



(19) 대한민국특허청(KR)

(12) 등록특허공보(B1)

(45) 공고일자 2021년03월15일

(11) 등록번호 10-2227161

(24) 등록일자 2021년03월08일

- (51) 국제특허분류(Int. Cl.)
H04N 21/431 (2016.01) *H04N 21/234* (2014.01)
H04N 21/433 (2011.01) *H04N 21/44* (2011.01)
- (52) CPC특허분류
H04N 21/4318 (2013.01)
H04N 21/23418 (2013.01)
- (21) 출원번호 10-2020-7033924(분할)
 (22) 출원일자(국제) 2016년12월16일
 심사청구일자 2020년11월25일
- (85) 번역문제출일자 2020년11월25일
 (65) 공개번호 10-2020-0135885
 (43) 공개일자 2020년12월03일
 (62) 원출원 특허 10-2020-7023726
 원출원일자(국제) 2016년12월16일
 심사청구일자 2020년08월18일
- (86) 국제출원번호 PCT/US2016/067250
 (87) 국제공개번호 WO 2017/106695
 국제공개일자 2017년06월22일
- (30) 우선권주장
 62/268,410 2015년12월16일 미국(US)
- (56) 선행기술조사문헌
 JP2015039216 A*
 US20090199230 A1*
 KR1020070047656 A
 US20130031582 A1
 *는 심사관에 의하여 인용된 문헌

- (73) 특허권자
 그레이스노트, 인코포레이티드
 미국, 캘리포니아 94608, 에머리빌, 슈트 1380,
 포웰 스트리트 2000
- (72) 발명자
 크리머 마르쿠스 케이.
 미국 캘리포니아주 94563 오린다 로스트 밸리 드
 라이브 15
 던커 피터
 미국 캘리포니아주 94530 엘세리토 산칼로스 애비
 뉴 146
 (뒷면에 계속)
- (74) 대리인
 문두현

전체 청구항 수 : 총 19 항

심사관 : 김응권

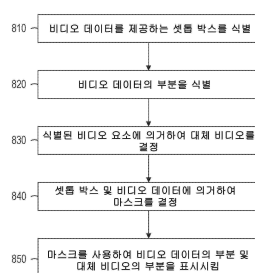
(54) 발명의 명칭 동적 비디오 오버레이

(57) 요약

클라이언트 디바이스는, 표시를 위해 중간 디바이스로부터의 비디오 입력 스트림에 액세스한다. 클라이언트 디바이스는 비디오 입력 스트림을 분석해서, 비디오 입력 스트림이 스크린 부분에 대응하는 템플릿과 매칭됨을 판정한다. 비디오 입력 스트림이 템플릿과 매칭됨에 의거하여, 비디오 출력 스트림을 생성하고, 디스플레이 상에

(뒷면에 계속)

대표도 - 도8



제시된다. 일부 예시적인 실시예에서, 분석은, 클라이언트 디바이스가, 중간 디바이스를 통해 콘텐츠 소스로부터 수신된 비디오 콘텐츠를 대체하는 동안에, 실행된다. 예를 들면, 전국 콘텐츠 프로바이더로부터 셋톱 박스를 통해 스마트 TV에 전송되는 커머셜은 타겟 커머셜로 대체될 수 있다. 교환 동안에, 셋톱 박스에 의해 생성된 메뉴가 검출되고, 대체 비디오가 스마트 TV에 의해 메뉴를 포함하도록 변경될 수 있다.

(52) CPC특허분류

H04N 21/4314 (2013.01)

H04N 21/4333 (2013.01)

H04N 21/44008 (2013.01)

H04N 21/44016 (2013.01)

(72) 발명자

셔프 스티븐 디.

미국 캘리포니아주 94555 프리몬트 몰리나리 테라스 6040

피어스 데이비드 헨리

미국 캘리포니아주 94608 에머리빌 스위트 1380 파월 스트리트 2000

리 셔먼 링 평

미국 캘리포니아주 94608 에머리빌 스위트 1380 파월 스트리트 2000

해런 월슨

미국 캘리포니아주 94705 버클리 새터 애비뉴 3032

디미트리우 콘스탄티노스 안토니오스

독일 바바리아 82008 운터하링 파자넨스트라쎄 134

리 듀이 호

미국 캘리포니아주 94608 에머리빌 스위트 1380 파월 스트리트 2000

명세서

청구범위

청구항 1

시스템으로서,

디스플레이;

명령어들 및 복수의 템플릿들을 저장하는 메모리; 및

상기 명령어들에 의해 동작들을 행하도록 구성되는 하나 이상의 프로세서를 포함하고,

상기 동작들은,

콘텐츠 프로바이더로부터의 제1 콘텐츠 및 중간 디바이스로부터의 제2 콘텐츠를 포함하는 비디오 입력 스트림에 액세스하는 동작 — 상기 비디오 입력 스트림의 프레임에서, 상기 제1 콘텐츠는 상기 디스플레이의 제1 스크린 부분에 대응하고 상기 제2 콘텐츠는 상기 디스플레이의 제2 스크린 부분에 대응함 —;

상기 중간 디바이스의 식별자에 액세스하는 동작;

상기 중간 디바이스의 식별자에 액세스하는 동작 후에, 상기 중간 디바이스의 식별자에 의거하여 상기 복수의 템플릿들의 서브셋을 선택하는 동작 — 상기 복수의 템플릿들의 서브셋은 상기 복수의 템플릿들 중 둘 이상의 템플릿을 포함함 —;

상기 복수의 템플릿들의 서브셋을 선택하는 동작 후에, 상기 비디오 입력 스트림의 프레임이 식별된 템플릿에 대응함을 판정하도록 상기 비디오 입력 스트림을 상기 복수의 템플릿들의 서브셋과 비교하는 동작 — 상기 식별된 템플릿은 상기 제2 스크린 부분이 상기 중간 디바이스로부터의 제2 콘텐츠를 표시하기 위한 오버레이를 포함함을 지시(indication)함 —;

상기 식별된 템플릿에 액세스하는 동작;

상기 비디오 입력 스트림의 프레임이 상기 식별된 템플릿에 대응한다는 판정에 응답해서, 상기 제2 스크린 부분에 대한 제2 콘텐츠 및 상기 제1 스크린 부분에 대한 제3 콘텐츠를 포함하는 비디오 출력 스트림을 생성하는 동작 — 상기 제3 콘텐츠는 상기 비디오 입력 스트림의 제1 콘텐츠와 상이함 —; 및

상기 비디오 출력 스트림이 상기 디스플레이 상에 제시되게 하는 동작을 포함하고,

상기 비디오 출력 스트림을 생성하는 동작은,

상기 식별된 템플릿에 의거하여, 상기 제2 스크린 부분의 사이즈 및 위치에 대한 복수의 좌표들을 판정하는 동작, 및

상기 비디오 입력 스트림에 액세스하는 동작 동안, 상기 복수의 좌표들을 이용하여, 상기 제2 스크린 부분에서 상기 제2 콘텐츠 및 상기 제1 스크린 부분에서 상기 제3 콘텐츠를 갖는 비디오 출력 스트림을 동적으로 생성하는 동작을 포함하는 시스템.

청구항 2

제1항에 있어서,

상기 동작들은,

상기 비디오 입력 스트림의 핑거프린트를 생성하는 동작;

상기 핑거프린트에 의거하여 상기 콘텐츠 프로바이더에 의해 제공된 비디오 콘텐츠를 식별하는 동작; 및

상기 콘텐츠 프로바이더에 의해 제공된 비디오 콘텐츠의 식별에 의거하여 대체 비디오 콘텐츠를 식별하는 동작을 더 포함하고;

상기 생성된 비디오 출력 스트림의 제3 콘텐츠는 상기 식별된 대체 비디오 콘텐츠의 적어도 일부를 포함하는 시스템.

청구항 3

제1항에 있어서,
상기 콘텐츠 프로바이더는 케이블 TV 스테이션인 시스템.

청구항 4

제1항에 있어서,
상기 중간 디바이스는 셋톱 박스인 시스템.

청구항 5

제1항에 있어서,
상기 시스템은 스마트 TV인 시스템.

청구항 6

제1항에 있어서,
상기 비디오 입력 스트림을 상기 식별된 템플릿과 비교하는 동작은,
상기 비디오 입력 스트림의 프레임을 다운샘플링하는 동작;
상기 다운샘플링된 프레임과 상기 식별된 템플릿 사이의 상호상관을 판정하는 동작; 및
상기 상호상관을 역치와 비교하는 동작을 포함하는 시스템.

청구항 7

제1항에 있어서,
상기 디스플레이는 사이즈를 갖고;
상기 비디오 입력 스트림의 제1 스크린 부분은 상기 디스플레이의 사이즈보다 작은 직사각형 영역인 시스템.

청구항 8

제1항에 있어서,
상기 비디오 입력 스트림의 제1 스크린 부분은 적어도 하나의 에지를 상기 디스플레이의 에지와 공유하는 시스템.

청구항 9

제1항에 있어서,
상기 비디오 입력 스트림의 제1 스크린 부분은 상기 디스플레이와 공통되는 에지가 없는 시스템.

청구항 10

제1항에 있어서,
상기 비디오 출력 스트림을 생성하는 동작은,
상기 제2 콘텐츠의 텍스트를 식별하도록, 상기 제2 콘텐츠를 분석하는 동작,
상기 텍스트가 상기 제1 콘텐츠에 관련됨을 판정하는 동작,
상기 제2 콘텐츠의 텍스트가 상기 제1 콘텐츠에 관련됨을 판정하는 동작에 응답해서, 상기 텍스트에 의거하여 상기 제2 콘텐츠의 적어도 일부를 억제하기 위한 마스크를 결정하는 동작, 및

상기 비디오 출력 스트림의 제2 콘텐츠의 적어도 일부를 억제하도록, 상기 마스크를 상기 제2 콘텐츠에 적용하는 동작을 포함하는 시스템.

청구항 11

제1항에 있어서,

상기 중간 디바이스의 식별자에 액세스하는 동작은,

(i) 상기 중간 디바이스의 EDID(Extended Display Identification Data)에 의거하여 상기 식별자에 액세스하는 동작,

(ii) 유저 인터페이스로부터 수신된 입력에 의거하여 상기 식별자에 액세스하는 동작, 또는

(iii) 상기 중간 디바이스의 IP(internet protocol) 어드레스에 의거하여 상기 식별자에 액세스하는 동작 중 적어도 하나를 포함하는 시스템.

청구항 12

디스플레이 디바이스에서, 콘텐츠 프로바이더로부터의 제1 콘텐츠 및 중간 디바이스로부터의 제2 콘텐츠를 포함하는 비디오 입력 스트림에 액세스하는 단계 - 상기 비디오 입력 스트림의 프레임에서, 상기 제1 콘텐츠는 디스플레이의 제1 스크린 부분에 대응하고 상기 제2 콘텐츠는 상기 디스플레이의 제2 스크린 부분에 대응함 -;

상기 중간 디바이스의 식별자에 액세스하는 단계;

상기 중간 디바이스의 식별자에 액세스하는 단계 후에, 상기 중간 디바이스의 식별자에 의거하여 복수의 템플릿들의 서브셋을 선택하는 단계 - 상기 복수의 템플릿들의 서브셋은 상기 복수의 템플릿들 중 둘 이상의 템플릿을 포함함 -;

상기 복수의 템플릿들의 서브셋을 선택하는 단계 후에, 상기 디스플레이 디바이스의 프로세서를 이용하여, 상기 비디오 입력 스트림의 프레임이 식별된 템플릿에 대응함을 판정하도록, 상기 비디오 입력 스트림을 상기 복수의 템플릿들의 서브셋과 비교하는 단계 - 상기 식별된 템플릿은 상기 제2 스크린 부분이 상기 중간 디바이스로부터의 제2 콘텐츠를 표시하기 위한 오버레이를 포함함을 지시함 -;

상기 식별된 템플릿에 액세스하는 단계;

상기 비디오 입력 스트림의 프레임이 상기 식별된 템플릿에 대응한다는 판정에 응답해서, 상기 제2 스크린 부분에 대한 제2 콘텐츠 및 상기 제1 스크린 부분에 대한 제3 콘텐츠를 포함하는 비디오 출력 스트림을 생성하는 단계 - 상기 제3 콘텐츠는 상기 비디오 입력 스트림의 제1 콘텐츠와 상이함 -; 및

상기 비디오 출력 스트림이 상기 디스플레이 디바이스의 디스플레이 상에 제시되게 하는 단계를 포함하고,

상기 비디오 출력 스트림을 생성하는 단계는,

상기 식별된 템플릿에 의거하여, 상기 제2 스크린 부분의 사이즈 및 위치에 대한 복수의 좌표들을 판정하는 단계, 및

상기 비디오 입력 스트림에 액세스하는 단계 동안, 상기 복수의 좌표들을 이용하여, 상기 제2 스크린 부분에서 상기 제2 콘텐츠 및 상기 제1 스크린 부분에서 상기 제3 콘텐츠를 갖는 비디오 출력 스트림을 동적으로 생성하는 단계를 포함하는 방법.

청구항 13

제12항에 있어서,

상기 비디오 입력 스트림의 핑거프린트를 생성하는 단계;

상기 핑거프린트에 의거하여 상기 콘텐츠 프로바이더에 의해 제공된 비디오 콘텐츠를 식별하는 단계; 및

상기 콘텐츠 프로바이더에 의해 제공된 비디오 콘텐츠의 식별에 의거하여 대체 비디오 콘텐츠를 식별하는 단계를 더 포함하고;

상기 생성된 비디오 출력 스트림의 제3 콘텐츠는 상기 식별된 대체 비디오 콘텐츠의 적어도 일부를 포함하는 방

법.

청구항 14

제12항에 있어서,

상기 콘텐츠 프로바이더는 케이블 TV 스테이션인 방법.

청구항 15

제12항에 있어서,

상기 중간 디바이스는 셋톱 박스인 방법.

청구항 16

제12항에 있어서,

상기 디스플레이 디바이스는 스마트 TV인 방법.

청구항 17

제12항에 있어서,

상기 비디오 입력 스트림을 상기 식별된 템플릿과 비교하는 단계는,

상기 비디오 입력 스트림의 프레임을 다운샘플링하는 단계;

상기 다운샘플링된 프레임과 상기 식별된 템플릿 사이의 상호상관을 판정하는 단계; 및

상기 상호상관을 역치와 비교하는 단계를 포함하는 방법.

청구항 18

제12항에 있어서,

상기 중간 디바이스의 식별자에 액세스하는 단계는,

(i) 상기 중간 디바이스의 EDID(Extended Display Identification Data)에 의거하여 상기 식별자에 액세스하는 단계,

(ii) 유저 인터페이스로부터 수신된 입력에 의거하여 상기 식별자에 액세스하는 단계, 또는

(iii) 상기 중간 디바이스의 IP(internet protocol) 어드레스에 의거하여 상기 식별자에 액세스하는 단계 중 적어도 하나를 포함하는 방법.

청구항 19

머신의 하나 이상의 프로세서에 의한 실행 시, 상기 머신에게 동작들을 행하게 하는 명령어들을 저장하는 비일시적인 머신 판독 가능한 매체로서,

상기 동작들은,

콘텐츠 프로바이더로부터의 제1 콘텐츠 및 중간 디바이스로부터의 제2 콘텐츠를 포함하는 비디오 입력 스트림에 액세스하는 동작 - 상기 비디오 입력 스트림의 프레임에서, 상기 제1 콘텐츠는 디스플레이의 제1 스크린 부분에 대응하고 상기 제2 콘텐츠는 상기 디스플레이의 제2 스크린 부분에 대응함 -;

상기 중간 디바이스의 식별자에 액세스하는 동작;

상기 중간 디바이스의 식별자에 액세스하는 동작 후에, 상기 중간 디바이스의 식별자에 의거하여 복수의 템플릿들의 서브셋을 선택하는 동작 - 상기 복수의 템플릿들의 서브셋은 상기 복수의 템플릿들 중 둘 이상의 템플릿을 포함함 -;

상기 복수의 템플릿들의 서브셋을 선택하는 동작 후에, 상기 비디오 입력 스트림의 프레임이 식별된 템플릿에 대응함을 판정하도록 상기 비디오 입력 스트림을 상기 복수의 템플릿들의 서브셋과 비교하는 동작

— 상기 식별된 템플릿은 상기 제2 스크린 부분이 상기 중간 디바이스로부터의 제2 콘텐츠를 표시하기 위한 오버레이를 포함함을 지시함 —;

상기 식별된 템플릿에 액세스하는 동작;

상기 비디오 입력 스트림의 프레임이 상기 식별된 템플릿에 대응한다는 판정에 응답해서, 상기 제2 스크린 부분에 대한 제2 콘텐츠 및 상기 제1 스크린 부분에 대한 제3 콘텐츠를 포함하는 비디오 출력 스트림을 생성하는 동작 — 상기 제3 콘텐츠는 상기 비디오 입력 스트림의 제1 콘텐츠와 상이함 —; 및

상기 비디오 출력 스트림이 상기 디스플레이 상에 제시되게 하는 동작을 포함하고,
상기 비디오 출력 스트림을 생성하는 동작은,

상기 식별된 템플릿에 의거하여, 상기 제2 스크린 부분의 사이즈 및 위치에 대한 복수의 좌표들을 판정하는 동작, 및

상기 비디오 입력 스트림에 액세스하는 동작 동안, 상기 복수의 좌표들을 이용하여, 상기 제2 스크린 부분에서 상기 제2 콘텐츠 및 상기 제1 스크린 부분에서 상기 제3 콘텐츠를 갖는 비디오 출력 스트림을 동적으로 생성하는 동작을 포함하는 비일시적인 머신 판독 가능한 매체.

발명의 설명

기술 분야

[0001] 본 출원은, 그 전체가 참조에 의해 본 명세서에 포함되는, "Dynamic Video Overlays"라는 명칭의 2015년 12월 16일에 출원된 미국 가특허출원 제62/268,410호에 대한 우선권을 주장한다.

[0002] 본 명세서에 개시되는 주제는 일반적으로 비디오 스트림의 처리에 관한 것이다. 구체적으로는, 본 개시는 오버레이된 비디오의 제시(presentation)를 일으키는 시스템 및 방법을 다룬다.

배경 기술

[0003] 일반적으로, 사람들은, 콘텐츠 소스로부터의 전송을 수신하는 디바이스를 통해, 텔레비전 쇼, 광고, 영화, 비디오 클립 등의 비디오 콘텐츠를 시청한다. 예를 들면, 브로드캐스터(예를 들면, HBO[®] 또는 CNN[®]), 웹 서버(예를 들면, YouTube[®]), 피어-투-피어 소스(예를 들면, 다른 디바이스), 또는 다른 콘텐츠 소스 또는 다른 방식의 것은, 텔레비전 및 연관 셋톱 박스, 컴퓨팅 및/또는 모바일 디바이스 및 연관 미디어 플레이어 또는 브라우저 등과 같이, 비디오 콘텐츠를 제시할 수 있는 다양한 디바이스에 비디오 콘텐츠를 전송한다.

발명의 내용

[0004] 일부 실시예는 예를 들어 나타나 있지만, 첨부 도면의 형태에 한정되는 것은 아니다.

도면의 간단한 설명

- [0005] 도 1은 비디오를 동적으로 오버레이하기 위한 예시적인 실시예에서의 네트워크 환경을 나타내는 블록도.
도 2는 일부 예시적인 실시예에 따른 비디오 식별 시스템 및 쿼리 핑거프린트 생성기의 컴포넌트를 나타내는 블록도.
도 3은 일부 예시적인 실시예에 따른 클라이언트 디바이스 및 대체 콘텐츠 서버의 컴포넌트를 나타내는 블록도.
도 4는 예시적인 실시예에서 동적 비디오 오버레이를 나타내는 표시도.
도 5는 예시적인 실시예에서 동적 비디오 오버레이를 나타내는 표시도.
도 6은 예시적인 실시예에서 동적 비디오 오버레이를 나타내는 표시도.
도 7은 예시적인 실시예에서 동적 비디오 오버레이를 나타내는 표시도.
도 8은 일부 예시적인 실시예에서 비디오 오버레이를 동적으로 제공하기 위한 방법을 나타내는 플로우차트.
도 9는 일부 예시적인 실시예에서 비디오 오버레이를 동적으로 제공하기 위한 방법을 나타내는 플로우차트.

도 10은 일부 예시적인 실시예에서 비디오 오버레이를 동적으로 제공하기 위한 방법을 나타내는 플로우차트.

도 11은 일부 예시적인 실시예에서 비디오 오버레이를 동적으로 제공하기 위한 방법을 나타내는 플로우차트.

도 12는 일부 예시적인 실시예에서 비디오 오버레이를 동적으로 제공하기 위한 방법을 나타내는 플로우차트.

도 13은 일부 예시적인 실시예에서 비디오 오버레이를 동적으로 제공하기 위한 방법을 나타내는 플로우차트.

도 14는 일부 예시적인 실시예에서 비디오 오버레이를 동적으로 제공하기 위한 방법을 나타내는 플로우차트.

도 15는 일부 예시적인 실시예에서 비디오 오버레이를 동적으로 제공하기 위한 방법을 나타내는 플로우차트.

도 16은 예시적인 실시예에서 비디오 오버레이를 동적으로 제공하기 위한 데이터 구조를 나타내는 블록도.

도 17은 일부 예시적인 실시예에 따라 머신 판독 가능한 매체로부터 명령어를 판독하고 본 명세서에서 설명하는 방법들 중 어느 하나 이상을 행할 수 있는 머신의 컴포넌트를 나타내는 블록도.

발명을 실시하기 위한 구체적인 내용

- [0006] 비디오 오버레이를 동적으로 제공하기 위한 예시적인 방법 및 시스템을 설명한다. 일부 예시적인 실시예에서, 클라이언트 디바이스(예를 들면, 스마트 텔레비전(TV))은, 비디오 스트림 상에 오버레이된 정보를 제시하는 셋톱 박스를 식별한다. 식별된 셋톱 박스 및 셋톱 박스로부터 수신된 비디오 데이터에 의거하여, 제시되는 정보에 대응하는 템플릿을 선택한다. 템플릿에 의거하여, 원래의 비디오 스트림이 셋톱 박스에 의해 제시된 방식을 모방하여, 대체 비디오 스트림이 정보와 함께 제시된다. 일부 예시적인 실시예에서, 대체 비디오 스트림은 원래의 비디오 스트림에 의거하여 선택된다. 예를 들면, 클라이언트 디바이스는, 원래의 비디오 콘텐츠의 프레임의 쿼리 핑거프린트를 생성할 수 있다. 클라이언트 디바이스는 생성된 쿼리 핑거프린트를 사용해서 기지(既知)의 레퍼런스 핑거프린트들의 데이터베이스에 질의하고, 비디오 콘텐츠의 프레임의 쿼리 핑거프린트가 기지의 레퍼런스 핑거프린트와 매칭됨을 판정하고, 핑거프린트의 매칭에 의거하여 비디오 콘텐츠를 식별한다. 식별된 비디오 콘텐츠에 의거하여, 대체 비디오 스트림이 선택된다. 예를 들면, 특정한 전국 커머셜을 나타내는 원래의 비디오 스트림을 대체하도록, 특정한 로컬 커머셜을 나타내는 대체 비디오 스트림이 선택될 수 있다.
- [0007] 핑거프린트는 단일 프레임 핑거프린트 또는 멀티 프레임 핑거프린트일 수 있다. 멀티 프레임 핑거프린트는, 프레임의 상대적인 타이밍과 함께, 단일 프레임 핑거프린트들의 집합이다. "핑거프린트"라는 용어는, 단일 프레임과 멀티 프레임 핑거프린트 모두를 의미하는데 사용된다. "컴포넌트"라는 용어는, 혼란이 생길 가능성이 있을 경우에, 멀티 프레임 핑거프린트 내의 단일 프레임 핑거프린트를 의미하는데 사용된다.
- [0008] 일부 예시적인 실시예에서, 클라이언트 디바이스는, 식별된 템플릿에 의거하여 특정한 위치 및 사이즈로 대체 비디오 스트림을 제시한다. 예를 들면, 셋톱 박스는, 원래의 비디오 스트림을 축소 사이즈의 픽처-인-픽처(picture-in-picture)로서 제시할 수 있다. 따라서, 셋톱 박스의 거동을 모방하도록, 대체 비디오 스트림은 원래의 비디오 스트림과 동일한 사이즈 및 위치에 제시된다.
- [0009] 추가적인 예시적인 실시예에서, 클라이언트 디바이스는, 식별된 템플릿에 의거하여 대체 비디오 스트림의 특정한 부분을 제시한다. 예를 들면, 셋톱 박스는, 원래의 비디오 스트림을 풀 스크린 화상으로서 제시해도 되지만, 스트림의 바닥부를 정보 바로 덮을 수 있다. 따라서, 셋톱 박스의 거동을 모방하도록, 대체 비디오 스트림의 상부만이 제시된다.
- [0010] 추가적인 예시적인 실시예에서, 클라이언트 디바이스는, 식별된 템플릿에 의거하여, 대체 비디오 스트림의 특정한 부분의 알파(투명) 블렌드를 행한다. 예를 들면, 셋톱 박스는, 원래의 비디오 스트림 상에 중첩된 메뉴를 제시할 수 있다. 클라이언트 디바이스는 합성 화상으로부터 메뉴를 추출하고, 템플릿에 정의된 알파 값을 사용해서 대체 비디오 스트림 상에 메뉴를 중첩시킬 수 있다.
- [0011] 일부 예시적인 실시예에서, 클라이언트 디바이스는, 식별된 템플릿에 의거하여 특정한 속도로 대체 비디오 스트림을 제시한다. 예를 들면, 셋톱 박스는, 빨리 감기 모드로 원래의 비디오 스트림을 제시할 수 있다. 따라서, 셋톱 박스의 거동을 모방하도록, 대체 비디오 스트림은 빨리 감기 모드로 제시된다.
- [0012] 이하의 설명에서, 설명을 위해, 예시적인 실시예의 완전한 이해를 제공하도록, 다수의 특정한 상세를 설명한다. 그러나, 통상의 기술자에게는, 본 주제가 이들 특정한 상세 없이 실시될 수 있음이 명백할 것이다.
- [0013] 도 1은, 일부 예시적인 실시예에 따른 동적 비디오 오버레이에 적합한 네트워크 환경(100)을 나타내는 네트워크 도면이다. 네트워크 환경(100)은, 브로드캐스터, 웹 서버, 케이블 TV 스테이션 등과 같은 콘텐츠 소스(110)로

부터 비디오 및 다른 멀티미디어 콘텐츠를 수신하는 와칭(watching) 스테이션(120)을 포함할 수 있다. 예를 들면, 콘텐츠 소스(110)는, 특히, TV 채널을 통해 와칭 스테이션(120)에 미디어를 스트리밍 또는 전송하는 TV 스테이션 또는 TV 네트워크 등의 브로드 캐스터일 수 있거나, 및/또는 네트워크(170)를 통해 와칭 스테이션(120)에 미디어를 스트리밍 또는 전송하는 웹사이트 등의 웹 서비스일 수 있다. 와칭 스테이션(120)은, 콘텐츠 소스(110)로부터 수신된 비디오 콘텐츠의 레퍼런스 핑거프린트를 생성하는 레퍼런스 핑거프린트 생성기(130)를 포함한다.

[0014] 하나 이상의 셋톱 박스(140)는, 또한, 브로드캐스트 채널 및/또는 네트워크(170) 등을 통해, 콘텐츠 소스(110)로부터 비디오 및 다른 멀티미디어 콘텐츠를 수신할 수도 있다. 셋톱 박스(140)는, 수신된 콘텐츠를 클라이언트 디바이스(150)에 송신하기 전에 변경할 수 있다. 클라이언트 디바이스(150)는 비디오 및/또는 다른 멀티미디어 콘텐츠의 스트림을 수신 및 제시할 수 있는 임의의 디바이스이다(예를 들면, TV, 제2 셋톱 박스, 랩톱 또는 다른 퍼스널 컴퓨터(PC), 태블릿 또는 다른 모바일 디바이스, 디지털 비디오 레코더(DVR), 또는 게임 디바이스).

[0015] 일부 예시적인 실시예에서, 셋톱 박스(140)는 비디오 콘텐츠의 입력 스트림을 수신하고, 입력 스트림을 처리함으로써 비디오 콘텐츠의 출력 스트림을 생성하도록 구성된 튜너를 포함한다. 셋톱 박스(140)는, 멀티채널 비디오 프로그래밍 디스트리뷰터에 의해 채워지는 지상파 방송, 케이블, 및/또는 위성 방송 네트워크 등의 비디오 콘텐츠 배신(distribution) 네트워크를 통해 비디오 콘텐츠에 액세스하도록 튜너, 디코더 등을 구비한 디바이스일 수 있다. 추가적으로 또는 대안으로, 셋톱 박스(140)는, 인터넷 비디오 스트리밍 서비스에 액세스하기 위한 디바이스에 의해 채워지는 광역 네트워크(예를 들면, 인터넷)를 통해 비디오 콘텐츠에 액세스하도록 네트워크 어댑터, 디코더 등을 구비한 디바이스일 수 있다. 비디오 콘텐츠의 출력 스트림은 다수의 컴포넌트로 구성되어도 된다. 예를 들면, 비디오 콘텐츠의 출력 스트림은, 입력 스트림에 대응하는 출력 스트림의 제1 영역에서의 제1 컴포넌트와, 셋톱 박스(140)에 의해 생성되는 비디오 데이터에 대응하는 출력 스트림의 제2 영역에서의 제2 컴포넌트를 포함할 수 있다. 제2 컴포넌트는, 셋톱 박스(140)를 위한 메뉴, 셋톱 박스(140)로부터의 정보 메시지, 또는 다른 디바이스(예를 들면, 전화 또는 보안 알람)에 의해 생성되고 제시를 위해 셋톱 박스(140)에 송신된 정보 메시지를 포함할 수 있다. 셋톱 박스(140)는, 출력 비디오 콘텐츠를 클라이언트 디바이스(150)와 연관된 유저에게 제시하도록, 클라이언트 디바이스(150)의 디스플레이에 의해 사용 가능한 신호(예를 들면, 디지털 또는 아날로그 신호)를 출력한다. 클라이언트 디바이스(150)의 관점에서, 셋톱 박스(140)에 의해 출력된 신호는 비디오 입력 스트림이다.

[0016] 클라이언트 디바이스(150)는 또한, 비디오 콘텐츠의 처리된 스트림을 표시하도록 구성된 디스플레이 또는 다른 유저 인터페이스를 포함할 수 있다. 디스플레이는 플랫 패널 스크린, 플라스마 스크린, 발광 다이오드(LED) 스크린, 음극선관(CRT), 액정 디스플레이(LCD), 프로젝터 등일 수 있다. 일부 예시적인 실시예에서, 셋톱 박스(140) 및 클라이언트 디바이스(150)는 단일 디바이스에 통합된다.

[0017] 네트워크(170)는, 유선 네트워크, 무선 네트워크(예를 들면, 모바일 네트워크) 등 디바이스들간의 통신을 가능하게 하는 임의의 네트워크일 수 있다. 네트워크(170)는, 프라이빗 네트워크(예를 들면, 케이블 TV 네트워크 또는 위성 TV 네트워크), 공중 네트워크(예를 들면, 공중파 브로드캐스트 채널 또는 인터넷) 등을 구성하는 하나 이상의 부분을 포함할 수 있다.

[0018] 일부 예시적인 실시예에서, 비디오 식별 시스템(190)은 네트워크(170)를 통해 와칭 스테이션(120) 및 클라이언트 디바이스(150)와 통신한다. 비디오 식별 시스템(190)은, 비디오 콘텐츠 내의 프레임 또는 프레임들의 블록의 핑거프린트 등 클라이언트 디바이스(150)의 쿼리 핑거프린트 생성기(160)에 의해 생성된 쿼리 핑거프린트를 수신하고, 쿼리 핑거프린트와 하나 이상의 레퍼런스 핑거프린트를 매칭함으로써 비디오 콘텐츠를 식별하도록 와칭 스테이션(120)의 레퍼런스 핑거프린트 생성기(130)에 의해 생성된 기지의 레퍼런스 핑거프린트의 인덱스를 질의할 수 있다.

[0019] 비디오 콘텐츠의 식별 시에, 비디오 식별 시스템(190)은, 비디오 콘텐츠와 연관된 대체 콘텐츠(예를 들면, 대체 프로그래밍, 대체 커머셜 등)에 대한 식별자를 클라이언트 디바이스(150)에 반환할 수 있다. 대체 콘텐츠는 대체 콘텐츠 서버(180)에 저장될 수 있다. 식별자를 사용해서, 클라이언트 디바이스(150)는, 대체 콘텐츠 서버(180)로부터 대체 콘텐츠에 액세스하고, 대체 콘텐츠를 셋톱 박스(140)로부터 수신된 비디오 콘텐츠와 오버레이할 수 있다. 예를 들면, 클라이언트 디바이스(150)는, 브로드캐스트 채널에 대한 대체 커머셜 등, 대체 콘텐츠 서버(180)로부터의 대체 콘텐츠에 액세스 및 제시할 수 있다.

[0020] 도 1에 나타난 머신, 데이터베이스, 또는 디바이스의 어느 것은, 해당 머신에 대해 본 명세서에서 설명되는 기

능을 행하기 위해 특수 목적용 컴퓨터가 되도록 소프트웨어에 의해 변형(예를 들면, 구성 또는 프로그래밍)된 범용 컴퓨터 내에서 실시될 수 있다. 예를 들면, 본 명세서에 기재된 방법들 중의 어느 하나 이상을 실시할 수 있는 컴퓨터 시스템이 도 16과 관련하여 이하에 설명된다. 본 명세서에서 사용되는 "데이터베이스"는 데이터 스토리지 리소스이고, 텍스트 파일, 테이블, 스프레드시트, 관계형 데이터베이스, 문서 스토어, 키-밸류 스토어, 트리플 스토어, 또는 그 임의의 적절한 조합 등과 같이 구성된 데이터를 저장할 수 있다. 또한, 도 1에 나타난 임의의 2개 이상의 머신은 단일 머신으로 조합될 수 있고, 임의의 단일 머신에 대해 본 명세서에서 설명한 기능은 복수의 머신들간에 세분화될 수 있다.

[0021] 또한, 모듈, 시스템, 및/또는 생성기 중 어느 것은, 도 1에 나타나는 머신, 데이터베이스, 또는 디바이스 중 어느 것에 위치되어도 된다. 예를 들면, 비디오 식별 시스템(190)은 쿼리 핑거프린트 생성기(160) 및 클라이언트 디바이스(150)로부터의 비디오 콘텐츠의 프레임들을 포함할 수 있고, 다른 구성 중에서도 특히, 포함된 쿼리 핑거프린트 생성기(160)를 사용해서 쿼리 핑거프린트를 생성할 수 있다.

[0022] 도 2는, 일부 예시적인 실시예에 따른 쿼리 핑거프린트 생성기(160), 레퍼런스 핑거프린트 생성기(130), 및 비디오 식별 시스템(190)의 컴포넌트들을 나타내는 블록도(200)이다.

[0023] 클라이언트 디바이스(150)의 쿼리 핑거프린트 생성기(160)는, (예를 들면, 버스, 공유 메모리, 또는 스위치를 통해) 서로 통신하도록 구성된 패치 선택 모듈(210) 및 값 계산 모듈(220)을 포함한다. 비디오 식별 시스템(190)은, (예를 들면, 버스, 공유 메모리, 또는 스위치를 통해) 모두 서로 통신하도록 구성된 인덱스 모듈(230), 핑거프린트 매치 모듈(240), 및 식별 모듈(250)을 포함한다. 와칭 스테이션(120)의 레퍼런스 핑거프린트 생성기(130)는, (예를 들면, 버스, 공유 메모리, 또는 스위치를 통해) 서로 통신하도록 구성된 패치 선택 모듈(260) 및 값 계산 모듈(270)을 포함한다. 본 명세서에서 설명되는 모듈들 중의 하나 이상은, 하드웨어(예를 들면, 머신의 프로세서, FPGA(field-programmable gate array), 또는 ASIC(application-specific integrated circuit)), 또는 하드웨어와 소프트웨어의 조합(예를 들면, 소프트웨어에 의해 구성된 프로세서)을 이용하여 실시될 수 있다. 또한, 이들 모듈들 중의 임의의 2개 이상이 단일의 모듈로 조합되어도 되고, 단일의 모듈에 대해 본 명세서에서 설명되는 기능이 복수의 모듈간에 세분화되어도 된다.

[0024] 일부 예시적인 실시예에서, 쿼리 핑거프린트 생성기(160)는, 클라이언트 디바이스(150)에서 캡처된 비디오 콘텐츠의 하나 이상의 프레임의 쿼리 핑거프린트를 생성하도록 구성 및/또는 프로그래밍된다. 예를 들면, 쿼리 핑거프린트 생성기(160)는, 비디오 콘텐츠 내의 하나 이상의 프레임의 하라이크 특징(Haar-like feature), 영역, 부분, 및/또는 다른 측면 등 패치들의 값을 계산할 수 있다. 예를 들면, 패치는, 다양한 서로 다른 지오메트리, 하라이크 특징 등을 갖는 프레임의 부분일 수 있다. 일부 예시적인 실시예에서, 일부 또는 모든 캡처된 패치는 각각 서로 다른 스케일을 갖고, 프레임 내의 서로 다른 위치에 있을 수 있다. 수신된 비디오 콘텐츠의 패치에 대한 핑거프린트를 보고함으로써, 쿼리 핑거프린트 생성기(160)는 각 패치의 콘텐츠를 판정할 수 있다. 따라서, 수신된 비디오가 서로 다른 소스(예를 들면, 브로드캐스터로부터의 입력 비디오 스트림 및 셋톱 박스(140)에 의해 생성된 오버레이 비디오 스트림)로부터의 다수의 컴포넌트를 포함할 경우, 다수의 컴포넌트에 대응하는 영역이 식별될 수 있다.

[0025] 쿼리 핑거프린트 생성기(160)(및 레퍼런스 핑거프린트 생성기(130))는, 콘텐츠 내의 프레임들로부터 비디오 콘텐츠를 식별하기 위한 핑거프린트를 생성 및/또는 작성한다. 전형적으로는, 클라이언트 디바이스(150)가 수신한 비디오 콘텐츠는 서로 다른 포맷 및 샘플 레이트이고, 쿼리 핑거프린트 생성기(160)는, 비디오 콘텐츠의 프레임들의 일부 또는 모두에 대해, 스케일 독립적이고 서로 다른 압축 아티팩트들에 대해 로버스트성인 프레임마다의 쿼리 핑거프린트를 생성한다. 일부 예시적인 실시예에서, 쿼리 핑거프린트 생성기(160)는 프레임 각각의 쿼리 핑거프린트를 조합하여, 비디오 콘텐츠의 프레임들의 블록(예를 들면, 다수의 프레임들)의 쿼리 핑거프린트를 생성할 수 있다. 비디오 식별 시스템(190)은 수백만의 쿼리 핑거프린트 생성기(160)로부터의 비디오 식별 요구에 응답할 수 있다.

[0026] 쿼리 핑거프린트 생성기(160)는, 비디오 콘텐츠 내의 프레임 또는 프레임들의 표시 영역과 연관된 패치들 등, 비디오 콘텐츠의 다수의 패치를 선택하도록 구성 및/또는 프로그래밍된 패치 선택 모듈(210)을 포함할 수 있다. 마찬가지로, 레퍼런스 핑거프린트 생성기(130)는, 비디오 콘텐츠 내의 프레임 또는 프레임들의 표시 영역과 연관된 패치들 등, 비디오 콘텐츠의 다수의 패치를 선택하도록 구성 및/또는 프로그래밍된 패치 선택 모듈(260)을 포함할 수 있다.

[0027] 패치들은, 프레임을 그리드(예를 들면, 2×2 그리드, 4×3 그리드, 또는 4×4 그리드)로 분할함으로써 생성될 수 있다. 패치들은 겹쳐질 수 있다. 예를 들면, 20개의 패치, 즉 프레임의 사분면에 대응하는 4개의 큰 패치,

및 프레임을 4×4 그리드로 분할하여 생기는 16개의 작은 패치를 사용할 수 있다. 다른 예로서, 5개의 패치, 즉 프레임의 사분면에 대응하는 4개의 큰 패치, 및 프레임의 중앙에 위치하는 동일한 사이즈의 제5 패치를 사용할 수 있다. 일부 예에서, 프레임의 부분들, 프레임의 외측 경계를 포함하지 않는 프레임의 중앙 부분 등의 그리드로 분할함으로써, 패치를 생성할 수 있다. 따라서, 패치들은, 프레임의 전체보다 작게 누적되게 걸쳐 있어도 된다.

[0028] 쿼리 핑거프린트 생성기(160)는 또한, 함께 에어리어 테이블 또는 영역의 직사각형 에어리어에서의 값의 합계를 생성하는 다른 데이터 구조를 사용해서 값을 계산하는 기술 등의 적분 화상 기술을 사용해서, 선택된 다수의 패치의 각각의 값을 계산하도록 구성 또는 프로그래밍된 값 계산 모듈(220)을 포함할 수 있다. 마찬가지로, 레퍼런스 핑거프린트 생성기(130)는, 함께 에어리어 테이블 또는 영역의 직사각형 에어리어에서의 값의 합계를 생성하는 다른 데이터 구조를 사용해서 값을 계산하는 기술 등의 적분 화상 기술을 사용해서, 선택된 다수의 패치의 각각의 값을 계산하도록 구성 또는 프로그래밍된 값 계산 모듈(270)을 포함할 수 있다.

[0029] 예를 들면, 패치 선택 모듈(210, 260)은, 프레임 또는 프레임들의 영역의 오브젝트 검출에서 일반적으로 사용되는 하라이크 특징 등 패치를 선택할 수 있다. 값 계산 모듈(220, 270)은 하라이크 특징을 이용하여, 오브젝트의 상대적인 사이즈에 상관없이, 프레임의 시각 화상 내의 오브젝트 등, 프레임 내의 오브젝트에 대한 동일한 값을 생성 또는 계산할 수 있다. 예를 들면, 값 계산 모듈(220, 270)은 가우시안 필터들 및 박스 필터(예를 들어, 프레임의 영역의 평균)를 사용한 그 도함수를 근사시킴으로써 값을 생성할 수 있으며, 여기서 가우시안 필터들의 도함수는 박스 필터들의 차분을 구함으로써 생성된다.

[0030] 쿼리 핑거프린트 생성기(160)는, 값 계산 모듈(220)을 통해, 비디오 콘텐츠의 프레임들의 표시 영역의 서로 다른 스케일 및 서로 다른 위치에서 패치 또는 하라이크 특징의 값을 계산함으로써 쿼리 핑거프린트를 생성할 수 있다. 레퍼런스 핑거프린트 생성기(130)는, 값 계산 모듈(270)을 통해, 비디오 콘텐츠의 프레임들의 표시 영역의 서로 다른 스케일 및 서로 다른 위치에서 패치 또는 하라이크 특징의 값을 계산함으로써 레퍼런스 쿼리 핑거프린트를 생성할 수 있다.

[0031] 일부 예에서, 패치 선택 모듈(210, 260)은, 프레임 또는 프레임들의 영역을 분할하는 그리드로부터의 패치를 포함하는 패치를 선택할 수 있다. 다음으로, 값 계산 모듈(220, 270)은, 각 패치 내의 픽셀값의 합계값, 중앙값 등과 같은 패치마다의 특성값을 계산할 수 있다. 또한, 값 계산 모듈(220, 270)은 그 특성값들간의 차분 및/또는 그 특성값들의 선형 조합을 계산할 수 있다. 일부 경우에, 얻어진 차분 및/또는 선형 조합을 사용해서, 역치와의 비교 등에 의해 2진값(예를 들면, 0 또는 1)을 결정할 수 있다. 따라서, 핑거프린트 생성기(160, 130)는, 계산 모듈(220, 260)을 통해, 비디오 콘텐츠의 프레임 또는 프레임들의 다수의 패치의 특성값들의 차분 및/또는 선형 조합에 의거하는 2진값을 계산함으로써 핑거프린트를 생성할 수 있다.

[0032] 비디오 식별 시스템(190)은, 모듈 중에서 특히, 쿼리 핑거프린트를 기지의 레퍼런스 핑거프린트와 매칭하도록 구성 및/또는 프로그래밍된 인덱스 모듈(230), 핑거프린트 매치 모듈(240), 및 식별 모듈(250)을 포함한다.

[0033] 일부 예시적인 실시예에서, 인덱스 모듈(230)은, 와칭 스테이션(120) 등의 레퍼런스 디바이스에서 캡처된 비디오 콘텐츠의 기지의 레퍼런스 핑거프린트의 데이터베이스에 질의하도록 구성 및/또는 프로그래밍된다. 예를 들면, 인덱스 모듈(230)은, 와칭 스테이션(120)의 데이터베이스 내에 저장된 인덱스, 및 비디오 식별 시스템(190) 내에 저장된 인덱스 등을 질의할 수 있다.

[0034] 예를 들면, 인덱스 모듈(230)은, 기지의 레퍼런스 핑거프린트의 양자화된 패치값의 인덱스를 질의하도록 구성될 수 있다. 인덱스 모듈(230)은, 레퍼런스 핑거프린트의 단일의 프레임과 연관된 1비트, 8비트, 16비트, 및/또는 24비트 수의 인덱스를 질의할 수 있다. 인덱스 모듈(230)은, 레퍼런스 핑거프린트의 하나 이상의 패치값을 양자화함으로써 수를 도출할 수 있다. 예를 들면, 인덱스 모듈(230)은 절대 합, 로그 정규화 등에 의해 핑거프린트값을 정규화해도 된다.

[0035] 인덱스 모듈(230)은, 프레임들의 큰 영역들의 최선 또는 보다 양호한 상관값을 사용해서, 레퍼런스 핑거프린트의 일부 또는 모든 프레임을 인덱싱할 수 있다. 인덱스 모듈(230)은, 특히 최선의 상관값과 연관된 피치가 프레임의 나머지 패치들을 대표할 수 있으므로, 전체 프레임 패치들의 높은 상관의 값을 사용하여 인덱스를 생성할 수 있다. 높은 상관의 값은 샘플 데이터 세트를 사용하여 미리 결정될 수 있다.

[0036] 예를 들면, 프레임의 3개의 영역이 최선의 상관값일 경우, 인덱스 모듈(230)은, 최대 및 최소 한도를 갖는 등간격의 히스토그램으로 배치하는 등 해서, 3개의 값의 각각을 8비트로 양자화하고, 그에 의해 24비트 수를 생성한다. 다음으로, 인덱스 모듈(230)은 24비트 수의 레퍼런스 핑거프린트 인덱스를 이용하여, 프레임들을 신속히

록업 및/또는 식별할 수 있다. 일부 예시적인 실시예에서, 퍼지 검색이 행해진다. 퍼지 검색은, 부정확한 매치들을 찾아낼 수 있다. 예를 들면, 3개의 8비트 값들에 대한 퍼지 매치는, 매칭 값들이 정확히 매칭되거나, 또는 대응하는 값의 하나 내에 있을 경우에 발견될 수 있다. 이 예에서, 3바이트의 각각에 대해 3개의 매칭 값을 갖고 핑거프린트를 매칭하고자 할 경우 파싱하는 인덱스마다 27개의 순열이 있다. 마찬가지로, 4개의 8비트 값이 동일한 정도의 퍼지 매칭으로 사용될 경우(즉, 값마다 최대 거리가 1로), 인덱스마다 81개의 순열이 있다. 보다 큰 정도 및/또는 보다 큰 값은, 가능한 순열의 수를 증가시킨다.

[0037] 다른 예로서, 인덱스 모듈(230)은, 각 값을 역치와 비교하는 것 등에 의해, 3개의 값의 각각을 1비트로 양자화하고, 그에 의해 3비트 수를 생성할 수 있다. 인덱스 모듈(230)은, 3비트 수의 레퍼런스 핑거프린트 인덱스를 이용해서, 프레임을 신속히 록업 및/또는 식별할 수 있고, 이는, 핑거프린트와의 매칭 시도 시에 비교할 인덱스마다 8개의 값밖에 없기 때문이다.

[0038] 다른 예시적인 실시예에서, 32의 영역이 사용되고, 각 영역은 단일의 비트에 의해 나타난다. 32비트값은 키를 생성하도록 해시되어도 된다. 예를 들면, 256의 해시값이 사용될 경우, 32비트값의 상위 8비트가 해시로서 사용될 수 있다. 핑거프린트를 록업하기 위해, 동일한 해시값을 갖는 핑거프린트들의 잠재적인 매칭이 반복되어 비교된다.

[0039] 일부 예시적인 실시예에서, 핑거프린트 매치 모듈(240)은, 쿼리 핑거프린트가 적어도 하나의 기지의 레퍼런스 핑거프린트와 매칭됨을 판정하도록 구성 및/또는 프로그래밍된다. 예를 들면, 핑거프린트 매치 모듈(240)은, 쿼리 핑거프린트와 기지의 레퍼런스 핑거프린트의 적어도 하나 사이의 유사도가, 타니모토(Tanimoto) 거리 측정, 맨하튼(Manhattan) 거리 측정, 및/또는 매칭 화상 또는 다른 시각(visual) 기반의 콘텐츠와 연관된 다른 거리 측정과 연관된 미리 결정된 역치를 만족시킴을 판정함으로써, 쿼리 핑거프린트가 적어도 하나의 기지의 레퍼런스 핑거프린트와 매칭됨을 판정할 수 있다.

[0040] 예를 들면, 핑거프린트 매치 모듈(240)은, 타니모토 또는 맨하튼 거리 측정을 사용하여 쿼리 핑거프린트를 하나 이상의 레퍼런스 핑거프린트와 비교하고, 그 비교가 거리 측정값이 미리 결정된 역치를 만족함(예를 들면, 일정한 거리 역치 내에 있음)을 나타낼 경우 핑거프린트들이 매칭됨을 판정할 수 있다. 물론, 핑거프린트 매치 모듈(240)은, 쿼리 핑거프린트가 레퍼런스 핑거프린트와 매칭되는지의 여부를 판정하기 위해, 특히, 유클리드(Euclidean), 코사인(Cosine), KL-디버전스(KL-Divergence) 및/또는 이타쿠라(Itakura) 거리 측정 기술 등의 다른 매칭 기술을 이용해도 된다.

[0041] 일부 예시적인 실시예에서, 비디오 식별 시스템(190)은, 쿼리 핑거프린트를 레퍼런스 핑거프린트와 매칭시킬 경우에, 핑거프린트들의 다양한 서로 다른 블록 사이즈(예를 들면, 프레임의 수 또는 프레임 레이트)를 이용할 수 있다. 예를 들면, 레퍼런스 핑거프린트는 5fps로 설정되고, 핑거프린트에 대해 8비트값(32바이트/프레임)이 주어지면 약 560KB/h 런타임을 점유하고, 오프셋 오차를 포함할 수 있는 쿼리 핑거프린트는 15fps 이상으로 설정될 수 있다. 이 예에서, 인덱스의 쿼리는, 레퍼런스 핑거프린트 인덱스의 다수의 기준 프레임(예를 들면, 레퍼런스 핑거프린트에 대해 3개의 프레임)을 질의하는 것을 수반할 수 있다. 다른 예로서, 레퍼런스 핑거프린트는 15fps로 설정되고, 쿼리 핑거프린트는 5fps로 설정될 수 있다. 이 예에서, 쿼리 핑거프린트를 레퍼런스 핑거프린트와 비교하는 것은, 쿼리 핑거프린트 프레임 각각을, 레퍼런스 핑거프린트 인덱스의 다수의 레퍼런스 프레임에 비교하는 것을 수반할 수 있다.

[0042] 따라서, 일부 예시적인 실시예에서, 비디오 식별 시스템(190)은, 핑거프린트의 블록 사이즈를 변경함으로써, 쿼리 핑거프린트를 레퍼런스 핑거프린트와 매칭하기 위한 매칭 레이트를 최적화할 수 있다. 예를 들면, 핑거프린트 매치 모듈(240)은, 쿼리 핑거프린트를 1초 레퍼런스 비디오 콘텐츠, 0.5초 레퍼런스 비디오 콘텐츠, 1분 레퍼런스 비디오 콘텐츠 등에 매칭할 수 있다. 일부 경우에, 매치의 정확한 시간이 판정될 수 없을 수 있다. 예를 들면, 비디오가 정지 화상을 포함할 경우, 정지 화상을 나타내는 비디오의 임의의 부분은, 정지 화상 세그먼트의 길이 전체와 매칭될 것이다.

[0043] 상술한 바와 같이, 일부 패치는, 다른 패치와는 다른 스케일을 가질 수 있다. 큰 스케일 패치는, 작은 스케일 패치보다 매치를 식별하는데 중요하다. 예를 들면, 1비트값을 사용하여, 모든 비트가 동등하게 처리되어, 미스 매치의 비트의 수가 역치 아래임에 의거하여 매치가 식별될 수 있다. 다른 예로서, 매치는, 미스매치의 낮은 스케일 비트의 수가 제1 역치 아래이고 미스 매치의 큰 스케일 비트의 수가 제2 역치 아래임에 의거하여 매치가 식별될 수 있다. 제2 역치는 제로일 수 있다.

[0044] 일부 예시적인 실시예에서, 식별 모듈(250)은, 쿼리 핑거프린트가 적어도 하나의 레퍼런스 핑거프린트와 매칭됨

의 판정에 의거하여, 클라이언트 디바이스(150)에서 캡처된 비디오 콘텐츠를 식별하도록 구성 및/또는 프로그래밍된다. 예를 들면, 식별 모듈(250)은, 비디오 콘텐츠의 이름 또는 타이틀, 클라이언트 디바이스(150)에 의해 현재 제시되는 비디오 콘텐츠 내의 위치, 비디오 콘텐츠를 제공하는 채널 또는 브로드캐스터 등을 식별할 수 있다.

[0045] 도 3은, 일부 예시적인 실시예에 따른 클라이언트 디바이스(150) 및 대체 콘텐츠 서버(180)의 컴포넌트들을 나타내는 블록도(300)이다. 클라이언트 디바이스(150)는, 모두 서로 통신하도록 구성된 셋톱 박스 식별자 모듈(310), 비디오 소스 식별자 모듈(320), 및 비디오 오버레이 모듈(330)을 포함한다. 대체 콘텐츠 서버(180)는 비디오 제공 모듈(340) 및 템플릿 제공 모듈(350)을 포함한다.

[0046] 셋톱 박스 식별자 모듈(310)은 클라이언트 디바이스(150)에 연결된 셋톱 박스(140)를 식별한다. 예를 들면, 셋톱 박스(140)와 클라이언트 디바이스(150)가 HDMI(High-Definition Multimedia Interface) 케이블에 의해 연결될 경우, 셋톱 박스(140)의 EDID(Extended Display Identification Data)는 셋톱 박스(140)를 식별하도록 기능할 수 있다.

[0047] 다른 예로서, 셋톱 박스 식별자 모듈(310)은, 사용자가 셋톱 박스(140)를 식별할 수 있는 클라이언트 디바이스(150) 상의 유저 인터페이스를 제시할 수 있다. 이것은, 사용자가 드롭-다운 리스트로부터 제조 및 모델 번호를 선택하거나 그 정보를 텍스트 필드에 입력함으로써 셋톱 박스(140)를 수동으로 식별함으로써 달성될 수 있다. 또한, 셋톱 박스(140)의 특정한 메뉴 옵션을 활성화하도록 유저에게 명령함으로써 유저-상호작용 식별이 달성될 수 있다. 클라이언트 디바이스(150)는, 사용자가 셋톱 박스(140)에 대한 메뉴를 활성화한 후에, 서로 다른 셋톱 박스로부터의 기지의 메뉴의 데이터베이스에 액세스하고, 수신된 비디오 데이터와 비교한다. 매치가 발견되면, 셋톱 박스(140)가 식별되었다. 일부 예시적인 실시예에서, 셋톱 박스(140)를 고유하게 식별하기 위해, 셋톱 박스(140) 상의 다수의 유저 액션이 필요하다. 예를 들면, 제1 메뉴는 가능한 매칭을 80% 컷하도록 기능하고, 제2 메뉴는 가능한 매치를 추가로 80%(즉, 가능한 매치의 원래의 세트의 4%까지) 컷하도록 기능하고, 제3 메뉴는 셋톱 박스(140)를 고유하게 식별하는데 필요할 수 있다.

[0048] 비디오 소스 식별자 모듈(320)은, 클라이언트 디바이스(150)에 제공되는 비디오의 소스를 식별한다. 일부 예시적인 실시예에서, 비디오 소스 식별자 모듈(320)은 쿼리 핑거프린트 생성기(160)에 의해 실시된다.

[0049] 비디오 오버레이 모듈(330)은, 셋톱 박스 식별자 모듈(310)에 의해 식별되는 특정한 셋톱 박스에 의거하여, 셋톱 박스(140)로부터 수신된 비디오 데이터와, 대체 콘텐츠 서버(180)로부터 수신된 비디오 데이터를 조합한다. 예를 들면, 비디오 소스 식별자 모듈(320)에 의한 로컬 비디오 소스의 식별에 의거하여, 로컬 커머셜이 비디오 제공 모듈(340)에 의해 제공될 수 있다. 로컬 커머셜은, 셋톱 박스(140)로부터 수신된 비디오 데이터의 부분과 오버레이되는 픽처-인-픽처로서 표시될 수 있다. 오버레이될 부분은, 셋톱 박스 식별자 모듈(310)에 의한 셋톱 박스(140)의 식별에 의거하여 판정될 수 있다.

[0050] 대체 콘텐츠 서버(180)의 비디오 제공 모듈(340)은, 식별된 비디오 소스에 의거하여, 클라이언트 디바이스(150)에 대체(또는 교체) 비디오 콘텐츠를 제공한다. 예를 들면, 비디오 제공 모듈(340)은, 다수의 지역에 대한 로컬 커머셜을, 전국의 특정한 브로드캐스터에 대한 전국 커머셜에 맵핑하는 데이터베이스에 액세스할 수 있다. 이렇게 해서, 셋톱 박스(140)에 신호를 제공하는 로컬 브로드캐스터에 의존하여, 전국 브로드캐스터의 특정한 전국 커머셜이 다른 로컬 커머셜로 대체될 수 있다.

[0051] 대체 콘텐츠 서버(180)의 템플릿 제공 모듈(350)은, 식별된 셋톱 박스에 의거하여 클라이언트 디바이스(150)에 템플릿을 제공한다. 예를 들면, 하나의 제조사에 의한 셋톱 박스는, 다른 제조사의 셋톱 박스와는 다른 메뉴 템플릿의 세트를 가질 수 있다. 마찬가지로, 동일한 제조사로부터의 서로 다른 모델은, 서로 다른 세트의 메뉴 템플릿을 가질 수 있다. 또한, 동일한 모델에 대한 서로 다른 펌웨어 리버전은, 서로 다른 세트의 메뉴 템플릿을 가질 수 있다. 따라서, 셋톱 박스 식별자를 템플릿들의 세트에 맵핑하는 데이터베이스는 대체 콘텐츠 서버(180) 상에서 유지되고, 요청 시에, 완전하게 또는 부분적으로 클라이언트 디바이스(150)에 전송된다.

[0052] 템플릿은 수동으로 또는 프로그램적으로 생성될 수 있다. 수동 생성을 위해, 사람은 클라이언트 디바이스 및 셋톱 박스를 갖고 작업하고, 셋톱 박스의 조작에 의해, 셋톱 박스에 의해 생성될 수 있는 서로 다른 메뉴의 각각의 제시를 일으킨다. 유저는, 클라이언트 디바이스의 스크린 상의 메뉴의 제시의 분석을 통해, 사람은 각각 다른 메뉴 포맷에 대한 템플릿을 생성한다. 템플릿은, 나중에 액세스하기 위해 대체 콘텐츠 서버(180)의 데이터베이스에 저장된다.

[0053] 템플릿의 프로그램적 생성을 위해, 사용 중에 (예를 들면, 클라이언트 디바이스에 의한 대체 콘텐츠 서버(180)

예의 상태의 보고를 통해) 하나 이상의 클라이언트 디바이스(예를 들면, 클라이언트 디바이스(150)) 및 대응하는 셋톱 박스(예를 들면, 셋톱 박스(140))가 모니터링된다. 일부 예시적인 실시예에서, 클라이언트 디바이스 또는 대체 콘텐츠 서버(180)는, 영역들(예를 들면, 스크린 에어리어의 4분의 1 또는 16분의 1)의 핑거프린트를 이용하여 셋톱 박스로부터 수신된 비디오가 기지의 비디오 데이터(예를 들면, 인식된 케이블 프로그램)의 부분 및 미지의 비디오 데이터의 부분을 포함할 때 판정한다. 미지의 비디오 데이터는, 셋톱 박스에 의해 공급되는 추가의 비디오 콘텐츠라고 상정된다. 따라서, 수신된 비디오 데이터의 픽셀-레벨 해석이 대체 콘텐츠 서버(180)에 의해 행해져, 기지의 비디오 데이터에 의해 점유된 영역, 미지의 비디오 데이터에 의해 점유된 영역, 및 (예를 들면, 알파 블렌딩을 통해) 기지의 및 미지의 비디오 데이터의 블렌드에 의해 점유된 영역을 판정한다. 이들 영역들로부터, 템플릿이 생성되고, 데이터베이스에 아직 저장되어 있지 않으면, 나중에 액세스하기 위해 대체 콘텐츠 서버(180)의 데이터베이스에 저장된다.

[0054] 일부 예시적인 실시예에서, 템플릿의 프로그램적 생성은, 셋톱 박스로부터 클라이언트 디바이스가 수신한 비디오 데이터의 다수의 프레임을 모니터링하는 것을 수반한다. 비디오 데이터의 시퀀셜 프레임들을 비교함으로써, 급속히 변화하는 비디오 데이터의 영역은, 정적이거나 또는 비교적 적은 변화를 겪는 영역과는 구별될 수 있다. 이 분석으로부터, 급속히 변화하는 영역은 비디오 프로그래밍을 표시하고, 정적 영역은 셋톱 박스에 의해 제공되는 정보를 표시하는 것으로 추정할 수 있다. 마찬가지로, 급속히 변화하지만 비디오 데이터의 나머지의 것과는 실질적으로 다른 평균 색을 갖는 영역은, 동적 비디오의 정적 오버레이와의 블렌딩이 발생하고 있음을 나타낼 수 있다. 따라서, 블렌딩 영역은, 멀티 프레임 분석을 통해 식별되어도 된다. 이들 영역으로부터, 템플릿이 생성되고, 데이터베이스에 아직 저장되어 있지 않으면, 나중에 액세스하기 위해 대체 콘텐츠 서버(180)의 데이터베이스에 저장된다.

[0055] 도 4는, 클라이언트 디바이스(150)의 표시 영역의 영역(410) 및 영역(420)을 포함하는 동적 비디오 오버레이를 나타내는 예시적인 실시예의 표시도(400)이다. 셋톱 박스(140)로부터 수신된 비디오는 영역(410)에서의 표시를 위해 통과된다. 대체 비디오 데이터는 영역(420)에서 제시된다. 도 4의 예에서 알 수 있는 바와 같이, 영역(420)은 클라이언트 디바이스(150)의 표시 에어리어보다 작은 직사각형 영역이다. 또한, 영역(420)은 표시 영역과 공통되는 에지가 없다. 다른 예시적인 실시예에서, 영역(420)은 다른 형상, 사이즈, 및 위치를 갖는다. 일부 예시적인 실시예에서, 셋톱 박스에 대한 메뉴가 영역(410)을 점유하고, 콘텐츠 소스(110)로부터 수신된 원래의 비디오 데이터가 스케일링(예를 들면, 축소)되고, 셋톱 박스(140)에 의해 재위치결정되고, 셋톱 박스(140)로부터 클라이언트 디바이스(150)로 영역(420)에 송신된다. 따라서, 대체 비디오 데이터가 영역(420)에서 제시될 경우에, 콘텐츠 소스(110)로부터 수신된 비디오 데이터는, 영역(410)에 나타난 메뉴가 유저에게 보기 가능한 채로, 대체된다.

[0056] 도 5는, 클라이언트 디바이스(150)의 표시 에어리어의 영역(510) 및 영역(520)을 포함하는 동적 비디오 오버레이를 나타내는 예시적인 실시예의 표시도(500)이다. 셋톱 박스(140)로부터 수신된 비디오는 영역(510)에서의 표시를 위해 통과된다. 대체 콘텐츠 서버(180)로부터 수신된 대체 비디오 데이터는 영역(520)에 대해 크로핑(cropping)되어 제시된다. 일부 예시적인 실시예에서, 셋톱 박스(140)에 대한 메뉴는 영역(510)을 점유하고, 콘텐츠 소스(110)로부터 수신된 원래의 비디오 데이터의 부분은 셋톱 박스(140)에 의해 영역(520)에 대해 크로핑되고, 셋톱 박스(140)로부터 클라이언트 디바이스(150)로 영역(520)에 송신된다. 따라서, 대체 비디오 데이터의 부분이 영역(520)에 제시될 경우, 콘텐츠 소스(110)로부터 수신된 비디오 데이터는, 영역(510)에 나타난 메뉴가 유저에게 보기 가능한 채로, 대체된다. 도 5에서 알 수 있는 바와 같이, 영역(520)의 상단은 표시 에어리어의 상단과 동일이고, 영역(520)의 좌단 및 우단은 표시 영역의 좌단 및 우단과 공유되지만, 영역(520)은 표시의 전체 높이를 점유하고 있지 않다.

[0057] 도 6은, 클라이언트 디바이스(150)의 표시 에어리어의 영역(610) 및 영역(620)을 포함하는 동적 비디오 오버레이를 나타내는 예시적인 실시예의 표시도(600)이다. 셋톱 박스(140)로부터 수신된 비디오는 영역(610)에서의 표시를 위해 통과된다. 대체 콘텐츠 서버(180)로부터 수신된 대체 비디오 데이터는, 영역(610)에 대응하는 부분을 제외하고, 영역(620)에서 제시된다. 일부 예시적인 실시예에서, 셋톱 박스에 대한 메뉴는 영역(610)을 점유하고, 영역(620)에 대응하는 콘텐츠 소스(110)로부터 수신된 원래의 비디오 데이터의 부분은 셋톱 박스(140)로부터 클라이언트 디바이스(150)로 영역(620)에 송신된다. 따라서, 대체 비디오 데이터의 부분이 영역(620)에 제시될 때, 콘텐츠 소스(110)로부터 수신된 비디오 데이터는, 영역(610)에 나타난 메뉴가 유저에게 보기 가능한 채로, 대체된다.

[0058] 도 7은, 클라이언트 디바이스(150)의 표시 에어리어의 영역(710), 영역(720), 및 영역(730)을 포함하는 동적 비디오 오버레이를 나타내는 예시적인 실시예에서의 표시도(700)이다. 셋톱 박스(140)로부터 수신된 비디오는 영

역(710)에서의 표시를 위해 통과된다. 대체 콘텐츠 서버(180)로부터 수신된 대체 비디오 데이터는, 영역(710)에 대응하는 부분을 제외하고, 영역(720)에 제시된다. 셋톱 박스(140)로부터 수신된 비디오는 대체 비디오 데이터와 블렌드되어 영역(730)에서 제시된다. 예를 들면, 영역(710)에 접하는 영역(730)의 부분이 셋톱 박스(140)로부터 수신된 비디오로 완전하게(또는 거의 완전하게) 구성되고, 영역(720)에 접하는 영역(730)의 부분이, 완전하게(또는 거의 완전하게) 대체 비디오로 구성되도록, 그라데이션트가 적용될 수 있다.

[0059] 일부 예시적인 실시예에서, 셋톱 박스(140)에 의한 화상 처리의 적용을 역으로 하기 위해, 영역(730)에 추가 화상 처리가 적용된다. 예를 들면, 셋톱 박스(140)는, 셋톱 박스(140)에 의해 제공된 정보를 입력 비디오 스트림과 블렌드할 수 있었다. 따라서, 이 블렌드된 정보는, 셋톱 박스(140)에의 입력 비디오에 대응하는 제1 비디오 컴포넌트 및 셋톱 박스(140)에 의해 제공되는 오버레이 정보에 대응하는 제2 비디오 컴포넌트와는 다른 제3 비디오 컴포넌트라고 간주할 수 있다. 입력 비디오 스트림을 비디오 오버레이 모듈(330)의 출력으로부터 완전하게 제거하기 위해, 추가 필터링이 제3 비디오 컴포넌트의 영역(예를 들면, 영역(730))에 적용된다.

[0060] 일부 예시적인 실시예에서, 적용되는 필터링은 밝기에 의존한다. 미리 결정된 밝기값을 초과하는 영역(730)의 부분은, 전경(foreground)에 있고, 유저에게 전해져야 할 정보를 포함하는 것으로 판정된다. 미리 결정된 밝기값을 밑도는 영역(730)의 부분은, 배경에 있는 것으로 판정되고, 따라서, 포함되는 임의의 정보는 필터링 제거되어야 한다. 다른 예시적인 실시예에서, 배경 정보를 필터링 제거하기 위해, 각 배경 픽셀의 색은 모든 배경 픽셀의 평균 색으로 변경된다. 예를 들면, 풋볼 게임의 비디오에 중첩되는 얇은 청색 배경은, 평균화되면, 밝은 청색 색조를 유지하고, 축구 시합에 관한 어떠한 정보도 전하지 않는다. 다른 예시적인 실시예에서, 배경 색은 템플릿에 저장된다. 따라서, 전경 픽셀의 색은 유지되고, 나머지의 픽셀은 배경 색으로 대체된다.

[0061] 추가적인 예시적인 실시예에서, 화상 처리는, 입력 비디오 스트림을 사용하여 셋톱 박스(140)에 의해 행해지는 화상 처리를 역으로 한다. 예를 들면, (예를 들면, 팽거프린팅을 통해) 입력 비디오 스트림을 식별함으로써, 원래의 입력 비디오 스트림에 (예를 들면, 비디오 식별 시스템(190)으로부터) 액세스할 수 있다. 셋톱 박스(140)에 의해 적용된 변경의 역을 적용함으로써, 셋톱 박스(140)에 의해 오버레이된 정보가 복구된다. 예를 들면, 80% 가중의 오버레이 정보를 20% 가중의 입력 비디오 정보와 블렌딩함으로써 셋톱 박스(140)가 출력 비디오를 생성했을 경우, 클라이언트 디바이스(150)는, 입력 비디오의 20% 가중을 빼고 나머지에 1.25를 곱함으로써 프로세스를 역으로 할 수 있다. 다음으로, 셋톱 박스(140)에 의해 이전에 적용된 것과 동일한 블렌드를 적용함으로써, 복원된 오버레이를 대체 비디오 콘텐츠에 적용할 수 있다. 이 예에서, 대체 비디오 콘텐츠의 20% 가중으로 오버레이의 80% 가중을 취함으로써 표시를 위한 픽셀을 생성하는 것을 의미할 것이다.

[0062] 도 8은, 일부 예시적인 실시예에서 비디오 오버레이를 동적으로 제공하기 위한 방법 800을 나타내는 플로우차트이다. 방법 800은, 동작 810, 820, 830, 840, 및 850을 포함한다. 제한이 아닌 예시로서, 동작 810~850은, 도 1의 컴포넌트(110~190), 도 2의 모듈(210~250), 도 3의 모듈(310~350)에 의해 행해지는 것으로서 설명된다.

[0063] 동작 810에서, 셋톱 박스 식별자 모듈(310)은, 클라이언트 디바이스(예를 들면, 클라이언트 디바이스(150))에 비디오 데이터를 제공하는 셋톱 박스(예를 들면, 셋톱 박스(140))를 식별한다. 예를 들면, 셋톱 박스(140)와 클라이언트 디바이스(150)가 HDMI 케이블에 의해 연결될 경우, 셋톱 박스(140)의 EDID는 셋톱 박스(140)를 식별하도록 기능할 수 있다. 다른 예로서, 셋톱 박스 식별자 모듈(310)은, (예를 들면, 옵션들의 리스트로부터 선택함으로써) 유저가 셋톱 박스(140)를 식별할 수 있는 클라이언트 디바이스(150) 상의 유저 인터페이스를 제시할 수 있다. 제3 예로서, 셋톱 박스 식별자 모듈(310)은, 룩업을 위해 그 인터넷 프로토콜(IP) 어드레스를 대체 콘텐츠 서버(180)(또는 다른 서버)에 송신할 수 있다. 이 예에서, 수신 서버는, IP 어드레스를 케이블 회사 서비스 에어리어에 매핑하고, 케이블 회사를 그들의 바람직한 셋톱 박스에 맵핑하는 데이터베이스를 유지한다. 따라서, 클라이언트 디바이스(150)의 IP 어드레스를 사용해서, 셋톱 박스(140)일 가능성이 가장 높은 디바이스의 타입을 식별할 수 있다.

[0064] 셋톱 박스(140)가 셋톱 박스 식별자 모듈(310)에 의해 식별되면, 나중에 액세스하기 위해 셋톱 박스(140)에 대한 식별자가 클라이언트 디바이스(150)에 저장된다. 예를 들면, 유저는, 셋톱 박스(140)를 식별하기 위해 클라이언트 디바이스(150)의 최초의 사용에서 촉구되고, 스토리지로부터 정보에 액세스할 수 있으므로, 장래의 사용에서는 촉구되지 않는다.

[0065] 비디오 소스 식별자 모듈(320)은 수신된 비디오 데이터의 부분을 식별한다(동작 820)). 예를 들면, 수신된 비디오 데이터는 셋톱 박스(140)에 의해 생성된 메뉴가 오버레이된 브로드캐스트 TV 신호를 포함할 수 있다. 패치 선택 모듈(210) 및 값 계산 모듈(220)을 사용하여 수신된 비디오 데이터에 대한 하나 이상의 팽거프린트를 생성해서, 클라이언트 디바이스(150)는 팽거프린트를 비디오 식별 시스템(190)에 송신한다. 이에 응답하여, 클

라이언트 디바이스(150)는, 브로드캐스트 TV 신호를 포함하는 수신된 비디오 데이터의 부분을 식별하도록 기능하는 브로드캐스트 TV 신호에 대응하는 식별자를 수신한다.

[0066] 혹은, 동작(820)은, 핑거프린팅 없이 특정한 패턴에 대해 비디오 데이터를 분석함으로써 행해질 수 있다. 예를 들면, 셋톱 박스(140) 또는 클라이언트 디바이스(150)에 의해 제공되는 특정한 비디오 오버레이는, 스크린 상의 특정한 위치에 브랜드 이름을 포함할 수 있다. 따라서, 스크린의 그 부분은, 특정한 비디오 오버레이가 존재할 때 판단될 브랜드 이름의 캡처된 화상에 대해 비교될 수 있다. 다른 예로서, 특정한 비디오 오버레이가, 스크린 또는 통지 상의 특정한 위치에 아이콘을 포함할 수 있다. 설명하면, 착신 전화에 관한 통지에 전화 아이콘이 나타나도 된다. 따라서, 통지가 존재할 경우, 스크린의 다양한 부분이 판단될 전화 아이콘의 저장된 화상에 대해 비교된다. 이하의 의사코드는, 일부 예시적인 실시예를 나타낸다.

```
templateID templateMatch(videoInStream, deviceID) {
    // iterate over available templates for the intermediate device
    // until a match is found for the input stream
    // return the templateID of the matching stream, or 0 if none.
    matchFound = 0;
    do {
        currentTemplate = getNextTemplate(deviceID);
        if (singleTemplateMatch(videoInStream, currentTemplate))
            matchFound = currentTemplate;
    } until (matchFound != 0)
    return matchFound;
}
```

[0067]

[0068]

```

Boolean singleTemplateMatch(videoInStream, template) {

    // compare the template to the input stream

    // return TRUE if the template matches, FALSE otherwise

    region = template.region;

    count = 0;

    for (x = region.x; x < region.width; x++)

        for (y = region.y; y < region.height; y++)

            if ((videoInStream.currentFrame(x,y) - template(x,y)) ** 2) <=
PIXEL_MATCH_DIFFERENCE)

                count++;

            // for exact pixel matching, PIXEL_MATCH_DIFFERENCE = 0

    maxCount = x*y;

    threshold = PERCENT_THRESHOLD * maxCount;

    if (count >= threshold)

        return TRUE;

    else

        return FALSE;

}

```

[0069]

[0070]

상기 singleTemplateMatch()의 예시적인 실시는, 거리 측도로서, 대응하는 픽셀들간의 차분의 제곱을 사용한다. 다른 예시적 실시에는, 차분의 절대값, 차분의 정규화 제곱, 상관 계수, 정규화 상관 계수, 상호 상관, 정규화 상호 상관, 또는 그 임의의 적절한 조합을 사용한다.

[0071]

동작 830에서, 클라이언트 디바이스(150) 또는 대체 콘텐츠 서버(180)는, 클라이언트 디바이스(150)에 연관된 프로파일 정보 및/또는 식별된 비디오 데이터에 의거하여 대체 비디오를 결정한다. 예를 들면, 특정한 R-등급 콘텐츠의 식별에 의거하여, R-등급 콘텐츠의 PG-등급 대체 버전이 선택될 수 있다. 다른 예로서, 특정한 전국의 커머셜을 대체하도록, 로컬 커머셜을 선택할 수 있다.

[0072]

비디오 오버레이 모듈(330)은, 동작 840에서, 식별된 셋톱 박스 및 비디오 데이터에 의거하여 오버레이 마스크를 판정한다. 일부 예시적인 실시예에서, 비디오 데이터는, 특정한 메뉴 또는 다른 정보 요소가 셋톱 박스(140)에 의해 표시되고 있는지의 여부를 판정하도록, 기지의 비디오 프레임의 세트와 비교된다. 예를 들면, 정보 바가 셋톱 박스(140)에 의해 스크린의 상부에 제시되어, 특정 픽셀은 블랙이고, 특정 영역은 적-녹-청(RGB) 색 튜플로서 표시되는 (110,110,110)~(160,160,160)의 범위의 평균 픽셀 색을 가질 수 있다. 따라서, 수신된 비디오 데이터의 하나의 프레임 또는 일련의 프레임들의 세트가 정보 바의 특성을 가질 경우, 비디오 오버레이 모듈(330)은 대응하는 오버레이 마스크를 식별하며, 통과되는 부분은 바에 의해 점유되는 에어리어에 대응하고, 대체 부분은 셋톱 박스(140)에 의해 수신된 원래의 비디오 콘텐츠에 의해 점유되는 에어리어에 대응한다.

[0073]

도 4 내지 도 7에서 제시되는 예에 대응하는 것 등의 템플릿은, 일부 서로 다른 방법들로 오버레이의 영역 또는

존을 나타낼 수 있다. 예를 들면, 오버레이 존의 경계의 좌표는 템플릿에 의해 제공되어도 된다. 혹은, 디스플레이 전체의 비트맵이 오버레이 존을 지시할 수 있다(예를 들면, 값 1은 대응하는 픽셀이 오버레이의 부분임을 나타내고, 값 0은 대응하는 픽셀이 오버레이의 부분이 아님을 나타냄).

[0074] 상기 예에서, 마스크는, 유저가 셋톱 박스에 의해 생성된 오버레이 정보 및 대체 비디오 콘텐츠의 양쪽을 볼 수 있도록 선택된다. 일부 예시적인 실시예에서, 마스크는, 셋톱 박스에 의해 생성된 오버레이 정보를 억제하도록 선택된다. 예를 들면, 오버레이 정보에 대해 OCR(optical character recognition)을 행해서, 거기에 포함되는 텍스트를 식별할 수 있다. 텍스트에 의거하여, 데이터베이스 룩업을 행해서, 오버레이가 통과되어야 할지 억제해야 할지를 판정할 수 있다. 오버레이가 억제될 경우, 오버레이의 영역 내에 대체 비디오 콘텐츠를 제시하는 마스크가 선택된다. 2가지 접근법은 또한 혼합하여, 일부 오버레이 콘텐츠가 유저에게 제시되고, 다른 오버레이 콘텐츠가 억제되게 할 수 있다. 예를 들면, 오버레이가, 표시되는 제품의 설명 등, 입력 비디오 콘텐츠와 연관된 텍스트 정보를 제시할 경우, 텍스트 정보는 대체 비디오 콘텐츠에 관련될 가능성은 낮다. 따라서, 텍스트 정보를 억제함으로써, 대체 비디오 콘텐츠를 보는데 있어 유저의 경험이 향상된다.

[0075] 동작 850에서, 비디오 오버레이 모듈(330)은, 마스크를 사용하여 비디오 데이터의 부분 및 대체 비디오의 부분을 표시시킨다. 상기 예를 계속하면, 마스크의 통과 부분에 대응하는 정보 바를 포함하는 비디오 데이터가, 마스크의 대체 부분에 대응하는 대체 비디오의 부분과 함께 표시된다. 일부 예시적인 실시예에서, 대체 비디오의 부분의 표시는 대체 비디오의 채워치결정 또는 사이즈 변경을 포함한다. 예를 들면, 도 4의 마스크가 사용될 경우, 대체 비디오는 사이즈가 축소되고, 영역(420)을 점유하도록 채워치결정된다.

[0076] 일부 예시적인 실시예에서, 마스크는 경시적으로 변화된다. 예를 들면, 셋톱 박스(140)에 의해 제시되는 오버레이되는 정보는 애니메이션화되고 경시적으로 위치를 변경할 수 있다. 따라서, 수신된 비디오 데이터 및 대체 비디오 데이터가 제시되는 영역의 위치는 경시적으로 변화될 수 있다. 일부 예에서, 경시적인 대체 비디오의 마스크 및/또는 사이즈/위치의 변경은, 동작 840 및 850을 반복함으로써 실행되어도 된다. 예를 들면, 동작 840 및 850은, 비디오 오버레이 모듈(330)이 오버레이 마스크가 사용되어야 한다고 판정하지 않을 때까지(예를 들면, 들어오는 비디오가 대응하는 마스크를 갖는 템플릿과 더이상 매칭되지 않을 경우) 반복적인 방식으로 반복될 수 있다. 일부 예에서, 경시적인 대체 비디오의 마스크 및/또는 사이즈/위치의 변경은, 특정한 마스크 및/또는 템플릿과 연관된 미리 결정된 타임 루틴(예를 들면, 예측 가능한 방법으로 경시적으로 마스크의 양태가 변화는 애니메이션 루틴)에 의거하여 실행될 수 있다.

[0077] 방법 800은, 클라이언트 디바이스(150)에 의해 제공되는 비디오 요소를 보상하도록, 약간의 변경을 갖고 행해질 수 있다. 클라이언트 디바이스(150)에 의해 제공되는 비디오 요소는, 셋톱 박스(140)에 의해 제공되는 것과 동일한 방식으로 비디오 데이터의 분석에 의해 검출되어도 된다. 혹은, 클라이언트 디바이스(150)는 비디오 오버레이 모듈(330)에 유저 상호작용을 보고할 수 있다. 예를 들면, 클라이언트 디바이스(150)가 메뉴를 표시하는 유저 커맨드를 수신하면, 그 정보는 비디오 오버레이 모듈(330)에 제공될 수 있다. 표시되는 특정한 메뉴에 의거하여, 비디오 오버레이 모듈(330)은 적용할 마스크를 식별한다.

[0078] 도 9는 일부 예시적인 실시예에서, 비디오 오버레이를 동적으로 제공하기 위한 방법 900을 나타내는 플로우차트이다. 방법 900은 동작 810-850 및 910-950을 포함한다. 동작 810-850은 도 8에 대해 상술되어 있다. 제한이 아닌 예시로서, 동작 910-950은, 도 1의 컴포넌트(110-190), 도 2의 모듈(210-250), 및 도 3의 모듈(310-350)에 의해 행해지는 것으로 설명되어 있다.

[0079] 동작 910에서, 클라이언트 디바이스(150)는, 셋톱 박스(140)에 의해 제2 비디오 데이터에 공급되는 비디오 데이터의 변경을 검출한다. 예를 들면, 셋톱 박스(140)의 유저는, 표시되는 채널을 변경했을 수 있다.

[0080] 변경의 검출에 응답해서, 비디오 오버레이 모듈(330)은 마스크의 적용을 중지하고, 셋톱 박스(140)로부터의 수신 비디오 전체를 표시하게 할 수 있다(동작920). 따라서, 셋톱 박스(140)의 유저는 변경된 채널의 효과를 볼 것이다.

[0081] 동작 930 및 940에서, 클라이언트 디바이스(150)는 셋톱 박스(140)에 의해 공급되는 비디오 데이터의 제2 변경을 검출한다. 제2 변경은 수신 비디오 데이터를 이전의 상태로 되돌린다. 예를 들면, 셋톱 박스(140)의 유저는, 표시되는 원래의 비디오 채널로 복귀할 수 있다. 검출은, 비디오 식별 시스템(190)으로부터 예상 핑거프린트의 세트를 수신하고, 예상 핑거프린트를 쿼리 핑거프린트 생성기(160)에 의해 생성된 핑거프린트와 비교함으로써, 달성될 수 있다. 따라서, 유저가 식별된 콘텐츠로부터 채널을 벗어나게 변경할 경우, 핑거프린트는 매칭을 중지할 것이고, 유저가 채널을 식별된 콘텐츠로 다시 변경할 경우, 핑거프린트는 매칭을 재개한다.

- [0082] 동작 950에서, 제2 변경이 제1 비디오 데이터로의 복귀인 판정에 응답해서, 비디오 오버레이 모듈(330)은 마스크의 사용을 재개해서, 제1 비디오 데이터의 부분 및 대체 비디오의 부분의 표시를 하게 한다. 따라서, 클라이언트 디바이스(150)의 유저는 동작 850에서 제시된 대체 비디오로의 복귀를 인식할 것이다.
- [0083] 도 10은, 일부 예시적인 실시예에서, 비디오 오버레이를 동적으로 제공하기 위한 방법(1000)을 나타내는 플로우차트이다. 방법(1000)은 동작 810, 820, 830, 840, 1010, 및 1020을 포함한다. 동작 810~840은 도 8에 대해 상술되어 있다. 제한이 아닌 예시로서, 동작 1010 및 동작 1020은 도 1의 컴포넌트(110~190), 도 2의 모듈(210~250), 및 도 3의 모듈(310~350)에 의해 행해지는 것으로 설명된다.
- [0084] 동작 1010에서, 비디오 소스 식별자 모듈(320)은 식별된 비디오 데이터의 재생 레이트를 판정한다. 예를 들면, 비디오 식별 시스템(190)은, 수신 비디오 데이터에 대해 생성되는 핑거프린트와 비교되는 비디오 데이터에 대한 예상 핑거프린트의 세트를 제공할 수 있다. 따라서, 수신 비디오 데이터의 연속되는 프레임의 핑거프린트가 비(non)일련의 예상 핑거프린트(예를 들면, 제2 또는 제4 핑거프린트마다)와 매칭될 경우, 수신 비디오가 비디오 식별 시스템(190)에 의해 사용되는 레퍼런스 비디오와는 다른 레이트(예를 들면, 2배 또는 4배 빠르기)로 플레이됨이 판정된다.
- [0085] 동작 1020에서, 비디오 오버레이 모듈(330)은, 마스크 및 재생 레이트에 의거하여 셋톱 박스(140)로부터 수신된 비디오 데이터의 부분 및 대체 비디오의 부분을 표시시킨다. 예를 들면, 동작 850에서와 같이, 마스크를 사용하여, 셋톱 박스(140)로부터 수신된 비디오의 어느 부분이 통과되고 어느 부분이 대체 비디오로 대체되는지를 판정한다. 또한, 재생 레이트는, 대체 비디오의 재생 레이트를 제어하는데 사용된다. 이렇게 해서, 클라이언트 디바이스(150) 및 셋톱 박스(140)의 유저는, 빨리 감기, 되감기, 및 일시 정지 등의 재생 제어가 원래의 표시 비디오에서 동작하는 것과 동일한 방식으로, 대체 비디오에서 동작함을 인식한다.
- [0086] 도 11은, 일부 예시적인 실시예에서, 비디오 오버레이를 동적으로 제공하기 위한 방법을 나타내는 플로우차트이다. 방법 1100은 동작 1110~1150을 포함한다. 제한이 아닌 예시로서, 동작 1110~1150은, 도 1의 컴포넌트(110~190), 도 2의 모듈(210~250), 및 도 3의 모듈(310~350)에 의해 행해지는 것으로서 설명된다.
- [0087] 동작 1110에서, 클라이언트 디바이스(150)의 셋톱 박스 식별자 모듈(310)은, 콘텐츠 프로바이더(예를 들면, 콘텐츠 소스(110)) 및 중간 디바이스(예를 들면, 셋톱 박스(140))로부터의 콘텐츠를 포함하는 비디오 입력 스트림에 액세스한다. 예를 들면, 셋톱 박스(140)는, 케이블 연결을 통해 콘텐츠 소스(110)로부터의 비디오 스트림을 수신하고, 비디오 스트림을 변경하고, 변경된 비디오 스트림을 HDMI 연결을 통해 클라이언트 디바이스(150)에 전송했을 수 있다. 클라이언트 디바이스(150)는, 비디오 입력 스트림이 수신됨에 따라, 또는 데이터 스토리지로부터(예를 들면, 클라이언트 디바이스(150)에 통합된 솔리드 스테이트 드라이브로부터) 비디오 입력 스트림에 액세스할 수 있다.
- [0088] 동작 1120에서, 셋톱 박스 식별자 모듈(310)은, 중간 디바이스에 의해 생성된 스크린 부분에 대응하는 템플릿에 액세스한다. 예를 들면, 중간 디바이스에 의해 행해지는 특정한 화상 조작은, 클라이언트 디바이스(150)에 알려질 수 있다. 화상 조작은 스크린의 특정한 부분에 대응할 수 있다. 예시를 위해, 셋톱 박스(140)는, 스크린의 상부 또는 하부에 메뉴를 배치함으로써, 콘텐츠 소스(110)로부터 수신된 비디오 스트림을 변경할 수 있다. 셋톱 박스 식별자 모듈(310)에 의해 액세스되는 템플릿은, 중간 디바이스에 의해 생성된 비디오를 포함하는 스크린 부분에 대응한다.
- [0089] 동작 1130에서, 셋톱 박스 식별자 모듈(310)은 비디오 입력 스트림을 분석해서, 비디오 입력 스트림의 프레임이 템플릿과 매칭됨을 판정한다. 예를 들면, 템플릿은, 비디오 입력 스트림이 셋톱 박스(140)에 의해 생성된 메뉴를 포함할 경우에, 특정한 색일 것인 특정한 픽셀을 지시할 수 있다. 비디오의 각각의 들어오는 프레임을 템플릿과 비교해서, 스크린 부분이 메뉴를 포함하고 있을 가능성이 있는지의 여부를 판정할 수 있다. 비디오의 일부 프레임은 분석되지 않을 수 있다. 예를 들면, 매 2번째 프레임 또는 매 5번째 프레임을 템플릿과 비교함으로써, 비교에 소비되는 프로세서 시간의 양을 삭감할 수 있다. 일부 예시적인 실시예에서, 동작 1130은 대체 콘텐츠가 제시되고 있는 동안에만 행해지고, 그에 따라, 오탐(false positive)이 발생할 가능성이 저감된다.
- [0090] 템플릿 매칭은, 비디오 입력 스트림의 프레임의 다운샘플링된 버전과 템플릿의 다운샘플링된 버전의 픽셀마다의 비교를 사용해서 행해질 수 있다. (다운샘플링된) 프레임과 템플릿 사이의 픽셀 거리를 계산할 수 있다. 예를 들면, 비매칭 픽셀의 수를 픽셀 차분으로서 사용할 수 있다. 대안으로, 각 픽셀에 대한 색거리를 계산할 수 있고(예를 들면, RGB에서, 휘도 및 채도(YUV); HSL(hue, saturation, 및 lightness), 또는 HSV(hue, saturation, 및 value) 색공간), 색거리의 절대값의 합이 픽셀 거리로서 취해질 수 있다. 판정된 픽셀 거리는 역치와 비교

되어, 템플릿이 비디오 입력 스트림에 대한 매칭인지의 여부를 판정한다. 일부 예시적인 실시예에서, 템플릿과 비디오 입력 스트림은 상호상관된다. 상호상관은, 하나의 시퀀스가 다른 시퀀스에 비해 늦어질 경우 프레임 시퀀스의 유사성의 측도를 판정함으로써, 멀티 프레임 템플릿의 처리를 가능하게 한다.

[0091] 일부 예시적인 실시예에서, 다수의 템플릿이 비디오 입력 스트림에 비교된다. 예를 들면, 도 16의 테이블(1640)은 복수의 셋톱 박스 각각에 대한 템플릿을 저장할 수 있다. 테이블(1640) 내의 각 템플릿은, 비디오 입력 스트림의 어느 픽셀이 템플릿의 화상의 픽셀과 비교되어야 할지를 지시하는 비트맵을 포함한다. 셋톱 박스(140)가 이미 식별되어 있을 경우에도, 다수의 타입의 다수의 템플릿이 비디오 입력 스트림과의 비교에 이용 가능하다. 일부 예시적인 실시예에서, 템플릿들이 계산 상의 복잡도에 의해 소팅되어, 계산 비용의 덜한 템플릿이 첫째로 테스트된다. 제1 매칭 템플릿을 사용하고, 비교 프로세스를 정지할 수 있다. 혹은, 모든 템플릿이 비교되고, 최선의 매치(예를 들면, 비디오 입력 스트림 프레임으로부터 가장 낮은 픽셀 거리를 갖는 템플릿)가 사용되어도 된다.

[0092] 비디오 오버레이 모듈(330)은, 동작 1140에서, 비디오 입력 스트림의 프레임이 템플릿과 매칭됨의 판정에 의거하여 비디오 출력 스트림을 생성한다. 예를 들면, 콘텐츠 소스(110)로부터의 콘텐츠가 클라이언트 디바이스(150)에 의해 대체 콘텐츠로 대체되고, 셋톱 박스(140)가 메뉴 등의 보충 콘텐츠를 제공하고 있다고 판정될 경우, 비디오 출력 스트림은, 메뉴가 검출되었을 경우, 적절한 오버레이 마스크를 적용하여 메뉴가 비디오 출력 스트림에 나타나도록 하는 등, 메뉴를 포함하도록 변경될 수 있다.

[0093] 동작 1150에서, 비디오 오버레이 모듈(330)은, 비디오 출력 스트림을 디스플레이(예를 들면, 클라이언트 디바이스(150)의 디스플레이) 상에 제시시킨다. 일부 클라이언트 디바이스(150)는, 비디오 오버레이 모듈(330)이 비디오 입력 스트림을 제시할 디스플레이의 부분, 및 대체 비디오 콘텐츠를 제시할 디스플레이의 다른 부분을 지정하는 것을 가능하게 하는 API(application programming interface)를 제공한다. 예를 들면, 픽셀 단위로 비디오 입력 스트림 픽셀 또는 대체 픽셀을 사용해야 하는지의 여부를 지시는 1비트 투과 마스크를 사용함으로써, 또는 픽셀 단위로 비디오 입력 스트림 픽셀과 대체 픽셀 사이의 블렌딩의 정도를 지시하는 알파 마스크를 사용함으로써, 직사각형 영역의 코너 좌표를 사용해서, 부분들을 지정할 수 있다. 해당 클라이언트 디바이스(150)에 대해서, API를 사용할 수 있다. 다른 방식으로, 비디오 오버레이 모듈(330)에 의해 임의의 부분 블렌딩 또는 대체가 행해진다.

[0094] 도 12는, 일부 예시적인 실시예에서, 비디오 오버레이를 동적으로 제공하기 위한 방법 1200을 나타내는 플로우 차트이다. 방법 1200은 동작 1210~1240을 포함한다. 제한이 아닌 예시로서, 동작 1210~1240은, 도 2의 모듈(210~250), 도 3의 모듈(310~350)에 의해 행해지는 것으로 설명된다.

[0095] 동작 1210에서, 클라이언트 디바이스(150)의 셋톱 박스 식별자 모듈(310)은, 제1 콘텐츠, 및 오버레이 존의 제1 콘텐츠 상에 적어도 부분적으로 오버레이된 반투명 오버레이를 포함하는 비디오 입력 스트림에 액세스한다. 예를 들면, 제1 콘텐츠는 콘텐츠 프로바이더(예를 들면, 콘텐츠 소스(110))로부터의 콘텐츠일 수 있고, 반투명 오버레이는 중간 디바이스(예를 들면, 셋톱 박스(140))에 의해 제공될 수 있다. 예를 들면, 셋톱 박스(140)는, 케이블 연결을 통해 콘텐츠 소스(110)로부터 비디오 스트림을 수신하고, 비디오 스트림을 변경하고, 변경된 비디오 스트림을 HDMI 연결을 통해 클라이언트 디바이스(150)에 전송했을 수 있다. 클라이언트 디바이스(150)는, 비디오 스트림이 수신됨에 따라, 또는 데이터 스토리지로부터(예를 들면, 클라이언트 디바이스(150)에 통합된 솔리드 스테이트 드라이브로부터) 비디오 스트림에 액세스할 수 있다.

[0096] 셋톱 박스 식별자 모듈(310)은, 동작 1220에서, 비디오 입력 스트림에 오버레이 존의 지시를 포함하는 템플릿에 액세스한다. 예를 들면, 중간 디바이스에 의해 행해지는 특정한 화상 조작이, 클라이언트 디바이스(150)에 알려질 수 있다. 화상 조작은 스크린의 특정한 부분에 대응할 수 있다. 예시를 위해, 셋톱 박스(140)는, 스크린의 상부 또는 하부에 반투명의 메뉴를 배치함으로써, 콘텐츠 소스(110)로부터 수신된 비디오 스트림을 변경할 수 있다. 셋톱 박스 식별자 모듈(310)에 의해 액세스되는 템플릿은, 중간 디바이스(예를 들면, 셋톱 박스(140))에 의해 생성된 반투명 오버레이를 포함하는 스크린 부분에 대응한다.

[0097] 비디오 소스 식별자 모듈(320)은 핑거 프린팅을 사용하여 콘텐츠를 식별할 수 있다. 예를 들면, 수신된 비디오 데이터는, 셋톱 박스(140)에 의해 생성된 반투명 메뉴가 오버레이된 브로드캐스트 TV 신호를 포함할 수 있다. 패치 선택 모듈(210) 및 값 계산 모듈(220)을 사용해서 수신된 비디오 데이터에 대해 하나 이상의 핑거프린트를 생성하여, 클라이언트 디바이스(150)는 핑거프린트를 비디오 식별 시스템(190)에 송신한다. 그에 응답해서, 클라이언트 디바이스(150)는, 브로드캐스트 TV 신호를 포함하는 수신 비디오 데이터의 부분을 식별하도록 기능하는 브로드캐스트 TV 신호에 대응하는 식별자를 수신한다.

[0098] 비디오 오버레이 모듈(330)은, 동작 1230에서, 대체 콘텐츠 상에 오버레이되는 변경된 반투명 오버레이를 포함하는 비디오 출력 스트림을 생성한다. 예를 들면, 콘텐츠 소스(110)로부터의 콘텐츠가 클라이언트 디바이스(150)에 의해 대체 콘텐츠로 대체되고 셋톱 박스(140)가 콘텐츠 소스(110)로부터의 콘텐츠에 반투명 오버레이를 추가했음의 판정이 이루어질 경우, 비디오 출력 스트림은 대체 콘텐츠 및 반투명 오버레이의 변경된 버전을 포함한다. 동작 1240에서, 비디오 오버레이 모듈(330)은, 비디오 출력 스트림을 디스플레이(예를 들면, 클라이언트 디바이스(150)의 디스플레이) 상에 제시시킨다.

[0099] 이하의 의사코드는, 일부 대안의 실시예를 나타낸다.

```
semitransparentOverlayTransfer(inputVideoStream, identifiedVideoContent,
overlayRegion) {
    // inputVideoStream is the received video stream
    // identifiedVideoContent is the content as transmitted from the content
    // provider, without the modifications performed by the intermediate device
    // overlayRegion identifies the region defined by the template that was detected
    // prior to invocation of this method, outputVideoStream contains
    // replacement content, without the semi-transparent overlay
    for (each pixel in overlayRegion) {
        // transfer the modifications performed by the intermediate device
        // to the replacement content
        delta = inputVideoStream(pixel) - identifiedVideoContent(pixel);
        outputVideoStream(pixel) += delta;
    }
}
```

[0101]

```

semitransparentOverlayTransfer(inputVideoStream, identifiedVideoContent,
overlayRegion) {
    // inputVideoStream is the received video stream
    // identifiedVideoContent is the content as transmitted from the content
    // provider, without the modifications performed by the intermediate device
    // overlayRegion identifies the region defined by the template that was detected
    // prior to invocation of this method, outputVideoStream contains
    // replacement content, without the semi-transparent overlay
    for (each pixel in overlayRegion) {
        // if the pixel has been substantially changed by the intermediate
        // device, use the changed pixel instead of the corresponding pixel of
        // replacement content
        delta = inputVideoStream(pixel) - identifiedVideoContent(pixel);
        if (delta > PIXEL_OVERLAY_THRESHOLD)
            outputVideoStream(pixel) = inputVideoStream(pixel);
    }
}

```

[0102]

```

semitransparentOverlayTransfer(inputVideoStream, identifiedVideoContent,
overlayRegion, transparency) {
    // inputVideoStream is the received video stream
    // identifiedVideoContent is the content as transmitted from the content
    // provider, without the modifications performed by the intermediate device
    // overlayRegion identifies the region defined by the template that was detected
    // transparency is a value between 0 and 1, indicating the degree of
    // transparency of the semitransparent overlay (0 indicating full opacity and
    // 1 full transparency) as used by the intermediate device
    // prior to invocation of this method, outputVideoStream contains
    // replacement content, without the semi-transparent overlay
    for (each pixel in overlayRegion) {
        // the result from the intermediate device is contentVideoStream *
        // (transparency) + overlayPixel * (1 - transparency)
        // solving for overlayPixel gives:
        overlayPixel = (inputVideoStream(pixel) - identifiedVideoContent(pixel) *
transparency) / (1 - transparency);
        // now apply the original overlay to our output
        outputVideoStream(pixel) *= transparency;
        outputVideoStream(pixel) += overlayPixel * (1 - transparency);
    }
}

```

```

semitransparentOverlayTransfer(inputVideoStream, overlayRegion, transparency) {

```

[0103]

```

// inputVideoStream is the received video stream
// overlayRegion identifies the region defined by the template that was detected
// transparency is a value between 0 and 1, indicating the degree of
// transparency of the semitransparent overlay (0 indicating full opacity and
// 1 full transparency) to use in the output. The transparency may be, but
// need not be, the same as that used by the intermediate device
// prior to invocation of this method, outputVideoStream contains
// replacement content, without the semi-transparent overlay
// identify the color of the semi-transparent overlay as the average color of
// the overlay region (e.g., random un-modified pixels should average to a
// neutral gray, such as RGB (128, 128, 128), pixels modified by a semi-
// transparent overlay should average to the color of the overlay)
// Alternatively, the overlayColor may be included as data for the template rather
// than calculated from the inputVideoStream
overlayColor = average(pixels in overlay region);
// pixels in the overlay region are either background pixels or foreground pixels.
// background pixels are those that have color values within a predetermined
// range of the overlay color; foreground pixels are those that are outside
// of the predetermined range
for (each pixel in overlayRegion) {
    // the calculation of delta is represented as a simple subtraction, but
    // could be implemented as a Euclidian distance in a color space
    delta = inputVideoStream(pixel) - overlayColor;
    // copy foreground pixels without change, but apply transparency to

```

[0104]

```

// background pixels
if (delta > PIXEL_OVERLAY_THRESHOLD)
    outputVideoStream(pixel) = inputVideoStream(pixel);
else {
    outputVideoStream(pixel) *= transparency;
    outputVideoStream(pixel) += overlayColor * (1 - transparency);
}
}
}

semitransparentOverlayTransfer(inputVideoStream, overlayRegion, transparency,
textColor) {
    // inputVideoStream is the received video stream
    // overlayRegion identifies the region defined by the template that was detected
    // transparency is a value between 0 and 1, indicating the degree of
    // transparency of the semitransparent overlay (0 indicating full opacity and
    // 1 full transparency) to use in the output. The transparency may be, but
    // need not be, the same as that used by the intermediate device
    // prior to invocation of this method, outputVideoStream contains
    // replacement content, without the semi-transparent overlay
    // identify the color of the semi-transparent overlay as the average color of
    // the overlay region (e.g., random un-modified pixels should average to a
    // neutral gray, such as RGB (128, 128, 128), pixels modified by a semi-
    // transparent overlay should average to the color of the overlay)

```

[0105]

```

// Alternatively, the overlayColor may be included as data for the template rather
// than calculated from the inputVideoStream
// textColor is a color value, the color value used for text by the intermediate
// device
overlayColor = average(pixels in overlay region);
for (each pixel in overlayRegion) {
    // copy text without change, but apply transparency to background pixels
    // the pixels containing input text are, in the aggregate, the text portion
    // of the inputVideoStream
    if (inputVideoStream(pixel) == textColor)
        outputVideoStream(pixel) = inputVideoStream(pixel);
    else {
        outputVideoStream(pixel) *= transparency;
        outputVideoStream(pixel) += overlayColor * (1 - transparency);
    }
}
}

```

```

semitransparentOverlayTransfer(inputVideoStream, overlayRegion, transparency,
messageType) {

```

```

    // inputVideoStream is the received video stream
    // overlayRegion identifies the region defined by the template that was detected
    /// transparency is a value between 0 and 1, indicating the degree of
    // transparency of the semitransparent overlay (0 indicating full opacity and

```

[0106]

```

// 1 full transparency) to use in the output. The transparency may be, but
// need not be, the same as that used by the intermediate device
// prior to invocation of this method, outputVideoStream contains
// replacement content, without the semi-transparent overlay
// identify the color of the semi-transparent overlay as the average color of
// the overlay region (e.g., random un-modified pixels should average to a
// neutral gray, such as RGB (128, 128, 128), pixels modified by a semi-
// transparent overlay should average to the color of the overlay)
// Alternatively, the overlayColor may be included as data for the template rather
// than calculated from the inputVideoStream
// messageType indicates an information type of text to be displayed
// E.g., MESSAGE_TIME = current time,
// MESSAGE_CHANNEL = current channel
overlayColor = average(pixels in overlay region);
for (each pixel in the overlay region) {
    // apply the overlay color to the region
    outputVideoStream(pixel) *= transparency;
    outputVideoStream(pixel) += overlayColor * (1 - transparency);
}
// set message to a string containing the appropriate information
message = getMessage(messageType);
// add the text of the message to the screen, for viewing by the user
printMessage(message, overlayRegion);

```

[0107] }

[0108] 도 13은, 일부 예시적인 실시예에서, 비디오 오버레이를 동적으로 제공하는 방법 1300을 나타내는 플로우차트이다. 방법 1300은 동작 1310~1360을 포함한다. 제한이 아닌 예시로서, 동작 1310~1360은 도 2의 모듈(210~250) 및 도 3의 모듈(310~350)에 의해 행해지는 것으로서 설명된다.

[0109] 동작 1310에서, 클라이언트 디바이스(150)는 비디오 출력 스트림을 디스플레이 상에 제시시킨다. 예를 들면, 쿼리 핑거프린트 생성기(160)는, 셋톱 박스(140)로부터 수신된 비디오 입력 스트림으로부터 쿼리 핑거프린트를 생성했을 수 있다. 생성된 쿼리 핑거프린트에 의거하여, 대체 비디오 콘텐츠가 대체 콘텐츠 서버(180)로부터 수신되고, 비디오 출력 스트림으로서 클라이언트 디바이스(150)의 디스플레이에 표시되었을 수 있다.

[0110] 클라이언트 디바이스(150)는, 동작 1320에서, 콘텐츠 프로바이더 및 중간 디바이스로부터의 콘텐츠를 포함하는 비디오 입력 스트림에 액세스한다. 예를 들면, 비디오 입력 스트림은, 셋톱 박스(140)로부터의 추가 콘텐츠를 포함하도록 변경된 콘텐츠 소스(110)로부터의 콘텐츠를 포함할 수 있다.

[0111] 동작 1330에서, 클라이언트 디바이스(150)는, 중간 디바이스에 의해 생성된 스크린 부분에 대응하는 템플릿에

액세스한다. 예를 들면, 비디오 오버레이 모듈(330)은, 셋톱 박스(140)의 디바이스 식별자에 대한 데이터를 포함하는 도 16의 데이터베이스 스키마 내의 레코드에 액세스할 수 있다.

[0112] 비디오 오버레이 모듈(330)은, 동작 1340에서, 비디오 입력 스트림을 분석해서, 비디오 입력 스트림의 프레임의 콘텐츠의 적어도 부분이 템플릿과 매칭됨을 판정한다. 일부 예시적인 실시예에서, 중간 디바이스에 대한 템플릿들 중 어느 것이 중간 디바이스로부터 수신된 비디오 입력 스트림의 프레임의 콘텐츠에 대응함의 여부를 판정하도록, 다수의 그러한 템플릿이 반복된다. 이 템플릿은, 중간 디바이스에 의해 비디오 입력 스트림에 추가된 아이콘(예를 들면, 전화 아이콘 또는 메일 아이콘), 중간 디바이스에 의해 비디오 입력 스트림에 추가된 미리 결정된 텍스트(예를 들면, "전화" 또는 "메일"), 중간 디바이스에 의해 비디오 입력 스트림에 추가된 특정한 위치의 윈도우, 또는 그 임의의 적절한 조합에 대응할 수 있다.

[0113] 하기 의사코드 방법은, 일부 예시적인 실시예를 나타낸다.

```
findMatchingTemplate(inputVideoStream, deviceId) {
    // inputVideoStream is the received video stream
    // device Id is the identifier of the intermediate device
    matchingTemplate = NULL;
    // matchingTemplate will store the identified match, or NULL if none
    while (matchingTemplate == NULL && template==getNextTemplate(deviceId))
    {
        // iterate over all of the templates for the intermediate device
        // until we have looked at them all or we have found a match
        // bitwise-AND the current frame of the inputVideoStream with the bitmap
        // of the template. This will zero out the portions of the frame that are not used
        // in determining if this template matches
        frame = inputVideoStream.currentFrame() & template.bitmap;
        // determine the difference between the frame and the template
        // this may be the sum of the absolute values of the distance in RGB space for
        // each compared pixel
        delta = compareImage(frame, image);
        // if the template matches the frame, this is the matching template
        if (delta < TEMPLATE_MATCH_THRESHOLD)
            matchingTemplate = template;
    }
}
```

[0114]

[0115] 비디오 오버레이 모듈(330)은, 동작 1350에서, 템플릿과 비디오 입력 스트림의 프레임의 콘텐츠의 적어도 부분 사이의 매칭에 의거하여, 중간 디바이스에 의해 생성되는 스크린 부분을 포함하도록, 비디오 출력 스트림을 변

경한다. 예를 들면, 셋톱 박스(140)가 콘텐츠 소스(110)로부터 수신된 콘텐츠에 통지 메시지를 추가할 경우, 비디오 입력 스트림의 하나 이상의 프레임은 통지에 대한 템플릿과 매칭될 것이다. 비디오 오버레이 모듈(330)은 통지를 포함하는 비디오 입력 스트림의 프레임의 부분을 식별하고, 통지를 비디오 출력 스트림에 카피한다. 일부 예시적인 실시예에서, 통지를 카피하는 것은, 템플릿에 의해 식별되는 통지 에어리어 내의 비디오 입력 스트림의 각 픽셀을 비디오 출력 스트림의 대응하는 픽셀에 카피하는 것을 포함한다. 다른 예시적인 실시예에서, 통지를 카피하는 것은, 통지의 콘텐츠를 (예를 들면, OCR 또는 화상 인식을 통해) 판정하는 것, 비디오 출력 스트림 내에 콘텐츠를 재생성하는 것을 포함한다. 이렇게 해서, 대체 콘텐츠 서버(180)로부터의 대체 콘텐츠가 표시되고 있는 동안, 셋톱 박스(140)로부터의 통지가 클라이언트 디바이스(150)의 시청자에게 제시된다. 동작 1360에서, 비디오 오버레이 모듈(330)은, 변경된 비디오 출력 스트림을 디스플레이(예를 들면, 클라이언트 디바이스(150)의 디스플레이) 상에 제시시킨다.

- [0116] 일부 예시적인 실시예에서, 비디오 입력 스트림이 더이상 템플릿과 매칭되지 않을 경우에, 비디오 출력 스트림의 변경이 중단된다. 다른 예시적인 실시예에서, 비디오 입력 스트림이 변경의 종료와 연관된 제2 템플릿과 매칭될 경우에, 비디오 출력 스트림의 변경이 중단된다. 예를 들면, 셋톱 박스(140)로부터의 통지는, 제2 템플릿에 의해 매칭될 수 있는 통지의 단음의 애니메이션에 서명 프레임을 가질 수 있다. 따라서, 제2 템플릿이 매칭되면, 비디오 출력 스트림은 통지를 포함하도록 더이상 변경되지 않는다.
- [0117] 도 14는, 일부 예시적인 실시예에서, 비디오 오버레이를 동적으로 제공하기 위한 방법을 나타내는 플로우차트이다. 방법 1400은 동작 1310~1340, 1360, 및 1450~1460을 포함한다. 동작 1310~1340 및 1360은 도 13에 대해 상술되어 있다. 제한이 아닌 예시로서, 동작 1450~1460은, 도 2의 모듈(210~250) 및 도 3의 모듈(310~350)에 의해 행해지는 것으로서 설명된다.
- [0118] 동작 1450에서, 비디오 오버레이 모듈(330)은, 매칭에 의거하여, 비디오 입력 스트림의 재생 속도가 디폴트 속도로부터 변경되었음을 판정한다. 예를 들면, 템플릿은, 비디오 입력 스트림의 특정한 부분에서 오른쪽을 가리키는 이중 삼각형을 지시할 수 있다. 이 이중 삼각형이 존재함에 의거하여(비디오 입력 스트림과 템플릿 사이의 매칭에 의해 판정), 비디오 오버레이 모듈(330)은 재생 속도가 증가했음을 판정한다. 마찬가지로, 템플릿은, 좌측을 가리키는 이중 삼각형 또는 일시 정지 심볼을 지시할 수 있고, 비디오 오버레이 모듈(330)은 비디오 입력 스트림의 재생 속도가 역으로 또는 일시 정지되었음을 판정할 수 있다.
- [0119] 일부 예시적인 실시예에서, 템플릿은 또한 재생 레이트를 지시한다. 예를 들면, "x2" 또는 "x4"가 존재하고, 이는 재생 속도가 2배 또는 4배임을 지시한다. 다른 예로서, 2개의 삼각형이 2배속을 지시하고, 3개의 삼각형이 4배속을 지시하는 등 할 수 있다. 특정한 템플릿에 대한 특별한 의미는, 셋톱 박스 식별자 모듈(310)에 의해 식별되는 셋톱 박스(140)에 의존할 수 있다.
- [0120] 다른 예시적인 실시예에서, 템플릿은 재생 레이트를 지시하지 않는다. 재생 레이트는, 비디오 입력 스트림의 연속되는 프레임의 핑거프린트를 기준 스트림의 연속되는 프레임의 핑거프린트와 비교함으로써 판정될 수 있다. 비디오 입력 스트림의 연속되는 프레임 핑거프린트가 기준 스트림의 2번째 핑거프린트마다 매칭될 경우, 재생은 2배속으로 일어나고 있다. 핑거프린트가 역순일 경우, 재생이 역으로 행해진다. 비디오 입력 스트림의 연속되는 프레임의 핑거프린트가 2배로 되면, 재생은 절반 속도로 일어난다.
- [0121] 비디오 오버레이 모듈(330)은, 동작 1460에서, 비디오 출력 스트림의 재생 속도를 변경한다. 일부 예시적인 실시예에서, 비디오 출력 스트림의 재생 속도는, 비디오 입력 스트림의 재생 속도와 매칭되도록 변경된다. 따라서, 비디오 입력 스트림의 속도가 디폴트 속도에 비해 2배로 되면, 비디오 출력 스트림의 속도도 2배로 된다.
- [0122] 본 명세서에 개시된 다양한 방법을 조합할 수 있다. 예를 들면, 방법 1300을 방법 1400과 조합하는 것은, 비디오 입력 스트림이 변경된 스트림을 가질 경우, 변경된 속도를 갖는 비디오 출력 스트림으로 되게 하고, 비디오 출력 스트림은 또한, 중간 디바이스에 의해 생성된 스크린 부분(예를 들면, 빨리 감기 인디케이터)을 포함한다.
- [0123] 도 15는, 일부 예시적인 실시예에서, 비디오 오버레이를 동적으로 제공하기 위한 방법을 나타내는 플로우차트이다. 방법 1500은 동작 1510~1590을 포함한다. 제한이 아닌 예시로서, 동작 1510~1590은 도 2의 모듈(210~250) 및 도 3의 모듈(310~350)에 의해 행해지는 것으로서 설명된다.
- [0124] 동작 1510에서, 클라이언트 디바이스(150)는, 콘텐츠 프로바이더로부터의 콘텐츠를 포함하는 비디오 입력 스트림에 액세스한다. 예를 들면, 클라이언트 디바이스(150)의 비디오 입력 디바이스(1714)(도 17에 도시)는 셋톱 박스(140)의 비디오 출력에 연결될 수 있다. 비디오 입력 스트림은, 셋톱 박스(140)로부터 클라이언트 디바이스(150)로 송신될 수 있다. 비디오 입력 스트림은, 콘텐츠 소스(110)로부터의 콘텐츠 및 셋톱 박스(140)에 의

해 생성된 콘텐츠를 포함할 수 있다. 예를 들면, 셋톱 박스(140)는, 콘텐츠 소스(110)로부터 수신된 비디오 스트림의 부분에 중첩된 정보 오버레이를 제공할 수 있다.

- [0125] 동작 1520에서, 클라이언트 디바이스(150)는, 비디오 입력 스트림의 콘텐츠에 의거하여 비디오 출력 스트림을 선택한다. 예를 들면, 쿼리 핑거프린트 생성기(160)는, 비디오 입력 스트림의 하나 이상의 프레임의 적어도 부분에 의거하여 쿼리 핑거프린트를 생성할 수 있다. 핑거프린트에 의거하여, 비디오 출력 스트림으로서 사용하기 위해 대체 비디오 콘텐츠를 선택한다(예를 들면, 데이터베이스 룩업을 통해, 대체 콘텐츠 서버(180)와의 통신을 통해, 또는 그 모두를 통해).
- [0126] 동작 1530에서, 클라이언트 디바이스(150)는 선택된 비디오 출력 스트림을 디스플레이 상에 제시시킨다. 예를 들면, 비디오 출력 스트림은 클라이언트 디바이스(150)의 디스플레이 상에 제시될 수 있다. 혹은, 비디오 출력 스트림은, 클라이언트 디바이스(150)의 비디오 출력에 의해 별개의 디스플레이 디바이스에 송신될 수 있다.
- [0127] 비디오 소스 식별자 모듈(320)은, 동작 1540에서, 비디오 입력 스트림에서의 변경이, 비디오 입력 스트림의 콘텐츠 프로바이더의 변경을 지시함을 판정한다. 예를 들면, 쿼리 핑거프린트 생성기(160)는, 비디오 입력 스트림의 원래의 콘텐츠와 매칭되는 일련의 핑거프린트에 액세스할 수 있다. 쿼리 핑거프린트 생성기(160)에 의해 생성된 핑거프린트가 원래의 콘텐츠에 대한 일련의 핑거프린트와의 매칭을 중지할 경우, 비디오 입력 스트림의 콘텐츠가 변경되었음을 판정할 수 있고, 이것은, 비디오 입력 스트림의 콘텐츠 프로바이더가 또한 변경되었음을 지시할 수 있다. 예를 들면, 중간 디바이스의 유저는, (예를 들면, 케이블 TV 소스로부터 브로드캐스트 TV 소스로) 채널을 변경하거나 입력을 변경했을 수 있다. 어느 경우에도, 콘텐츠 프로바이더는 변경되었다.
- [0128] 비디오 입력 스트림의 콘텐츠 프로바이더가 변경되었음의 판정에 응답해서, 클라이언트 디바이스(150)는, 비디오 입력 스트림을 디스플레이 상에 제시시킨다(동작 1550). 따라서, 대체 비디오 출력 스트림이, 동작 1510에서 액세스된 원래의 비디오 입력 스트림에 대한 대체로서 제시되지만, 동작 1540에서 검출된 변경된 비디오 입력 스트림에 대해서는 아니다.
- [0129] 스텝 1560에서, 클라이언트 디바이스(150)는 중간 디바이스에 의해 생성된 스크린 부분에 대응하는 템플릿에 액세스한다. 예를 들면, 비디오 오버레이 모듈(330)은, 셋톱 박스(140)의 디바이스 식별자에 대한 데이터를 포함하는 도 16의 데이터베이스 스키마 내의 레코드에 액세스할 수 있다.
- [0130] 비디오 오버레이 모듈(330)은, 동작 1570에서, 비디오 입력 스트림을 분석해서, 비디오 입력 스트림의 프레임의 콘텐츠의 적어도 부분이 템플릿과 매칭됨을 판정한다. 일부 예시적인 실시예에서, 중간 디바이스에 대한 템플릿들 중의 어느 것이, 중간 디바이스로부터 수신된 비디오 입력 스트림의 프레임의 콘텐츠에 대응함의 여부를 판정하도록, 다수의 그러한 템플릿에 대해 반복된다. 템플릿은, 중간 디바이스에 의해 비디오 입력 스트림에 추가된 아이콘(예를 들면, 케이블 아이콘 또는 TV 아이콘), 중간 디바이스에 의해 비디오 입력 스트림에 추가된 미리 결정된 텍스트(예를 들면, 채널을 나타내는 숫자 또는 영숫자 시퀀스), 중간 디바이스에 의해 비디오 입력 스트림에 추가된 특정한 위치의 윈도우, 또는 그 임의의 적절한 조합에 대응할 수 있다.
- [0131] 동작 1580, 1590에서, 매칭에 의거하여, 클라이언트 디바이스(150)는, 비디오 입력 스트림의 제2 변경이 제1 콘텐츠 프로바이더로의 복귀를 지시함을 판정하고, 그 판정에 응답해서, 비디오 출력 스트림의 제시를 재개시킨다. 예를 들면, 원래의 비디오 입력 스트림이, 대체되어야 할 매칭 콘텐츠로서 식별된 채널 15로부터의 콘텐츠를 포함했을 경우, 동작 1530에서 대체 비디오 콘텐츠가 제시되었다. 채널의 변경 후에, 동작 1550에서 대체 비디오 콘텐츠의 제시는 중단되었다. (예를 들면, 현재의 채널이 15임을 지시하는 중간 디바이스에 의해 제공되는 오버레이에 대한 템플릿과 매칭시킴으로써) 원래의 채널로의 복귀의 검출 후에, 비디오 출력 스트림에서의 대체 비디오 콘텐츠의 제시가 재개된다(동작 1590). 따라서, 방법 1500을 사용함으로써, 대체되는 콘텐츠 프로바이더로부터 떠나는 스위칭 및 그에 대한 복귀에 의한 중단 없이, 유저가 다른 프로바이더에 대한 콘텐츠를 시청하게 하면서, 하나의 콘텐츠 프로바이더에 대한 대체 콘텐츠를 제공할 수 있다.
- [0132] 일부 예시적인 실시예에서, 비디오 출력 스트림은, 템플릿과 매칭되는 비디오 입력 스트림의 부분을 포함하도록 변경된다. 예를 들면, 도 6의 토폴로지 형태의 템플릿이, 셋톱 박스(140)에 의해 생성된 채널 정보를 포함하는 영역(610)과 매칭되면, 영역(610)을 비디오 입력 스트림으로부터 비디오 출력 스트림에 카피하여, 비디오 출력 스트림이 제시되는 동안 중간 디바이스에 의해 생성된 정보를 제시하게 할 수 있다.
- [0133] 도 16은, 예시적인 실시예에서, 비디오 오버레이를 동적으로 제공하기 위한 데이터 구조를 나타내는 블록도(1600)이다. 도 16은 PictureInPicture 테이블(1610), MenuOverlay 테이블(1620), MenuBlend 테이블(1630), 및 Template 테이블(1640)을 나타낸다.

- [0134] PictureInPicture 테이블(1610)은 디바이스 식별자(DeviceID), 템플릿 식별자(TemplateID), 및 윈도우 좌표(WindowCoords)에 대한 필드를 포함한다. 일부 예시적인 실시예에서, PictureInPicture 테이블(1610)은, 각 셋톱 박스에 대한 픽처-인-픽처 템플릿마다 하나의 행을 포함한다. 따라서, 특정한 셋톱 박스가 그 DeviceID에 의해 식별되고 특정한 템플릿이 그 TemplateID에 의해 식별되면, 대체 콘텐츠의 픽처-인-픽처 디스플레이의 사이즈 및 위치에 대한 좌표를 얻을 수 있다. PictureInPicture 테이블(1610)은, 도 4의 템플릿과 토폴로지적으로 유사한 템플릿을 지원하는데 사용될 수 있다.
- [0135] MenuOverlay 테이블(1620)은 디바이스 식별자, 템플릿 식별자, 및 메뉴 좌표(MenuCoords)에 대한 필드를 포함한다. 일부 예시적인 실시예에서, MenuOverlay 테이블(1620)은, 각 셋톱 박스에 대한 메뉴 오버레이 템플릿마다 하나의 행을 포함한다. 따라서, 특정한 셋톱 박스가 그 DeviceID에 의해 식별되고 특정한 템플릿이 그 TemplateID에 의해 식별되면, 메뉴 오버레이에 의해 점유되는 비디오의 부분의 사이즈 및 위치의 좌표는 MenuCoords로부터 판정될 수 있다. MenuOverlay 테이블(1620)은, 도 5 및 도 6의 템플릿과 토폴로지적으로 유사한 템플릿을 지원하는데 사용될 수 있다.
- [0136] MenuBlend 테이블(1630)은 디바이스 식별자, 템플릿 식별자, 메뉴 좌표, 및 블렌드 좌표(BlendCoords)에 대한 필드들을 포함한다. 일부 예시적인 실시예에서, MenuBlend 테이블(1630)은, 각 셋톱 박스의 메뉴 블렌드 템플릿마다 하나의 행을 포함한다. 따라서, 특정한 셋톱 박스가 그 DeviceID에 의해 식별되고 특정한 템플릿이 그 TemplateID에 의해 식별되면, 메뉴 오버레이에 의해 점유되는 비디오의 부분의 사이즈 및 위치에 대한 좌표는 MenuCoords로부터 판정되고, 셋톱 박스로부터의 비디오와 대체 비디오를 블렌딩함으로써 생성되는 스크린의 부분의 사이즈 및 위치에 대한 좌표는 BlendCoords로부터 판정될 수 있다. MenuBlend 테이블(1630)은, 도 7의 템플릿에 토폴로지적으로 유사한 템플릿을 지원하도록 사용될 수 있다. 일부 예시적인 실시예에서, BlendCoords가 제로 에어리어의 영역을 정의할 경우, MenuBlend 테이블(1630)을 사용해서, 도 5 및 도 6의 템플릿과 토폴로지적으로 유사한 템플릿을 지원할 수 있으므로, MenuOverlay 테이블(1620)은 MenuBlend 테이블(1630)에 포함된다.
- [0137] 템플릿 테이블(1640)은 디바이스 식별자, 템플릿 식별자, 타입, 비트맵, 및 화상에 대한 필드를 포함한다. 비트맵은 비디오 입력 스트림의 프레임 및 화상과 동일한 사이즈이다. 비트맵의 각 값은, 템플릿이 비디오 입력 스트림과 매칭되는지의 여부를 판정하기 위해, 비디오 입력 스트림 및 화상 내의 대응하는 픽셀을 사용할지의 여부를 지시한다. 비트맵의 "1" 값에 대응하는 화상의 부분은 비디오 입력 스트림과 비교된다. 비트맵의 "0" 값에 대응하는 화상의 부분은 흑색이다. 타입은, 매칭이 발견되면 사용하기 위한 템플릿의 타입을 지시한다. 예를 들면, PICTUREINPICTURE의 타입은 템플릿이 픽처-인-픽처 표시를 위한 것이고, 임베디드 픽처에 대한 윈도우 좌표가 PictureInPicture 테이블(1610)로부터 얻어질 수 있음을 지시한다.
- [0138] 다양한 예시적인 실시예에 따르면, 본 명세서에서 설명되는 하나 이상의 방법론은, 동적 비디오 오버레이를 용이하게 할 수 있다. 또한, 본 명세서에 설명되는 하나 이상의 방법론은, 클라이언트 디바이스가 동적 비디오 오버레이를 제공하는 방법을 행하는 계산 시간, 클라이언트 디바이스와 서버 사이의 전송 시간, 동적 비디오 오버레이의 방법을 행하는 서버 상의 메모리 스토리지 요건, 및 동적 비디오 오버레이의 방법을 행하는 서버 상에서의 계산 시간을 저감할 수 있다.
- [0139] 이들 영향을 종합적으로 고려될 경우, 본 명세서에 기재된 하나 이상의 방법론은, 다른 방식에서 동적 비디오 오버레이에서 수반될 특정한 노력 또는 리소스에 대한 필요성을 제거할 수 있다. 동적 비디오 오버레이에서 유저가 들이는 노력은, 본 명세서에 기재된 하나 이상의 방법론에 의해 저감될 수 있다. (예를 들면, 네트워크 환경(100) 내) 하나 이상의 머신, 데이터베이스, 또는 디바이스에 의해 사용되는 컴퓨팅 리소스도 마찬가지로 저감될 수 있다. 그러한 컴퓨팅 리소스의 예는, 프로세서 사이클, 네트워크 트래픽, 메모리 사용량, 데이터 스토리지 용량, 전력 소비, 및 냉각 능력이 포함한다.
- [0140] 도 17은 머신 관독 가능한 매체(예를 들면, 머신 관독 가능한 스토리지 매체, 컴퓨터 관독 가능한 스토리지 매체, 또는 그 임의의 적절한 조합)로부터 명령어를 관독하고 본 명세서에서 설명되는 어느 하나 이상의 방법론을 전체적으로 또는 부분적으로 행할 수 있는, 일부 예시적인 실시예에 따른 머신(1700)의 컴포넌트를 나타내는 블록도이다. 구체적으로, 도 17은, 컴퓨터 시스템의 예시적인 형태의 머신(1700)의 도식적 표현을 나타내고, 그 내에서, 본 명세서에서 논해지는 어느 하나 이상의 방법론을 머신(1700)에 실행시키는 명령어(1724)(예를 들면, 소프트웨어, 프로그램, 애플리케이션, 애플릿, 앱, 또는 다른 실행 가능한 코드)가 전체적으로 또는 부분적으로 실행될 수 있다. 대안의 실시예에서, 머신(1700)은 스탠드얼론 디바이스로서 동작하거나, 또는 다른 머신에 연결(예를 들면, 네트워킹)될 수 있다. 네트워킹 배치에서, 머신(1700)은, 서버-클라이언트 네트워크 환경에서

서버 머신 또는 클라이언트 머신의 능력으로, 또는 분산형(예를 들면, 피어-투-피어) 네트워크 환경에서 피어 머신으로서 동작할 수 있다. 머신(1700)은 서버 컴퓨터, 클라이언트 컴퓨터, PC, 태블릿 컴퓨터, 랩톱 컴퓨터, 넷북, 셋톱 박스, 스마트 TV, PDA(personal digital assistant), 셀룰러 전화, 스마트폰, 웹 어플라이언스, 네트워크 라우터, 네트워크 스위치, 네트워크 브리지, 또는 해당 머신이 취해야 할 액션을 지정하는 명령어(1724)를 순차적으로 또는 다른 방식으로 실행할 수 있는 임의의 머신일 수 있다. 또한, 단일 머신만이 도시되어 있지만, 용어 "머신"은 또한, 본 명세서에서 논의되는 방법론 중 어느 하나 이상의 모두 또는 일부를 행하도록 명령어(1724)를 개개로 또는 공동해서 실행하는 머신의 집합을 포함하는 것으로 간주되어야 한다.

[0141] 머신(1700)은 프로세서(1702)(예를 들면, CPU(central processing unit), GPU(graphics processing unit), DSP(digital signal processor), ASIC, RFIC(radio-frequency integrated circuit), 또는 그 임의의 적절한 조합), 메인 메모리(1704), 및 정적 메모리(1706)를 포함하고, 이들은 버스(1708)를 통해 서로 통신하도록 구성된다. 머신(1700)은 그래픽 디스플레이(예를 들면, PDP(plasma display panel), LED 디스플레이, LCD, 프로젝터, 또는 CRT)를 더 포함할 수 있다. 머신(1700)은 또한, 영숫자 입력 디바이스(1712)(예를 들면, 키보드), 비디오 입력 디바이스(1714)(예를 들면, 컴포넌트 비디오 입력, 콤포지트 비디오 입력, 또는 HDMI(high-definition multimedia interface)), 스토리지 유닛(1716), 신호 생성 디바이스(1718)(예를 들면, 스피커), 및 네트워크 인터페이스 디바이스(1720)를 포함할 수 있다.

[0142] 스토리지 유닛(1716)은, 본 명세서에 기재된 방법론 또는 기능 중의 어느 하나 이상을 구현하는 명령어(1724)가 저장된 머신 판독 가능한 매체(1722)를 포함한다. 명령어(1724)는 또한, 머신(1700)에 의한 실행 중에, 메인 메모리(1704) 내, 프로세서(1702) 내(예를 들면, 프로세서의 캐시 메모리 내), 또는 그 모두에, 완전하게 또는 적어도 부분적으로 존재할 수 있다. 따라서, 메인 메모리(1704) 및 프로세서(1702)는 머신 판독 가능한 매체로서 간주될 수 있다. 명령어(1724)는 네트워크(1726)(예를 들면, 도 1의 네트워크(170))에서 네트워크 인터페이스 디바이스(1720)를 통해 전송 또는 수신될 수 있다.

[0143] 본 명세서에서 사용될 경우, 용어 "메모리"는 데이터를 일시적으로 또는 영구적으로 저장할 수 있는 머신 판독 가능한 매체를 의미하고, 제한이 아닌 예시로서, RAM(random-access memory), ROM(read-only memory), 버퍼 메모리, 플래시 메모리, 및 캐시 메모리를 포함하는 것으로 간주될 수 있다. 예시적인 실시예에서, 머신 판독 가능한 매체(1722)는 단일의 매체인 것으로 나타나 있지만, 용어 "머신 판독 가능한 매체"는 명령어를 저장할 수 있는 단일의 매체 또는 다수의 매체(예를 들면, 집중형 데이터베이스 또는 분산형 데이터베이스, 또는 연관 캐시 및 서버)를 포함하는 것으로 간주될 수 있다. 용어 "머신 판독 가능한 매체"는 또한, 머신(예를 들면, 머신(1700))에 의한 실행을 위한 명령어를 저장할 수 있는 임의의 매체 또는 다수의 매체의 조합을 포함하는 것으로 간주될 수 있고, 명령어는, 머신의 하나 이상의 프로세서(예를 들면, 프로세서(1702))에 의한 실행 시, 본 명세서에 기재된 방법론 중의 어느 하나 이상을 머신에 실행시킨다. 따라서, "머신 판독 가능한 매체"는 단일 스토리지 장치 또는 디바이스뿐만 아니라, 다수의 스토리지 장치 또는 디바이스를 포함하는 "클라우드 기반" 스토리지 시스템 또는 스토리지 네트워크를 의미한다. 따라서, 용어 "머신 판독 가능한 매체"는, 제한이 아닌 예시로서, 솔리드 스테이트 메모리, 광학 매체, 자기 매체, 또는 그 임의의 적절한 조합의 형태의 하나 이상의 데이터 레포지토리를 포함하는 것으로 간주되어야 한다. "비일시적인 머신 판독 가능한 매체"는, 일시적인 신호를 제외한 모든 머신 판독 가능한 매체를 의미한다.

[0144] 본 명세서 전반에 걸쳐, 복수의 인스턴스는, 단일의 인스턴스로서 기술되는 컴포넌트, 동작, 또는 구조를 실현할 수 있다. 하나 이상의 방법의 개개의 동작이 개개의 동작으로서 도시 및 설명되어 있지만, 하나 이상의 개개의 동작이 동시에 행해져도 되고, 조작이 도시된 순서로 실행될 필요는 없다. 예시적인 구성에서 별개의 컴포넌트로서 제시되는 구조 및 기능은, 조합된 구조 또는 컴포넌트로서 실시되어도 된다. 마찬가지로, 단일의 컴포넌트로서 제시되는 구조 및 기능은 별개의 컴포넌트로서 실시되어도 된다. 이들 및 다른 변형, 변경, 추가 및 개선은 본 명세서의 주제의 범위 내에 들어간다.

[0145] 특정한 실시예는, 본 명세서에서, 로직 또는 다수의 컴포넌트, 모듈, 또는 메커니즘을 포함하는 것으로서 기재된다. 모듈은 하드웨어 모듈을 구성할 수 있다. "하드웨어 모듈"은 특정한 동작을 행할 수 있는 유형의 유닛이고, 특정한 물리적인 방법으로 구성 또는 배치될 수 있다. 다양한 예시적인 실시예에서, 하나 이상의 컴퓨터 시스템(예를 들면, 스탠드얼론 컴퓨터 시스템, 클라이언트 컴퓨터 시스템, 또는 서버 컴퓨터 시스템) 또는 컴퓨터 시스템의 하나 이상의 하드웨어 모듈(예를 들면, 프로세서 또는 프로세서의 그룹)은, 본 명세서에서 설명되는 특정한 동작을 행하도록 동작하는 하드웨어 모듈로서 소프트웨어(예를 들면, 애플리케이션 또는 애플리케이션 부분)에 의해 구성될 수 있다.

- [0146] 일부 실시예에서, 하드웨어 모듈은 머신적으로, 전자적으로, 또는 그 임의의 적절한 조합으로 실시될 수 있다. 예를 들면, 하드웨어 모듈은 특정한 동작을 행하도록 영구적으로 구성되는 전용의 회로 또는 로직을 포함할 수 있다. 예를 들면, 하드웨어 모듈은, FPGA 또는 ASIC 등의 특수 목적용 프로세서일 수 있다. 하드웨어 모듈은 또한, 특정한 동작을 행하도록 소프트웨어에 의해 일시적으로 구성되는 프로그래머블 로직 또는 회로를 포함할 수 있다. 예를 들면, 하드웨어 모듈은, 범용 프로세서 또는 다른 프로그래머블 프로세서 내에 포함되는 소프트웨어를 포함할 수 있다. 전용 및 영구적으로 구성된 회로에서, 또는 일시적으로 구성된 회로에서(예를 들면, 소프트웨어에 의해 구성됨) 하드웨어 모듈을 기계적으로 실현한다는 결정은, 비용 및 시간의 고려에 의해 진행될 수 있음이 이해될 것이다.
- [0147] 따라서, 어구 "하드웨어 모듈"은, 특정한 방식으로 동작하거나 본 명세서에 기재된 특정한 동작을 행하도록, 물리적으로 구성되거나, 영구적으로 구성되거나(예를 들면, 하드와이어드), 또는 일시적으로 구성되는(예를 들면, 프로그래밍되는) 유형의 엔티티를 포함함을 이해해야 한다. 본 명세서에서 사용되는 "하드웨어 구현 모듈"은 하드웨어 모듈을 의미한다. 하드웨어 모듈이 일시적으로 구성되는(예를 들면, 프로그래밍되는) 실시예를 고려하면, 각 하드웨어 모듈은, 어느 하나의 시간에 구성 또는 인스턴스화될 필요는 없다. 예를 들면, 하드웨어 모듈이, 특수 목적용 프로세서가 되도록 소프트웨어에 의해 구성된 범용 프로세서를 포함할 경우, 범용 프로세서는, 서로 다른 시간에서 각각 서로 다른 특수 목적용 프로세서(예를 들면, 서로 다른 하드웨어 모듈을 포함함)로서 구성될 수 있다. 따라서, 소프트웨어는, 예를 들면 하나의 시점에 특정한 하드웨어 모듈을 구성하고 다른 시점에 다른 하드웨어 모듈을 구성하도록, 프로세서를 구성할 수 있다.
- [0148] 하드웨어 모듈은 다른 하드웨어 모듈에 대해 정보를 주고받을 수 있다. 따라서, 기재된 하드웨어 모듈들은 통신 가능하게 연결되는 것으로 간주될 수 있다. 다수의 하드웨어 모듈이 동시에 존재할 경우, 통신은, 2개 이상의 하드웨어 모듈들간의 신호 전송을 통해 (예를 들면, 적절한 회로 및 버스를 통해) 달성될 수 있다. 다수의 하드웨어 모듈이 서로 다른 시간에 구성 또는 인스턴스화되는 실시예에서, 그러한 하드웨어 모듈들간의 통신은, 예를 들면 다수의 하드웨어 모듈이 액세스하는 메모리 구조 내의 정보의 저장 및 검색에 의해 달성될 수 있다. 예를 들면, 하나의 하드웨어 모듈은 동작을 행하고, 그 동작의 출력을, 통신 가능하게 연결된 메모리 디바이스에 저장할 수 있다. 이어서, 나중에, 추가 하드웨어 모듈이 메모리 디바이스에 액세스해서 저장된 출력을 검색 및 처리할 수 있다. 하드웨어 모듈은 또한, 입력 디바이스 또는 출력 디바이스와 통신을 개시하고, 리소스(예를 들면, 정보의 집합)를 조작할 수도 있다.
- [0149] 본 명세서에 기재되는 예시적 방법의 다양한 동작은, 관련 동작을 행하도록(예를 들면, 소프트웨어에 의해) 일시적으로 구성되거나, 또는 영구적으로 구성되는 하나 이상의 프로세서에 의해 적어도 부분적으로 행해질 수 있다. 일시적으로 또는 영구적으로 구성되는지에 상관 없이, 이러한 프로세서는, 본 명세서에서 설명되는 하나 이상의 동작 또는 기능을 행하도록 동작하는 프로세서 구현 모듈을 구성할 수 있다. 본 명세서에서 사용되는 "프로세서 구현 모듈"은, 하나 이상의 프로세서를 사용해서 구현되는 하드웨어 모듈을 의미한다.
- [0150] 마찬가지로, 본 명세서에서 설명하는 방법은 적어도 부분적으로 프로세서 실현될 수 있고, 프로세서는 하드웨어의 일례이다. 예를 들면, 방법의 동작의 적어도 일부는, 하나 이상의 프로세서 또는 프로세서 구현 모듈에 의해 행해질 수 있다. 또한, 하나 이상의 프로세서는, "클라우드 컴퓨팅" 환경에서 또는 "SaaS(software as a service)"로서 관련 동작의 퍼포먼스를 지원하도록 동작할 수도 있다. 예를 들면, 동작의 적어도 일부는 (프로세서를 포함하는 머신들의 예로서) 컴퓨터의 그룹에 의해 행해질 수 있고, 이들 동작은 네트워크(예를 들면, 인터넷)를 통해, 및 하나 이상의 적절한 인터페이스(예를 들면, API)를 통해 액세스 가능하다.
- [0151] 특정한 동작의 퍼포먼스는, 단일의 머신 내에 존재할뿐만 아니라, 다수의 머신에 걸쳐 배치된 하나 이상의 프로세서 중에 분산되어도 된다. 일부 예시적인 실시예에서, 하나 이상의 프로세서 또는 프로세서 구현 모듈은, 단일 지리적 위치(예를 들면, 가정 환경, 오피스 환경, 또는 서버 팜 내)에 위치될 수 있다. 다른 예시적인 실시예에서, 하나 이상의 프로세서 또는 프로세서 구현 모듈은, 다수의 지리적 위치에 걸쳐 분산될 수 있다.
- [0152] 본 명세서에서 논의되는 주제의 일부는 머신 메모리(예를 들면, 컴퓨터 메모리) 내에 비트 또는 2진 디지털 신호로서 저장된 데이터에 대한 연산의 알고리즘 또는 심볼 표현의 관점에서 제시될 수 있다. 이러한 알고리즘 또는 심볼 표현은, 데이터 처리 기술 분야에서 통상의 지식을 가진자가 그들의 작업 내용을 다른 통상의 지식을 가진자에게 전달하기 위해 사용되는 기술의 예이다. 본 명세서에 사용되는 "알고리즘"은 원하는 결과에 도달하는 자기-일관된 시퀀스의 연산 또는 유사한 처리이다. 이러한 맥락에서, 알고리즘 및 연산은 물리량의 물리적 조작을 수반한다. 통상적으로, 반드시 그러한 것은 아니지만, 그러한 양은 기계에 의해 저장, 액세스, 전달, 조합, 비교, 또는 달리 조작될 수 있는 전기, 자기, 또는 광학 신호의 형태를 취할 수 있다. 때때로 주로, "데

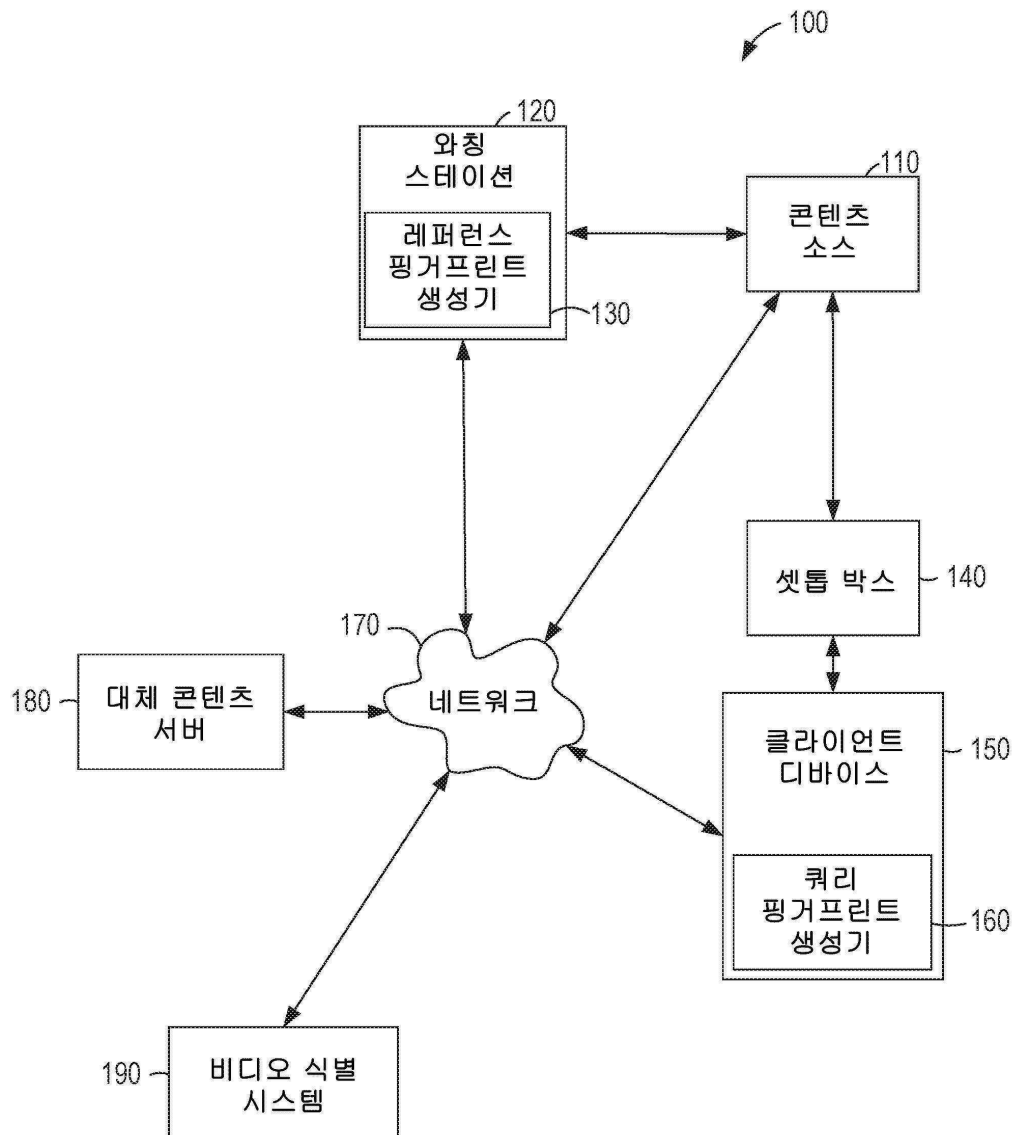
이터", "콘텐츠", "비트", "값", "요소", "심볼", "문자", "용어", "번호", "수치" 등과 같은 단어를 사용하여 이러한 신호를 기술하는 것이 주로 공통된 사용의 이유로 인해 편리하다. 그러나, 이 단어들은 단지 편리한 레이블일 뿐이며 적절한 물리량과 연관될 것이다.

[0153]

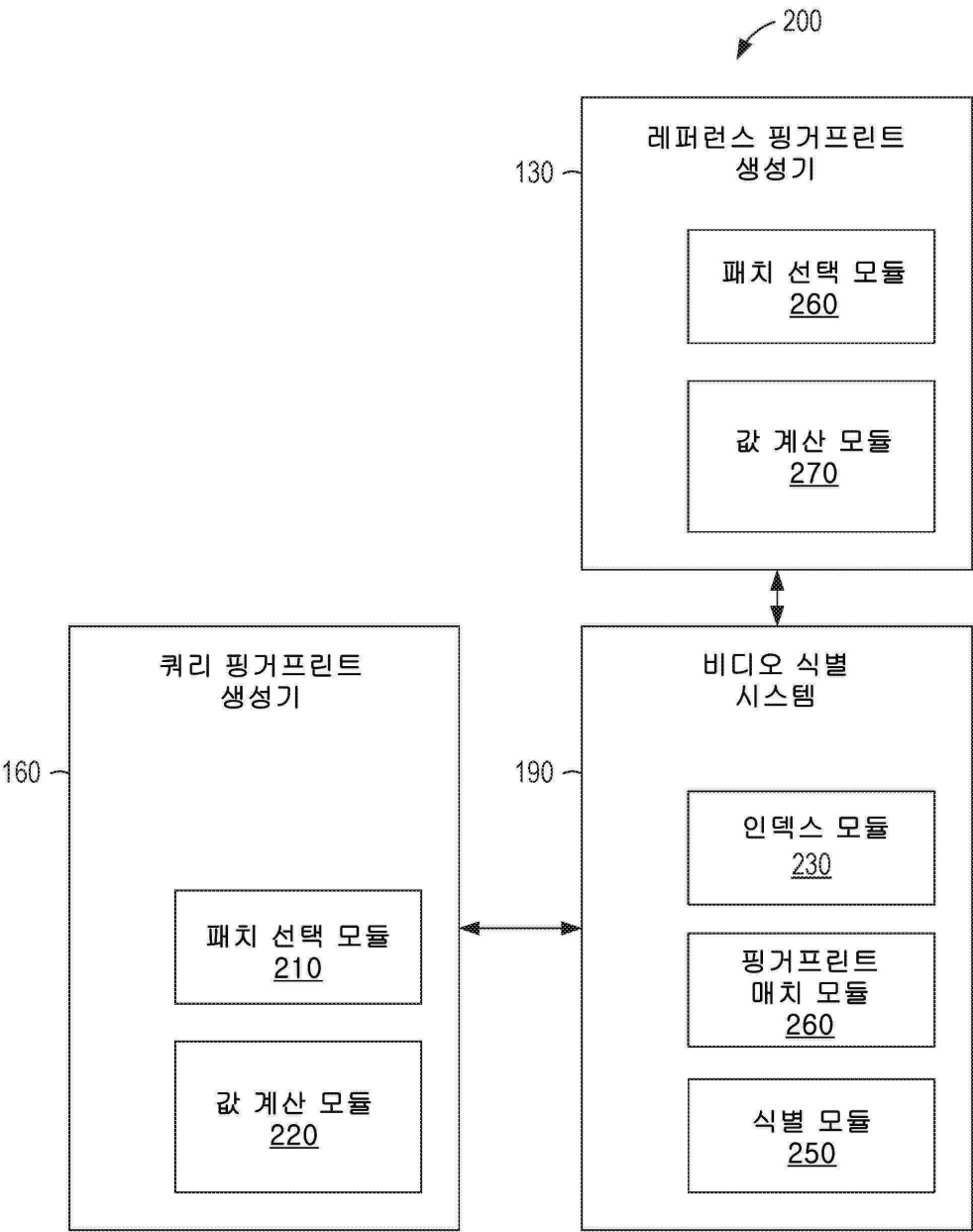
달리 특별히 언급하지 않는 한, "처리하는", "컴퓨팅하는", "계산하는", "판정하는", "제시하는", "표시하는" 등의 단어를 사용한 본 명세서에서의 논의는, 하나 이상의 메모리(예를 들면, 휘발성 메모리, 비휘발성 메모리, 또는 그 임의의 적절한 조합), 레지스터, 또는 정보를 수신, 저장, 전송, 또는 표시하는 다른 머신 컴포넌트 내에서 물리적(예를 들면, 전자적, 자기적, 또는 광학적) 양으로서 표시되는 데이터를 조작 또는 변환하는 머신(예를 들면, 컴퓨터)의 동작 또는 프로세스를 의미할 수 있다. 또한, 달리 특별히 언급하지 않는 한, 단수 표현(용어 "a" 또는 "an")은, 특허문헌에서 일반적인 바와 같이, 하나 이상의 인스턴스를 포함하도록 본 명세서에서 사용된다. 마지막으로, 본 명세서 중에서 사용되는 연결사 "또는"는, 달리 특별히 언급하지 않는 한, 비배타적인 "또는"을 의미한다.

도면

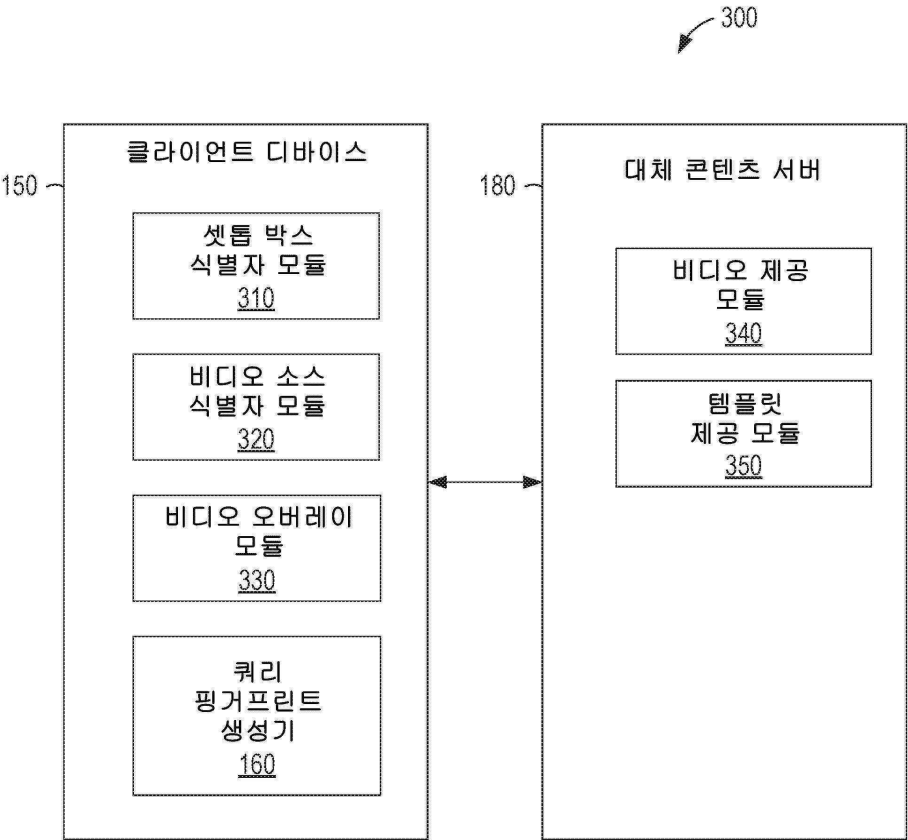
도면1



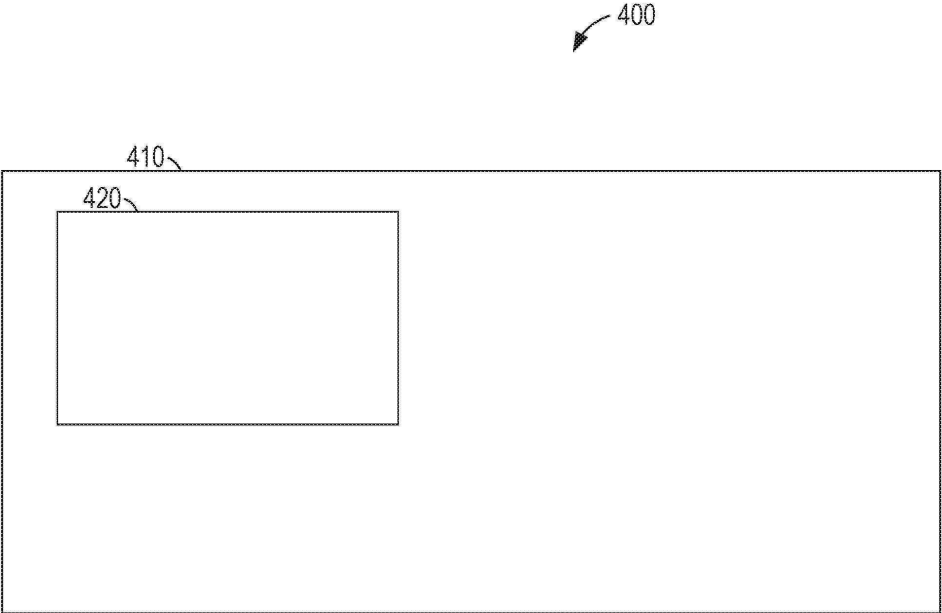
도면2



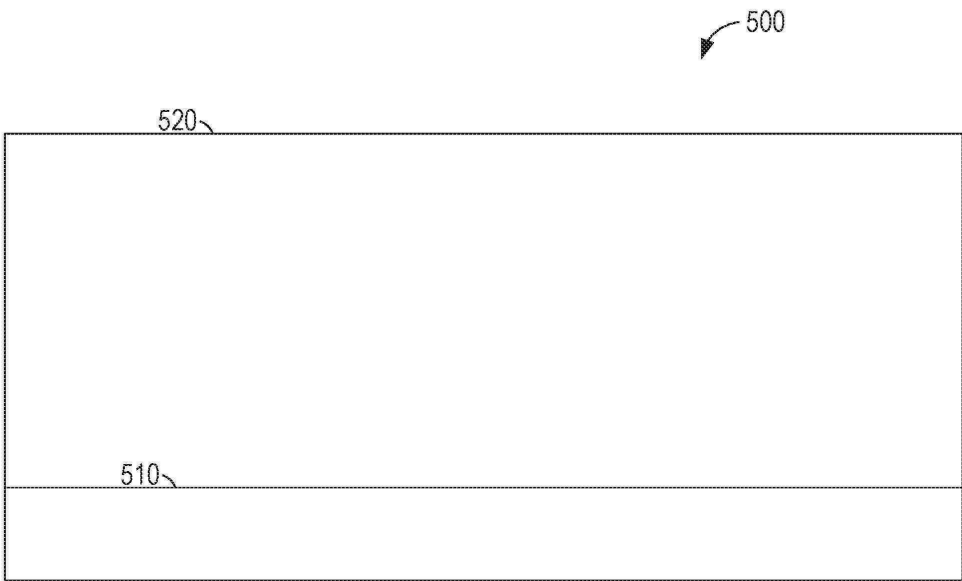
도면3



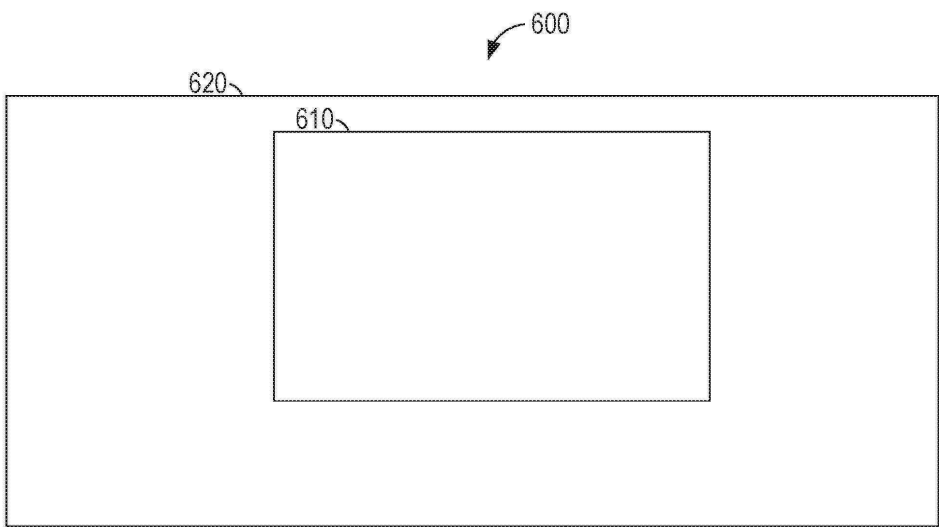
도면4



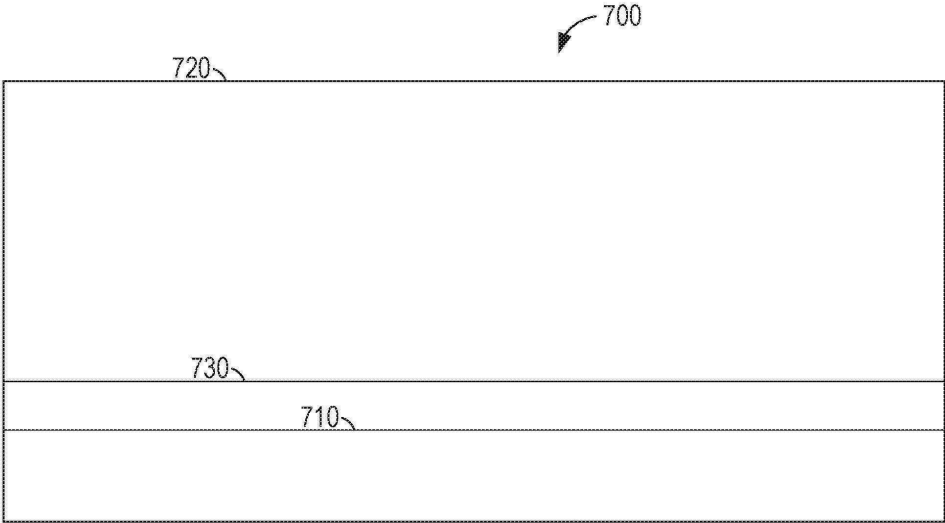
도면5



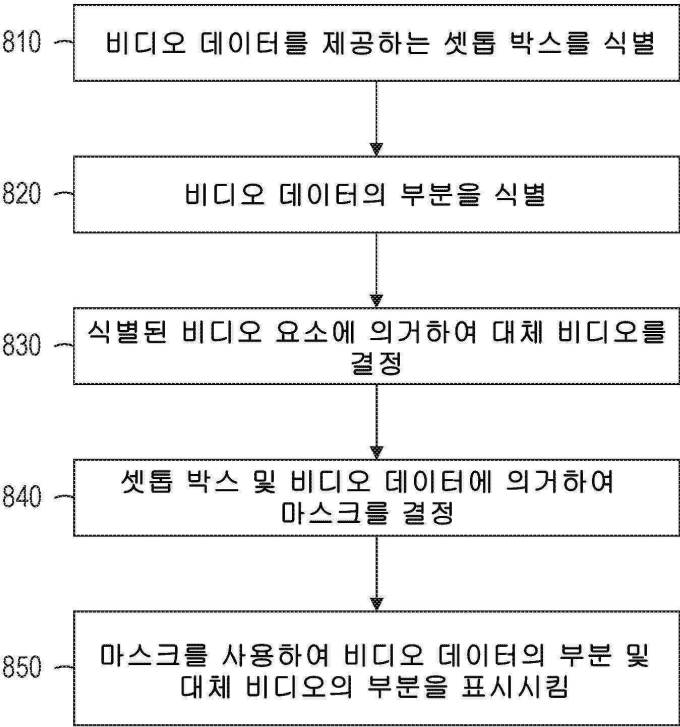
도면6



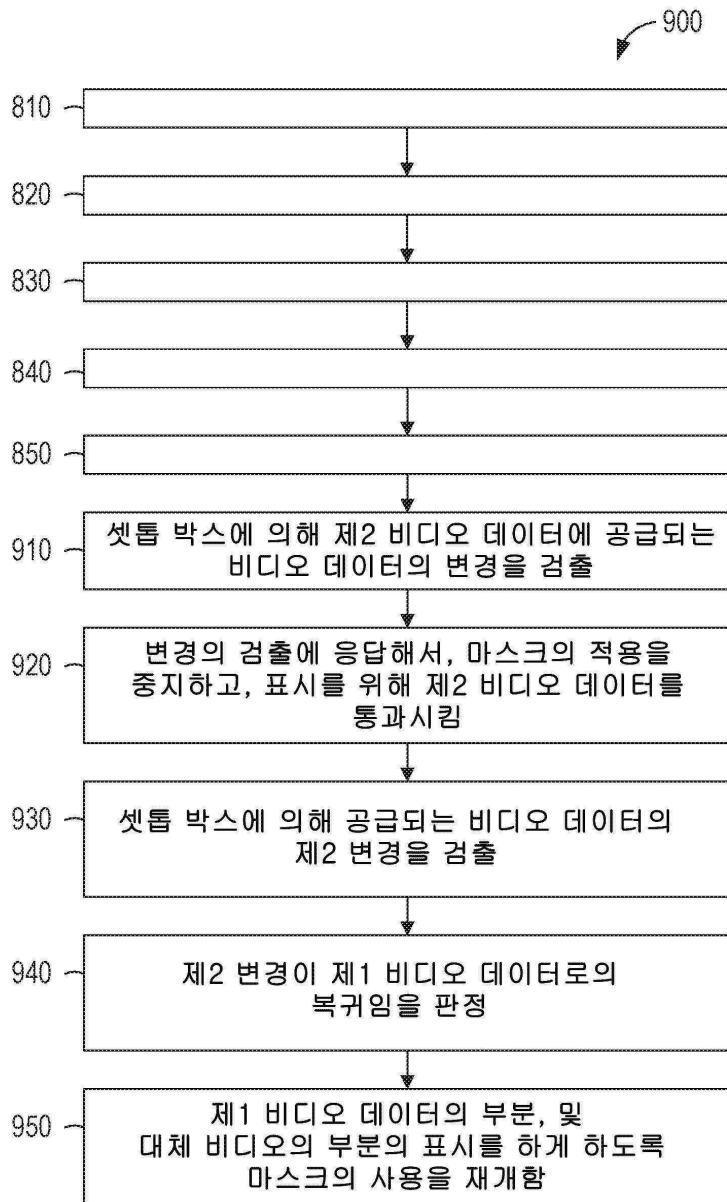
도면7



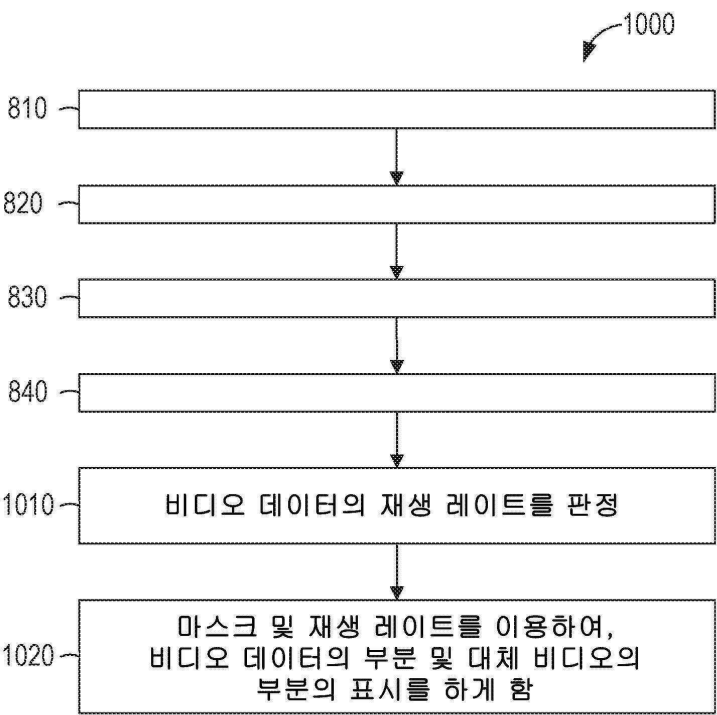
도면8



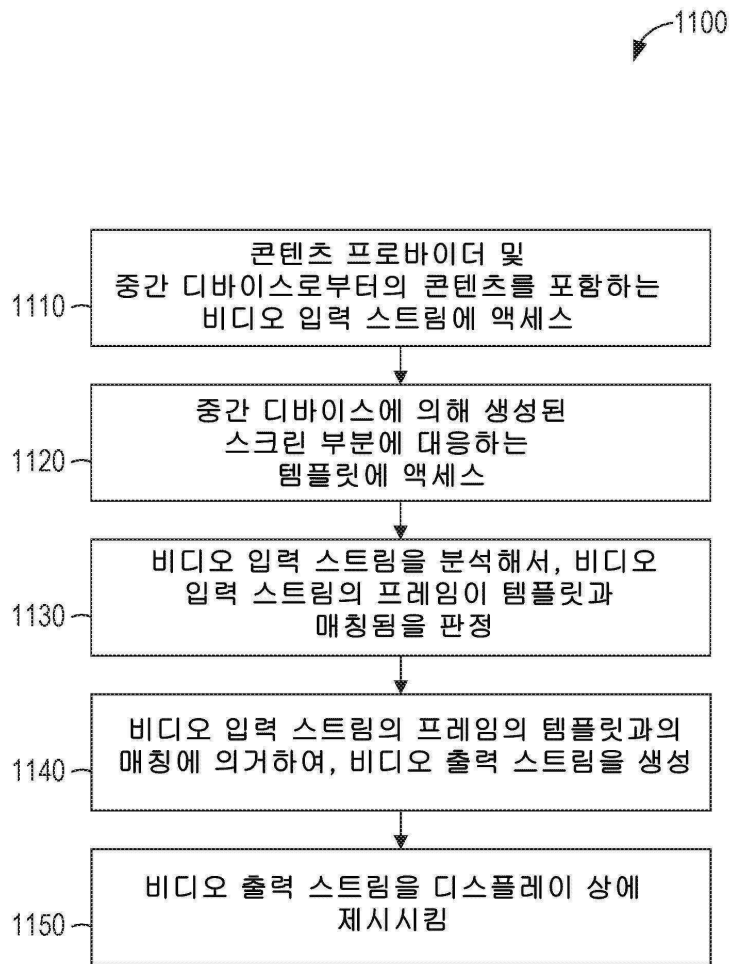
도면9



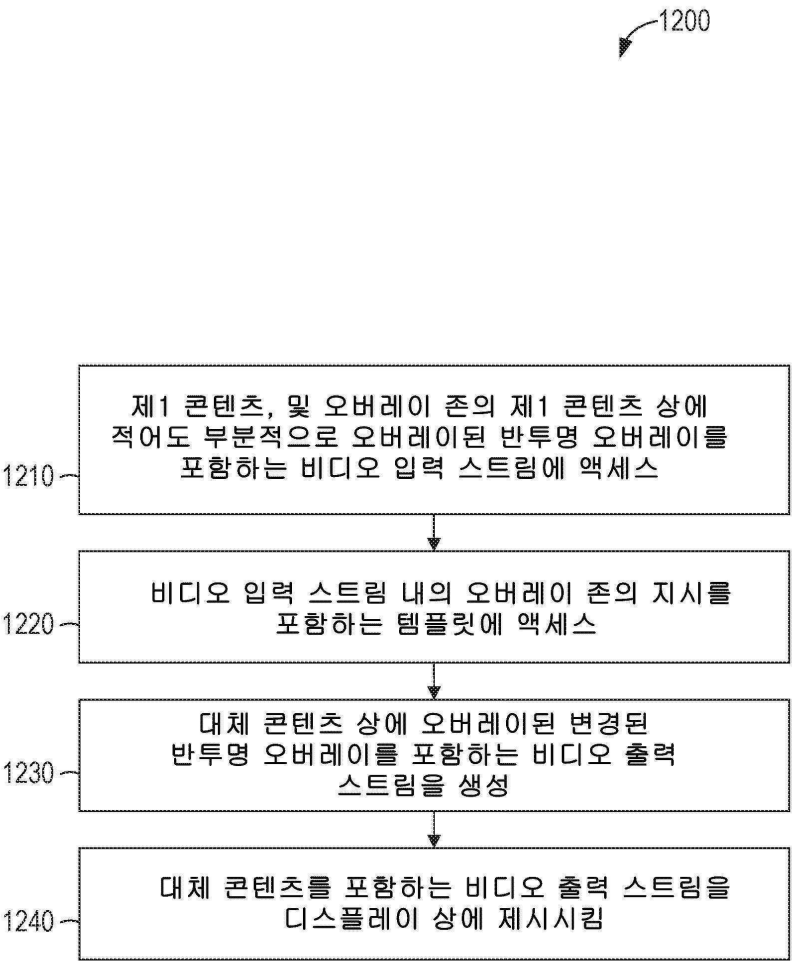
도면10



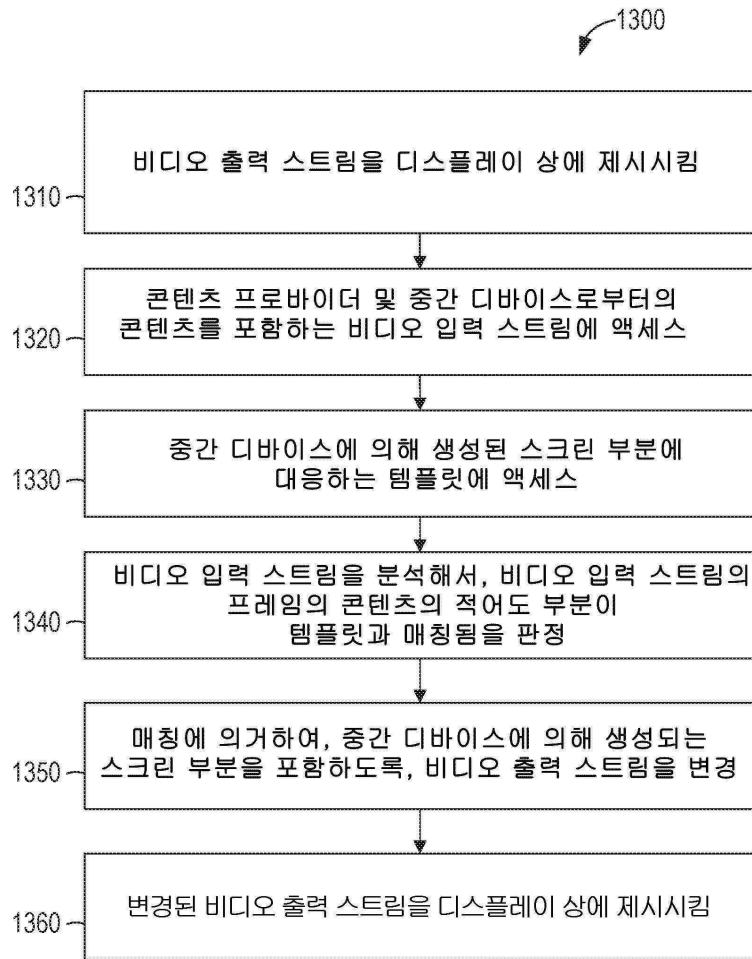
도면11



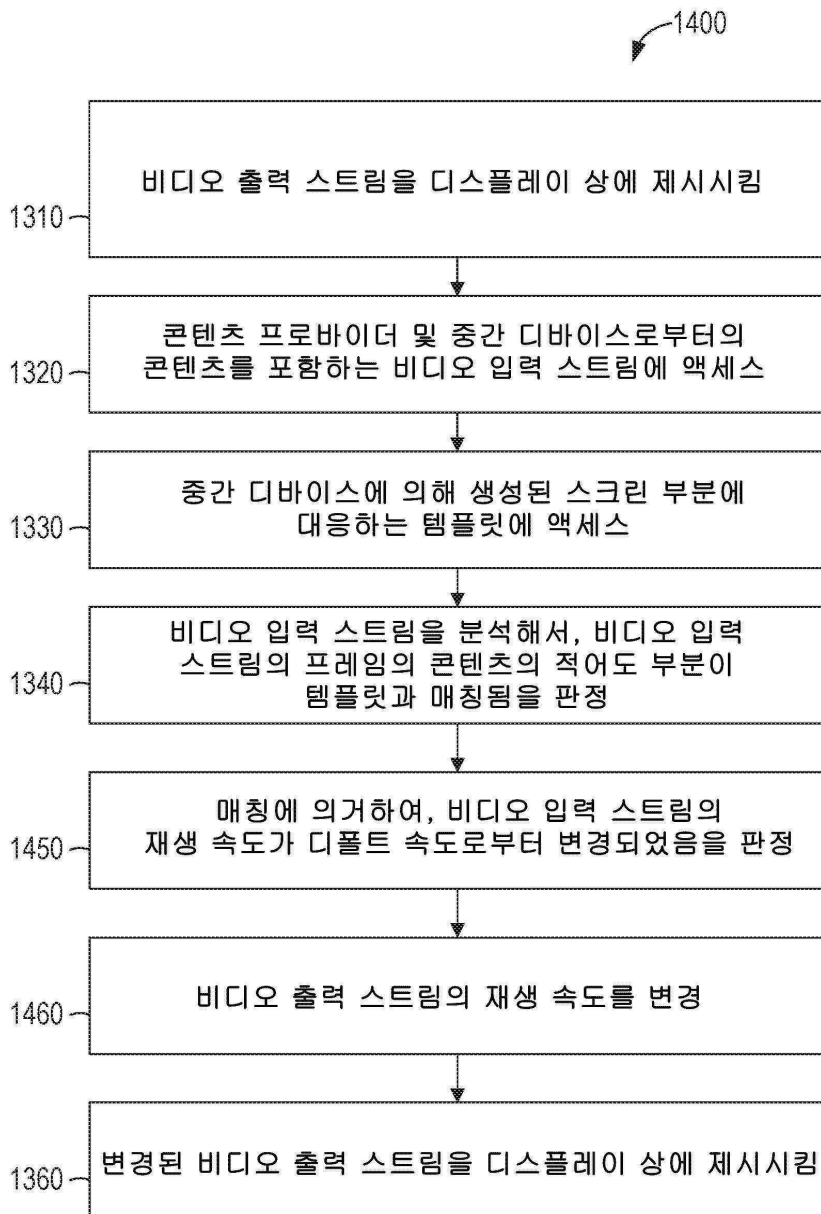
도면12



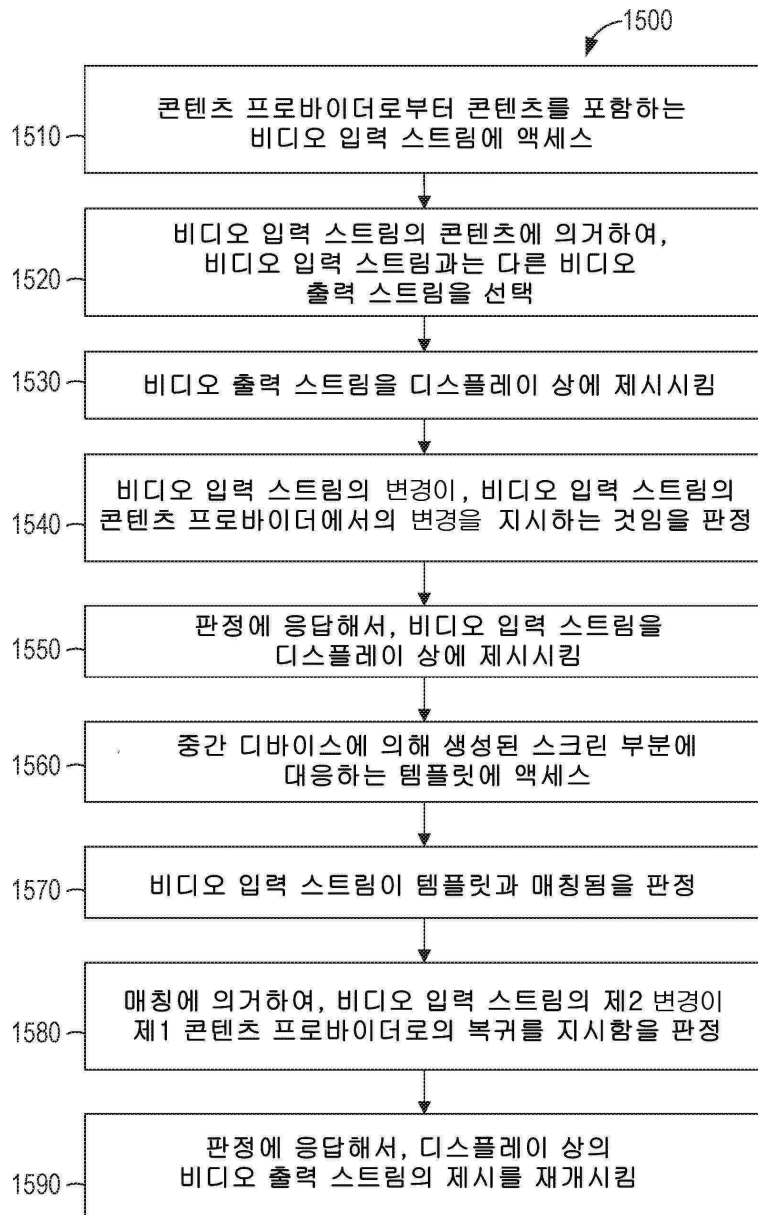
도면13



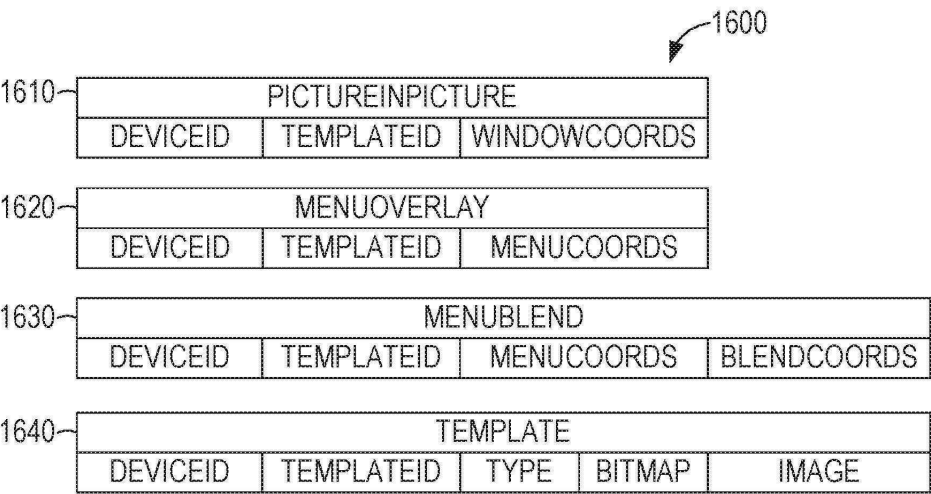
도면14



도면15



도면16



도면17

