할 수 있다. 이때, 움직임 정보에 기초하여 획득되는 코스트 값은 초기 움직임 정보와 재구성된 움직임 정보 간의 수평 방향 성분들의 절대값의 차이 값과 수직 방향 성분들의 절대값의 차이 값을 더한 후, 임의의 가중치 값을 곱하여 계산될 수 있다. 임의의 가중치 값은 '4'일 수 있다. 디코더는 움직임 정보에 기초하여 획득한 코스트 값 화소(픽셀)에 기초하여 획득한 초기 움직임 정보의 코스트 값보다 작은 경우에만 재구성된 움직임 정보에 대한 화소(픽셀)기반 코스트 값을 계산할 수 있다.

[0149] 디코더는 모든 탐색 후보에 대한 보정 값들에 대하여 평가한 후, 가장 작은 코스트 값에 대응되는 움직임 정보를 최종 움직임 정보로 설정할 수 있다.

[0150] 움직임 정보에 기초하여 획득되는 코스트 값들 각각은 초기 움직임 정보에 따라 선정되는 위치에 대응되는 복수의 움직임 후보들 각각에 보정 값을 더하여 획득되는 움직임 정보들 간의 차분 값을 통해 획득될 수 있다. 움직

임 정보에 기초하여 획득되는 코스트 값은 보정 값의 크기에 따라 달라질 수 있다. 즉, 보정 값이 작을수록 코스트 값은 작아질 수 있다. 가장 작은 코스트 값에 대응되는 움직임 정보가 최종 움직임 정보로 설정되기 때문에, 보정 값이 작은 초기 움직임 정보가 나타내는 위치의 주변 움직임 후보에 대해서만 평가가 수행될 수 있다. 그러나, 보정 값이 큰 움직임 후보가 최적의 움직임 후보일 수 있다. 따라서, 다양한 움직임 후보에 대한 평가를 수행하여 최적의 움직임 후보를 선정하기 위해, 코스트 값은 후술하는 방법을 이용하여 획득될 수 있다. 코스트 값은 주변 블록 각각의 움직임 정보 값 간의 차이, 양자화 파라미터, 현재 블록의 크기 등을 이용하여 획득될 수 있다.

[0151] 코스트 값은 주변 블록의 움직임 정보의 분포를 이용하여 획득될 수 있다. 다양한 움직임 후보에 대한 평가를 위해, 디코더는 보정된 움직임 정보와 주변 블록의 움직임 정보 간의 차이 값을 이용하여 코스트 값을 획득할 수 있다. 예를 들어, 보정된 움직임 정보와 주변 블록의 움직임 정보 간의 차이 값과 임의의 정해진 값을 비교하여 비교 결과에 따라 코스트 값을 획득할 수 있다. 구체적으로 보정된 움직임 정보와 주변 블록의 움직임 정보 간의 차이 값이 임의의 정해진 값보다 큰 경우(또는 작은 경우, 같은 경우) 디코더는 코스트 값을 획득할 수 있다. 이때 주변 블록은 현재 블록에 인접한 주변 블록이거나 또는 대응 픽처(Collocated picture)에서 현재 블록과 동일한 위치에 있는 시간적인 주변 블록일 수 있다.

[0152] 코스트 값은 현재 블록의 크기에 기초하여 획득될 수 있다. 예를 들어, 상술한 코스트 값을 획득하기 위한 가중치는 현재 블록의 크기에 따라 설정될 수 있다. 가중치는 현재 블록의 크기와 반비례하게 설정될 수 있다. 즉, 현재 블록의 크기가 클수록 가중치는 낮게 설정될 수 있다. 이는, 적합한 움직임 후보를 선정하기 위해 보다 더 넓은 범위의 움직임 후보를 평가하기 위함이다. 한편 가중치는 현재 블록의 크기와 비례하게 설정될 수 있다. 즉, 현재 블록의 크기가 클수록 가중치는 높게 설정될 수 있다. 이는 복잡도를 낮추기 위함이다. 예를 들어, 현재 블록의 크기는 16x16, 32x32가 될 수 있고, 현재 블록의 가로 및 세로 크기의 합으로 설정될 수 있다. 가중치는 1, 2, 3, 4, 5, 6 등의 정수 값이 될 수 있다. 또한 가중치가 커질수록 코스트 값이 높아지므로 가중치가 특정 값 이상인 경우 디코더는 코스트 값을 획득하기 위한 평가를 수행하지 않을 수 있다.

[0153] 도 15는 본 발명의 일 실시예에 따른 현재 픽처 내 객체의 경계 부분에 대한 움직임 특성을 나타내는 도면이다.

[0154] 일반적으로 현재 블록의 움직임 정보는 주변 블록의 움직임 정보와 상관성이 높다. 하지만, 현재 블록이 객체의 경계 부분에 위치한 경우, 현재 블록은 배경 영역일 수 있다. 이 경우, 주변 블록의 움직임은 현재 블록의 움직임과 다를 수 있다. 또한 카메라의 위치(시점) 변경 없이 촬영된 영상의 경우, 배경은 움직이지 않으므로, 배경에 대한 움직임 정보는 영 벡터(0, 0)일 수 있다. 디코더는 TM을 위해 주변 블록의 움직임 정보를 이용하여 움직임 후보를 탐색할 수 있다. 이때 현재 블록이 배경에 해당하는 경우, 주변 블록의 움직임 정보를 이용하여 움직임 후보를 탐색하기 보다 영 벡터에 대한 탐색이 보다 효과적일 수 있다. 따라서, TM이 수행되는 경우, 탐색되는 움직임 후보에 영 벡터가 포함될 수 있다.

[0155] 카메라가 일정한 속도로 이동하며 촬영한 영상의 경우, 배경 부분에 대한 움직임 정보는 일정한 값(전역 움직임)을 가질 수 있다. 이 경우에는 TM을 위해 탐색되는 움직임 후보에 전역(global) 움직임 후보가 포함될 수 있다.

[0156] 현재 픽처에 대한 참조 픽처는 콘텐츠를 사용하는 환경에 따라 다르게 구성될 수 있다. 예를 들어, 실시간 방송 환경에서는 지연 속도가 낮은 low delay B와 같은 참조 픽처가 사용될 수 있다. 즉, 픽처의 재생 시간 순서상 과거에 위치하는 디코딩된 픽처만이 참조 픽처로 사용될 수 있다. 또한 VoD(Video on Demand)와 같은 스트리밍 서비스에서는 임의 접근이 중요하므로 GOP(Group of Picture) 단위로 I, P 픽처가 부호화되는 랜덤 액세스(random access)와 같은 참조 픽처 구성이 사용될 수 있다. 즉, 픽처의 재생 시간 순서상 과거 그리고 미래에 위치하는 디코딩된 픽처가 참조 픽처로 사용될 수 있다. 현재 픽처에 대한 참조 픽처를 구성하는 방법은 슬라이스(Slice) 헤더를 통해 시그널링되므로, 디코더는 현재 픽처를 부호화할 때 이미 참조 픽처로 사용할 픽처들을 확인할 수 있다. 이때 블록 단위로 사용되는 참조 픽처가 달라질 수 있으며, 어떠한 참조 픽처를 사용되는지에 대한 정보는 블록 단위로 시그널링될 수 있다.

[0157] 도 16은 본 발명의 일 실시예에 따른 템플릿 매칭을 이용하여 참조 픽처를 결정하는 방법을 나타내는 도면이다.

[0158] 참조 픽처는 현재 블록과 가장 유사한 블록을 포함하는 픽처로 디코더는 템플릿 매칭을 통해 참조 픽처를 찾을 수 있다. 참조 픽처와 관련된 정보는 비트스트림에 포함되지 않고, 탬플릿 매칭을 통해 유도될 수 있다.

[0159] 먼저, 디코더는 평가할 참조 픽처 후보들과 평가할 움직임 후보들을 선정할 수 있다. 다음으로, 디코더는 각각의 참조 픽처에 대한 움직임 후보들을 평가할 수 있다. 이때, 각 움직임 후보들은 참조 픽처와 현재 픽처와의

거리 및 현재 픽쳐와 움직임 후보들의 참조 픽쳐 간의 거리(즉, Picture Order Count(POC) 차이)에 기초하여 스케일링될 수 있다. 디코더는 스케일링된 움직임 후보에 대한 평가를 수행할 수 있다. 모든 참조 픽쳐 후보와 모든 움직임 후보들에 대한 조합에 대하여 평가가 완료되었다면, 가장 작은 코스트 값에 대응되는 참조 픽쳐의 움직임 후보의 움직임 정보가 최종 움직임 정보일 수 있다. 예를 들어, 현재 픽쳐가 B 픽쳐인 경우(양 방향 예측이 사용되는 경우), 디코더는 가장 작은 코스트 값에 대응되는 참조 픽쳐의 움직임 후보의 움직임 정보와 두 번째로 작은 코스트 값에 대응되는 참조 픽쳐의 움직임 후보의 움직임 정보를 이용하여 양방향 움직임 정보를 구성할 수 있다. 또는 각 참조 픽쳐 리스트 내의 참조 픽쳐들 중 가장 작은 코스트 값을 가지는 각 참조 픽쳐의 움직임 후보를 이용하여 최종 움직임 정보가 획득될 수 있다. 참조 픽쳐 리스트내 참조 픽쳐 수는 복잡도 측면에서 한정될 수 있다. 이때, 한정되는 참조 픽쳐 수는 별도의 시그널링을 통해 설정될 수 있다.

[0160] 또 다른 방법으로 모든 참조 픽쳐 후보와 모든 움직임 후보들에 대한 조합에 대하여 평가가 완료되었다면, 인코더는 조합 정보에 대한 리스트를 구성한 후 코스트 값을 기반으로 오름차순으로 재정렬을 수행한 후, 최적의 조합 정보에 대한 인덱스 정보를 포함한 비트스트림을 생성할 수 있다. 디코더는 비트스트림에 포함된 인덱스 정보에 기초하여 현재 블록에 대한 참조 픽쳐 혹은 움직임 후보를 결정할 수 있다.

[0161] 도 16(a)는 랜덤 액세스 환경에서의 참조 픽쳐 구성을 나타낸다. 도 16(a)를 참조하면 L0 참조 픽쳐들 중에서 첫 번째 참조 픽쳐(reference index 0)의 참조 블록이 가장 작은 코스트 값을 가질 수 있고, L1 참조 픽쳐들 중에서 두 번째 참조 픽쳐(reference index 1)의 참조 블록이 가장 작은 코스트 값을 가질 수 있다. 디코더는 L0 참조 픽쳐들 중에서 첫 번째 참조 픽쳐(reference index 0)의 참조 블록에 대한 움직임 정보와 L1 참조 픽쳐들 중에서 두 번째 참조 픽쳐(reference index 1)의 참조 블록에 대한 움직임 정보를 이용하여 현재 블록에 대한 움직임 정보를 획득할 수 있다. 이때, L0 및 L1에 대한 참조 픽쳐 인덱스 정보는 비트스트림을 통해 시그널링되지 않고 템플릿 매칭을 통해 유도될 수 있다.

[0162] 도 16(b)는 저지연 환경에서의 참조 픽쳐 구성을 나타낸다. 도 16(b)를 참조하면 L0 참조 픽쳐와 L1 참조 픽쳐는 동일하게 구성될 수 있다. L0 참조 픽쳐들 중에서 두 번째 참조 픽쳐(reference index 1)의 참조 블록이 가장 작은 코스트 값을 가질 수 있고, L1 참조 픽쳐들 중에서 L0 참조 블록을 제외하고 세 번째 참조 픽쳐(reference index 2)의 참조 블록이 가장 작은 코스트 값을 가질 수 있다. 디코더는 L0 참조 픽쳐들 중에서 두 번째 참조 픽쳐(reference index 1)의 참조 블록에 대한 움직임 정보와 L1 참조 픽쳐들 중에서 세 번째 참조 픽쳐(reference index 2)의 참조 블록에 대한 움직임 정보를 이용하여 현재 블록에 대한 움직임 정보를 획득할 수 있다. 이때, L0 및 L1에 대한 참조 픽쳐 인덱스 정보는 비트스트림을 통해 시그널링되지 않고 템플릿 매칭을 통해 유도될 수 있다.

[0163] 상기 방법을 적용하는 참조 픽쳐 리스트 내 참조 픽쳐 수를 한정할 수 있다. 한정된 참조 픽쳐 수 내에서만 적용함으로써 복잡도를 줄일 수 있다. 한정된 개수는 미리 정해진 값으로 설정되거나 혹은 SPS 레벨, PPS 레벨, 픽쳐 레벨, 또는 슬라이스 레벨, CU(Coding Unit) 레벨에서 시그널링되어 비트스트림에 포함될 수 있으며, 디코더는 비트스트림으로부터 한정된 개수를 획득하여 적응적으로 참조 픽쳐 리스트 내 참조 픽쳐 수를 변경할 수 있다.

[0164] 도 17, 도 18은 본 발명의 일 실시예에 따른 템플릿 매칭을 수행하는 과정을 나타내는 도면이다.

[0165] 도 17을 참조하면 먼저 현재 블록의 주변 블록들로부터 움직임 후보가 유도될 수 있다.

[0166] 현재 블록의 상단 블록을 통해 첫 번째 초기 움직임 후보가 유도될 수 있다. 첫 번째 초기 움직임 후보에 기초하여 참조 픽쳐 내에서 움직임 후보를 탐색하기 위한 위치가 선정될 수 있다. 움직임 후보가 탐색되는 위치는 ◇(다이아몬드 패턴), +(크로스 패턴)로 표시된 위치가 될 수 있다. 디코더는 초기 움직임 후보에서 가장 멀리 떨어진 위치의 움직임 후보부터 가장 가까운 위치의 움직임 후보 순으로 평가를 수행할 수 있다. 또한, 이전 반복 단계에서 보정된 움직임 후보의 위치는 다음 반복 단계의 초기 움직임 후보 위치로 설정될 수 있다. 움직임 후보들 중에서 가장 작은 코스트 값에 대응되는 움직임 후보의 움직임 정보가 첫 번째 보정된 움직임 후보(1710)가 될 수 있다.

[0167] 다음으로, 현재 블록의 좌측 블록을 통해 두 번째 초기 움직임 후보가 유도될 수 있다. 디코더는 템플릿 매칭을 이용하여 가장 작은 코스트 값에 대응되는 두 번째 보정된 움직임 후보(1720)를 유도할 수 있다.

[0168] 첫 번째 보정된 움직임 후보(1710)의 코스트 값과 두 번째 보정된 움직임 후보(1720)의 코스트 값 중 작은 코스트 값에 대응되는 보정된 움직임 후보가 현재 블록에 대한 최종 움직임 후보가 된다.

[0169] 움직임 후보들을 탐색하는 범위가 넓을 수록, 탐색되는 위치가 많을수록 보다 정확한 최종 움직임 후보를 획득

할 수 있다. 본 명세서에서 기술하는 TM에서도 탐색 범위, 탐색 위치에 따라 서로 다른 결과가 유도될 수 있다. 도 18을 참조하면 현재 블록의 주변 블록으로부터 유도되는 초기 움직임 정보(1801, 1802, 1803)의 분포가 서로 떨어져 있는 경우, 탐색되는 위치가 서로 중복되지 않으므로 다양하고 넓은 범위의 움직임 후보가 평가될 수 있고 가장 최적의 움직임 정보가 유도될 수 있다.

[0170] 디코더는 새로운 위치의 움직임 후보를 추가로 평가할 수 있다. 이하에서는 추가로 평가되는 움직임 후보의 새로운 위치에 대해 설명한다.

[0171] 현재 블록의 주변 블록으로부터 유도되는 움직임 정보들을 사용하여 유도된 가운데 위치의 움직임 후보가 추가적으로 평가될 수 있다. 즉, 디코더는 2개 이상의 움직임 후보들의 중앙 위치에 대응하는 움직임 후보를 추가적으로 평가할 수 있다.

[0172] 디코더는 현재 블록의 주변 블록으로부터 유도되는 움직임 정보들과 서로 다른 위치의 움직임 후보를 추가적으로 평가할 수 있다. 디코더는 초기 움직임 후보들 각각에 대한 탐색 범위를 모두 포함하는 더 큰 영역의 전역 탐색 범위(global search range)를 설정한 후, 전역 탐색 범위 내 비어있는 공간을 지시하는 움직임 후보에 대해 추가적으로 평가할 수 있다. 먼저, 디코더는 현재 블록의 주변 블록으로부터 유도되는 움직임 후보들이 지시하는 참조 블록들을 모두 포함하는 영역들을 전역 탐색 범위로 설정할 수 있다. 또는 디코더는 현재 블록의 좌상단 픽셀 위치와 현재 블록의 크기를 기준으로 미리 약속된 범위를 전역 탐색 범위로 설정될 수 있다. 전역 탐색 범위는 동일한 크기로 4등분되고, 각 사분면의 움직임 후보의 개수가 임의의 정해진 수보다 작을 경우, 임의의 정해진 수보다 작은 움직임 후보의 개수에 대응되는 사분면의 중앙 위치는 새로운 움직임 후보 위치로 설정되고, 디코더는 새로운 움직임 후보 위치의 움직임 후보에 대한 평가를 수행할 수 있다.

[0173] 현재 블록이 Merge 모드로 부호화된 경우, 디코더는 MMVD 방법으로 표현될 수 있는 모든 움직임 후보들에 대해 추가적인 평가를 수행할 수 있다 MMVD는 미리 정해진 움직임 차분 값들에 대한 거리 테이블과 수직 또는 수평 방향에 대한 부호 테이블을 정의하고, 차분 값의 거리에 대한 인덱스 정보와 방향에 대한 인덱스 정보를 시그널링하는 방법이다. MMVD를 통해 유도할 수 있는 모든 움직임 후보들이 TM에 사용될 수 있다. 이때, 별도의 시그널링 없이 움직임 후보들 중 가장 작은 코스트 값에 대응되는 움직임 후보가 현재 블록의 예측에 사용될 수 있다. 또는, 인코더는 MMVD를 통해 유도할 수 있는 모든 움직임 후보들을 사용하여 움직임 후보 리스트를 구성한 후, 움직임 후보 리스트 내의 움직임 후보들을 코스트 값에 기초하여 오름차순으로 재정렬한 후, 최적의 움직임 후보에 대한 인덱스에 대한 정보를 포함하는 비트스트림을 획득할 수 있다. 디코더는 비트스트림으로부터 인덱스에 대한 정보를 파싱하여 인덱스가 지시하는 움직임 후보를 통해 현재 블록을 예측(복원)할 수 있다.

[0174] 도 19는 본 발명의 일 실시예에 따른 현재 블록의 주변 블록으로부터 유도되는 초기 움직임 정보들이 서로 붙어 있는 경우를 나타내는 도면이다.

[0175] 초기 움직임 정보들(1901, 1902, 1903) 간에 근접한 경우 초기 움직임 정보들에 의해 설정되는 탐색 위치의 일부가 중복될 수 있다. 그러나 움직임 후보들이 탐색되는 위치가 보다 세분화되어 정밀한 움직임 후보들에 대한 평가가 수행될 수 있어, 정밀한 움직임 정보의 획득이 가능하다는 효과가 있다. 이는, 보다 정밀한 움직임 정보가 요구되는 환경(과정)에서 적용될 수 있다.

[0176] 초기 움직임 정보에 대한 분포는 현재 블록의 주변 블록의 움직임 정보에 따라 결정된다. 따라서, 디코더는 초기 움직임 정보들이 서로 붙어 있거나 너무 떨어져 있는 경우, 해당 초기 움직임 정보의 위치를 수정함으로써 탐색 범위를 조정할 수 있다. 이때, 움직임 정보들의 분포는 특정 값을 기준으로 판단될 수 있다. 예를 들어, 움직임 정보들 간 거리가 특정 값보다 큰 경우 움직임 정보들은 떨어져 있는 것으로 판단될 수 있고, 움직임 정보들 간 거리가 특정 값보다 작은 경우 움직임 정보들은 서로 붙어 있는 것으로 판단될 수 있다. 이때, 특정 값은 움직임 해상도에 따라 달라지는 값으로 정수 또는 소수의 값일 수 있다.

[0177] 도 20, 21은 본 발명의 일 실시예에 따른 초기 움직임 정보를 변경하는 방법을 나타내는 도면이다.

[0178] 도 20은 초기 움직임 정보들이 떨어져 있는 경우, 가까워지도록 초기 움직임 정보를 변경하는 일 예를 나타내고, 도 21은 초기 움직임 정보들이 가까이 있는 경우, 떨어지도록 초기 움직임 정보를 변경하는 일 예를 나타낸다. 도 20, 21을 참조하면 점선 화살표는 초기 움직임 후보(정보)(Initial MV Candidate 1, Initial MV Candidate 2)를 나타내고, 실선 화살표는 변경된 움직임 후보(정보)(Modified MV Candidate 1, Modified MV Candidate 2)를 나타낸다. 점선 원은 초기 움직임 후보에 따른 탐색 영역을 나타내고, 실선 원은 변경된 움직임 정보에 따른 탐색 영역을 나타낸다. 초기 움직임 정보의 수평 또는 수직 값을 조정하여 움직임 정보들을 가까워지도록 또는 떨어지도록 변경할 수 있다.

[0179] 도 22는 본 발명의 일 실시예에 따른 초기 움직임 정보를 통해 유도되는 움직임 후보가 탐색되는 위치를 나타내는 도면이다.

[0180] 디코더는 초기 움직임 정보를 기준으로 임의의 정해진 간격만큼 떨어진 위치의 움직임 후보를 탐색할 수 있다. 이때, 임의의 정해진 간격은 다양하게 설정될 수 있다. 수평 방향으로 임의의 정해진 간격만큼 변경된 위치의 움직임 후보가 탐색될 수 있고, 수직 방향으로 임의의 정해진 간격만큼 변경된 위치의 움직임 후보가 탐색될 수 있고 수평 및 수직 방향으로 임의의 정해진 간격만큼 변경된 위치의 움직임 후보가 탐색될 수 있다. 예를 들어, 도 22(a)는 초기 움직임 정보를 통해 유도되는 움직임 후보가 탐색되는 기본적인 위치를 나타낸다. 도 22(b)는 수직 방향으로 임의의 정해진 간격만큼 변경된 탐색되는 움직임 후보의 위치를 나타낸다. 도 22(b)를 참조하면 움직임 후보가 탐색되는 위치는 수직 방향으로 1/2 만큼 줄어들 수 있다. 도 22(c)는 수평 방향으로 임의의 정해진 간격만큼 변경된 탐색되는 움직임 후보의 위치를 나타낸다. 도 22(c)를 참조하면 움직임 후보가 탐색되는 위치는 수평 방향으로 1/2 만큼 줄어들 수 있다.

[0181] 도 23은 본 발명의 일 실시예에 따른 움직임 후보가 탐색되는 탐색 위치를 변경하는 방법을 나타내는 도면이다.

[0182] 도 23을 참조하면 현재 블록의 주변 블록으로부터 유도된 움직임 정보(Initial MV Candidate 1, Initial MV Candidate 2)를 사용하여 움직임 후보가 탐색되는 위치는 적응적으로 설정될 수 있다. 움직임 후보의 탐색을 위한 탐색 간격, 탐색 위치는 유도된 움직임 정보들이 서로 직선으로 연결되는 지점들의 위치로 변경될 수 있다. 디코더는 첫 번째 초기 움직임 정보에 대한 제1 탐색 범위를 수직 방향 및 수평 방향 중 적어도 어느 하나의 방향으로 줄여서 움직임 후보가 탐색되는 위치를 변경할 수 있다. 그리고 디코더는 두 번째 움직임 정보에 대한 제2 탐색 범위를 회전하여 제2 탐색 범위 내 움직임 후보가 탐색되는 위치를 첫 번째 초기 움직임 정보에 대한 변경된 움직임 후보 탐색 위치와 일직선 상에 위치하도록 변경할 수 있다. 이후, 디코더는 변경된 움직임 후보 탐색 위치들의 움직임 후보를 평가할 수 있다. 마찬가지로, 디코더는 두 번째 초기 움직임 정보에 대한 제2 탐색 범위를 수직 방향 및 수평 방향 중 적어도 어느 하나의 방향으로 줄여서 움직임 후보가 탐색되는 위치를 변경할 수 있다. 그리고 디코더는 첫 번째 움직임 정보에 대한 제1 탐색 범위를 회전하여 제1 탐색 범위 내 움직임 후보가 탐색되는 위치를 두 번째 초기 움직임 정보에 대한 변경된 움직임 후보 탐색 위치와 일직선 상에 위치하도록 변경할 수 있다. 이후, 디코더는 변경된 움직임 후보 탐색 위치들의 움직임 후보를 평가할 수 있다. 움직임 후보가 탐색되는 위치가 탐색 범위 내에 있더라도, 픽쳐 밖에 위치하는 경우 픽쳐 밖에 위치하는 움직임 후보는 평가되지 않는다. 탐색 범위가 변경(줄이거나 늘리는 경우)되는 경우 움직임 후보가 탐색되는 위치는 움직임 해상도와 일치하지 않을 수 있다. 이 경우, 가까운 움직임 해상도와 일치되도록 움직임 후보가 탐색되는 위치가 조정될 수 있다.

[0183] TM은 현재 블록에 인접한 주변 화소(픽셀)들을 사용하여 획득되는 코스트 값을 이용하므로 현재 블록의 주변 블록들이 복원되어 있어야 한다. 따라서, 현재 블록과 주변 블록들이 서로 의존적이므로 병렬적으로 TM이 처리될 수 없어 처리 속도에 불리하다는 문제점이 있다. 이러한 문제점을 해결하기 위해 현재 블록과 주변 블록 간의 의존성을 제거할 필요가 있다. 구체적으로 의존성을 제거하기 위해 디코더는 현재 블록의 주변 화소(픽셀)들을 이용하여 코스트 값을 획득하지 않고, 주변 블록들의 움직임 정보를 화소 단위로 유도하여 움직임 정보를 이용하여 코스트 값을 획득할 수 있다. 이 경우, 주변 블록들의 움직임 정보는 디코더가 비트스트림을 파싱함에 따라 병렬적으로 복원될 수 있어, 블록 간 병렬적으로 복원이 가능하다는 효과가 있다. 즉 디코더는 병렬적으로 TM을 이용한 블록 복원을 수행할 수 있다.

[0184] 도 24는 본 발명의 일 실시예에 따른 움직임 벡터의 코스트 값에 기초한 TM을 나타내는 도면이다.

[0185] 도 24를 참조하면 현재 블록의 주변 블록으로부터 유도된 초기 움직임 정보(Initial MV Candidate)를 사용하여 움직임 후보의 탐색을 위한 탐색 영역과 위치가 설정될 수 있다. 움직임 벡터의 코스트 값을 획득하기 위해 초기 움직임 정보가 지시하는 참조 블록 주변의 움직임 정보들을 이용하여 화소(픽셀) 단위 optical flow MV 맵이 생성될 수 있다. 현재 블록에 인접한 주변 템플릿에 대해서도 주변 움직임 정보들을 통해 화소(픽셀) 단위 optical flow MV 템플릿이 생성될 수 있다. 디코더는 움직임 후보의 탐색을 위한 위치들 중에서 현재 블록 MV 템플릿과 가장 유사한 optical flow MV 참조 템플릿을 구해서 보정된 움직임 정보로 사용할 수 있다. MV 템플릿 간의 코스트 값은 다양한 방법으로 획득될 수 있다. 예를 들어, MV 템플릿 간의 코스트 값은 수평 또는 수직 방향의 MV 차이의 절대합을 통해 계산될 수 있다. 복잡도를 감소시키기 위해서 MV 템플릿을 화소 단위가 아닌 2x2 또는 4x4 단위의 MV 템플릿이 유도될 수 있다.

[0186] 이하에서 DMVR을 사용하여 움직임 정보를 보정하는 방법에 대해 설명한다.

[0187] 도 25, 도 26은 본 발명의 일 실시예에 따른 DMVR을 사용하여 움직임 정보를 보정하는 방법을 나타내는 도면이다.

[0188] DMVR은 BM(Bilateral Matching) 방법을 사용하여 현재 블록에 대한 보정된 움직임 정보를 획득하는 방법이다. BM(Bilateral Matching) 방법은 양방향 움직임을 가지는 블록에서 L0 참조 블록의 주변 탐색 영역(Search area)과 L1 참조 블록의 주변 탐색 영역에서 서로 간의 가장 유사한 부분을 찾아서 초기 움직임 정보를 보정하고, 보정된 움직임 정보를 현재 블록의 예측에 사용하는 방법이다. 탐색 영역의 크기는 참조 블록의 특정 지점을 기준으로 임의의 (m x n)크기로 설정될 수 있다. 예를 들어, 특정 지점은 참조 블록의 좌상단 위치 또는 참조 블록의 중앙 위치일 수 있고, 임의의 크기는 16 x 16일 수 있다. 가장 유사한 부분은 각 블록 간 화소 단위의 코스트 값을 계산하여 가장 작은 코스트 값에 대응되는 지점일 수 있다. 코스트 값은 SAD(Sum of Absolute Differences) 혹은 MRSAD(Mean-Removed SAD)를 이용하여 계산될 수 있다. 탐색 영역에 따라 코스트 값은 달라질 수 있고, 보정된 움직임 정보 또한 달라질 수 있다. 디코더는 현재 블록을 복수 개의 서브 블록으로 분할하여 각 서브 블록에 대하여 DMVR을 사용한 움직임 정보를 보정할 수 있다. 크기가 큰 블록들 대비 크기가 작은 블록들에 대한 움직임 정보가 더 정확하기 때문이다. 이때, 크기가 큰 블록에서는 DMVR이 수행되지 않고, 분할된 작은 블록(예, 서브 블록)에서만 DMVR이 수행될 수 있다. 도 26을 참조하면 하나의 블록은 4개의 서브 블록으로 분할될 수 있다. 디코더는 분할된 각 서브 블록마다 DMVR을 사용하여 보정된 움직임 정보를 획득할 수 있다. 또는 디코더는 큰 블록에서 DMVR을 통해 찾은 보정된 움직임 정보를 사용하여 분할된 작은 블록에서 DMVR을 통해 보다 정확한 움직임 정보를 유도하는 다중 DMVR 방법을 사용할 수 있다.

[0189] 이하에서 다중 DMVR 방법에 대해 설명한다.

[0190] 디코더는 현재 코딩 블록에 대한 파싱(parsing) 단계 후, 현재 코딩 블록에 대한 디코딩을 시작하기 전에, 현재 코딩 블록에 대한 움직임 정보를 유도하는 단계에서 후술하는 조건들을 판단하여 다중 DMVR 보정 수행 여부를 결정할 수 있다. 다중 DMVR 수행 조건은 i) merge 모드의 움직임이 양방향인 경우, ii) 참조 픽쳐들과 현재 픽쳐 간의 거리가 동일한 경우, iii) merge 모드의 움직임 차분 값이 존재하는 경우, iv) 참조 블록들 간의 가중치 예측이 적용되지 않은 블록인 경우일 수 있다. 디코더는 i) 내지 iv) 조건 중 어느 하나라도 만족하는 경우, 다중 DMVR을 수행할 수 있다.

[0191] 만일, 현재 코딩 블록의 머지 후보 리스트 내의 제1 머지 움직임 후보와 다른 머지 후보 간의 움직임 차이가 임의의 한계치보다 작은 경우 다중 DMVR은 수행되지 않을 수 있다. 또한, 머지 후보 리스트 내의 제1 머지 움직임 후보와 다른 머지 움직임 후보가 복수 개인 경우, 움직임 차이가 임의의 한계치보다 작은 경우가 하나라도 있으면, 다중 DMVR은 수행되지 않을 수 있다. 이때, 머지 후보 리스트의 머지 후보들 중 현재 코딩 블록의 머지 움직임 후보를 지시하는 인덱스보다 낮은 인덱스로 지시되는 머지 후보만 움직임 차이를 비교할 수 있다. 복잡도를 감소하기 위함이다. 즉, 머지 후보 리스트 내의 머지 움직임 후보들 각각과 현재 코딩 블록의 머지 후보와의 움직임 차이를 계산하고 계산된 움직임 차이들 전부가 임의의 지정된 한계치보다 큰 경우 디코더는 다중 DMVR을 수행할 수 있다. 현재 코딩 블록의 머지 움직임 후보와 머지 후보 리스트 내 머지 움직임 후보들 간의 유사성을 비교하는 것은 주변 블록들이 다양한 움직임을 가지는지 판단하기 위함이다. 일반적으로 주변 블록의 움직임이 유사할 경우, 현재 블록의 움직임도 주변 블록의 움직임과 유사할 가능성이 높다. 이때에는 주변 블록과 현재 블록은 동일한 움직임을 가지는 객체 내의 블록으로 유추될 수 있다. 반대로, 주변 블록의 움직임이 유사하지 않고 서로 다른 움직임 특성을 나타내는 경우, 현재 코딩 블록을 포함한 주변 블록들은 동일 객체가 아닐 수 있으므로, 보다 정확한 움직임을 찾는 과정이 필요할 수 있다.

[0192] 도 27은 본 발명의 일 실시예에 따른 다중 DMVR이 수행되는 과정을 나타내는 도면이다.

[0193] 도 27(a)는 다중 DMVR이 수행되는 일반적인 과정을 나타내고, 도 27(b)는 다중 DMVR의 일반적인 과정을 보다 구체적으로 나타낸다.

[0194] 도 27(a)를 참조하면, 다중 DMVR은 초기 움직임 정보에 기초하여 코딩 유닛(블록) 단위의 DMVR을 수행하여 하나 이상의 보정된 움직임 정보를 획득할 수 있다(S2710). 현재 코딩 블록에 TM이 적용되는 경우, 디코더는 S2710 단계에서 획득한 하나 이상의 보정된 움직임 정보를 이용하여 TM을 수행할 수 있다(S2720). TM을 수행하여 결정되는 현재 코딩 블록의 보정된 움직임 정보가 양방향에서 단방향으로 변경되었다면, DMVR 과정을 수행할 수 없으므로 S2720 이후의 단계들(S2730, S2740)은 수행되지 않고, 현재 블록의 움직임은 단방향으로 최종 결정된다. TM을 수행한 결과, 현재 코딩 블록의 보정된 움직임 정보가 양방향이라면, 디코더는 서브 블록 단위의 DMVR을 수행하여 각 서브 블록마다 서브 블록 단위의 보정된 움직임 정보를 획득할 수 있다(S2730). 그리고 디코더는 S2740 단계에서 획득한 서브 블록 단위의 보정된 움직임 정보를 BDOF에 기반하여 다시 보정하고, BDOF에 기반하

여 보정된 움직임 정보를 최종적으로 획득할 수 있다.

[0195] 한편, S2710 단계에서 획득된 하나 이상의 보정된 움직임 정보들에 대한 코스트 값들 중 가장 작은 코스트 값이 임의의 값보다 크다면 S2730 단계가 수행될 수 있다. 한편, 가장 작은 코스트 값이 임의의 값보다 같거나 작다면 디코더는 S2710 단계에서 획득한 하나 이상의 보정된 움직임 정보들을 서브 블록 단위로 저장하고, 현재 코딩 블록의 움직임 정보를 초기 움직임 정보로 재설정하여 S2710 내지 S2740 단계를 수행할 수 있다. 이때, 서브 블록 단위로 저장된 하나 이상의 보정된 움직임 정보들은 S2740 단계에서의 움직임 보정을 위해 사용될 수 있다. 최종적으로 초기 움직임 정보를 사용한 코딩 블록의 코스트 값과 S2740 단계를 수행하여 획득되는 서브 블록 단위로 보정된 움직임을 사용한 코딩 블록의 코스트 값을 비교하여 작은 코스트 값을 가지는 블록 단위와 그에 대한 움직임 정보가 최종적으로 현재 코딩 블록에 사용될 수 있다. 임의의 값은 현재 코딩 블록 내 화소(픽셀)의 개수 또는 현재 코딩 블록의 크기(가로 또는 세로 중 어느 하나의 크기 또는 가로 및 세로의 합)일 수 있다.

[0196] 도 27(b)를 참조하면, 도 27(a)의 S2710 단계는 정수 단위의 탐색을 이용하여 코딩 유닛(블록) 단위의 DMVR을 수행하는 단계(S2701)과 반화소 단위의 탐색을 이용하여 코딩 유닛(블록) 단위의 DMVR을 수행하는 단계(S2702)로 세분화 될 수 다. 이때, 움직임 해상도는 S2701 단계에서는 정수 단위로, S2702 단계에서는 반화소 단위로 설정될 수 있다. S2701 단계에서 디코더는 초기 움직임 정보와 정수 단위 보정 값을 사용하여 보정된 움직임 정보를 획득할 수 있다. S2702 단계에서 디코더는 초기 움직임 정보와 반화소 단위 보정 값을 사용하여 보정된 움직임 정보를 획득할 수 있다. 디코더는 S2701 단계에서 획득된 보정된 움직임 정보를 S2702 단계를 위한 기준 움직임 정보로 사용할 수 있다. 즉, 디코더는 S2701 단계에서 획득된 보정된 움직임 정보에 기초하여 새로운 움직임 후보를 생성하고, 새로운 움직임 후보에 대한 평가를 수행할 수 있다. 이때, 새로운 움직임 후보에 대한 코스트 값이 S2701 단계에서 획득된 보정된 움직임 정보의 코스트 값보다 크다면 S2701 단계에서 획득된 보정된 움직임 정보가 코딩 유닛(블록) 단위 DMVR의 출력이 될 수 있다. 반대로, 새로운 움직임 후보에 대한 코스트 값이 S2701 단계에서 획득된 보정된 움직임 정보의 코스트 값보다 작다면 S2702 단계에서 획득된 보정된 움직임 정보가 코딩 유닛(블록) 단위 DMVR의 출력이 될 수 있다. 즉 S2701 단계에서 획득한 코스트 값과 S2702 단계에서 획득한 코스트 값 중 작은 코스트 값에 대응되는 보정된 움직임 정보를 TM(S2703, S2720)을 위해 사용할 수 있다. 이때, 새로운 움직임 후보 전부에 대한 코스트 값을 계산하는 경우, 복잡도가 증가하는 문제가 있다. 이를 위해, 디코더는 MV(움직임 정보) 기반 코스트 값을 계산하고, 움직임 정보에 기초하여 획득되는 코스트 값이 임의의 값보다 작은 경우에만 새로운 움직임 후보에 대한 화소(픽셀) 기반 코스트 값을 계산할 수 있다. 움직임 정보에 기초하여 획득되는 코스트 값은 새로운 움직임 후보와 초기 움직임 정보 간의 수평 및 수직 성분 값의 차이에 대한 절대값의 합에 가중치를 곱해서 획득될 수 있다. 또한 움직임 정보에 기초하여 획득되는 코스트 값은 새로운 움직임 후보와 S2701 단계에서 획득된 보정된 움직임 정보 간의 차이를 이용하여 획득될 수 있다.

[0197] 움직임 정보에 기초하여 획득되는 코스트 값들 각각은 초기 움직임 정보에 따라 선정되는 위치에 대응되는 복수의 움직임 후보들 각각에 보정 값을 더하여 획득되는 움직임 정보들 간의 차분 값을 통해 획득될 수 있다. 움직임 정보에 기초하여 획득되는 코스트 값은 보정 값의 크기에 따라 달라질 수 있다. 즉, 보정 값이 작을수록 코스트 값은 작아질 수 있다. 가장 작은 코스트 값에 대응되는 움직임 정보가 최종 움직임 정보로 설정되기 때문에, 보정 값이 작은 초기 움직임 정보가 나타내는 위치의 주변 움직임 후보에 대해서만 평가가 수행될 수 있다. 그러나, 보정 값이 큰 움직임 후보가 최적의 움직임 후보일 수 있다. 따라서, 다양한 움직임 후보에 대한 평가를 수행하여 최적의 움직임 후보를 선정하기 위해, 코스트 값은 후술하는 방법을 이용하여 획득될 수 있다. 코스트 값은 주변 블록 각각의 움직임 정보 값 간의 차이, 양자화 파라미터, 현재 블록의 크기 등을 이용하여 획득될 수 있다.

[0198] 코스트 값은 주변 블록의 움직임 정보의 분포를 이용하여 획득될 수 있다. 다양한 움직임 후보에 대한 평가를 위해, 디코더는 보정된 움직임 정보와 주변 블록의 움직임 정보 간의 차이 값을 이용하여 코스트 값을 획득할 수 있다. 예를 들어, 보정된 움직임 정보와 주변 블록의 움직임 정보 간의 차이 값과 임의의 정해진 값을 비교하여 비교 결과에 따라 코스트 값을 획득할 수 있다. 구체적으로 보정된 움직임 정보와 주변 블록의 움직임 정보 간의 차이 값이 임의의 정해진 값보다 큰 경우(또는 작은 경우, 같은 경우) 디코더는 코스트 값을 획득할 수 있다. 이때 주변 블록은 현재 블록에 인접한 주변 블록이거나 또는 대응 픽처(Collocated picture)에서 현재 블록과 동일한 위치에 있는 시간적인 주변 블록일 수 있다.

[0199] 코스트 값은 현재 블록의 크기에 기초하여 획득될 수 있다. 예를 들어, 상술한 코스트 값을 획득하기 위한 가중치는 현재 블록의 크기에 따라 설정될 수 있다. 가중치는 현재 블록의 크기와 반비례하게 설정될 수 있다. 즉, 현재 블록의 크기가 클수록 가중치는 낮게 설정될 수 있다. 이는, 적합한 움직임 후보를 선정하기 위해 보다 더

넓은 범위의 움직임 후보를 평가하기 위함이다. 한편 가중치는 현재 블록의 크기와 비례하게 설정될 수 있다. 즉, 현재 블록의 크기가 클수록 가중치는 높게 설정될 수 있다. 이는 복잡도를 낮추기 위함이다. 예를 들어, 현재 블록의 크기는 16x16, 32x32가 될 수 있고, 현재 블록의 가로 및 세로 크기의 합으로 설정될 수 있다. 가중치는 1, 2, 3, 4, 5, 6, 등의 정수 값이 될 수 있다. 또한 가중치가 커질수록 코스트 값이 높아지므로 가중치가 특정 값 이상인 경우 디코더는 코스트 값을 획득하기 위한 평가를 수행하지 않을 수 있다.

[0200] 이하에서 도 27(a)의 S2730 단계에 대해 자세히 설명한다.

[0201] S2730 단계에서 디코더는 현재 코딩 블록을 여러 개의 서브 블록으로 분할한 후, 각 서브 블록마다 도 27(b)의 S2704, S2705 단계가 수행될 수 있다. 서브 블록의 너비는 코딩 블록의 너비와 기 설정된 최대 너비와 비교하여 작은 값이 서브 블록의 너비가 될 수 있다. 서브 블록의 높이는 코딩 블록의 높이와 기 설정된 최대 높이와 비교하여 최소값이 서브 블록의 높이가 될 수 있다. 이때 서브 블록 크기는 최대 16x16으로 설정될 수 있다.

[0202] 디코더는 S2730 단계에서 획득한 보정된 움직임 정보를 S2704, S2705 단계를 위한 초기 움직임 정보로 설정할 수 있다. 디코더는 초기 움직임 정보를 이용하여 정수 단위의 전역 탐색을 수행하고, 현재 서브 블록에 대한 최적의 움직임 정보와 최적의 움직임 정보에 대한 코스트 값을 획득할 수 있다(S2704). 최적의 움직임 정보를 획득하기 위해 서브 블록에 대한 움직임 정보들 각각에 대한 코스트 값이 획득될 수 있다. 임의의 서브 블록에 대한 코스트 값이 임의의 값보다 작다면, 정수 단위의 전역 탐색에서 획득된 현재 서브 블록에 대한 움직임 정보가 저장되고 이후 과정은 수행되지 않을 수 있다. 만일 임의의 서브 블록에 대한 코스트 값이 임의의 값보다 같거나 크다면 임의의 서브 블록에 대한 코스트 값은 기 설정된 다른 값으로 설정될 수 있다. 임의의 값은 현재 서브 블록의 픽셀 수 또는 현재 서브 블록의 크기가 될 수 있다. 예를 들어, 서브 블록의 크기는 16 x 16일 수 있다. 또한 기 설정된 다른 값은, 인코더 및 디코더가 표현할 수 있는 수의 범위에서의 최대 값일 수 있다. S2704단계 이후, 디코더는 반화소(1/2) 단위의 3x3 Square 탐색을 수행할 수 있다. S2704 단계를 통해 획득되는 움직임 정보는 S2705 단계의 기준 움직임 정보로 사용될 수 있다. 즉, S2704 단계를 통해 획득된 정보에 기초하여 새로운 움직임 후보가 획득되고, 디코더는 새로운 움직임 후보를 평가할 수 있다(S2705). 디코더는 새로운 움직임 후보에 대한 평가를 수행하고, 최종 움직임 정보를 현재 서브 블록에 대해 저장할 수 있다. S2704, S2705 단계는 모든 서브 블록에 대해 반복되어 수행될 수 있다. 서브 블록 단위의 DMVR 과정은 서브 블록 간의 의존성이 없으므로, 모든 서브 블록이 병렬적으로 DMVR을 수행할 수 있다는 장점이 있다.

[0203] 도 28은 본 발명의 일 실시예에 따른 코딩 블록의 보정된 움직임 정보와 관련된 코스트 값을 획득하기 위한 탐색 방법을 나타내는 도면이다.

[0204] 도 28을 참조하면 디코더는 초기 움직임 정보를 획득하고, 획득된 초기 움직임 정보에 기초하여 탐색할 움직임 후보들을 설정할 수 있다. 그리고 디코더는 설정된 움직임 후보들에 대한 평가를 수행하여 획득되는 코스트 값에 기초하여 보정된 움직임 정보을 획득할 수 있다. 그리고 디코더는 현재 블록의 움직임 정보 해상도에 따라 모델 기반 fractional MVD 최적화를 이용하여 최종적인 보정된 움직임 정보를 획득할 수 있다. 도 27(b)의 S2701, S2702, S2705 단계에서 획득되는 보정된 움직임 정보는 도 28과 도 29를 통해 설명한 방법을 통해 획득될 수 있다.

[0205] 도 29는 본 발명의 일 실시예에 따른 3x3 Square 탐색 방법을 나타내는 도면이다.

[0206] 이하에서 도 29를 참조하여 도 28을 통해 설명한 최종적인 보정된 움직임 정보를 획득하는 방법을 보다 구체적으로 설명한다.

[0207] 디코더는 이전 단계에서 보정된 움직임 정보에 대한 코스트 값을 획득(계산)할 수 있다(S2901). 이때, 기존에 획득한 보정된 움직임 정보에 대한 코스트 값이 존재한다면 S2901 단계는 생략될 수 있다. S2901 단계에서 획득되는 코스트 값은 움직임 해상도에 따라 다른 방법으로 획득될 수 있다. 예를 들어, 움직임 해상도가 정수 단위라면, 보정된 움직임 정보에 대한 화소(픽셀) 기반 코스트 값으로 획득될 수 있으며 기준 코스트 값으로 사용된다. 움직임 해상도가 반화소(1/2) 단위라면, 디코더는 초기 움직임 정보와 보정된 움직임 정보를 사용하여 획득되는 움직임 정보에 기반한 코스트 값을 획득하고 보정된 움직임 정보에 대한 화소 기반 코스트 값을 획득한다. 다음으로 움직임 정보에 기반한 코스트 값이 화소(픽셀) 기반으로 계산된 코스트 값보다 크다면, 보정된 움직임 정보가 저장되고 다음 과정은 수행되지 않고 생략된다. 만일 움직임 정보에 기반한 코스트 값이 화소(픽셀) 기반으로 계산된 코스트 값보다 같거나 작다면, 화소(픽셀) 기반 코스트 값에 움직임 정보에 기반한 코스트 값을 더한 코스트 값이 기준 코스트 값으로 사용된다. 디코더는 탐색할 움직임 후보들에 대한 보정 값을 설정할 수 있다(S2902). 구체적으로, 정해진 3x3 Square 탐색 패턴에 따라 움직임 보정 후보 값은 유도될 수 있다. 이때

움직임 보정 후보 값은 움직임 해상도에 기초하여 설정될 수 있다. 움직임 해상도에 따라 움직임 보정 후보 값은 스케일되어 범위가 확장될 수 있다. 예를 들어, 정해진 3x3 Square 탐색 패턴의 수평 및 수직 방향은 Mv(-1, 1), Mv(0, 1), Mv(1, 1), Mv(1, 0), Mv(1, -1), Mv(0, -1), Mv(-1, -1), Mv(-1, 0)일 수 있다. 움직임 해상도가 반화소(1/2) 단위일 경우, 계산 복잡도를 감소시키기 위해 3x3 Square 탐색 패턴에서 홀수 번째만 평가가 수행되고, 짝수 번째는 평가가 수행되지 않을 수 있다. 디코더는 보정된 움직임 후보와 움직임 보정 후보 값을 이용하여 새로운 움직임 후보를 설정할 수 있다. 디코더는 L0 픽처리스트 내 픽처의 참조 블록에 대한 움직임 정보와 L1 픽처리스트 내 픽처의 참조 블록에 대한 움직임 정보를 재구성할 수 있다(S2904). 선형적인 특성을 가지는 L0 및 L1 움직임을 평가하기 위해서, 디코더는 보정된 L0 움직임 후보에는 움직임 보정 후보 값을 합산하고, 보정된 L1 움직임 후보에는 움직임 보정 후보 값을 차분하여 새로운 움직임 후보를 생성할 수 있다. 또한, 비선형적인 특성을 지니는 L0 및 L1 움직임을 평가하기 위해서, 디코더는 보정된 L0 움직임 후보는 수정하지 않고, 보정된 L1 움직임 후보에는 움직임 보정 후보 값을 합산하여 새로운 움직임 후보를 생성할 수 있다. 또는, 디코더는 보정된 L1 움직임 후보는 수정하지 않고, 보정된 L0 움직임 후보에는 움직임 보정 후보 값을 합산하여 새로운 움직임 후보를 생성할 수 있다. 디코더는 새로운 움직임 후보에 대한 움직임 정보에 기초한 코스트 값을 계산할 수 있다(S2905). 디코더는 새로운 움직임 후보에 대한 움직임 정보에 기초한 코스트 값을 기준 코스트 값과 비교할 수 있다(S2906). 움직임 정보에 기초한 코스트 값이 기준 코스트 값보다 큰 경우, 디코더는 움직임 정보에 기초한 코스트 값의 2배에 해당하는 값을 메모리에 저장할 수 있다. 그리고 디코더는 S2906 단계 이후의 과정은 수행하지 않고 S2902 단계부터 반복하여 다시 수행할 수 있다. 움직임 정보에 기초한 코스트 값이 기준 코스트 값보다 작은 경우, 디코더는 새로운 움직임 후보에 대한 화소(픽셀) 기반 코스트 값을 획득할 수 있다(S2907). 그리고, 기준 코스트 값을 움직임 정보에 기초한 코스트 값과 새로운 움직임 후보에 대한 화소 기반 코스트 값을 합산한 값으로 갱신할 수 있다. 만약 움직임 해상도가 반화소(1/2) 단위일 경우, 디코더는 다른 새로운 움직임 후보를 평가하기 위해서 S2901 단계부터 다시 반복하여 수행할 수 있다. 즉, 움직임 해상도가 반화소(1/2) 단위일 경우에는 디코더는 새로운 움직임 후보에 대한 평가 결과만 저장하고 최적의 움직임 후보로는 사용하지 않을 수 있다. 저장된 새로운 움직임 후보에 대한 평가 결과(코스트 값)는 이후에 수행될 수 있는 모델 기반 fractional MVD 최적화 단계에서 사용될 수 있다. 만일 움직임 해상도가 정수 단위일 경우, 새로운 움직임 후보에 대한 움직임 정보에 기초한 코스트 값과 화소(픽셀) 기반 코스트 값을 합산한 값이 초기 보정된 움직임 정보에 대한 화소(픽셀) 기반 코스트 값보다 작다면, 디코더는 새로운 움직임 후보를 최적의 움직임 정보로 저장할 수 있다. 만약 움직임 해상도가 정수 단위일 경우, 디코더는 S2907 단계에서 획득한 새로운 움직임 후보에 대한 화소(픽셀) 기반 코스트 값을 최적의 움직임 정보에 대한 코스트 값으로 저장할 수 있다. 이후, 디코더는 아직 탐색하지 못한 움직임 후보가 있는지 확인할 수 있다(S2908). 디코더는 아직 탐색하지 못한 움직임 후보가 있다면 새로운 움직임 후보를 평가하기 위해서 S2901 단계를 반복하여 수행할 수 있고, 모든 움직임 후보에 대해 평가를 하였다면 탐색한 움직임 후보에 대한 코스트 값들 중 가장 작은 코스트 값에 대응하는 움직임 후보를 최종 보정 움직임으로 저장할 수 있다(S2909). 디코더는 움직임 해상도가 반화소(1/2) 단위일 경우, 이전 단계에서 저장된 움직임 후보들에 대한 코스트 값들을 사용하여 모델 기반 fractional MVD 최적화 과정이 수행되어 최적의 반화소 단위의 보정 움직임 정보를 다시 계산할 수 있다(S2910). S2910 단계는 VVC(Versatile Video Coding)의 DMVR에서 정수 단위의 화소(픽셀) 기반 코스트 값을 사용하여 소수 화소 단위의 움직임을 유도하는 과정과 유사할 수 있다. 디코더는 최적 움직임 정보의 위치에 인접한 주변(좌, 우, 위, 아래) 움직임 정보에 대한 코스트 값을 사용하여 최적 움직임 정보를 획득할 수 있다.

[0208] 도 29를 참조하여 설명한 탐색 방법은 임의의 정해진 횟수만큼 반복 수행될 수 있고, 반복 수행될 때, 디코더는 이전 탐색에서 획득한 최적 움직임 정보의 주변 위치만을 탐색할 수 있다.

[0209] 도 30은 본 발명의 일 실시예에 따른 정수 단위 전역 탐색 과정에서 탐색 영역을 구역별로 나누어 진 것을 나타낸 도면이다.

[0210] 도 30을 참조하면, 정수 단위 전역 탐색은 탐색 영역을 가운데 위치를 중심으로 여러 개의 구역으로 나누고, 구역별로 서로 다른 가중치를 설정해서 수행되는 탐색일 수 있다. 탐색 영역의 가운데 위치는 초기 움직임 정보가 지시하는 참조 블록의 좌상단 화소(픽셀) 위치일 수 있다. 각 구역별 가중치는 현재 블록의 주변 블록 및 현재 블록의 움직임 특성에 따라 다르게 설정될 수 있다. 예를 들어, 현재 블록의 주변 블록 및 현재 블록의 움직임 간 유사성이 높을 경우, 가중치는 가운데 구역부터 가장자리 구역 순으로 높아지도록 설정될 수 있다. 반대로, 현재 블록의 주변 블록 및 현재 블록의 움직임 간 유사성이 낮을 경우, 가중치는 가운데 구역부터 가장자리 구역 순으로 낮아지도록 설정될 수 있다. 도 27(a)의 S2710 단계에서 획득되는 보정된 움직임 정보에 대한 화소(픽셀) 기반 코스트 값이 임의의 값보다 작다면, 정수 단위 전역 탐색은 수행되지 않을 수 있다. 또한, 탐색 영역 내에서 임의의 움직임 보정 값에 대한 화소(픽셀) 기반 코스트 값이 임의의 값보다 작다면, 임의의 움직임

보정 값이 최종 보정된 움직임 정보가 되고, 정수 단위 전역 탐색은 종료될 수 있다. 임의의 값은 현재 블록의 화소(픽셀) 수 또는 현재 블록의 크기가 될 수 있다. 예를 들어, 현재 블록의 크기는 16x16 일 수 있다.

[0211] 도 31, 도 32는 본 발명의 일 실시예에 따른 정수 단위의 전역 탐색 과정을 나타내는 도면이다.

[0212] 도 31을 참조하면 디코더는 입력된 초기 움직임 정보에 기초하여 탐색할 움직임 후보들을 설정할 수 있다. 그리고 디코더는 움직임 후보들에 대한 평가를 수행하여 최종 보정 움직임 정보를 획득할 수 있다. 정수 단위의 전역 탐색은 도 30에 도시된 바와 같이 가운데 구역부터 가장자리 구역 순으로, 각 구역별 모든 위치에 대하여 수행될 수 있다.

[0213] 이하에서 도 32를 참조하여 정수 단위의 전역 탐색에 대해 보다 자세히 설명한다. 도 32를 참조하면 디코더는 블록의 크기에 따라 탐색 영역 내의 각 구역별 탐색 위치를 재설정할 수 있다(S3201). 이때, 블록의 크기가 임의의 값보다 작다면, 디코더는 계산 복잡도 감소를 위해 도 30의 가운데 위치를 기준으로 임의의 값만큼 축소된 탐색 영역만을 평가한다. 임의의 값은 현재 블록의 크기의 절반이거나 양의 정수가 될 수 있으며, 예를 들어, '8'일 수 있다. 초기 움직임 정보를 사용하여 화소(픽셀) 기반 코스트 값을 계산한 후, 그것을 최소 코스트 값으로 설정한다. 디코더는 움직임 후보의 탐색 위치에 대한 움직임 보정 값을 설정하고, 움직임 보정 값을 현재 블록에서 사용될 움직임 해상도(resolution)에 맞게 재설정할 수 있다(S3202, S3203). 디코더는 S3203 단계에서 재설정된 움직임 보정 값을 이용하여 새로운 움직임 후보를 설정할 수 있다. 디코더는 L0 픽처리스트 내 픽처의 참조 블록에 대한 움직임 정보와 L1 픽처리스트 내 픽처의 참조 블록에 대한 움직임 정보를 재구성할 수 있다(S3204). 선형적인 특성을 지니는 L0 및 L1 움직임을 평가하기 위해서, 디코더는 보정된 L0 움직임 후보에는 움직임 보정 후보 값을 합산하고, 보정된 L1 움직임 후보에는 움직임 보정 후보 값을 차분하여 새로운 움직임 후보를 생성할 수 있다. 또한 비선형적인 특성을 지니는 L0 및 L1 움직임을 평가하기 위해서, 디코더는 보정된 L0 움직임 후보는 수정하지 않고, 보정된 L1 움직임 후보에는 움직임 보정 후보 값을 합산하여 새로운 움직임 후보를 생성할 수 있다. 또한 디코더는 보정된 L1 움직임 후보는 수정하지 않고, 보정된 L0 움직임 후보에는 움직임 보정 후보 값을 합산하여 새로운 움직임 후보를 생성할 수 있다. 디코더는 새로운 움직임 후보에 대한 화소(픽셀) 기반 코스트 값을 획득할 수 있다(S3205). 각 구역별로 코스트 값을 차별화하기 위해 디코더는 미리 정의된 각 구역별 가중치 값을 사용하여 코스트 값을 재획득할 수 있다(S3206). 가중치 값은 실험을 통해 구해진 값으로 양의 정수 값이 될 수 있다. 예를 들어, 가중치 값은 '1', '2', … , '63'일 수 있다. 디코더는 재획득된 코스트 값이 최소 코스트 값보다 작다면, 최소 코스트 값을 재획득된 코스트 값으로 갱신할 수 있다. 디코더는 모든 움직임 후보에 대한 평가가 완료되었는지 여부를 평가할 수 있다(S3207). 아직 평가가 완료되지 않은 움직임 후보가 있다면 최소 코스트 값을 이용하여 S3202 단계부터 다시 수행할 수 있다. 모든 움직임 후보에 대한 평가가 완료되었다면 디코더는 움직임 후보들 각각의 코스트 값 중 가장 작은 코스트 값에 대응되는 움직임 후보의 정보를 최종 보정된 움직임 정보로 설정할 수 있다. 정수 단위의 전역 탐색은 복잡도를 낮추기 위해서 조기 종료 방법이 적용될 수 있다. S3206 단계에서 재획득된 코스트 값이 최소 코스트 값보다 작고, 재획득된 코스트 값에서 최소 코스트 값을 뺀 차이 값이 임의의 값보다 작을 경우, S3207 단계를 검사하지 않고 S3208 단계가 수행되고 정수 단위의 전역 탐색이 종료될 수 있다. 임의의 값은 현재 블록의 크기 혹은 픽셀 수가 될 수 있으며, 예를 들어, 16x16 일 수 있다.

[0214] 도 33, 34는 본 발명의 일 실시예에 따른 BDOF에 기반한 움직임 정보의 보정을 수행하는 방법을 나타내는 도면이다.

[0215] 도 33, 34의 BDOF에 기반한 움직임 정보의 보정은 도 27의 S2740, S2706 단계의 BDOF 기반 움직임 정보의 보정을 구체적으로 나타낸다.

[0216] 도 33을 참조하면, 디코더는 현재 블록을 서브 블록으로 분할한 후, 입력된 초기 움직임 정보에 기초하여 BDOF 기반 움직임 정보의 보정 값을 계산하여 최종 보정 움직임 정보를 획득할 수 있다. BDOF는 양방향 움직임으로 구성된 블록의 참조 블록으로부터 화소의 변화량을 추정하여 예측 블록을 보정하는데 사용될 수 있다. BDOF에서 유도된 움직임 정보를 이용하여 현재 블록의 움직임을 보정할 수 있다. 현재 블록이 affine, LIC, OBMC, sub-block MC, CIIP, SMVD, BCW with different weight, MMVD 중에서 적어도 하나의 모드로 부호화 되었다면, BDOF 기반 움직임 보정은 수행되지 않을 수 있다. 한편, BDOF 기반 움직임 보정은 다음의 조건 중 어느 하나라도 만족할 때 수행될 수 있다. BDOF 기반 움직임 보정이 수행되기 위한 조건은 현재 블록이 i) merge 모드의 움직임이 양방향인 경우, ii) 참조 픽처들과 현재 픽처 간의 거리가 동일한 경우, iii) 참조 블록들 간의 가중치 예측이 적용되지 않은 블록인 경우, iv) 현재 블록의 크기가 임의의 정해진 크기 이상일 경우일 수 있다. 임의의 정해진 크기는 블록의 가로, 세로일 수 있다. 예를 들어, 블록의 가로의 크기는 '8', 세로의 크기는 '8'일 수 있

다. 또한, BDOF 기반 움직임 보정은 서브 블록 단위로 수행될 수 있고, 서브 블록의 크기는 최대 16x16일 수 있다.

[0217] 이하에서는 각 서브 블록에 대해 BDOF 기반 움직임 보정을 수행하는 과정에 대해 설명한다. 도 34를 참조하면 인접한 서브 블록들 간에는 움직임 정보가 동일할 수 있으며, 동일한 움직임 정보를 가지는 서브 블록들을 병합하여 BDOF는 한 번만 수행할 수 있다(S3401, S3402). 현재 블록의 너비와 높이의 크기 차이에 따라 서브 블록들을 가로 혹은 세로 방향으로 병합할지 결정할 수 있다. 예를 들어, 현재 블록의 너비보다 높이가 크다면, 서브 블록들은 세로 방향으로 병합될 수 있고, 현재 블록의 높이보다 너비가 크다면, 서브 블록들은 가로 방향으로 병합될 수 있다. 디코더는 각각의 서브 블록 또는 병합된 서브 블록들에 대하여 S3403 내지 S3414 단계를 반복적으로 수행하여 보정된 움직임 정보 혹은 예측 블록을 획득할 수 있다.

[0218] 디코더는 L0 참조 블록과 L1 참조 블록을 이용하여 BDOF 파라미터를 획득할 수 있다(S3403). BDOF 파라미터는 화소 값 간의 변화량 등이 있을 수 있다. 디코더는 현재 블록을 최대 8x8 크기의 서브 블록으로 분할한 후(S3404), 각각의 서브 블록마다 S3405 내지 S3412의 단계를 반복 수행할 수 있다.

[0219] 디코더는 L0 및 L1 참조 블록 간의 화소(픽셀) 기반 코스트 값을 획득할 수 있다(S3405). 디코더는 코스트 값을 기 설정된 한계 값과 비교할 수 있다(S3406). 비교 결과 코스트 값이 기 설정된 한계 값보다 작다면 디코더는 움직임 보정을 수행하지 않고 움직임 보정 값에 (0,0)을 대입하여 L0 및 L1 참조 블록을 사용하여 서브 블록에 대한 예측 블록을 생성할 수 있다(S3407, S3409). 반대로, 비교 결과 코스트 값이 기 설정된 한계 값보다 같거나 크다면 디코더는 BDOF에 기반한 움직임 보정 값을 획득할 수 있다(S3408). 디코더는 움직임 보정 값이 (0, 0)인지 여부를 확인할 수 있다(S3410). 움직임 보정 값이 (0,0)이라면, 디코더는 BDOF에 기반한 예측 블록을 생성할 수 있고, 움직임 보정 값이 (0,0)이 아니라면, S3408 단계에서 획득한 보정 값을 저장할 수 있다(S3411, S3412). S3412 단계에서 저장되는 보정된 움직임 정보는 도 33의 최종 보정된 움직임 정보일 수 있다.

[0220] 디코더는 S3403 내지 S3412 단계가 모든 8x8 서브 블록마다 수행되었는지 확인할 수 있다(S3413). S3405 내지 S3412 단계가 수행되지 않은 8x8 서브 블록이 있다면 디코더는 S3405 내지 S3412 단계를 반복하여 수행할 수 있다. S3405 내지 S3412 단계가 모든 8x8 서브 블록에 수행되었다면 디코더는 모든 16x16 서브 블록에 대하여 S3403 내지 S3412 단계가 수행되었는지 확인할 수 있다(S3414). S3403 내지 S3412 단계가 수행되지 않은 16x16 서브 블록이 있다면 디코더는 S3403 내지 S3412 단계를 반복하여 수행할 수 있다. S3403 내지 S3412 단계가 모든 16x16 서브 블록에 수행되었다면 BDOF에 기반한 움직임 보정을 종료할 수 있다. 이때, 8x8 서브 블록은 16x16 서브 블록을 구성하는 서브 블록일 수 있다.

[0221] 도 35, 도 36은 본 발명의 일 실시예에 따른 BDOF에 기반한 현재 블록에 대한 예측 블록을 생성하는 방법을 나타내는 도면이다.

[0222] 도 35를 참조하면 디코더는 도 33, 도 34에서 설명한 과정을 수행하여 획득된 최종 보정된 움직임 정보를 이용하여 현재 블록에 대한 예측 블록을 생성할 수 있다. 디코더는 현재 블록을 서브 블록으로 분할하여 각 서브 블록 단위로 최종 보정된 움직임 정보를 이용하여 BDOF에 기반한 현재 블록에 대한 예측 블록을 생성할 수 있다.

[0223] 이하에서 BDOF에 기반한 예측 블록을 생성하는 구체적인 과정에 대해 설명한다.

[0224] 도 36을 참조하면 디코더는 현재 블록을 8x8 서브 블록으로 분할할 수 있다(S3601). 디코더는 도 27의 S2730 또는 S2704, S2705 단계를 수행하여 획득되는 보정된 움직임 정보와 S2740 또는 S2706 단계를 수행하여 획득되는 움직임 보정 값을 합산하여 최종 보정된 움직임 정보를 획득할 수 있다(S3602). 이때, 디코더는 도 27의 S2740 또는 S2706 단계에서 획득된 움직임 보정 값이 (0,0)인지 여부를 확인할 수 있다(S3603). 확인 결과 움직임 보정 값이 (0,0)인 경우, 디코더는 휘도 성분에 대한 예측 블록으로 도 34의 S3409, S3411 단계에서 획득된 예측 블록을 사용할 수 있다(S3604). 또한, 확인 결과 움직임 보정 값이 (0,0)인 경우, 디코더는 색차 성분에 대한 예측 블록을 생성할 수 있다(S3606). 한편, 확인 결과 움직임 보정 값이 (0,0)이 아닌 경우, 디코더는 BDOF에 기반한 휘도 성분 및 색차 성분에 대한 예측 블록을 생성할 수 있다(S3605). 디코더는 상술한 S3602 내지 S3606 단계가 모든 8x8 서브 블록에 대해 수행되었는지 여부를 확인하고, 수행되지 않은 서브 블록이 있다면 S3602 내지 S3606 단계를 반복하여 수행할 수 있다(S3607).

[0225] 도 37은 본 발명의 일 실시예에 따른 DMVR이 적용되는 서브블록을 나타내는 도면이다.

[0226] 이하에서 도 27을 참조하여 상술한 다중 DMVR을 수행하는 다양한 방법에 대해 설명한다.

[0227] 도 27을 통해 설명한 DMVR의 각 단계가 수행되는지 여부는 현재 블록의 크기, 현재 블록의 컬러 성분(현재 블록

이 휘도 성분 블록인지 색차 성분 블록인지), 현재 블록에 대한 양자화 파라미터 정보, 현재 블록의 움직임 해상도 정보, 현재 블록의 주변 블록의 잔차 신호의 존재 여부와 관련된 정보들 적어도 어느 하나에 기초하여 결정될 수 있다. 예를 들어, 현재 블록이 색차 성분 블록이거나 현재 블록의 주변 블록의 잔차 신호가 임의의 값 이내인 경우, 도 27의 단계들 중 일부는 생략되거나 특정 단계 이후의 단계는 생략될 수 있다. 계산 복잡도를 감소시키기 위함이다.

[0228] 도 27을 통해 설명한 DMVR의 각 단계의 수행 여부는 현재 블록의 크기에 따라 결정될 수 있다. 크기가 큰 블록일수록 분할되는 서브 블록의 수가 증가하기 때문에 현재 블록의 크기가 임의의 값보다 같거나 크다면, DMVR의 각 단계 중 일부 또는 전부는 수행되지 않을 수 있다. 이때, 임의의 값은 현재 블록의 가로와 세로의 크기의 합일 수 있고, 구체적으로 128일 수 있다.

[0229] 한편, 현재 블록(예, 코딩 블록)의 크기가 임의의 값보다 크다면 서브 블록의 크기도 비례적으로 크게 분할되도록 설정하여 도 27을 통해 설명한 DMVR의 각 단계가 수행될 수 있다. 이때의 임의의 값은 16일 수 있다. 현재 블록이 임의의 값보다 크다면 서브 블록의 너비 및 높이를 기 설정된 크기보다 n(예, 자연수)배만큼 확장하여 확장된 크기로 서브 블록은 분할될 수 있다. 또는 현재 블록에 적용될 수 있는 서브 블록의 최대 개수가 설정될 수 있고, 최대 개수에 기초하여 서브 블록의 너비 및 높이가 결정될 수 있다.

[0230] 비디오 신호 처리 장치의 허용 가능한 병렬 프로세스의 개수에 따라 도 27을 통해 설명한 DMVR의 각 단계의 수행 여부가 결정될 수 있다. 비디오 신호 처리 장치마다 병렬적으로 처리 가능한 자원(예, 프로세스 및 쓰레드)의 개수가 제한적이기 때문이다. 따라서, 제한된 처리 가능한 자원 하에서 DMVR의 단계별 수행 여부가 달라질 수 있고, 블록마다 DMVR이 선택적으로 적용될 수 있다. 예를 들어 처리 가능한 자원의 개수가 임의의 값보다 같거나 작은 경우 도 27을 통해 설명한 DMVR의 각 단계의 일부가 생략되거나 특정 단계 이후의 단계가 생략될 수 있다. 또한 서브 블록들 중에서 특정 위치의 서브 블록에만 DMVR이 적용될 수 있다. 특정 위치는 분할된 서브 블록 중에서 홀수 또는 짝수 번째 블록이 될 수 있다. 또는 도 37에 나타난 바와 같이 DMVR이 적용되는 서브 블록은 좌상단, 우상단, 좌하단, 우하단에 위치한 서브 블록일 수 있다. DMVR이 적용되지 않은 서브 블록에 대한 움직임 정보는 DMVR이 적용된 블록의 주변 블록들의 보정된 움직임 정보를 이용하여 획득될 수 있다.

[0231] 병렬적으로 처리 가능한 자원의 개수에 따라 분할되는 서브 블록의 크기가 결정될 수 있다. 분할되는 서브 블록의 개수가 많을수록, 비디오 신호 처리 장치의 처리 속도는 느려질 수 있다. 병렬적으로 처리 가능한 자원의 개수와 서브 블록의 크기는 반비례할 수 있다. 예를 들어 병렬적으로 처리 가능한 자원의 개수가 많을수록 분할되는 서브 블록의 크기는 작아질 수 있다. 즉, 병렬적으로 처리 가능한 자원의 개수가 많을수록 분할되는 서브 블록의 개수는 많아질 수 있다. 반대로, 병렬적으로 처리 가능한 자원의 개수가 작을수록 분할되는 서브 블록의 크기는 커질 수 있다. 또한, 병렬적으로 처리 가능한 자원의 개수가 많고 적고는 임의의 값보다 큰지 작은지 여부로 결정될 수 있고, 임의의 값에 따라 결정되는 자원의 개수에 따라 서브 블록의 크기도 결정될 수 있다. 또한 서브 블록의 크기는 병렬적으로 처리 가능한 자원의 개수뿐 아니라 현재 블록의 양자화 파라미터 정보, 현재 블록의 크기 또는 현재 블록의 컬러 성분(휘도 또는 색차 성분), 현재 블록의 움직임 해상도 정보, 현재 블록의 주변 블록의 잔차 신호의 존재 여부와 관련된 정보들 중 적어도 어느 하나에 기초하여 결정될 수 있다. 이때, 서브 블록의 크기, 분할되는 서브 블록의 개수는 별도로 시그널링될 수 있다. 예를 들어, 디코더는 비트스트림에 포함된 신택스 요소를 파싱하여 서브 블록의 크기, 분할되는 서브 블록의 개수를 결정할 수 있다. 이때, 신택스 요소는 SPS 레벨, PPS 레벨, 픽처 레벨, 또는 슬라이스 레벨, CU(Coding Unit) 레벨에서 시그널링될 수 있다.

[0232] 상기 다중 DMVR 과정에서 움직임 보정 과정은 현재 블록의 휘도 성분에 대해서만 수행되고, 색차 성분에는 수행되지 않을 수 있다. 이때, 다중 DMVR을 통해 획득되는 현재 블록의 휘도 성분에 대한 보정된 움직임 정보는 현재 블록의 색차 성분에 사용될 수 있다.

[0233] 도 38은 본 발명의 일 실시예에 따른 다중 DMVR을 수행하면서 현재 블록에 대한 예측 블록을 생성하는 방법을 나타내는 도면이다.

[0234] DMVR 방법은 보다 더 정확한 움직임 정보를 찾기 위한 방법으로써, 디코더는 최종적으로 보정된 움직임 정보를 사용하여 하나의 예측 블록을 생성할 수 있다. DMVR은 이미 복원된 참조 영상들의 상관성을 통해 움직임 정보를 보정하는 방법이므로, 실제 원본 영상에서 찾은 움직임 정보에 비해 정확도가 낮을 수 있다. 이러한 낮은 정확도를 보완해주기 위해서, 도 38을 참조하면 디코더는 DMVR을 수행하는 과정에서 획득되는 각 단계별 움직임 정보들을 선택적으로 활용하여 여러 개의 예측 블록들을 생성할 수 있다. 디코더는 예측 블록들 간의 가중치 평균을 통해 최종적으로 현재 블록에 대한 예측 블록을 생성할 수 있다. 컨트롤러에서는 각 단계별로 생성된 예측

블록들의 사용 여부를 결정할 수 있으며, 가중치 평균 단계에서 각 예측 블록들의 가중치를 설정할 수 있다. 각 단계별 예측 블록들의 사용 여부는 예측 블록을 생성할 때 이미 구해진 화소(픽셀) 기반 코스트 값에 기초하여 결정될 수 있다. 또한 코스트 값을 임의의 값과 비교한 결과에 따라 사용되는 예측 블록의 수가 결정될 수 있다. 예를 들어, 코스트 값이 임의의 값보다 작다면 기 설정된 X개의 예측 블록이 사용될 수 있고, 임의의 값보다 크다면 기 설정된 Y개의 예측 블록이 사용될 수 있다. 이때, X, Y는 같거나 다른 정수 값일 수 있다. 컨트롤러는 DMVR의 각 단계별 수행 여부도 결정할 수 있다. DMVR의 각 단계별 수행 여부는 현재 블록의 양자화 파라미터, 현재 블록의 크기 현재 블록의 컬러 성분(휘도 성분인지 색차 성분인지) 현재 블록의 움직임 해상도 정보, 현재 블록의 주변 블록의 잔차 신호의 존재 여부와 관련된 정보들 중 적어도 어느 하나에 기초하여 결정될 수 있다.

[0235] 도 39는 본 발명의 일 실시예에 따른 현재 블록을 예측하는 방법을 나타내는 순서도이다.

[0236] 도 39는 도1 내지 도 38을 통해 설명한 코스트 값에 기초하여 현재 블록을 예측하는 방법을 나타낸다.

[0237] 도 39를 참조하면 디코더는 현재 블록과 관련된 하나 이상의 제1 움직임 정보들을 포함하는 제1 움직임 정보 리스트를 획득할 수 있다(S3910). 디코더는 상기 하나 이상의 제1 움직임 정보들을 보정하여 하나 이상의 제1 보정된 움직임 정보들을 획득할 수 있다(S3920). 디코더는 상기 하나 이상의 제1 보정된 움직임 정보들 각각에 대한 하나 이상의 제1 코스트 값들을 획득할 수 있다(S3930). 디코더는 상기 하나 이상의 제1 보정된 움직임 정보들을 상기 제1 코스트 값들에 기초하여 재정렬하여 제2 움직임 정보 리스트를 획득할 수 있다(S3940). 디코더는 상기 제2 움직임 정보 리스트에 포함되는 제1 움직임 정보에 기초하여 하나 이상의 제2 움직임 정보들을 획득할 수 있다(S3950). 디코더는 상기 하나 이상의 제2 움직임 정보들을 보정하여 하나 이상의 제2 보정된 움직임 정보들을 획득할 수 있다(S3960). 디코더는 상기 제2 보정된 움직임 정보들 각각에 대한 하나 이상의 제2 코스트 값들을 획득할 수 있다(S3970). 디코더는 상기 제2 코스트 값들에 기초하여 결정되는 제2 움직임 정보에 기초하여 상기 현재 블록을 예측할 수 있다(S3980).

[0238] 이때, 상기 하나 이상의 제1 코스트 값들은 상기 현재 블록과 상기 하나 이상의 제1 보정된 움직임 정보들 각각에 대응되는 참조 블록간의 유사도와 관련된 값일 수 있다. 상기 하나 이상의 제2 코스트 값들은 상기 현재 블록과 상기 하나 이상의 제2 보정된 움직임 정보들 각각에 대응되는 참조 블록간의 유사도와 관련된 값일 수 있다.

[0239] 상기 제2 움직임 정보 리스트의 제1 보정된 움직임 정보들은 상기 하나 이상의 제1 보정된 움직임 정보들에 각각 대응되는 코스트 값이 오름차순이 되도록 정렬될 수 있다.

[0240] 상기 제1 움직임 정보는 상기 제1 코스트 값들 중 가장 작은 코스트 값에 대응하는 보정된 움직임 정보이고, 상기 제2 움직임 정보는 상기 제2 코스트 값들 중 가장 작은 코스트 값에 대응하는 보정된 움직임 정보일 수 있다.

[0241] 상기 하나 이상의 제1 움직임 정보들은 각각 서로 다른 픽처에 포함되고, 상기 하나 이상의 제2 움직임 정보들은 각각 서로 다른 픽처에 포함될 수 있다.

[0242] 상기 하나 이상의 제1 보정된 움직임 정보들 및 상기 하나 이상의 제2 보정된 움직임 정보들은, MVD(Motion Vector Difference), TM(Template Matching), BM(Bilateral Matching), MMVD(Merge mode with MVD), MMVD(Merge mode with MVD) 기반의 TM, 광 흐름(Optical flow)에 기초한 기반 TM, 멀티 패스 DMVR (Multi pass Decoder-side Motion Vector Refinement) 중 적어도 어느 하나 이상에 기초하여 보정될 수 있다.

[0243] 상기 현재 블록의 부호화 모드가 머지(merge) 모드인 경우, 상기 하나 이상의 제1 움직임 정보들 및 상기 하나 이상의 제2 움직임 정보들은 MMVD(Merge mode with MVD)에 의해 유도되는 움직임 정보일 수 있다.

[0244] 상기 하나 이상의 제1 움직임 정보들 각각에 대응되는 블록들은 제1 탐색 영역 내에 위치하는 블록이고, 상기 하나 이상의 제2 움직임 정보들 각각에 대응되는 블록들은 제2 탐색 영역 내에 위치하는 블록이고, 상기 제1 탐색 영역과 상기 제2 탐색 영역은 서로 상이할 수 있다.

[0245] 상기 제1 코스트 값들은 상기 하나 이상의 제1 보정된 움직임 정보들 각각에 대해 순차적으로 계산되고, 상기 제2 코스트 값들은 상기 하나 이상의 제2 보정된 움직임 정보들 각각에 대해 순차적으로 계산될 수 있다. 이때, 상기 제1 코스트 값들이 순차적으로 계산되는 중 기 설정된 제1 값보다 작은 코스트 값이 계산되는 경우, 나머지 보정된 움직임 정보들에 대한 코스트 값들은 계산되지 않을 수 있다. 상기 제2 코스트 값들이 순차적으로 계산되는 중 기 설정된 제2 값보다 작은 코스트 값이 계산되는 경우, 나머지 보정된 움직임 정보들에 대한 코스트

값들은 계산되지 않을 수 있다.

[0246] 본 명세서에서 상술한 방법들은 디코더 또는 인코더의 프로세서를 통해 수행될 수 있다. 또한, 인코더는 상술한 방법들에 의해 디코딩되는 비트스트림을 생성할 수 있다. 또한, 인코더가 생성한 비트스트림은 컴퓨터 판독 가능한 비 일시적 저장 매체(기록 매체)에 저장될 수 있다.

[0247] 본 명세서는 주로 디코더 관점에서 기술되었으나 인코더에서도 동일하게 동작될 수 있다. 본 명세서의 파싱이라는 용어는 비트스트림으로부터 정보를 획득하는 과정을 중점으로하여 설명되었으나 인코더 측면에서는 비트스트림에 해당 정보를 구성하는 것으로 해석될 수 있다. 따라서 파싱이라는 용어는 디코더 동작으로만 한정되지 않고 인코더에서는 비트스트림을 구성하는 행위로까지 해석될 수 있다. 또한, 이러한 비트스트림은 컴퓨터 판독 가능한 기록 매체에 저장되어 구성될 수 있다.

[0248] 상술한 본 발명의 실시예들은 다양한 수단을 통해 구현될 수 있다. 예를 들어, 본 발명의 실시예들은 하드웨어, 펌웨어(firmware), 소프트웨어 또는 그것들의 결합 등에 의해 구현될 수 있다.

[0249] 하드웨어에 의한 구현의 경우, 본 발명의 실시예들에 따른 방법은 하나 또는 그 이상의 ASICs(Application Specific Integrated Circuits), DSPs(Digital Signal Processors), DSPDs(Digital Signal Processing Devices), PLDs(Programmable Logic Devices), FPGAs(Field Programmable Gate Arrays), 프로세서, 컨트롤러, 마이크로 컨트롤러, 마이크로 프로세서 등에 의해 구현될 수 있다.

[0250] 펌웨어나 소프트웨어에 의한 구현의 경우, 본 발명의 실시예들에 따른 방법은 이상에서 설명된 기능 또는 동작들을 수행하는 모듈, 절차 또는 함수 등의 형태로 구현될 수 있다. 소프트웨어 코드는 메모리에 저장되어 프로세서에 의해 구동될 수 있다. 상기 메모리는 프로세서의 내부 또는 외부에 위치할 수 있으며, 이미 공지된 다양한 수단에 의해 프로세서와 데이터를 주고받을 수 있다.

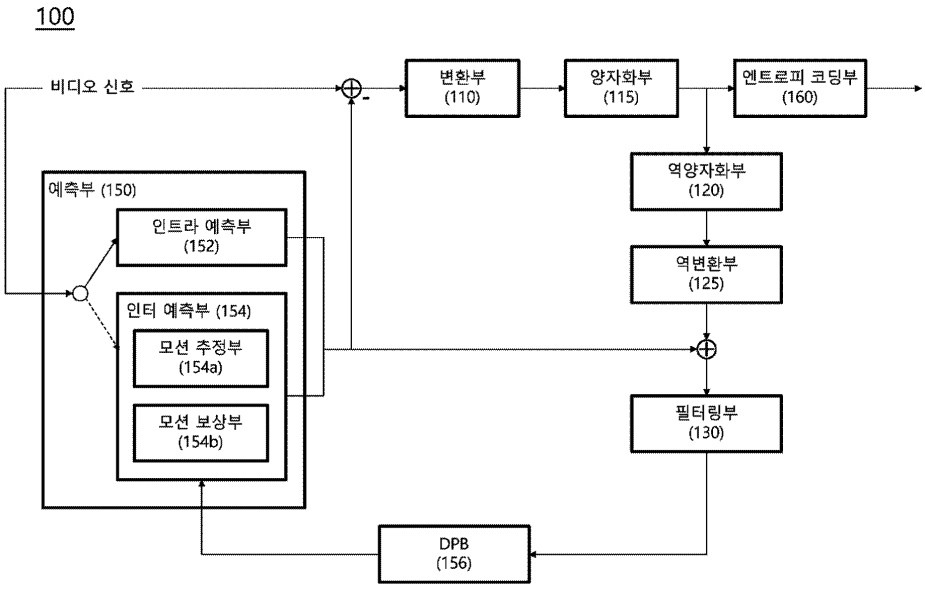
[0251] 일부 실시예는 컴퓨터에 의해 실행되는 프로그램 모듈과 같은 컴퓨터에 의해 실행가능한 명령어를 포함하는 기록 매체의 형태로도 구현될 수 있다. 컴퓨터 판독 가능 매체는 컴퓨터에 의해 액세스될 수 있는 임의의 가용 매체일 수 있고, 휘발성 및 비휘발성 매체, 분리형 및 비분리형 매체를 모두 포함한다. 또한, 컴퓨터 판독 가능 매체는 컴퓨터 저장 매체 및 통신 매체를 모두 포함할 수 있다. 컴퓨터 저장 매체는 컴퓨터 판독가능 명령어, 데이터 구조, 프로그램 모듈 또는 기타 데이터와 같은 정보의 저장을 위한 임의의 방법 또는 기술로 구현된 휘발성 및 비휘발성, 분리형 및 비분리형 매체를 모두 포함한다. 통신 매체는 전형적으로 컴퓨터 판독가능 명령어, 데이터 구조 또는 프로그램 모듈과 같은 변조된 데이터 신호의 기타 데이터, 또는 기타 전송 메커니즘을 포함하며, 임의의 정보 전달 매체를 포함한다.

[0252] 전술한 본 발명의 설명은 예시를 위한 것이며, 본 발명이 속하는 기술분야의 통상의 지식을 가진 자는 본 발명의 기술적 사상이나 필수적인 특징을 변경하지 않고서 다른 구체적인 형태로 쉽게 변형이 가능하다는 것을 이해할 수 있을 것이다. 그러므로 이상에서 기술한 실시예들은 모든 면에서 예시적인 것이며 한정적인 것이 아는 것으로 해석해야 한다. 예를 들어, 단일형으로 설명되어 있는 각 구성 요소는 분산되어 실시될 수도 있으며, 마찬가지로 분산된 것으로 설명되어 있는 구성 요소들도 결합된 형태로 실시될 수 있다.

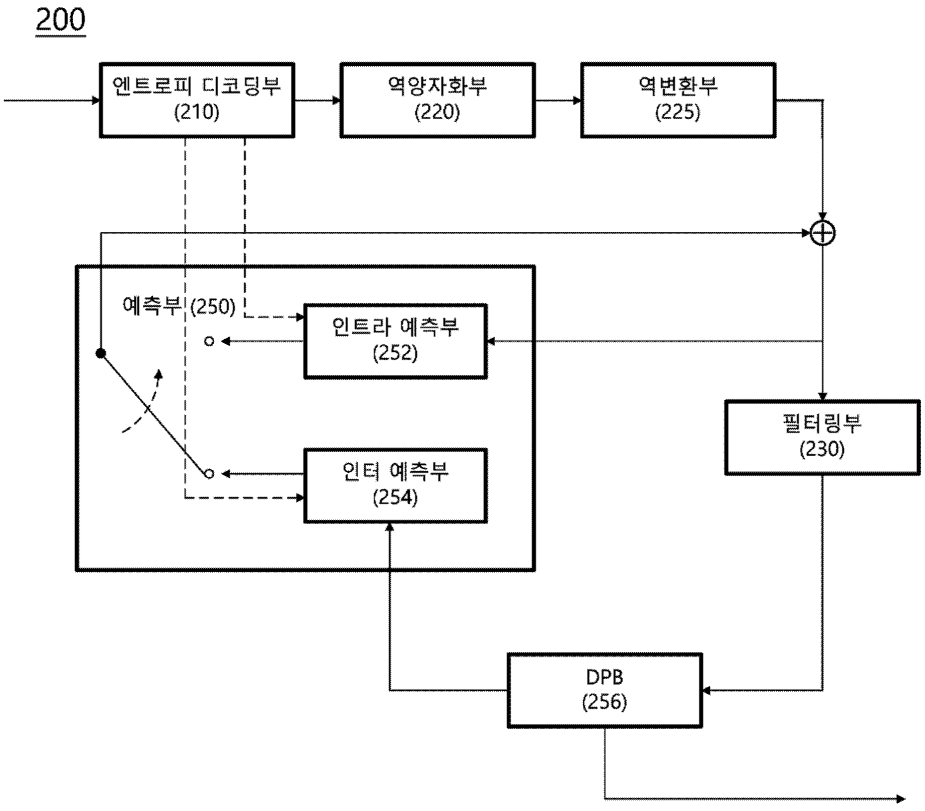
[0253] 본 발명의 범위는 상기 상세한 설명보다는 후술하는 특허청구범위에 의하여 나타내어지며, 특허청구범위의 의미 및 범위 그리고 그 균등 개념으로부터 도출되는 모든 변경 또는 변형된 형태가 본 발명의 범위에 포함되는 것으로 해석되어야 한다.

도면

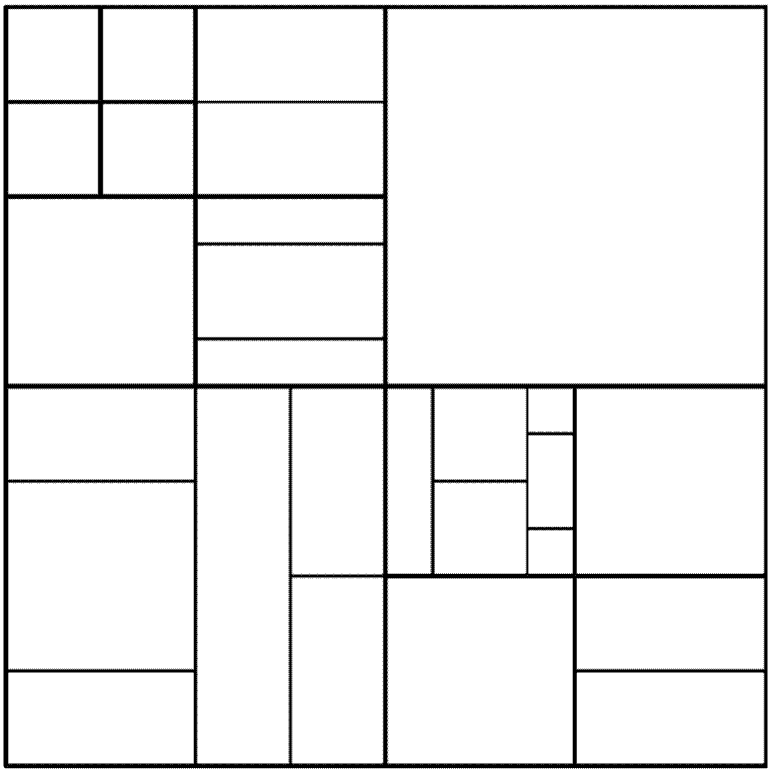
도면1


100
비디오 신호
변환부
(110)
양자화부
(115)
엔트로피 코딩부
(160)
역양자화부
(120)
역변환부
(125)
예측부 (150)
인트라 예측부
(152)
인터 예측부 (154)
모션 추정부
(154a)
모션 보상부
(154b)
필터링부
(130)
DPB
(156)

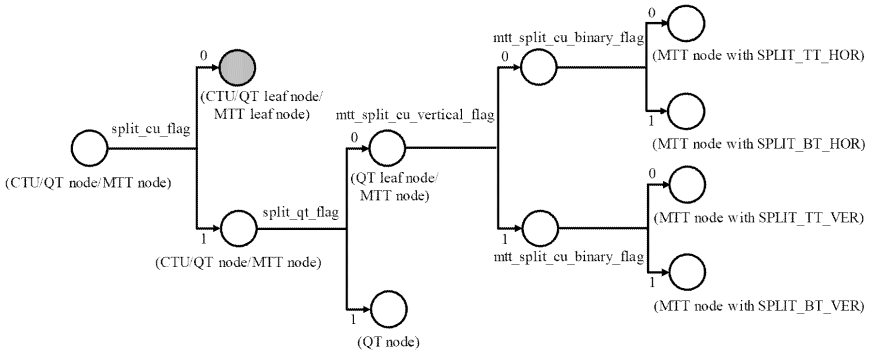
도면2


200
엔트로피 디코딩부
(210)
역양자화부
(220)
역변환부
(225)
예측부 (250)
인트라 예측부
(252)
인터 예측부
(254)
필터링부
(230)
DPB
(256)

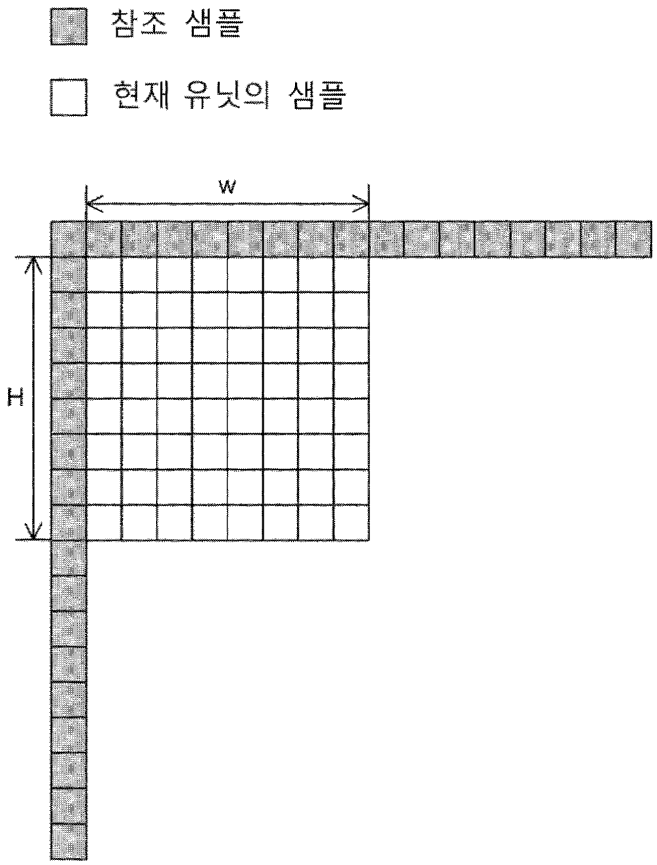
도면3



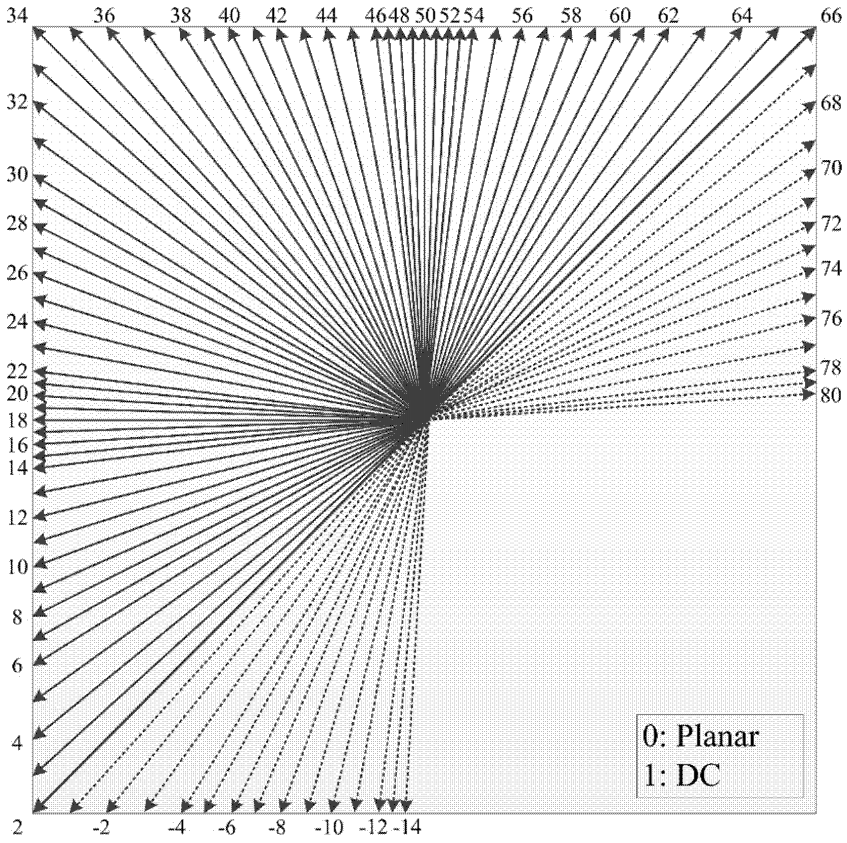
도면4


split_cu_flag
(CTU/QT node/MTT node)
0
(CTU/QT leaf node/
MTT leaf node)
1
(CTU/QT node/MTT node)
split_qt_flag
mtt_split_cu_vertical_flag
0
(QT leaf node/
MTT node)
1
(QT node)
mtt_split_cu_binary_flag
0
1
mtt_split_cu_binary_flag
0
(MTT node with SPLIT_TT_HOR)
1
(MTT node with SPLIT_BT_HOR)
0
(MTT node with SPLIT_TT_VER)
1
(MTT node with SPLIT_BT_VER)

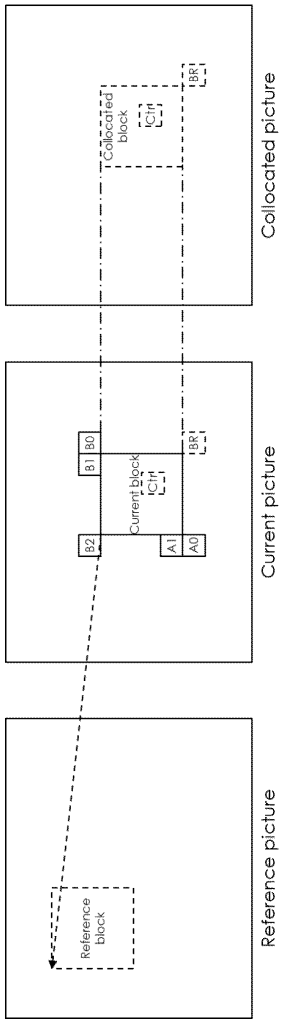
도면5


참조 샘플
현재 유닛의 샘플
W
H

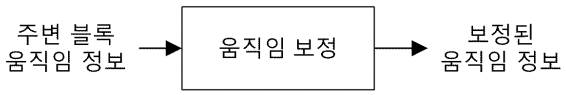
도면6


34 36 38 40 42 44 46 48 50 52 54 56 58 60 62 64 66
32 30 28 26 24 22 20 18 16 14 12 10 8 6 4 2
68 70 72 74 76 78 80
-2 -4 -6 -8 -10 -12 -14
0: Planar
1: DC

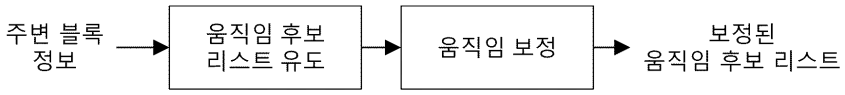
도면7


Collocated picture
Collocated block
CTR
BR1
Current picture
Current block
CTR
BR1
B0
B1
B2
A1
A0
Reference picture
Reference block

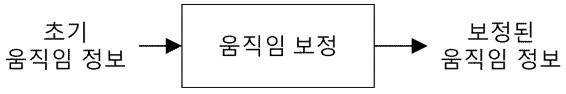
도면8


주변 블록
움직임 정보
움직임 보정
보정된
움직임 정보

(a)

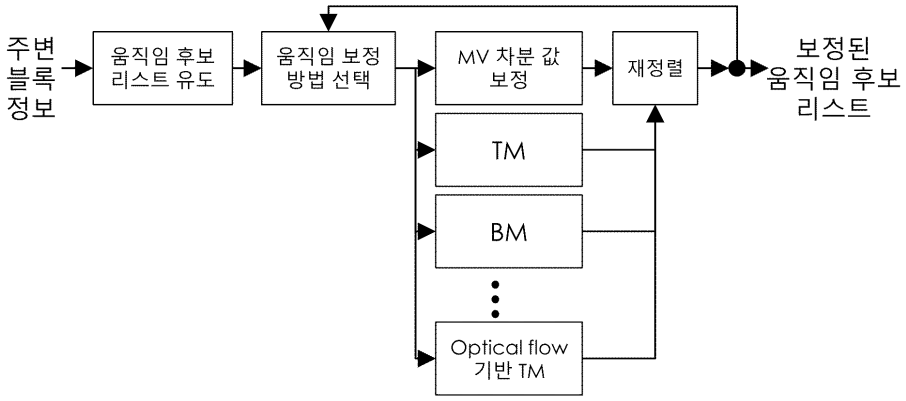

주변 블록
정보
움직임 후보
리스트 유도
움직임 보정
보정된
움직임 후보 리스트

(b)

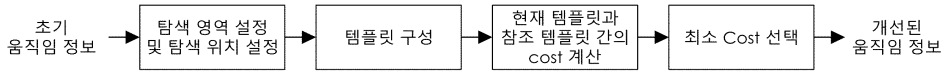

초기
움직임 정보
움직임 보정
보정된
움직임 정보

(c)

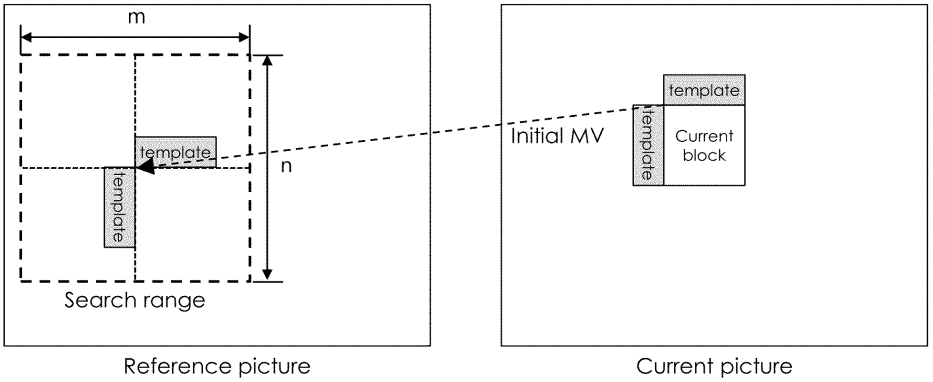
도면9



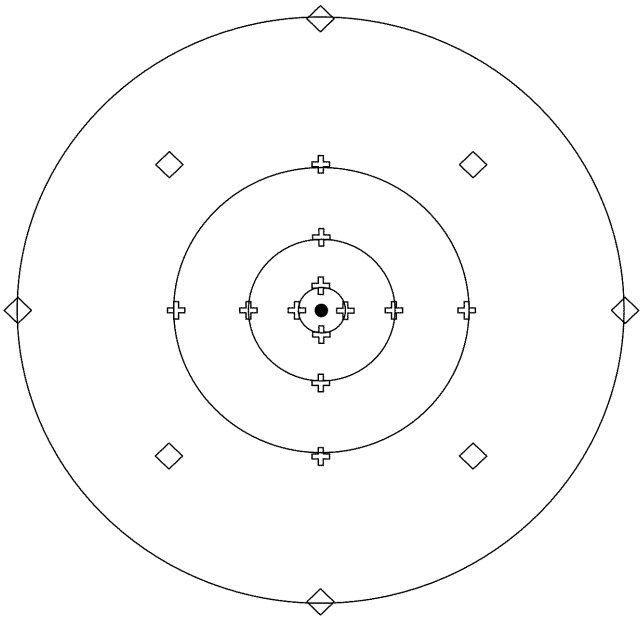
도면10



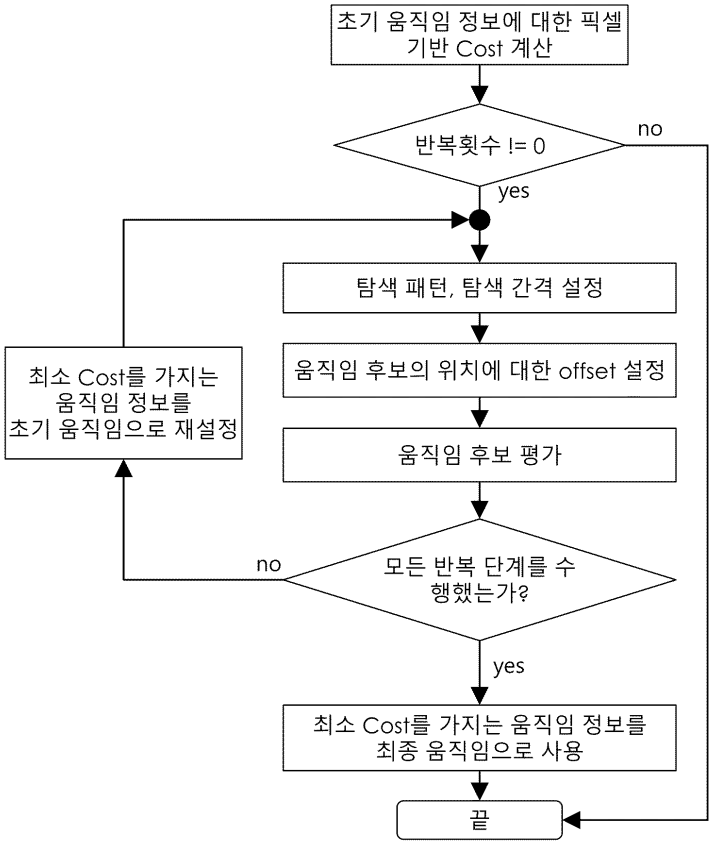
도면11



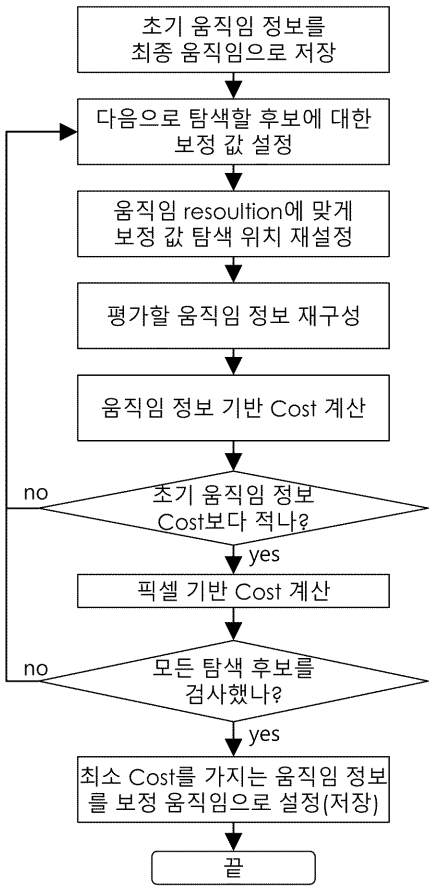
도면12



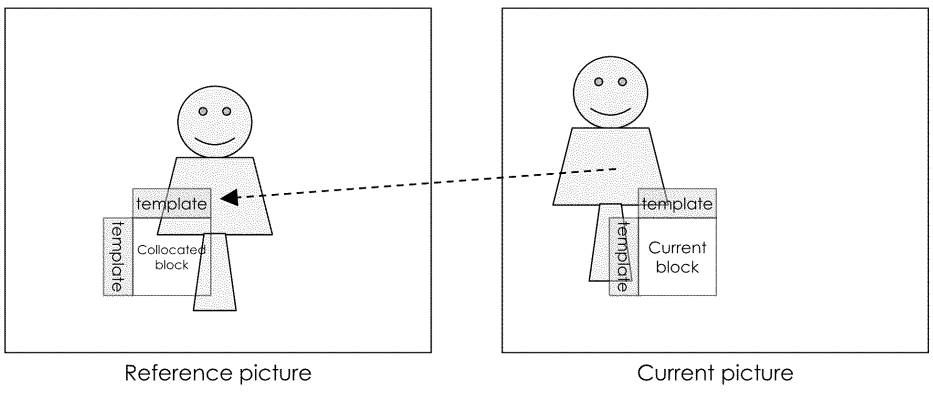
도면13


초기 움직임 정보에 대한 픽셀 기반 Cost 계산
반복횟수 != 0
no
yes
탐색 패턴, 탐색 간격 설정
움직임 후보의 위치에 대한 offset 설정
움직임 후보 평가
모든 반복 단계를 수 행했는가?
no
yes
최소 Cost를 가지는 움직임 정보를 초기 움직임으로 재설정
최소 Cost를 가지는 움직임 정보를 최종 움직임으로 사용
끝

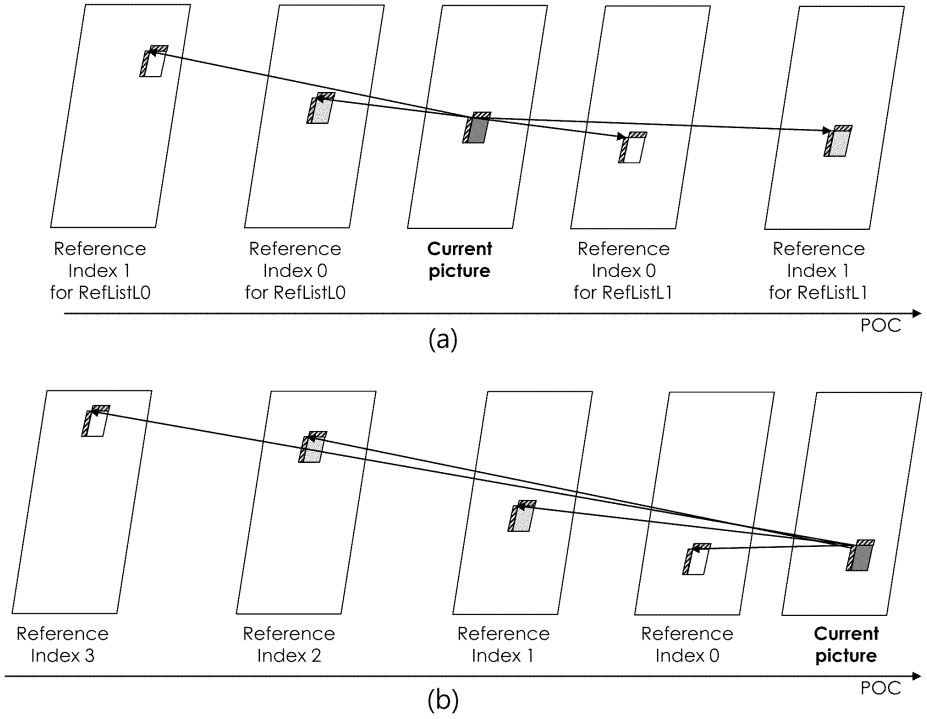
도면14


초기 움직임 정보를
최종 움직임으로 저장
다음으로 탐색할 후보에 대한
보정 값 설정
움직임 resoultion에 맞게
보정 값 탐색 위치 재설정
평가할 움직임 정보 재구성
움직임 정보 기반 Cost 계산
no
초기 움직임 정보
Cost보다 적나?
yes
픽셀 기반 Cost 계산
no
모든 탐색 후보를
검사했나?
yes
최소 Cost를 가지는 움직임 정보
를 보정 움직임으로 설정(저장)
끝

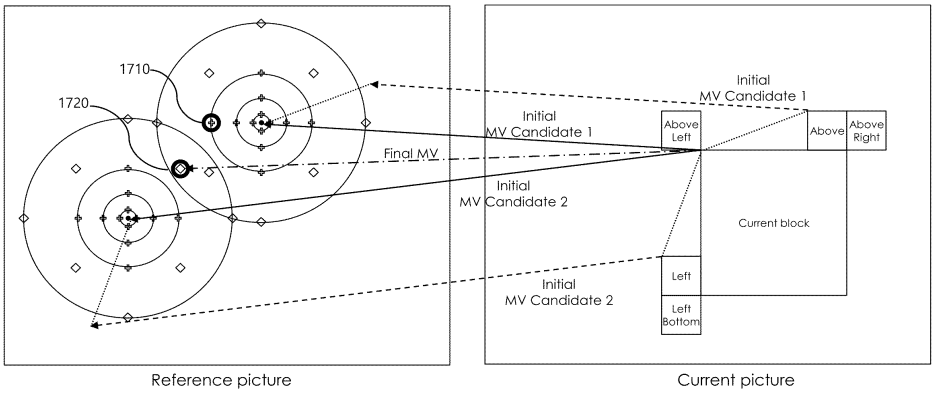
도면15


template
template
Collocated
block
template
template
Current
block
Reference picture
Current picture

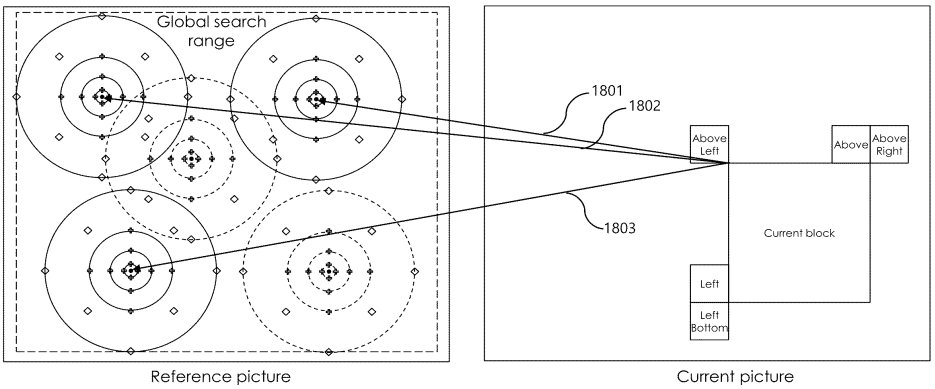
도면16



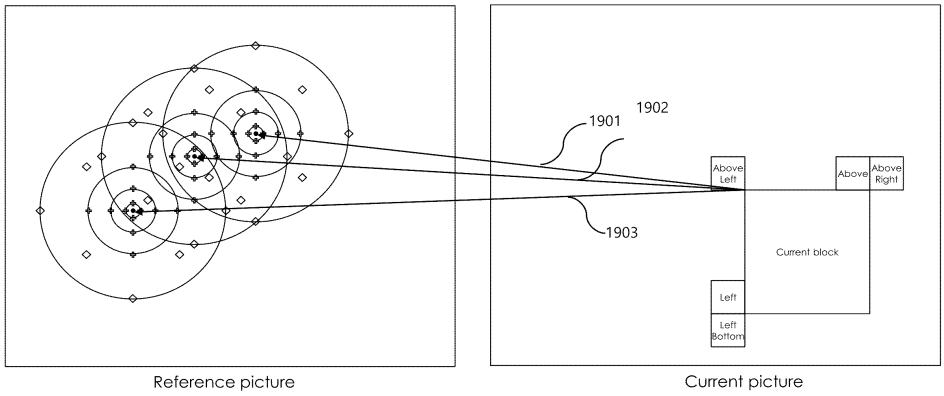
도면17



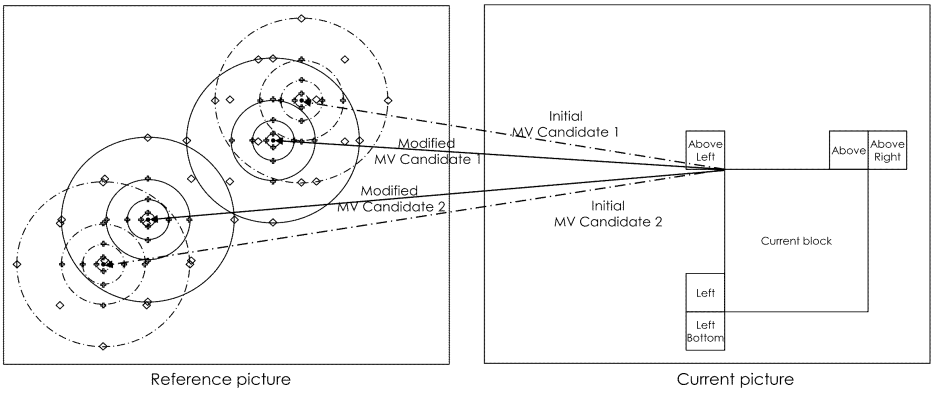
도면18



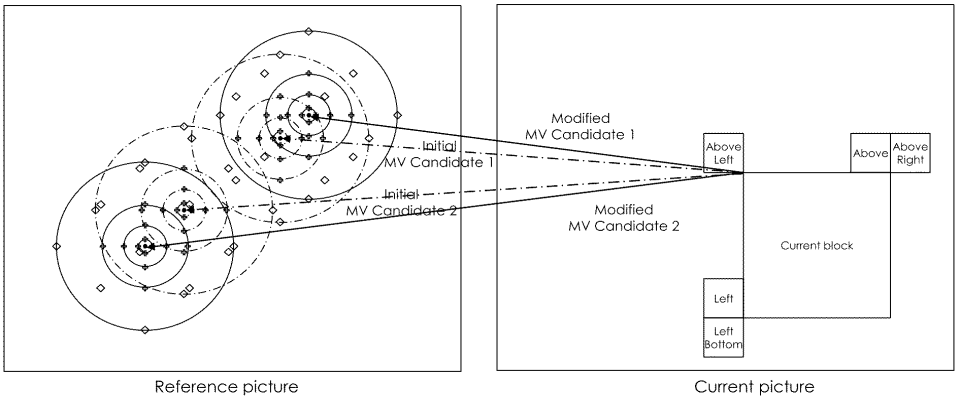
도면19



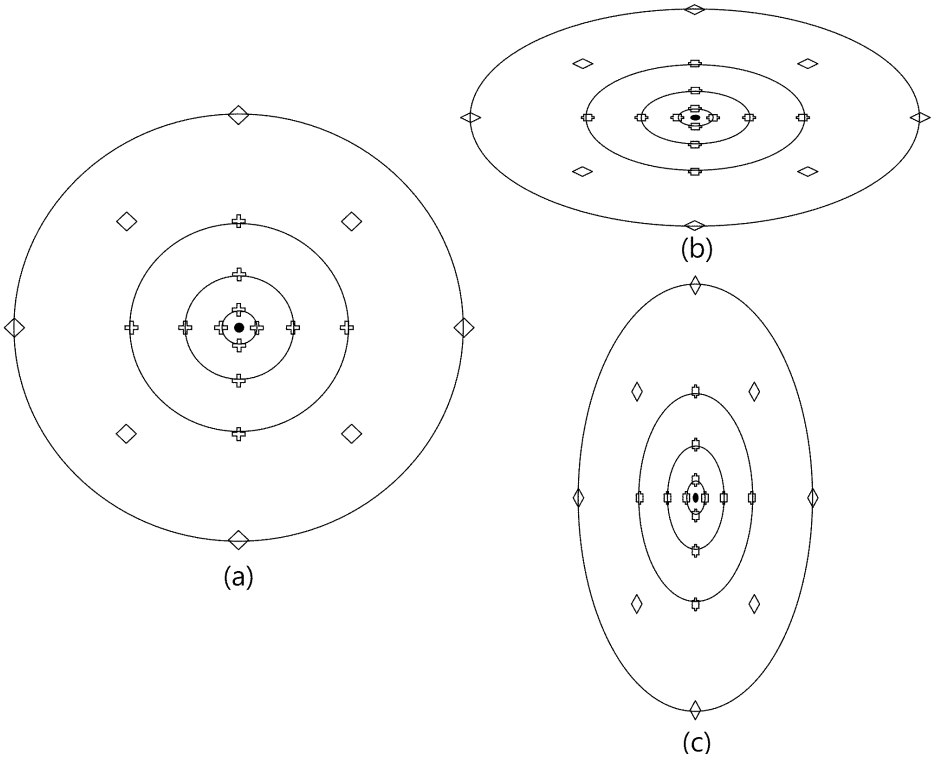
도면20



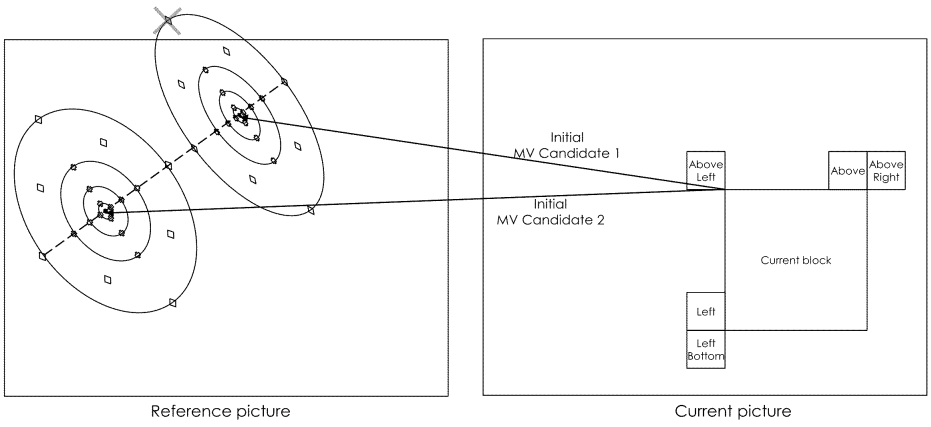
도면21



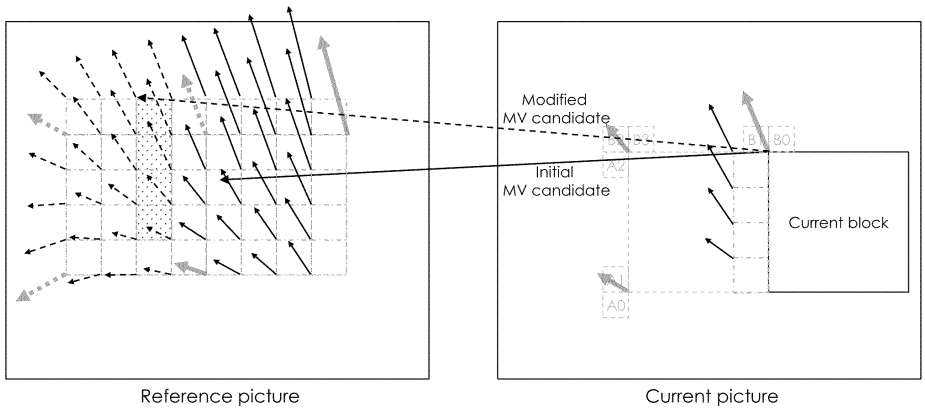
도면22



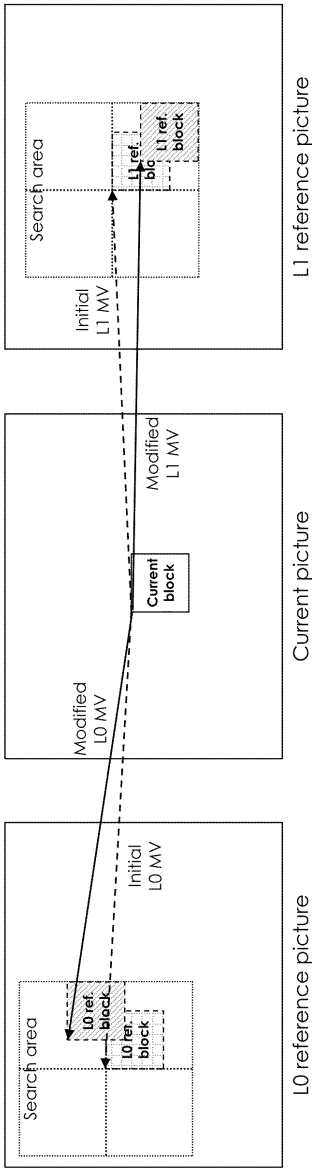
도면23



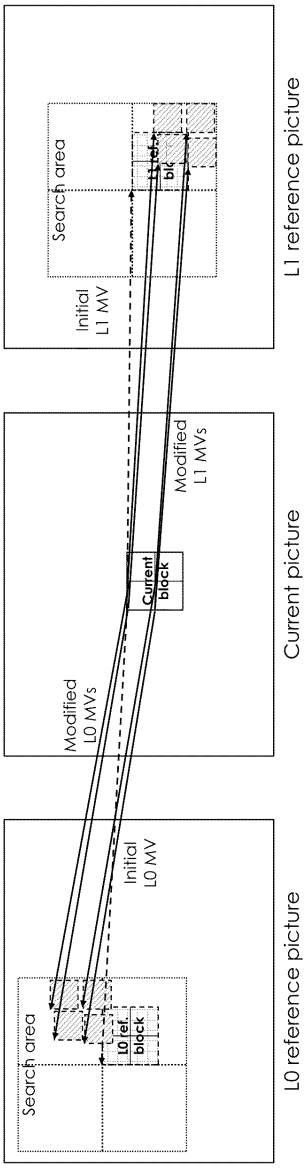
도면24



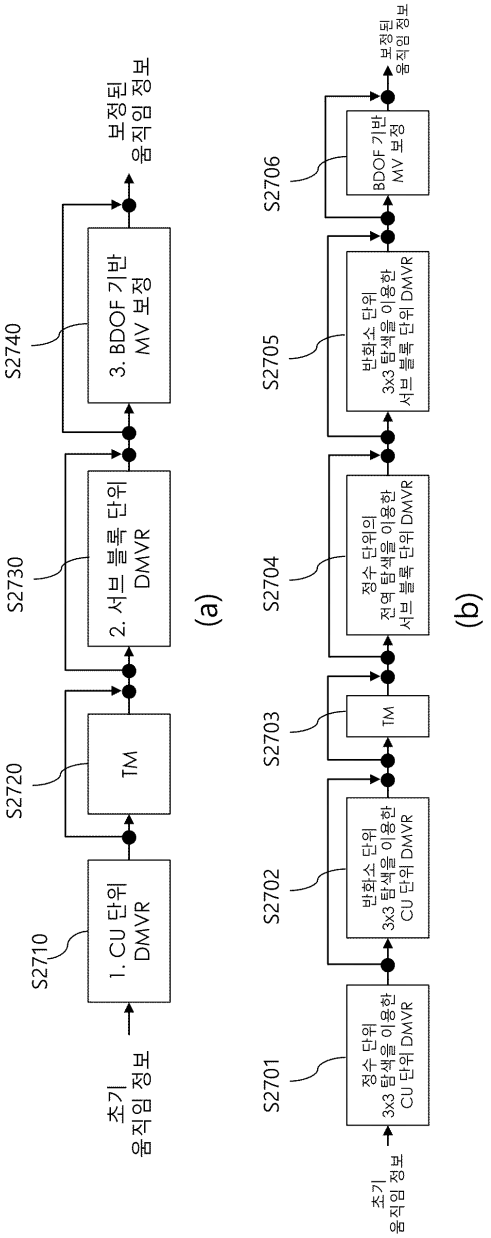
도면25


L1 reference picture
Search area
L1 ref. block
L1 ref. block
Initial L1 MV
Modified L1 MV
Current block
Current picture
Modified L0 MV
Initial L0 MV
L0 ref. block
L0 ref. block
Search area
L0 reference picture

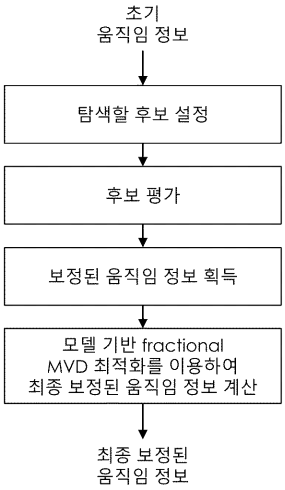
도면26


L1 reference picture
Search area
Initial
L1 MV
L1 ref.
block
Current picture
Modified
L1 MVs
Current
block
Modified
L0 MVs
L0 reference picture
Initial
L0 MV
Search area
L0 ref.
block

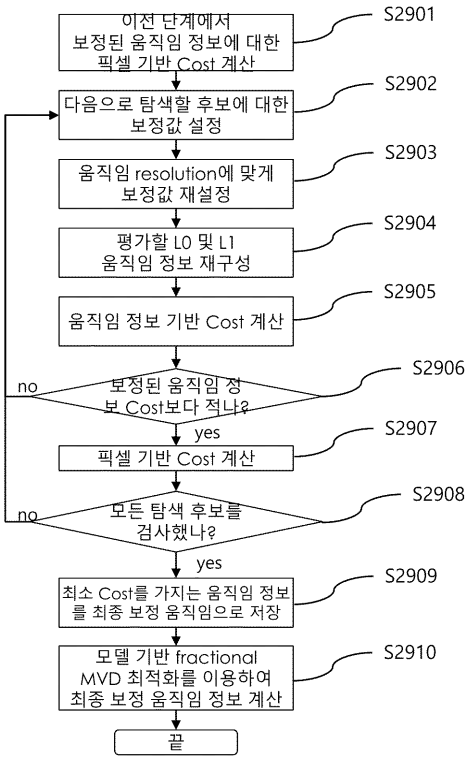
도면27


초기
움직임 정보
S2710
1. CU 단위
DMVR
S2720
TM
S2730
2. 서브 블록 단위
DMVR
S2740
3. BDOF 기반
MV 보정
보정된
움직임 정보
(a)
초기
움직임 정보
S2701
정수 단위
3x3 탐색을 이용한
CU 단위 DMVR
S2702
반화소 단위
3x3 탐색을 이용한
CU 단위 DMVR
S2703
TM
S2704
정수 단위의
전역 탐색을 이용한
서브 블록 단위 DMVR
S2705
반화소 단위
3x3 탐색을 이용한
서브 블록 단위 DMVR
S2706
BDOF 기반
MV 보정
보정된
움직임 정보
(b)

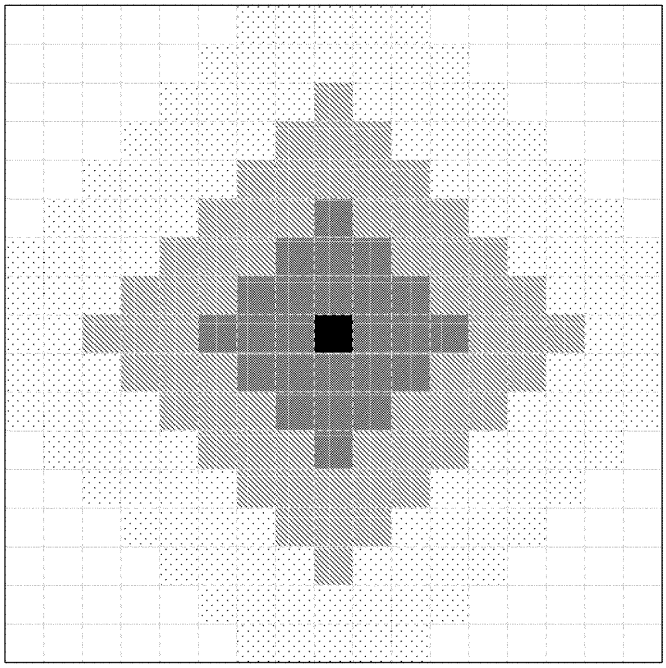
도면28


초기
움직임 정보
탐색할 후보 설정
후보 평가
보정된 움직임 정보 획득
모델 기반 fractional
MVD 최적화를 이용하여
최종 보정된 움직임 정보 계산
최종 보정된
움직임 정보

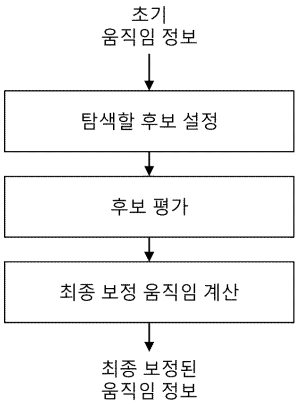
도면29


이전 단계에서
보정된 움직임 정보에 대한
픽셀 기반 Cost 계산
S2901
다음으로 탐색할 후보에 대한
보정값 설정
S2902
움직임 resolution에 맞게
보정값 재설정
S2903
평가할 L0 및 L1
움직임 정보 재구성
S2904
움직임 정보 기반 Cost 계산
S2905
보정된 움직임 정
보 Cost보다 적나?
S2906
no
yes
픽셀 기반 Cost 계산
S2907
모든 탐색 후보를
검사했나?
S2908
no
yes
최소 Cost를 가지는 움직임 정보
를 최종 보정 움직임으로 저장
S2909
모델 기반 fractional
MVD 최적화를 이용하여
최종 보정 움직임 정보 계산
S2910
끝

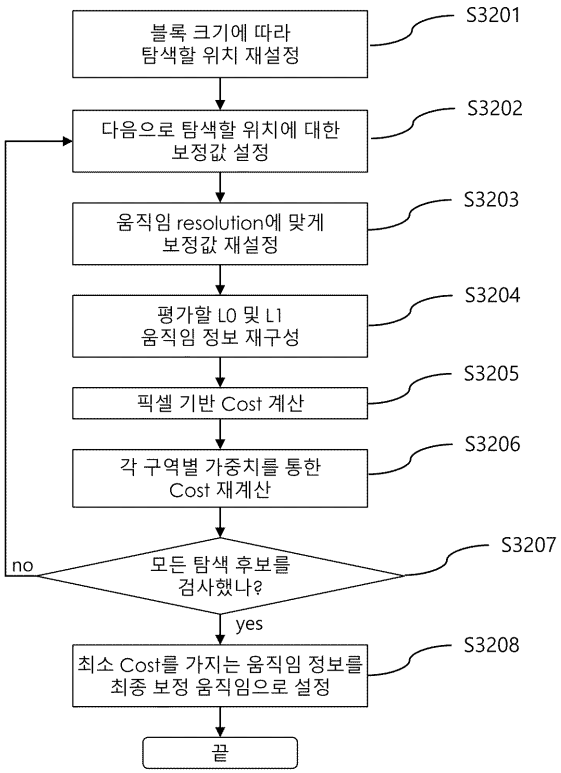
도면30



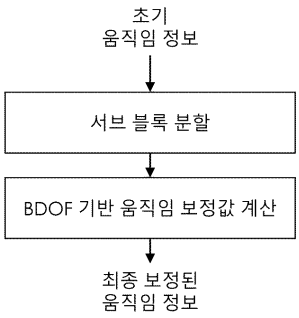
도면31



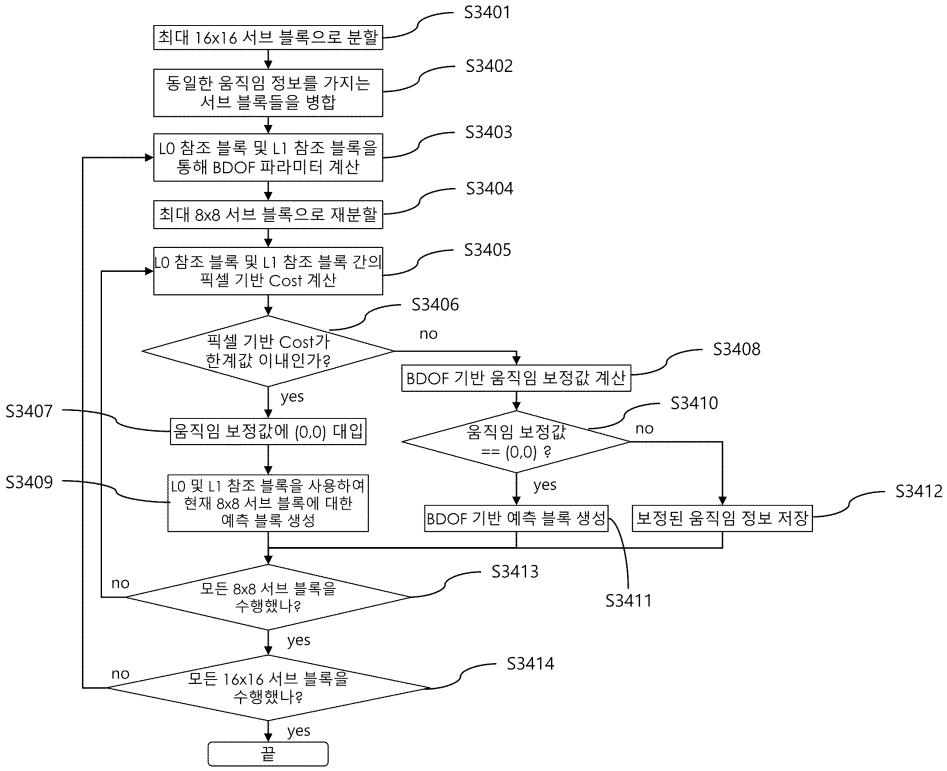
도면32


블록 크기에 따라
탐색할 위치 재설정
S3201
다음으로 탐색할 위치에 대한
보정값 설정
S3202
움직임 resolution에 맞게
보정값 재설정
S3203
평가할 L0 및 L1
움직임 정보 재구성
S3204
픽셀 기반 Cost 계산
S3205
각 구역별 가중치를 통한
Cost 재계산
S3206
모든 탐색 후보를
검사했나?
S3207
no
yes
최소 Cost를 가지는 움직임 정보를
최종 보정 움직임으로 설정
S3208
끝

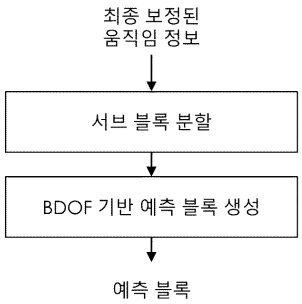
도면33


초기
움직임 정보
서브 블록 분할
BDOF 기반 움직임 보정값 계산
최종 보정된
움직임 정보

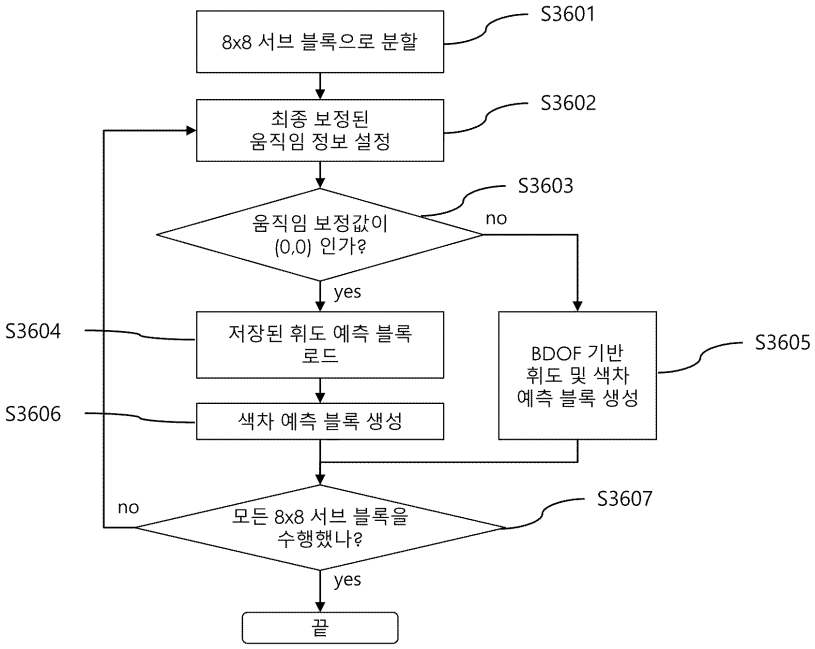
도면34


최대 16x16 서브 블록으로 분할
S3401
동일한 움직임 정보를 가지는 서브 블록들을 병합
S3402
L0 참조 블록 및 L1 참조 블록을 통해 BDOF 파라미터 계산
S3403
최대 8x8 서브 블록으로 재분할
S3404
L0 참조 블록 및 L1 참조 블록 간의 픽셀 기반 Cost 계산
S3405
픽셀 기반 Cost가 한계값 이내인가?
S3406
no
yes
BDOF 기반 움직임 보정값 계산
S3408
움직임 보정값에 (0,0) 대입
S3407
움직임 보정값 == (0,0) ?
S3410
no
yes
L0 및 L1 참조 블록을 사용하여 현재 8x8 서브 블록에 대한 예측 블록 생성
S3409
BDOF 기반 예측 블록 생성
보정된 움직임 정보 저장
S3412
S3411
모든 8x8 서브 블록을 수행했나?
S3413
no
yes
모든 16x16 서브 블록을 수행했나?
S3414
no
yes
끝

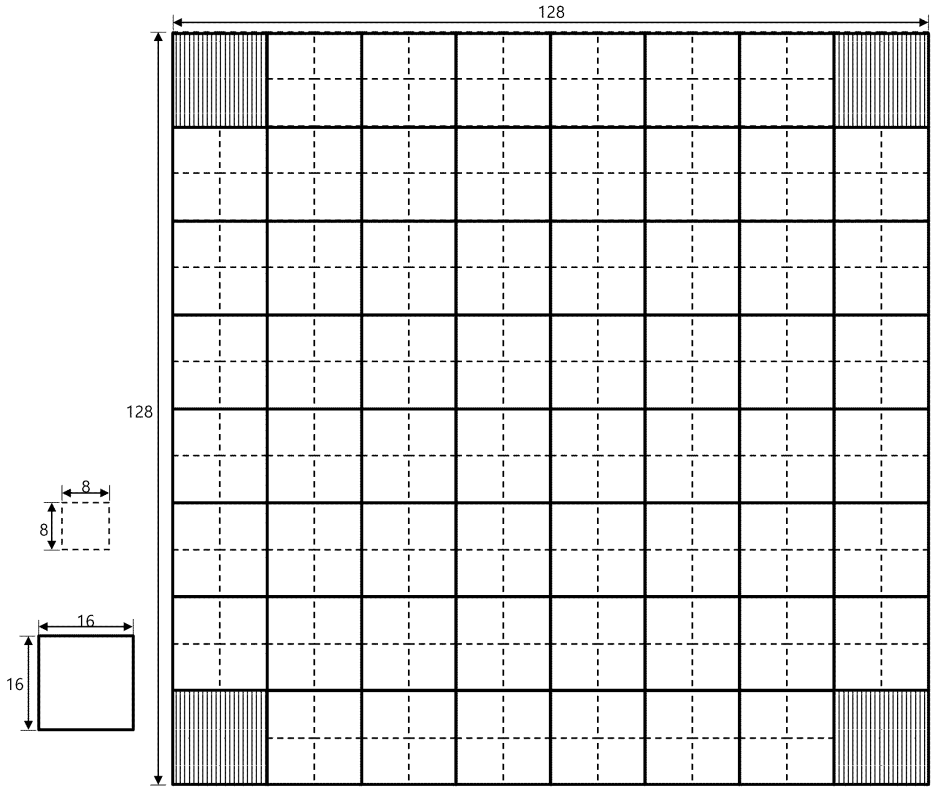
도면35


최종 보정된 움직임 정보
서브 블록 분할
BDOF 기반 예측 블록 생성
예측 블록

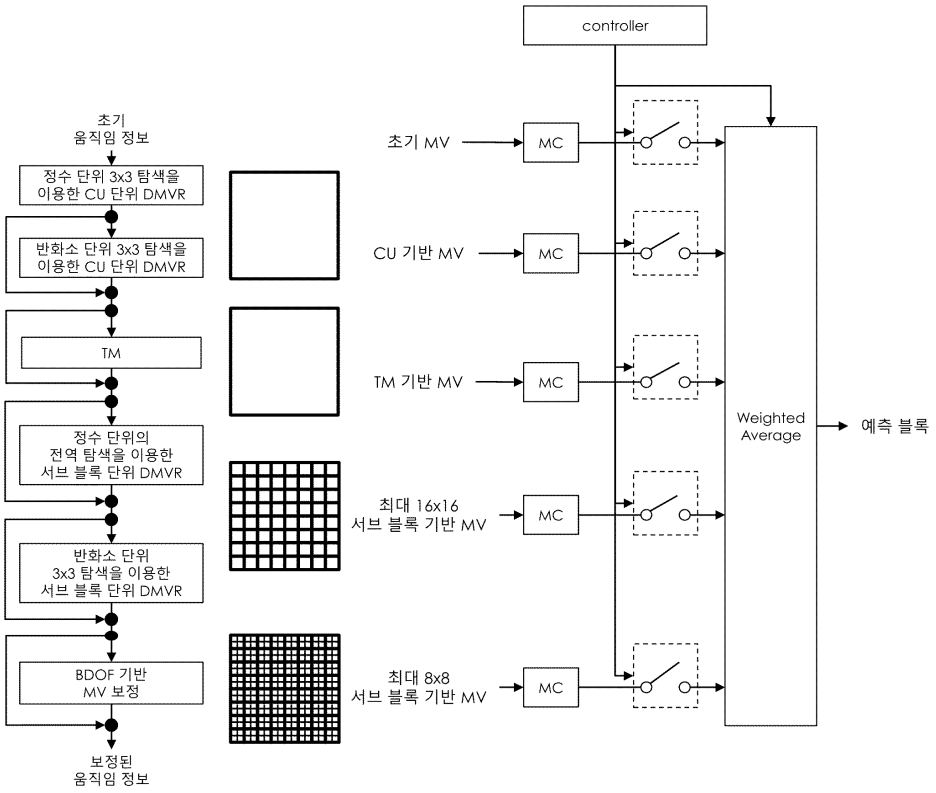
도면36


8x8 서브 블록으로 분할
S3601
최종 보정된
움직임 정보 설정
S3602
움직임 보정값이
(0,0) 인가?
S3603
no
yes
S3604
저장된 휘도 예측 블록
로드
BDOF 기반
휘도 및 색차
예측 블록 생성
S3605
S3606
색차 예측 블록 생성
no
모든 8x8 서브 블록을
수행했나?
S3607
yes
끝

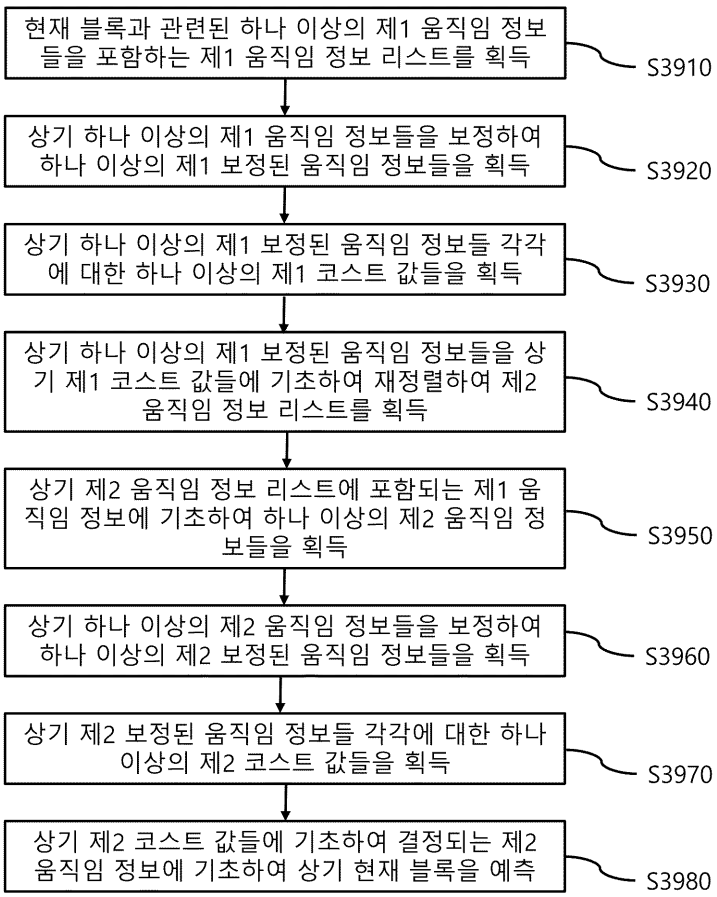
도면37


128
128
8
8
16
16

도면38


controller
초기
움직임 정보
정수 단위 3x3 탐색을
이용한 CU 단위 DMVR
반화소 단위 3x3 탐색을
이용한 CU 단위 DMVR
TM
정수 단위의
전역 탐색을 이용한
서브 블록 단위 DMVR
반화소 단위
3x3 탐색을 이용한
서브 블록 단위 DMVR
BDOF 기반
MV 보정
보정된
움직임 정보
초기 MV
CU 기반 MV
TM 기반 MV
최대 16x16
서브 블록 기반 MV
최대 8x8
서브 블록 기반 MV
MC
MC
MC
MC
MC
Weighted
Average
예측 블록

도면39


현재 블록과 관련된 하나 이상의 제1 움직임 정보들을 포함하는 제1 움직임 정보 리스트를 획득
S3910
상기 하나 이상의 제1 움직임 정보들을 보정하여 하나 이상의 제1 보정된 움직임 정보들을 획득
S3920
상기 하나 이상의 제1 보정된 움직임 정보들 각각에 대한 하나 이상의 제1 코스트 값들을 획득
S3930
상기 하나 이상의 제1 보정된 움직임 정보들을 상기 제1 코스트 값들에 기초하여 재정렬하여 제2 움직임 정보 리스트를 획득
S3940
상기 제2 움직임 정보 리스트에 포함되는 제1 움직임 정보에 기초하여 하나 이상의 제2 움직임 정보들을 획득
S3950
상기 하나 이상의 제2 움직임 정보들을 보정하여 하나 이상의 제2 보정된 움직임 정보들을 획득
S3960
상기 제2 보정된 움직임 정보들 각각에 대한 하나 이상의 제2 코스트 값들을 획득
S3970
상기 제2 코스트 값들에 기초하여 결정되는 제2 움직임 정보에 기초하여 상기 현재 블록을 예측
S3980