(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2021년12월31일
(11) 등록번호 10-2345442
(24) 등록일자 2021년12월27일

(51) 국제특허분류(Int. Cl.)
H04N 19/513 (2014.01) *H04N 19/119* (2014.01)
H04N 19/176 (2014.01) *H04N 19/44* (2014.01)
H04N 19/70 (2014.01)
(52) CPC특허분류
H04N 19/513 (2015.01)
H04N 19/119 (2015.01)
(21) 출원번호 10-2020-7004148
(22) 출원일자(국제) 2018년08월03일
심사청구일자 2020년02월12일
(85) 번역문제출일자 2020년02월12일
(65) 공개번호 10-2020-0022506
(43) 공개일자 2020년03월03일
(86) 국제출원번호 PCT/KR2018/008845
(87) 국제공개번호 WO 2019/027286
국제공개일자 2019년02월07일
(30) 우선권주장
62/541,083 2017년08월03일 미국(US)

(56) 선행기술조사문헌
Feng Zou, Jianle Chen, ET. AL., "Improved affine motion prediction", Joint Video Exploration Team (JVET) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 3rd Meeting: Geneva, CH, 2016.05.26., Document: JVET-C0062_v2*
Jianle Chen, ET. AL., "Algorithm Description of Joint Exploration Test Model 3", Joint Video Exploration Team (JVET) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 3rd Meeting: Geneva, CH, 2016.07.06., Document: JVET-C1001_v3*
*는 심사관에 의하여 인용된 문헌

(73) 특허권자
엘지전자 주식회사
서울특별시 영등포구 여의대로 128 (여의도동)
(72) 발명자
이재호
서울특별시 서초구 양재대로11길 19, LG전자 특허센터
(74) 대리인
특허법인로얄

전체 청구항 수 : 총 7 항 심사관 : 박상철

(54) 발명의 명칭 **어파인 예측을 이용하여 비디오 신호를 처리하는 방법 및 장치**

(57) 요 약

본 발명은, 어파인 움직임 예측 모드(AF 모드, Affine mode)에 기초하여 현재 블록을 포함하는 비디오 신호를 디코딩하는 방법에 있어서, 상기 현재 블록에 대해 상기 AF 모드가 적용되는지 여부를 확인하는 단계, 여기서 상기 AF 모드는 어파인 움직임 모델(affine motion model)을 이용하는 움직임 예측 모드를 나타냄; 상기 현재 블록에 대해 상기 AF 모드가 적용되는 경우, AF4 모드가 이용되는지 여부를 확인하는 단계, 여기서 상기 AF4 모드는 상기 어파인 움직임 모델(affine motion model)을 구성하는 4개의 파라미터를 이용하여 움직임 벡터를 예측하는 모드를 나타냄; AF4 모드가 이용되면 상기 4개 파라미터를 이용하여 움직임 벡터 예측자를 생성하고, AF4 모드가 이용되지 않으면 상기 어파인 움직임 모델(affine motion model)을 구성하는 6개 파라미터를 이용하여 움직임 벡터 예측자를 생성하는 단계; 및 상기 움직임 벡터 예측자에 기초하여 상기 현재 블록의 움직임 벡터를 획득하는 단계를 포함하는 것을 특징으로 하는 방법을 제공한다.

(52) CPC특허분류
H04N 19/176 (2015.01)
H04N 19/44 (2015.01)
H04N 19/70 (2015.01)

명 세 서

청구범위

청구항 1

어파인 움직임 예측 모드(AF 모드, Affine mode)에 기초하여 현재 블록을 포함하는 비디오 신호를 디코딩하는 방법에 있어서,

상기 비디오 신호로부터 머지 플래그를 획득하는 단계, 여기서 상기 머지 플래그는 움직임 파라미터들이 이웃 블록으로부터 유도되는지 여부를 나타냄;

상기 움직임 파라미터들이 이웃 블록으로부터 유도되지 않는 것에 기초하여, 상기 현재 블록의 너비 및 높이가 16 이상인지 여부를 확인하는 단계;

상기 현재 블록의 너비 및 높이가 16 이상인 것에 기초하여 상기 비디오 신호로부터 어파인 플래그를 획득하는 단계, 여기서 상기 어파인 플래그는 상기 현재 블록에 대해 상기 AF 모드가 적용되는지 여부를 나타내고, 상기 AF 모드는 어파인 움직임 모델(affine motion model)을 이용하는 움직임 예측 모드를 나타냄;

상기 현재 블록에 대해 상기 AF 모드가 적용되는 것에 기초하여, 상기 어파인 움직임 모델에 4개 파라미터들 또는 6개 파라미터들이 이용되는지 여부를 나타내는 어파인 파라미터 플래그를 획득하는 단계;

상기 어파인 움직임 모델에 이용되는 상기 4개 파라미터들 또는 상기 6개 파라미터들에 기초하여 움직임 벡터 예측자를 획득하는 단계; 및

상기 움직임 벡터 예측자에 기초하여 상기 현재 블록에 대해 예측을 수행하는 단계

를 포함하되,

상기 머지 플래그가 상기 움직임 파라미터들이 이웃 블록으로부터 유도되지 않는 것을 나타내는 것에 기초하여, 상기 어파인 플래그 및 상기 어파인 파라미터 플래그가 획득되는 것을 특징으로 하는 방법.

청구항 2

삭제

청구항 3

삭제

청구항 4

제1항에 있어서,

상기 어파인 플래그 및 상기 어파인 파라미터 플래그는 슬라이스, 최대 코딩 유닛, 코딩 유닛 또는 예측 유닛 중 적어도 하나의 레벨에서 정의되는 것을 특징으로 하는 방법.

청구항 5

삭제

청구항 6

제1항에 있어서,

상기 현재 블록의 너비 및 높이가 16 보다 작은 것에 기초하여, 상기 현재 블록은 상기 AF 모드가 아닌 다른 코딩 모드에 기초하여 디코딩되는 것을 특징으로 하는 방법.

청구항 7

삭제

청구항 8

어파인 움직임 예측 모드(AF 모드)에 기초하여 현재 블록을 포함하는 비디오 신호를 디코딩하는 장치에 있어서,

상기 비디오 신호로부터 머지 플래그를 획득하고, 여기서 상기 머지 플래그는 움직임 파라미터들이 이웃 블록으로부터 유도되는지 여부를 나타내고,

상기 움직임 파라미터들이 이웃 블록으로부터 유도되지 않는 것에 기초하여 상기 현재 블록의 너비 및 높이가 16 이상인지 여부를 확인하고,

상기 현재 블록의 너비 및 높이가 16 이상인 것에 기초하여 상기 비디오 신호로부터 어파인 플래그를 획득하고, 여기서 상기 어파인 플래그는 상기 현재 블록에 대해 상기 AF 모드가 적용되는지 여부를 나타내고, 상기 AF 모드는 어파인 움직임 모델(affine motion model)을 이용하는 움직임 예측 모드를 나타내고,

상기 현재 블록에 대해 상기 AF 모드가 적용되는 것에 기초하여 상기 어파인 움직임 모델에 4개 파라미터들 또는 6개 파라미터들이 이용되는지 여부를 나타내는 어파인 파라미터 플래그를 획득하고,

상기 어파인 움직임 모델에 이용되는 상기 4개 파라미터들 또는 상기 6개 파라미터들에 기초하여 움직임 벡터 예측자를 획득하고,

상기 움직임 벡터 예측자에 기초하여 상기 현재 블록에 대해 예측을 수행하는 프로세서

를 포함하되,

상기 머지 플래그가 상기 움직임 파라미터들이 이웃 블록으로부터 유도되지 않는 것을 나타내는 것에 기초하여, 상기 어파인 플래그 및 상기 어파인 파라미터 플래그가 획득되는 것을 특징으로 하는 장치.

청구항 9

삭제

청구항 10

삭제

청구항 11

삭제

청구항 12

삭제

청구항 13

삭제

청구항 14

삭제

청구항 15

제1항에 있어서,

상기 움직임 파라미터들은 움직임 벡터 및 참조 픽쳐 인덱스를 포함하는 것을 특징으로 하는 방법.

청구항 16

어파인 움직임 예측 모드(AF 모드)에 기초하여 현재 블록을 포함하는 비디오 신호를 인코딩하는 방법에 있어서,

상기 비디오 신호로부터 머지 플래그를 생성하는 단계, 여기서 상기 머지 플래그는 움직임 파라미터들이 이웃 블록으로부터 유도되는지 여부를 나타냄;

상기 움직임 파라미터들이 이웃 블록으로부터 유도되지 않는 것에 기초하여 상기 현재 블록의 너비 및 높이가 16 이상인지 여부를 확인하는 단계;

상기 현재 블록의 너비 및 높이가 16 이상인 것에 기초하여 상기 비디오 신호로부터 어파인 플래그를 생성하는 단계, 여기서 상기 어파인 플래그는 상기 현재 블록에 대해 상기 AF 모드가 적용되는지 여부를 나타내고, 상기 AF 모드는 어파인 움직임 모델(affine motion model)을 이용하는 움직임 예측 모드를 나타냄;

상기 현재 블록에 대해 상기 AF 모드가 적용되는 것에 기초하여 상기 어파인 움직임 모델에 4개 파라미터들 또는 6개 파라미터들이 이용되는지 여부를 나타내는 어파인 파라미터 플래그를 생성하는 단계;

상기 어파인 움직임 모델에 이용되는 상기 4개 파라미터들 또는 상기 6개 파라미터들에 기초하여 움직임 벡터 예측자를 획득하는 단계; 및

상기 움직임 벡터 예측자에 기초하여 상기 현재 블록에 대해 예측을 수행하는 단계를 포함하되,

상기 머지 플래그가 상기 움직임 파라미터들이 이웃 블록으로부터 유도되지 않는 것을 나타내는 것에 기초하여, 상기 어파인 플래그 및 상기 어파인 파라미터 플래그가 생성되는 것을 특징으로 하는 방법.

청구항 17

컴퓨터 판독 가능한 디지털 저장 매체로서, 청구항 1항에 기재된 비디오 디코딩 방법을 수행하도록 야기하는 비트스트림이 저장된 디지털 저장 매체.

발명의 설명

기 술 분 야

[0001] 본 발명은 비디오 신호의 인코딩/디코딩 방법 및 장치에 관한 것이며, 보다 구체적으로 어파인 예측(affine prediction)을 적응적으로 수행하는 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. 영상, 이미지, 음성 등의 미디어가 압축 부호화의 대상이 될 수 있으며, 특히 영상을 대상으로 압축 부호화를 수행하는 기술을 비디오 영상 압축이라고 일컫는다.

[0003] 차세대 비디오 컨텐츠는 고해상도(high spatial resolution), 고프레임율(high frame rate) 및 영상 표현의 고차원화(high dimensionality of scene representation)라는 특징을 갖게 될 것이다. 그러한 컨텐츠를 처리하기 위해서는 메모리 저장(memory storage), 메모리 액세스율(memory access rate) 및 처리 전력(processing power) 측면에서 엄청난 증가를 가져올 것이다.

[0004] 따라서, 차세대 비디오 컨텐츠를 보다 효율적으로 처리하기 위한 코딩 툴을 디자인할 필요가 있다.

발명의 내용

해결하려는 과제

[0005] 본 발명은 보다 효율적으로 비디오 신호를 인코딩, 디코딩하는 방법을 제안하고자 한다.

[0006] 또한, 본 발명은 4개 파라미터를 이용하는 어파인 예측(Four parameter affine prediction) 모드인 AF4 모드 및 6개 파라미터를 이용하는 어파인 예측(six parameter affine prediction) 모드인 AF6 모드를 모두 고려해서 인코딩 또는 디코딩하는 방법을 제안하고자 한다.

[0007] 또한, 본 발명은 블록 크기에 기초하여 AF4 모드 또는 AF6 모드 중 적어도 하나에 기초하여 최적의 코딩 모드를 적응적으로 결정(또는 선택)하는 방법을 제안하고자 한다.

[0008] 또한, 본 발명은 이웃 블록이 어파인 예측에 의해 코딩되었는지 여부에 기초하여 AF4 모드 또는 AF6 모드 중 적어도 하나에 기초하여 최적의 코딩 모드를 적응적으로 결정(또는 선택)하는 방법을 제안하고자 한다.

과제의 해결 수단

[0009] 상기의 기술적 과제를 해결하기 위해,

[0010] 본 발명은, 블록 크기에 기초하여 어파인 예측(affine prediction)을 적응적으로 수행하는 방법을 제공한다.

[0011] 또한, 본 발명은 이웃 블록이 어파인 예측에 의해 코딩되었는지 여부에 기초하여 어파인 예측(affine prediction)을 적응적으로 수행하는 방법을 제공한다.

[0012] 또한, 본 발명은 AF4 모드 또는 AF6 모드 중 적어도 하나에 기초하여 최적의 코딩 모드를 적응적으로 결정(또는 선택)하는 방법을 제공한다.

[0013] 또한, 본 발명은 적어도 하나의 기설정된 조건을 만족하는지 여부에 기초하여 어파인 예측(affine prediction)을 적응적으로 수행하는 방법을 제공하며, 이 경우 상기 기설정된 조건은 블록 크기, 블록의 픽셀 개수, 블록의 너비, 블록의 높이, 이웃 블록이 어파인 예측에 의해 코딩되었는지 여부 중 적어도 하나를 포함할 수 있다.

발명의 효과

[0014] 본 발명은 어파인 예측(affine prediction)을 적응적으로 수행하는 방법을 제공함으로서, 어파인 예측(affine prediction)의 성능을 향상시킬 수 있고, 어파인 예측(affine prediction)의 복잡도를 감소시킴으로써 보다 효율적인 코딩을 수행할 수 있다.

도면의 간단한 설명

[0015] 도 1은 본 발명이 적용되는 실시예로서, 비디오 신호의 인코딩이 수행되는 인코더의 개략적인 블록도를 나타낸다.

도 2는 본 발명이 적용되는 실시예로서, 비디오 신호의 디코딩이 수행되는 디코더의 개략적인 블록도를 나타낸다.

도 3은 본 발명이 적용될 수 있는 실시예로서, QT(QuadTree, 이하 'QT'라 함) 블록 분할 구조를 설명하기 위한 도면이다.

도 4는 본 발명이 적용될 수 있는 실시예로서, BT(Binary Tree, 이하 'BT'라 함) 블록 분할 구조를 설명하기 위한 도면이다.

도 5는 본 발명이 적용될 수 있는 실시예로서, TT(Ternary Tree, 이하 'TT'라 함) 블록 분할 구조를 설명하기 위한 도면이다.

도 6은 본 발명이 적용될 수 있는 실시예로서, AT(Asymmetric Tree, 이하 'AT'라 함) 블록 분할 구조를 설명하기 위한 도면이다.

도 7은 본 발명이 적용되는 실시예로서, 인터 예측 모드를 설명하기 위한 도면이다.

도 8은 본 발명이 적용되는 실시예로서, 어파인 움직임 모델(affine motion model)을 설명하기 위한 도면이다.

도 9는 본 발명이 적용되는 실시예로서, 제어점 움직임 벡터(control point motion vector)를 이용한 어파인 움직임 예측 방법을 설명하기 위한 도면이다.

도 10은 본 발명이 적용되는 실시예로서, 어파인 예측 모드(Affine prediction mode)를 이용하여 현재 블록을 포함하는 비디오 신호를 처리하는 과정을 설명하는 흐름도이다.

도 11은 본 발명이 적용되는 실시예(1-1)로서, AF4 모드 또는 AF6 모드 중 적어도 하나에 기초하여 최적의 코딩 모드를 적응적으로 결정하는 흐름도를 나타낸다.

도 12는 본 발명이 적용되는 실시예(1-2)로서, AF4 모드 또는 AF6 모드에 기초하여 적응적으로 디코딩을 수행하는 흐름도를 나타낸다.

도 13은 본 발명이 적용되는 실시예(1-3)로서, AF4 모드 또는 AF6 모드에 기초하여 디코딩을 수행하는 신택스 구조를 나타낸다.

도 14는 본 발명이 적용되는 실시예(2-1)로서, 조건 A(condition A)에 기초하여 AF4 모드 또는 AF6 모드를 포함하는 움직임 벡터 예측 모드들 중 최적의 코딩 모드를 적응적으로 결정하는 흐름도를 나타낸다.

도 15는 본 발명이 적용되는 실시예(2-2)로서, 조건 A(condition A)에 기초하여 AF4 모드 또는 AF6 모드에 따라

적응적으로 디코딩을 수행하는 흐름도를 나타낸다.

도 16은 본 발명이 적용되는 실시예(2-3)로서, 조건 A(condition A)에 기초하여 AF4 모드 또는 AF6 모드에 따라 디코딩을 수행하는 신택스 구조를 나타낸다.

도 17은 본 발명이 적용되는 실시예(3-1)로서, 조건 B(condition B) 또는 조건 C(condition C) 중 적어도 하나에 기초하여, AF4 모드 또는 AF6 모드를 포함하는 움직임 벡터 예측 모드들 중 최적의 코딩 모드를 적응적으로 결정하는 흐름도를 나타낸다.

도 18은 본 발명이 적용되는 실시예(3-2)로서, 조건 B(condition B) 또는 조건 C(condition C) 중 적어도 하나에 기초하여, AF4 모드 또는 AF6 모드에 따라 적응적으로 디코딩을 수행하는 흐름도를 나타낸다.

도 19는 본 발명이 적용되는 실시예(3-3)로서, 조건 B(condition B) 또는 조건 C(condition C) 중 적어도 하나에 기초하여, AF4 모드 또는 AF6 모드에 따라 디코딩을 수행하는 신택스 구조를 나타낸다.

도 20은 본 발명이 적용되는 실시예(4-1)로서, 이웃 블록의 코딩 모드에 기초하여 AF4 모드 또는 AF6 모드를 포함하는 움직임 벡터 예측 모드들 중 최적의 코딩 모드를 적응적으로 결정하는 흐름도를 나타낸다.

도 21은 본 발명이 적용되는 실시예(4-2)로서, 이웃 블록의 코딩 모드에 기초하여 AF4 모드 또는 AF6 모드에 따라 적응적으로 디코딩을 수행하는 흐름도를 나타낸다.

도 22는 본 발명이 적용되는 실시예(4-3)로서, 이웃 블록의 코딩 모드에 기초하여 AF4 모드 또는 AF6 모드에 따라 디코딩을 수행하는 신택스 구조를 나타낸다.

도 23은 본 발명이 적용되는 실시예(5-1)로서, 조건 A(condition A), 조건 B(condition B) 또는 조건 C(condition C) 중 적어도 하나에 기초하여, AF4 모드 또는 AF6 모드를 포함하는 움직임 벡터 예측 모드들 중 최적의 코딩 모드를 적응적으로 결정하는 흐름도를 나타낸다.

도 24는 본 발명이 적용되는 실시예(5-2)로서, 조건 A(condition A), 조건 B(condition B) 또는 조건 C(condition C) 중 적어도 하나에 기초하여, AF4 모드 또는 AF6 모드에 따라 적응적으로 디코딩을 수행하는 흐름도를 나타낸다.

도 25은 본 발명이 적용되는 실시예(5-3)로서, 조건 A(condition A), 조건 B(condition B) 또는 조건 C(condition C) 중 적어도 하나에 기초하여, AF4 모드 또는 AF6 모드에 따라 디코딩을 수행하는 신택스 구조를 나타낸다.

도 26은 본 발명이 적용되는 실시예(6-1)로서, 조건 A(condition A) 또는 이웃 블록의 코딩 모드 중 적어도 하나에 기초하여 AF4 모드 또는 AF6 모드를 포함하는 움직임 벡터 예측 모드들 중 최적의 코딩 모드를 적응적으로 결정하는 흐름도를 나타낸다.

도 27은 본 발명이 적용되는 실시예(6-2)로서, 조건 A(condition A) 또는 이웃 블록의 코딩 모드 중 적어도 하나에 기초하여 AF4 모드 또는 AF6 모드에 따라 적응적으로 디코딩을 수행하는 흐름도를 나타낸다.

도 28은 본 발명이 적용되는 실시예(6-3)로서, 조건 A(condition A) 또는 이웃 블록의 코딩 모드 중 적어도 하나에 기초하여 AF4 모드 또는 AF6 모드에 따라 디코딩을 수행하는 신택스 구조를 나타낸다.

도 29는 본 발명이 적용되는 실시예로서, AF4 모드 또는 AF6 모드 중 적어도 하나에 기초하여 움직임 벡터 예측자를 생성하는 흐름도를 나타낸다.

도 30은 본 발명이 적용되는 실시예로서, AF4_flag 및 AF6_flag 에 기초하여 움직임 벡터 예측자를 생성하는 흐름도를 나타낸다.

도 31은 본 발명이 적용되는 실시예로서, 이웃 블록이 AF 모드로 코딩되었는지 여부에 기초하여 AF4 모드 또는 AF6 모드에 따라 적응적으로 디코딩을 수행하는 흐름도를 나타낸다.

도 32는 본 발명이 적용되는 실시예로서, AF4_flag 및 AF6_flag 에 기초하여 적응적으로 디코딩을 수행하는 신택스를 나타낸다.

도 33은 본 발명이 적용되는 실시예로서, 이웃 블록이 AF 모드로 코딩되었는지 여부에 기초하여 AF4 모드 또는 AF6 모드에 따라 적응적으로 디코딩을 수행하는 신택스를 나타낸다.

도 34는 본 발명이 적용되는 비디오 코딩 시스템을 나타낸다.

도 35는 본 발명이 적용되는 컨텐츠 스트리밍 시스템을 나타낸다.

발명을 실시하기 위한 구체적인 내용

[0016] **발명의 실시를 위한 최선의 형태**

[0017] 본 발명은, 어파인 움직임 예측 모드(AF 모드, Affine mode)에 기초하여 현재 블록을 포함하는 비디오 신호를 디코딩하는 방법에 있어서, 상기 현재 블록에 대해 상기 AF 모드가 적용되는지 여부를 확인하는 단계, 여기서 상기 AF 모드는 어파인 움직임 모델(affine motion model)을 이용하는 움직임 예측 모드를 나타냄; 상기 현재 블록에 대해 상기 AF 모드가 적용되는 경우, AF4 모드가 이용되는지 여부를 확인하는 단계, 여기서 상기 AF4 모드는 상기 어파인 움직임 모델(affine motion model)을 구성하는 4개의 파라미터를 이용하여 움직임 벡터를 예측하는 모드를 나타냄; AF4 모드가 이용되면 상기 4개 파라미터를 이용하여 움직임 벡터 예측자를 생성하고, AF4 모드가 이용되지 않으면 상기 어파인 움직임 모델(affine motion model)을 구성하는 6개 파라미터를 이용하여 움직임 벡터 예측자를 생성하는 단계; 및 상기 움직임 벡터 예측자에 기초하여 상기 현재 블록의 움직임 벡터를 획득하는 단계를 포함하는 것을 특징으로 하는 방법을 제공한다.

[0018] 본 발명에서, 상기 방법은, 상기 비디오 신호로부터 어파인 플래그를 획득하는 단계를 더 포함하되, 상기 어파인 플래그는 상기 현재 블록에 대해 상기 AF 모드가 적용되는지 여부를 나타내고, 상기 현재 블록에 대해 상기 AF 모드가 적용되는지 여부는 상기 어파인 플래그에 기초하여 확인되는 것을 특징으로 한다.

[0019] 본 발명에서, 상기 방법은, 상기 어파인 플래그에 따라 상기 현재 블록에 대해 상기 AF 모드가 적용되는 경우, 상기 비디오 신호로부터 어파인 파라미터 플래그를 획득하는 단계를 더 포함하되, 상기 어파인 파라미터 플래그는 상기 움직임 벡터 예측자가 상기 4개의 파라미터를 이용하여 생성되는지 또는 상기 6개의 파라미터를 이용하여 생성되는지 여부를 나타내는 것을 특징으로 한다.

[0020] 본 발명에서, 상기 어파인 플래그 및 상기 어파인 파라미터 플래그는 슬라이스, 최대 코딩 유닛, 코딩 유닛 또는 예측 유닛 중 적어도 하나의 레벨에서 정의되는 것을 특징으로 한다.

[0021] 본 발명에서, 상기 방법은, 상기 현재 블록의 크기가 기설정된 조건을 만족하는지를 확인하는 단계를 더 포함하되, 상기 기설정된 조건은 상기 현재 블록 내 픽셀 개수, 상기 현재 블록의 너비 및/또는 높이 중 적어도 하나가 기설정된 임계값보다 큰지 여부를 나타내고, 상기 현재 블록의 크기가 기설정된 조건을 만족하는 경우, 상기 현재 블록에 대해 상기 AF 모드가 적용되는지 여부를 확인하는 단계가 수행되는 것을 특징으로 한다.

[0022] 본 발명에서, 상기 현재 블록의 크기가 기설정된 조건을 만족하지 않는 경우, 상기 현재 블록은 상기 AF 모드가 아닌 다른 코딩 모드에 기초하여 디코딩되는 것을 특징으로 한다.

[0023] 본 발명에서, 상기 방법은, 상기 현재 블록에 대해 상기 AF 모드가 적용되는 경우, 이웃 블록에 대해 AF 모드가 적용되었는지 여부를 확인하는 단계를 더 포함하되, 상기 이웃 블록에 대해 AF 모드가 적용된 경우, 움직임 벡터 예측자는 상기 4개 파라미터를 이용하여 생성되고, 상기 이웃 블록에 대해 AF 모드가 적용되지 않은 경우, 상기 AF4 모드가 이용되는지 여부를 확인하는 단계를 수행하는 것을 특징으로 한다.

[0024] 본 발명은, 어파인 움직임 예측 모드(AF 모드)에 기초하여 현재 블록을 포함하는 비디오 신호를 디코딩하는 장치에 있어서, 상기 현재 블록에 대해 상기 AF 모드가 적용되는지 여부를 확인하고, 상기 현재 블록에 대해 상기 AF 모드가 적용되는 경우, AF4 모드가 이용되는지 여부를 확인하고, AF4 모드가 이용되면 4개 파라미터를 이용하여 움직임 벡터 예측자를 생성하고, AF4 모드가 이용되지 않으면 어파인 움직임 모델(affine motion model)을 구성하는 6개 파라미터를 이용하여 움직임 벡터 예측자를 생성하고, 상기 움직임 벡터 예측자에 기초하여 상기 현재 블록의 움직임 벡터를 획득하는 인터 예측부를 포함하되, 상기 AF 모드는 상기 어파인 움직임 모델(affine motion model)을 이용하는 움직임 예측 모드를 나타내고, 상기 AF4 모드는 상기 어파인 움직임 모델(affine motion model)을 구성하는 4개의 파라미터를 이용하여 움직임 벡터를 예측하는 모드를 나타내는 것을 특징으로 하는 장치를 제공한다.

[0025] 본 발명에서, 상기 장치는, 상기 비디오 신호로부터 어파인 플래그를 파싱하는 파싱부를 더 포함하되, 상기 어파인 플래그는 상기 현재 블록에 대해 상기 AF 모드가 적용되는지 여부를 나타내고, 상기 현재 블록에 대해 상기 AF 모드가 적용되는지 여부는 상기 어파인 플래그에 기초하여 확인되는 것을 특징으로 한다.

[0026] 본 발명에서, 상기 장치는, 상기 어파인 플래그에 따라 상기 현재 블록에 대해 상기 AF 모드가 적용되는 경우, 상기 비디오 신호로부터 어파인 파라미터 플래그를 획득하는 상기 파싱부를 포함하되, 상기 어파인 파라미터 플

래그는 상기 움직임 벡터 예측자가 상기 4개의 파라미터를 이용하여 생성되는지 또는 상기 6개의 파라미터를 이용하여 생성되는지 여부를 나타내는 것을 특징으로 한다.

[0027] 본 발명에서, 상기 장치는, 상기 현재 블록의 크기가 기설정된 조건을 만족하는지를 확인하는 상기 인터 예측부를 포함하되, 상기 기설정된 조건은 상기 현재 블록 내 픽셀 개수, 상기 현재 블록의 너비 및/또는 높이 중 적어도 하나가 기설정된 임계값보다 큰지 여부를 나타내고, 상기 현재 블록의 크기가 기설정된 조건을 만족하는 경우, 상기 현재 블록에 대해 상기 AF 모드가 적용되는지 여부를 확인하는 단계가 수행되는 것을 특징으로 한다.

[0028] 본 발명에서, 상기 장치는, 상기 현재 블록에 대해 상기 AF 모드가 적용되는 경우, 이웃 블록에 대해 AF 모드가 적용되었는지 여부를 확인하는 상기 인터 예측부를 포함하되, 상기 이웃 블록에 대해 AF 모드가 적용된 경우, 움직임 벡터 예측자는 상기 4개 파라미터를 이용하여 생성되고, 상기 이웃 블록에 대해 AF 모드가 적용되지 않은 경우, 상기 AF4 모드가 이용되는지 여부를 확인하는 단계를 수행하는 것을 특징으로 한다.

[0029] **발명의 실시를 위한 형태**

[0030] 이하, 첨부된 도면을 참조하여 본 발명의 실시예의 구성과 그 작용을 설명하며, 도면에 의해서 설명되는 본 발명의 구성과 작용은 하나의 실시예로서 설명되는 것이며, 이것에 의해서 본 발명의 기술적 사상과 그 핵심 구성 및 작용이 제한되지는 않는다.

[0031] 아울러, 본 발명에서 사용되는 용어는 가능한 한 현재 널리 사용되는 일반적인 용어를 선택하였으나, 특정한 경우는 출원인이 임의로 선정한 용어를 사용하여 설명한다. 그러한 경우에는 해당 부분의 상세 설명에서 그 의미를 명확히 기재하므로, 본 발명의 설명에서 사용된 용어의 명칭만으로 단순 해석되어서는 안 될 것이며 그 해당 용어의 의미까지 파악하여 해석되어야 함을 밝혀두고자 한다.

[0032] 또한, 본 발명에서 사용되는 용어들은 발명을 설명하기 위해 선택된 일반적인 용어들이나, 유사한 의미를 갖는 다른 용어가 있는 경우 보다 적절한 해석을 위해 대체 가능할 것이다. 예를 들어, 신호, 데이터, 샘플, 픽처, 프레임, 블록 등의 경우 각 코딩 과정에서 적절하게 대체되어 해석될 수 있을 것이다. 또한, 파티셔닝(partitioning), 분해(decomposition), 스플리팅 (splitting) 및 분할(division) 등의 경우에도 각 코딩 과정에서 적절하게 대체되어 해석될 수 있을 것이다.

[0033] 도 1은 본 발명이 적용되는 실시예로서, 비디오 신호의 인코딩이 수행되는 인코더의 개략적인 블록도를 나타낸다.

[0034] 도 1을 참조하면, 인코더(100)는 영상 분할부(110), 변환부(120), 양자화부(130), 역양자화부(140), 역변환부(150), 필터링부(160), 복호 픽처 버퍼(DPB: Decoded Picture Buffer) (170), 인터 예측부(180), 인트라 예측부(185) 및 엔트로피 인코딩부(190)를 포함하여 구성될 수 있다.

[0035] 영상 분할부(110)는 인코더(100)에 입력된 입력 영상(Input image) (또는, 픽처, 프레임)를 하나 이상의 처리 유닛으로 분할할 수 있다. 예를 들어, 상기 처리 유닛은 코딩 트리 유닛(CTU: Coding Tree Unit), 코딩 유닛(CU: Coding Unit), 예측 유닛(PU: Prediction Unit) 또는 변환 유닛(TU: Transform Unit)일 수 있다. 이때, 상기 분할은 QT(QuadTree), BT(Binary Tree), TT(Ternary Tree), AT(Asymmetric Tree) 중 적어도 하나의 방식에 의해 수행될 수 있다.

[0036] 다만, 상기 용어들은 본 발명에 대한 설명의 편의를 위해 사용할 뿐이며, 본 발명은 해당 용어의 정의에 한정되지 않는다. 또한, 본 명세서에서는 설명의 편의를 위해, 비디오 신호를 인코딩 또는 디코딩하는 과정에서 이용되는 단위로써 코딩 유닛이라는 용어를 사용하지만, 본 발명은 그에 한정되지 않으며 발명 내용에 따라 적절하게 해석 가능할 것이다.

[0037] 인코더(100)는 입력 영상 신호에서 인터 예측부(180) 또는 인트라 예측부(185)로부터 출력된 예측 신호(prediction signal)를 감산하여 레지듀얼 신호(residual signal)를 생성할 수 있고, 생성된 레지듀얼 신호는 변환부(120)로 전송된다.

[0038] 변환부(120)는 레지듀얼 신호에 변환 기법을 적용하여 변환 계수(transform coefficient)를 생성할 수 있다. 변환 과정은 정사각형의 동일한 크기를 갖는 픽셀 블록에 적용될 수도 있고, 정사각형이 아닌 가변 크기의 블록에도 적용될 수 있다.

[0039] 양자화부(130)는 변환 계수를 양자화하여 엔트로피 인코딩부(190)로 전송하고, 엔트로피 인코딩부(190)는 양자

화된 신호(quantized signal)를 엔트로피 코딩하여 비트스트림으로 출력할 수 있다.

[0040] 양자화부(130)로부터 출력된 양자화된 신호(quantized signal)는 예측 신호를 생성하기 위해 이용될 수 있다. 예를 들어, 양자화된 신호(quantized signal)는 루프 내의 역양자화부(140) 및 역변환부(150)를 통해 역양자화 및 역변환을 적용함으로써 레지듀얼 신호를 복원할 수 있다. 복원된 레지듀얼 신호를 인터 예측부(180) 또는 인트라 예측부(185)로부터 출력된 예측 신호(prediction signal)에 더함으로써 복원 신호(reconstructed signal)가 생성될 수 있다.

[0041] 한편, 위와 같은 압축 과정에서 인접한 블록들이 서로 다른 양자화 파라미터에 의해 양자화됨으로써 블록 경계가 보이는 열화가 발생될 수 있다. 이러한 현상을 블록킹 열화(blocking artifacts)라고 하며, 이는 화질을 평가하는 중요한 요소 중의 하나이다. 이러한 열화를 줄이기 위해 필터링 과정을 수행할 수 있다. 이러한 필터링 과정을 통해 블록킹 열화를 제거함과 동시에 현재 픽처에 대한 오차를 줄임으로써 화질을 향상시킬 수 있게 된다.

[0042] 필터링부(160)는 복원 신호에 필터링을 적용하여 이를 재생 장치로 출력하거나 복호 픽처 버퍼(170)에 전송한다. 복호 픽처 버퍼(170)에 전송된 필터링된 신호는 인터 예측부(180)에서 참조 픽처로 사용될 수 있다. 이처럼, 필터링된 픽처를 화면간 예측 모드에서 참조 픽처로 이용함으로써 화질 뿐만 아니라 부호화 효율도 향상시킬 수 있다.

[0043] 복호 픽처 버퍼(170)는 필터링된 픽처를 인터 예측부(180)에서의 참조 픽처로 사용하기 위해 저장할 수 있다.

[0044] 인터 예측부(180)는 복원 픽처(reconstructed picture)를 참조하여 시간적 중복성 및/또는 공간적 중복성을 제거하기 위해 시간적 예측 및/또는 공간적 예측을 수행한다. 여기서, 예측을 수행하기 위해 이용되는 참조 픽처는 이전 시간에 부호화/복호화 시 블록 단위로 양자화와 역양자화를 거친 변환된 신호이기 때문에, 블로킹 아티팩트(blocking artifact)나 링잉 아티팩트(ringing artifact)가 존재할 수 있다.

[0045] 따라서, 인터 예측부(180)는 이러한 신호의 불연속이나 양자화로 인한 성능 저하를 해결하기 위해, 로우패스 필터(lowpass filter)를 적용함으로써 픽셀들 사이의 신호를 서브 픽셀 단위로 보간할 수 있다. 여기서, 서브 픽셀은 보간 필터를 적용하여 생성된 가상의 픽셀을 의미하고, 정수 픽셀은 복원된 픽처에 존재하는 실제 픽셀을 의미한다. 보간 방법으로는 선형 보간, 양선형 보간(bi-linear interpolation), 위너 필터(wiener filter) 등이 적용될 수 있다.

[0046] 보간 필터는 복원 픽처(reconstructed picture)에 적용되어 예측의 정밀도를 향상시킬 수 있다. 예를 들어, 인터 예측부(180)는 정수 픽셀에 보간 필터를 적용하여 보간 픽셀을 생성하고, 보간 픽셀들(interpolated pixels)로 구성된 보간 블록(interpolated block)을 예측 블록(prediction block)으로 사용하여 예측을 수행할 수 있다.

[0047] 인트라 예측부(185)는 현재 부호화를 진행하려고 하는 블록의 주변에 있는 샘플들을 참조하여 현재 블록을 예측할 수 있다. 상기 인트라 예측부(185)는 인트라 예측을 수행하기 위해 다음과 같은 과정을 수행할 수 있다. 먼저, 예측 신호를 생성하기 위해 필요한 참조 샘플을 준비할 수 있다. 그리고, 준비된 참조 샘플을 이용하여 예측 신호를 생성할 수 있다. 이후, 예측 모드를 부호화하게 된다. 이때, 참조 샘플은 참조 샘플 패딩 및/또는 참조 샘플 필터링을 통해 준비될 수 있다. 참조 샘플은 예측 및 복원 과정을 거쳤기 때문에 양자화 에러가 존재할 수 있다. 따라서, 이러한 에러를 줄이기 위해 인트라 예측에 이용되는 각 예측 모드에 대해 참조 샘플 필터링 과정이 수행될 수 있다.

[0048] 상기 인터 예측부(180) 또는 상기 인트라 예측부(185)를 통해 생성된 예측 신호(prediction signal)는 복원 신호를 생성하기 위해 이용되거나 레지듀얼 신호를 생성하기 위해 이용될 수 있다.

[0049] 도 2는 본 발명이 적용되는 실시예로서, 비디오 신호의 디코딩이 수행되는 디코더의 개략적인 블록도를 나타낸다.

[0050] 도 2를 참조하면, 디코더(200)는 파싱부(미도시), 엔트로피 디코딩부(210), 역양자화부(220), 역변환부(230), 필터링부(240), 복호 픽처 버퍼(DPB: Decoded Picture Buffer Unit) (250), 인터 예측부(260), 인트라 예측부(265) 및 복원부(미도시)를 포함하여 구성될 수 있다.

[0051] 디코더(200)는 도 1의 인코더(100)로부터 출력된 신호를 수신할 수 있고, 파싱부(미도시)를 통해 신택스 엘리먼트를 파싱 또는 획득할 수 있다. 파싱 또는 획득된 신호는 엔트로피 디코딩부(210)를 통해 엔트로피 디코딩될

수 있다.

[0052] 역양자화부(220)에서는 양자화 스텝 사이즈 정보를 이용하여 엔트로피 디코딩된 신호로부터 변환 계수(transform coefficient)를 획득한다.

[0053] 역변환부(230)에서는 변환 계수를 역변환하여 레지듀얼 신호(residual signal)를 획득하게 된다.

[0054] 복원부(미도시)는 획득된 레지듀얼 신호를 인터 예측부(260) 또는 인트라 예측부(265)로부터 출력된 예측 신호(prediction signal)에 더함으로써 복원 신호(reconstructed signal)를 생성한다.

[0055] 필터링부(240)는 복원 신호(reconstructed signal)에 필터링을 적용하여 이를 재생 장치로 출력하거나 복호 픽처 버퍼부(250)에 전송한다. 복호 픽처 버퍼부(250)에 전송된 필터링된 신호는 인터 예측부(260)에서 참조 픽처로 사용될 수 있다.

[0056] 본 명세서에서, 인코더(100)의 필터링부(160), 인터 예측부(180) 및 인트라 예측부(185)에서 설명된 실시예들은 각각 디코더의 필터링부(240), 인터 예측부(260) 및 인트라 예측부(265)에도 동일하게 적용될 수 있다.

[0057] 상기 디코더(200)를 통해 출력된 복원 영상 신호(reconstructed video signal)는 재생 장치를 통해 재생될 수 있다.

[0058] 도 3은 본 발명이 적용될 수 있는 실시예로서, QT(QuadTree, 이하 'QT'라 함) 블록 분할 구조를 설명하기 위한 도면이다.

[0059] 비디오 코딩에서 하나의 블록은 QT(QuadTree) 기반으로 분할될 수 있다. 또한, QT에 의해서 분할된 하나의 서브 블록 (sub block)은 QT를 사용하여 재귀적으로 더 분할될 수 있다. 더 이상 QT 분할되지 않는 리프 블록 (leaf block)은 BT(Binary Tree), TT(Ternary Tree) 또는 AT(Asymmetric Tree) 중 적어도 하나의 방식에 의해서 분할될 수 있다. BT는 horizontal BT (2NxN, 2NxN)과 vertical BT (Nx2N, Nx2N)의 두 가지 형태의 분할을 가질 수 있다. TT는 horizontal TT (2Nx1/2N, 2NxN, 2Nx1/2N)와 vertical TT (1/2Nx2N, Nx2N, 1/2Nx2N)의 두 가지 형태의 분할을 가질 수 있다. AT는 horizontal-up AT (2Nx1/2N, 2Nx3/2N), horizontal-down AT (2Nx3/2N, 2Nx1/2N), vertical-left AT (1/2Nx2N, 3/2Nx2N), vertical-right AT (3/2Nx2N, 1/2Nx2N) 의 네 가지 형태의 분할을 가질 수 있다. 각각의 BT, TT, AT는 BT, TT, AT를 사용하여 재귀적으로 더 분할될 수 있다.

[0060] 상기 도 3은 QT 분할의 예를 보여준다. 블록 A는 QT에 의해서 4개의 서브 블록 (A0, A1, A2, A3)으로 분할될 수 있다. 서브 블록 A1은 다시 QT에 의해서 4개의 서브 블록 (B0, B1, B2, B3)로 분할될 수 있다.

[0061] 도 4는 본 발명이 적용될 수 있는 실시예로서, BT(Binary Tree, 이하 'BT'라 함) 블록 분할 구조를 설명하기 위한 도면이다.

[0062] 상기 도 4는 BT 분할의 예를 보여준다. QT에 의해서 더 이상 분할되지 않는 블록 B3은 vertical BT (C0, C1) 또는 horizontal BT (D0, D1)으로 분할될 수 있다. 블록 C0와 같이 각각의 서브 블록은 horizontal BT (E0, E1) 또는 vertical BT (F0, F1)의 형태와 같이 재귀적으로 더 분할될 수 있다.

[0063] 도 5는 본 발명이 적용될 수 있는 실시예로서, TT(Ternary Tree, 이하 'TT'라 함) 블록 분할 구조를 설명하기 위한 도면이다.

[0064] 상기 도 5는 TT 분할의 예를 보여준다. QT에 의해서 더 이상 분할되지 않는 블록 B3은 vertical TT (C0, C1, C2) 또는 horizontal TT (D0, D1, D2)으로 분할될 수 있다. 블록 C1와 같이 각각의 서브 블록은 horizontal TT (E0, E1, E2) 또는 vertical TT (F0, F1, F2)의 형태와 같이 재귀적으로 더 분할될 수 있다.

[0065] 도 6은 본 발명이 적용될 수 있는 실시예로서, AT(Asymmetric Tree, 이하 'AT'라 함) 블록 분할 구조를 설명하기 위한 도면이다.

[0066] 상기 도 6은 AT 분할의 예를 보여준다. QT에 의해서 더 이상 분할되지 않는 블록 B3은 vertical AT (C0, C1) 또는 horizontal AT (D0, D1)으로 분할 될 수 있다. 블록 C1와 같이 각각의 서브 블록은 horizontal AT (E0, E1) 또는 vertical TT (F0, F1)의 형태와 같이 재귀적으로 더 분할 될 수 있다.

[0067] 한편, BT, TT, AT 분할은 함께 사용하여 분할이 가능하다. 예를 들어, BT에 의해 분할된 서브 블록은 TT 또는 AT에 의한 분할이 가능하다. 또한, TT에 의해 분할된 서브 블록은 BT 또는 AT에 의한 분할이 가능하다. AT에 의해 분할된 서브 블록은 BT 또는 TT에 의한 분할이 가능하다. 예를 들어, horizontal BT 분할 이후, 각각의 서브 블록이 vertical BT로 분할될 수 있고, 또는 vertical BT 분할 이후, 각각의 서브 블록이 horizontal BT로 분

할될 수도 있다. 상기 두 종류의 분할 방법은 분할 순서는 다르지만 최종적으로 분할되는 모양은 동일하다.

[0068] 또한, 블록이 분할되면 블록을 탐색하는 순서를 다양하게 정의할 수 있다. 일반적으로, 좌측에서 우측으로, 상단에서 하단으로 탐색을 수행하며, 블록을 탐색한다는 것은 각 분할된 서브 블록의 추가적인 블록 분할 여부를 결정하는 순서를 의미하거나, 블록이 더 이상 분할되지 않을 경우 각 서브 블록의 부호화 순서를 의미하거나, 또는 서브 블록에서 다른 이웃 블록의 정보를 참조할 때의 탐색 순서를 의미할 수 있다.

[0069] 도 7은 본 발명이 적용되는 실시예로서, 인터 예측 모드를 설명하기 위한 도면이다.

[0070] 인터 예측 모드

[0071] 본 발명이 적용되는 인터 예측 모드에서는 움직임 정보의 양을 줄이기 위하여 머지(Merge) 모드, AMVP(Advanced Motion Vector Prediction) 모드 또는 어파인 예측 모드(Affine prediction mode, 이하 'AF 모드'라 함)가 이용될 수 있다.

[0072] 1) 머지(Merge) 모드

[0073] 머지(Merge) 모드는 공간적(spatially) 또는 시간적(temporally)으로 이웃하는 블록으로부터 움직임 파라미터(또는 정보)를 도출하는 방법을 의미한다.

[0074] 머지 모드에서 이용 가능한 후보의 세트는 공간적으로 이웃하는 후보(spatial neighbor candidates), 시간적 후보(temporal candidates) 및 생성된 후보(generated candidates)로 구성된다.

[0075] 도 7(a)를 참조하면, {A1, B1, B0, A0, B2}의 순서에 따라 각 공간적 후보 블록이 이용 가능한지 여부가 판단된다. 이때, 후보 블록이 인트라 예측 모드로 인코딩되어 움직임 정보가 존재하지 않는 경우 또는 후보 블록이 현재 픽처(또는 슬라이스)의 밖에 위치하는 경우에는 해당 후보 블록은 이용할 수 없다.

[0076] 공간적 후보의 유효성의 판단 후, 현재 처리 블록의 후보 블록에서 불필요한 후보 블록을 제외함으로써 공간적 머지 후보가 구성될 수 있다. 예를 들어, 현재 예측 블록의 후보 블록이 동일 코딩 블록 내 첫 번째 예측 블록인 경우 해당 후보 블록을 제외하고 또한 동일한 움직임 정보를 가지는 후보 블록들을 제외할 수 있다.

[0077] 공간적 머지 후보 구성이 완료되면, {T0, T1}의 순서에 따라 시간적 머지 후보 구성 과정이 진행된다.

[0078] 시간적 후보 구성에 있어서, 참조 픽처의 동일 위치(collocated) 블록의 우하단(right bottom) 블록(T0)이 이용 가능한 경우, 해당 블록을 시간적 머지 후보로 구성한다. 동일 위치(collocated) 블록은 선택된 참조 픽처에서 현재 처리 블록에 대응되는 위치에 존재하는 블록을 의미한다. 반면, 그렇지 않은 경우, 동일 위치(collocated) 블록의 중앙(center)에 위치하는 블록(T1)을 시간적 머지 후보로 구성한다.

[0079] 머지 후보의 최대 개수는 슬라이스 헤더에서 특정될 수 있다. 머지 후보의 개수가 최대 개수보다 큰 경우, 최대 개수 보다 작은 개수의 공간적 후보와 시간적 후보가 유지된다. 그렇지 않은 경우, 머지 후보의 개수는 후보 개수가 최대 개수가 될 때까지 현재까지 추가된 후보들을 조합하여 추가적인 머지 후보(즉, 조합된 쌍예측 머지 후보(combined bi-predictive merging candidates))가 생성된다.

[0080] 인코더에서는 위와 같은 방법으로 머지 후보 리스트를 구성하고, 움직임 추정(Motion Estimation)을 수행함으로써 머지 후보 리스트에서 선택된 후보 블록 정보를 머지 인덱스(merge index)(예를 들어, merge_idx[x0][y0]')로써 디코더에게 시그널링한다. 도 7(b)에서는 머지 후보 리스트에서 B1 블록이 선택된 경우를 예시하고 있으며, 이 경우, 머지 인덱스(merge index)로 "인덱스 1(Index 1)"이 디코더로 시그널링될 수 있다.

[0081] 디코더에서는 인코더와 동일하게 머지 후보 리스트를 구성하고, 머지 후보 리스트에서 인코더로부터 수신한 머지 인덱스(merge index)에 해당하는 후보 블록의 움직임 정보로부터 현재 블록에 대한 움직임 정보를 도출한다. 그리고, 디코더는 도출한 움직임 정보를 기반으로 현재 처리 블록에 대한 예측 블록을 생성한다.

[0082] 2) AMVP(Advanced Motion Vector Prediction) 모드

[0083] AMVP 모드는 주변 블록으로부터 움직임 벡터 예측 값을 유도하는 방법을 의미한다. 따라서, 수평 및 수직 움직임 벡터 차분 값(MVD: motion vector difference), 참조 인덱스 및 인터 예측 모드가 디코더로 시그널링된다. 수평 및 수직 움직임 벡터 값은 유도된 움직임 벡터 예측 값과 인코더로부터 제공된 움직임 벡터 차분 값(MVD: motion vector difference)를 이용하여 계산된다.

[0084] 즉, 인코더에서는 움직임 벡터 예측값 후보 리스트를 구성하고, 움직임 추정(Motion Estimation)을 수행함으로써 움직임 벡터 예측값 후보 리스트에서 선택된 움직임 참조 플래그(즉, 후보 블록 정보)(예를 들어,

'mvp_lX_flag[x0][y0]')를 디코더에게 시그널링한다. 디코더에서는 인코더와 동일하게 움직임 벡터 예측값 후보 리스트를 구성하고, 움직임 벡터 예측값 후보 리스트에서 인코더로부터 수신한 움직임 참조 플래그에서 지시된 후보 블록의 움직임 정보를 이용하여 현재 처리 블록의 움직임 벡터 예측값을 도출한다. 그리고, 디코더는 도출된 움직임 벡터 예측값과 인코더로부터 전송된 움직임 벡터 차이값을 이용하여 현재 처리 블록에 대한 움직임 벡터값을 획득하게 된다. 그리고, 디코더는 도출한 움직임 정보를 기반으로 현재 처리 블록에 대한 예측 블록을 생성한다(즉, 움직임 보상).

[0085] AMVP 모드의 경우, 앞서 도 7에서 5개의 이용 가능한 후보들 중에서 2개의 공간적 움직임 후보가 선택된다. 첫 번째 공간적 움직임 후보는 좌측에 위치한 {A0, A1} 세트로부터 선택되고, 두 번째 공간적 움직임 후보는 상위에 위치한 {B0, B1, B2} 세트로부터 선택된다. 이때, 이웃한 후보 블록의 참조 인덱스가 현재 예측 블록과 동일하지 않은 경우, 움직임 벡터가 스케일링된다.

[0086] 공간적 움직임 후보의 탐색 결과 선택된 후보 개수가 2개라면 후보 구성을 종료하나, 2개 미만인 경우 시간적 움직임 후보가 추가된다.

[0087] 디코더(예를 들어, 인터 예측부)는 처리 블록(예를 들어, 예측 유닛)에 대한 움직임 파라미터를 디코딩한다.

[0088] 예를 들어, 처리 블록이 머지 모드를 이용하는 경우, 디코더는 인코더로부터 시그널링된 머지 인덱스를 디코딩할 수 있다. 그리고, 머지 인덱스에서 지시된 후보 블록의 움직임 파라미터로부터 현재 처리 블록의 움직임 파라미터를 도출할 수 있다.

[0089] 또한, 처리 블록이 AMVP 모드가 적용된 경우, 디코더는 인코더로부터 시그널링된 수평 및 수직 움직임 벡터 차분 값(MVD: motion vector difference), 참조 인덱스 및 인터 예측 모드를 복호화할 수 있다. 그리고, 움직임 참조 플래그로부터 지시된 후보 블록의 움직임 파라미터로부터 움직임 벡터 예측값을 도출하고, 움직임 벡터 예측값과 수신한 움직임 벡터 차분 값을 이용하여 현재 처리 블록의 움직임 벡터값을 도출할 수 있다.

[0090] 디코더는 디코딩된 움직임 파라미터(또는 정보)를 이용하여 예측 유닛에 대한 움직임 보상을 수행한다.

[0091] 즉, 인코더/디코더에서는 복호화된 움직임 파라미터를 이용하여, 이전에 디코딩된 픽쳐로부터 현재 유닛의 영상을 예측하는 움직임 보상(motion compensation)을 수행한다.

[0092] 3) AF 모드(Affine Mode)

[0093] 상기 AF 모드는 어파인 움직임 모델(affine motion model)을 이용하는 움직임 예측 모드를 의미하며, 어파인 머지 모드(affine merge mode) 또는 어파인 인터 모드(affine inter mode) 중 적어도 하나를 포함할 수 있다. 상기 어파인 인터 모드(affine inter mode)는 AF4 모드 또는 AF6 모드 중 적어도 하나를 포함할 수 있다. 여기서, AF4 모드는 4개 파라미터를 이용하는 어파인 예측(Four parameter affine prediction) 모드를 나타내고, AF6 모드는 6개 파라미터를 이용하는 어파인 예측(six parameter affine prediction) 모드를 나타낸다.

[0094] 다만, 본 발명에서는 설명의 편의상 AF4 모드 또는 AF6 모드라 표현하지만, 이는 반드시 별도의 예측 모드로 정의되어야 할 필요는 없고, 상기 AF4 모드 또는 AF6 모드는 단지 4개 파라미터를 이용하는지 또는 6개 파라미터를 이용하는지에 의해 구분되어 이해될 수 있다.

[0095] 상기 AF 모드에 대해서는 도 8 내지 도 10에서 보다 상세히 설명하도록 한다.

[0096] 도 8은 본 발명이 적용되는 실시예로서, 어파인 움직임 모델(affine motion model)을 설명하기 위한 도면이다.

[0097] 일반적인 영상 부호화 기술은 코딩 블록의 움직임을 표현하기 위하여 병진 움직임 모델(translation motion model)을 사용한다. 여기서, 병진 움직임 모델(translation motion model)은 평행 이동된 블록 기반의 예측 방법을 나타낸다. 즉, 코딩 블록의 움직임 정보는 하나의 움직임 벡터를 이용하여 표현된다. 그러나, 실제 코딩 블록 내에서 각 픽셀별 최적의 움직임 벡터는 서로 다를 수 있다. 만약, 적은 정보만으로 픽셀별 또는 서브 블록 단위별로 최적의 움직임 벡터를 결정할 수 있다면 코딩 효율을 높일 수 있다.

[0098] 따라서, 본 발명은 인터 예측의 성능을 높이기 위해 평행 이동된 블록 기반의 예측 방법뿐만 아니라, 영상의 다양한 움직임을 반영한 인터 예측 기반 영상 처리 방법을 제안한다.

[0099] 또한, 본 발명은 어파인 움직임 모델(affine motion model)을 사용하여 인코딩/디코딩을 수행하는 어파인 움직임 예측 방법에 대해 제안한다. 어파인 움직임 모델은 제어점의 움직임 벡터를 이용하여 픽셀 단위 또는 서브 블록 단위로 움직임 벡터를 유도하는 예측 방법을 나타낸다. 본 명세서에서, 어파인 움직임 모델(affine motion

model)을 이용하는 어파인 움직임 예측 모드를 AF 모드(Affine Mode)라 한다.

[0100] 또한, 본 발명은 블록 크기에 기초하여 어파인 예측(affine prediction)을 적응적으로 수행하는 방법을 제공한다.

[0101] 또한, 본 발명은 이웃 블록이 어파인 예측에 의해 코딩되었는지 여부에 기초하여 어파인 예측(affine prediction)을 적응적으로 수행하는 방법을 제공한다.

[0102] 또한, 본 발명은 AF4 모드 또는 AF6 모드 중 적어도 하나에 기초하여 최적의 코딩 모드를 적응적으로 결정(또는 선택)하는 방법을 제공한다. 여기서, AF4 모드는 4개 파라미터를 이용하는 어파인 예측(Four parameter affine prediction) 모드를 나타내고, AF6 모드는 6개 파라미터를 이용하는 어파인 예측(six parameter affine prediction) 모드를 나타낸다.

[0103] 상기 도 8을 참조하면, 영상의 왜곡을 움직임 정보로서 표현하기 위해 다양한 방법이 사용될 수 있으며, 특히 어파인 움직임 모델은 도 8에 도시된 4가지 움직임을 표현할 수 있다.

[0104] 예를 들어, 어파인 움직임 모델은 영상의 이동(translate), 영상의 확대/축소(scale), 영상의 회전(rotate), 영상의 비뚤림(shear)를 비롯하여 유발되는 임의의 영상 왜곡을 모델링할 수 있다.

[0105] 어파인 움직임 모델은 다양한 방법으로 표현될 수 있으나, 그 중에서 본 발명에서는 블록의 특정 기준점(또는 기준 픽셀/샘플)에서의 움직임 정보를 활용하여 왜곡을 표시(또는 식별)하고, 이를 이용하여 인터 예측을 수행하는 방법을 제안한다. 여기서, 기준점은 제어점(CP: Control Point)(또는 제어 픽셀, 제어 샘플)이라고 지칭될 수 있으며, 이러한 기준점에서의 움직임 벡터는 제어점 움직임 벡터(CPMV: Control Point Motion Vector)라고 지칭될 수 있다. 이러한 제어점의 개수에 따라 표현할 수 있는 왜곡의 정도가 달라질 수 있다.

[0106] 어파인 움직임 모델은 다음 수학식 1과 같이 6개의 파라미터(a, b, c, d, e, f)를 이용하여 표현될 수 있다.

수학식 1

[0107]
$$\begin{cases} v_x = a*x + b*y + c \\ v_y = d*x + e*y + f \end{cases}$$

[0108] 여기서, (x,y)는 코딩 블록의 좌상측 픽셀의 위치를 나타낸다. 그리고, v_x 및 v_y는 각각 (x,y) 에서의 움직임 벡터를 나타낸다.

[0109] 도 9는 본 발명이 적용되는 실시예로서, 제어점 움직임 벡터(control point motion vector)를 이용한 어파인 움직임 예측 방법을 설명하기 위한 도면이다.

[0110] 도 9(a)를 참조하면, 현재 블록(901)의 좌상측 제어점(control point) (CP_0) (902)(이하, 제1 제어점이라 함), 우상측 제어점(control point) (CP_1) (903)(이하, 제2 제어점이라 함) 및 좌하측 제어점(control point) (CP_2) (904)(이하, 제3 제어점이라 함)은 각각 독립적인 움직임 정보를 가질 수 있다. 이를 각각 CP_0, CP_1, CP_2 라 표현할 수 있다. 그러나, 이는 본 발명의 일실시예에 해당하고, 본 발명은 이에 한정되지 않는다. 예를 들어, 우하측 제어점, 센터 제어점, 그외 서브 블록의 위치별 제어점 등 다양하게 제어점을 정의할 수 있다.

[0111] 본 발명의 일실시예로, 상기 제1 제어점 내지 제3 제어점 중 적어도 하나는 현재 블록에 포함된 픽셀일 수 있다. 또는, 다른 예로, 상기 제1 제어점 내지 제3 제어점 중 적어도 하나는 현재 블록에 포함되지 않는 현재 블록에 인접한 픽셀일 수 있다.

[0112] 상기 제어점들 중 하나 이상의 제어점의 움직임 정보를 이용하여 현재 블록(901)의 픽셀 별 또는 서브 블록 별 움직임 정보가 유도될 수 있다.

[0113] 예를 들어, 현재 블록(901)의 좌상측 제어점(902), 우상측 제어점(903) 및 좌하측 제어점(904)의 움직임 벡터를 이용한 어파인 움직임 모델은 다음 수학식 2와 같이 정의될 수 있다.

수학식 2

[0114] $$\begin{cases} v_x = \frac{(v_{1x}-v_{0x})}{w} * x + \frac{(v_{2x}-v_{0x})}{h} * x + v_{0x} \\ v_y = \frac{(v_{1y}-v_{0y})}{w} * x - \frac{(v_{2y}-v_{0y})}{h} * y + v_{0y} \end{cases}$$

[0115] 여기서, $\overrightarrow{v_0}$를 좌상측 제어점(902)의 움직임 벡터, $\overrightarrow{v_1}$를 우상측 제어점(903)의 움직임 벡터, $\overrightarrow{v_2}$를 좌하측 제어점(904)의 움직임 벡터라고 할 때, $\overrightarrow{v_0} = \{v_{0x}, v_{0y}\}$, $\overrightarrow{v_1} = \{v_{1x}, v_{1y}\}$, $\overrightarrow{v_2} = \{v_{2x}, v_{2y}\}$로 정의될 수 있다. 그리고, 수학식 2에서 w는 현재 블록(901)의 너비(width), h는 현재 블록(901)의 높이(height)를 나타낸다. 그리고, $\vec{v} = \{v_x, v_y\}$는 {x,y} 위치의 움직임 벡터를 나타낸다.

[0116] 본 발명에서는, 어파인 움직임 모델이 표현할 수 있는 움직임 중 병진(translation), 스케일(scale), 회전(rotate)의 3가지 움직임을 표현하는 어파인 움직임 모델을 정의할 수 있다. 본 명세서에서는 이를 간이 어파인 움직임 모델(simplified affine motion model or similarity affine motion model)이라 부르기로 한다.

[0117] 상기 간이 어파인 움직임 모델은 다음 수학식 3과 같이 4개의 파라미터(a, b, c, d)를 이용하여 표현될 수 있다.

수학식 3

[0118] $$\begin{cases} v_x = a * x - b * y + c \\ v_y = b * x + a * y + d \end{cases}$$

[0119] 여기서, $\{v_x, v_y\}$는 각각 {x,y} 위치의 움직임 벡터를 나타낸다. 이와 같이 4개의 파라미터를 이용하는 어파인 움직임 모델을 AF4라 부를 수 있다. 본 발명은 이에 한정되지 않으며, 6개의 파라미터를 이용하는 경우에는 AF6라 하며, 위 실시예들을 동일하게 적용할 수 있다.

[0120] 상기 도 9(b)를 살펴보면, $\overrightarrow{v_0}$를 현재 블록의 좌상측 제어점(1001)의 움직임 벡터, $\overrightarrow{v_1}$를 우상측 제어점(1002)의 움직임 벡터라고 할 때, $\overrightarrow{v_0}=\{v_{0x}, v_{0y}\}$, $\overrightarrow{v_1}=\{v_{1x}, v_{1y}\}$로 정의될 수 있다. 이때, AF4의 어파인 움직임 모델을 다음의 수학식 4와 같이 정의될 수 있다.

수학식 4

[0121] $$\begin{cases} v_x = \frac{(v_{1x}-v_{0x})}{w} * x - \frac{(v_{1y}-v_{0y})}{w} * y + v_{0x} \\ v_y = \frac{(v_{1y}-v_{0y})}{w} * x - \frac{(v_{1x}-v_{0x})}{w} * y + v_{0y} \end{cases}$$

[0122] 수학식 4에서 w는 현재 블록의 너비(width), h는 현재 블록의 높이(height)를 나타낸다. 그리고, $\vec{v} = \{v_x, v_y\}$는 각각 {x,y} 위치의 움직임 벡터를 나타낸다.

[0123] 인코더 또는 디코더는 제어점 움직임 벡터(예를 들어, 좌상측 제어점(1001) 및 우상측 제어점(1002)의 움직임 벡터)를 이용하여 각 픽셀 위치의 움직임 벡터를 결정(또는 유도)할 수 있다.

[0124] 본 발명에서, 어파인 움직임 예측을 통해 결정되는 움직임 벡터들의 집합을 어파인 움직임 벡터 필드로 정의할 수 있다. 상기 어파인 움직임 벡터 필드는 상기 수학식 1 내지 4 중 적어도 하나를 이용하여 결정될 수 있다.

[0125] 부호화/복호화 과정에서 어파인 움직임 예측을 통한 움직임 벡터는 픽셀 단위 또는 미리 정의된(또는 미리 설정된) 블록(또는 서브 블록) 단위로 결정될 수 있다. 예를 들어, 픽셀 단위로 결정되는 경우 블록 내 각 픽셀을 기준으로 움직임 벡터가 유도될 수 있고, 서브 블록 단위로 결정되는 경우 현재 블록 내 각 서브 블록 단위를 기준으로 움직임 벡터가 유도될 수 있다. 다른 예로, 서브 블록 단위로 결정되는 경우, 움직임 벡터는 좌상측

픽셀 또는 중앙 픽셀을 기준으로 해당 서브 블록의 움직임 벡터가 유도될 수 있다.

[0126] 이하, 본 발명의 설명에 있어 설명의 편의를 위해, 어파인 움직임 예측을 통한 움직임 벡터가 4x4 블록 단위로 결정되는 경우를 위주로 설명하나, 본 발명은 이에 한정되는 것은 아니며, 본 발명은 픽셀 단위 또는 다른 크기의 블록 단위로 적용될 수 있다.

[0127] 한편, 상기 도 9(b)를 참조하면, 현재 블록의 크기가 16x16 인 경우를 가정한다. 인코더 또는 디코더는 현재 블록의 좌상측 제어점(801) 및 우상측 제어점(802)의 움직임 벡터를 이용하여 이용하여 4x4 서브 블록 단위로 움직임 벡터를 결정할 수 있다. 그리고, 각 서브 블록의 중앙 픽셀 값을 기준으로 해당 서브 블록의 움직임 벡터가 결정될 수 있다.

[0128] 상기 도 9(b)에서, 각 서브 블록의 중앙에 표시된 화살표는 어파인 움직임 모델(Affin motion model)에 의해 획득된 움직임 벡터를 나타낸다.

[0129] 어파인 움직임 예측은 어파인 머지 모드(이하, 'AF 머지 모드'라 함)와 어파인 인터 모드(이하, 'AF 인터 모드'라 함)로 이용될 수 있다. AF 머지 모드는 스킵 모드 또는 머지 모드와 유사하게 움직임 벡터 차이(motion vector difference)를 부호화하지 않고, 2개의 제어점 움직임 벡터를 유도하여 부호화 또는 복호화하는 방법이다. AF 인터 모드는 제어점 움직임 벡터 예측자(motion vector predictor)와 제어점 움직임 벡터를 결정한 후, 그 차이에 해당하는 제어점 움직임 벡터 차이(control point motion vector difference, CPMVD)를 부호화 또는 복호화 방법이다. 이 경우, AF4 모드의 경우에는 2개의 제어점의 움직임 벡터 차이가 전송되고, AF6 모드의 경우에는 3개의 제어점의 움직임 벡터 차이가 전송된다.

[0130] 이때, AF4 모드는 AF6 모드에 비해 적은 개수의 움직임 벡터 차이값을 전송하기 때문에 적은 비트로 제어점 움직임 벡터(Control Point Motion Vector, CPMV) 를 표현할 수 있다는 장점이 있고, AF6 모드는 3개의 CPMVD를 전송하기 때문에 우수한 예측자 생성이 가능하여 차분 코딩(residual coding)을 위한 비트를 줄일 수 있다는 장점이 있다.

[0131] 따라서, 본 발명에서는 AF 인터 모드에서 AF4 모드 및 AF6 모드를 모두(또는 동시에) 고려하는 방법을 제안한다.

[0132] 도 10은 본 발명이 적용되는 실시예로서, 어파인 예측 모드(이하, 'AF 모드'라 함)를 이용하여 현재 블록을 포함하는 비디오 신호를 처리하는 과정을 설명하는 흐름도이다.

[0133] 본 발명은, AF 모드를 이용하여 현재 블록을 포함하는 비디오 신호를 처리하는 방법을 제공한다.

[0134] 먼저, 비디오 신호 처리 장치는 현재 블록의 적어도 2개의 제어점에 인접한 픽셀 또는 블록의 움직임 벡터를 이용하여 움직임 벡터 쌍(motion vector pair)의 후보 리스트를 생성할 수 있다(S1010). 여기서, 상기 제어점은 상기 현재 블록의 코너 픽셀을 의미하고, 상기 움직임 벡터 쌍은 상기 현재 블록의 좌상측 코너 픽셀 및 우상측 코너 픽셀의 움직임 벡터를 나타낸다.

[0135] 일실시예로, 상기 제어점은 상기 현재 블록의 좌상측 코너 픽셀, 우상측 코너 픽셀, 좌하측 코너 픽셀 또는 우하측 코너 픽셀 중 적어도 2개를 포함하고, 상기 후보 리스트는 상기 좌상측 코너 픽셀, 상기 우상측 코너 픽셀 및 상기 좌하측 코너 픽셀에 인접한 픽셀들 또는 블록들에 의해 구성될 수 있다.

[0136] 일실시예로, 상기 후보 리스트는 상기 좌상측 코너 픽셀의 대각 인접 픽셀(A), 상측 인접 픽셀(B) 및 좌측 인접 픽셀(C)의 움직임 벡터들, 상기 우상측 코너 픽셀의 상측 인접 픽셀(D) 및 대각 인접 픽셀(E)의 움직임 벡터들, 및 상기 좌하측 코너 픽셀의 좌측 인접 픽셀(F) 및 대각 인접 픽셀(G)의 움직임 벡터들에 기초하여 생성될 수 있다.

[0137] 일실시예로, 상기 방법은, 상기 후보 리스트의 움직임 벡터 쌍이 2개보다 작은 경우, AMVP 후보 리스트를 상기 후보 리스트에 추가하는 단계를 더 포함할 수 있다.

[0138] 일실시예로, 상기 현재 블록이 Nx4 크기인 경우, 상기 현재 블록의 제어점 움직임 벡터는 상기 현재 블록 내 좌측 서브 블록 및 우측 서브 블록의 중앙 위치를 기준으로 유도된 움직임 벡터로 결정되고, 상기 현재 블록이 4xN 크기인 경우, 상기 현재 블록의 제어점 움직임 벡터는 상기 현재 블록 내 상측 서브 블록 및 하측 서브 블록의 중앙 위치를 기준으로 유도된 움직임 벡터로 결정되는 것을 특징으로 한다.

[0139] 일실시예로, 상기 현재 블록이 Nx4 크기인 경우, 상기 현재 블록 내 좌측 서브 블록의 제어점 움직임 벡터는 제1제어점 움직임 벡터와 제3제어점 움직임 벡터의 평균값에 의해 결정되고, 우측 서브 블록의 제어점 움직임 벡

터는 제2제어점 움직임 벡터와 제4제어점 움직임 벡터의 평균값에 의해 결정되며, 상기 현재 블록이 4xN 크기인 경우, 상기 현재 블록 내 상측 서브 블록의 제어점 움직임 벡터는 제1제어점 움직임 벡터와 제2제어점 움직임 벡터의 평균값에 의해 결정되고, 하측 서브 블록의 제어점 움직임 벡터는 제3제어점 움직임 벡터와 제4제어점 움직임 벡터의 평균값에 의해 결정되는 것을 특징으로 한다.

[0140] 다른 일실시예로, 상기 방법은, 상기 AF 모드가 수행되는지 여부를 나타내는 예측 모드 또는 플래그 정보를 시그널링할 수 있다.

[0141] 이 경우, 상기 비디오 신호 처리 장치는 상기 예측 모드 또는 플래그 정보를 수신하고, 상기 예측 모드 또는 상기 플래그 정보에 따라 상기 AF 모드를 수행하고, 상기 AF 모드에 따라 움직임 벡터를 유도할 수 있다. 여기서, 상기 AF 모드는 상기 현재 블록의 제어점 움직임 벡터를 이용하여 픽셀 또는 서브 블록 단위로 움직임 벡터를 유도하는 모드를 나타내는 것을 특징으로 한다.

[0142] 한편, 상기 비디오 신호 처리 장치는 상기 움직임 벡터 쌍의 발산값(divergence value)에 기초하여 기설정된 개수의 움직임 벡터 쌍의 최종 후보 리스트를 결정할 수 있다(S1020). 여기서, 상기 최종 후보 리스트는 발산값이 작은 순서대로 결정되고, 상기 발산값은 움직임 벡터들의 방향의 유사성을 나타내는 값을 의미한다.

[0143] 상기 비디오 신호 처리 장치는 상기 최종 후보 리스트로부터 율-왜곡 비용에 기초하여 상기 현재 블록의 제어점 움직임 벡터를 결정할 수 있다(S1030).

[0144] 상기 비디오 신호 처리 장치는 상기 제어점 움직임 벡터에 기초하여 상기 현재 블록의 움직임 벡터 예측자를 생성할수 있다(S1040).

[0145] 도 11은 본 발명이 적용되는 실시예(1-1)로서, AF4 모드 또는 AF6 모드 중 적어도 하나에 기초하여 최적의 코딩 모드를 적응적으로 결정하는 흐름도를 나타낸다.

[0146] 비디오 신호 처리 장치는 스킵 모드, 머지 모드, 또는 인터 모드 중 적어도 하나에 기초하여 예측을 수행할 수 있다(S1110). 여기서, 머지 모드를 일반적인 머지 모드 뿐만 아니라 앞서 설명한 AF 머지 모드를 포함할 수 있고, 인터 모드는 일반적인 인터 모드 뿐만 아니라 앞서 설명한 AF 인터 모드를 포함할 수 있다.

[0147] 상기 비디오 신호 처리 장치는 AF4 모드 또는 AF6 모드 중 적어도 하나에 기초하여 움직임 벡터 예측을 수행할 수 있다(S1120). 여기서, 상기 S1110 단계와 상기 S1120 단계는 그 순서에 제한되지 않는다.

[0148] 상기 비디오 신호 처리 장치는 상기 S1120 단계의 결과들을 비교하여, 상기 모드들 중 최적의 코딩 모드를 결정할 수 있다(S1130). 이때, 상기 S1120 단계의 결과들은 율-왜곡 비용(Rate-Distortion Cost)에 기초하여 비교될 수 있다.

[0149] 이후, 상기 비디오 신호 처리 장치는 최적의 코딩 모드에 기초하여 현재 블록의 움직임 벡터 예측자를 생성할 수 있고, 상기 현재 블록의 움직임 벡터에서 상기 움직임 벡터 예측자를 감산하여 움직임 벡터 차이값을 획득할 수 있다.

[0150] 이후, 상기 도 1 및 도 2에서 설명하였던 인코딩/디코딩 과정이 동일하게 적용될 수 있다.

[0151] 도 12는 본 발명이 적용되는 실시예(1-2)로서, AF4 모드 또는 AF6 모드에 기초하여 적응적으로 디코딩을 수행하는 흐름도를 나타낸다.

[0152] 디코더는 비트스트림을 수신할 수 있다(S1210). 상기 비트스트림은 비디오 신호 내 현재 블록의 코딩 모드에 대한 정보를 포함할 수 있다.

[0153] 상기 디코더는 상기 현재 블록의 코딩 모드가 AF 모드인지 여부를 확인할 수 있다(S1220). 여기서, 상기 AF 모드는 어파인 움직임 모델(affine motion model)을 이용하는 어파인 움직임 예측 모드를 의미하며, 예를 들어, 어파인 머지 모드(affine merge mode) 또는 어파인 인터 모드(affine inter mode) 중 적어도 하나를 포함할 수 있고, 상기 어파인 인터 모드(affine inter mode)는 AF4 모드 또는 AF6 모드 중 적어도 하나를 포함할 수 있다.

[0154] 이때, 상기 S1220 단계는 AF 모드가 수행되는지 여부를 나타내는 어파인 플래그에 의해 확인될 수 있다. 예를 들어, 상기 어파인 플래그는 affine_flag 로 표현될 수 있다. 상기 affine_flag = 1 이면, 현재 블록에 대해 AF 모드가 수행되는 것을 나타내고, 상기 affine_flag = 0 이면, 현재 블록에 대해 AF 모드가 수행되지 않는 것을 나타낸다.

[0155] 현재 블록에 대해 AF 모드가 수행되지 않는 경우, 상기 디코더는 상기 AF 모드가 아닌 코딩 모드에 따라 디코딩

(즉, 움직임 벡터 예측)을 수행할 수 있다(S1230). 예를 들어, 스킵 모드, 머지 모드 또는 인터 모드가 이용될 수 있다.

[0156] 현재 블록에 대해 AF 모드가 수행되는 경우, 상기 디코더는 상기 현재 블록에 대해 AF4 모드가 적용되는지 여부를 확인할 수 있다(S1240).

[0157] 이때, 상기 S1240 단계는 AF4 모드가 수행되는지 여부(또는 4개의 파라미터에 의해 어파인 움직임 예측이 수행되는지 여부)를 나타내는 어파인 파라미터 플래그에 의해 확인될 수 있다. 예를 들어, 상기 어파인 파라미터 플래그는 affine_param_flag 로 표현될 수 있다. 상기 affine_param_flag = 0 이면, AF4 모드에 따라 움직임 벡터 예측이 수행되고(S1250), 상기 affine_param_flag = 1 이면, AF6 모드에 따라 움직임 벡터 예측이 수행되는 것을 의미할 수 있으나(S1260), 본 발명은 이에 한정되지 않는다.

[0158] 예를 들어, 상기 어파인 파라미터 플래그는 AF4_flag 및 AF6_flag 중 적어도 하나를 포함할 수 있다.

[0159] AF4_flag 는 현재 블록에 대해 AF4 모드가 수행되는지 여부를 나타낸다. AF4_flag = 1이면 현재 블록에 대해 AF4 모드가 수행되고, AF4_flag = 0이면 현재 블록에 대해 AF4 모드가 수행되지 않는다. 여기서, AF4 모드가 수행된다는 것은 4개의 파라미터로 표현되는 어파인 움직임 모델을 이용하여 움직임 벡터 예측을 수행하는 것을 의미한다.

[0160] AF6_flag 는 현재 블록에 대해 AF6 모드가 수행되는지 여부를 나타낸다. AF6_flag = 1이면 현재 블록에 대해 AF6 모드가 수행되고, AF4_flag = 0이면 현재 블록에 대해 AF6 모드가 수행되지 않는다. 여기서, AF6 모드가 수행된다는 것은 4개의 파라미터로 표현되는 어파인 움직임 모델을 이용하여 움직임 벡터 예측을 수행하는 것을 의미한다.

[0161] 상기 어파인 플래그 및 상기 어파인 파라미터 플래그는 슬라이스, 최대 코딩 유닛, 코딩 유닛 또는 예측 유닛 중 적어도 하나의 레벨에서 정의될 수 있다.

[0162] 예를 들어, AF_flag, AF4_flag 및 AF6_flag 중 적어도 하나는 슬라이스 레벨에서 정의되고, 다시 블록 레벨 또는 예측 유닛 레벨에서 정의될 수 있다.

[0163] 도 13은 본 발명이 적용되는 실시예(1-3)로서, AF4 모드 또는 AF6 모드에 기초하여 디코딩을 수행하는 신택스 구조를 나타낸다.

[0164] 디코더는 merge_flag를 획득하여, 현재 블록에 대해 머지 모드가 적용되는지 여부를 확인할 수 있다(S1310).

[0165] 상기 현재 블록에 대해 머지 모드가 적용되지 않는 경우, 상기 디코더는 affine_flag 를 획득할 수 있다(S1320). 여기서, affine_flag 는 AF 모드가 수행되는지 여부를 나타낸다.

[0166] 상기 affine_flag = 1 이면, 즉 현재 블록에 대해 AF 모드가 수행되면, 상기 디코더는 affine_param_flag 를 획득할 수 있다(S1330). 여기서, affine_param_flag 는 AF4 모드가 수행되는지 여부(또는 4개의 파라미터에 의해 어파인 움직임 예측이 수행되는지 여부)를 나타낸다.

[0167] 상기 affine_param_flag = 0 이면, 즉 AF4 모드에 따라 움직임 벡터 예측이 수행되면, 상기 디코더는 2개의 움직임 벡터 차이값인, mvd_CP0 및 mvd_CP1을 획득할 수 있다(S1340). 여기서, mvd_CP0 는 제어점 0에 대한 움직임 벡터 차이값을 나타내고, mvd_CP1 은 제어점 1에 대한 움직임 벡터 차이값을 나타낸다.

[0168] 그리고, 상기 affine_param_flag = 1 이면, 즉 AF6 모드에 따라 움직임 벡터 예측이 수행되면, 상기 디코더는 3개의 움직임 벡터 차이값인, mvd_CP0, mvd_CP1 및 mvd_CP2를 획득할 수 있다(S1350).

[0169] 도 14는 본 발명이 적용되는 실시예(2-1)로서, 조건 A(condition A)에 기초하여 AF4 모드 또는 AF6 모드를 포함하는 움직임 벡터 예측 모드들 중 최적의 코딩 모드를 적응적으로 결정하는 흐름도를 나타낸다.

[0170] 인코더는 스킵 모드, 머지 모드, 또는 인터 모드 중 적어도 하나에 기초하여 예측을 수행할 수 있다(S1410).

[0171] 상기 인코더는, 움직임 벡터 예측을 위한 최적의 코딩 모드를 결정하기 위해 현재 블록에 대해 조건 A가 만족되는지 여부를 확인할 수 있다(S1420).

[0172] 여기서, 상기 조건 A는 블록 크기에 대한 조건을 의미할 수 있다. 예를 들어, 다음 표 1의 실시예들이 적용될 수 있다.

표 1

[0173]

	CONDITION A	TH1 값
Example 1	pixNum (= width * height) > TH1	TH1 = 64, 128, 256, 512, 1024, …
Example 2	width > TH1 && height > TH1	TH1 = 4, 8, 16, 32, …
Example 3	width > TH1 || height > TH1	TH1 = 4, 8, 16, 32, …

[0174] 상기 표 1의 Example 1에서, 상기 조건 A는 상기 현재 블록의 픽셀 개수(pixNum)가 임계값(TH1)보다 큰지 여부를 나타낸다. 이때, 상기 임계값은 64, 128, 256, 512, 1024, … 등의 값을 가질 수 있으며, 예를 들어, TH1 = 64 는 블록 크기가 4x16, 8x8, 또는 16x4 인 경우를 의미하고, TH1 = 128 은 블록 크기가 32x4, 16x8, 8x16 또는 4x32 인 경우를 의미할 수 있다.

[0175] Example 2의 경우, 상기 현재 블록의 너비(width) 및 높이(height)가 모두 임계값(TH1)보다 큰지 여부를 나타낸다.

[0176] Example 3의 경우, 상기 현재 블록의 너비(width)가 임계값(TH1)보다 크거나 또는 상기 현재 블록의 높이(height)가 임계값(TH1)보다 큰지 여부를 나타낸다.

[0177] 상기 조건 A가 만족되는 경우, 상기 인코더는 AF4 모드 또는 AF6 모드 중 적어도 하나에 기초하여 움직임 벡터 예측을 수행할 수 있다(S1430).

[0178] 상기 인코더는 상기 S1410 및 상기 S1430 단계들의 결과들을 비교하여, AF4 모드 또는 AF6 모드를 포함하는 움직임 벡터 예측 모드들 중 최적의 코딩 모드를 결정할 수 있다(S1440).

[0179] 한편, 상기 조건 A가 만족되지 않는 경우, 상기 인코더는 AF 모드가 아닌 모드들 중에서 최적의 코딩 모드를 결정할 수 있다(S1440).

[0180] 이후, 상기 인코더는 최적의 코딩 모드에 기초하여 현재 블록의 움직임 벡터 예측자를 생성할 수 있고, 상기 현재 블록의 움직임 벡터에서 상기 움직임 벡터 예측자를 감산하여 움직임 벡터 차이값을 획득할 수 있다.

[0181] 이후, 상기 도 1 및 도 2에서 설명하였던 인코딩/디코딩 과정이 동일하게 적용될 수 있다.

[0182] 도 15는 본 발명이 적용되는 실시예(2-2)로서, 조건 A(condition A)에 기초하여 AF4 모드 또는 AF6 모드에 따라 적응적으로 디코딩을 수행하는 흐름도를 나타낸다.

[0183] 디코더는 비트스트림을 수신할 수 있다(S1510). 상기 비트스트림은 비디오 신호 내 현재 블록의 코딩 모드에 대한 정보를 포함할 수 있다.

[0184] 상기 디코더는, 움직임 벡터 예측을 위한 최적의 코딩 모드를 결정하기 위해 현재 블록에 대해 조건 A가 만족되는지 여부를 확인할 수 있다(S1520). 여기서, 상기 조건 A는 블록 크기에 대한 조건을 의미할 수 있다. 예를 들어, 상기 표 1의 실시예들이 적용될 수 있다.

[0185] 상기 조건 A가 만족되는 경우, 상기 디코더는 상기 현재 블록의 코딩 모드가 AF 모드인지 여부를 확인할 수 있다(S1530). 여기서, 상기 AF 모드는 어파인 움직임 모델(affine motion model)을 이용하는 어파인 움직임 예측 모드를 의미하며, 본 명세서에서 설명한 실시예들이 적용될 수 있다.

[0186] 이때, 상기 S1530 단계는 AF 모드가 수행되는지 여부를 나타내는 어파인 플래그에 의해 확인될 수 있다. 예를 들어, 상기 어파인 플래그는 affine_flag 로 표현될 수 있다. 상기 affine_flag = 1 이면, 현재 블록에 대해 AF 모드가 수행되는 것을 나타내고, 상기 affine_flag = 0 이면, 현재 블록에 대해 AF 모드가 수행되지 않는 것을 나타낸다.

[0187] 상기 조건 A가 만족되지 않는 경우이거나, 상기 현재 블록에 대해 AF 모드가 수행되지 않는 경우, 상기 디코더는 상기 AF 모드가 아닌 코딩 모드에 따라 디코딩(즉, 움직임 벡터 예측)을 수행할 수 있다(S1540). 예를 들어, 스킵 모드, 머지 모드 또는 인터 모드가 이용될 수 있다.

[0188] 상기 현재 블록에 대해 AF 모드가 수행되는 경우, 상기 디코더는 상기 현재 블록에 대해 AF4 모드가 적용되는지 여부를 확인할 수 있다(S1550).

[0189] 이때, 상기 S1550 단계는 AF4 모드가 수행되는지 여부(또는 4개의 파라미터에 의해 어파인 움직임 예측이 수행되는지 여부)를 나타내는 어파인 파라미터 플래그에 의해 확인될 수 있다. 예를 들어, 상기 어파인 파라미터 플래그는 affine_param_flag 로 표현될 수 있다. 상기 affine_param_flag = 0 이면, AF4 모드에 따라 움직임 벡터 예측이 수행되고(S1560), 상기 affine_param_flag = 1 이면, AF6 모드에 따라 움직임 벡터 예측이 수행되는 것을 의미할 수 있으나(S1570), 본 발명은 이에 한정되지 않는다.

[0190] 도 16은 본 발명이 적용되는 실시예(2-3)로서, 조건 A(condition A)에 기초하여 AF4 모드 또는 AF6 모드에 따라 디코딩을 수행하는 신택스 구조를 나타낸다.

[0191] 디코더는 merge_flag를 획득하여, 현재 블록에 대해 머지 모드가 적용되는지 여부를 확인할 수 있다(S1610).

[0192] 상기 현재 블록에 대해 머지 모드가 적용되지 않는 경우, 상기 디코더는 조건 A가 만족되는지 여부를 확인할 수 있다(S1620). 여기서, 상기 조건 A는 블록 크기에 대한 조건을 의미할 수 있다. 예를 들어, 상기 표 1의 실시예들이 적용될 수 있다.

[0193] 상기 조건 A가 만족되는 경우, 상기 디코더는 affine_flag 를 획득할 수 있다(S1620). 여기서, affine_flag 는 AF 모드가 수행되는지 여부를 나타낸다.

[0194] 상기 affine_flag = 1 이면, 즉 현재 블록에 대해 AF 모드가 수행되면, 상기 디코더는 affine_param_flag 를 획득할 수 있다(S1630). 여기서, affine_param_flag 는 AF4 모드가 수행되는지 여부(또는 4개의 파라미터에 의해 어파인 움직임 예측이 수행되는지 여부)를 나타낸다.

[0195] 상기 affine_param_flag = 0 이면, 즉 AF4 모드에 따라 움직임 벡터 예측이 수행되면, 상기 디코더는 2개의 움직임 벡터 차이값인, mvd_CP0 및 mvd_CP1을 획득할 수 있다(S1640). 여기서, mvd_CP0 는 제어점 0에 대한 움직임 벡터 차이값을 나타내고, mvd_CP1 은 제어점 1에 대한 움직임 벡터 차이값을 나타낸다.

[0196] 그리고, 상기 affine_param_flag = 1 이면, 즉 AF6 모드에 따라 움직임 벡터 예측이 수행되면, 상기 디코더는 3개의 움직임 벡터 차이값인, mvd_CP0, mvd_CP1 및 mvd_CP2를 획득할 수 있다(S1650).

[0197] 도 17은 본 발명이 적용되는 실시예(3-1)로서, 조건 B(condition B) 또는 조건 C(condition C) 중 적어도 하나에 기초하여, AF4 모드 또는 AF6 모드를 포함하는 움직임 벡터 예측 모드들 중 최적의 코딩 모드를 적응적으로 결정하는 흐름도를 나타낸다.

[0198] 본 발명은 현재 블록의 크기에 기초하여 AF4 모드와 AF6 모드를 적응적으로 선택하는 방법을 제공한다.

[0199] 예를 들어, AF6 모드는 AF4 모드보다 추가적으로 1개의 움직임 벡터 차이값을 전송하여 하므로, 상대적으로 큰 블록에서는 효과적이다. 따라서, 현재 블록의 크기가 기설정된 크기보다 작은(또는 작거나 같은) 경우 AF4 모드만을 고려하여 인코딩을 수행하고 상기 현재 블록이 기설정된 크기보다 크거나 같은(또는 큰) 경우 AF6 모드만을 고려하여 인코딩을 수행할 수 있다.

[0200] 한편, AF4 모드 또는 AF6 모드 중 명확하게 하나만 유리하다고 판단되지 않는 영역의 경우에는, AF4 모드 및 AF6 모드 모두를 고려하고 이 중 최적의 모드만을 시그널링 할 수 있다.

[0201] 상기 도 17을 살펴보면, 인코더는 스킵 모드, 머지 모드, 또는 인터 모드 중 적어도 하나에 기초하여 예측을 수행할 수 있다(S1710).

[0202] 상기 인코더는, 현재 블록에 대해 조건 B가 만족되는지 여부를 확인할 수 있다(S1720). 여기서, 상기 조건 B는 블록 크기에 대한 조건을 의미할 수 있다. 예를 들어, 다음 표 2의 실시예들이 적용될 수 있다.

표 2

[0203]

	CONDITION B	TH2 값
Example 1	pixNum (= width * height) < TH2	TH2 = 64, 128, 256, 512, 1024, …
Example 2	width < TH2 && height < TH2	TH2 = 4, 8, 16, 32, …
Example 3	width < TH2 || height < TH2	TH2 = 4, 8, 16, 32, …

[0204] 상기 표 2의 Example 1에서, 상기 조건 B는 상기 현재 블록의 픽셀 개수(pixNum)가 임계값(TH2)보다 작은지 여부를 나타낸다. 이때, 상기 임계값은 64, 128, 256, 512, 1024, … 등의 값을 가질 수 있으며, 예를 들어, TH2

= 64 는 블록 크기가 4x16, 8x8, 또는 16x4 인 경우를 의미하고, TH2 = 128 은 블록 크기가 32x4, 16x8, 8x16 또는 4x32 인 경우를 의미할 수 있다.

[0205] Example 2의 경우, 상기 조건 B는 상기 현재 블록의 너비(width) 및 높이(height)가 모두 임계값(TH2)보다 작은지 여부를 나타낸다.

[0206] Example 3의 경우, 상기 조건 B는 기 현재 블록의 너비(width)가 임계값(TH2)보다 작거나 또는 상기 현재 블록의 높이(height)가 임계값(TH2)보다 작은지 여부를 나타낸다.

[0207] 상기 조건 B가 만족되는 경우, 상기 인코더는 AF4 모드에 기초하여 움직임 벡터 예측을 수행할 수 있다(S1730).

[0208] 상기 조건 B가 만족되지 않는 경우, 상기 인코더는 상기 현재 블록에 대해 조건 C가 만족되는지 여부를 확인할 수 있다(S1740). 여기서, 상기 조건 C는 블록 크기에 대한 조건을 의미할 수 있다. 예를 들어, 다음 표 3의 실시예들이 적용될 수 있다.

표 3

[0209]

	CONDITION C	TH3 값
Example 1	pixNum (= width * height) ≥ TH3	TH3 = 64, 128, 256, 512, 1024, …
Example 2	width ≥ TH3 && height ≥ TH3	TH3 = 4, 8, 16, 32, …
Example 3	width ≥ TH3 || height ≥ TH3	TH3 = 4, 8, 16, 32, …

[0210] 상기 표 3의 Example 1에서, 상기 조건 A는 상기 현재 블록의 픽셀 개수(pixNum)가 임계값(TH3)보다 크거나 같은지 여부를 나타낸다. 이때, 상기 임계값은 64, 128, 256, 512, 1024, … 등의 값을 가질 수 있으며, 예를 들어, TH3 = 64 는 블록 크기가 4x16, 8x8, 또는 16x4 인 경우를 의미하고, TH3 = 128 은 블록 크기가 32x4, 16x8, 8x16 또는 4x32 인 경우를 의미할 수 있다.

[0211] Example 2의 경우, 상기 현재 블록의 너비(width) 및 높이(height)가 모두 임계값(TH3)보다 크거나 같은지 여부를 나타낸다.

[0212] Example 3의 경우, 상기 현재 블록의 너비(width)가 임계값(TH1)보다 크거나 같은지, 또는 상기 현재 블록의 높이(height)가 임계값(TH1)보다 크거나 같은지 여부를 나타낸다.

[0213] 상기 조건 C가 만족되는 경우, 상기 인코더는 AF6 모드에 기초하여 움직임 벡터 예측을 수행할 수 있다(S1760).

[0214] 상기 조건 C가 만족되지 않는 경우, 상기 인코더는 AF4 모드 및 AF6 모드에 기초하여 움직임 벡터 예측을 수행할 수 있다(S1750).

[0215] 한편, 상기 조건 B(condition B) 및 상기 조건 C(condition C)에서, 임계값(TH2)와 임계값(TH3)은 다음 수학식 5를 만족하도록 결정될 수 있다.

수학식 5

[0216] $$TH2 \leq TH3$$

[0217] 상기 인코더는 상기 S1710, S1730, S1750, S1760 단계들의 결과들을 비교하여, AF4 모드 또는 AF6 모드를 포함하는 움직임 벡터 예측 모드들 중 최적의 코딩 모드를 결정할 수 있다(S1770).

[0218] 이후, 상기 인코더는 최적의 코딩 모드에 기초하여 현재 블록의 움직임 벡터 예측자를 생성할 수 있고, 상기 현재 블록의 움직임 벡터에서 상기 움직임 벡터 예측자를 감산하여 움직임 벡터 차이값을 획득할 수 있다.

[0219] 이후, 상기 도 1 및 도 2에서 설명하였던 인코딩/디코딩 과정이 동일하게 적용될 수 있다.

[0220] 도 18은 본 발명이 적용되는 실시예(3-2)로서, 조건 B(condition B) 또는 조건 C(condition C) 중 적어도 하나에 기초하여, AF4 모드 또는 AF6 모드에 따라 적응적으로 디코딩을 수행하는 흐름도를 나타낸다.

[0221] 디코더는 현재 블록의 코딩 모드가 AF 모드인지 여부를 확인할 수 있다(S1810). 여기서, 상기 AF 모드는 어파인

움직임 모델(affine motion model)을 이용하는 어파인 움직임 예측 모드를 의미하며, 본 명세서에서 설명한 실시예들이 적용될 수 있으며, 중복되는 설명은 생략하도록 한다.

[0222] 상기 현재 블록에 대해 AF 모드가 수행되는 경우, 상기 디코더는 상기 현재 블록에 대해 조건 B가 만족되는지 여부를 확인할 수 있다(S1820). 여기서, 상기 조건 B는 블록 크기에 대한 조건을 의미할 수 있다. 예를 들어, 상기 표 2의 실시예들이 적용될 수 있으며, 중복되는 설명은 생략하도록 한다.

[0223] 상기 조건 B가 만족되는 경우, 상기 디코더는 AF4 모드에 기초하여 움직임 벡터 예측을 수행할 수 있다(S1830).

[0224] 상기 조건 B가 만족되지 않는 경우, 상기 디코더는 상기 현재 블록에 대해 조건 C가 만족되는지 여부를 확인할 수 있다(S1840). 여기서, 상기 조건 C는 블록 크기에 대한 조건을 의미할 수 있다. 예를 들어, 상기 표 3의 실시예들이 적용될 수 있으며, 중복되는 설명은 생략하도록 한다.

[0225] 한편, 상기 조건 B(condition B) 및 상기 조건 C(condition C)에서, 임계값(TH2)와 임계값(TH3)은 상기 수학식 5를 만족하도록 결정될 수 있다.

[0226] 상기 조건 C가 만족되는 경우, 상기 디코더는 AF6 모드에 기초하여 움직임 벡터 예측을 수행할 수 있다(S1860).

[0227] 상기 조건 C가 만족되지 않는 경우, 상기 디코더는 상기 현재 블록에 대해 AF4 모드가 적용되는지 여부를 확인할 수 있다(S1850).

[0228] 이때, 상기 S1850 단계는 AF4 모드가 수행되는지 여부(또는 4개의 파라미터에 의해 어파인 움직임 예측이 수행되는지 여부)를 나타내는 어파인 파라미터 플래그에 의해 확인될 수 있다.

[0229] 예를 들어, 상기 어파인 파라미터 플래그는 affine_param_flag 로 표현될 수 있다. 상기 affine_param_flag = 0 이면, AF4 모드에 따라 움직임 벡터 예측이 수행되고(S1830), 상기 affine_param_flag = 1 이면, AF6 모드에 따라 움직임 벡터 예측이 수행되는 것을 의미할 수 있으나(S1860), 본 발명은 이에 한정되지 않는다.

[0230] 한편, 상기 현재 블록에 대해 AF 모드가 수행되지 않는 경우, 상기 디코더는 상기 AF 모드가 아닌 코딩 모드에 따라 디코딩(즉, 움직임 벡터 예측)을 수행할 수 있다(S1870). 예를 들어, 스킵 모드, 머지 모드 또는 인터 모드가 이용될 수 있다.

[0231] 도 19는 본 발명이 적용되는 실시예(3-3)로서, 조건 B(condition B) 또는 조건 C(condition C) 중 적어도 하나에 기초하여, AF4 모드 또는 AF6 모드에 따라 디코딩을 수행하는 신택스 구조를 나타낸다.

[0232] 디코더는 merge_flag를 획득하여, 현재 블록에 대해 머지 모드가 적용되는지 여부를 확인할 수 있다(S1910).

[0233] 상기 현재 블록에 대해 머지 모드가 적용되지 않는 경우, 상기 디코더는 affine_flag 를 획득할 수 있다(S1920). 여기서, affine_flag 는 AF 모드가 수행되는지 여부를 나타낸다.

[0234] 상기 affine_flag = 1 이면, 즉 현재 블록에 대해 AF 모드가 수행되면, 상기 디코더는 조건 B가 만족되는지 여부를 확인할 수 있다(S1620). 여기서, 상기 조건 B는 블록 크기에 대한 조건을 의미할 수 있다. 예를 들어, 상기 표 2의 실시예들이 적용될 수 있다.

[0235] 상기 조건 B가 만족되는 경우, 상기 디코더는 affine_param_flag 를 0으로 설정할 수 있다(S1930). 여기서, affine_param_flag 는 AF4 모드가 수행되는지 여부(또는 4개의 파라미터에 의해 어파인 움직임 예측이 수행되는지 여부)를 나타낸다. affine_param_flag = 0 은, AF4 모드에 따라 움직임 벡터 예측을 수행하는 것을 의미한다.

[0236] 상기 조건 B가 만족되지 않고 조건 C가 만족되는 경우, 상기 디코더는 affine_param_flag 를 1로 설정할 수 있다(S1940). 여기서, affine_param_flag = 1 은, AF6 모드에 따라 움직임 벡터 예측을 수행하는 것을 의미한다.

[0237] 반면, 상기 조건 B가 만족되지 않고 상기 조건 C도 만족되지 않는 경우, 상기 디코더는 affine_param_flag 를 획득할 수 있다(S1950).

[0238] 상기 affine_param_flag = 0 이면, 상기 디코더는 2개의 움직임 벡터 차이값인, mvd_CP0 및 mvd_CP1을 획득할 수 있다(S1960).

[0239] 상기 affine_param_flag = 1 이면, 상기 디코더는 3개의 움직임 벡터 차이값인, mvd_CP0, mvd_CP1 및 mvd_CP2를 획득할 수 있다(S1970).

[0240] 도 20은 본 발명이 적용되는 실시예(4-1)로서, 이웃 블록의 코딩 모드에 기초하여 AF4 모드 또는 AF6 모드를 포

함하는 움직임 벡터 예측 모드들 중 최적의 코딩 모드를 적응적으로 결정하는 흐름도를 나타낸다.

[0241] 인코더는 스킵 모드, 머지 모드, 또는 인터 모드 중 적어도 하나에 기초하여 예측을 수행할 수 있다(S2010).

[0242] 상기 인코더는, 이웃 블록이 AF 모드로 코딩되었는지 여부를 확인할 수 있다(S2020). 여기서, 이웃 블록이 AF 모드로 코딩되었는지 여부는 isNeighborAffine() 로 표현될 수 있다. 예를 들어, isNeighborAffine() = 0 이면, 이웃 블록은 AF 모드로 코딩되지 않은 경우를 의미하고, isNeighborAffine() = 1 이면, 이웃 블록은 AF 모드로 코딩된 경우를 의미할 수 있다.

[0243] 상기 이웃 블록이 AF 모드로 코딩되지 않은 경우, 상기 인코더는 AF4 모드에 기초하여 움직임 벡터 예측을 수행할 수 있다(S2030).

[0244] 상기 이웃 블록이 AF 모드로 코딩된 경우, 상기 인코더는 AF4 모드에 기초하여 움직임 벡터 예측을 수행하고, AF6 모드에 기초해서도 움직임 벡터 예측을 수행할 수 있다(S2040).

[0245] 상기 인코더는 상기 S2030 및 상기 S2040 단계들의 결과들을 비교하여, AF4 모드 또는 AF6 모드를 포함하는 움직임 벡터 예측 모드들 중 최적의 코딩 모드를 결정할 수 있다(S2050).

[0246] 이후, 상기 인코더는 최적의 코딩 모드에 기초하여 현재 블록의 움직임 벡터 예측자를 생성할 수 있고, 상기 현재 블록의 움직임 벡터에서 상기 움직임 벡터 예측자를 감산하여 움직임 벡터 차이값을 획득할 수 있다.

[0247] 이후, 상기 도 1 및 도 2에서 설명하였던 인코딩/디코딩 과정이 동일하게 적용될 수 있다.

[0248] 도 21은 본 발명이 적용되는 실시예(4-2)로서, 이웃 블록의 코딩 모드에 기초하여 AF4 모드 또는 AF6 모드에 따라 적응적으로 디코딩을 수행하는 흐름도를 나타낸다.

[0249] 디코더는 비트스트림을 수신할 수 있다(S2110). 상기 비트스트림은 비디오 신호 내 현재 블록의 코딩 모드에 대한 정보를 포함할 수 있다.

[0250] 상기 디코더는 상기 현재 블록의 코딩 모드가 AF 모드인지 여부를 확인할 수 있다(S2120).

[0251] 상기 현재 블록에 대해 AF 모드가 수행되지 않는 경우, 상기 디코더는 상기 AF 모드가 아닌 코딩 모드에 따라 디코딩(즉, 움직임 벡터 예측)을 수행할 수 있다(S2170). 예를 들어, 스킵 모드, 머지 모드 또는 인터 모드가 이용될 수 있다.

[0252] 현재 블록에 대해 AF 모드가 수행되는 경우, 상기 디코더는 이웃 블록이 AF 모드로 코딩되었는지 여부를 확인할 수 있다(S2130). 여기서, 이웃 블록이 AF 모드로 코딩되었는지 여부는 isNeighborAffine() 로 표현될 수 있다. 예를 들어, isNeighborAffine() = 0 이면, 이웃 블록은 AF 모드로 코딩되지 않은 경우를 의미하고, isNeighborAffine() = 1 이면, 이웃 블록은 AF 모드로 코딩된 경우를 의미할 수 있다.

[0253] 상기 이웃 블록이 AF 모드로 코딩된 경우, 상기 디코더는 AF4 모드에 기초하여 움직임 벡터 예측을 수행할 수 있다(S2140).

[0254] 상기 이웃 블록이 AF 모드로 코딩되지 않은 경우, 상기 디코더는 상기 현재 블록에 대해 AF4 모드가 적용되는지 여부를 확인할 수 있다(S2150).

[0255] 이때, 상기 S2150 단계는 AF4 모드가 수행되는지 여부(또는 4개의 파라미터에 의해 어파인 움직임 예측이 수행되는지 여부)를 나타내는 어파인 파라미터 플래그에 의해 확인될 수 있다. 예를 들어, 상기 어파인 파라미터 플래그는 affine_param_flag 로 표현될 수 있다. 상기 affine_param_flag = 0 이면, AF4 모드에 따라 움직임 벡터 예측이 수행되고(S2140), 상기 affine_param_flag = 1 이면, AF6 모드에 따라 움직임 벡터 예측이 수행된다(S2160).

[0256] 도 22는 본 발명이 적용되는 실시예(4-3)로서, 이웃 블록의 코딩 모드에 기초하여 AF4 모드 또는 AF6 모드에 따라 디코딩을 수행하는 신택스 구조를 나타낸다.

[0257] 디코더는 merge_flag를 획득하여, 현재 블록에 대해 머지 모드가 적용되는지 여부를 확인할 수 있다(S2210).

[0258] 상기 현재 블록에 대해 머지 모드가 적용되지 않는 경우, 상기 디코더는 affine_flag 를 획득할 수 있다(S2220). 여기서, affine_flag 는 AF 모드가 수행되는지 여부를 나타낸다.

[0259] 상기 affine_flag = 1 이면, 즉 현재 블록에 대해 AF 모드가 수행되면, 상기 디코더는 이웃 블록이 AF 모드로 코딩되었는지 여부를 확인할 수 있다(S2230).

[0260] 이웃 블록이 AF 모드로 코딩된 경우, 상기 디코더는 affine_param_flag 를 획득할 수 있다(S2230). 여기서, affine_param_flag 는 AF4 모드가 수행되는지 여부(또는 4개의 파라미터에 의해 어파인 움직임 예측이 수행되는지 여부)를 나타낸다.

[0261] 이웃 블록이 AF 모드로 코딩되지 않은 경우, 상기 디코더는 affine_param_flag 를 0으로 설정할 수 있다(S2240).

[0262] 상기 affine_param_flag = 0 이면, 즉 AF4 모드에 따라 움직임 벡터 예측이 수행되면, 상기 디코더는 2개의 움직임 벡터 차이값인, mvd_CP0 및 mvd_CP1을 획득할 수 있다(S2250).

[0263] 그리고, 상기 affine_param_flag = 1 이면, 즉 AF6 모드에 따라 움직임 벡터 예측이 수행되면, 상기 디코더는 3개의 움직임 벡터 차이값인, mvd_CP0, mvd_CP1 및 mvd_CP2를 획득할 수 있다(S2260).

[0264] 도 23은 본 발명이 적용되는 실시예(5-1)로서, 조건 A(condition A), 조건 B(condition B) 또는 조건 C(condition C) 중 적어도 하나에 기초하여, AF4 모드 또는 AF6 모드를 포함하는 움직임 벡터 예측 모드들 중 최적의 코딩 모드를 적응적으로 결정하는 흐름도를 나타낸다.

[0265] 본 발명은 실시예 2와 실시예 3을 조합하는 실시예를 나타낸다. 도 23에서는 조건 A, B, C를 모두 고려하는 경우의 한 예를 설명하고 있으며, 조건의 순서는 다르게 적용 가능하다.

[0266] 상기 도 23을 살펴보면, 인코더는 스킵 모드, 머지 모드, 또는 인터 모드 중 적어도 하나에 기초하여 예측을 수행할 수 있다(S1710).

[0267] 상기 인코더는, 현재 블록에 대해 조건 A가 만족되는지 여부를 확인할 수 있다(S2320). 여기서, 상기 조건 A는 블록 크기에 대한 조건을 의미할 수 있으며, 상기 표 1의 실시예들이 적용될 수 있다.

[0268] 상기 조건 A가 만족되는 경우, 상기 인코더는 AF 모드를 제외한 모드들 중에서 최적의 코딩 모드를 결정할 수 있다(S2380).

[0269] 한편, 상기 조건 A가 만족되지 않는 경우, 상기 인코더는 상기 현재 블록에 대해 조건 B가 만족되는지 여부를 확인할 수 있다(S2330). 여기서, 상기 조건 B는 블록 크기에 대한 조건을 의미할 수 있으며, 상기 표 2의 실시예들이 적용될 수 있다.

[0270] 상기 조건 B가 만족되는 경우, 상기 인코더는 AF4 모드에 기초하여 움직임 벡터 예측을 수행할 수 있다(S2340).

[0271] 상기 조건 B가 만족되지 않는 경우, 상기 인코더는 상기 현재 블록에 대해 조건 C가 만족되는지 여부를 확인할 수 있다(S2350). 여기서, 상기 조건 C는 블록 크기에 대한 조건을 의미할 수 있으며, 상기 표 3의 실시예들이 적용될 수 있다.

[0272] 상기 조건 C가 만족되는 경우, 상기 인코더는 AF6 모드에 기초하여 움직임 벡터 예측을 수행할 수 있다(S2370).

[0273] 상기 조건 C가 만족되지 않는 경우, 상기 인코더는 AF4 모드에 기초하여 움직임 벡터 예측을 수행하고, 또한 AF6 모드에 기초하여 움직임 벡터 예측을 수행할 수 있다(S2360).

[0274] 한편, 상기 조건 B(condition B) 및 상기 조건 C(condition C)에서, 임계값(TH2)와 임계값(TH3)은 상기 수학식 5를 만족하도록 결정될 수 있다.

[0275] 상기 인코더는 상기 S2310, S2340, S2360, S2370 단계들의 결과들을 비교하여, 최적의 코딩 모드를 결정할 수 있다(S2380).

[0276] 이후, 상기 인코더는 최적의 코딩 모드에 기초하여 현재 블록의 움직임 벡터 예측자를 생성할 수 있고, 상기 현재 블록의 움직임 벡터에서 상기 움직임 벡터 예측자를 감산하여 움직임 벡터 차이값을 획득할 수 있다.

[0277] 이후, 상기 도 1 및 도 2에서 설명하였던 인코딩/디코딩 과정이 동일하게 적용될 수 있다.

[0278] 도 24는 본 발명이 적용되는 실시예(5-2)로서, 조건 A(condition A), 조건 B(condition B) 또는 조건 C(condition C) 중 적어도 하나에 기초하여, AF4 모드 또는 AF6 모드에 따라 적응적으로 디코딩을 수행하는 흐름도를 나타낸다.

[0279] 디코더는, 현재 블록에 대해 조건 A가 만족되는지 여부를 확인할 수 있다(S2410). 여기서, 상기 조건 A는 블록 크기에 대한 조건을 의미할 수 있다. 예를 들어, 상기 표 1의 실시예들이 적용될 수 있다.

[0280] 상기 조건 A가 만족되는 경우, 상기 디코더는 상기 현재 블록의 코딩 모드가 AF 모드인지 여부를 확인할 수 있다(S2420). 여기서, 상기 AF 모드는 어파인 움직임 모델(affine motion model)을 이용하는 어파인 움직임 예측 모드를 의미하며, 본 명세서에서 설명한 실시예들이 적용될 수 있으며, 중복되는 설명은 생략하도록 한다.

[0281] 상기 조건 A가 만족되지 않는 경우이거나, 상기 현재 블록에 대해 AF 모드가 수행되지 않는 경우, 상기 디코더는 상기 AF 모드가 아닌 코딩 모드에 따라 디코딩(즉, 움직임 벡터 예측)을 수행할 수 있다(S2480). 예를 들어, 스킵 모드, 머지 모드 또는 인터 모드가 이용될 수 있다.

[0282] 상기 현재 블록에 대해 AF 모드가 수행되는 경우, 상기 디코더는 상기 현재 블록에 대해 조건 B가 만족되는지 여부를 확인할 수 있다(S2430). 여기서, 상기 조건 B는 블록 크기에 대한 조건을 의미할 수 있다. 예를 들어, 상기 표 2의 실시예들이 적용될 수 있으며, 중복되는 설명은 생략하도록 한다.

[0283] 상기 조건 B가 만족되는 경우, 상기 디코더는 AF4 모드에 기초하여 움직임 벡터 예측을 수행할 수 있다(S2440).

[0284] 상기 조건 B가 만족되지 않는 경우, 상기 디코더는 상기 현재 블록에 대해 조건 C가 만족되는지 여부를 확인할 수 있다(S2450). 여기서, 상기 조건 C는 블록 크기에 대한 조건을 의미할 수 있다. 예를 들어, 상기 표 3의 실시예들이 적용될 수 있으며, 중복되는 설명은 생략하도록 한다.

[0285] 한편, 상기 조건 B(condition B) 및 상기 조건 C(condition C)에서, 임계값(TH2)와 임계값(TH3)은 상기 수학식 5를 만족하도록 결정될 수 있다.

[0286] 상기 조건 C가 만족되는 경우, 상기 디코더는 AF6 모드에 기초하여 움직임 벡터 예측을 수행할 수 있다(S2470).

[0287] 상기 조건 C가 만족되지 않는 경우, 상기 디코더는 상기 현재 블록에 대해 AF4 모드가 적용되는지 여부를 확인할 수 있다(S2460).

[0288] 이때, 상기 S2460 단계는 AF4 모드가 수행되는지 여부(또는 4개의 파라미터에 의해 어파인 움직임 예측이 수행되는지 여부)를 나타내는 어파인 파라미터 플래그에 의해 확인될 수 있다.

[0289] 예를 들어, 상기 어파인 파라미터 플래그는 affine_param_flag 로 표현될 수 있다. 상기 affine_param_flag = 0 이면, AF4 모드에 따라 움직임 벡터 예측이 수행되고(S2440), 상기 affine_param_flag = 1 이면, AF6 모드에 따라 움직임 벡터 예측이 수행되는 것을 의미할 수 있으나(S2470), 본 발명은 이에 한정되지 않는다.

[0290] 도 25은 본 발명이 적용되는 실시예(5-3)로서, 조건 A(condition A), 조건 B(condition B) 또는 조건 C(condition C) 중 적어도 하나에 기초하여, AF4 모드 또는 AF6 모드에 따라 디코딩을 수행하는 신택스 구조를 나타낸다.

[0291] 디코더는 merge_flag를 획득하여, 현재 블록에 대해 머지 모드가 적용되는지 여부를 확인할 수 있다(S2510).

[0292] 상기 현재 블록에 대해 머지 모드가 적용되지 않는 경우, 상기 디코더는 조건 A가 만족되는지 여부를 확인할 수 있다(S2520). 여기서, 상기 조건 A는 블록 크기에 대한 조건을 의미할 수 있다. 예를 들어, 상기 표 1의 실시예들이 적용될 수 있다.

[0293] 상기 조건 A가 만족되는 경우, 상기 디코더는 affine_flag 를 획득할 수 있다(S2520). 여기서, affine_flag 는 AF 모드가 수행되는지 여부를 나타낸다.

[0294] 상기 affine_flag = 1 이면, 즉 현재 블록에 대해 AF 모드가 수행되면, 상기 디코더는 조건 B가 만족되는지 여부를 확인할 수 있다(S2530). 여기서, 상기 조건 B는 블록 크기에 대한 조건을 의미할 수 있다. 예를 들어, 상기 표 2의 실시예들이 적용될 수 있다.

[0295] 상기 조건 B가 만족되는 경우, 상기 디코더는 affine_param_flag 를 0으로 설정할 수 있다(S2540). 여기서, affine_param_flag 는 AF4 모드가 수행되는지 여부(또는 4개의 파라미터에 의해 어파인 움직임 예측이 수행되는지 여부)를 나타낸다. affine_param_flag = 0 은, AF4 모드에 따라 움직임 벡터 예측을 수행하는 것을 의미한다.

[0296] 상기 조건 B가 만족되지 않고 조건 C가 만족되는 경우, 상기 디코더는 affine_param_flag 를 1로 설정할 수 있다(S2550). 여기서, affine_param_flag = 1 은, AF6 모드에 따라 움직임 벡터 예측을 수행하는 것을 의미한다.

[0297] 반면, 상기 조건 B가 만족되지 않고 상기 조건 C도 만족되지 않는 경우, 상기 디코더는 affine_param_flag 를 획득할 수 있다(S2560).

[0298] 상기 affine_param_flag = 0 이면, 상기 디코더는 2개의 움직임 벡터 차이값인, mvd_CP0 및 mvd_CP1을 획득할 수 있다(S2570).

[0299] 상기 affine_param_flag = 1 이면, 상기 디코더는 3개의 움직임 벡터 차이값인, mvd_CP0, mvd_CP1 및 mvd_CP2를 획득할 수 있다(S2580).

[0300] 도 26은 본 발명이 적용되는 실시예(6-1)로서, 조건 A(condition A) 또는 이웃 블록의 코딩 모드 중 적어도 하나에 기초하여 AF4 모드 또는 AF6 모드를 포함하는 움직임 벡터 예측 모드들 중 최적의 코딩 모드를 적응적으로 결정하는 흐름도를 나타낸다.

[0301] 인코더는 스킵 모드, 머지 모드, 또는 인터 모드 중 적어도 하나에 기초하여 예측을 수행할 수 있다(S2610).

[0302] 상기 인코더는, 현재 블록에 대해 조건 A가 만족되는지 여부를 확인할 수 있다(S2620). 여기서, 상기 조건 A는 블록 크기에 대한 조건을 의미할 수 있으며, 상기 표 1의 실시예들이 적용될 수 있다.

[0303] 상기 조건 A가 만족되는 경우, 상기 인코더는 AF 모드를 제외한 모드들 중에서 최적의 코딩 모드를 결정할 수 있다(S2660).

[0304] 한편, 상기 조건 A가 만족되지 않는 경우, 상기 인코더는 이웃 블록이 AF 모드로 코딩되었는지 여부를 확인할 수 있다(S2630). 여기서, 이웃 블록이 AF 모드로 코딩되었는지 여부는 isNeighborAffine() 로 표현될 수 있다. 예를 들어, isNeighborAffine() = 0 이면, 이웃 블록은 AF 모드로 코딩되지 않은 경우를 의미하고, isNeighborAffine() = 1 이면, 이웃 블록은 AF 모드로 코딩된 경우를 의미할 수 있다.

[0305] 상기 이웃 블록이 AF 모드로 코딩되지 않은 경우, 상기 인코더는 AF4 모드에 기초하여 움직임 벡터 예측을 수행할 수 있다(S2640).

[0306] 상기 이웃 블록이 AF 모드로 코딩된 경우, 상기 인코더는 AF4 모드에 기초하여 움직임 벡터 예측을 수행하고, 또한 AF6 모드에 기초하여 움직임 벡터 예측을 수행할 수 있다(S2650).

[0307] 상기 인코더는 상기 S2610, 상기 S2640 및 상기 S2650 단계들의 결과들을 비교하여, 최적의 코딩 모드를 결정할 수 있다(S2660).

[0308] 이후, 상기 인코더는 최적의 코딩 모드에 기초하여 현재 블록의 움직임 벡터 예측자를 생성할 수 있고, 상기 현재 블록의 움직임 벡터에서 상기 움직임 벡터 예측자를 감산하여 움직임 벡터 차이값을 획득할 수 있다.

[0309] 이후, 상기 도 1 및 도 2에서 설명하였던 인코딩/디코딩 과정이 동일하게 적용될 수 있다.

[0310] 도 27은 본 발명이 적용되는 실시예(6-2)로서, 조건 A(condition A) 또는 이웃 블록의 코딩 모드 중 적어도 하나에 기초하여 AF4 모드 또는 AF6 모드에 따라 적응적으로 디코딩을 수행하는 흐름도를 나타낸다.

[0311] 디코더는 비트스트림을 수신할 수 있다(S2710). 상기 비트스트림은 비디오 신호 내 현재 블록의 코딩 모드에 대한 정보를 포함할 수 있다.

[0312] 상기 디코더는, 움직임 벡터 예측을 위한 최적의 코딩 모드를 결정하기 위해 현재 블록에 대해 조건 A가 만족되는지 여부를 확인할 수 있다(S2720). 여기서, 상기 조건 A는 블록 크기에 대한 조건을 의미할 수 있다. 예를 들어, 상기 표 1의 실시예들이 적용될 수 있다.

[0313] 상기 조건 A가 만족되는 경우, 상기 디코더는 상기 현재 블록의 코딩 모드가 AF 모드인지 여부를 확인할 수 있다(S2730).

[0314] 이하, S2730 단계 내지 S2780 단계들은, 상기 도 21의 S2120 단계 내지 S2170 단계에서 설명한 내용들이 적용될 수 있으며, 중복되는 설명은 생략하도록 한다.

[0315] 도 28은 본 발명이 적용되는 실시예(6-3)로서, 조건 A(condition A) 또는 이웃 블록의 코딩 모드 중 적어도 하나에 기초하여 AF4 모드 또는 AF6 모드에 따라 디코딩을 수행하는 신택스 구조를 나타낸다.

[0316] 디코더는 merge_flag를 획득하여, 현재 블록에 대해 머지 모드가 적용되는지 여부를 확인할 수 있다(S2810).

[0317] 상기 현재 블록에 대해 머지 모드가 적용되지 않는 경우, 상기 디코더는 조건 A가 만족되는지 여부를 확인할 수 있다(S2820). 여기서, 상기 조건 A는 블록 크기에 대한 조건을 의미할 수 있다. 예를 들어, 상기 표 1의 실시예들이 적용될 수 있다.

[0318] 상기 조건 A가 만족되는 경우, 상기 디코더는 affine_flag 를 획득할 수 있다(S2820). 여기서, affine_flag 는 AF 모드가 수행되는지 여부를 나타낸다.

[0319] 이하, S2830 단계 내지 S2860 단계들은, 상기 도 22의 S2230 단계 내지 S2260 단계에서 설명한 내용들이 적용될 수 있으며, 중복되는 설명은 생략하도록 한다.

[0320] 도 29는 본 발명이 적용되는 실시예로서, AF4 모드 또는 AF6 모드 중 적어도 하나에 기초하여 움직임 벡터 예측자를 생성하는 흐름도를 나타낸다.

[0321] 디코더는, 현재 블록에 대해 AF 모드가 적용되는지 여부를 확인할 수 있다(S2910). 여기서 상기 AF 모드는 어파인 움직임 모델(affine motion model)을 이용하는 움직임 예측 모드를 나타낸다.

[0322] 예를 들어, 상기 디코더는 비디오 신호로부터 어파인 플래그를 획득할 수 있고, 상기 현재 블록에 대해 상기 AF 모드가 적용되는지 여부는 상기 어파인 플래그에 기초하여 확인될 수 있다.

[0323] 상기 디코더는, 상기 현재 블록에 대해 상기 AF 모드가 적용되는 경우, AF4 모드가 이용되는지 여부를 확인할 수 있다(S2920). 여기서 상기 AF4 모드는 상기 어파인 움직임 모델(affine motion model)을 구성하는 4개의 파라미터를 이용하여 움직임 벡터를 예측하는 모드를 나타낸다.

[0324] 예를 들어, 상기 어파인 플래그에 따라 상기 현재 블록에 대해 상기 AF 모드가 적용되는 경우, 상기 디코더는 상기 비디오 신호로부터 어파인 파라미터 플래그를 획득할 수 있고, 상기 어파인 파라미터 플래그는 상기 움직임 벡터 예측자가 상기 4개의 파라미터를 이용하여 생성되는지 또는 상기 6개의 파라미터를 이용하여 생성되는지 여부를 나타낸다.

[0325] 여기서, 상기 어파인 플래그 및 상기 어파인 파라미터 플래그는 슬라이스, 최대 코딩 유닛, 코딩 유닛 또는 예측 유닛 중 적어도 하나의 레벨에서 정의될 수 있다.

[0326] 상기 디코더는, AF4 모드가 이용되면 상기 4개 파라미터를 이용하여 움직임 벡터 예측자를 생성하고, AF4 모드가 이용되지 않으면 상기 어파인 움직임 모델(affine motion model)을 구성하는 6개 파라미터를 이용하여 움직임 벡터 예측자를 생성할 수 있다(S2930).

[0327] 상기 디코더는, 상기 움직임 벡터 예측자에 기초하여 상기 현재 블록의 움직임 벡터를 획득할 수 있다(S2940).

[0328] 일실시예로, 상기 디코더는, 상기 현재 블록의 크기가 기설정된 조건을 만족하는지를 확인할 수 있다. 이때, 상기 기설정된 조건은 상기 현재 블록 내 픽셀 개수, 상기 현재 블록의 너비 및/또는 높이 중 적어도 하나가 기설정된 임계값보다 큰지 여부를 나타낸다.

[0329] 예를 들어, 상기 현재 블록의 크기가 기설정된 조건을 만족하는 경우, 상기 디코더는 상기 현재 블록에 대해 상기 AF 모드가 적용되는지 여부를 확인할 수 있다.

[0330] 반면, 상기 현재 블록의 크기가 기설정된 조건을 만족하지 않는 경우, 상기 현재 블록은 상기 AF 모드가 아닌 다른 코딩 모드에 기초하여 디코딩될 수 있다.

[0331] 일실시예로, 상기 디코더는, 상기 현재 블록에 대해 상기 AF 모드가 적용되는 경우, 이웃 블록에 대해 AF 모드가 적용되었는지 여부를 확인할 수 있다.

[0332] 상기 이웃 블록에 대해 AF 모드가 적용된 경우, 움직임 벡터 예측자는 상기 4개 파라미터를 이용하여 생성되고, 상기 이웃 블록에 대해 AF 모드가 적용되지 않은 경우, 상기 디코더는 상기 AF4 모드가 이용되는지 여부를 확인하는 단계를 수행할 수 있다.

[0333] 도 30은 본 발명이 적용되는 실시예로서, AF4_flag 및 AF6_flag 에 기초하여 움직임 벡터 예측자를 생성하는 흐름도를 나타낸다.

[0334] 디코더는 비디오 신호로부터 AF4 flag 및 AF6 flag 중 적어도 하나를 획득할 수 있다(S3010). 여기서, AF4_flag 는 현재 블록에 대해 AF4 모드가 수행되는지 여부를 나타내고, AF6_flag 는 현재 블록에 대해 AF6 모드가 수행되는지 여부를 나타낸다.

[0335] 이때, 상기 AF4 flag 및 상기 AF6 flag 중 적어도 하나는 슬라이스 레벨에서 정의되고, 다시 블록 레벨 또는 예측 유닛 레벨에서 정의될 수 있다. 다만, 본 발명은 이에 한정되지 않으며, 상기 AF4 flag 및 상기 AF6 flag 중 적어도 하나는 슬라이스, 최대 코딩 유닛, 코딩 유닛 또는 예측 유닛 중 적어도 하나의 레벨에서 정의될 수 있

다.

[0336] 상기 디코더는 AF4 flag 및 AF6 flag 의 값들을 확인할 수 있다(S3020).

[0337] AF4_flag = 1이면 현재 블록에 대해 AF4 모드가 수행되고, AF4_flag = 0이면 현재 블록에 대해 AF4 모드가 수행되지 않는다. 이때, AF4 모드가 수행된다는 것은 4개의 파라미터로 표현되는 어파인 움직임 모델을 이용하여 움직임 벡터 예측을 수행하는 것을 의미한다.

[0338] AF6_flag = 1이면 현재 블록에 대해 AF6 모드가 수행되고, AF4_flag = 0이면 현재 블록에 대해 AF6 모드가 수행되지 않는다. 여기서, AF6 모드가 수행된다는 것은 4개의 파라미터로 표현되는 어파인 움직임 모델을 이용하여 움직임 벡터 예측을 수행하는 것을 의미한다.

[0339] AF4 flag =0 , AF6 flag =0 이면, 상기 디코더는 AF4 모드 및 AF6 모드 이외의 모드에 따라 움직임 벡터 예측을 수행할 수 있다(S3030).

[0340] AF4 flag =1 , AF6 flag =0 이면, 상기 디코더는 AF4 모드에 따라 움직임 벡터 예측을 수행할 수 있다(S3040).

[0341] AF4 flag =0 , AF6 flag =1 이면, 상기 디코더는 AF6 모드에 따라 움직임 벡터 예측을 수행할 수 있다(S3050).

[0342] AF4 flag =1 , AF6 flag =1 이면, 상기 디코더는 AF4 모드 또는 AF6 모드에 따라 움직임 벡터 예측을 수행할 수 있다(S3060).

[0343] 도 31은 본 발명이 적용되는 실시예로서, 이웃 블록이 AF 모드로 코딩되었는지 여부에 기초하여 AF4 모드 또는 AF6 모드에 따라 적응적으로 디코딩을 수행하는 흐름도를 나타낸다.

[0344] 디코더는 현재 블록에 대해 AF 모드가 적용되는지 여부를 확인할 수 있다(S3110).

[0345] 상기 디코더는 상기 현재 블록에 대해 상기 AF 모드가 적용되는 경우, 이웃 블록이 AF 모드로 코딩되었는지 여부를 확인할 수 있다(S3120).

[0346] 상기 디코더는, 이웃 블록이 AF 모드로 코딩된 경우, AF4 flag 또는 AF6_flag 중 적어도 하나를 획득할 수 있다(S3130).

[0347] 상기 디코더는 AF4 flag 또는 AF6_flag 중 적어도 하나에 기초하여 4개 또는 6개 파라미터를 이용하여 움직임 벡터 예측자를 생성할 수 있다(S3140). 예를 들어, AF4 flag =1 이면, 상기 디코더는 AF4 모드에 따라 움직임 벡터 예측을 수행할 수 있고, AF6 flag =1 이면, 상기 디코더는 AF6 모드에 따라 움직임 벡터 예측을 수행할 수 있다.

[0348] 상기 디코더는 상기 움직임 벡터 예측자에 기초하여 상기 현재 블록의 움직임 벡터를 획득할 수 있다(S3150).

[0349] 도 32는 본 발명이 적용되는 실시예로서, AF4_flag 및 AF6_flag 에 기초하여 적응적으로 디코딩을 수행하는 신택스를 나타낸다.

[0350] 디코더는 슬라이스 레벨에서 AF4 flag 및 AF6 flag를 획득할 수 있다(S3010). 여기서, AF4_flag 는 현재 블록에 대해 AF4 모드가 수행되는지 여부를 나타내고, AF6_flag 는 현재 블록에 대해 AF6 모드가 수행되는지 여부를 나타낸다. 상기 AF4_flag 는 affine_4_flag 로 표현될 수 있고, 상기 AF6_flag 는 affine_6_flag 로 표현될 수 있다.

[0351] 상기 디코더는, 블록 레벨 또는 예측 유닛 레벨에서 AF4_flag 및 AF6_flag 에 기초하여 적응적으로 디코딩을 수행할 수 있다.

[0352] affine_4_flag 가 0이 아니거나, affine_6_flag 가 0이 아닌 경우 (즉, affine_4_flag = 0 && affine_6_flag = 0 외의 경우), 상기 디코더는 어파인 플래그를 획득할 수 있다(S3220). 상기 어파인 플래그는 AF 모드가 수행되는지 여부를 나타낼 수 있다.

[0353] AF 모드가 수행되는 경우, 상기 디코더는 AF4_flag 및 AF6_flag 값에 따라 적응적으로 디코딩을 수행할 수 있다.

[0354] affine_4_flag = 1 && affine_6_flag = 0 인 경우, 상기 디코더는 affine_param_flag 를 0으로 설정할 수 있다. 즉, affine_param_flag = 0 은 AF4 모드가 수행되는 것을 의미한다.

[0355] affine_4_flag = 0 && affine_6_flag = 1 인 경우, 상기 디코더는 affine_param_flag 를 1로 설정할 수 있다.

즉, affine_param_flag = 1 은 AF6 모드가 수행되는 것을 의미한다.

[0356] affine_4_flag = 1 && affine_6_flag = 1 인 경우, 상기 디코더는 affine_param_flag 를 파싱 또는 획득할 수 있다. 이때, 상기 디코더는 블록 레벨 또는 예측 유닛 레벨에서 affine_param_flag 값에 따라 AF4 모드 또는 AF6 모드에 따라 디코딩을 수행할 수 있다.

[0357] 그 외 신택스 구조는 앞서 설명한 실시예들이 적용될 수 있으며, 중복되는 설명은 생략한다.

[0358] 도 33은 본 발명이 적용되는 실시예로서, 이웃 블록이 AF 모드로 코딩되었는지 여부에 기초하여 AF4 모드 또는 AF6 모드에 따라 적응적으로 디코딩을 수행하는 신택스를 나타낸다.

[0359] 본 실시예의 경우, 도 32와 중복되는 내용은 앞선 설명이 적용될 수 있으며, 다른 부분만 설명하도록 한다.

[0360] affine_4_flag = 1 && affine_6_flag = 1 인 경우, 상기 디코더는 이웃 블록이 AF 모드로 코딩되었는지 여부를 확인할 수 있다.

[0361] 이웃 블록이 AF 모드로 코딩된 경우, 상기 디코더는 affine_param_flag 를 파싱 또는 획득할 수 있다(S3310). 이때, 상기 디코더는 블록 레벨 또는 예측 유닛 레벨에서 affine_param_flag 값에 따라 AF4 모드 또는 AF6 모드에 따라 디코딩을 수행할 수 있다.

[0362] 반면, 이웃 블록이 AF 모드로 코딩되지 않은 경우, 상기 디코더는 affine_param_flag를 0으로 설정할 수 있다. 즉, affine_param_flag = 0 은 AF4 모드가 수행되는 것을 의미한다.

[0363] 도 34는 본 발명이 적용되는 비디오 코딩 시스템을 나타낸다.

[0364] 비디오 코딩 시스템은 소스 디바이스(source device) 및 수신 디바이스(receiving device)를 포함할 수 있다. 소스 디바이스는 인코딩된 비디오/영상 정보 또는 데이터를 파일 또는 스트리밍 형태로 디지털 저장매체 또는 네트워크를 통하여 수신 디바이스로 전달할 수 있다.

[0365] 상기 소스 디바이스는 비디오 소스(video source), 인코딩 장치(encoding apparatus), 전송부(transmitter)를 포함할 수 있다. 상기 수신 디바이스는 수신부(receiver), 디코딩 장치(decoding apparatus) 및 렌더러(renderer)를 포함할 수 있다. 상기 인코딩 장치는 비디오/영상 인코딩 장치라고 불릴 수 있고, 상기 디코딩 장치는 비디오/영상 디코딩 장치라고 불릴 수 있다. 송신기는 인코딩 장치에 포함될 수 있다. 수신기는 디코딩 장치에 포함될 수 있다. 렌더러는 디스플레이부를 포함할 수도 있고, 디스플레이부는 별개의 디바이스 또는 외부 컴포넌트로 구성될 수도 있다.

[0366] 비디오 소스는 비디오/영상의 캡쳐, 합성 또는 생성 과정 등을 통하여 비디오/영상을 획득할 수 있다. 비디오 소스는 비디오/영상 캡쳐 디바이스 및/또는 비디오/영상 생성 디바이스를 포함할 수 있다. 비디오/영상 캡쳐 디바이스는 예를 들어, 하나 이상의 카메라, 이전에 캡쳐된 비디오/영상을 포함하는 비디오/영상 아카이브 등을 포함할 수 있다. 비디오/영상 생성 디바이스는 예를 들어 컴퓨터, 타블렛 및 스마트폰 등을 포함할 수 있으며 (전자적으로) 비디오/영상을 생성할 수 있다. 예를 들어, 컴퓨터 등을 통하여 가상의 비디오/영상이 생성될 수 있으며, 이 경우 관련 데이터가 생성되는 과정으로 비디오/영상 캡쳐 과정이 갈음될 수 있다.

[0367] 인코딩 장치는 입력 비디오/영상을 인코딩할 수 있다. 인코딩 장치는 압축 및 코딩 효율을 위하여 예측, 변환, 양자화 등 일련의 절차를 수행할 수 있다. 인코딩된 데이터(인코딩된 비디오/영상 정보)는 비트스트림(bitstream) 형태로 출력될 수 있다.

[0368] 전송부는 비트스트림 형태로 출력된 인코딩된 비디오/영상 정보 또는 데이터를 파일 또는 스트리밍 형태로 디지털 저장매체 또는 네트워크를 통하여 수신 디바이스의 수신부로 전달할 수 있다. 디지털 저장 매체는 USB, SD, CD, DVD, 블루레이, HDD, SSD 등 다양한 저장 매체를 포함할 수 있다. 전송부는 미리 정해진 파일 포멧을 통하여 미디어 파일을 생성하기 위한 엘리먼트를 포함할 수 있고, 방송/통신 네트워크를 통한 전송을 위한 엘레멘트를 포함할 수 있다. 수신부는 상기 비트스트림을 추출하여 디코딩 장치로 전달할 수 있다.

[0369] 디코딩 장치는 인코딩 장치의 동작에 대응하는 역양자화, 역변환, 예측 등 일련의 절차를 수행하여 비디오/영상을 디코딩할 수 있다.

[0370] 렌더러는 디코딩된 비디오/영상을 렌더링할 수 있다. 렌더링된 비디오/영상은 디스플레이부를 통하여 디스플레이될 수 있다.

[0371] 도 35는 본 발명이 적용되는 컨텐츠 스트리밍 시스템을 나타낸다.

[0372] 상기 도 35를 살펴보면, 본 발명이 적용되는 컨텐츠 스트리밍 시스템은 크게 인코딩 서버, 스트리밍 서버, 웹 서버, 미디어 저장소, 사용자 장치 및 멀티미디어 입력 장치를 포함할 수 있다.

[0373] 상기 인코딩 서버는 스마트폰, 카메라, 캠코더 등과 같은 멀티미디어 입력 장치들로부터 입력된 컨텐츠를 디지털 데이터로 압축하여 비트스트림을 생성하고 이를 상기 스트리밍 서버로 전송하는 역할을 한다. 다른 예로, 스마트폰, 카메라, 캠코더 등과 같은 멀티미디어 입력 장치들이 비트스트림을 직접 생성하는 경우, 상기 인코딩 서버는 생략될 수 있다.

[0374] 상기 비트스트림은 본 발명이 적용되는 인코딩 방법 또는 비트스트림 생성 방법에 의해 생성될 수 있고, 상기 스트리밍 서버는 상기 비트스트림을 전송 또는 수신하는 과정에서 일시적으로 상기 비트스트림을 저장할 수 있다.

[0375] 상기 스트리밍 서버는 웹 서버를 통한 사용자 요청에 기초하여 멀티미디어 데이터를 사용자 장치에 전송하고, 상기 웹 서버는 사용자에게 어떠한 서비스가 있는지를 알려주는 매개체 역할을 한다. 사용자가 상기 웹 서버에 원하는 서비스를 요청하면, 상기 웹 서버는 이를 스트리밍 서버에 전달하고, 상기 스트리밍 서버는 사용자에게 멀티미디어 데이터를 전송한다. 이때, 상기 컨텐츠 스트리밍 시스템은 별도의 제어 서버를 포함할 수 있고, 이 경우 상기 제어 서버는 상기 컨텐츠 스트리밍 시스템 내 각 장치 간 명령/응답을 제어하는 역할을 한다.

[0376] 상기 스트리밍 서버는 미디어 저장소 및/또는 인코딩 서버로부터 컨텐츠를 수신할 수 있다. 예를 들어, 상기 인코딩 서버로부터 컨텐츠를 수신하게 되는 경우, 상기 컨텐츠를 실시간으로 수신할 수 있다. 이 경우, 원활한 스트리밍 서비스를 제공하기 위하여 상기 스트리밍 서버는 상기 비트스트림을 일정 시간동안 저장할 수 있다.

[0377] 상기 사용자 장치의 예로는, 휴대폰, 스마트 폰(smart phone), 노트북 컴퓨터(laptop computer), 디지털방송용 단말기, PDA(personal digital assistants), PMP(portable multimedia player), 네비게이션, 슬레이트 PC(slate PC), 태블릿 PC(tablet PC), 울트라북(ultrabook), 웨어러블 디바이스(wearable device, 예를 들어, 워치형 단말기 (smartwatch), 글래스형 단말기 (smart glass), HMD(head mounted display)), 디지털 TV, 데스크탑 컴퓨터, 디지털 사이니지 등이 있을 수 있다.

[0378] 상기 컨텐츠 스트리밍 시스템 내 각 서버들은 분산 서버로 운영될 수 있으며, 이 경우 각 서버에서 수신하는 데이터는 분산 처리될 수 있다.

[0379] 상기 기술된 것과 같이, 본 발명에서 설명한 실시예들은 프로세서, 마이크로 프로세서, 컨트롤러 또는 칩 상에서 구현되어 수행될 수 있다. 예를 들어, 상기 도 1, 도 2, 도 34 및 도 35에서 도시한 기능 유닛들은 컴퓨터, 프로세서, 마이크로 프로세서, 컨트롤러 또는 칩 상에서 구현되어 수행될 수 있다.

[0380] 또한, 본 발명이 적용되는 디코더 및 인코더는 멀티미디어 방송 송수신 장치, 모바일 통신 단말, 홈 시네마 비디오 장치, 디지털 시네마 비디오 장치, 감시용 카메라, 비디오 대화 장치, 비디오 통신과 같은 실시간 통신 장치, 모바일 스트리밍 장치, 저장 매체, 캠코더, 주문형 비디오(VoD) 서비스 제공 장치, 인터넷 스트리밍 서비스 제공 장치, 3차원(3D) 비디오 장치, 화상 전화 비디오 장치, 및 의료용 비디오 장치 등에 포함될 수 있으며, 비디오 신호 및 데이터 신호를 처리하기 위해 사용될 수 있다.

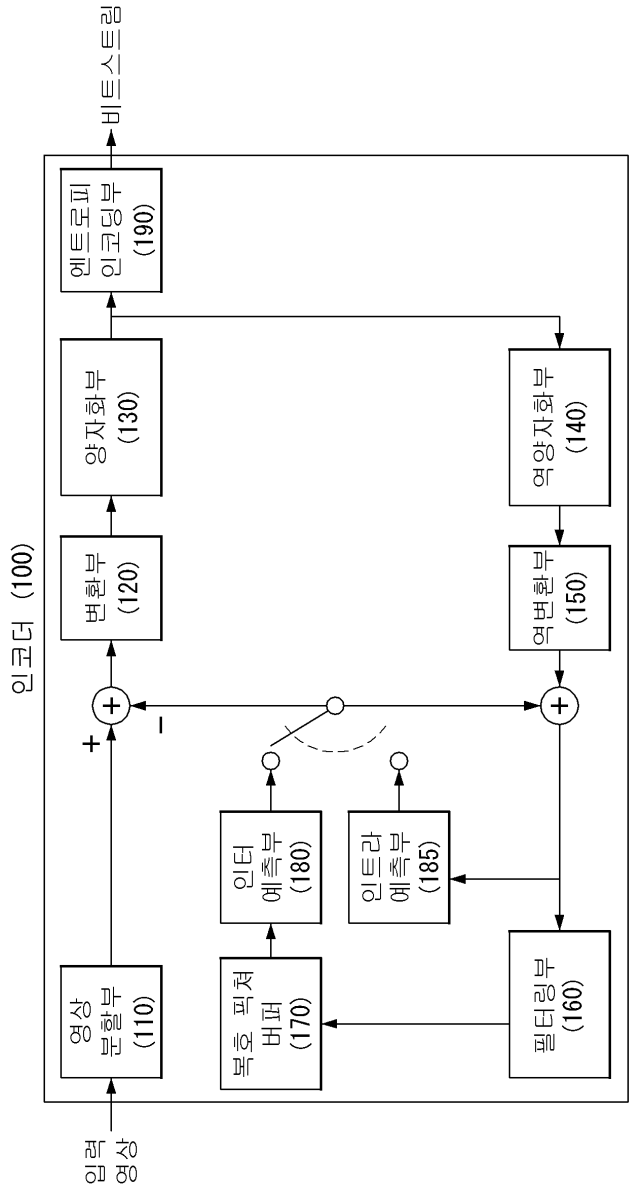
[0381] 또한, 본 발명이 적용되는 처리 방법은 컴퓨터로 실행되는 프로그램의 형태로 생산될 수 있으며, 컴퓨터가 판독할 수 있는 기록 매체에 저장될 수 있다. 본 발명에 따른 데이터 구조를 가지는 멀티미디어 데이터도 또한 컴퓨터가 판독할 수 있는 기록 매체에 저장될 수 있다. 상기 컴퓨터가 판독할 수 있는 기록 매체는 컴퓨터로 읽을 수 있는 데이터가 저장되는 모든 종류의 저장 장치를 포함한다. 상기 컴퓨터가 판독할 수 있는 기록 매체는, 예를 들어, 블루레이 디스크(BD), 범용 직렬 버스(USB), ROM, RAM, CD-ROM, 자기 테이프, 플로피 디스크 및 광학적 데이터 저장 장치를 포함할 수 있다. 또한, 상기 컴퓨터가 판독할 수 있는 기록 매체는 반송파(예를 들어, 인터넷을 통한 전송)의 형태로 구현된 미디어를 포함한다. 또한, 인코딩 방법으로 생성된 비트 스트림이 컴퓨터가 판독할 수 있는 기록 매체에 저장되거나 유무선 통신 네트워크를 통해 전송될 수 있다.

산업상 이용가능성

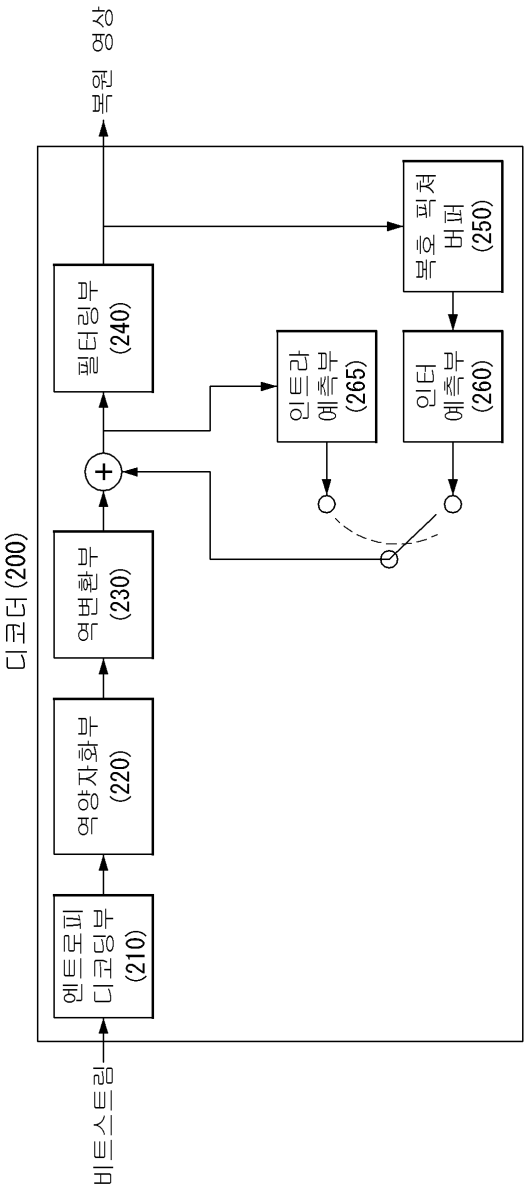
[0382] 이상, 전술한 본 발명의 바람직한 실시예는, 예시의 목적을 위해 개시된 것으로, 당업자라면 이하 첨부된 특허청구범위에 개시된 본 발명의 기술적 사상과 그 기술적 범위 내에서, 다양한 다른 실시예들을 개량, 변경, 대체 또는 부가 등이 가능할 것이다.

도면

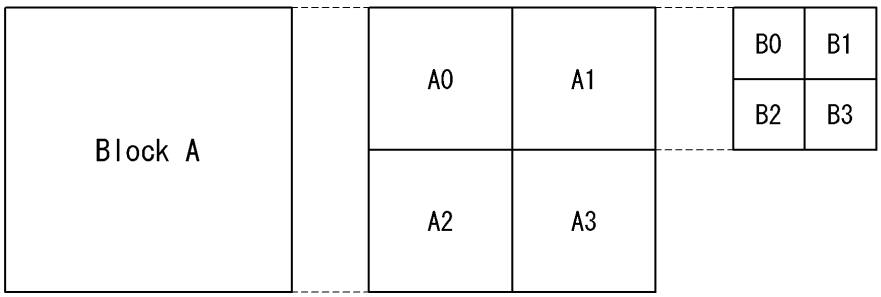
도면1


비트스트림
엔트로피 인코딩부 (190)
양자화부 (130)
변환부 (120)
인코더 (100)
영상 분할부 (110)
입력 영상
역양자화부 (140)
역변환부 (150)
필터링부 (160)
복호 픽쳐 버퍼 (170)
인터 예측부 (180)
인트라 예측부 (185)
+
-

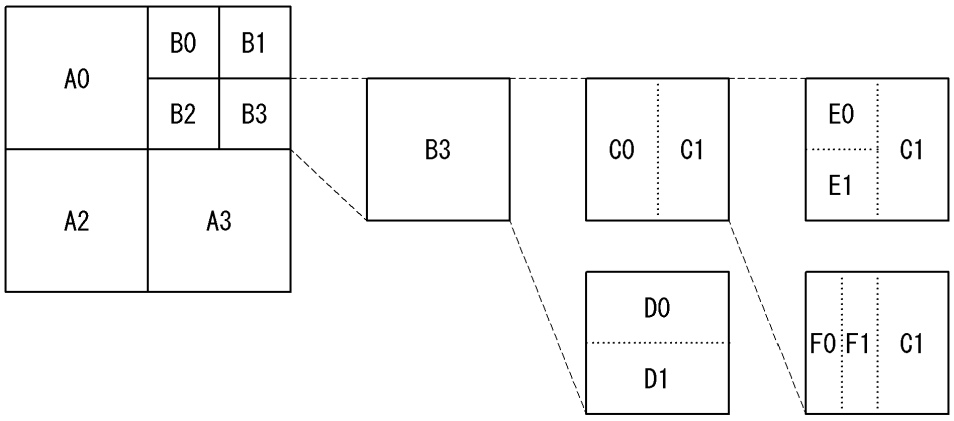
도면2


비트스트림
디코더 (200)
엔트로피 디코딩부 (210)
역양자화부 (220)
역변환부 (230)
필터링부 (240)
복호 픽처 버퍼 (250)
인터 예측부 (260)
인트라 예측부 (265)
복원 영상

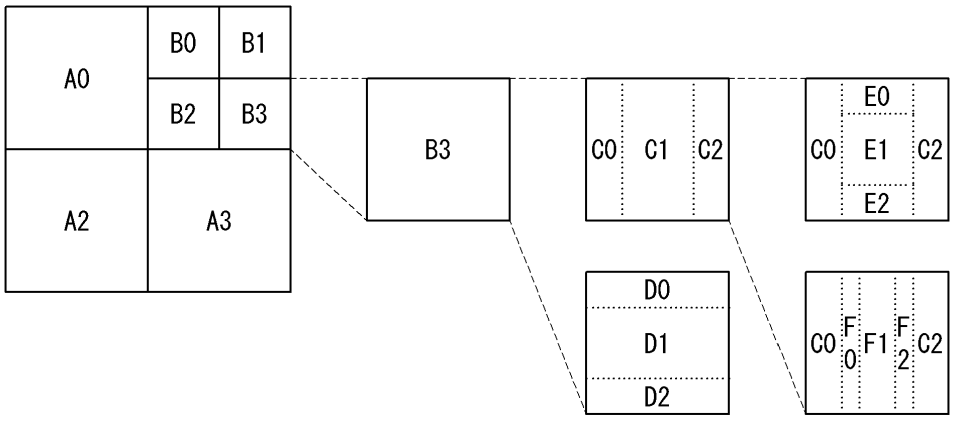
도면3


Block A
A0
A1
A2
A3
B0
B1
B2
B3

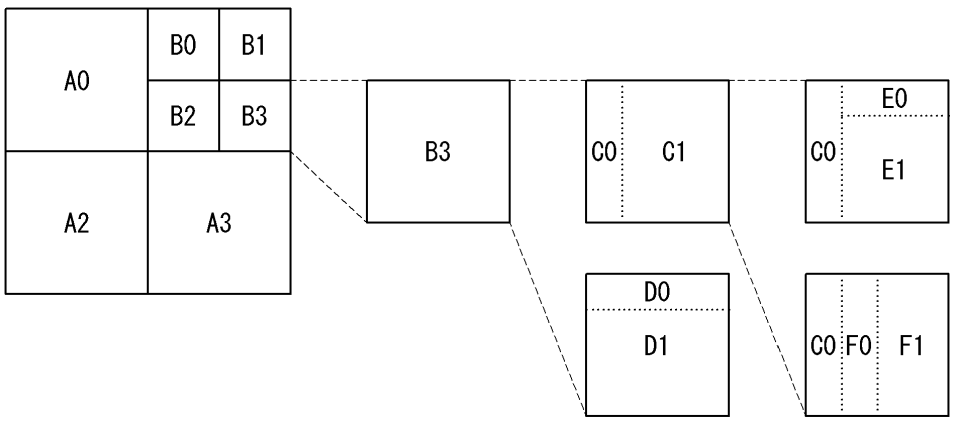
도면4


A0
B0
B1
B2
B3
A2
A3
B3
C0
C1
E0
E1
C1
D0
D1
F0
F1
C1

도면5

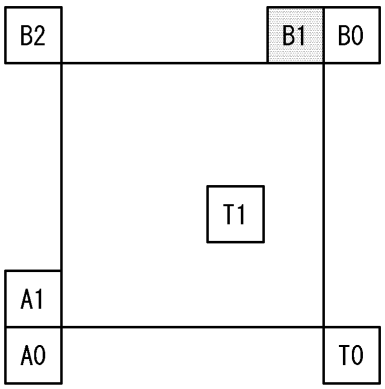

A0
B0
B1
B2
B3
A2
A3
B3
C0
C1
C2
E0
E1
E2
C0
C2
D0
D1
D2
C0
F0
F1
F2
C2

도면6


A0
B0
B1
B2
B3
A2
A3
B3
C0
C1
E0
E1
C0
D0
D1
C0
F0
F1

도면7

(a)



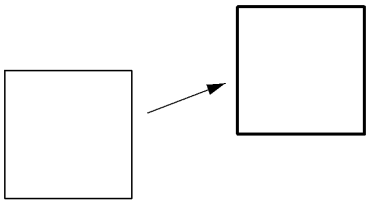
(b)

Merge Candidate List

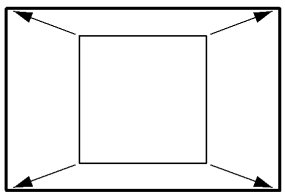
Index	Motion parameter
0	Motion parameter of A1
1	Motion parameter of B1
2	Motion parameter of B0
3	Motion parameter of A0
4	Motion parameter of T0

Signaling of “Index 1”

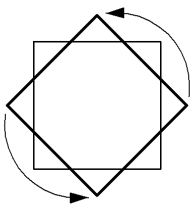
도면8



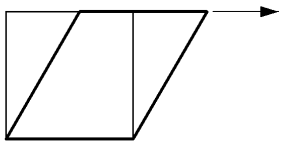
병진



스케일



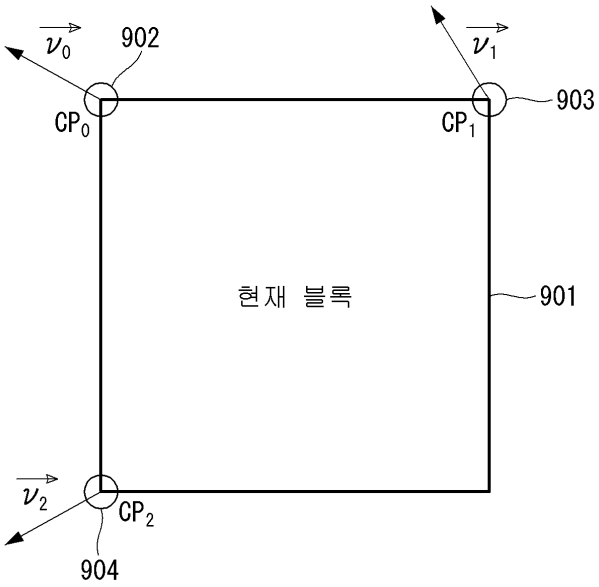
회전



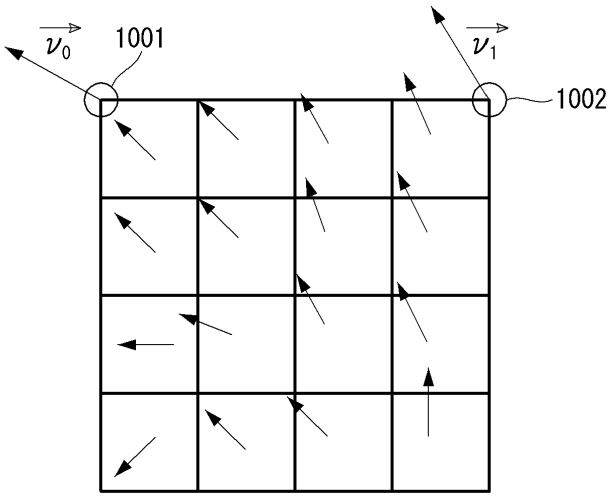
비뚤림

도면9

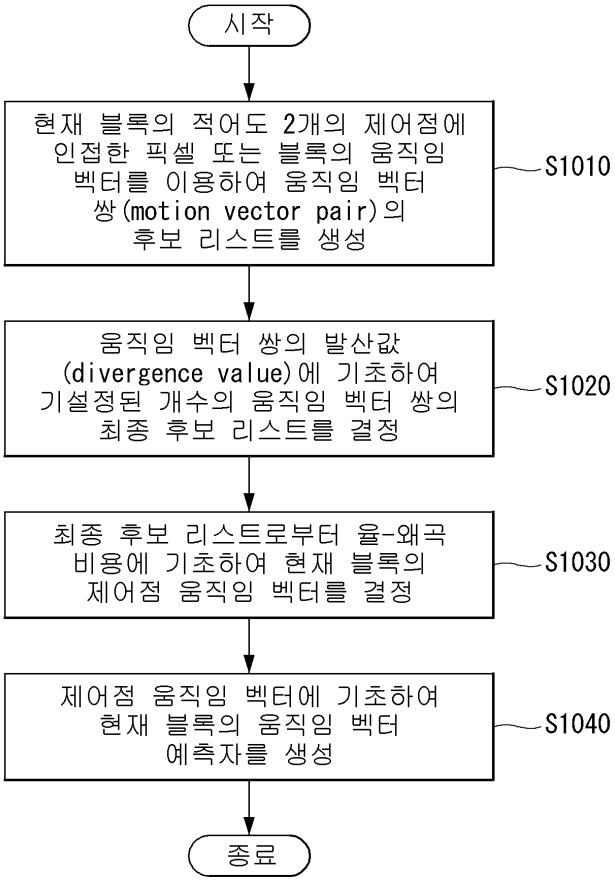
(a)


$\vec{v}_0$
902
$\vec{v}_1$
903
CP_0
CP_1
현재 블록
901
$\vec{v}_2$
CP_2
904

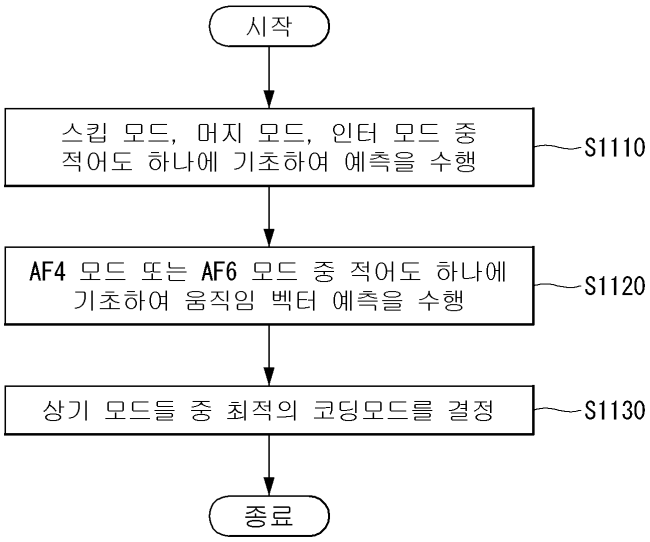
(b)


$\vec{v}_0$
1001
$\vec{v}_1$
1002

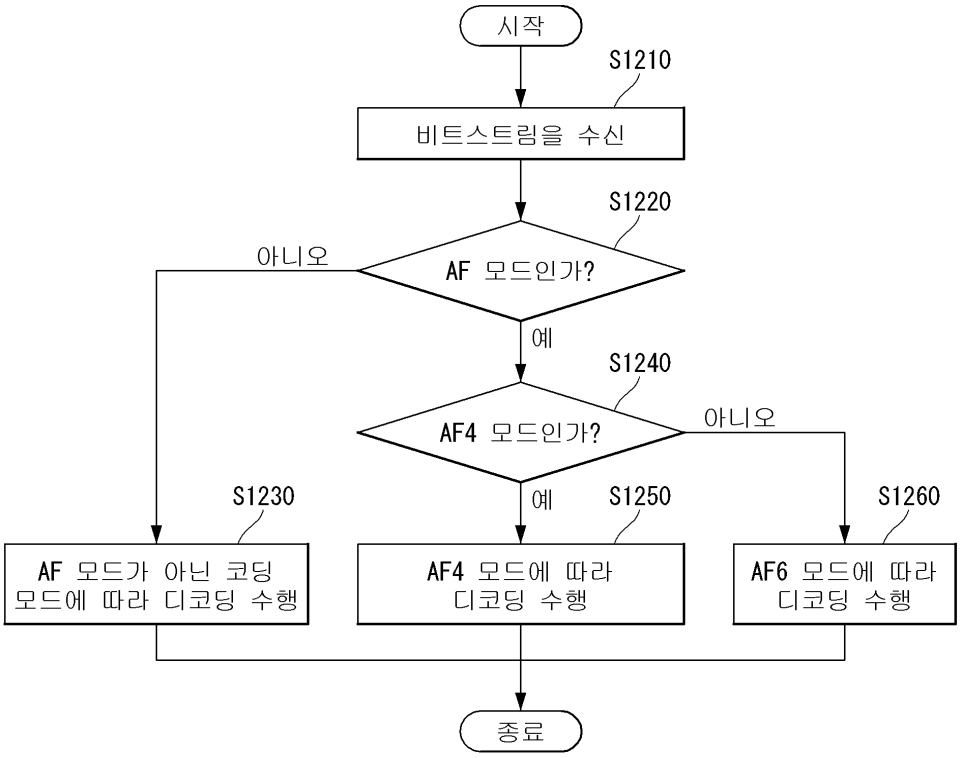
도면10


시작
현재 블록의 적어도 2개의 제어점에 인접한 픽셀 또는 블록의 움직임 벡터를 이용하여 움직임 벡터 쌍(motion vector pair)의 후보 리스트를 생성
S1010
움직임 벡터 쌍의 발산값(divergence value)에 기초하여 기설정된 개수의 움직임 벡터 쌍의 최종 후보 리스트를 결정
S1020
최종 후보 리스트로부터 율-왜곡 비용에 기초하여 현재 블록의 제어점 움직임 벡터를 결정
S1030
제어점 움직임 벡터에 기초하여 현재 블록의 움직임 벡터 예측자를 생성
S1040
종료

도면11


시작
스킵 모드, 머지 모드, 인터 모드 중 적어도 하나에 기초하여 예측을 수행
S1110
AF4 모드 또는 AF6 모드 중 적어도 하나에 기초하여 움직임 벡터 예측을 수행
S1120
상기 모드들 중 최적의 코딩모드를 결정
S1130
종료

도면12



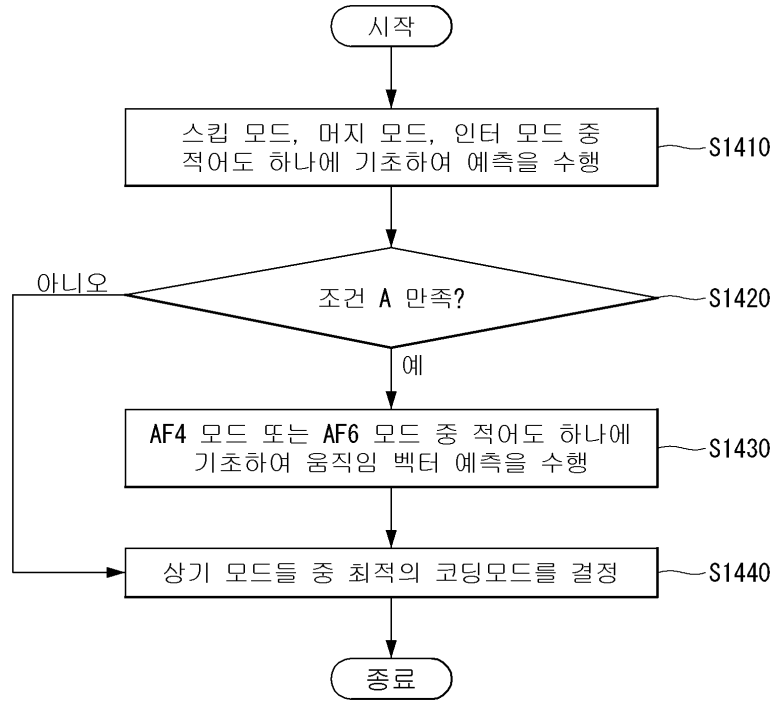
도면13

(S1310)	merge_flag
	if (merge_flag) {
	...
	}
	else {

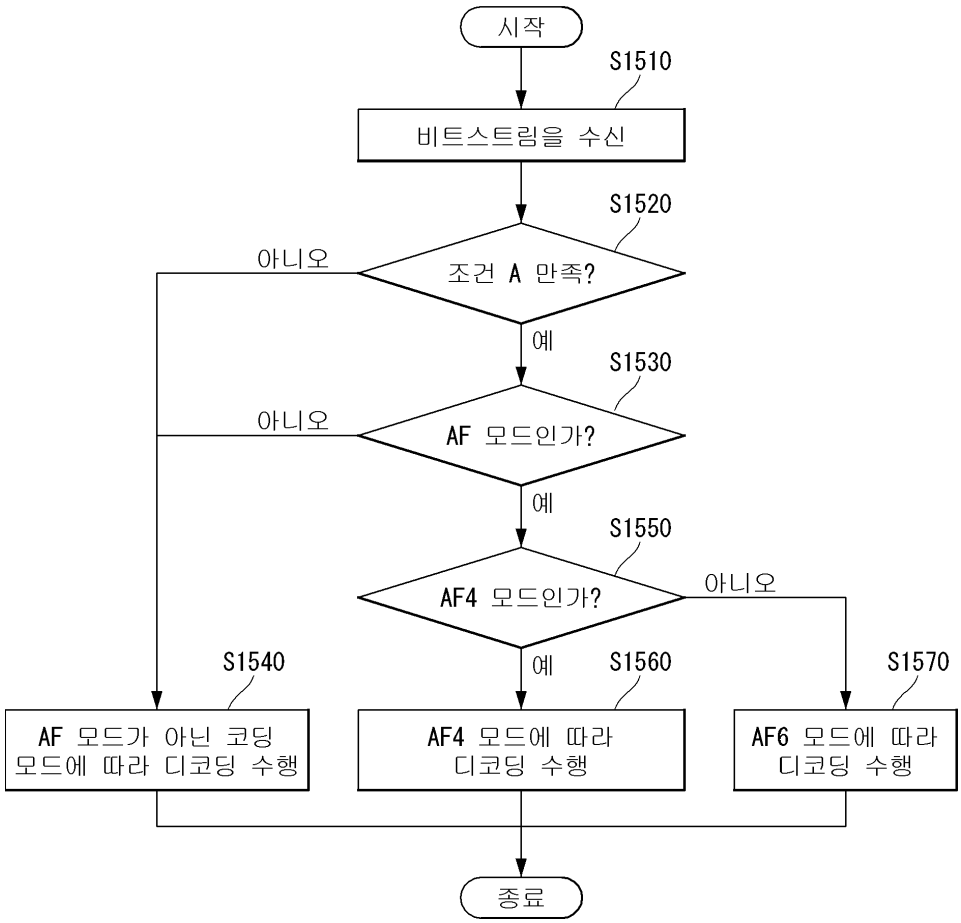
(S1320)	affine_flag if (affine_flag) {
(S1330)	affine_param_flag if (affine_param_flag == 0) {
(S1340)	mvd_CP0 mvd_CP1 }

(S1350)	else {
	mvd_CP0
	mvd_CP1
	mvd_CP2
	}
	}

도면14


시작
스킵 모드, 머지 모드, 인터 모드 중 적어도 하나에 기초하여 예측을 수행
S1410
조건 A 만족?
아니오
S1420
예
AF4 모드 또는 AF6 모드 중 적어도 하나에 기초하여 움직임 벡터 예측을 수행
S1430
상기 모드들 중 최적의 코딩모드를 결정
S1440
종료

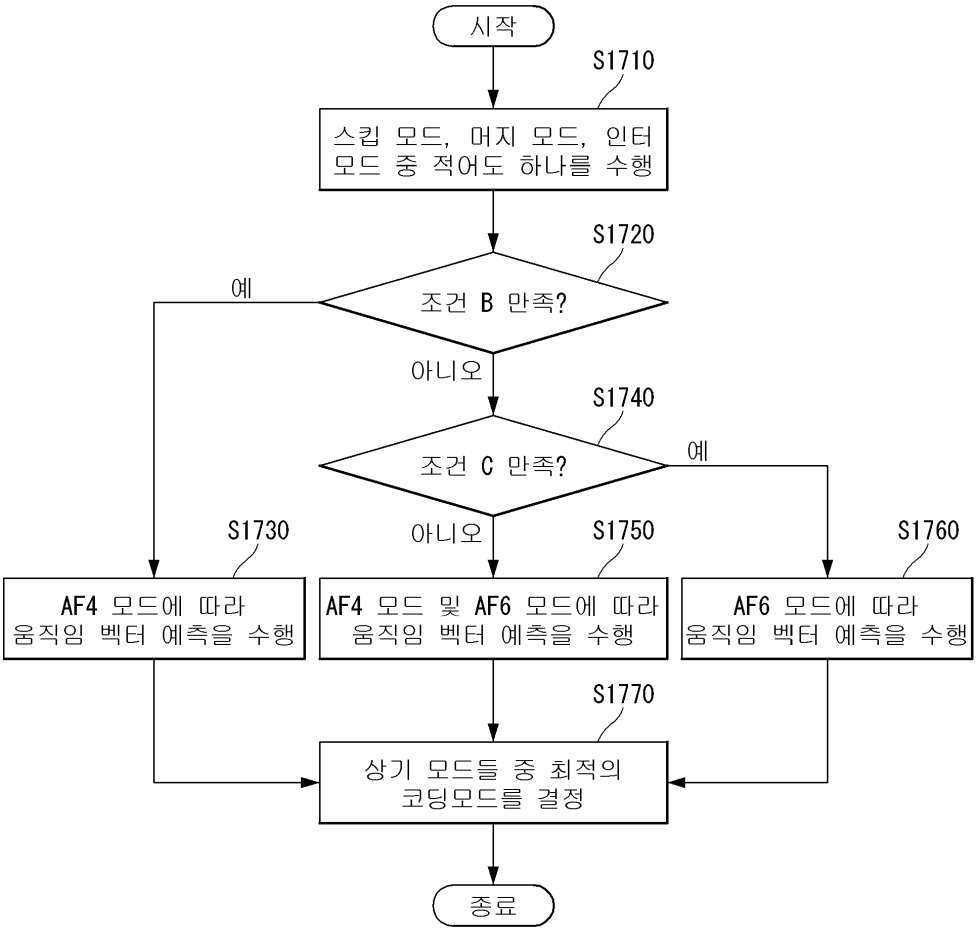
도면15


시작
S1510
비트스트림을 수신
S1520
조건 A 만족?
아니오
예
S1530
AF 모드인가?
아니오
예
S1550
AF4 모드인가?
아니오
예
S1540
AF 모드가 아닌 코딩 모드에 따라 디코딩 수행
S1560
AF4 모드에 따라 디코딩 수행
S1570
AF6 모드에 따라 디코딩 수행
종료

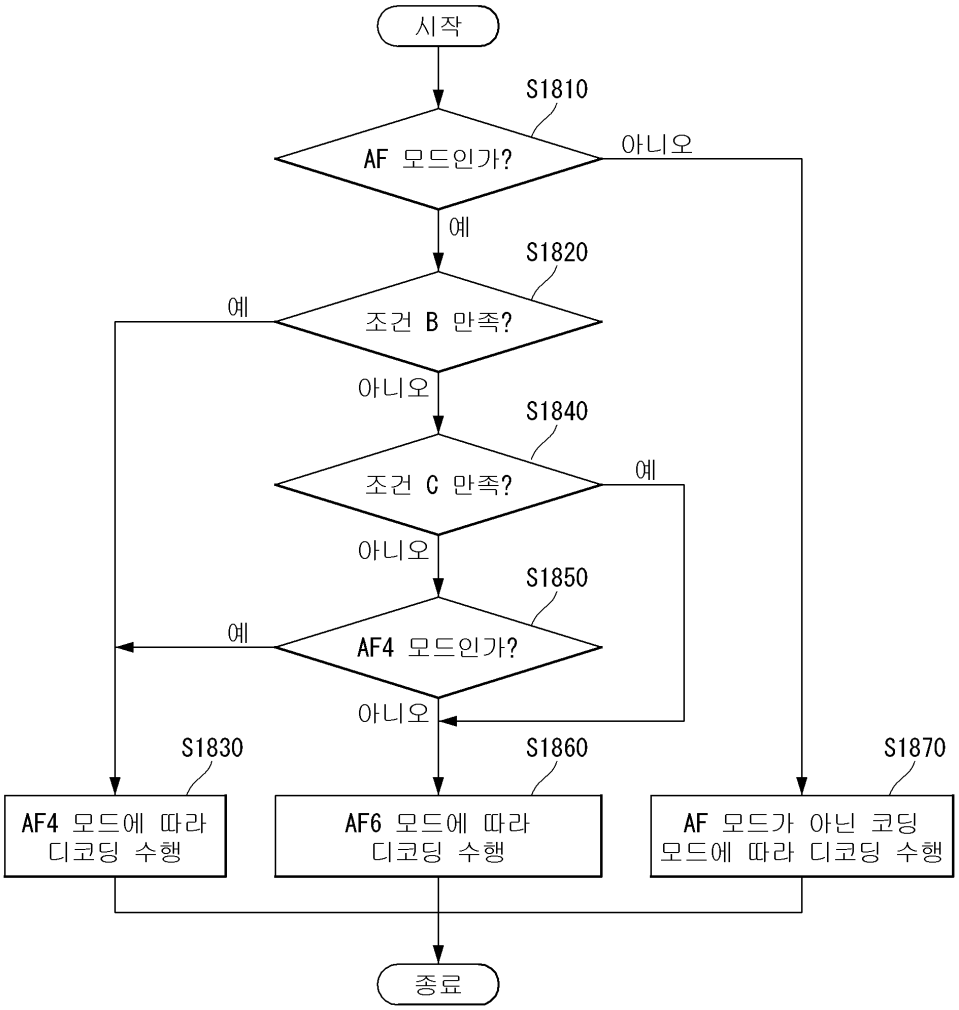
도면16

```
(S1610)   merge_flag
          if (merge_flag) {
            ...
          }
          else {
(S1620)     if (CONDITION A) {
              affine_flag
              if (affine_flag) {
(S1630)         affine_param_flag
                if (affine_param_flag == 0) {
(S1640)           mvd_CP0
                  mvd_CP1
                  }
                else {
(S1650)           mvd_CP0
                  mvd_CP1
                  mvd_CP2
                }
            }
          }
```

도면17


시작
S1710
스킵 모드, 머지 모드, 인터 모드 중 적어도 하나를 수행
S1720
조건 B 만족?
예
아니오
S1740
조건 C 만족?
예
아니오
S1730
AF4 모드에 따라 움직임 벡터 예측을 수행
S1750
AF4 모드 및 AF6 모드에 따라 움직임 벡터 예측을 수행
S1760
AF6 모드에 따라 움직임 벡터 예측을 수행
S1770
상기 모드들 중 최적의 코딩모드를 결정
종료

도면18


시작
S1810
AF 모드인가?
아니오
예
S1820
조건 B 만족?
예
아니오
S1840
조건 C 만족?
예
아니오
S1850
AF4 모드인가?
예
아니오
S1830
AF4 모드에 따라 디코딩 수행
S1860
AF6 모드에 따라 디코딩 수행
S1870
AF 모드가 아닌 코딩 모드에 따라 디코딩 수행
종료

도면19

(S1910)	merge_flag
	if (merge_flag) {
	...
	}
	else {

(S1920)	affine_flag if (affine_flag) {
(S1930)	if (CONDITION B) set affine_param_flag as 0
(S1940)	else if (CONDITION C) set affine_param_flag as 1

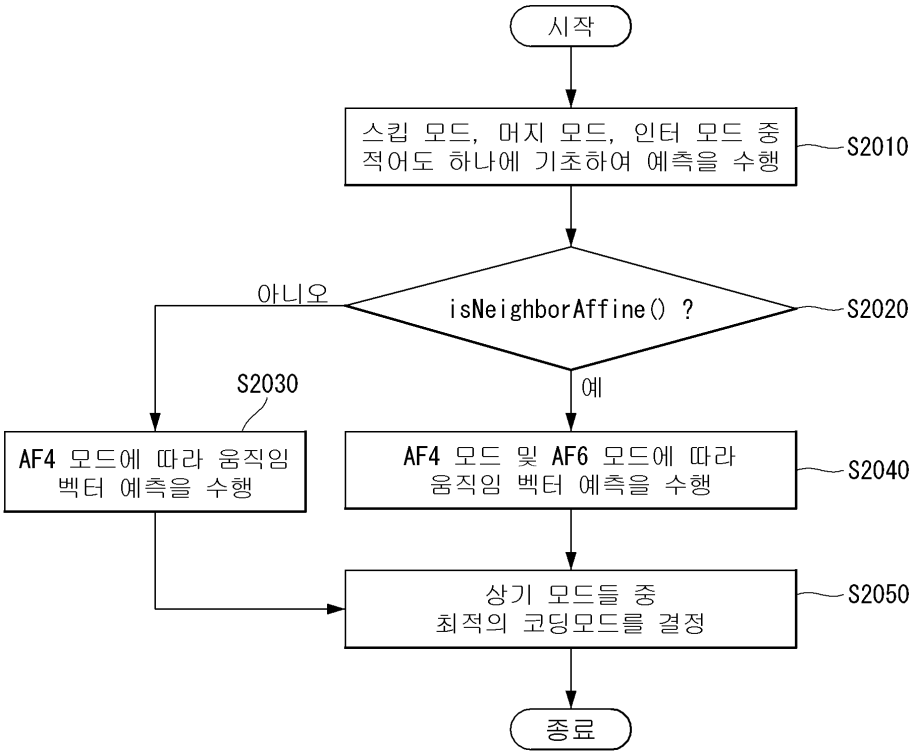
(S1950)	else
	affine_param_flag
	if (affine_param_flag == 0) {

(S1960)	mvd_CP0 mvd_CP1 }

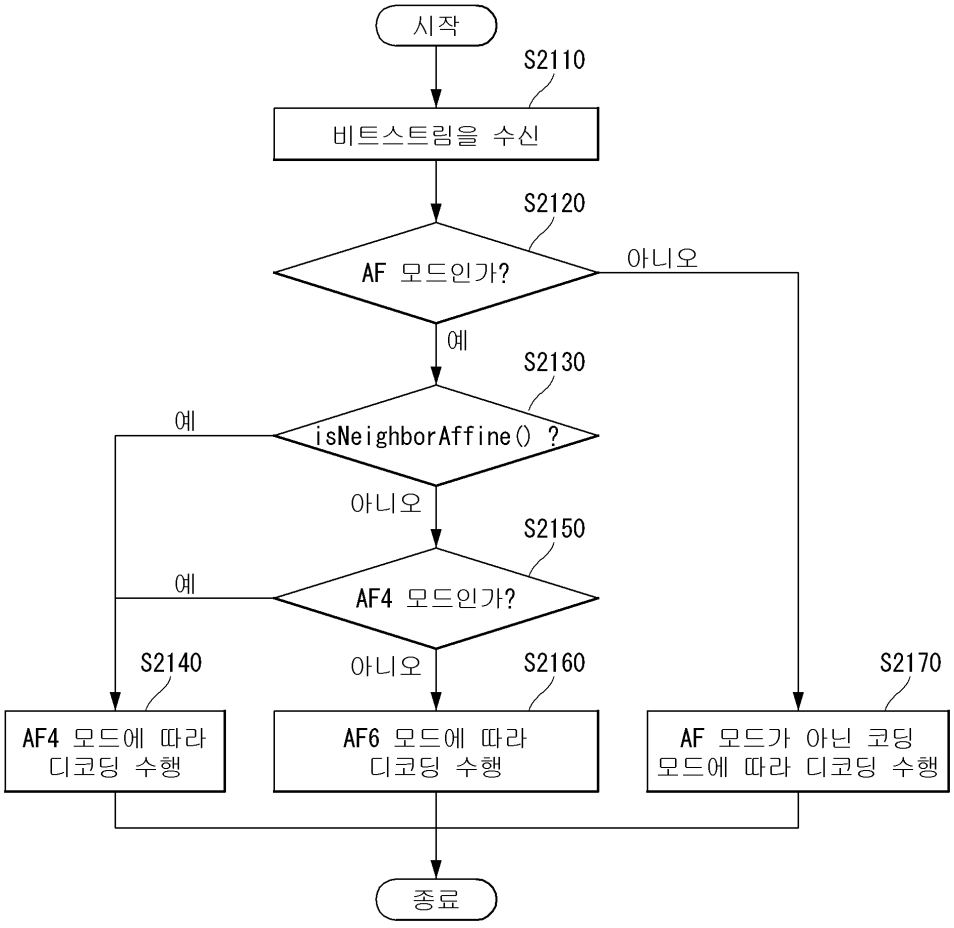
(S1970)	else {
	mvd_CP0
	mvd_CP1
	mvd_CP2
	}

	}
	}

도면20


시작
스킵 모드, 머지 모드, 인터 모드 중 적어도 하나에 기초하여 예측을 수행
S2010
isNeighborAffine() ?
S2020
아니오
예
S2030
AF4 모드에 따라 움직임 벡터 예측을 수행
AF4 모드 및 AF6 모드에 따라 움직임 벡터 예측을 수행
S2040
상기 모드들 중 최적의 코딩모드를 결정
S2050
종료

도면21


시작
S2110
비트스트림을 수신
S2120
AF 모드인가?
아니오
예
S2130
isNeighborAffine() ?
예
아니오
S2150
AF4 모드인가?
예
아니오
S2140
AF4 모드에 따라 디코딩 수행
S2160
AF6 모드에 따라 디코딩 수행
S2170
AF 모드가 아닌 코딩 모드에 따라 디코딩 수행
종료

도면22

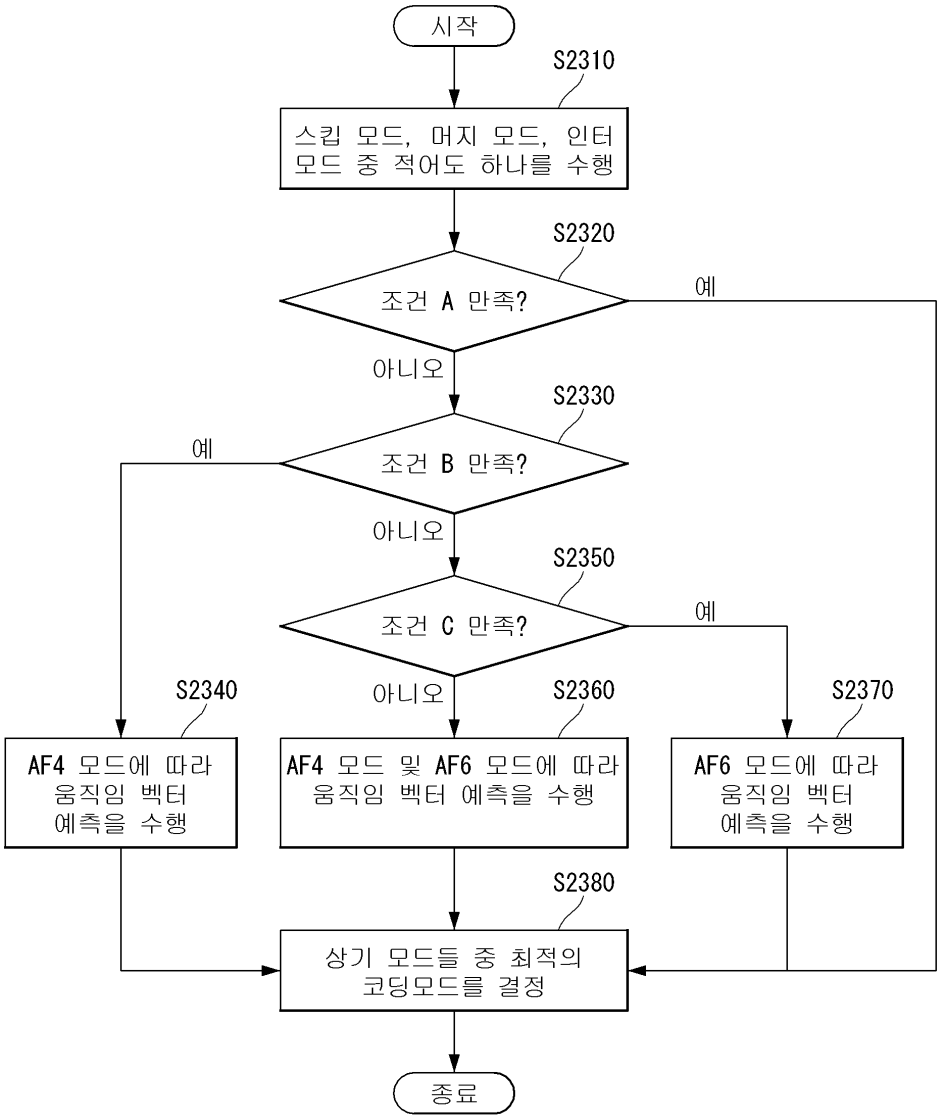
(S2210)	merge_flag
	if (merge_flag) {
	...
	}
	else {

(S2220)	affine_flag
	if (affine_flag) {
(S2230)	if (isNeighborAffine()) affine_param_flag
(S2240)	else set affine_param_flag as 0
	if (affine_param_flag == 0) {
(S2250)	mvd_CP0 mvd_CP1 }

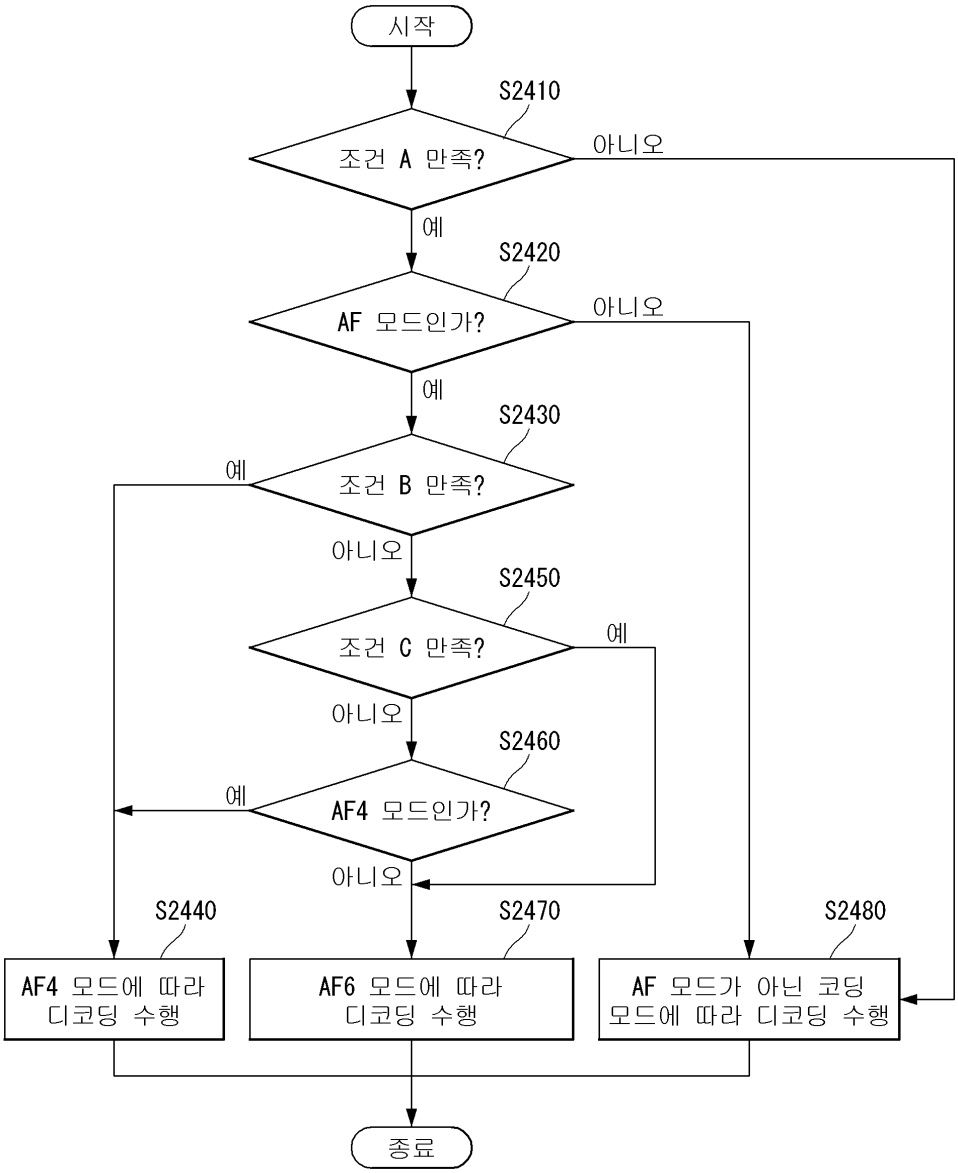
(S2260)	else {
	mvd_CP0
	mvd_CP1
	mvd_CP2
	}

	}
	}

도면23


시작
S2310
스킵 모드, 머지 모드, 인터 모드 중 적어도 하나를 수행
S2320
조건 A 만족?
예
아니오
S2330
조건 B 만족?
예
아니오
S2350
조건 C 만족?
예
아니오
S2340
AF4 모드에 따라 움직임 벡터 예측을 수행
S2360
AF4 모드 및 AF6 모드에 따라 움직임 벡터 예측을 수행
S2370
AF6 모드에 따라 움직임 벡터 예측을 수행
S2380
상기 모드들 중 최적의 코딩모드를 결정
종료

도면24

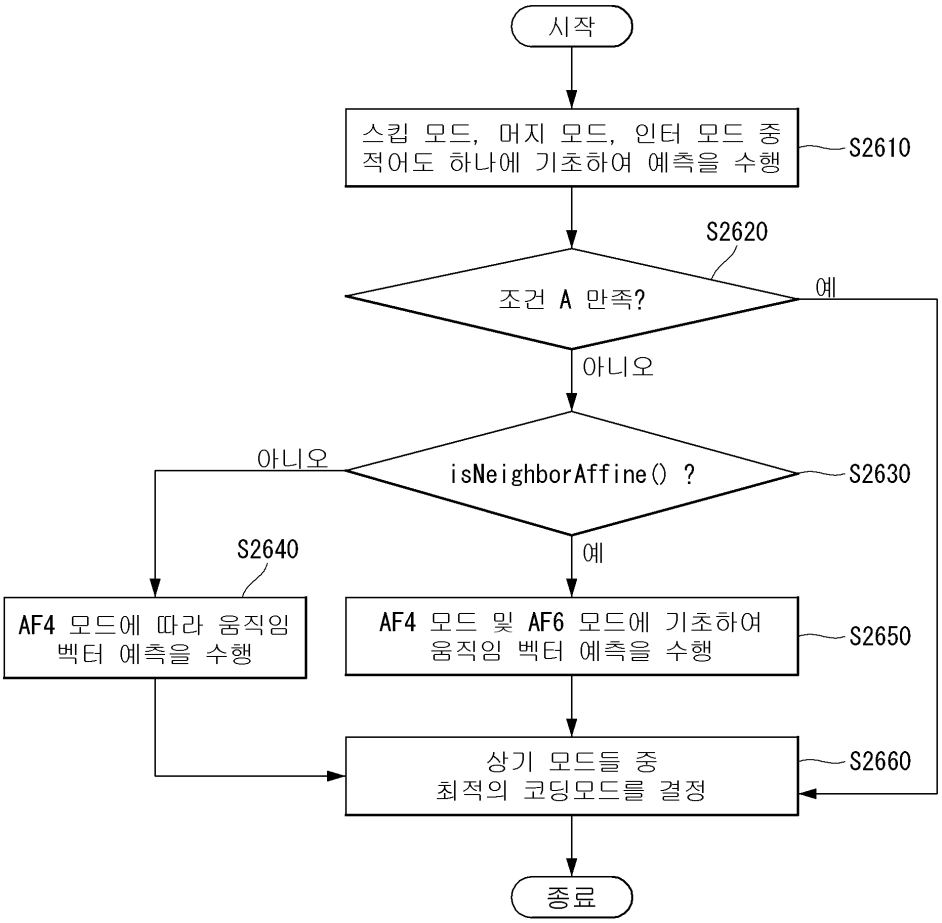

시작
S2410
조건 A 만족?
아니오
예
S2420
AF 모드인가?
아니오
예
S2430
조건 B 만족?
예
아니오
S2450
조건 C 만족?
예
아니오
S2460
AF4 모드인가?
예
아니오
S2440
AF4 모드에 따라 디코딩 수행
S2470
AF6 모드에 따라 디코딩 수행
S2480
AF 모드가 아닌 코딩 모드에 따라 디코딩 수행
종료

도면25

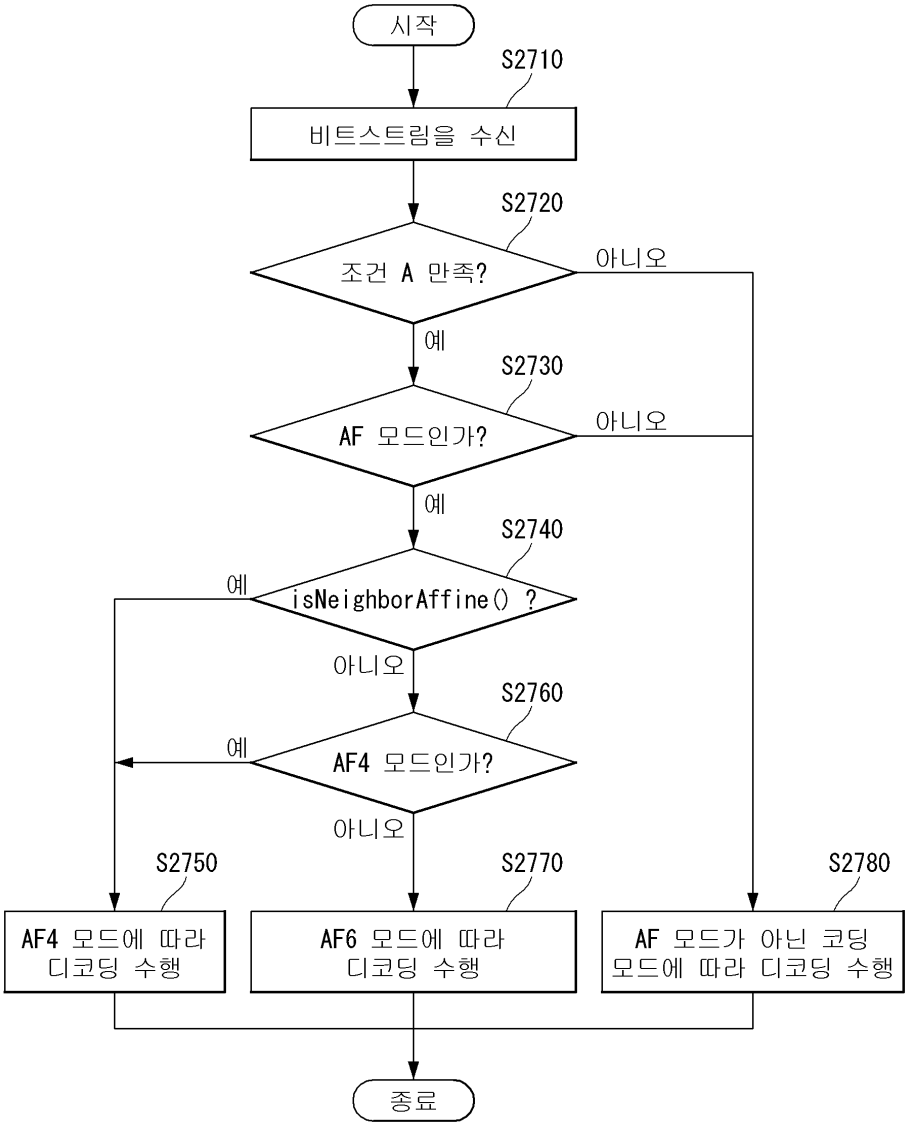
(S2510)	merge_flag
	if (merge_flag) {
	...
	}
	else {
	if (CONDITION A)
	affine_flag
	if (affine_flag) {
	if (CONDITION B)
	set affine_param_flag as 0
	else if (CONDITION C)
	set affine_param_flag as 1
	else
	affine_param_flag
	if (affine_param_flag == 0) {
	mvd_CP0
	mvd_CP1
	}
	else {
	mvd_CP0
	mvd_CP1
	mvd_CP2
	}
	}
	}

(S2520)	
(S2530)	
(S2540)	
(S2550)	
(S2560)	
(S2570)	
(S2580)	

도면26


시작
스킵 모드, 머지 모드, 인터 모드 중 적어도 하나에 기초하여 예측을 수행
S2610
S2620
조건 A 만족?
예
아니오
아니오
isNeighborAffine() ?
S2630
예
S2640
AF4 모드에 따라 움직임 벡터 예측을 수행
AF4 모드 및 AF6 모드에 기초하여 움직임 벡터 예측을 수행
S2650
상기 모드들 중 최적의 코딩모드를 결정
S2660
종료

도면27

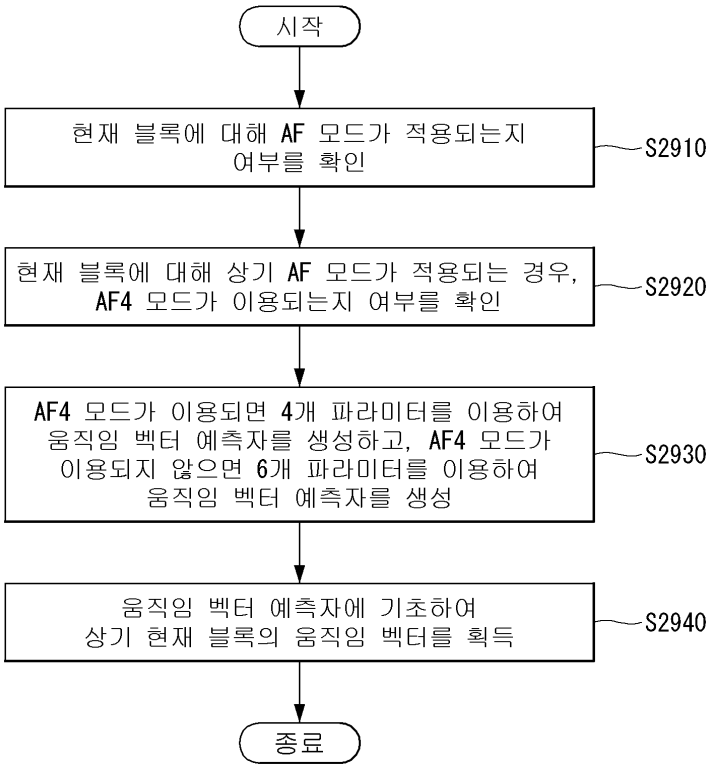

시작
S2710
비트스트림을 수신
S2720
조건 A 만족?
아니오
예
S2730
AF 모드인가?
아니오
예
S2740
예
isNeighborAffine() ?
아니오
S2760
예
AF4 모드인가?
아니오
S2750
S2770
S2780
AF4 모드에 따라 디코딩 수행
AF6 모드에 따라 디코딩 수행
AF 모드가 아닌 코딩 모드에 따라 디코딩 수행
종료

도면28

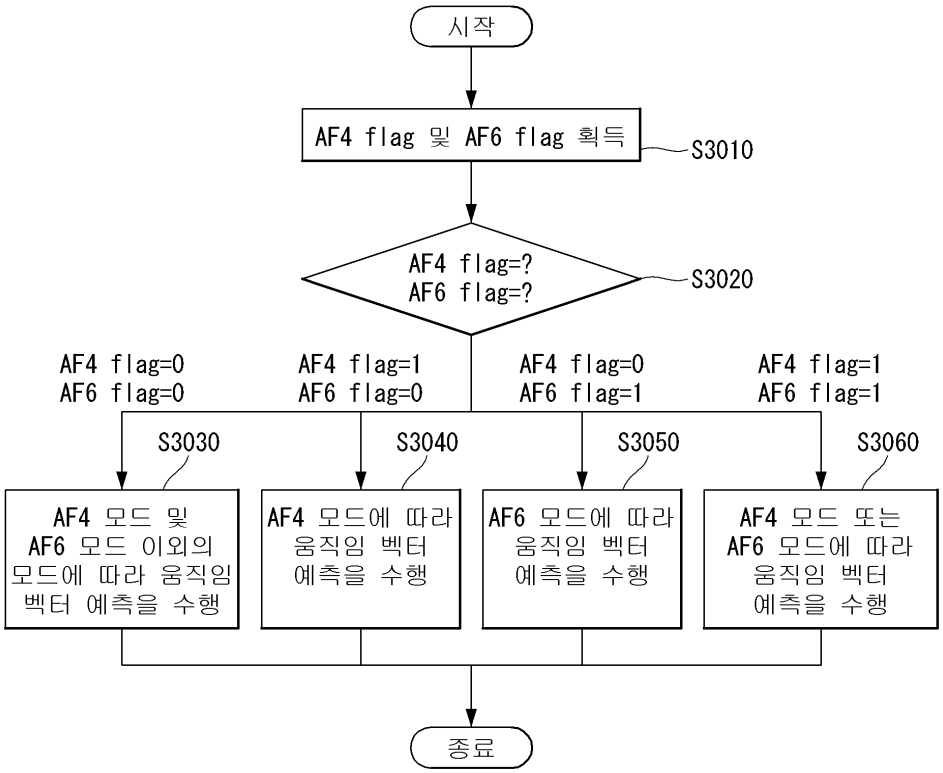
(S2810)	merge_flag
	if (merge_flag) {
	...
	}
	else {
	if (CONDITION A)
	affine_flag
	if (affine_flag) {
	if (isNeighborAffine())
	affine_param_flag
	else
	set affine_param_flag as 0
	if (affine_param_flag == 0) {
	mvd_CP0
	mvd_CP1
	}
	else {
	mvd_CP0
	mvd_CP1
	mvd_CP2
	}
	}
	}

(S2820)	
(S2830)	
(S2840)	
(S2850)	
(S2860)	

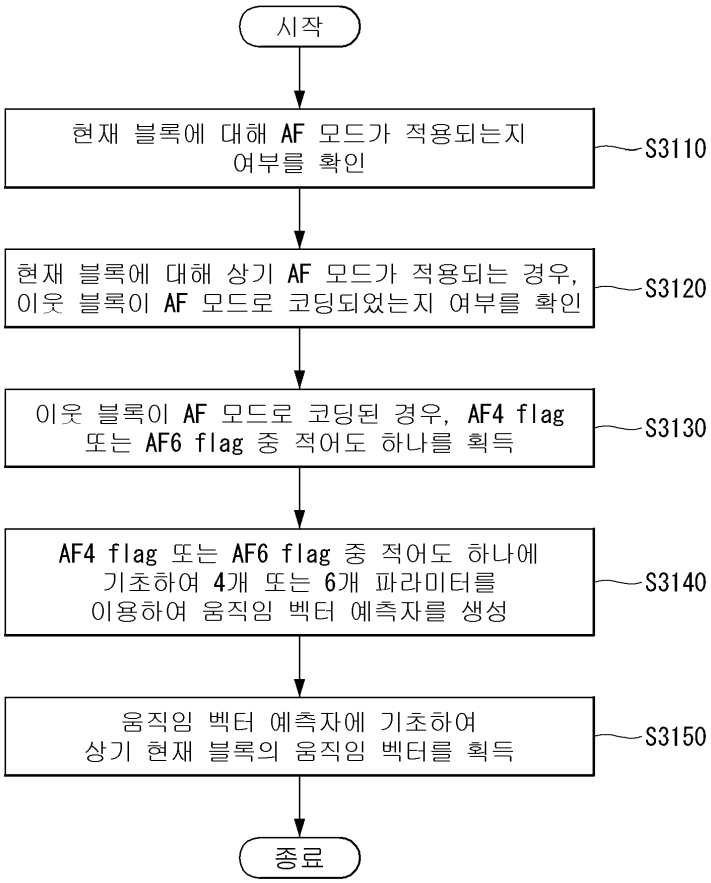
도면29


시작
현재 블록에 대해 AF 모드가 적용되는지 여부를 확인
S2910
현재 블록에 대해 상기 AF 모드가 적용되는 경우, AF4 모드가 이용되는지 여부를 확인
S2920
AF4 모드가 이용되면 4개 파라미터를 이용하여 움직임 벡터 예측자를 생성하고, AF4 모드가 이용되지 않으면 6개 파라미터를 이용하여 움직임 벡터 예측자를 생성
S2930
움직임 벡터 예측자에 기초하여 상기 현재 블록의 움직임 벡터를 획득
S2940
종료

도면30


시작
AF4 flag 및 AF6 flag 획득
S3010
AF4 flag=?
AF6 flag=?
S3020
AF4 flag=0
AF6 flag=0
AF4 flag=1
AF6 flag=0
AF4 flag=0
AF6 flag=1
AF4 flag=1
AF6 flag=1
S3030
S3040
S3050
S3060
AF4 모드 및 AF6 모드 이외의 모드에 따라 움직임 벡터 예측을 수행
AF4 모드에 따라 움직임 벡터 예측을 수행
AF6 모드에 따라 움직임 벡터 예측을 수행
AF4 모드 또는 AF6 모드에 따라 움직임 벡터 예측을 수행
종료

도면31


시작
현재 블록에 대해 AF 모드가 적용되는지 여부를 확인
S3110
현재 블록에 대해 상기 AF 모드가 적용되는 경우, 이웃 블록이 AF 모드로 코딩되었는지 여부를 확인
S3120
이웃 블록이 AF 모드로 코딩된 경우, AF4 flag 또는 AF6 flag 중 적어도 하나를 획득
S3130
AF4 flag 또는 AF6 flag 중 적어도 하나에 기초하여 4개 또는 6개 파라미터를 이용하여 움직임 벡터 예측자를 생성
S3140
움직임 벡터 예측자에 기초하여 상기 현재 블록의 움직임 벡터를 획득
S3150
종료

도면32

(S3210)	Slice ... affine_4_flag affine_6_flag

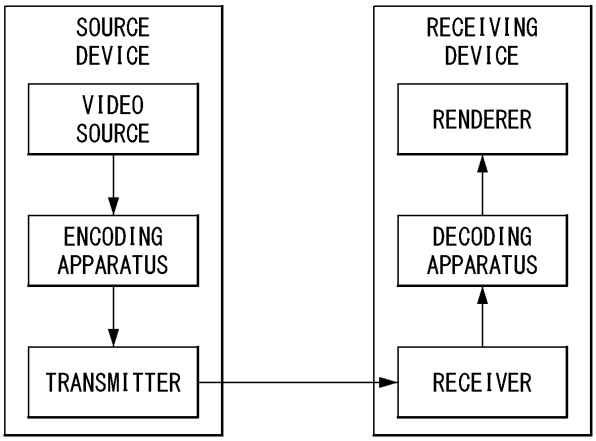
	parse merge_flag
	if (merge_flag) {
	...
	}
	else { // inter
	if (affine_4_flag != 0 || affine_6_flag != 0) { //
	NOT (affine_4_flag == 0 && affine_6_flag == 0)
	(S3220) parse affine_flag
	if (affine_flag) {
	(S3230) if (affine_4_flag == 1 && affine_6_flag == 0)
	set affine_param_flag as 0 // AF4
	(S3240) else if (affine_4_flag == 0 && affine_6_flag == 1)
	set affine_param_flag as 1 // AF6
	(S3250) else if (affine_4_flag == 1 && affine_6_flag == 1)
	parse affine_param_flag
	if (parse affine_param_flag == 0) { // AF4
	parse mvd for CP0
	parse mvd for CP1
	}
	else { // if (parse affine_param_flag == 1) // AF6
	parse mvd for CP0
	parse mvd for CP1
	parse mvd for CP2
	}
	}
	}
	}

도면33

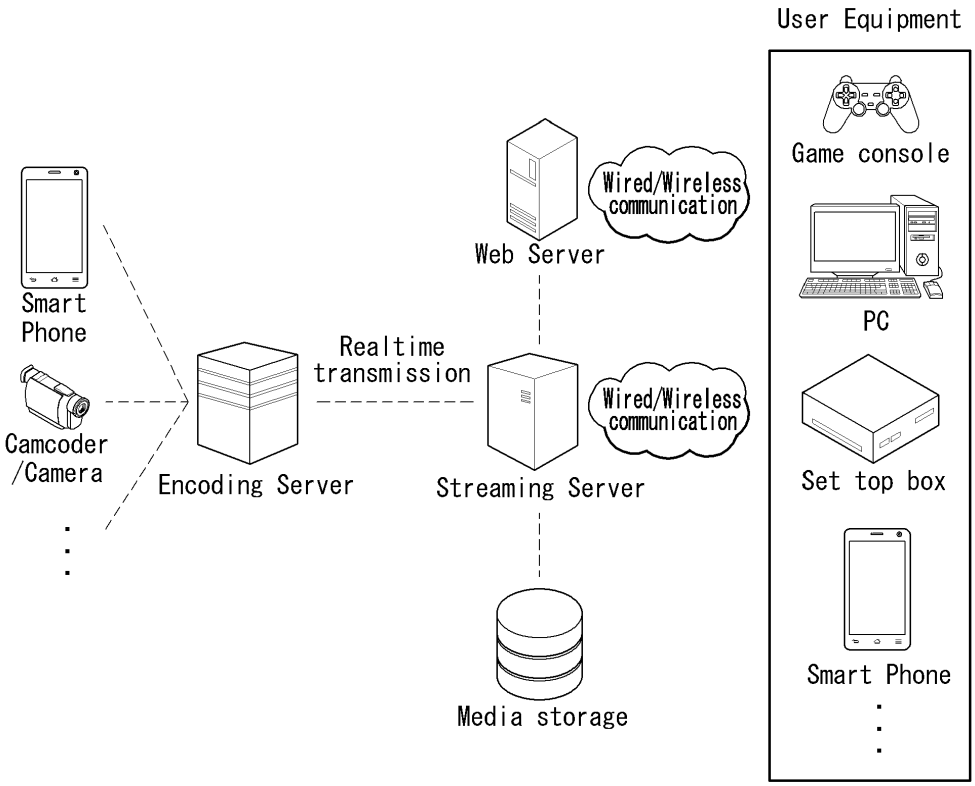
	Slice affine_4_flag affine_6_flag

	parse merge_flag
	if (merge_flag) {
	...
	}
	else { // inter
	if (affine_4_flag != 0 || affine_6_flag != 0) { //
	NOT (affine_4_flag == 0 && affine_6_flag == 0)
	parse affine_flag
	if (affine_flag) {
	if (affine_4_flag == 1 && affine_6_flag == 0)
	set affine_param_flag as 0 // AF4
	else if (affine_4_flag == 0 && affine_6_flag == 1)
	set affine_param_flag as 1 // AF6
	else if (affine_4_flag == 1 && affine_6_flag == 1)
	if (isNeighborAffine())
	(S3310) parse affine_param_flag
	else
	(S3320) set affine_param_flag as 0 // AF4
	if (parse affine_param_flag == 0) { // AF4
	parse mvd for CP0
	parse mvd for CP1
	}
	else { // if (parse affine_param_flag == 1) // AF6
	parse mvd for CP0
	parse mvd for CP1
	parse mvd for CP2
	}
	}
	}
	}

도면34


SOURCE DEVICE
VIDEO SOURCE
ENCODING APPARATUS
TRANSMITTER
RECEIVING DEVICE
RENDERER
DECODING APPARATUS
RECEIVER

도면35


User Equipment
Game console
PC
Set top box
Smart Phone
Wired/Wireless communication
Web Server
Smart Phone
Camcoder /Camera
Encoding Server
Realtime transmission
Streaming Server
Wired/Wireless communication
Media storage