



(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2013년04월12일
(11) 등록번호 10-1254355
(24) 등록일자 2013년04월08일

(51) 국제특허분류(Int. Cl.)

H04N 7/26 (2006.01)

(21) 출원번호 10-2007-7023526

(22) 출원일자(국제) 2006년03월16일

심사청구일자 2011년01월13일

(85) 번역문제출일자 2007년10월12일

(65) 공개번호 10-2008-0006555

(43) 공개일자 2008년01월16일

(86) 국제출원번호 PCT/US2006/009587

(87) 국제공개번호 WO 2006/113003

국제공개일자 2006년10월26일

(30) 우선권주장

60/671,255 2005년04월13일 미국(US)

60/700,834 2005년07월20일 미국(US)

(56) 선행기술조사문헌

Woo-Shik Kim, et al., 'Proposal for the unsolved issues in Professional Extensions I', Joint Video Team (JVT) of ISO/IEC MPEG & ITU-T VCEG (ISO/IEC JTC1/SC29/WG11 and ITU-T SG16 Q.6) 10thJVT Meeting*

KR1020060109844 A

*는 심사관에 의하여 인용된 문헌

(73) 특허권자

툼슨 라이선싱

프랑스 92130 이씨레플리노 잔 다르크 뢰 1-5

(72) 발명자

유, 하오평

미국 46032 인디애나주 카멜 글렌 아베 코트 3471

(74) 대리인

백만기, 전경석, 양영준

전체 청구항 수 : 총 12 항

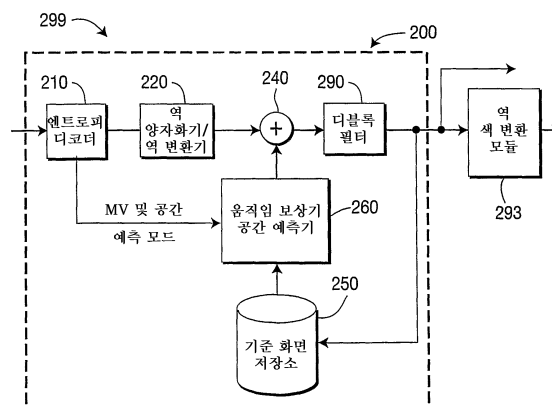
심사관 : 조우연

(54) 발명의 명칭 **공용 예측기를 사용하는 루마 및 크로마 디코딩**

(57) 요약

비디오 인코더, 비디오 디코더 및 해당 방법을 제공한다. 영상 블록에 대한 비디오 신호 데이터를 인코딩하기 위한 비디오 인코더는 공용 예측기를 사용하여 비디오 신호 데이터의 모든 색 성분을 인코딩(315)하기 위한 인코더(100)를 포함한다. 영상 블록에 대한 비디오 신호 데이터를 디코딩하기 위한 비디오 디코더는 공용 예측기를 사용하여 비디오 신호 데이터의 모든 색 성분을 디코딩(430)하기 위한 디코더(200)를 포함한다. 추가로, 영상 블록에 대한 신호 데이터를 인코딩 및 디코딩하기 위한 장치 및 방법은 잔여 색 변환을 적용하지 않으면서 비디오 신호 데이터의 색 성분을 인코딩/디코딩하기 위한 인코더 및 디코더를 포함한다. 또한, 영상 블록에 대한 비디오 신호 데이터를 인코딩/디코딩하기 위한 비디오 인코더 및 디코더는 비디오 신호 데이터의 각 색 성분에 대하여 특유한 예측기를 사용하여 비디오 신호 데이터를 인코딩/디코딩하기 위한 인코더 및 디코더를 포함한다.

대표도 - 도2



특허청구의 범위

청구항 1

영상 블록을 위한 비디오 신호 데이터를 디코딩하는 비디오 디코더로서,

공용 예측기(common predictor)를 사용하여 상기 비디오 신호 데이터의 모든 색 성분들을 디코딩하기 위한 디코더(200)를 포함하고,

상기 디코더(200)는 상기 비디오 신호 데이터의 색 성분들 모두에 대하여 B 및 P 프레임들을 위한 공용 보간 필터들을 사용하고, 디코딩 단계에 후속하는 사후 처리 단계에서 상기 비디오 신호 데이터에 대한 역 색 변환이 수행되는

비디오 디코더.

청구항 2

제1항에 있어서,

상기 공용 예측기는 상기 비디오 신호 데이터의 루마 및 크로마 성분들 둘 모두에 대하여 사용되는 루마 예측기(luma predictor)인 비디오 디코더.

청구항 3

제1항에 있어서,

상기 디코더(200)는 상기 비디오 신호 데이터의 색 성분들 모두에 대하여 공용 공간 예측 모드를 이용하는 비디오 디코더.

청구항 4

제3항에 있어서,

상기 공용 공간 예측 모드는 국제전기통신연합 통신 부문 H.264 표준의 prev_intra8×8_pred_mode_flag, rem_intra8×8_pred_mode, prev_intra4×4_pred_mode_flag 및 rem_intra4×4_pred_mode 파라미터들에 의해 설정되는 비디오 디코더.

청구항 5

삭제

청구항 6

제1항에 있어서,

상기 디코딩시에, 잔여 색 변환(residual color transform)이 생략되는 비디오 디코더.

청구항 7

제1항에 있어서,

상기 비디오 신호 데이터의 샘플링은 국제전기통신연합 통신 부문 H.264 표준의 4:4:4, 4:2:2 및 4:2:0 포맷들 중 어느 하나에 대응하는 비디오 디코더.

청구항 8

영상 블록을 위한 비디오 신호 데이터를 디코딩하는 방법으로서,

공용 예측기를 사용하여 상기 비디오 신호 데이터의 모든 색 성분들을 디코딩하는 단계(430)를 포함하고,

상기 비디오 신호 데이터의 색 성분들 모두에 대하여 B 및 P 프레임들을 위해 공용 보간 필터들이 사용되고, 상기 디코딩 단계에 후속하는 사후 처리 단계에서 상기 비디오 신호 데이터에 대한 역 색 변환이 수행되는

디코딩 방법.

청구항 9

제8항에 있어서,

상기 공용 예측기는 상기 비디오 신호 데이터의 루마 및 크로마 성분들 둘 모두에 대하여 사용되는 루마 예측기인 디코딩 방법.

청구항 10

제8항에 있어서,

상기 비디오 신호 데이터의 색 성분들 모두에 대하여 공용 공간 예측 모드를 이용하는 디코딩 방법.

청구항 11

제10항에 있어서,

상기 공용 공간 예측 모드는 국제전기통신연합 통신 부문 H.264 표준의 `prev_intra8×8_pred_mode_flag`, `rem_intra8×8_pred_mode`, `prev_intra4×4_pred_mode_flag` 및 `rem_intra4×4_pred_mode` 파라미터들에 의해 설정되는 디코딩 방법.

청구항 12

삭제

청구항 13

제8항에 있어서,

상기 디코딩 단계는 상기 비디오 신호 데이터의 색 성분들 모두를 디코딩하고, 잔여 색 변환이 생략되는 디코딩 방법.

청구항 14

삭제

청구항 15

제8항에 있어서,

상기 비디오 신호 데이터의 샘플링은 국제전기통신연합 통신 부문 H.264 표준의 4:4:4, 4:2:2 및 4:2:0 포맷들 중 어느 하나에 대응하는 디코딩 방법.

명세서

기술분야

[0001] 관련 출원의 상호 참조

[0002] 이 출원은 2005년 4월 13일에 출원한 미국 특허 가출원번호 제60/671,255호 및 2005년 7월 20일에 출원한 미국 특허 가출원번호 제60/700,834호에 대한 우선권을 주장하고, 각 내용은 본 명세서에 참조로서 포함한다. 또한, 이 출원은 이 출원과 동시에 출원한 "METHOD AND APPARATUS FOR VIDEO ENCODING"이라는 명칭의 미국 특허출원(대리인 정리번호 PU050051, PU060023), "METHOD AND APPARATUS FOR VIDEO DECODING"이라는 명칭의 미국 특허출원(대리인 정리번호 PU060030) 및 "METHOD AND APPARATUS FOR VIDEO ENCODING AND DECODING"이라는 명칭의 미국 특허출원(대리인 정리번호 PU050159)과 관련이 있다.

[0003] 본 발명은 일반적으로 비디오 인코더 및 디코더에 관한 것으로서, 특히 비디오 인코딩 및 디코딩 방법 및 장치에 관한 것이다.

배경기술

- [0004] 현재, 국제전기통신연합 통신 부문(ITU-T) H.264 표준(이하, "H.264 표준")의 4:4:4 포맷은 3개 채널 중 하나만을 루마(luma)로서 코딩하고, 다른 2개 채널은 저효율의 툴을 이용하여 크로마(chroma)로서 코딩한다. 코덱에 대한 입력이 모든 입력 성분에서 완전한 해상도를 갖는 4:4:4 포맷인 경우, 저효율의 크로마 코딩 알고리즘으로 3개 입력 성분 중 2개를 코딩하면 그러한 2개 채널에 더 많은 비트를 이용하게 된다. 이 특별한 문제점은 인트라 프레임에서 더욱 두드러진다. 예를 들어, Intra-Only 모드에 실행하는 H.264 표준은 40dB (PSNR) 이상에서 전체 압축 품질에 대하여 JPEG2k보다 비효율적이다.
- [0005] 따라서, 상술한 종래기술의 단점을 극복하는 비디오 인코딩 및 디코딩 방법 및 장치는 바람직하고 매우 유용하다.
- [0006] <발명의 개요>
- [0007] 종래기술의 이러한 약점과 단점 및 다른 약점과 단점은 비디오 인코딩 및 디코딩 방법 및 장치에 관한 본 발명이 처리한다.
- [0008] 본 발명의 한 양상에 따르면, 영상 블록에 대한 비디오 신호 데이터를 인코딩하기 위한 비디오 인코더를 제공한다. 비디오 인코더는 공용 예측기를 사용하여 비디오 신호 데이터의 모든 색 성분을 인코딩하기 위한 인코더를 포함한다.
- [0009] 본 발명의 다른 양상에 따르면, 영상 블록에 대한 비디오 신호 데이터를 인코딩하기 위한 방법을 제공한다. 본 방법은 공용 예측기를 사용하여 비디오 신호 데이터의 모든 색 성분을 인코딩하는 것을 포함한다.
- [0010] 본 발명의 또 다른 양상에 따르면, 영상 블록에 대한 비디오 신호 데이터를 디코딩하기 위한 비디오 디코더를 제공한다. 비디오 디코더는 공용 예측기를 사용하여 비디오 신호 데이터의 모든 색 성분을 디코딩하기 위한 디코더를 포함한다.
- [0011] 본 발명의 또 다른 양상에 따르면, 영상 블록에 대한 비디오 신호 데이터를 디코딩하기 위한 방법을 제공한다. 본 방법은 공용 예측기를 사용하여 비디오 신호 데이터의 모든 색 성분을 디코딩하는 것을 포함한다.
- [0012] 본 발명의 이러한 양상, 특징과 장점 및 다른 양상, 특징과 장점은 첨부한 도면과 함께 설명하는 예시적인 실시예에 대한 이하의 상세한 설명으로부터 명확해진다.
- 실시예**
- [0030] 본 발명은 비디오 신호 데이터를 인코딩 및 디코딩하기 위한 방법 및 장치에 관한 것이다. 우선, 본 발명은 국제전기통신연합 통신 부문(ITU-T) H.264 표준의 4:4:4 포맷을 이용하여 샘플링한 비디오 신호 데이터에 관하여 설명하지만, 본 발명의 범위를 유지하면서 H.264 표준의 다른 포맷(예를 들어, 4:2:2 및/또는 4:2:0 포맷)뿐만 아니라 다른 비디오 압축 표준을 이용하여 샘플링한 비디오 신호 데이터에 적용할 수도 있다는 점을 인식하게 된다.
- [0031] 본 원리에 따른 방법 및 장치는 루마 또는 크로마 압축 알고리즘을 위하여 새로운 툴을 이용할 필요가 없다는 점을 인식하게 된다. 대신, 기존의 루마 코딩 툴은 이용할 수 있다. 따라서, 4:4:4 포맷의 코딩 성능을 최대화할 수도 있으면서 후방 호환성을 유지하고, 기존의 H.264 (또는 다른 적용가능한) 표준에 대한 변경을 최소화할 수도 있다는 장점이 있다.
- [0032] 실시예에서 구성한 본 발명의 원리에 따르면, 루마 코딩 알고리즘은 예를 들어 4:4:4 콘텐츠의 3개 성분 채널 모두를 코딩하는 데 이용한다. 이 실시예의 장점은 종래기술과 관련하여 4:4:4 콘텐츠를 압축하기 위한 전체 코딩 성능의 향상에 있다. 현재, 기존의 H.264 표준에서는 3개 채널 중 하나만을 루마로서 코딩하고, 다른 2개는 저효율 툴을 이용하여 크로마로서 코딩한다.
- [0033] 또한, 실시예에서 구성한 본 발명의 원리에 따르면, 색 변환은 사전 처리 공정으로서 수행한다. 따라서, 이 실시예에 따르면, 잔여 색 변환(RCT: Residual Color Transform)은 압축 루프 내에서 수행하지 않는다. 이 실시예의 장점은 일관된 인코더/디코더 구조를 모든 색 포맷에 제공하는 데 있다.
- [0034] 게다가, 실시예에서 구성한 본 발명의 원리에 따르면, 3개 성분 모두에 대하여 동일한 움직임/공간 예측 모드를 이용한다. 이 실시예의 장점은 감소한 코덱 복잡도 및 후방 호환성에 있다.
- [0035] 또한, 다른 실시예에 따르면, 3개 성분 모두에 대하여 동일한 예측기를 사용하는 대신, 3개 성분에 대하여 3개의 제한 공간 예측기의 세트(또는 서브세트)를 사용할 수도 있다. 이 실시예의 장점은 종래기술과 관련하여

4:4:4 콘텐츠를 압축하기 위한 전체 코딩 성능의 향상에 있다.

- [0036] 상술하고, 본 명세서에서 후술하는 다양한 실시예는 독립적인 실시예로서 구현할 수도 있거나, 본 기술분야의 당업자가 이미 알고 있는 방식으로 조합할 수도 있다는 점을 인식하게 된다. 따라서, 예를 들어 제1 조합 실시예에서 루마 코딩 알고리즘은 3개 성분 채널 모두를 코딩하는 데 유용하게 이용하고, 색 변환은 사전 처리 공정으로서 수행하며, 3개 성분 채널 모두에 대하여 단일 예측기를 사용한다. 제2 조합 실시예에서 루마 코딩 알고리즘은 3개 성분 채널 모두를 코딩하는 데 유용하게 이용하고, 색 변환은 사전 처리 공정으로서 수행하며, 3개 성분 채널에 대하여 3개의 제한 공간 예측기의 세트(또는 서브세트)를 이용할 수도 있다. 물론, 상술한 바와 같이, 본 발명의 범위를 유지하면서, 본 명세서에서 제공하는 본 원리의 내용에 따라 다양한 실시예의 다른 조합을 구현할 수도 있다.
- [0037] 본 설명은 본 발명의 원리를 나타낸다. 따라서, 본 기술분야의 숙련자는 본 명세서에서 명확하게 기술하거나 도시하지 않지만 본 발명의 원리를 실시하고, 본 발명의 사상 및 범위 내에 존재하는 다양한 장치를 고안할 수 있다는 점을 인식하게 된다.
- [0038] 본 명세서에 기술하는 모든 예와 조건 언어는 발명자가 본 기술분야를 증진시키는 데 기여한 본 발명의 원리 및 개념을 독자가 이해하도록 돕는 데 그 목적이 있고, 그와 같이 구체적으로 기술한 예 및 조건에 한정하는 것으로 해석해서는 안 된다.
- [0039] 또한, 본 발명의 원리, 양상 및 실시예뿐만 아니라 특정 예를 기술하는 모든 문장은 본 발명의 구조적이고 기능적인 등가물을 포함하는 의도가 있다. 추가로, 그와 같은 등가물은 현재 알려진 등가물뿐만 아니라 앞으로 개발될 등가물, 즉 구조에 상관없이 동일한 기능을 수행하도록 개발하는 구성요소를 포함하는 의도가 있다.
- [0040] 따라서, 예를 들어 본 기술분야의 숙련자는 본 명세서에서 제공한 블록도는 본 발명의 원리를 구현하는 예시적인 회로의 개념도를 나타낸다는 점을 인식하게 된다. 유사하게, 플로우차트, 흐름도, 상태 천이도, 의사 코드 등은 실질적으로 컴퓨터 판독가능한 매체에서 재생할 수도 있고, 컴퓨터 또는 프로세서를 명확하게 도시하지 않더라도 컴퓨터 또는 프로세서가 실행할 수도 있는 다양한 프로세스를 나타낸다는 점을 인식하게 된다.
- [0041] 도면에 도시한 다양한 구성요소의 기능은 전용 하드웨어뿐만 아니라 적합한 소프트웨어와 관련된 소프트웨어를 실행할 수 있는 하드웨어를 사용하여 제공할 수도 있다. 프로세서가 제공하는 경우, 그 기능은 단일 전용 프로세서, 단일 공유 프로세서 또는 일부를 공유할 수도 있는 복수의 개별 프로세서가 제공할 수도 있다. 또한, "프로세서" 또는 "컨트롤러"라는 용어는 소프트웨어를 실행할 수 있는 하드웨어만을 의미하는 것으로 해석해서는 안 되고, 디지털 신호 프로세서(DSP) 하드웨어, 소프트웨어를 저장하기 위한 ROM, RAM 및 비휘발성 저장 장치를 포함할 수도 있지만, 이에 한정하지 않는다.
- [0042] 종래의 그리고/또는 주문형의 다른 하드웨어를 포함할 수도 있다. 유사하게, 도면에 도시한 스위치는 개념적인 것일 뿐이다. 그 기능은 프로그램 로직의 연산, 전용 로직, 프로그램 제어 및 전용 로직의 상호 작용 또는 수동으로 실행할 수도 있는데, 컨텍스트로부터 구체적으로 이해한 바와 같이 구현자는 특정 기법을 선택할 수 있다.
- [0043] 본 발명의 청구범위에서 특정 기능을 수행하기 위한 수단으로서 표현한 구성요소는 그 기능을 수행하는 어떤 방식, 예를 들어 a) 그 기능을 수행하는 회로 소자의 조합 또는 b) 그 기능을 수행하는 소프트웨어를 실행하기 위한 적합한 회로와 조합하는 펌웨어, 마이크로코드 등을 포함하는 형태의 소프트웨어를 포함하는 의도가 있다. 그와 같은 청구범위로 정의한 본 발명에는 다양한 인용 수단이 제공하는 기능성은 청구범위에서 요구하는 방식으로 조합된다는 점이 존재한다. 따라서, 그러한 기능성을 제공할 수 있는 수단은 본 명세서에 도시한 수단과 동일하다고 간주한다.
- [0044] 도 1을 참조하면, 예시적인 비디오 인코딩 장치는 일반적으로 참조번호 199로 나타낸다. 비디오 인코딩 장치(199)는 비디오 인코더(100) 및 사전 인코딩 색 변환 모듈(105)을 포함한다.
- [0045] 사전 인코딩 색 변환 모듈(105)은 비디오 신호를 비디오 인코더(100)에 입력하기 전에 비디오 신호의 색 사전 처리를 수행한다. 사전 인코딩 색 변환 모듈(105)이 수행하는 색 사전 처리는 후술한다. 사전 인코딩 색 변환 모듈(105)은 몇몇 실시예에서는 생략할 수도 있다는 점을 인식하게 된다.
- [0046] 사전 인코딩 색 변환 모듈(105)의 입력 및 비디오 인코더(100)의 입력은 비디오 인코딩 장치(199)의 입력으로서 이용가능하다.

- [0047] 사전 인코딩 색 변환 모듈(105)의 출력부는 비디오 인코더(100)의 입력부와 신호 통신으로 연결한다.
- [0048] 비디오 인코더(100)의 입력부는 가산기(110)의 비반전 입력부와 신호 통신으로 연결한다. 가산기(110)의 출력부는 변환기/양자화기(120)와 신호 통신으로 연결한다. 변환기/양자화기(120)의 출력부는 엔트로피 코더(140)와 신호 통신으로 연결한다. 엔트로피 코더(140)의 출력은 비디오 인코더(100)의 출력 및 비디오 인코딩 장치(199)의 출력으로서 이용가능하다.
- [0049] 또한, 변환기/양자화기(120)의 출력부는 역 변환기/역 양자화기(150)와 신호 통신으로 연결한다. 역 변환기/역 양자화기(150)의 출력부는 디블록 필터(160)의 입력부와 신호 통신으로 연결한다. 디블록 필터(160)의 출력부는 기준 화면 저장소(170)와 신호 통신으로 연결한다. 기준 화면 저장소(170)의 제1 출력부는 움직임 및 공간 예측 추정기(180)의 제1 입력부와 신호 통신으로 연결한다. 비디오 인코더(100)의 입력부는 움직임 및 공간 예측 추정기(180)의 제2 입력부와 신호 통신으로 또한 연결한다. 움직임 및 공간 예측 추정기(180)의 출력부는 움직임 및 공간 예측 보상기(190)의 제1 입력부와 신호 통신으로 연결한다. 기준 화면 저장소(170)의 제2 출력부는 움직임 및 공간 보상기(190)의 제2 입력부와 신호 통신으로 연결한다. 움직임 및 공간 보상기(190)의 출력부는 가산기(110)의 반전 입력부와 신호 통신으로 연결한다.
- [0050] 도 2를 참조하면, 예시적인 비디오 디코딩 장치는 일반적으로 참조번호 299로 나타낸다. 비디오 디코딩 장치(299)는 비디오 디코더(200) 및 사후 디코더 역 색 변환 모듈(293)을 포함한다.
- [0051] 비디오 디코더(200)의 입력은 비디오 디코딩 장치(299)의 입력으로서 이용가능하다. 비디오 디코더(200)의 입력부는 엔트로피 디코더(210)의 입력부와 신호 통신으로 연결한다. 엔트로피 디코더(210)의 제1 출력부는 역 양자화기/역 변환기(220)의 입력부와 신호 통신으로 연결한다. 역 양자화기/역 변환기(220)의 출력부는 가산기(240)의 제1 입력부와 신호 통신으로 연결한다.
- [0052] 가산기(240)의 출력부는 디블록 필터(290)와 신호 통신으로 연결한다. 디블록 필터(290)의 출력부는 기준 화면 저장소(250)와 신호 통신으로 연결한다. 기준 화면 저장소(250)는 움직임 및 공간 예측 보상기(260)의 제1 입력부와 신호 통신으로 연결한다. 움직임 및 공간 예측 보상기(260)의 출력부는 가산기(240)의 제2 입력부와 신호 통신으로 연결한다. 엔트로피 디코더(210)의 제2 출력부는 움직임 보상기(260)의 제2 입력부와 신호 통신으로 연결한다. 디블록 필터(290)의 출력은 비디오 디코더(200)의 출력 및 비디오 디코딩 장치(299)의 출력으로서 이용가능하다.
- [0053] 또한, 사후 디코딩 역 색 변환 모듈(293)의 출력은 비디오 디코딩 장치(299)의 출력으로서 이용가능할 수도 있다. 그와 같은 경우, 비디오 디코더(200)의 출력부는 비디오 디코더(200)에 관한 사후 처리 모듈인 사후 디코딩 역 색 변환 모듈(293)의 입력부와 신호 통신으로 연결할 수도 있다. 사후 디코딩 역 색 변환 모듈(293)의 출력부는 비디오 디코더(200)의 출력에 대하여 사후 처리한 역 색 변환된 신호를 제공한다. 사후 디코딩 역 색 변환 모듈(293)의 사용은 옵션이라는 점을 인식하게 된다.
- [0054] 이제, 본 발명의 원리에 따른 강화된 4:4:4 코딩을 설명한다. 제1 실시예는 모든 색 성분에 대하여 루마 코딩 알고리즘을 이용하고, 모든 색 성분에 대하여 동일한 공간 예측 모드를 이용하며, 잔여 색 변환(RCT)을 압축 루프 내에서 생략한 조합 실시예이다. 이 조합 실시예에 대한 테스트 결과를 또한 제공한다. 그 후, 모든 색 성분에 대하여 루마 코딩 알고리즘을 이용하고, 모든 색 성분에 대하여 제한 공간 예측기의 세트(또는 서브세트)를 이용하며(단일 공간 예측 모드 대신), 잔여 색 변환(RCT)을 압축 루프 내에서 생략한 제2 조합 실시예를 설명한다. 따라서, 제1 및 제2 조합 실시예의 차이는 제1 조합 실시예에서는 모든 색 성분에 대하여 단일 공간 예측 모드를 이용하고, 제2 조합 실시예에서는 모든 색 성분에 대하여 제한 공간 예측기의 세트(또는 서브세트)를 이용한다는 데 있다. 물론, 상술한 바와 같이, 본 명세서에 기술하는 실시예는 독립적인 실시예로서 구현할 수도 있거나, 본 기술분야의 당업자가 쉽게 인식하는 방식으로 조합할 수도 있다. 예를 들어, 실시예에서 구성한 본 발명의 원리에 따르면, 압축 루프로부터 RCT를 생략하는 다른 실시예와 조합하지 않고 단일 공간 예측 모드만을 이용한다. 본 명세서에 제공하는 본 원리의 내용에 따라 본 발명의 범위를 유지하면서 본 기술분야의 당업자는 본 발명의 실시예에 대한 이와 같은 변형, 구현 및 조합을 용이하게 규명할 수 있다는 점을 인식하게 된다.
- [0055] 도 3을 참조하면, 사전 인코딩 색 변환 블록이 있는 예시적인 비디오 인코딩 프로세스는 일반적으로 참조번호 300 및 301로 각각 나타낸다.
- [0056] 사전 인코딩 색 변환 블록(301)은 블록 306, 308 및 310을 포함한다는 점을 인식하게 된다. 또한, 사전 인코딩 색 변환 블록(301)은 옵션이고, 따라서 본 발명의 몇몇 실시예에서는 생략할 수도 있다는 점을 인식하게 된다.

- [0057] 사전 인코딩 색 변환 블록(301)은 영상의 각 블록에 대한 루프를 시작하고, 기능 블록(308)으로 제어를 전달하는 루프 제한 블록(306)을 포함한다. 기능 블록(308)은 현재 영상 블록의 비디오 신호 데이터의 색 사전 처리를 수행하고, 루프 제한 블록(310)으로 제어를 전달한다. 루프 제한 블록(310)은 루프를 종료한다. 또한, 루프 제한 블록(310)은 비디오 인코딩 프로세스(300)에 포함되는 루프 제한 블록(312)으로 제어를 전달한다.
- [0058] 루프 제한 블록(312)은 영상의 각 블록에 대한 루프를 시작하고, 기능 블록(315)으로 제어를 전달한다. 기능 블록(315)은 현재 영상 블록의 각 색 성분에 대하여 공용 예측기를 이용하여 현재 영상 블록의 움직임 보상 또는 공간 예측을 형성하고, 기능 블록(320)으로 제어를 전달한다. 기능 블록(320)은 현재 영상 블록으로부터 움직임 보상 또는 공간 예측을 감산하여 예측 잔여를 형성하고, 기능 블록(330)으로 제어를 전달한다. 기능 블록(330)은 예측 잔여를 변환 및 양자화하고, 기능 블록(335)으로 제어를 전달한다. 기능 블록(335)은 예측 잔여를 역변환 및 역 양자화하여 코딩된 예측 잔여를 형성하고, 기능 블록(345)으로 제어를 전달한다. 기능 블록(345)은 코딩된 잔여를 예측에 가산하여 코딩된 화면 블록을 형성하고, 종료 루프 블록(350)으로 제어를 전달한다. 종료 루프 블록(350)은 루프를 종료하고, 종료 블록(355)으로 제어를 전달한다.
- [0059] 도 4를 참조하면, 사후 디코딩 역 색 변환 블록이 있는 예시적인 비디오 디코딩 프로세스는 일반적으로 참조번호 400 및 460으로 각각 나타낸다.
- [0060] 사후 디코딩 역 색 변환 블록(460)은 블록 462, 464, 466 및 468을 포함한다는 점을 인식하게 된다. 또한, 사후 디코딩 역 색 변환 블록(460)은 옵션이고, 따라서 본 발명의 몇몇 실시예에서는 생략할 수도 있다는 점을 인식하게 된다.
- [0061] 디코딩 프로세스(400)는 영상의 현재 블록에 대한 루프를 시작하고, 기능 블록(415)으로 제어를 전달하는 루프 제한 블록(410)을 포함한다. 기능 블록(415)은 코딩된 잔여를 엔트로피 디코딩하고, 기능 블록(420)으로 제어를 전달한다. 기능 블록(420)은 디코딩된 잔여를 역변환 및 역 양자화하여 코딩된 잔여를 형성하고, 기능 블록(430)으로 제어를 전달한다. 기능 블록(430)은 각 색 성분에 대하여 공용 예측기로부터 형성한 예측에 코딩된 잔여를 가산하여 코딩된 화면 블록을 형성하고, 루프 제한 블록(435)으로 제어를 전달한다. 루프 제한 블록(435)은 루프를 종료하고, 종료 블록(440)으로 제어를 전달한다.
- [0062] 몇몇 실시예에서 루프 제한 블록(435)은 사후 디코딩 역 색 변환 블록(460), 특히 사후 디코딩 역 색 변환 블록(460)에 포함되는 루프 제한 블록(462)으로 제어를 선택적으로 전달한다. 루프 제한 블록(462)은 영상의 각 블록에 대한 루프를 시작하고, 기능 블록(464)으로 제어를 전달한다. 기능 블록(464)은 현재 영상 블록의 비디오 신호 데이터의 역 색 사후 처리를 수행하고, 루프 제한 블록(466)으로 제어를 전달한다. 루프 제한 블록(466)은 루프를 종료하고, 종료 블록(468)으로 제어를 전달한다.
- [0063] H.264 4:4:4 포맷에서 모든 성분 채널은 완전한 해상도를 갖는다. 따라서, 상술한 제1 조합 실시예에 따르면, 루마 코딩 알고리즘을 모든 색 성분에 대하여 이용하여 최대 전체 압축 효율을 얻는다. 따라서, 이 실시예에서, 인트라 프레임의 경우, 모든 색 성분은 예를 들어 Table 8-2, Table 8-3 and Table 8-4 in ISO/IEC 14496 10 Advanced Video Coding 3rd Edition(ITU-T Rec.H.264), ISO/IEC JTC1/SC29/WG11 and ITU-T SG16 Q.6, Document N6540, July 2004에서 열거한 예측 모드를 이용하여 압축할 수도 있다.
- [0064] 또한, 이 실시예에서는 3개 픽셀 성분 모두에 대하여 동일한 공간 예측 모드를 이용하여 코텍의 복잡도를 줄이고, 성능을 향상시킨다. 예를 들어, 매크로블록 예측 헤더에 있는 루마에 대한 prev_intra4×4_pred_mode_flag, rem_intra4×4_pred_mode, prev_intra8×8_pred_mode_flag 및 rem_intra8×8_pred_mode 파라미터에 의해 설정된 예측 모드가 3개 성분 모두에 의해 이용될 수도 있다. 그러므로 추가 비트 및 선택스 구성요소는 불필요하다. B 및 P (예측) 프레임에 대하여, 분수형(fractional) 픽셀 위치에 있는 기준 픽셀은 H.264 표준의 섹션 8.4.2.2.1에서 기술한 보간 방법으로 3개 채널 모두에 대하여 계산할 수도 있다. 현재의 H.264 표준에 대한 상세한 선택스 및 시맨틱 변경은 후술한다.
- [0065] 잔여 색 변환(RCT)은 High 4:4:4 프로파일로 인코더/디코더에 추가하였다. 그 결과, 4:4:4 포맷을 위한 압축 구조는 4:2:0 및 4:2:2 포맷을 위한 H.264 표준의 다른 프로파일 모두에서 현재 이용하는 압축 구조와 다르다. 이로 인해, 구현 복잡도가 추가된다. 또한, 다른 색 변환과 유사하게, YCOCG는 전체 압축 성능을 항상 향상시키는 것은 아니다. YCOCG의 효과는 콘텐츠에 매우 의존한다. 따라서, 전체 압축 및 강건성을 향상시키기 위하여, 이 실시예에서 색 변환은 사전 처리 블록의 일부분으로서 예측 루프의 외부에 배치한다. 이와 같이 함으로써, 특정 압축 작업을 위한 최적의 색 변환을 선택하는 것은 연산의 이슈가 되고, 특정 입력 시퀀스를 위한 최상의 응답은 수많은 옵션 중에서 발견할 수 있다. 3개 성분 모두가 인트라 프레임에 대하여 동일한 공간 예측

기를 사용하고, B 및 P (예측 또는 인터 코딩한) 프레임에 대하여 동일한 보간 필터를 사용하는 실시예에 따르면, 예측 잔여에 대하여 색 변환을 수행하는 것은 라운딩/잘라버림 에러가 무시되는 경우에는 코덱의 외부에서 소스 영상에 대한 색 변환을 수행하는 것과 동일하다. 이에 대해서는 후술한다. 따라서, RCT 블록을 코딩 구조로부터 제거하여 코딩 구조를 모든 색 포맷에 일관성 있게 한다.

[0066] 도 5를 참조하면, RCT의 간소화 모델은 일반적으로 참조번호 500으로 나타낸다. RCT 모델(500)은 기준 픽셀 발생기(510), 가산기(520) 및 선형 변환 모듈(530)을 포함한다. 기준 픽셀 발생기(510)의 입력부는 움직임/예지 정보 및 벡터 $[X_1], [X_2], \dots, [X_n]$ 를 수신하도록 구성한다. 기준 픽셀 발생기(510)의 출력부는 가산기(520)의 반전 입력부와 신호 통신으로 연결하고, 예측 벡터 $[X_p]$ 를 제공한다. 가산기(520)의 비-반전 입력부는 입력 벡터 $[X_{in}]$ 를 수신하도록 구성한다. 가산기(520)의 출력부는 선형 변환 모듈(530)의 입력부와 신호 통신으로 연결하고, 벡터 $[X_d]$ 를 제공한다. 선형 변환 모듈(530)의 출력부는 벡터 $[Y_d]$ 를 제공하도록 구성한다.

[0067] RCT(500)의 간소화 모델에서 3×3 행렬 $[A]$ 로 표현하는 색 변환(선형 변환)은 다음과 같이 정의한다.

수학식 1

$$\begin{bmatrix} Y \\ u \\ v \end{bmatrix} = [A] \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

[0068]

[0069] $[X_{in}], [X_d], [X_p], [X_1], [X_2], \dots, [X_n]$ 은 RGB 도메인 내 픽셀을 나타내는 3×1 벡터이다. $[Y_d]$ 는 색 변환의 결과를 나타내는 3×1 벡터이다. 그러므로 다음과 같다.

수학식 2

$$[Y_d] = [A][X_d] = [A][X_{in}] - [A][X_p]$$

[0070]

[0071] 이 실시예에서는 실시예에서 구성한 본 발명의 원리에 따라 매크로블록의 3개 성분 모두에 대하여 동일한 공간 예측기 및 보간 필터를 사용하므로, 기준 픽셀 $[X_p]$ 은 다음과 같이 표현할 수 있고,

수학식 3

$$[X_p] = [X_1 X_2 X_3 \dots X_n] \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ \vdots \\ c_n \end{bmatrix} = \begin{bmatrix} R_1 & R_2 & R_3 & \dots & R_n \\ G_1 & G_2 & G_3 & \dots & G_n \\ B_1 & B_2 & B_3 & \dots & B_n \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ \vdots \\ c_n \end{bmatrix}$$

[0072]

[0073] 여기서, $n \times 1$ 벡터 $[C]$ 는 H.264 표준에서 정의한 공간 예측기 및 보간 필터와 관련된 선형 연산을 나타낸다. 여기서, 기준 픽셀은 총 n개 인접 픽셀 $[X_1], [X_2], \dots, [X_n]$ 을 이용함으로써 계산한다고 가정한다.

[0074] 수학식 3의 $[X_p]$ 를 수학식 2에 대입하면 다음과 같다.

수학식 4

$$[Y_d] = [A][X_{in}] - [A] \left(\begin{bmatrix} R_1 & R_2 & R_3 & \dots & R_n \\ G_1 & G_2 & G_3 & \dots & G_n \\ B_1 & B_2 & B_3 & \dots & B_n \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ \vdots \\ c_n \end{bmatrix} \right)$$

[0075]

[0076] 라운딩/잘라버림 에러를 무시하고, 동일한 예측 모드는 RGB 도메인 또는 Y 도메인에서 선택된다고 가정하면, 다음과 같다.

수학식 5

$$[Y_d] = [A][X_{in}] - \begin{bmatrix} [A] \begin{bmatrix} R_1 & R_2 & R_3 & \dots & R_n \\ G_1 & G_2 & G_3 & \dots & G_n \\ B_1 & B_2 & B_3 & \dots & B_n \end{bmatrix} \\ C_1 \\ C_2 \\ C_3 \\ \vdots \\ C_n \end{bmatrix} = [Y_{in}] - \begin{bmatrix} Y_1 & Y_2 & Y_3 & \dots & Y_n \\ u_1 & u_2 & u_3 & \dots & u_n \\ v_1 & v_2 & v_3 & \dots & v_n \end{bmatrix} \begin{bmatrix} C_1 \\ C_2 \\ C_3 \\ \vdots \\ C_n \end{bmatrix}$$

[0077]

[0078] 그러므로 다음과 같다.

수학식 6

$$[Y_d] = [Y_{in}] - [Y_1 Y_2 Y_3 \dots Y_n] \begin{bmatrix} C_1 \\ C_2 \\ C_3 \\ \vdots \\ C_n \end{bmatrix}$$

[0079]

[0080] 따라서, 수학식 6은 YUV를 이 실시예에서 구성한 본 발명의 원리에 따라 인코더/디코더의 입력으로서 이용하는 것은 RCT를 수행하는 것과 동일하다는 점을 명확하게 보여준다.

[0081] 또한, 실시예에서 구성한 본 발명의 원리에 따르면, 본 명세서에서 "profile_idc=166이 있는 Advanced 4:4:4 프로파일"로 간주하는 새로운 4:4:4 프로파일이 H.264 표준에 추가된다. 이 새로운 profile_idc는 시퀀스 파라미터 헤더에 추가될 수도 있고, 매크로블록 계층 헤더뿐만 아니라 잔여 데이터 헤더에 이용될 수도 있다.

[0082] 루마 알고리즘을 이용하여 3개의 색 성분 모두를 코딩하는 것을 지원하기 위하여, 잔여 데이터 선택스가 일부 변경될 수도 있다. 또한, 매크로블록 헤더, 잔여 데이터 헤더 등에 있는 구성요소 중 일부의 시맨틱이 변경될 수도 있다. 일반적으로, H.264 사양 내 루마에 대한 기존의 선택스는 변경되지 않고, 3개 성분 중 하나를 코딩하는 데 이용된다. 변경은 후방 호환성이다. 상세한 선택스 및 시맨틱 변경은 이하에서 후술한다.

[0083] 이제, 다양한 실시예에서 구성한 본 발명의 원리에 따라 수행한 시뮬레이션 결과에 관하여 설명한다.

[0084] 도 6a 및 6b를 참조하면, ATV Intra-Only에 대한 평균 PSNR 대 비트율의 도면은 일반적으로 참조번호 600 및 650으로 각각 나타낸다.

[0085] 도 7a 및 7b를 참조하면, CT Intra-Only에 대한 평균 PSNR 대 비트율의 도면은 일반적으로 참조번호 700 및 750으로 각각 나타낸다.

[0086] 도 8a 및 8b를 참조하면, DT Intra-Only에 대한 평균 PSNR 대 비트율의 도면은 일반적으로 참조번호 800 및 850으로 각각 나타낸다.

[0087] 도 9a 및 9b를 참조하면, MIR_HD Intra-Only에 대한 평균 PSNR 대 비트율의 도면은 일반적으로 참조번호 900 및 950으로 각각 나타낸다.

[0088] 도 10a 및 10b를 참조하면, RT Intra-Only에 대한 평균 PSNR 대 비트율의 도면은 일반적으로 참조번호 1000 및 1050으로 각각 나타낸다.

[0089] 도 11a 및 11b를 참조하면, STB_HD Intra-Only에 대한 평균 PSNR 대 비트율의 도면은 일반적으로 참조번호 1100 및 1150으로 나타낸다.

[0090] 특히, 도 6a, 7a, 8a, 9a, 10a 및 11a는 ("새로운"이라는 용어로 전술한) 제안된 Advanced 4:4:4 프로파일에 대한 테스트 결과 대 대응하는 근사화 결과를 나타낸다. 또한, 도 6b, 7b, 8b, 9b, 10b 및 11b는 ("새로운"이라는 용어로 전술한) 제안된 Advanced 4:4:4 프로파일에 대한 테스트 결과 대 JPEK2k를 나타낸다.

[0091] 도 6a, 6b 내지 11a, 11b 모두에서 PSNR은 데시벨(dB)로 나타내고, 비트율은 bps(bits per second)로 나타낸다. ATV, CT, DT, MIR, RT, STB는 테스트 클립의 이름이다.

[0092] JVT-J042, Film-Originated Test Sequences에서 기술한 모든 JVT/FRExt 테스트 시퀀스를 테스트에 이용하였다. 테스트는 모두는 4:4:4 10비트 영화 매체이고, 각 클립은 58개 프레임을 갖는다.

- [0093] 제안된 Advanced 4:4:4 프로파일은 JVT 기준 소프트웨어 JM9.6에서 구현하였다. Intra-Only 및 IBBP 코딩 구조 둘 다를 테스트에 이용하였다. 양자화 파라미터는 각 R-D 곡선에 대하여 6, 12, 18, 24, 30 및 42로 설정하였다. RD-최적화 모드 선택을 이용하였다.
- [0094] 또한, 제안된 Advanced 4:4:4 프로파일은 YUVFormat=0(4:0:0)이 있는 기준 소프트웨어를 모든 각 입력 성분에 대하여 실행함으로써 얻은 결과와 비교하였다. 3개의 별도의 각 압축 비트 카운트는 압축 비트율을 계산하기 위한 총 압축 비트를 얻도록 함께 가산하였다.
- [0095] JPEG2k와 관련하여, KaKadu V2.2.3 소프트웨어를 테스트에 이용하였다. 테스트 결과는 9/7-탭 이중 직교 웨이브렛 필터로 웨이브렛 압축 해제에 5 레벨을 이용함으로써 얻었다. 프레임당 하나의 타일만이 존재하였고, 주어진 목표율을 위한 RD-최적화도 사용하였다.
- [0096] PSNR 측정 모두는 RGB 도메인에서 이루어졌다. $(PSNR(red)+PSNR(green)+PSNR(blue))/3$ 으로 정의한 평균 PSNR은 전체 압축 품질을 비교하는 데 이용한다. 이는, 주로 JPEG2k 압축 데이터는 소프트웨어가 제공하는 미지의 속도 제어 알고리즘을 이용하여 계산하기 때문이다. 몇몇의 경우, 특히 JPEG2k 색 변환을 이용하였을 경우에는 RGB PSNR 값은 서로 매우 멀리 떨어져 있다.
- [0097] 압축 비교는 다음과 같이 수행하였다.
- [0098] • New1: 단일 예측 모드가 있는 제안된 Advanced 4:4:4 프로파일.
- [0099] • New3: 3개의 예측 모드가 있는 제안된 Advanced 4:4:4 프로파일.
- [0100] • RCT-OFF: RCT=off가 있는 RGB 입력.
- [0101] • RCT-ON: RCT=on이 있는 RGB 입력.
- [0102] • YCOCG: RGB 대 YCOCG 변환은 코덱 외부에서 이루어졌다. 다음으로, 변환된 YCOCG는 JVT 소프트웨어에 대한 입력으로 이용하였다.
- [0103] • R+G+B: R, G 및 B 신호를 별도로 압축함으로써 근사화되는 제안된 방법.
- [0104] • Y+CO+CG: 변환된 Y, CO 및 CG 신호를 별도로 압축함으로써 근사화되는 제안된 방법.
- [0105] • J2k_RGB: JPEG2k 압축은 RGB 도메인에서 이루어졌다. JPEG2k 색 변환은 턴 오프되었다.
- [0106] • J2k_YUV: JPEG2k 압축은 YUV 도메인에서 이루어졌다. JPEG2k 색 변환을 이용하였다.
- [0107] 테스트 결과에 따르면, 실시예에서 구성한 본 발명의 원리에 따른 구현은 일반적으로 전체 압축 효율의 관점에서 JPEG2k와 매우 유사하다. 몇몇 경우에는 약간 더 양호하다.
- [0108] 또한, 실시예에서 구성한 본 발명의 원리에 따른 구현은 40dB (PSNR) 이상의 품질에 대하여 현재의 High 4:4:4 프로파일보다 상당히 높은 성능(압축)을 제공한다. 구체적으로, New1-YCOCG 또는 New3-YCOCG는 YCOCG 및 RCT-ON보다 양호하고, New1-RGB 또는 New3-RGB는 RCT-OFF보다 양호하다. 45dB (PSNR) 이상의 PSNR에서, 평균 PSNR의 평균적인 향상은 1.5dB보다 크다. 마지막 예에서, 이와 같은 향상은 45dB과 동일한 PSNR에서 25%보다 높은 비트 절감으로 해석할 수 있다.
- [0109] 테스트 결과에 따르면, 콘텐츠가 TP, RT와 같이 더욱 색 포화되는 경우, 색 변환은 코딩 성능에 도움을 준다. 즉, 색이 회색이고, 적게 포화되는 경우, RGB 도메인에서의 코딩이 올바른 선택일 수도 있다. 상술한 관찰은 어떤 색 변환을 이용하는지에 무관하다.
- [0110] New1-YCOCG 또는 New3-YCOCG 및 JPEG-2k_YUV의 결과를 비교하여, 코딩 효율을 향상시키는 관점에서 특정 색 변환의 성능은 매우 콘텐츠 의존적이라는 점을 알게 되었다. 단일 색 변환이 항상 최선인 것은 아니다. 그러므로 우리의 데이터는 인코딩(또는 디코딩) 루프 내에 RCT와 같은 색 변환이 있다는 점이 좋은 아이디어가 아닐 수도 있다는 점을 확인시킨다. 대신, 필요하다면, 인코더/디코더 외부에서 색 변환을 수행하여, 전체 압축 시스템이 더욱 양호하고 더욱 강한 성능을 제공하게 한다.
- [0111] YCOCG를 RCT-ON과 비교하면, 테스트 결과는 RCT로부터의 어떤 코딩 효율 향상도 나타내지 않는다. 또한, 턴 온된 RCT가 있는 기준 소프트웨어를 실행하는 것이 코딩 시간을 상당히 증가시켰다는 점을 알아야 한다. 실행 시간은 2.5배 이상 길었다.

- [0112] 이제, 실시예에서 구성한 본 발명의 원리에 따른 선택스 및 시맨틱 변화에 관하여 설명한다.
- [0113] 도 12를 참조하면, H.264 시퀀스 파라미터 선택스에 대한 테이블은 일반적으로 참조번호 1200으로 나타낸다. 실시예에서 구성한 본 발명의 원리에 따른 선택스에 대한 변화는 이탤릭체로 나타낸다.
- p>
- [0114] 도 13을 참조하면, H.264 잔여 데이터 선택스에 대한 테이블은 일반적으로 참조번호 1300으로 나타낸다. 실시예에서 구성한 본 발명의 원리에 따른 선택스에 대한 추가/변화는 이탤릭체로 나타낸다. 테이블(1300)에서, 잔여 데이터 헤더 내 루마 섹션은 어떤 필요한 텍스트 수정과 함께 2회 반복되어 luma1 및 luma2를 각각 지원한다.
- [0115] 전술한 바와 같이, 상술한 제1 조합 실시예는 본 원리를 JVT 기준 소프트웨어 JM9.6에 구현함으로써 평가 및 테스트하였다. New1-RGB 또는 New1-YCOCG로 표기한 테스트 결과는 제1 조합 실시예를 나타낸다.
- [0116] 전술한 바와 같이, 실시예에서 구성한 본 발명의 원리에 따르면, 단일 공간 예측 모드 대신, 3개의 제한 공간 예측기의 세트(또는 서브세트)를 성분 채널(예를 들어 RGB, YUV, YCrCb 포맷 등)에 대하여 이용한다. 또한, 전술한 바와 같이, 이 실시예는 본 명세서에 기술한 다른 실시예, 예를 들어 루마 코딩 알고리즘만을 이용하여 콘텐츠의 3개 성분 채널 모두를 코딩하고, 또는 색 변환을 사전 처리 공정으로서 이용하는 실시예와 조합할 수도 있다.
- [0117] 이제, 색 성분에 대하여 3개의 제한 공간 예측기의 세트(또는 서브세트)를 이용하고, 루마 코딩 알고리즘만을 이용하여 3개의 색 성분 모두를 코딩하며, 색 변환을 사전 처리 공정(즉, 압축 루프 내에 RCT 없음)으로서 이용하는 상술한 제2 조합 실시예에 관하여 설명한다. 이 실시예의 일부 변형도 설명한다.
- [0118] 도 14를 참조하면, 사전 인코딩 색 변환 공정이 있는 예시적인 비디오 인코딩 프로세스는 일반적으로 참조번호 1400 및 1401로 각각 나타낸다.
- [0119] 사전 인코딩 색 변환 블록(1401)은 블록 1406, 1408 및 1410을 포함한다는 점을 인식하게 된다. 또한, 사전 인코딩 색 변환 블록(1401)은 옵션이고, 따라서 본 발명의 몇몇 실시예에서는 생략할 수도 있다는 점을 인식하게 된다.
- [0120] 사전 인코딩 색 변환 블록(1401)은 영상의 각 블록에 대한 루프를 시작하고, 기능 블록(1408)으로 제어를 전달하는 루프 제한 블록(1406)을 포함한다. 기능 블록(1408)은 현재 영상 블록의 비디오 신호 데이터의 색 사전 처리를 수행하고, 루프 제한 블록(1410)으로 제어를 전달한다. 루프 제한 블록(1410)은 루프를 종료한다. 또한, 루프 제한 블록(1410)은 비디오 인코딩 프로세스(1400)에 포함되는 루프 제한 블록(1412)으로 제어를 전달한다.
- [0121] 루프 제한 블록(1412)은 영상의 각 블록에 대한 루프를 시작하고, 기능 블록(1415)으로 제어를 전달한다. 기능 블록(1415)은 현재 영상 블록의 각 색 성분에 대하여 공용 예측기를 사용하여 현재 영상 블록의 움직임 보상 또는 공간 예측을 형성하고, 기능 블록(1420)으로 제어를 전달한다. 기능 블록(1420)은 현재 영상 블록으로부터 움직임 보상 또는 공간 예측을 감산하여 예측 잔여를 형성하고, 기능 블록(1430)으로 제어를 전달한다. 기능 블록(1430)은 예측 잔여를 변환 및 양자화하고, 기능 블록(1435)으로 제어를 전달한다. 기능 블록(1435)은 예측 잔여를 역변환 및 역 양자화하여 코딩된 예측 잔여를 형성하고, 기능 블록(1445)으로 제어를 전달한다. 기능 블록(1445)은 코딩된 잔여를 예측에 가산하여 코딩된 화면 블록을 형성하고, 종료 루프 블록(1450)으로 제어를 전달한다. 종료 루프 블록(1450)은 루프를 종료하고, 종료 블록(1455)으로 제어를 전달한다.
- [0122] 도 15를 참조하면, 사후 디코딩 역 색 변환 공정이 있는 예시적인 비디오 디코딩 프로세스는 일반적으로 참조번호 1500 및 1560으로 각각 나타낸다.
- [0123] 사후 디코딩 역 색 변환 블록(1560)은 블록 1562, 1564, 1566 및 1568을 포함한다는 점을 인식하게 된다. 또한, 사후 디코딩 역 색 변환 블록(1560)은 옵션이고, 따라서 본 발명의 몇몇 실시예에서는 생략할 수도 있다는 점을 인식하게 된다.
- [0124] 디코딩 프로세스(1500)는 영상의 현재 블록에 대한 루프를 시작하고, 기능 블록(1515)으로 제어를 전달하는 루프 제한 블록(1510)을 포함한다. 기능 블록(1515)은 코딩된 잔여를 엔트로피 디코딩하고, 기능 블록(1520)으로 제어를 전달한다. 기능 블록(1520)은 디코딩된 잔여를 역변환 및 역 양자화하여 코딩된 잔여를 형성하고, 기능 블록(1530)으로 제어를 전달한다. 기능 블록(1530)은 코딩된 잔여를 각 색 성분에 대하여 공용 예측기로부터 형성한 예측에 가산하여 코딩된 화면 블록을 형성하고, 루프 제한 블록(1535)으로 제어를 전달한다. 루프 제한

블록(1535)은 루프를 종료하고, 종료 블록(1540)으로 제어를 전달한다.

- [0125] 몇몇 실시예에서, 루프 제한 블록(1535)은 사후 디코딩 역 색 변환 블록(1560), 특히 사후 디코딩 역 색 변환 블록(1560)에 포함되는 루프 제한 블록(1562)으로 선택적으로 제어를 전달한다. 루프 제한 블록(1562)은 영상의 각 블록에 대한 루프를 시작하고, 기능 블록(1564)으로 제어를 전달한다. 기능 블록(1564)은 현재 영상 블록의 비디오 신호 데이터의 역 색 사후 처리를 수행하고, 루프 제한 블록(1566)으로 제어를 전달한다. 루프 제한 블록(1566)은 루프를 종료하고, 종료 블록(1568)으로 제어를 전달한다.
- [0126] 전술한 바와 같이, Advanced 4:4:4 프로파일을 위한 새로운 프로파일(profile_idc=166)이 개시된다. 이 새로운 프로파일은 제2 조합 실시예에 이용할 수도 있고, 제2 조합 실시예에 대응하는 시맨틱 및 신택스 변화는 이하에서 후술한다. 이 새로운 profile_idc는 시퀀스 파라미터 세트에 추가하고, 후속 헤더에 주로 이용하여 입력 포맷은 4:4:4이고, 3개 입력 채널 모두는 루마로서 유사하게 코딩한다는 점을 나타낸다.
- [0127] H.264 표준에 대한 필수적인 변화를 최소화하기 위하여, 새로운 매크로블록 타입은 Advanced 4:4:4 프로파일에 대하여 개시하지 않는다. 대신, H.264 표준의 테이블 7-11, 테이블 7-13 및 테이블 7-14에 열거한 관련 코딩 파라미터와 함께 모든 매크로블록 타입은 여전히 유효하다. 인트라 매크로블록의 경우에 대하여, 3개 입력 채널, 루마, Cr 및 Cb 모두는 H.264 표준의 테이블 7-11에 정의한 MbPartPredMode에 기초하여 인코딩한다. 예를 들어, Advanced 4:4:4 프로파일의 Intra_4×4 매크로블록은 모든 입력 성분 채널은 H.264 표준의 테이블 8-2에 주어진 9개의 가능한 예측 모드 모두를 이용함으로써 인코딩할 수도 있다는 점을 의미한다. 참고로, 현재의 High 4:4:4 프로파일에서 Intra_4×4 매크로블록을 위한 채널 중 2개는 크로마로서 취급하고, H.264 표준의 테이블 8-5의 4개의 가능한 인트라 예측 모드 중 하나만이 이용된다. B 및 P 매크로블록에 대하여, Advanced 4:4:4 프로파일에 대하여 이루어진 변화는 분수형 픽셀 위치에 있는 기준 픽셀 값의 계산을 위한 보간 프로세스에서 발생한다. 여기서, H.264 표준의 섹션 8.4.2.2.1에 기술한 절차인 루마 샘플 보간 프로세스는 루마, Cr 및 Cb에 대하여 적용한다. 참고로, 현재의 High 4:4:4 프로파일은 H.264 표준의 섹션 8.4.2.2.2인 크로마 샘플 보간 프로세스를 입력 채널 중 2개에 대하여 이용한다.
- [0128] CABAC가 엔트로피 코딩 모드로서 선택되는 경우, 루마에 대하여 현재 정의된 것과 동일한 컨텍스트 모델의 2개의 개별 세트가 Cr 및 Cb에 대하여 생성된다. 또한, 그 세트는 인코딩 과정 동안 독립적으로 업데이트된다.
- [0129] 마지막으로, 이 실시예에서는 코딩 루프에 RCT 블록이 존재하지 않으므로, ResidueColorTransformFlag는 Advanced 4:4:4 프로파일의 시퀀스 파라미터 세트로부터 제거된다.
- [0130] 지금까지, 대부분의 신택스 변화는 도 13에 도시한 바와 같이 잔여 데이터에서 발생하는데, 루마에 대한 본래의 신택스는 제안된 Advanced 4:4:4 프로파일에 Cr 및 Cb를 지원하도록 2회 반복한다.
- [0131] H.264 매크로블록 계층 테이블(도시하지 않음)과 관련하여, 대응 신택스에 대한 시맨틱 변화는 다음을 포함한다.
- [0132] coded_block_pattern (Add). chroma_format_idc가 3과 동일하고, coded_block_pattern이 존재하는 경우, CodedBlockPatternChroma는 0으로 설정된다. 또한, CodedBlockPatternLuma는 매크로블록의 12개 8×8 루마, Cb 및 Cr 블록 각각에 대하여 (1) 8×8 루마, 8×8 Cb 및 8×8 Cr 블록 내 12개 4×4 루마 블록의 모든 변환 계수 레벨이 0과 동일한 경우 및 (2) 8×8 루마, 8×8 Cb 및 8×8 Cr 블록 내 4×4 루마 블록 중 하나 이상의 변환 계수 레벨 중 하나 이상이 0 값이 아닌 경우 중 하나를 지정한다.
- [0133] 이제, 제2 조합 실시예(또는 3개의 제한 공간 예측기의 세트(또는 서브세트)의 이용에 관한 단독 실시예)에 따른 인트라 블록에 대한 공간 예측 모드 선택에 관하여 설명한다.
- [0134] 최상의 MbPartPredMode 및 차선의 공간 예측 모드를 독립적으로 선택하는 각 성분에 대하여, 각 입력 채널을 별도로 인코딩하는 경우와 같이, 몇몇 새로운 인트라 블록 타입은 H.264 표준의 테이블 7-11에 추가될 수도 있다. 그 결과, H.264 표준에 대하여 많은 변화가 생긴다. 제2 조합 실시예에 관한 실시예에서, 현재의 mb_types는 변하지 않고, 다른 해결책이 제공된다. 이 실시예에서, 3개 입력 채널은 동일한 MbPartPredMode 또는 매크로블록 타입으로 인코딩하도록 제한된다. 다음으로, 소량의 새로운 구성요소가 매크로블록 예측 신택스에 추가되어 3개의 개별 예측 모드를 지원한다. 그러므로 각 성분 채널에 대한 예측 에러를 최소화하기 위하여, 각 성분은 여전히 이론적으로 최상의 공간 예측 모드를 독립적으로 선택할 수 있다. 예를 들어, Intra_4×4 매크로블록은 mb_type으로서 선택된다고 가정하면, 루마, Cr 또는 Cb는 H.264 표준의 섹션 8.3.1.1 내 테이블 8-2에서 최상의 공간 예측 모드, 예를 들어 루마를 위한 Intra_4×4_Vertical, Cr을 위한 Intra_4_4_Horizontal 및 Cb를 위한

Intra_4×4_Diagonal_Down_Left를 여전히 발견할 수 있다.

- [0135] 상술한 제1 조합 실시예에 관한 또 다른 접근법은 3개 입력 채널 모두를 제한하여 동일한 예측 모드를 공유하는 것이다. 이는, 매크로블록 예측 선택스에 있는 prev_intra4×4_pre_mode_flag, rem_intra4×4_pred_mode, pre_intra8×8_pred_mode_flag 및 rem_intra8×8_pred_mode와 같은 기존의 선택스 구성요소에 의해 현재 전달하는 예측 정보를 이용함으로써 이루어질 수 있다. 이 옵션은 H.264 표준에 대한 변화를 적게 하고, 코딩 효율의 손실도 적게 한다.
- [0136] 테스트 결과에 기초하면, 3개 예측 모드를 이용하여 제1 조합 실시예에 비해 약 0.2dB만큼 전체 코딩 성능을 향상시킬 수 있다.
- [0137] 도 16을 참조하면, H.264 매크로블록 예측 선택스에 대한 테이블은 일반적으로 참조번호 1700으로 나타낸다. 참고로, 3개 예측 모드를 이용하여 지원하는 변형된 매크로블록 예측 선택스는 아래와 같이 열거하는데,
- [0138] prev_intra4×4_pred_mode_flag0 및 rem_intra4×4_pred_mode0는 루마를 위한 것이고,
- [0139] prev_intra4×4_pred_mode_flag1 및 rem_intra4×4_pred_mode1은 Cr을 위한 것이며,
- [0140] prev_intra4×4_pred_mode_flag2 및 rem_intra4×4_pred_mode2는 Cb를 위한 것이다.
- [0141] 이제, 제2 조합 실시예에 대하여 실시예에서 구성한 본 발명의 원리에 따라 수행한 시뮬레이션 결과에 관하여 설명한다.
- [0142] 모든 JVT/FRExt 테스트 시퀀스는 JVT-J042, Film-Originated Test Sequence, JVT-J039(Viper)에 설명하였다. 테스트는 모두 4:4:4 10-비트 매체이고, 각 클립은 58개 프레임을 갖는다.
- [0143] 제안된 알고리즘은 JVT 기준 소프트웨어 JM9.6에서 구현하였고, 변형된 소프트웨어를 테스트에 이용하였다. Intra-Only 및 IBRRBP 둘 다를 테스트하였다. "Br"은 기록된 B 화면을 의미한다. Intra-Only 경우는 6, 12, 18, 24, 30, 36 및 42와 동일한 양자화 파라미터가 있는 시퀀스 모두에 대하여 이루어졌다. 시뮬레이션에 필요한 많은 시간 때문에, IBRRBP GOP 구조는 12, 18, 24, 30 및 36과 동일한 양자화 파라미터가 있는 영화 클립에 대해서만 이루어졌다. 4:4:4 AHG의 논의에 따르면, 다음의 주요 파라미터를 테스트에 이용하였다.
- [0144] SymbolMode = 1
- [0145] RDOptimization = 1
- [0146] ScalingMatrixPresentFlag = 0
- [0147] OffsetMatrixPresentFlag = 1
- [0148] QoffsetMatrixFile = "q_offset.cfg"
- [0149] AdaptiveRounding = 1
- [0150] AdaptRndPeriod = 1
- [0151] AdaptRndChroma = 1
- [0152] AdaptRndWFactorX = 8
- [0153] SearchRange = 64
- [0154] UseFME = 1
- [0155] JPEG2k에 관하여, KaKadu V2.2.3 소프트웨어를 테스트에 이용하였다. 테스트 결과는 9/7-탭 이중 직교 웨이브렛 필터로 웨이브렛 압축 해제 5 레벨을 이용함으로써 얻었다. 프레임당 하나의 타일만이 존재하였고, 주어진 목표율을 위한 RD-Optimization도 이용하였다.
- [0156] PSNR 측정은 상술한 클립을 위한 RGB인 소스 콘텐츠의 본래 색 도메인에서 주로 계산하였다. $(PSNR(red)+PSNR(green)+PSNR(blue))/3$ 으로서 정의한 평균 PSNR은 전체 압축 품질을 비교하는 데 이용한다.
- [0157] 압축 비교는 다음과 같이 수행하였다.
- [0158] New1: 단일 예측 모드가 있는 제안된 Advanced 4:4:4 프로파일.

- [0159] New3: 3개 예측 모드가 있는 제안된 Advanced 4:4:4 프로파일.
- [0160] RCT-OFF: RCT=off가 있는 RGB 입력.
- [0161] RCT-ON: RCT=on이 있는 RGB 입력.
- [0162] YCOCG: RGB 대 YCOCG 변환은 코텍 외부에서 수행하였다. 다음으로, 변환된 YCOCG는 JVT 소프트웨어에 대한 입력으로서 이용하였다.
- [0163] R+G+B: R, G 및 B 신호를 개별 압축함으로써 근사화되는 제안 방법.
- [0164] Y+CO+CG: 변환된 Y, CO 및 CG 신호를 개별 압축함으로써 근사화되는 제안 방법.
- [0165] JPEG2k_RGB: JPEG2k 압축은 RGB 도메인에서 수행하였다. JPEG2k 색 변환은 턴 오프되었다.
- [0166] JPEG2k_YUV: JPEG2k 압축은 YUV 도메인에서 수행하였다. JPEG2k 색 변환을 이용하였다.
- [0167] Intra-Only 경우에 대하여, 본 원리에 따른 제안된 Advanced 4:4:4 프로파일은 전체 압축 효율의 관점에서 JPEK2k와 매우 유사하다. 몇몇 경우에는 약간 더 양호하다.
- [0168] 본 발명의 원리에 따른 접근법은 현재의 High 4:4:4 프로파일보다 명백히 양호하다. 45dB (PSNR) 이상의 PSNR 에서, 평균 PSNR에서의 평균 향상은 1.5dB보다 크다. 몇몇의 경우, 이와 같은 향상은 45dB과 동일한 PSNR에서의 25%보다 큰 비트 절감으로 해석할 수 있다.
- [0169] 동일한 블록 타입이더라도, 단일 예측 모드보다 3개 예측 모드를 이용하는 것이 더 양호하다. 그러나 더 많은 신택스 및 시맨틱 변화를 이용할 수도 있다.
- [0170] 이제, 본 발명의 실시예의 원리에 따라 제공되는 많은 부수적인 장점/특징 중 일부를 설명한다.
- [0171] 테스트 결과는 본 발명의 원리에 대응하는 제안된 Advanced 4:4:4 프로파일은 현재의 High 4:4:4 프로파일과 비교하여 향상된 성능을 전달한다는 점을 설명한다. 성능 이득이 중요하다. 또한, 색 변환을 코텍 외부로 이동하여 코텍 구조를 모든 색 포맷에 일관성 있게 한다. 그 결과, 구현이 쉽고, 비용이 감소한다. 또한, 최적의 색 변환을 선택하여 더욱 양호한 코딩 효율을 얻는다는 관점에서 코텍을 더욱 견고하게 한다. 또한, 제안된 접근법은 새로운 코딩 톨을 추가하지 않고, 신택스 및 시맨틱에 대한 약간의 변화만을 필요로 한다.
- [0172] 따라서, 실시예에서 구성한 본 발명의 원리에 따르면, 비디오 인코딩 및 디코딩 방법 및 장치를 제공한다. 현재의 성능을 뛰어넘은 성능을 얻도록 기존의 H.264 표준에 대한 변형을 제공한다. 또한, 고품질 애플리케이션을 위한 JPEG-2000을 초과하는 향상된 성능을 얻는다. 실시예에서 구성한 본 발명의 원리에 따르면, 루마 코딩 알고리즘을 이용하여 4:4:4 콘텐츠의 3개 색 성분 모두를 코딩함으로써 H.264 표준에서 상당한 4:4:4 코딩 성능 향상을 이룰 수 있다. 즉, 루마(또는 이용하지 않는 크로마) 압축 알고리즘에 새로운 톨은 필요 없다. 대신, 기존의 루마 코딩 톨을 이용한다. 또한, 현재의 4:4:4 프로파일에 대한 신택스 및 시맨틱 변화는 본 원리에 따라 구현하여 3개 성분 채널 모두의 루마 코딩을 지원할 수도 있다. 본 발명의 실시예에 따라 수행한 테스트에서, 소스 콘텐츠가 많은 공간 텍스처 및 에지를 갖는 경우, 루마에 이용한 공간 예측 톨은 크로마에 이용한 공간 예측 톨보다 명백하게 우수한 성능을 나타내었다. 테스트 시퀀스 중 일부에 대하여, 모든 색 성분을 루마로서 인코딩했을 경우, 30%보다 큰 비트 감소가 45dB (평균 PSNR) 이상의 압축 품질에서 관찰되었다.
- [0173] 본 명세서에서 본 발명은 H.264 표준의 4:4:4 포맷을 이용하여 샘플링한 비디오 신호 데이터에 관하여 주로 기술하지만, 본 발명은 H.264 표준의 다른 포맷(예를 들어, 4:2:0 포맷 및/또는 4:2:2 포맷)뿐만 아니라 다른 비디오 압축 표준을 이용하여 샘플링한 비디오 신호 데이터에 관하여 쉽게 구현할 수도 있다는 점을 인식하게 된다. 본 명세서에서 제공하는 본 발명의 내용에 따르면, 본 기술분야의 당업자는 본 발명의 범위를 유지하면서 본 발명의 이러한 장점 및 다른 장점을 용이하게 구현할 수도 있다.
- [0174] 본 발명의 이러한 특징과 장점 및 다른 특징과 장점은 본 명세서의 내용에 기초하여 본 기술분야의 당업자가 용이하게 규명할 수도 있다. 본 발명의 내용은 하드웨어, 소프트웨어, 펌웨어, 특수용 프로세서 또는 그 조합의 다양한 형태로 구현할 수도 있다는 점을 이해하게 된다.
- [0175] 더욱 바람직하게, 본 발명의 내용은 하드웨어와 소프트웨어의 조합으로서 구현한다. 또한, 소프트웨어는 프로그램 저장 유닛을 통해 명백히 구현하는 애플리케이션 프로그램으로서 구현할 수도 있다. 애플리케이션 프로그램은 적당한 구조가 있는 머신으로 업로드 및 실행될 수도 있다. 바람직하게, 머신은 하나 이상의 중앙 처리 유닛(CPU), RAM 및 입/출력(I/O) 인터페이스와 같은 하드웨어를 구비한 컴퓨터 플랫폼상에 구현한다. 컴퓨터

플랫폼은 오퍼레이팅 시스템 및 마이크로 명령어 코드를 포함할 수도 있다. 본 명세서에 기술한 다양한 프로세스 및 기능은 마이크로 명령어 코드의 일부분 또는 애플리케이션 프로그램의 일부분 혹은 그 조합일 수도 있고, CPU가 실행할 수도 있다. 또한, 추가 데이터 저장 유닛 및 인쇄 유닛과 같은 다양한 다른 주변 유닛을 컴퓨터 플랫폼에 연결할 수도 있다.

[0176] 첨부한 도면에 도시한 구성 시스템 컴포넌트 및 방법 중 일부는 바람직하게 소프트웨어로 구현하기 때문에, 시스템 컴포넌트 또는 프로세스 기능 블록 간의 실제 연결은 본 발명이 프로그램되는 방식에 따라 달라질 수도 있다는 점을 이해하게 된다. 본 명세서의 내용에 따르면, 본 기술분야의 당업자는 본 발명의 이러한 구현 또는 구성 및 유사한 구현 또는 구성을 예상할 수 있다.

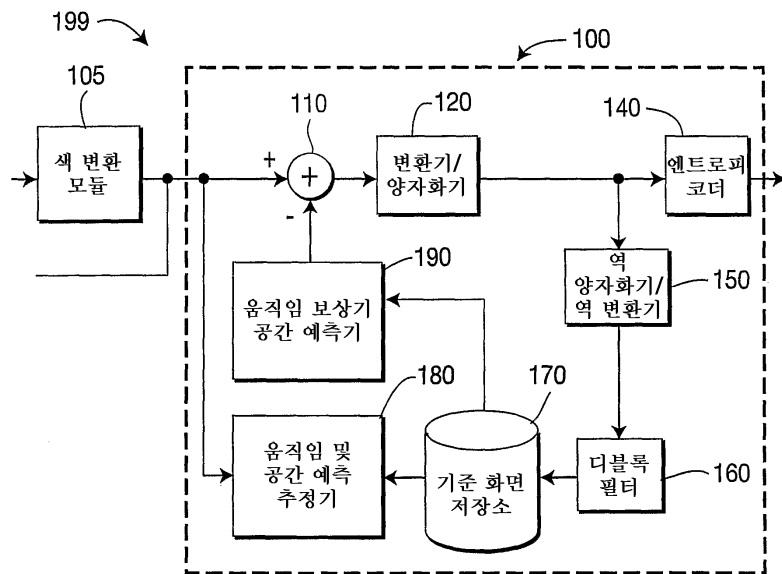
[0177] 본 명세서에서는 첨부한 도면을 참조하여 예시적 실시예를 기술하지만, 본 발명은 그와 같은 실시예에 한정하지 않고, 본 기술분야의 당업자는 본 발명의 범위 또는 사상을 벗어나지 않으면서 본 발명을 다양한 변화 및 변형할 수도 있다는 점을 이해하게 된다. 그와 같은 변화 및 변형 모두는 첨부한 청구범위에 기술하는 본 발명의 범위 내에 포함된다.

도면의 간단한 설명

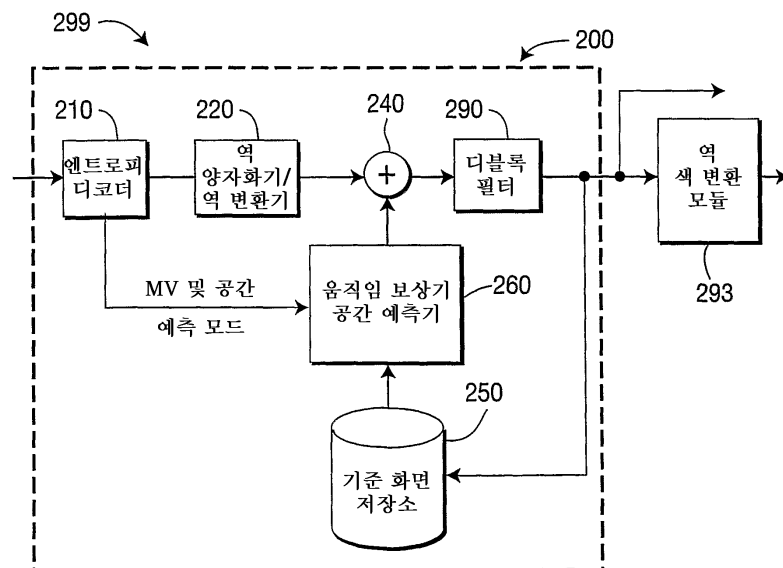
- [0013] 본 발명은 이하의 예시적인 도면에 따라 더욱 양호하게 이해할 수도 있다.
- [0014] 도 1은 본 원리를 적용할 수도 있는 예시적인 비디오 인코딩 장치를 나타내는 블록도이다.
- [0015] 도 2는 본 원리를 적용할 수도 있는 예시적인 비디오 디코딩 장치를 나타내는 블록도이다.
- [0016] 도 3은 본 원리에 따른 사전 인코딩 색 변환 블록이 있는 예시적인 비디오 인코딩 프로세스를 나타내는 흐름도이다.
- [0017] 도 4는 본 원리에 따른 사후 디코딩 역 색 변환 블록이 있는 예시적인 비디오 디코딩 프로세스를 나타내는 흐름도이다.
- [0018] 도 5는 잔여 색 변환(RCT)의 간소화 모델을 나타내는 블록도이다.
- [0019] 도 6a 및 6b는 본 원리에 따른 ATV Intra-Only에 대한 평균 PSNR 대 비트율의 도면이다.
- [0020] 도 7a 및 7b는 본 원리에 따른 CT Intra-Only에 대한 평균 PSNR 대 비트율의 도면이다.
- [0021] 도 8a 및 8b는 본 원리에 따른 DT Intra-Only에 대한 평균 PSNR 대 비트율의 도면이다.
- [0022] 도 9a 및 9b는 본 원리에 따른 MIR_HD Intra-Only에 대한 평균 PSNR 대 비트율의 도면이다.
- [0023] 도 10a 및 10b는 본 원리에 따른 RT Intra-Only에 대한 평균 PSNR 대 비트율의 도면이다.
- [0024] 도 11a 및 11b는 본 원리에 따른 STB_HD Intra-Only에 대한 평균 PSNR 대 비트율의 도면이다.
- [0025] 도 12는 본 원리에 따른 H.264 시퀀스 파라미터 섹터를 나타내는 테이블이다.
- [0026] 도 13은 본 원리에 따른 H.264 잔여 데이터 섹터를 나타내는 테이블이다.
- [0027] 도 14는 본 원리에 따른 사전 인코딩 색 변환 블록이 있는 예시적인 비디오 인코딩 프로세스를 나타내는 흐름도이다.
- [0028] 도 15는 본 원리에 따른 사후 디코딩 역 색 변환 공정 블록이 있는 예시적인 비디오 디코딩 프로세스를 나타내는 흐름도이다.
- [0029] 도 16은 본 원리에 따른 H.264 매크로블록 예측 섹터를 나타내는 테이블이다.

도면

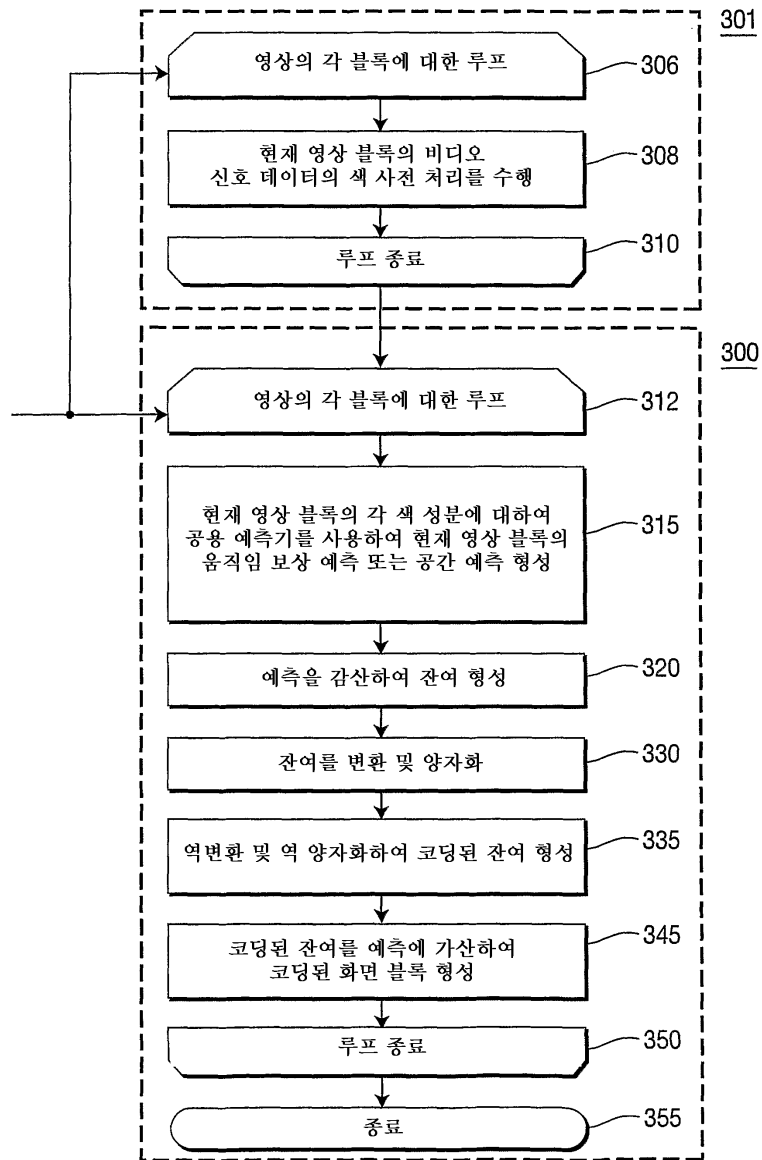
도면1



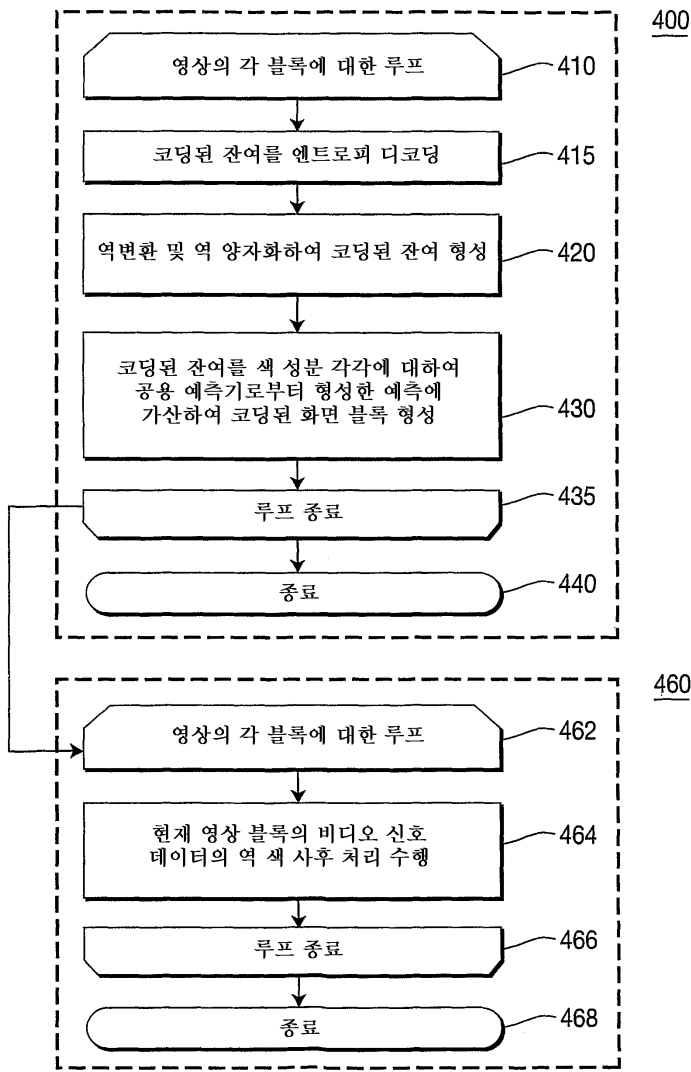
도면2



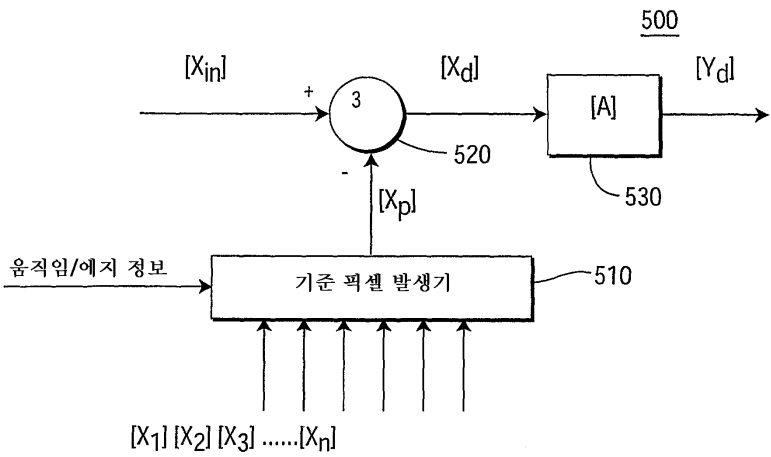
도면3



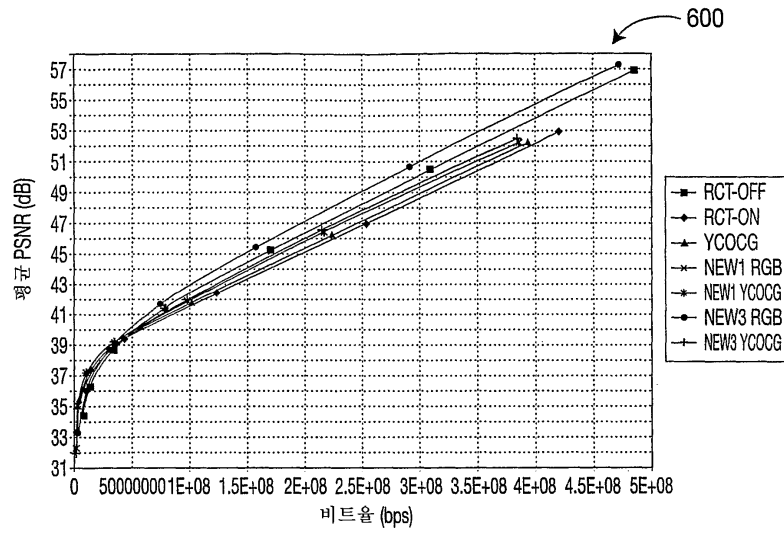
도면4



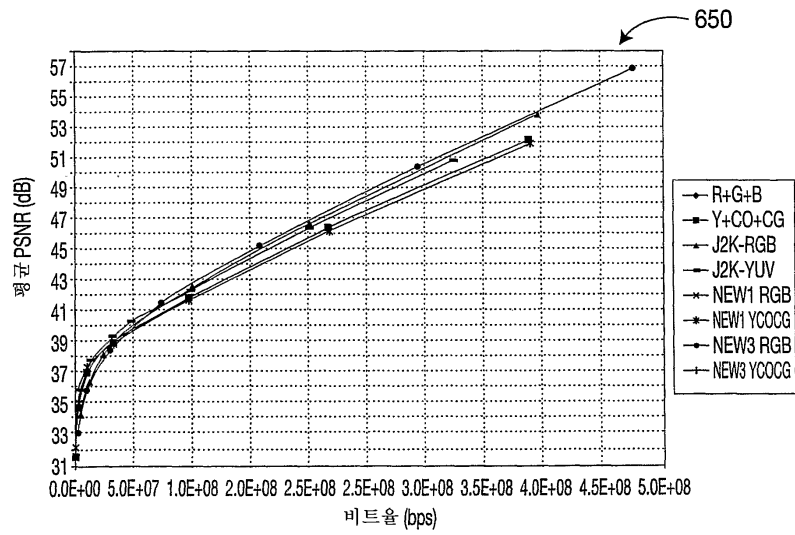
도면5



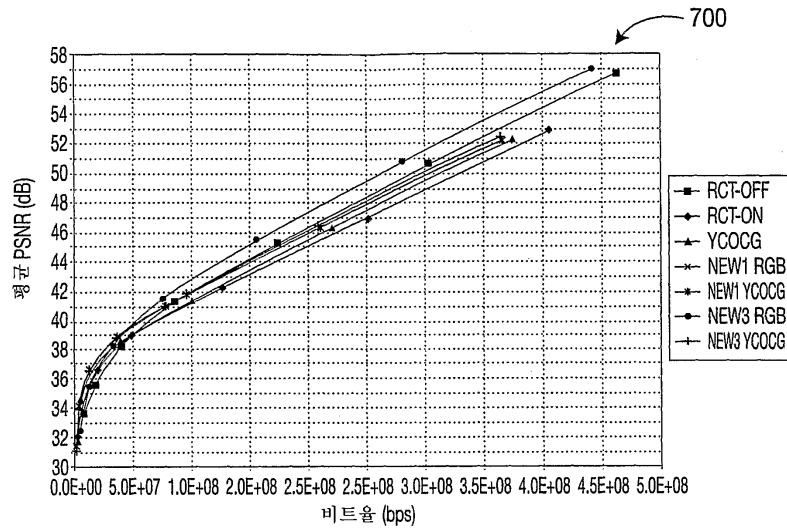
도면6a



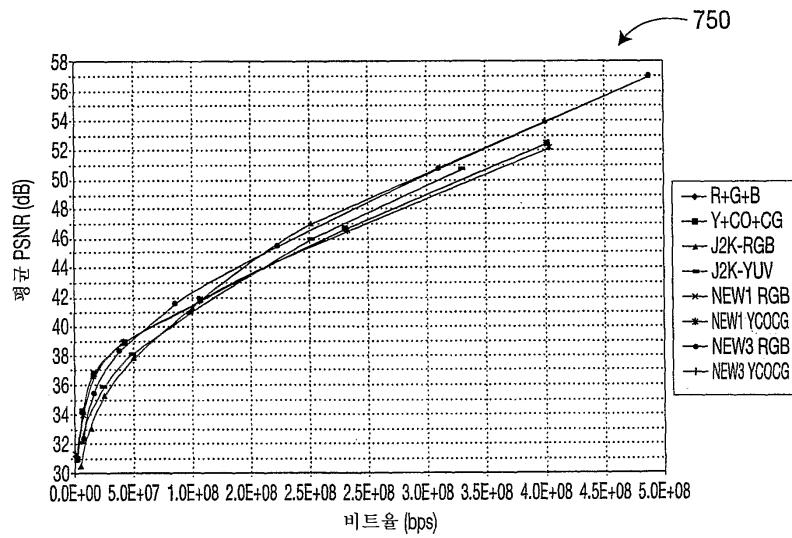
도면6b



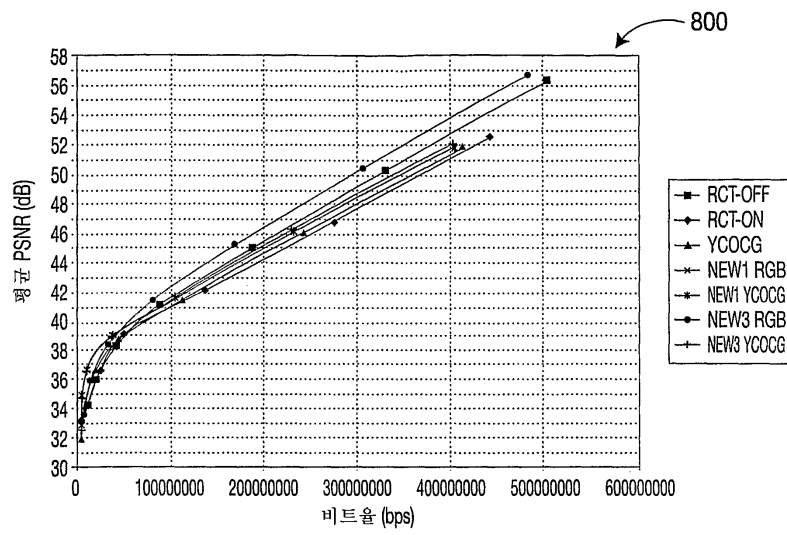
도면7a



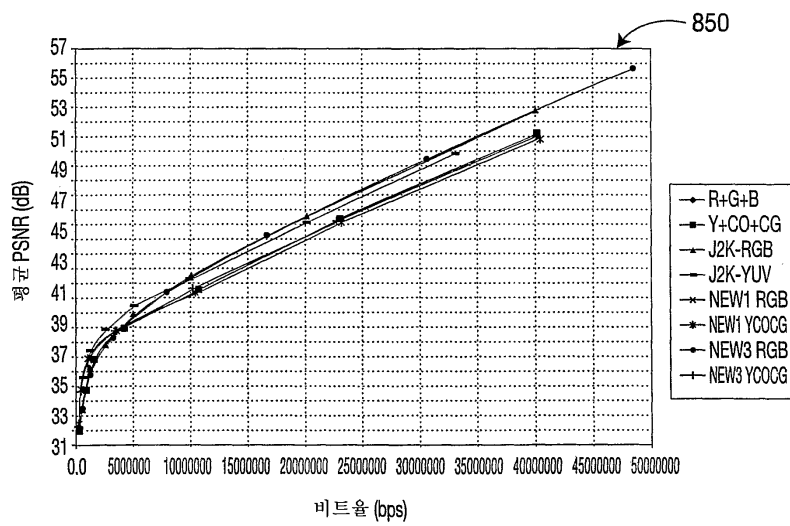
도면7b



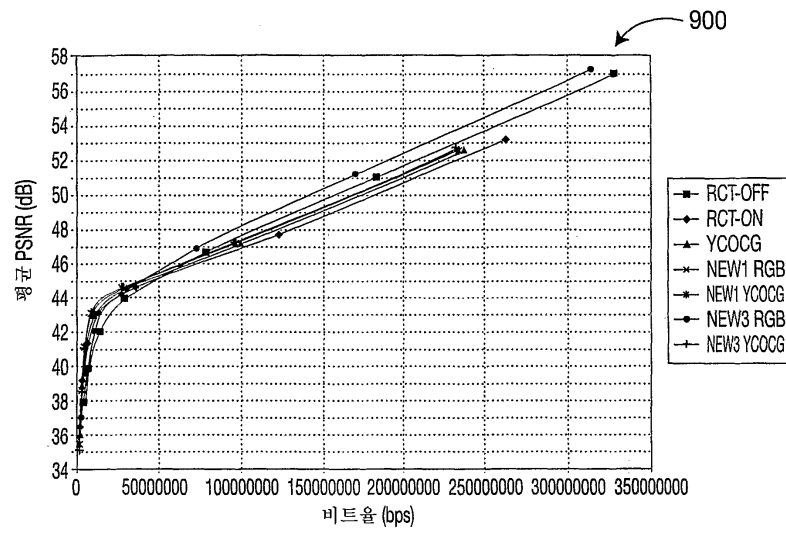
도면8a



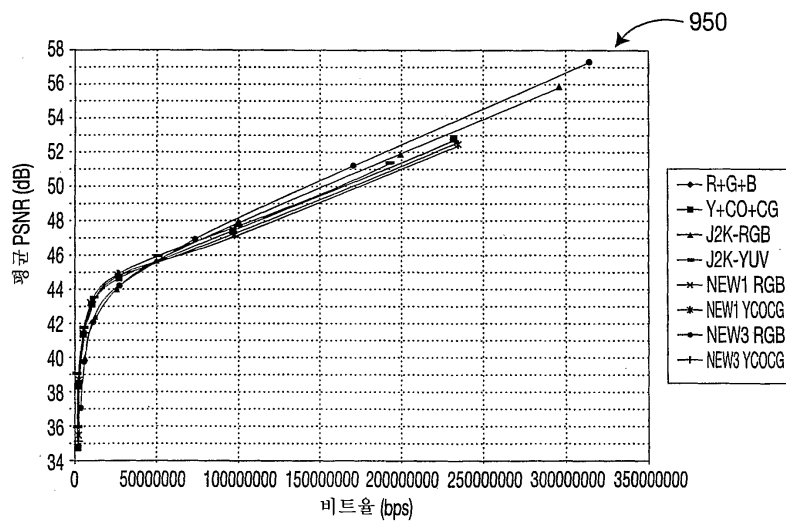
도면8b



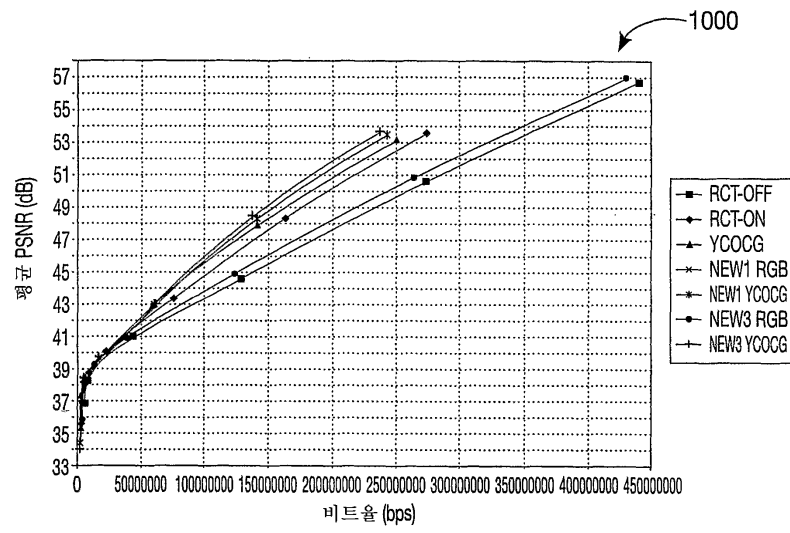
도면9a



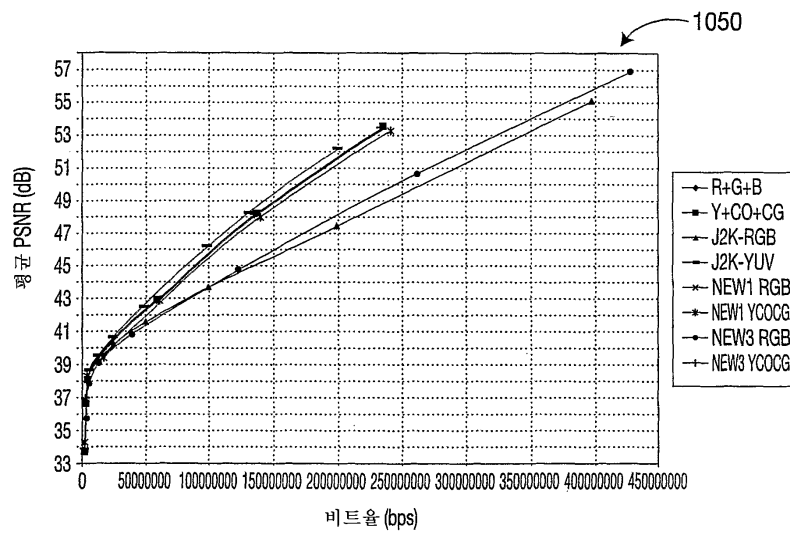
도면9b



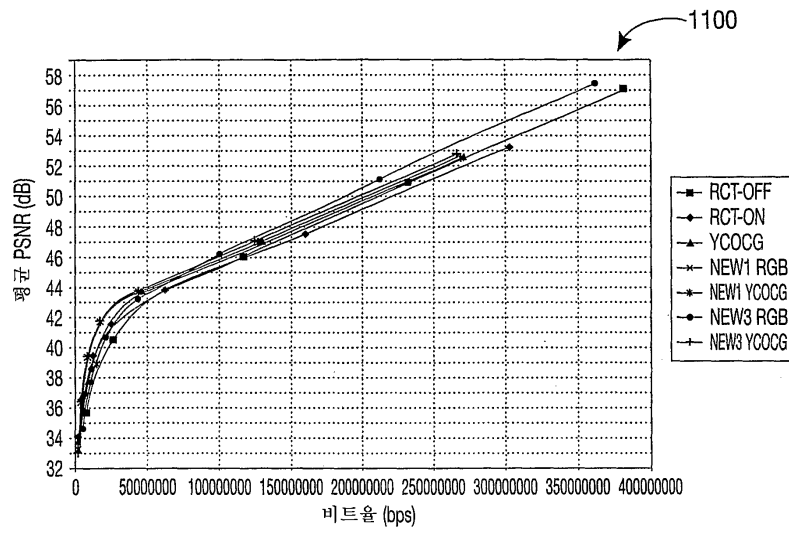
도면10a



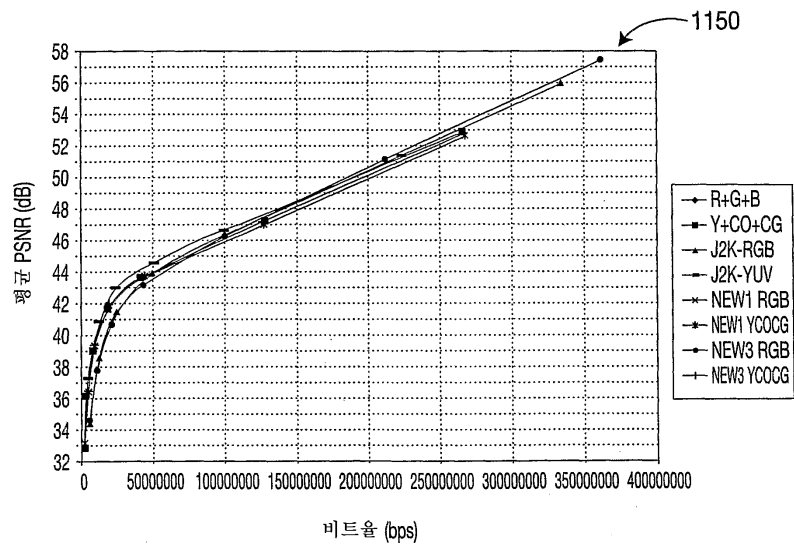
도면10b



도면11a



도면11b



도면12

1200

seq_parameter_set_rbsp(){	C	디스크립터
profile_idc	0	u(8)
constraint_set0_flag	0	u(1)
constraint_set1_flag	0	u(1)
constraint_set2_flag	0	u(1)
constraint_set3_flag	0	u(1)
reserved_zero_4bits /* equal to 0 */	0	u(4)
level_idc	0	u(8)
seq_parameter_set_id	0	ue(v)
if(profile_idc == 100 profile_idc == 110 profile_idc == 122 profile_idc == 166 profile_idc == 188){		
chroma_format_idc	0	ue(v)
bit_depth_luma_minus8	0	ue(v)

도면13a

Residual() {	C	디스크립터
if(!entropy_coding_mode_flag)		
residual_block = residual_block_cavlc		
else		
residual_block = residual_block_cabac		
if(MbPartPredMode(mb_type, 0) == Intra_16x16)		
residual_block(Intra16x16DCLLevel, 16)	3	
for(i8x8 = 0; i8x8 < 4; i8x8++) /* each luma 8x8 block */		
if(!transform_size_8x8_flag !entropy_coding_mode_flag)		
for(i4x4 = 0; i4x4 < 4; i4x4++) { /* each 4x4 sub-block of block */		
if(CodedBlockPatternLuma & (1 << i8x8))		
if(MbPartPredMode(mb_type, 0) == Intra_16x16)		
residual_block(Intra16x16ACLevel[i8x8 * 4 + i4x4], 15)	3	
else		
residual_block(LumaLevel[i8x8 * 4 + i4x4], 16)	3 4	
else if(MbPartPredMode(mb_type, 0) == Intra_16x16)		
for(i = 0; i < 15; i++)		
Intra16x16ACLevel[i8x8 * 4 + i4x4][i] = 0		
else		
for(i = 0; i < 16; i++)		
LumaLevel[i8x8 * 4 + i4x4][i] = 0		
if(entropy_coding_mode_flag && transform_size_8x8_flag)		
for(i = 0; i < 16; i++)		
LumaLevel8x8[i8x8][4 * i + i4x4] =		
LumaLevel[i8x8 * 4 + i4x4][i]		
}		

도면13b

else if(CodedBlockPatternLuma & (1 << i8x8))		
residual_block(LumaLevel8x8[i8x8], 64)	3 4	
else		
for(i = 0; i < 64; i++)		
LumaLevel8x8[i8x8][i] = 0		
if(chroma_format_idc == 1 chroma_format_idc == 2) {		
NumC8x8 = 4 / (SubWidthC * SubHeightC)		
for(iCbCr = 0; iCbCr < 2; iCbCr++)		
if(CodedBlockPatternChroma & 3) /* chroma DC residual present */		
residual_block(ChromaDCLevel[iCbCr], 4 * NumC8x8)	3 4	
else		
for(i = 0; i < 4 * NumC8x8; i++)		
ChromaDCLevel[iCbCr][i] = 0		
for(iCbCr = 0; iCbCr < 2; iCbCr++)		
for(i8x8 = 0; i8x8 < NumC8x8; i8x8++)		
for(i4x4 = 0; i4x4 < 4; i4x4++)		
if(CodedBlockPatternChroma & 2)		
/* chroma AC residual present */		
residual_block(ChromaACLevel[iCbCr][i8x8*4+i4x4], 15)	3 4	
else		
for(i = 0; i < 15; i++)		
ChromaACLevel[iCbCr][i8x8*4+i4x4][i] = 0		
} else if(chroma_format_idc == 3) {		
if(MbPartPredMode(mb_type, 0) == Intra_16x16) /* Cb for 4:4:4 */		
residual_block(CbIntra16x16DCLevel, 16)	3	
for(i8x8 = 0; i8x8 < 4; i8x8++) /* each Cb 8x8 block */		
if(!transform_size_8x8_flag !entropy_coding_mode_flag)		
for(i4x4 = 0; i4x4 < 4; i4x4++) { /* each 4x4 sub-block of block		
*/		
if(CodedBlockPatternLuma & (1 << i8x8))		

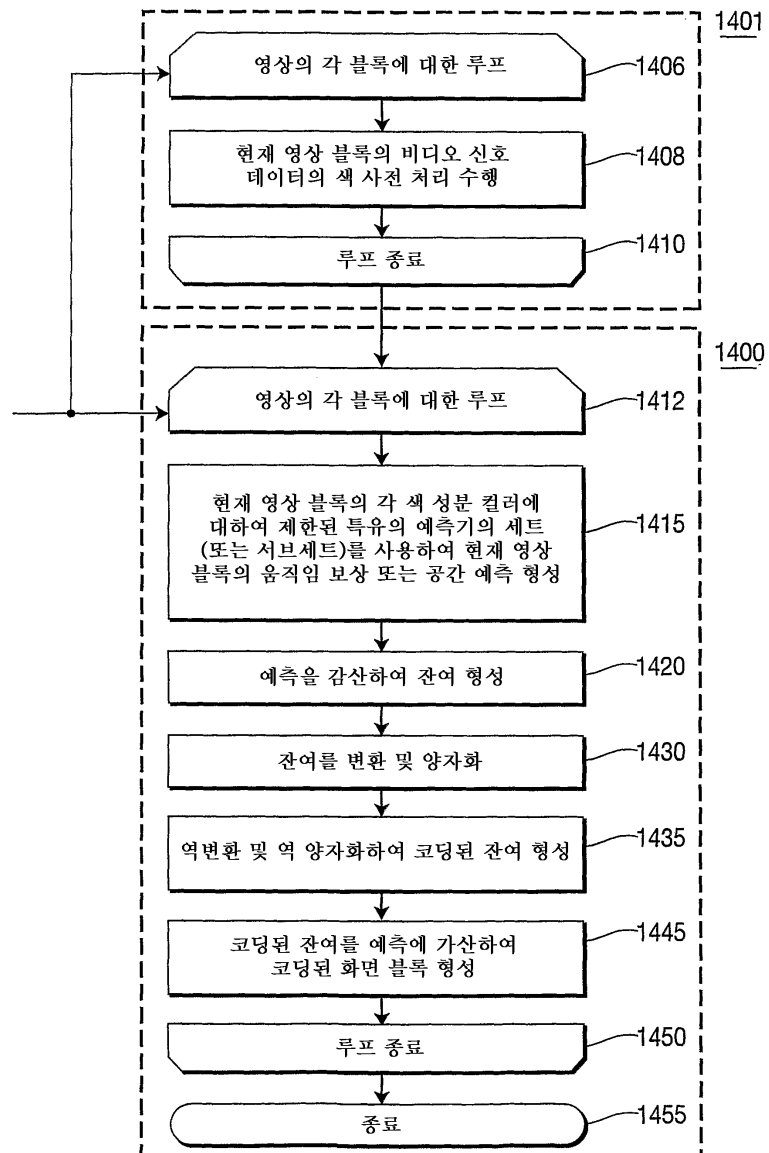
도면13c

if(MbPartPredMode(mb_type, 0) == Intra_16x16)		
residual_block(CbIntra16x16ACLevel[i8x8*4 + i4x4], 15)	3	
else		
residual_block(CbLevel[i8x8 * 4 + i4x4], 16)	3 4	
else if(MbPartPredMode(mb_type, 0) == Intra_16x16)		
for(i = 0; i < 15; i++)		
CbIntra16x16ACLevel[i8x8 * 4 + i4x4][i] = 0		
else		
for(i = 0; i < 16; i++)		
CbLevel[i8x8 * 4 + i4x4][i] = 0		
if(!entropy_coding_mode_flag && transform_size_8x8_flag)		
for(i = 0; i < 16; i++)		
CbLevel8x8[i8x8][4 * i + i4x4] =		
CbLevel[i8x8 * 4 + i4x4][i]		
}		
else if(CodedBlockPatternLuma & (1 << i8x8))		
residual_block(CbLevel8x8[i8x8], 64)	3 4	
else		
for(i = 0; i < 64; i++)		
CbLevel8x8[i8x8][i] = 0		
if(MbPartPredMode(mb_type, 0) == Intra_16x16) /*Cr for 4:4:4 */		
residual_block(CrIntra16x16DCLevel, 16)	3	
for(i8x8 = 0; i8x8 < 4; i8x8++) /* each Cr 8x8 block */		
if(!transform_size_8x8_flag !entropy_coding_mode_flag)		
for(i4x4 = 0; i4x4 < 4; i4x4++) { /* each 4x4 sub-block of block		
*/		
if(CodedBlockPatternLuma & (1 << i8x8))		
if(MbPartPredMode(mb_type, 0) == Intra_16x16)		
residual_block(CrIntra16x16ACLevel[i8x8*4+i4x4], 15)	3	
else		

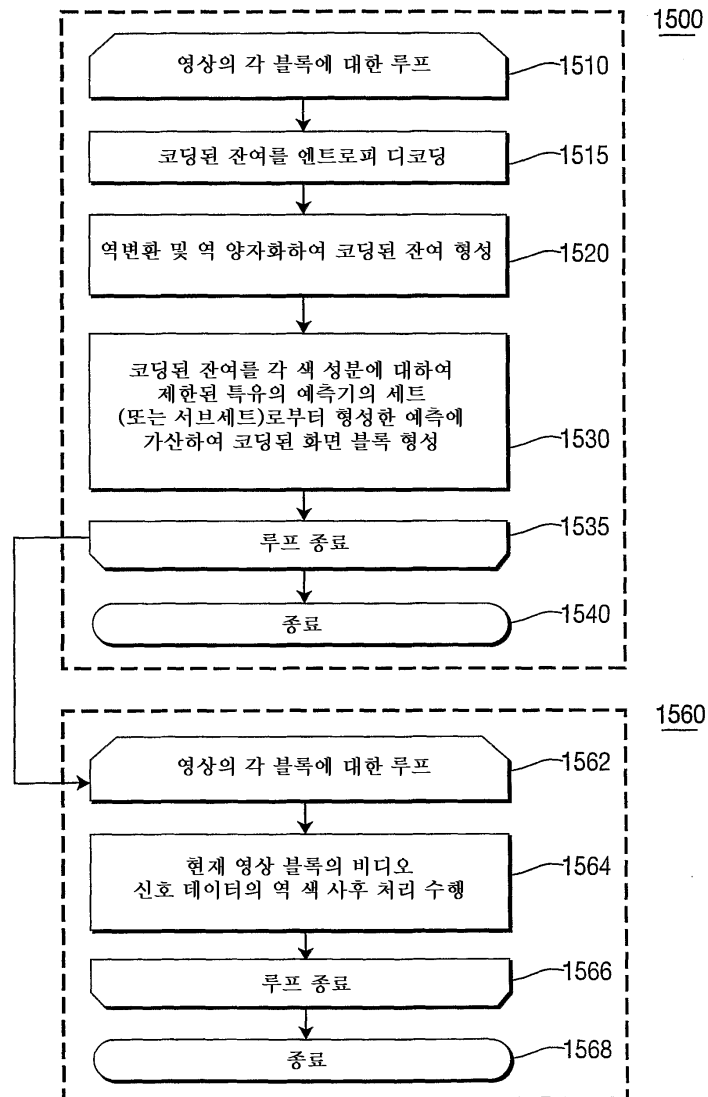
도면13d

residual_block(CrLevel[i8x8 * 4 + i4x4], 16)	3 4	
else if(MbPartPredMode(mb_type, 0) == Intra_16x16)		
for(i = 0; i < 15; i++)		
CrIntra16x16ACLevel[i8x8 * 4 + i4x4][i] = 0		
else		
for(i = 0; i < 16; i++)		
CrLevel[i8x8 * 4 + i4x4][i] = 0		
if(!entropy_coding_mode_flag && transform_size_8x8_flag)		
for(i = 0; i < 16; i++)		
CrLevel8x8[i8x8][4 * i + i4x4] = CrLevel[i8x8 * 4 + i4x4][i]		
}		
else if(CodedBlockPatternLuma & (1 << i8x8))		
residual_block(CrLevel8x8[i8x8], 64)	3 4	
else		
for(i = 0; i < 64; i++)		
CrLevel8x8[i8x8][i] = 0		
}		

도면14



도면15



도면16a

1600

mb_pred(mb_type) {	C	Descriptor
if(MbPartPredMode(mb_type, 0) == Intra_4x4 MbPartPredMode(mb_type, 0) == Intra_8x8 MbPartPredMode(mb_type, 0) == Intra_16x16) {		
if (profile_idc != 166) {		
if(MbPartPredMode(mb_type, 0) == Intra_4x4)		
for(luma4x4BlkIdx=0; luma4x4BlkIdx<16; luma4x4BlkIdx++) {		
prev_intra4x4_pred_mode_flag[luma4x4BlkIdx]	2	u(1) ae(v)
if(!prev_intra4x4_pred_mode_flag[luma4x4BlkIdx])		
rem_intra4x4_pred_mode[luma4x4BlkIdx]	2	u(3) ae(v)
}		
if(MbPartPredMode(mb_type, 0) == Intra_8x8)		
for(luma8x8BlkIdx=0; luma8x8BlkIdx<4; luma8x8BlkIdx++) {		
prev_intra8x8_pred_mode_flag[luma8x8BlkIdx]	2	u(1) ae(v)
if(!prev_intra8x8_pred_mode_flag[luma8x8BlkIdx])		
rem_intra8x8_pred_mode[luma8x8BlkIdx]	2	u(3) ae(v)
}		
if(chroma_format_idc != 0)		
intra_chroma_pred_mode	2	u(v) ae(v)
} else {		
if(MbPartPredMode(mb_type, 0) == Intra_4x4)		
for(luma4x4BlkIdx=0; luma4x4BlkIdx<16; luma4x4BlkIdx++) {		
prev_intra4x4_pred_mode_flag0[luma4x4BlkIdx]	2	u(1) ae(v)
if(!prev_intra4x4_pred_mode_flag0[luma4x4BlkIdx])		
rem_intra4x4_pred_mode0[luma4x4BlkIdx]	2	u(3) ae(v)
prev_intra4x4_pred_mode_flag1[luma4x4BlkIdx]	2	u(1) ae(v)

도면16b

if(!prev_intra4x4_pred_mode_flag[luma4x4BlkIdx])		
rem_intra4x4_pred_mode1[luma4x4BlkIdx]	2	u(3) ae(v)
prev_intra4x4_pred_mode_flag2[luma4x4BlkIdx]	2	u(1) ae(v)
if(!prev_intra4x4_pred_mode_flag2[luma4x4BlkIdx])		
rem_intra4x4_pred_mode2[luma4x4BlkIdx]	2	u(3) ae(v)
}		
if(MbPartPredMode(mb_type, 0) == Intra_8x8)		
for(luma8x8BlkIdx=0; luma8x8BlkIdx<4; luma8x8BlkIdx++) {		
prev_intra8x8_pred_mode_flag0[luma8x8BlkIdx]	2	u(1) ae(v)
if(!prev_intra8x8_pred_mode_flag0[luma8x8BlkIdx])		
rem_intra8x8_pred_mode0[luma8x8BlkIdx]	2	u(3) ae(v)
prev_intra8x8_pred_mode_flag1[luma8x8BlkIdx]	2	u(1) ae(v)
if(!prev_intra8x8_pred_mode_flag1[luma8x8BlkIdx])		
rem_intra8x8_pred_mode1[luma8x8BlkIdx]	2	u(3) ae(v)
prev_intra8x8_pred_mode_flag2[luma8x8BlkIdx]	2	u(1) ae(v)
if(!prev_intra8x8_pred_mode_flag2[luma8x8BlkIdx])		
rem_intra8x8_pred_mode2[luma8x8BlkIdx]	2	u(3) ae(v)
}		
}		
} else if(MbPartPredMode(mb_type, 0) != Direct) {		
.....		