(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2017년10월19일
(11) 등록번호 10-1788330
(24) 등록일자 2017년10월13일

(51) 국제특허분류(Int. Cl.)
H04N 7/24 (2011.01) *H04N 19/107* (2014.01)
H04N 19/169 (2014.01) *H04N 19/436* (2014.01)
H04N 21/2343 (2011.01) *H04N 21/2381* (2011.01)
H04N 21/266 (2011.01) *H04N 21/6377* (2011.01)
H04N 21/658 (2011.01)
(21) 출원번호 10-2011-7024936
(22) 출원일자(국제) 2010년03월17일
심사청구일자 2015년03월17일
(85) 번역문제출일자 2011년10월21일
(65) 공개번호 10-2011-0132608
(43) 공개일자 2011년12월08일
(86) 국제출원번호 PCT/US2010/027716
(87) 국제공개번호 WO 2010/111096
국제공개일자 2010년09월30일
(30) 우선권주장
12/538,054 2009년08월07일 미국(US)
61/210,888 2009년03월23일 미국(US)
(56) 선행기술조사문헌
KR1020060079085 A*
US20020004838 A1
US20060146830 A1
US20070009029 A1
*는 심사관에 의하여 인용된 문헌

(73) 특허권자
소니 인터랙티브 엔터테인먼트 아메리카 엘엘씨
미국 캘리포니아 (우편번호 94404) 산 마테오 브릿지포인트 파크웨이 2207
(72) 발명자
펄만, 스테판, 지.
미국, 캘리포니아 94301, 팔로 알토, 리튼 애비뉴 181
반 데 란, 로저
미국, 캘리포니아 94301, 팔로 알토, 리튼 애비뉴 181
(뒷면에 계속)
(74) 대리인
청운특허법인

전체 청구항 수 : 총 18 항 심사관 : 장석환

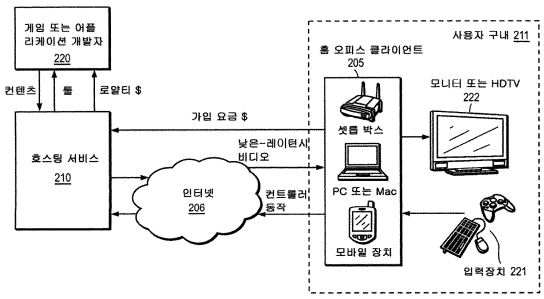
(54) 발명의 명칭 **클라이언트 장치로부터의 피드백 정보에 기초하여 비디오 프레임 또는 그 일부를 압축하기 위한 시스템 및 방법**

(57) 요 약

비디오 압축을 수행하기 위한 컴퓨터-실행 시스템 및 방법이 기술된다. 예컨대, 본 발명의 일 실시예에 따른 방법은: 제1인코딩 포맷에 따라 다수의 비디오 프레임 또는 그 일부를 인코딩하는 단계; 다수의 인코드된 비디오 프레임 또는 일부를 클라이언트 장치로 전송하는 단계; 상기 비디오 프레임 또는 일부에 포함된 데이터가 성공적

(뒷면에 계속)

대 표 도



으로 수신 및/또는 디코드되지 않았는지를 결정하기 위해 사용할 수 있는 피드백 정보를 클라이언트 장치로부터 수신하는 단계; 비디오 프레임 또는 그 일부가 성공적으로 수신 및/또는 디코드되지 않은 것을 검출함에 따라, 제2인코딩 포맷에 따라 비디오 프레임 또는 그 일부를 인코딩하는 단계; 및 비디오 프레임 또는 그 일부를 클라이언트 장치로 전송하는 단계를 포함한다.

(72) 발명자

코터, 티모시

미국, 캘리포니아 94301, 팔로 알토, 리튼 애비뉴 181

퍼먼, 스콧

미국, 캘리포니아 94301, 팔로 알토, 리튼 애비뉴 181

맥쿨, 로버트

미국, 캘리포니아 94301, 팔로 알토, 리튼 애비뉴 181

버클리, 이안

미국, 캘리포니아 94301, 팔로 알토, 리튼 애비뉴 181

명 세 서

청구범위

청구항 1

컴퓨터가 비디오 압축을 수행하기 위해 실행하는 방법에 있어서,

다수의 인코드된 비디오 프레임 또는 그 일부를 생성하기 위해 제1인코딩 포맷에 따라 다수의 비디오 프레임 또는 그 일부를 인코딩하는 단계;

상기 다수의 인코드된 비디오 프레임 또는 그 일부를 클라이언트 장치로 서버에 의해 전송하는 단계;

상기 서버와 클라이언트 장치간 통신 채널의 특성의 변화를 검출하기 위해 사용할 수 있는 피드백 정보를 클라이언트 장치로부터 서버에서 수신하는 단계; 및

상기 통신 채널 특성의 변화를 검출함에 따라, 상기 검출된 통신 채널 특성의 변화에 기초하여 상기 다수의 인코드된 비디오 프레임 또는 그 일부에 대한 압축 비율을 동적으로 조정하는 단계를 포함하는 것을 특징으로 하는 컴퓨터가 비디오 압축을 수행하기 위해 실행하는 방법.

청구항 2

청구항 1에 있어서,

상기 조정된 압축 비율은 제1인코딩 포맷 또는 제2인코딩 포맷의 선택에 기초하는 것을 특징으로 하는 컴퓨터가 비디오 압축을 수행하기 위해 실행하는 방법.

청구항 3

청구항 1에 있어서,

상기 피드백 정보는 상기 클라이언트 장치에 의해 성공적으로 수신된 이전 비디오 프레임을 확인하기 위해 더 사용할 수 있으며, 새로운 비디오 프레임은 성공적으로 수신된 이전 비디오 프레임에 따른 제2인코딩 포맷에 따라 인코드되는 것을 특징으로 하는 컴퓨터가 비디오 압축을 수행하기 위해 실행하는 방법.

청구항 4

청구항 1에 있어서,

피드백 정보는 다수의 인코드된 비디오 프레임 또는 그 일부가 클라이언트 장치에 의해 성공적으로 수신되는 표시를 포함하는 것을 특징으로 하는 컴퓨터가 비디오 압축을 수행하기 위해 실행하는 방법.

청구항 5

청구항 1에 있어서,

피드백 정보는 다수의 인코드된 비디오 프레임 또는 그 일부가 클라이언트 장치에에 의해 성공적으로 수신되지 않은 표시를 포함하는 것을 특징으로 하는 컴퓨터가 비디오 압축을 수행하기 위해 실행하는 방법.

청구항 6

청구항 1에 있어서,

비디오 프레임은 클라이언트 장치에 의해 플레이되는 비디오 게임의 이미지를 포함하는 것을 특징으로 하는 컴퓨터가 비디오 압축을 수행하기 위해 실행하는 방법.

청구항 7

청구항 1에 있어서,

피드백 정보는 다수의 인코드된 비디오 프레임 또는 그 일부가 클라이언트 장치에서 수신되는 표시 또는 각각의

다수의 인코드된 비디오 프레임 또는 그 일부와 관련된 이미지들이 클라이언트 장치의 디스플레이 장치에 디스플레이되는 표시의 적어도 하나를 포함하는 것을 특징으로 하는 컴퓨터가 비디오 압축을 수행하기 위해 실행하는 방법.

청구항 8

프로그램 코드를 저장하기 위한 메모리 및 다수의 동작을 수행하기 위해 프로그램 코드를 처리하기 위한 프로세서를 포함하는 시스템에 있어서, 상기 동작은:

다수의 인코드된 비디오 프레임 또는 그 일부를 생성하기 위해 제1인코딩 포맷에 따라 다수의 비디오 프레임 또는 그 일부를 인코딩하는 동작;

상기 다수의 인코드된 비디오 프레임 또는 그 일부를 클라이언트 장치로 서버에 의해 전송하는 동작;

상기 서버와 클라이언트 장치간 통신 채널의 특성의 변화를 검출하기 위해 사용할 수 있는 피드백 정보를 클라이언트 장치로부터 서버에서 수신하는 동작; 및

상기 통신 채널 특성의 변화를 검출함에 따라, 상기 검출된 통신 채널 특성의 변화에 기초하여 상기 다수의 인코드된 비디오 프레임 또는 그 일부에 대한 압축 비율을 동적으로 조정하는 동작을 포함하는 것을 특징으로 하는 시스템.

청구항 9

청구항 8에 있어서,

상기 조정된 압축 비율은 제1인코딩 포맷 또는 제2인코딩 포맷의 선택에 기초하는 것을 특징으로 하는 시스템.

청구항 10

청구항 8에 있어서,

상기 피드백 정보는 상기 클라이언트 장치에 의해 성공적으로 디코드된 이전 비디오 프레임을 확인하기 위해 더 사용할 수 있으며, 새로운 비디오 프레임은 성공적으로 수신된 이전 비디오 프레임에 따른 제2인코딩 포맷에 따라 인코드되는 것을 특징으로 하는 시스템.

청구항 11

청구항 8에 있어서,

피드백 정보는 다수의 인코드된 비디오 프레임 또는 그 일부가 클라이언트 장치에 의해 성공적으로 수신되는 표시를 포함하는 것을 특징으로 하는 시스템.

청구항 12

청구항 8에 있어서,

피드백 정보는 다수의 인코드된 비디오 프레임 또는 그 일부가 클라이언트 장치에 의해 성공적으로 디코드되지 않은 표시를 포함하는 것을 특징으로 하는 시스템.

청구항 13

청구항 8에 있어서,

비디오 프레임은 클라이언트 장치에 의해 플레이되는 비디오 게임의 이미지를 포함하는 것을 특징으로 하는 시스템.

청구항 14

청구항 8에 있어서,

피드백 정보는 다수의 인코드된 비디오 프레임 또는 그 일부가 클라이언트 장치에서 디코드되는 표시 또는 각각의 다수의 인코드된 비디오 프레임 또는 그 일부와 관련된 이미지들이 클라이언트 장치의 디스플레이 장치에 디

스플레이되는 표시의 적어도 하나를 포함하는 것을 특징으로 하는 시스템.

청구항 15

삭제

청구항 16

삭제

청구항 17

삭제

청구항 18

삭제

청구항 19

삭제

청구항 20

삭제

청구항 21

삭제

청구항 22

청구항 1에 있어서,

통신 채널의 특성 변화의 검출은:

동작하지 않는 통신 채널의 검출; 및

서버에서의 비디오 게임 실행의 일시정지를 포함하며,

상기 비디오 게임 실행은 다수의 비디오 프레임 또는 그 일부를 생성하는 것을 특징으로 하는 컴퓨터가 비디오 압축을 수행하기 위해 실행하는 방법.

청구항 23

청구항 1에 있어서,

서버에서 비디오 게임을 실행하는 단계; 및

제1클라이언트 장치가 이용가능하지 않은 다수의 클라이언트 장치의 적어도 하나를 통지하는 단계를 더 포함하며,

상기 비디오 게임의 실행은 다수의 비디오 프레임 또는 그 일부를 생성하고, 상기 비디오 게임은 멀티플레이어 비디오 게임이고, 클라이언트 장치는 다수의 클라이언트 장치 중 하나이며, 통신 채널은 서버와 다수의 클라이언트 장치의 각각의 하나와의 사이에 확립된 다수의 통신 채널 중 하나이고;

통신 채널의 특성 변화의 검출은 서버와 다수의 클라이언트 장치의 각각의 제1클라이언트 장치간 제1통신 채널의 특성의 검출을 포함하고, 상기 제1통신 채널은 다수의 통신 채널 중 하나인 것을 특징으로 하는 컴퓨터가 비디오 압축을 수행하기 위해 실행하는 방법.

청구항 24

청구항 23에 있어서,

제1통신 채널이 동작하지 않을 때, 다수의 클라이언트 장치에 대한 서버에서 멀티플레이어 비디오 게임의 실행

을 일시정지하는 단계를 더 포함하는 것을 특징으로 하는 컴퓨터가 비디오 압축을 수행하기 위해 실행하는 방법.

청구항 25

청구항 1에 있어서,

다수의 인코드된 비디오 프레임 또는 그 일부가 스트리밍 포맷으로 클라이언트 장치로 전송되는 동안, 통신 채널의 최대 전송률 용량이 상기 다수의 인코드된 비디오 프레임 또는 그 일부의 전송의 검출된 레이턴시의 증가 또는 검출된 패킷 손실의 증가의 적어도 하나에 기초하여 감소하는지를 결정하는 단계; 및

상기 패킷 손실 또는 레이턴시의 적어도 하나가 대응하는 미리 결정된 수용가능한 값에 도달할 때까지 다수의 인코드된 비디오 프레임 또는 그 일부의 전송률을 응답하여 동적으로 감소시키는 단계를 더 포함하는 것을 특징으로 하는 컴퓨터가 비디오 압축을 수행하기 위해 실행하는 방법.

발명의 설명

기 술 분 야

[0001] 본 출원은 "통신 채널을 통해 전송된 소정 타입의 멀티미디어 데이터를 보호하기 위한 방법 및 시스템"이란 명칭으로 2009년 1월 23일 출원된 계류중인 미국출원 12/359,150의 부분 계속출원이면서, CIP 출원의 양수인에 의해서 양수되고 "무선 비디오 게이밍을 위한 방법 및 장치"란 명칭으로 2002년 12월 10일 출원된 10/315,460의 부분 계속출원(continuation-in-part; CIP)인 "피드백을 이용하여 비디오를 압축하기 위한 장치 및 방법"이란 명칭으로 2009년 3월 23일 출원된 미국 가출원 61/210,888에 대한 우선권을 주장한다.

[0002] 본 발명은 일반적으로 오디오 및 비디오 매체의 접근 및 조작을 위한 사용자의 능력을 향상시키는 데이터 처리 시스템 분야에 관하여 개시한다.

배 경 기 술

[0003] 기록된 오디오 및 동영상 매체는 토마스 에디슨의 날 이래로 사회의 한 측면이 되었다. 20세기의 시작에는 기록된 오디오 미디어(실린더(cylinders) 및 레코드) 및 동영상 미디어(니켈로디언(nickelodeons) 및 무비)가 널리 보급되었으나, 양 기술은 아직 초창기에 놓여 있었다. 1920년대 후반에 대중 시장에서 영화는 오디오와 결합하였고, 오디오를 갖는 컬러 동영상이 뒤따랐다. 라디오 방송은 크게 방송 대중 시장 오디오 미디어의 광고 지지 형태로 점진적으로 발전했다. 텔레비전 방송 표준이 1940년대 중반에 확립되었을 때, 텔레비전은 이미 기록되거나 라이브 영화를 홈으로 가져오는 방송 대중 시장 매체의 한 형태로 라디오와 결합하였다.

[0004] 20세기 중반까지, 미국 가정의 대다수는 기록된 오디오 매체를 재생하기 위해 포노그래프(phonograph) 레코드 플레이어, 생방송 오디오를 수신하기 위한 라디오, 및 라이브 방송 오디오/비디오(A/V) 미디어를 재생하기 위한 텔레비전 세트를 갖고 있었다. 매우 자주 이러한 3가지 "미디어 플레이어(media players)"(레코드 플레이어, 라디오 및 텔레비전)은 가정에서 "미디어 센터(media center)"가 되는 공통 스피커를 공유하는 하나의 캐비닛으로 합쳐진다. 비록 미디어 선택은 소비자에게 제한되었을지라도, 미디어 "환경(ecosystem)"은 꽤 안정적이었다. 대부분 소비자는 "미디어 플레이어"를 어떻게 사용하는지 알았고, 그들 능력의 전체 범위를 즐길 수 있었다. 동시에, 미디어 발행인(media publishers)(주로 동영상 및 텔레비전 스튜디오 및 음반 회사)은 그들의 미디어를 중고 미디어의 재판매(resale)와 같은 "두 번째 판매(second sales)" 또는 광범위한 저작권 침해(piracy)를 겪지 않고 극장과 가정에 배급할 수 있었다. 전형적인 발행인은 두 번째 판매로부터는 수익을 얻지 못하고, 그것만으로는 새로운 판매를 위해 중고 미디어의 구매자로부터 얻어지게 될 수익은 줄어든다. 비록 20세기 중반에 중고 레코드가 팔렸을지 모르지만, 그러한 판매는 레코드 발행인에게 큰 영향을 미치지 못하였다. 왜냐하면, 영화 또는 비디오 프로그램 - 어른들에 의해 한번 또는 적은 수로 시청 되는 것-과는 다르게 음악 트랙은 수백 또는 수천 번 듣게 될 것이기 때문이다. 그래서 음악 미디어는 영화/비디오 미디어보다 훨씬 적게 "손상되기 쉬운(perishable)"(예컨대 성인 소비자에게 지속적인 가치가 있다)것이다. 레코드가 한번 구매되면, 만약 소비자가 음악을 좋아한다면, 소비자는 오랫동안 그것을 보유할 수 있을 것이다.

[0005] 20세기 중반부터 현재까지, 미디어 환경은 소비자 및 발행인의 이익 및 손해에 있어서 일련의 급진적인 변화들을 겪어왔다. 오디오 레코드, 특히 높은 품질의 스테레오 사운드를 가진 카세트 테이프의 광범위한 보급에 따르면, 확실히 거기에는 높은 정도의 소비자 편리성이 있었다. 그러나 그것은 또한 소비자 미디어로 광범위한 실시

(저작권 침해)의 시작임을 보여준다. 확실히, 많은 소비자는 편리함을 위해 순수하게 그들 자신의 레코드를 테이프 녹음한 카세트 테이프를 사용했으나, 점점 더 소비자(예컨대, 서로 레코드 수집품에 쉽게 접근하는 기숙사의 학생들)는 저작권 침해된 카피들을 만들게 될 것이다. 또한, 소비자는 발행인으로부터 레코드 또는 테이프를 사는 것보다 라디오를 통해 재생된 음악을 테이프 녹음할 것이다.

[0006] 소비자 VCR의 도래는 보다 많은 소비자 편의를 가져왔고, 현재 VCR이 늦은 시간에 시청될 수 있는 TV 쇼를 기록하기 위해 설치된 이래로, 그것은 또한 비디오 대여사업의 창조를 가져왔으며, TV 프로그래밍뿐만 아니라 영화는 "주문(On demand)"에 기초하여 접근할 수 있게 되었다. 1980년대 중반 이후에 홈 미디어 장치의 대중 시장의 빠른 발전은 전례가 없이 소비자를 위한 편의와 선택의 수준을 이끌어왔고, 미디어 출판 시장의 빠른 확장을 이끌어왔다.

[0007] 오늘날, 소비자는 대부분 미디어의 특정 형태 또는 특정 발행인에 얽매여서, 미디어 장치의 과잉뿐만 아니라 미디어 선택의 과잉과 직면하고 있다. 미디어의 열렬한 소비자는 집의 다양한 방들에서 텔레비전 및 컴퓨터와 연결된 다수의 장치를 소유하여, 한 그룹의 무선 제어들뿐만 아니라 개인 컴퓨터(PCs) 및/또는 하나 이상의 텔레비전과 케이블의 "혼란 상태(rat's nest)"를 초래한다. (본 출원의 내용에서, "개인용 컴퓨터(Personal Computer)" 또는 "PC"는 홈이나 오피스에 적합한 어떠한 종류의 컴퓨터를 칭하고, 이는 데스크톱, Macintosh® 또는 그 밖의 윈도우 기반이 아닌 컴퓨터, 윈도우 기반의 장치, 유닉스 계열, 랩탑(laptop) 등을 포함한다.) 이러한 장치들은 비디오 게임 콘솔(Video game console), VCR, DVD 플레이어, 오디오 서라운드-사운드 프로세서/앰프(Audio sorround-sound processor/amplifier), 위성 셋톱 박스(satellite set-top box), 케이블 TV 셋톱 박스(cable TV set-top box) 등이다. 그리고 열광적인 소비자를 위해, 양립가능한 이슈들 때문에 다수의 비슷한 기능을 하는 장치가 있다. 예컨대, 소비자는 HD-DVD 및 Blu-ray DVD 플레이어, 또는 Microsoft XBOX®, Sony Playstation® 비디오 게임 시스템을 소유할 것이다. 더욱이, 어떤 게임간에 게임 콘솔 버전의 양립불가능성 때문에, 소비자는 XBox 및 XBox 360®과 같은 후 버전을 둘 다 소유할 것이다. 종종, 소비자들은 사용에 대한 어느 비디오 입력 및 어느 원격 조작에 관하여 어리둥절해진다. 디스크가 올바른 플레이어(예컨대, DVD, HD-DVD, Blu-ray, Xbox 또는 Playstation)에 위치하고, 그 비디오 및 오디오 입력이 해당 장치를 위해 선택되며, 올바른 리모트 컨트롤이 발견된 후에도, 소비자는 여전히 기술적 도전에 직면하게 된다. 예컨대, 와이드 스크린 DVD의 경우에, 사용자는 먼저 결정하고, TV 또는 모니터 스크린에 올바른 애스펙트비(aspect ratio)(예컨대. 4:3, Full, Zoom, Wide Zoom, Cinema Wide 등)로 설정하는 것이 필요로 된다. 유사하게, 사용자는 먼저 결정하고, 이어 올바른 오디오 서라운드 사운드 시스템 포맷(예컨대, AC-3, Dolby Digital, DTS 등)으로 설정하는 것이 필요로 된다. 매번, 소비자는 그들의 텔레비전 또는 오디오 시스템의 충분한 성능으로 미디오 컨텐츠를 즐기지 못한다는 것을 모른다(예컨대, 잘못된 애스팩트비로 일그러진 영화를 시청하는 것, 또는 서라운드 사운드(surround sound)보다 스테레오(stereo)에서 오디오를 청취하는 것).

[0008] 점차로, 인터넷 기반의 미디어 장치들이 장치의 스택(stack)에 부가된다. Sonos® 디지털 음악 시스템(Digital Music System)과 같은 오디오 장치들은 인터넷으로부터 직접 오디오를 스트림(stream)한다. 마찬가지로, Slingbox™ 엔터테인먼트 플레이어와 같은 장치들은 홈네트워크를 통해서 비디오를 스트림하고, 레코드하거나 인터넷을 통해서 PC와 떨어져서 시청할 수 있다. 그리고 인터넷 프로토콜 텔레비젼(IPTV) 서비스는 종종 디지털 가입자 회선(Digital subscriber line ; DSL) 또는 그 밖의 홈 인터넷 연결을 통해서 케이블 TV와 같은 서비스를 제공한다. 또한 Moxi® 미디어 센터 및 윈도우XP 미디어 센터 에디션이 구동되는 PC들과 같은 단일 장치로 다수의 미디어 기능들을 통합하려는 최근의 노력이 있다. 이들 장치의 각각이 그를 수행하는 기능을 위한 편리성의 요소를 제공하는 반면에, 각각은 대부분의 미디어에 대한 유비쿼터스(ubiquitous) 및 단순한 접근이 부족하다. 더욱이, 잦은 비싼 처리 및/또는 로컬 기억장치에 대한 필요성 때문에 그러한 장치는 생산을 위해 수백 달러의 비용이 소비된다. 추가적으로 이러한 현대 소비자 전자 장치(consumer electronic devices)는 전형적으로 동작 중이 아님에도 많은 전력을 소모하고, 이는 에너지 자원의 낭비 및 시간 경과에 따라 고가로 됨을 의미한다. 예컨대, 어떤 장치는 소비자가 그것을 끄거나 다른 비디오 입력으로 전환하는 것을 무시하면, 작동을 계속하게 될 것이다. 그리고, 어떠한 장치도 완전한 솔루션이 아니기 때문에, 가정 내에서 다른 장치의 스택(stack)과 통합되어야 하고, 이는 여전히 리모트 컨트롤의 바다 및 혼란스런 전선으로 사용자를 떠나게 한다.

[0009] 더욱이, 많은 새로운 인터넷 기반의 장치가 적절히 동작할 때, 그들은 전형적으로 그 밖의 형태로 이용할 수 있는 것보다 포괄적인 형태(generic form)로 미디어를 제안한다. 예컨대, 인터넷을 통해 비디오를 스트림하는 장치는 단지 비디오, 게임, 또는 감독의 해설의 "구성(making of)" 과 같은, 종종 DVD에 동반되는 대화형 "추가

(extras)"가 아닌 비디오 재료(video material)를 스트림한다. 이는 종종 대화형 재료가 국부적으로 대화(interactivity)를 다루는 특정 장치를 위해 의도된 특정 형식으로 제조된다는 사실에 기인한다. 예컨대, DVD, HD-DVD, 및 Blu-ray 디스크 각각은 그들 자신의 특정한 인터랙티브 형식을 가지고 있다. 모든 인기있는 형식을 지원하기 위해서 개발되었던 어떤 홈 미디어 장치 또는 로컬 컴퓨터는 소비자가 작동시키는데 복잡하고 상당히 고가로 되는 정교화(sophistication) 및 유연성(flexibility)의 수준을 요구한다.

[0010] 그 문제에 추가하여, 만약 앞으로 새로운 포맷이 소개되면, 로컬 장치는 새로운 포맷을 지원하기 위한 하드웨어 성능을 갖지 못할 것이고, 이는 소비자가 업그레이드된 로컬 미디어 장치를 구입해야함을 의미한다. 예컨대, 만약 고해상도(higher-resolution) 비디오 또는 입체음향(stereoscopic) 비디오(예컨대, 각각의 눈을 위한 하나의 비디오 스트림)는 최근에 소개되었고, 로컬 장치가 비디오를 디코드하기 위해 컴퓨터적인 성능을 갖추지 않거나 새로운 포맷(예컨대, 가령 입체음향이 각각의 눈으로 60fps 전달되는 셔터된 안경(shuttered glasses)과 동기화된 120fps 비디오를 통해 달성된다면, 이러한 옵션은 업그레이드된 하드웨어가 부재할 경우 이용될 수 없을 것이다)으로 비디오를 출력하기 위한 하드웨어를 갖지 않을 것이다.

[0011] 미디어 장치의 진부화 및 복잡성의 이슈는 그것이 정교한 인터랙티브 미디어가 될 때, 특히 비디오 게임에서 심각한 문제이다.

[0012] 현재 비디오 게임 어플리케이션은 크게 4대 메이저 비-휴대용(non-portable) 하드웨어 플랫폼 : Sony PlayStation® 1,2 및 3(PS1, PS2, 및 PS3); Microsoft Xbox®; 및 Nintendo Gamecube® 및 Wii™; 및 PC 기반 게임이다. 이러한 플랫폼(platforms) 각각은 서로 달라서 한 플랫폼에서 동작하는 기록된 게임은 보통 다른 플랫폼에서 동작하지 않는다. 또한, 장치의 한 세대에서 다음 세대로 양립가능성(compatibility)의 문제가 있다. 소프트웨어 게임 개발자의 대다수가 특정 플랫폼에 독립적으로 디자인된 소프트웨어를 창조함에도 불구하고, 특정 플랫폼에서 특정 게임을 구동하기 위해 소프트웨어의 독점 레이어(proprietary layer)(흔히 "게임 개발자 엔진(game development engine)"으로 불림)은 특정 플랫폼에서 사용을 위해 게임을 채택하는 것이 필요해진다. 각 플랫폼은 "콘솔"(예컨대, TV 또는 모니터/스피커에 부착된 독립형(standalone) 박스)로서 소비자에게 팔리거나 그것은 PC 자체이다. 전형적으로, 비디오 게임은 Blu-ray, DVD, DVD-ROM 또는 CD-ROM과 같은 광학적 미디어로 팔리고, 그것은 정교한 실시간 소프트웨어 어플리케이션으로서 구체화된 비디오 게임을 포함한다.

[0013] 비디오 게임 소프트웨어와 양립가능한 플랫폼을 달성하기 위한 특정 요구는 실시간 특성 및 진보된 비디오 게임의 높은 컴퓨터적인 요구(computational requirement)에 기인하여 부득이해질 것이다. 예컨대, 한 PC에서 더 빠른 처리 유닛과 코어를 가진 다른 것으로 생산성 어플리케이션(예컨대, Misrosoft Word)의 일반적 양립가능성이 있는 것처럼 비디오 게임의 한 세대에서 다음 세대(예컨대, Xbox에서 Xbox 360으로 또는 Playstation 2("PS2")에서 Playstation 3("PS3"))로 충분히 양립가능한 게임을 기대할 수 있을 것이다. 그러나, 이는 비디오 게임의 경우에는 그렇지 않다. 왜냐하면 전형적으로 비디오 게임 제조자들은 비디오 게임 세대가 출시된 때, 주어진 가격 포인트(point)에서 가능한 최상의 성능을 찾고, 시스템의 획기적인 구조적인 변화가 빈번하게 생겨서 이전 세대 시스템을 위해 기록된 많은 게임은 최신 세대 시스템에서 동작하지 않게 된다. 예컨대, XBox는 x86- 계열 프로세서에 기초하고, 반면, XBox 360은 PowerPC 계열에 기초한다.

[0014] 이전 아키텍쳐를 에뮬레이트(emulate)하기 위한 기술이 이용될 수 있지만, 주어진 비디오 게임은 실시간 어플리케이션으로, 에뮬레이션에서 정확하게 동일한 동작을 달성하도록 실행될 수는 없다. 이는 소비자, 비디오 게임 콘솔 제조자 및 비디오 게임 소프트웨어 발행인에게 손실이 된다. 소비자를 위해서는, 모든 게임을 플레이할 수 있도록 TV와 연결된 비디오 게임 콘솔의 구세대 및 신세대 모두를 유지하는 것이 필요하다는 것을 의미한다. 그것은 콘솔 제조자에게는 새로운 콘솔의 더 늦은 채택과 에뮬레이션과 관련된 비용을 의미한다. 그리고 발행인에게는 새로운 게임의 다양한 버전들이 모든 잠재적인 소비자에게 접근하기 위해서 출시되어야만 하는 것--비디오 게임의 각 브랜드(예, XBox, Playstation)를 위한 버전뿐만 아니라 기존 브랜드(예, PS2 및 PS3)의 각 버전을 위한 버전을 출시하는 것을 의미한다. 예컨대, Electronic Art의 "Madden NFL 08"는 XBox, XBox360, PS2, PS3, Gamecube, Wii, 및 PC, 그 밖의 플랫폼을 위해 개발되었다.

[0015] 셀룰러("셀(cell)") 폰 및 휴대용 미디어 플레이어와 같은 휴대용 장치는 또한 게임 개발자에게 도전을 하고 있다. 점차적으로 그러한 장치는 무선 데이터 네트워크와 연결되고 비디오 게임을 다운로드 받을 수 있다. 그러나, 시장에는 다른 디스플레이 해상도 및 컴퓨팅 성능을 갖춘 넓은 다양한 셀 폰 및 미디어 장치가 존재한다. 또한, 전형적으로 그러한 장치가 전력 소비, 가격 및 무게의 제약이 있으므로, 그것들은 전형적으로 캘리포니아(CA) 산타클라라의 NVIDIA에 의해 만들어진 장치인 진보적인 그래픽 처리 유닛(Graphics Processing Unit ; "GPU")과 같은 그래픽 가속 하드웨어가 부족하다. 결과적으로, 게임 소프트웨어 개발자들은 전형적으로 기존

게임 타이틀을 동시에 많은 다른 형태의 휴대용 장치를 위해 개발한다. 한 사용자가 기존 게임 타이틀이 그의 셀 폰 또는 휴대용 미디어 플레이어에서 이용될 수 없다는 것을 발견하게 될 것이다.

[0016] 홈 게임 콘솔의 경우에, 하드웨어 플랫폼 제조업체는 전형적으로 소프트웨어 게임 개발자에게 그들의 플랫폼에 게임을 출시할 수 있도록 로얄티를 청구한다. 셀 폰 무선 캐리어(cell phone wireless carriers)는 전형적으로 셀 폰으로 게임을 다운로드하기 위해 게임 발행인에게 로얄티를 청구한다. PC 게임의 경우에, 공개 게임(publish game)에 대해 지급하는 로얄티는 없지만, 게임 개발자들은 전형적으로 넓은 범위의 PC 구성을 지원하기 위해 부담하는 높은 고객 서비스와 발생 가능한 설치 문제로 인한 높은 비용의 문제에 직면한다. 또한, PC는 전형적으로 기술적인 지식을 가진 사용자에 의해서 쉽게 다시 프로그램되기 때문에 게임 소프트웨어의 해적 행위에 대한 장벽이 낮고, 게임은 보다 쉽게 약탈되고 배포된다(예컨대, 인터넷을 통해서). 따라서, 소프트웨어 게임 개발자에게는, 게임 콘솔, 셀 폰 및 PC에 출시하는 것은 불이익 및 비용이 존재한다.

[0017] 콘솔 및 PC 소프트웨어 게임 출판인에게, 비용은 거기서 끝이 아니다. 소매 채널을 통해서 게임을 보급하기 위해서, 소매상이 이익을 낼 수 있도록 출판인은 판매가격(selling price) 아래의 도매가격(wholesale price)을 청구한다. 출판인들은 또한 전형적으로 게임을 담고 있는 물리적 미디어를 생산하고 보급하는 비용을 지불해야 한다. 출판인은 또한 만약 게임이 안팔리거나, 게임의 가격이 하락되거나, 만약 소매자가 도매가격의 일부 또는 전체를 환불하고 그리고/또는 구매자로부터 게임을 교환받는 것과 같은 가능한 우연성을 감당하기 위해 소매자에 의해 "가격 보호 수수료(price protection fee)"를 부과당하게 된다. 추가로, 소매자는 또한 전형적으로 광고 전단지에서 게임 판매를 돕기 위해 출판인에게 수수료를 청구한다. 더욱이, 소매자들은 점차적으로 플레이를 마친 사용자로부터 게임을 다시 구매하고, 그 후에 그것들을 중고 게임으로 판매하며, 전형적으로 게임 출판인과 중고 게임 수익 중 아무것도 분배하지 않는다. 게임 출판인에게 추가되는 비용부담은 게임이 자주 불법복제되고 인터넷을 통해서 배포되어 사용자들이 다운로드하여 무료로 복사한다는 것이다.

[0018] 인터넷 광대역 속도는 증가되어 왔고 광대역 연결도(connectivity)는 미국 및 전세계에서 보다 광범위하게 되었으며, 특히 홈 그리고 인터넷 "카페"에 인터넷이 연결된 PC가 대여되고, 게임은 점차적으로 PC 또는 콘솔을 위해 다운로드를 통해서 배포된다. 또한, 광대역 접속은 멀티플레이어 및 대량 멀티유저 온라인 게임(양쪽 모두 본 개시에 있어서, "MMOG"로 칭함)을 플레이 하기 위해 점차적으로 사용된다. 이러한 변화는 소매 유통과 관련된 비용 및 이슈를 완화시킨다. 온라인 게임을 다운로드하는 것은 유통 비용이 전형적으로 적고 팔리지 않은 미디어로부터 비용이 없거나 거의 없다는 점에서 어떤 불이익을 게임 출판인에게 제기한다. 그러나 다운로드된 게임은 여전히 저작권 침해되기 쉽고, 그들의 크기(종종 많은 기가바이트 크기)때문에 그들은 다운로드 하기에 매우 많은 시간이 걸릴 수 있다. 덧붙여, 멀티플 게임은 휴대용 컴퓨터와 함께 판매되거나 비디오 게임 콘솔과 함께 판매되는 것처럼 작은 디스크 드라이브를 가득 채울 수 있다. 그러나, 확장 게임 또는 MMOG는 플레이하기 위해 게임과 온라인 연결이 필요하고, 저작권 침해 문제는 사용자가 통상적으로 합법적인 사용자 계정(valid user account)을 가질 것이 요구됨으로써 완화될 수 있다. 스피커로부터 오디오를 기록하는 마이크로폰(microphone) 또는 디스플레이 화면의 비디오를 촬영하는 카메라에 의해서 복제될 수 있는 선형 미디어(linear media)(예컨대, 비디오 및 음악)와는 달리 각 비디오 게임 경험은 독특하고, 단순히 비디오/오디오 기록을 사용하여 복제될 수 없다. 따라서, 저작권법이 있는 지역에서조차 강력하게 제재되지 못하고, 불법 복제가 만연하게 된다. MMOG는 불법복제로부터 보호될 수 있고 따라서 사업은 지원을 받을 수 있다. 예컨대, Vivendi SA의 "월드 오브 워크래프트" MMOG는 전세계적으로 불법복제를 겪지 않고 성공적으로 배치되었다. 그리고 Linden Lab의 "세컨드 라이프" MMOG와 같은 많은 온라인 또는 MMOG 게임은 자본이 구매, 판매 및 온라인 툴을 사용해서 창조될 수 있는 게임으로 구축된 경제 모델을 통해서 게임의 오퍼레이터(operator)를 위해 수익을 창출한다. 따라서, 전형적인 게임 소프트웨어 구입 또는 구독에 추가된 메카니즘은 온라인 게임의 사용을 위해 지불될 수 있다.

[0019] 반면 불법복제는 자주 온라인 또는 MMOG의 특성에 기인하여 완화될 수 있고, 온라인 게임 오퍼레이터는 여전히 잔존하는 도전에 직면하게 된다. 많은 게임은 적당히 작업하기 위한 온라인 또는 MMOG를 위해 실질적으로 로컬(예컨대, 가정) 프로세싱 리소스를 요구한다. 만약 사용자가 낮은 성능의 로컬 컴퓨터(예컨대, 낮은 성능의 랩탑과 같은 GPU가 없는)를 갖고 있다면, 게임을 플레이할 수 없다. 더욱이, 게임 콘솔의 년차(age)에 따라, 그들이 최신 기술에 훨씬 뒤떨어져 있고 더 이상 진보된 게임을 다룰 수 없을 것이다. 사용자의 로컬 PC가 게임의 컴퓨터적인 요구사항을 다룰 수 있을지라도, 거기에는 종종 설치의 복잡성이 있다. 드라이버의 비양립성이 있을 수 있다(예컨대, 만약 새로운 게임이 다운로드되면, 구 버전의 그래픽 드라이버에 의존하는 먼저 설치된 게임은 동작하지 않게 되는 새로운 버전의 그래픽 드라이버를 인스톨할 것이다). 콘솔은 많은 게임이 다운로드됨으로써 로컬 디스크 공간이 바닥나게 될 것이다. 복잡한 게임은 버그가 발견되어 고쳐짐에 따라, 또는 만약 변경이 게임에 만들어지면(예컨대, 만약 게임 개발자가 게임의 수준이 플레이하기에 너무 어렵거나 너무 쉽다는 것을 발

견하면) 게임 개발자로부터 시간 경과에 따라 전형적으로 다운로드된 패치를 수신한다. 이러한 패치들은 새로운 다운로드를 요구한다. 그러나 때때로 모든 사용자가 모든 패치의 다운로드를 완료하지는 않는다. 다른 한편, 다운로드된 패치는 그 밖의 호환 가능성, 디스크 공간의 소비와 같은 이슈를 소개한다.

[0020] 또한, 게임을 플레이하는 동안, 많은 데이터 다운로드가 로컬 PC 또는 콘솔에 대해 그래픽 또는 동작 정보를 제공하기 위해 요구된다. 예컨대, 만약 사용자가 MMOG 룸에 들어가서 사용자의 로컬 머신에 이용될 수 없는 그래픽 데이터와 행동으로 구성되는 장면 또는 캐릭터를 직면하면, 장소 또는 캐릭터의 데이터는 다운로드되어야 한다. 이는 만약 인터넷 연결이 충분히 빠르지 않다면, 게임 플레이 동안에 실질적인 지연을 초래한다. 그리고, 만약 직면하는 장소 및 캐릭터가 저장 공간 또는 로컬 PC 혹은 콘솔의 컴퓨터적인 성능을 넘어서는 요구를 한다면, 이는 게임에서 사용자가 진행을 할 수 없거나, 저품질의 그래픽으로 계속해야되는 상황을 초래한다. 따라서, 온라인 또는 MMOG 게임은 자주 그들의 저장소 및/또는 컴퓨터적인 복잡한 요구를 제한한다. 더욱이, 게임 도중에 종종 많은 양의 데이터 전송을 제한한다. 온라인 또는 MMOG 게임은 또한 게임을 플레이할 수 있는 사용자의 시장을 좁힐 수도 있다.

[0021] 뿐만 아니라, 기술적-지식을 가진 사용자는 점차적으로 게임의 로컬 카피를 역-엔지니어링(reverse-engineering)할 것이고 게임을 변형하게 되어서 그들을 속일 수 있다. 속임수(cheats)는 인간이 할 수 있는 것보다 더 빠르게 버튼을 누르는 것을 만드는 것만큼 쉬울 것이다(예컨대, 매우 반복적으로 총을 쏘는 것). 게임 내 자산 거래(in-game asset transaction)를 지원하는 게임에 있어서, 속임수는 실제적 경제 가치의 자산을 포함하는 사기 거래를 초래하는 정교한 수준에 도달할 수 있다. 온라인 또는 MMOG 경제 모델은 그러한 자본 거래에 기초할 때, 이는 게임 오퍼레이터에게 실질적인 해로운 결과를 초래하게 할 수 있다.

[0022] 새로운 게임의 개발 비용은 PC 및 콘솔이 점차적으로 세련된 게임(예컨대, 실시간 레이-트레이싱(ray-tracing)과 같은 더욱 사실적인 그래픽, 그리고 실시간 물리적 시뮬레이션과 같은 보다 사실적인 동작)을 생산하도록 함으로써 증가된다. 비디오 게임 사업의 초창기에, 비디오 게임 개발은 어플리케이션 소프트웨어 개발에 대한 프로세스와 매우 유사하였는 바, 즉 대부분의 개발 비용은 광범위한 특별한 효과를 갖는 움직임 화상을 위해 개발되어지는 것과 같은, 그래픽, 오디오, 및 동작적인 요소 또는 "애셋(assets)"과 반대로, 소프트웨어의 개발에 들어갔다. 오늘날, 많은 정교한 비디오 게임 개발은 소프트웨어 개발 보다 풍부한 특수 효과 움직임 화상 개발을 더욱 닮기 위해 노력한다. 예컨대, 많은 비디오 게임은 3-D 월드의 시뮬레이션을 제공하고, 점차적으로 포토리얼리즘(photorealistic)(예컨대, 포토그래픽으로 찍은 생동감 있는 이미지처럼 실사적인 컴퓨터 그래픽) 캐릭터, 소품, 및 환경을 창출한다. 포토리얼리즘 게임 개발의 가장 도전적인 측면 중 하나는 생생하게 동작하는 인간 얼굴과 구별할 수 없는 컴퓨터에 의해 창조된 인간 얼굴을 창조하는 것이다. 캘리포니아 샌프란시스코의 Mova에 의해서 개발된 Contour™ Reality Capture와 같은 얼굴 캡쳐 기술은, 동작중인 동작자의 얼굴을 높은 해상도의 정확한 기하학적 구조로 추적하고 캡쳐한다. 이는 캡쳐된 살아있는 얼굴 움직임과 시각적으로 구별할 수 없도록 PC 또는 게임 콘솔에 랜더링된 3D 얼굴을 허용하게 된다. 정확하게 인간의 얼굴을 "포토리얼(photoreal)"로 캡쳐링(capturing) 및 랜더링(rendering)하는 것은 여러 측면에서 유용하다. 먼저, 잘 알아볼 수 있는 유명인사 또는 운동선수는 자주 비디오 게임에서 이용되고(종종 높은 비용으로 고용됨), 결함이 사용자에게 보일 수 있어 집중이 안되거나 즐겁지 않은 관찰 체험을 만들게 된다. 종종, 높은 정도의 상세함은 잠재적으로 페이스 무비(face movie)로서 프레임마다 변하는 폴리건(polygon) 및/또는 텍스쳐(textures)를 갖는 다수의 폴리건 및 높은 해상도 텍스쳐의 랜더링을 요구하는, 높은 정도의 포토리얼리즘을 달성하기 위해 요구된다.

[0023] 상세한 텍스쳐를 가진 높은 폴리건-카운트 장면이 빠르게 변화할 때, 게임을 지원하는 PC 또는 게임 콘솔은 게임 세그먼트(segment)에서 만들어지는 애니메이션 프레임이 요구하는 수를 위해 충분한 폴리건 및 텍스쳐 데이터를 저장하기 위해서 충분한 RAM을 가지고 있지 않게 될 것이다. 더 나아가, 전형적으로 PC 또는 게임 콘솔에 이용되는 단일 광학 드라이버 또는 단일 디스크 드라이버는 보통 RAM 보다 속도가 느리고, 전형적으로 GPU가 다각형 및 텍스쳐를 랜더링하는데 허용될 수 있는 최대 전송률(즉, 데이터 비율(data rate))을 유지하지 못할 수 있다. 현재 게임은 전형적으로 대부분의 폴리건 및 텍스쳐를 RAM으로 로드하고, 이는 주어진 장면이 대체로 RAM의 성능에 의해서 복잡성 및 지속성에 제한된다는 것을 의미한다. 얼굴 애니메이션의 경우에, 예컨대 이는 PC 또는 게임 콘솔은 게임을 멈추기 전에 실사 사진 같지 않은 낮은 해상도 얼굴, 또는 오직 제한된 수의 프레임으로 애니메이팅될 수 있는 실사 같은 얼굴 어느 하나로 제한되고, 폴리건 및 텍스쳐를 보다 많은 프레임을 위해 로드한다.

[0024] PC 또는 콘솔은 "로딩..."과 비슷한 메세지를 디스플레이하는 것처럼 화면을 가로질러 천천히 움직이는 진행 바(progress bar)를 지켜보는 것은 복잡한 비디오 게임의 오늘날의 사용자에 의해서 내재하는 결점으로 받아들여

진다. 디스크(여기서 "디스크(disk)"는, 다른 것으로 한정하지 않는 한, 반도체"플래시(flash)" 메모리와 같은 비디스크 미디어뿐만 아니라 비휘발성 광학 또는 자기 미디어로 언급된다.)에서 다음 장면이 로드되는 동안 지연은 몇 초 또는 몇 분이 걸릴 수 있다. 이는 시간의 낭비이고 게임 플레이어게 꽤 당혹스럽게 할 수 있다. 앞서 논의한 것처럼, 지연의 많은 부분 또는 전부는 폴리건, 텍스쳐 또는 그 밖의 디스크로부터 데이터를 위한 로드 시간에 기인할 것이지만, 또한 프로세서 및/또는 PC 또는 콘솔에서 GPU가 장면을 위한 데이터를 준비하는 동안에 소비되는 로드 시간의 일부일 것이다. 예컨대, 축구 비디오 게임은 플레이어가 많은 플레이어, 팀, 운동장 및 날씨 상태 등을 선택하도록 허용한다. 따라서 특정 조합(combination)이 선택된 것에 따라, 다른 폴리건, 텍스쳐 및 그 밖의 데이타(집합적으로"객체(object)")가 장면을 위해서 요구된다(예컨대, 다른 팀은 그들의 유니폼에 다른 색 및 패턴을 가진다). 게임을 저장하기 위해 사용되는 디스크에 객체를 저장하고 미리 모든 객체 또는 많은 객체를 미리 연산하고 많은 또는 모든 다양한 치환(permutations)을 열거하는 것이 가능할 것이다. 그러나 만약 치환이 너무 크다면, 모든 객체를 위해 요구되는 많은 스토리지는 디스크에 고정하기 위해 너무 클 것이다(또는 다운로드 하기에 너무 실제적이지 않다). 따라서, 현존하는 PC 또는 콘솔 시스템은 전형적으로 복잡성 및 주어진 장면의 기간 모두에 강요되어 복잡한 장면을 위해 긴 로드 시간을 겪게 된다.

[0025] 종래 기술 비디오 게임 시스템 및 어플리케이션 소프트웨어 시스템의 다른 현저한 제한은 처리를 위해 PC 또는 게임 콘솔으로 로드되는 것이 필요한 예컨대 폴리건 및 텍스쳐와 같은 3D 객체의, 큰 데이터베이스를 점차적으로 사용하게 된다는 것이다. 상술한 것처럼, 그러한 데이터베이스는 디스크에 국지적으로 저장된 때 로드를 위해 긴 시간이 걸렸다. 그러나, 로드 시간은 만약 데이터베이스가 원격 위치에 저장되고 인터넷을 통해서 접근될 수 있다면 보다 훨씬 심각한다. 그러한 상황에서 많은 데이터베이스를 다운로드 하기 위해서는 몇 분, 몇 시간 또는 몇 일이 걸릴 수도 있다. 보다 더, 그러한 데이터베이스는 종종 매우 비싸게(예컨대, 게임, 영화, 또는 역사적 다큐멘터리에서 사용되는 상세한 톨-마스트된(tall-masted) 범선의 3D 모델) 만들어지고, 로컬 최종-사용자에게 판매를 위해 의도된다. 그러나 데이터베이스가 로컬 사용자에게 다운로드되어지면서 불법복제되는 위험이 있다. 많은 경우에, 사용자는 사용자의 요구(예컨대, 만약 게임 캐릭터를 위한 3D 의상은 사용자가 특정 움직임을 수행할 때, 만족스러운 외형 또는 외관을 갖는다)에 어울린다면 그것을 보기 위한 평가의 이유를 위해 단순히 데이터베이스를 다운로드하기를 원한다. 긴 로드 시간은 구입을 결정하기 전에 3D 데이터베이스를 사용자가 평가하는 것을 제지할 수 있다.

[0026] 유사한 이슈가, 특히 게임처럼 사용자가 점차적으로 맞춤형 캐릭터를 이용하는 것을 허용하는 게임과 같은 MMOG에서 야기된다. 캐릭터를 디스플레이하기 위한 PC 또는 게임 콘솔에 대해, 캐릭터를 위한 행동뿐만 아니라(예컨대, 만약 캐릭터가 쉴드를 갖는다면, 그 쉴드(shield)가 스피어(spear)를 피하기에 충분히 강한지 아닌지) 3D 지오메트리 데이터베이스(폴리곤, 텍스처 등)에 접근하는 것이 필요하다. 전형적으로 먼저 사용자에 의해서 MMOG가 플레이된 때, 캐릭터를 위한 많은 데이터베이스가 이미 게임의 초기 카피로 이용될 수 있고, 이는 게임의 광학적 디스크 또는 디스크로 다운로드되어 국지적으로 이용된다. 그러나, 게임이 진행됨에 따라, 만약 사용자가 캐릭터 또는 객체의 데이터베이스와 직면하면, 그것의 데이터베이스는 캐릭터 또는 객체가 디스플레이되기 전에, 국지적으로 이용할 수 없고(예컨대, 만약 다른 사용자가 맞춤형 캐릭터를 창조했다면), 그것의 데이터베이스는 다운로드되어야 한다. 이는 실질적으로 게임의 지연을 초래한다.

[0027] 주어진 비디오 게임의 세련과 복잡함, 종래 기술 비디오 게임 콘솔을 가진 비디오 게임 개발자 및 출판인에 대한 다른 도전은, 비디오 게임을 개발하는데 수천만달러의 비용으로 종종 2~3년간 걸릴 것이다. 새로운 비디오 게임 콘솔 플랫폼이 대략 매 5년의 비율로 소개된다고 하면, 게임 개발자는 새 플랫폼이 출시될 때, 동시에 이용가능한 비디오 게임을 가지도록 새로운 게임 콘솔의 개시에 앞선 몇 년에 미리 이러한 게임 개발 업무를 시작하는 것이 필요하다. 경쟁 업체들의 몇몇 콘솔은 때때로 동시에 출시되거나(예컨대, 1년 또는 2년 안에), 보여지는 나머지는 콘솔의 인기도이고, 이는 콘솔이 가장 큰 비디오 게임 소프트웨어 판매를 만들것이다. 예컨대, 최근 콘솔 주기(cycle에서, 마이크로 소프트 XBox 360, Sony Playstation 3, 및 Nintendo Wii는 동일한 일반 기간에 소개되었다. 그러나 소개되기 전 몇년에 게임 개발자들은 콘솔 플랫폼이 다른 것보다 더 성공적일 것인지에 관하여 "그들의 추측에 놓여(place their bets)"져야 하고, 그들의 개발 자원을 일치하도록 헌신한다. 동영상 제작 회사는 또한 영화의 출시에 앞서서 상당히 영화가 성공할 것인지 평가한 것에 기초하여 그들의 제한된 생산 자원을 배분해야 한다. 비디오 게임을 위해 요구되는 투자의 증가하는 수준이 주어지면, 게임 제작은 동영상 제작처럼 점차로 증가될 것이고, 게임 제작 회사는 특정 비디오 게임의 미래 성공에 대한 그들의 평가에 기초하여 통상적으로 그들의 생산 리소스를 충당한다. 그러나, 동영상 제작 회사와 달리, 이러한 추측은 단지 생산의 성공 자체에 기초하지 않는다; 꽤, 그 게임이 동작되도록 의도된 게임 콘솔의 성공에 기초하여 예측된다. 동시에 멀티 콘솔에서 게임을 출시하는 것은 위험을 완화시키지만, 이러한 추가되는 노력은 비용을

증가시키고 종종 게임의 실질적인 출시를 지연한다.

[0028] PC에서 어플리케이션 소프트웨어 및 사용자 환경은 보다 컴퓨터적으로 집중적, 동적 및 인터랙티브이고, 사용자에게 보다 시각적으로 표현하도록 하는 것뿐만 아니라, 보다 유용하고 이용하기 쉽도록 만드는 것이다. 예컨대, 새로운 Window Vista™ 운영 시스템 및 Macintosh® 운영 시스템의 연속하는 버전은 시각적 애니메이션 효과를 포함한다. Autodesk, Inc의 Maya™와 같은 진보된 그래픽 도구는 매우 최신 기술의 CPU 및 GPU에 제한을 가하는 정교한 3D 랜더링 및 애니메이션 성능을 제공한다. 그러나 이러한 새로운 도구의 컴퓨터적인 요구는 그러한 제품의 사용자 및 소프트웨어 개발자에게 많은 실질적인 이슈를 만든다.

[0029] 운영 시스템(OS)의 시각적(visual) 디스플레이는--더 이상 팔리지 않지만, 여전히 새로운 OS로 업그레이드 가능한 이전 세대의 컴퓨터를 포함하는 넓은 범위의 등급의 컴퓨터에서 동작해야만 하기 때문에, OS 그래픽 요구사항이 OS가 대상이 되는 컴퓨터의 최소한의 공통된 요구사항(requirement)에 의해서 큰 정도로 제한되고, 이는 전형적으로 GPU가 없는 컴퓨터를 포함한다. 이는 OS의 그래픽 성능을 심각하게 제한한다. 더욱이, 휴대용 컴퓨터(예, 랩탑) 배터리 파워(battery-powered)는 CPU 또는 GPU에서 높은 연산적 성능이 전형적으로 높은 전력 소모와 짧은 배터리 수명을 초래하기 때문에 시각적 디스플레이 성능을 제한한다. 휴대용 컴퓨터는 프로세스가 이용되지 않을 때 전력 소비를 줄이기 위해서 프로세서 활동을 자동적으로 낮추는 소프트웨어를 포함한다. 어떤 컴퓨터 모델에서 사용자는 수동으로 프로세서의 동작을 낮추게 할 것이다. 예컨대, Sony의 VGN-SZ280P 랩탑은 일측면에 "스태미너(Stamina)"(낮은 성능을 위해, 보다 긴 배터리 수명) 및 다른 측면에 "스피드(Speed)"(높은 성능을 위해, 보다 짧은 배터리 수명)라고 라벨된 스위치를 포함한다. 휴대용 컴퓨터에서 OS 구동은 컴퓨터가 그것의 최고조 성능의 일부에서 동작하는 경우에도 사용가능하게 기능할 수 있도록 해야한다. 따라서, OS 그래픽 성능은 자주 최신 기술의 이용가능한 컴퓨터적인 성능보다 훨씬 아래 놓인다.

[0030] Maya와 같은 하이-엔드(high-end) 컴퓨터적으로 집중된(intense) 어플리케이션은 종종 고성능 PC에서 사용될 것이라는 기대를 갖고 판매된다. 이는 일반적으로 매우 높은 성능, 더 많은 비용과 적은 휴대성이라는 적어도 공통 분모 요구 사항을 확립한다. 이는 결과적으로, 그러한 어플리케이션은 일반 목적의 OS보다 훨씬 제한된 타켓 관중을 갖고(또는 일반 목적 생산 어플리케이션, Microsoft Office와 같은), 전형적으로 일반 목적 OS 소프트웨어 또는 일반 목적 어플리케이션 소프트웨어 보다 훨씬 적은 규모로 팔린다. 장래 사용자가 사전에 그러한 컴퓨터적으로 집중된 어플리케이션을 시도하는 것이 어렵기 때문에 잠재 소비자들은 더욱 제한된다. 예컨대, 한 학생이 Maya를 어떻게 사용하는지 배우기를 원한다고 고려하거나 이미 그러한 어플리케이션에 대한 지식이 있는 잠재적 구매자가 구입에 투자하기에 앞서서 Maya를 사용하도록 시도하길 원하는(이는 Maya를 구동할 수 있는 하이-엔드 컴퓨터를 또한 구입하는 것을 포함할 것이다) 것을 고려해 본다. 반면에 학생이나 잠재적인 구매자가 다운로드를 할 수도 있겠지만, Maya의 데모버전의 물리적 미디어 카피를 얻을 수 있고, 만약 그들은 컴퓨터의 전체 잠재력(full potential)(예컨대, 복잡한 3D 장면을 다루는 것)으로 Maya를 구동하는 컴퓨터의 성능이 부족하면, 그때 그들은 제품의 정보를 완벽하게 평가를 할 수 없게 될 것이다. 이는 실질적으로 그러한 하이-엔드 어플리케이션을 위한 소비자를 제한한다. 또한 개발 비용이 일반 목적 어플리케이션보다 매우 적은 수량의 구입으로 분할되기 때문에 높은 판매 가격을 형성하게 할 것이다.

[0031] 또한, 고가의 어플리케이션은 개인 및 기업이 어플리케이션 소프트웨어의 해적판 복사를 하는 것에 대한 인센티브(incentive)를 더 만들 수 있다. 그 결과로서, 다양한 기술에 의해서 그러한 불법 복제를 완화하기 위한 그러한 소프트웨어의 출판인의 상당한 노력에도 불구하고 하이-엔드 어플리케이션 소프트웨어는 만연된 불법 복제를 겪는다. 여전히, 불법 복제된 하이-엔드 어플리케이션을 사용할 때, 사용자들은 불법 복제 카피를 동작하기 위해 값비싼 최신 기술의 PC에 투자하기 위한 필요를 제거할 수 없다. 그래서, 그들이 실제 소매 가격의 일부로 소프트웨어 어플리케이션의 사용을 얻을 수 있는 동안, 불법 복제된 소프트웨어의 사용자들은 여전히 그 어플리케이션을 충분히 이용하기 위해서 값비싼 PC를 얻거나 사는 것이 필요하다.

[0032] 이는 높은-성능 불법 복제된 비디오 게임의 사용자에게 동일한 사실이다. 비록 해적들이 실제 가격의 극히 일부로 게임을 얻을 수 있지만, 그들은 여전히 게임을 적절히 플레이하기 위해서 필요한 고가의 컴퓨팅 하드웨어(예컨대, 향상된 GPU의 PC 또는 Xbox 360과 같은 하이-엔드 비디오 게임 콘솔)를 사는 것이 요청된다. 비디오 게임은 일반적으로 소비자를 위한 취미 생활이 되면, 하이-엔드 비디오 게임 시스템을 위한 추가 비용이 금지될 수 있다. 이러한 상황은 노동자의 평균 연수입이 현재 대체적으로 미국에 비하여 꽤 낮은 나라(예컨대, 중국)에서 악화된다. 그 결과, 전체 인구의 아주 작은 비율이 하이-엔드 비디오 게임 시스템 또는 하이-엔드 PC를 소유하고 있다. 그러한 나라에서, 사용자가 인터넷에 연결된 컴퓨터의 사용을 위해 요금을 지불하는, "인터넷 카페"는 꽤 보편적이다. 자주, 이러한 인터넷 카페는 플레이어가 컴퓨터적으로 집중된 비디오 게임을 플레이를 할 수 있

는 GPU와 같은 높은 성능 특징이 없는 오래된 모델 또는 로우-엔드 PC를 보유한다. 이는 대단히 성공한 Vivendi의 "World of Warcraft"와 처럼 로우-엔드 PC에서 동작하는 게임의 성공의 주된 요소이고 중국에 인터넷 카페에서 흔하게 플레이된다. 반면, "Second life"와 같은 컴퓨터적으로 집중된 게임은, 중국 인터넷 카페에서 인스톨된 PC에서 플레이하는 것이 상당히 쉽지 않다. 이러한 게임은 인터넷 카페에서 낮은 성능 PC에 대한 접근만 가능한 사용자에게는 사실상 받아들여질 수 없다.

[0033] 또한, 비디오 게임 구입을 고려하는 사용자에게 장벽이 존재하고, 사용자는 홈에서 인터넷을 통해서 데모를 다운로드함으로써 게임의 데모 버전을 시도하기 쉬울 것이다. 비디오 게임 데모는 종종 일부 기능이 중지되거나 제한된 게임 플레이의 크기를 가진 게임의 본격적인 버전이다. 이는 게임이 PC 또는 콘솔에서 실행되거나 설치되기 전에 수 기가바이트의 데이터를 다운로딩하는 긴 과정(아마도 시간)을 포함할 것이다. PC의 경우에, 게임을 위해 특별한 드라이버(예컨대, DirectX, 또는 OpenGL)가 필요할지에 대해 이해하고, 올바른 버전을 다운로드하며, 그것을 설치하고, 그 후 PC가 그 게임을 플레이할 수 있는지를 결정하는 것을 포함할 것이다. 마지막 단계는 PC가 충분한 처리(CPU 및 GPU)성능, 충분한 RAM, 호환가능한 OS(예컨대, Window XP에서 구동되는 어떤 게임, 그러나 Vista에서 구동되지 않음)를 가졌는지 결정하는 것을 포함할 것이다. 따라서, 비디오 게임 데모를 실행하려고 시도하는 긴 과정 후, 사용자는 그 비디오 게임 데모가 잘 플레이되지 않을 수 있다는 것과, 주어진 사용자의 PC 환경설정에 대해서 이해할 수 있을 것이다. 더 심각하게, 사용자가 데모를 시행하기 위해 새로운 드라이버를 다운로드하게 되면, 이 드라이버 버전은 사용자가 PC에서 정기적으로 사용하는 다른 게임 또는 어플리케이션과 호환가능하지 않을지도 모르고, 따라서 데모의 설치는 미리 게임을 동작가능하게 하거나 어플리케이션을 동작가능하지 않게 할 것이다. 사용자를 당혹하게 하는 이러한 장벽뿐만 아니라, 그들은 게임을 판매하기 위한 비디오 게임 소프트웨어 출판인 및 비디오 게임 개발자에게 장벽을 만든다.

[0034] 경제적 비효율의 결과인 또 다른 문제는 주어진 PC 또는 게임 콘솔이 보통 어플리케이션 및/또는 게임을 위해 어떤 수준의 성능 요구를 수용하도록 디자인되었는지에 관한 사실과 관련되어 있다. 예컨대, 만약 그들이 전혀 GPU를 갖고 있지 않다면 어떤 PC는 더 많은 혹은 적은 RAM, 더 느린 또는 빠른 CPU, 및 더 느린 또는 빠른 GPU를 가진다. 어떤 게임 또는 어플리케이션은 주어진 PC 또는 콘솔의 충분한 컴퓨팅 파워의 이점을 가져다준다, 반면에 많은 게임 및 어플리케이션은 아니다. 만약 어떤 사용자의 게임 또는 어플리케이션 선택은 로컬 PC 또는 콘솔의 최고 수행 성능의 부족에 빠진다면, 사용자는 이용될 수 없는 특징을 위해 PC 또는 콘솔에 돈을 낭비하게 될 것이다. 콘솔의 경우에, 콘솔 제조자는 콘솔 비용으로 보조금을 지급하기 위해 필요했던 것보다 더 많이 지불하게 될 것이다.

[0035] 비디오 게임의 마케팅과 즐거움에 존재하는 또 다른 문제는 사용자가 그 게임을 구입을 하기 전에 그 밖의 플레이 게임을 볼 수 있도록 허락하는 것을 포함한다. 몇 가지 선행 기술 접근은 후에 리플레이를 위한 비디오 게임의 레코딩을 위해 존재한다. 예컨대, 미국 특허 제5,558,339호는 비디오 게임 클라이언트 컴퓨터(동일한 또는 다른 사용자에 의해서 소유된 것)에서 "게임플레이" 동안에 게임 컨트롤러 동작을 포함하는 게임 상태 정보를 기록하는 것에 대해 알려준다. 이러한 상태 정보는 나중에 비디오 게임 클라이언트 컴퓨터(예컨대, PC 또는 콘솔)에서 게임 동작의 일부 또는 전부를 리플레이하기 위해 사용될 수 있다. 이 접근 방법의 현저한 단점은 사용자가 기록된 경기를 보기 위해서, 사용자는 게임을 플레이를 할 수 있는 비디오 게임 클라이언트 컴퓨터를 보유해야 하고 컴퓨터에서 동작하는 비디오 게임 어플리케이션을 소유해야하므로, 게임플레이는 기록된 게임의 상태가 리플레이된 때와 동일하다. 그 외에도, 비디오 게임 어플리케이션은 기록된 게임과 재생되는 게임 사이의 가능한 실행 차이가 존재하지 않는 방식으로 쓸 수 있다.

[0036] 예컨대, 게임 그래픽은 일반적으로 프레임마다 계산된다. 많은 게임에서, 게임 로직은 때때로 한 장면이 특히 복잡한지, 또는 만약 슬로우 다운 실행(예컨대, PC에서 다른 처리가 게임 어플리케이션으로부터 CPU 사이클을 가져가도록 실행될 것이다)과 같은 그 밖의 지연이 있는지에 따라, 다음 프레임을 디스플레이하기 위한 그래픽을 연산하기 위해 한 프레임 시간보다 짧거나 길게 걸릴 수 있을 것이다. 그러한 게임에서, 한 프레임 시간(소위, CPU 클럭 사이클보다 적음)보다 훨씬 적은 시간에 연산되는 "문턱(threshold)"프레임이 결국 일어날 수 있다. 동일한 장면은 정확히 동일한 게임 상태 정보를 사용하여 다시 연산될 때, 한 프레임 시간(예컨대, 만약 내부 CPU 버스가 외부 DRAM 버스와 조금 위상이 벗어나고(out of phase) 그것이 CPU 사이클 시간의 지연을 유도한다면, 그럼에도 불구하고 게임 처리에서 CPU 시간의 수 밀리세컨드(millisecond)가 걸리는 다른 처리에서 큰 지연이 있지 않다)보다 적은 CPU 클럭 사이클이 걸릴 수 있다. 따라서, 게임이 플레이 백(play back)된 때, 프레임은 단일 프레임 시간 보다 2개 프레임 시간으로 연산되어 얻어진다. 어떤 동작은 게임이 얼마나 자주 새로운 프레임(예컨대, 게임이 게임 컨트롤러로부터 입력을 샘플할 때)을 연산하는지에 기초한다. 게임이 재생되는 동안, 다른 동작을 위한 시간 기준에서 이러한 차이는 게임 플레이에 영향을 주지 않지만, 그것은 플레이백 게임

에서 다른 결과를 생산하는 결과를 초래할 수 있다. 예컨대, 만약 농구공의 탄도는 안정적으로 60fps 속도로 연산되지만, 게임 컨트롤러 입력은 연산된 프레임의 속도에 기초하여 샘플되고, 게임이 기록된 때에 연산된 프레임의 속도는 53fps 일 것이지만 게임이 리플레이될 때는 52fps이므로, 이는 농구공이 바스켓으로 들어가는 것이 차단되는지 또는 아닌지 사이에 차이를 만들 수 있어, 다른 결과를 초래한다. 따라서, 비디오 게임을 기록하기 위해 게임 상태를 사용하는 것은 정확히 동일한 결과를 생산하는 동일한 게임 상태 정보를 사용하고, 리플레이를 보장하기 위해 매우 신중한 게임 소프트웨어 디자인이 요구된다.

[0037] 비디오 게임을 기록하기 위한 다른 종래 기술 접근은 단지 PC 또는 비디오 게임 시스템(예컨대, PC에서 비디오 캡처 보드를 위해 또는, VCR 또는 DVD 레코더를 위해서)의 비디오 출력을 단순히 기록하기 위한 것이다. 그 다음 비디오는 되감고 리플레이될 수 있거나, 또는 택일적으로, 전형적으로 압축된 후에 인터넷으로 업로드된 기록된 비디오일 수 있다. 이러한 접근의 단점은 3D 게임 시퀀스가 플레이 백될 때, 사용자는 시퀀스가 기록된 하나의 시점에서 시퀀스를 볼 수 있도록 제한된다. 다르게 말해서, 사용자는 장면의 시점을 변화할 수 없다.

[0038] 더욱이, 홈 PC 또는 게임 콘솔에서 플레이된 기록된 게임 시퀀스의 압축된 비디오가 인터넷을 통해 그 밖의 사용자에게 이용가능하도록 만들어진 때, 비디오가 실시간으로 압축되었음에도 불구하고, 그것은 인터넷으로 실시간으로 압축 비디오를 업로드하는 것이 불가능할 것이다. 그 이유는 전세계에 많은 가정은 높은 비대칭의 광대역 연결을 갖는 인터넷으로 연결된다(예컨대, DSL 및 케이블 모뎀은 전형적으로 업스트림 대역폭보다 다운스트림 대역폭이 훨씬 크다). 압축된 고해상도 비디오 시퀀스는 종종 네트워크의 업스트림 대역폭 성능보다 더 높은 대역폭을 갖고, 실시간으로 업로드하는 것을 불가능하게 만든다. 따라서, 게임 시퀀스가 플레이된 후에 인터넷에서 다른 사용자가 그 게임을 시청할 수 있기 전에 현저한 지연이 있을 것이다(아마도 수 분 또는 수 시간). 비록 이러한 지연은 어떤 상황에서는 견딜 수 있다(예컨대, 이전 시간에 일어난 게임 플레이어의 결과를 보는 것), 이는 게임을 실시간으로 볼 수 있는 성능을 없애거나(예컨대, 챔피언 플레이어에 의해서 플레이되는 농구 토너먼트), 게임이 라이브 플레이할 때는 "즉각적인 리플레이" 성능을 제거한다.

[0039] 또 다른 선행 기술 접근은 텔레비전 수신기를 구비한 시청자가 라이브로 비디오 게임을 시청할 수 있도록 하지만, 오직 텔레비전 제작자들의 제어하에 놓이게 된다. 일부 텔레비전 채널은 미국과 다른 나라에서 비디오 게임 관람 채널을 제공하고, 거기서 텔레비전 시청자는 비디오 게임 채널에서 어떤 비디오 게임 사용자(예컨대, 토너먼트에서 최상위 등급 선수 플레이)를 볼 수 있다. 이는 텔레비전 채널을 위한 처리 기구 및 비디오 분배에 반영되는 비디오 게임 시스템(PC 및/또는 콘솔)의 비디오 출력을 가짐으로써 달성된다. 이는 텔레비전 채널이 농구 코트 주변에 다른 각도를 반영하는 몇 개 카메라가 제공하는 라이브 농구 경기를 실시간으로 방송하는 것과는 다르다. 다음으로 텔레비전 채널은 다양한 비디오 게임 시스템으로부터 출력을 다루기 위해서 비디오/오디오 처리 및 효과 기구의 사용을 만들게 할 수 있다. 예컨대, 텔레비전 채널은 (라이브 농구 게임 동안 텍스트를 오버레이(overlay)할 수 있는 것과 처럼) 다른 플레이어의 상태를 가리키는 비디오 게임으로부터 비디오 상에 텍스트를 오버레이할 수 있고, 텔레비전 채널은 게임 중에 일어나는 동작에 대해 논의할 수 있는 해설자의 오디오에 다중 녹음(overdub)을 할 수 있다. 추가적으로, 비디오 게임 출력은 게임의 실제 플레어의 게임의 카메라 녹화된 비디오와 조합할 수 있다(예컨대, 게임에 대한 그들의 감정적인 응답을 보여주는).

[0040] 이러한 접근의 문제는 생방송의 흥분을 갖기 위해서 그러한 라이브 비디오 공급(feeds)이 실시간으로 텔레비전 채널의 비디오 분배 및 처리 기구를 이용해야 한다는 것이다. 그러나, 앞서 논의한 것처럼, 특히 만약 방송의 일부가 게임 플레이어의 실제 비디오를 캡처링하는 카메라로부터 라이브 비디오를 포함한다면, 가정에서 비디오 게임 시스템이 동작할 때 이는 불가능하게 된다. 또한, 토너먼트 상황에서, 앞서 언급한 것처럼 가정에 있는 게이머가 게임을 수정하고 속일 수 있는 우려가 있다. 이러한 이유로, 텔레비전 채널에 그러한 비디오 게임 방송은 종종 잠재적인 라이브 카메라 및 복수의 비디오 게임 시스템에서 반영된 비디오를 받아들일 수 있는 텔레비전 생산 장비가 있는 공통 위치(예컨대, 텔레비전 스튜디오 또는 운동장)에서 합쳐진 비디오 게임 시스템 및 플레이어와 배열된다.

[0041] 비록 그러한 선행 비디오 게임 TV 채널은 예컨대 라이브 스포츠 이벤트와 유사한 경험인 매우 흥분되는 발표를 텔레비전 시청자에게 제공할 수 있고, "운동선수"로 나타나는 비디오 게임 플레이어와, 비디오 게임 세계에서 그들의 동작 형태 및 실제 세계에서 그들의 동작 형태 모두를, 이러한 비디오 시스템은 플레이어가 다른 것과 물리적으로 매우 가까운 곳에 있는 상황으로 제한된다. 그리고, 텔레비전 채널이 방송된 이후에, 각 방송된 채널은 오직 하나의 비디오 스트림으로 보일수 있고, 이는 텔레비전 생산 업체에 의해서 선택된다. 이러한 제한과 방송 시간, 생산 장비 및 생산 업체의 높은 비용으로 인하여 정상권 토너먼트에서 정상권 플레이어 경기만을 오직 시청할 수 있다.

[0042] 또한, 텔레비전 시청자에게 비디오 게임의 풀 스크린 이미지를 방송하는 기존 텔레비전 채널은 한번에 오직 하나의 비디오 게임을 보인다. 이는 심하게는 텔레비전 시청자의 선택을 제한한다. 예컨대, 텔레비전 시청자는 정해진 시간에 게임을 보는 것에 흥미가 없을 수 있을 것이다. 다른 시청자는 주어진 시간에 텔레비전 채널에 의해서 특징적이지 않는 특정 플레이어의 게임 플레이를 보는 것에 오직 관심이 있을 수 있다. 다른 경우에, 시청자는 전문 플레이어가 게임에서 특정 수준을 어떻게 다룰수 있는지에 관하여 관심이 있을 수 있다. 여전히 다른 시청자는 제작팀에 의해서 선택된 것과 다른, 비디오 게임이 보여지는 관점을 제어하길 바랄지도 모른다. 간단히 말해서, 텔레비전 시청자는 비록 몇 개 다른 텔레비전 채널이 이용가능할지라도, 텔레비전 네트워크의 특정 방송에 의해서 공급되지 않는 비디오 게임을 시청하는 것에 많은 선호를 가질 것이다. 전술한 모든 이유 때문에, 선행 기술 비디오 게임 텔레비전 채널은 텔레비전 시청자에게 비디오 게임을 나타내는 것에 상당한 제한을 가지고 있다.

[0043] 선행 기술 비디오 게임 시스템 및 어플리케이션 소프트웨어 시스템의 다른 단점들은 그것들이 복잡하고, 흔히 오류, 충돌 및/또는 의도되지 않고 선호되지 않는 동작(총괄해서, "버그")를 겪는다. 비록 게임 및 어플리케이션이 출시 전에 일반적으로 디버깅, 튜닝 과정(흔히 "소프트웨어 품질 보증(Software Quality Assurance)" 또는 SQA)을 겪을지라도, 거의 변함없이 게임 또는 어플리케이션이 넓은 청중에게 출시되면 그 분야에서 버그가 발생한다. 불행하게도 소프트웨어 개발자가 제품 출시 후에 많은 버그를 찾아내고 식별하는 것은 굉장히 어렵다. 소프트웨어 개발자가 버그를 알아내는 것도 굉장히 어렵다. 그들이 버그에 대해 알게 되더라도, 그 버그의 원인이 무엇인지를 인식하기 위해 그것들을 이용할 수 있는 많은 정보는 제한될지 모른다. 예컨대, 어떤 사용자가 게임 개발자의 고객 서비스로 전화를 걸 것이고 게임 플레이를 할 때 스크린이 깜빡이기 시작하고 솔리드 블루 컬러로 바뀌면서 PC가 멈춰버린다는 것을 진술하는 메세지를 남길 것이다. 이는 버그를 찾는데 매우 적은 유용한 정보를 SQA 팀에 제공한다. 온라인에 연결되는 일부 게임 또는 어플리케이션은 가끔 어떤 경우에 더 많은 정보를 제공할 수 있다. 예컨대, "감시(watchdog)" 프로세스는 때때로 "충돌(crashes)"에 대해 게임 또는 어플리케이션을 감시하기 위해 사용될 수 있다. 그것이 충돌되고 인터넷을 통해 SQA팀에게 정보를 업로드할 때, 감시 프로세서는 게임 또는 어플리케이션 프로세서(예컨대, 스택의 상태, 메모리 사용량, 게임 또는 어플리케이션이 얼마나 진척되었는지 등)의 상태에 대해 통계를 수집할 수 있다. 하지만 복잡한 게임 또는 어플리케이션에서, 그러한 정보는 충돌의 시간에 사용자가 사용할지를 정확하게 결정하기 위해 판독하는 것은 매우 긴 시간이 걸릴 수 있다. 설령 그렇더라도, 충돌을 이끄는 연속적인 이벤트가 무엇인지를 결정하는 것은 가능할 것이다.

[0044] PC 및 게임 콘솔과 관련된 다른 문제는 소비자에게 매우 불편한 서비스 이슈에 관한 것이다. 또한, 서비스 이슈는 전형적으로 그들이 파손된 PC 또는 콘솔을 안전하게 선적하기 위해서 특정한 상자에 보내도록 요구되고, 그 후에 만약 PC 또는 콘솔이 보증기간 내이면 수리 비용을 초래하게 된다. 게임 또는 어플리케이션 소프트웨어 출판인은 또한 수리 상태에 놓인 PC 및 콘솔에 의해서 판매의 손실(또는 온라인 서비스 사용)에 따른 영향을 받을 수 있다.

[0045] 도 1은 소니 플레이 스테이션® 3, 마이크로소프트 X 박스 360®, 닌텐도 Wii™, 윈도우 기반의 개인용 컴퓨터 또는 애플 매킨토시와 같은 종래 기술의 비디오 게이밍 시스템을 도시하고 있다. 이러한 각각의 시스템은 프로그램 코드를 실행하기 위해서 중앙 처리 장치(CPU), 전형적으로 진보된 그래픽 연산을 수행하기 위한 그래픽 처리 장치(GPU), 외부 장치 및 사용자와 통신(communication)하기 위한 다중 형태의 입력/출력(I/O)을 포함한다. 간단화를 위해, 이러한 구성은 단일 유닛(100)과 함께 결합된 것으로 도시된다. 또한, 도 1의 선행 기술 비디오 게임 시스템은 광학 미디어 드라이버(104)(예컨대, DVD-ROM 드라이브); 비디오 게임 프로그램 코드 및 데이터를 저장하기 위한 하드 드라이브(103); 멀티 플레이어 게임 플레이를 하고, 게임, 패치, 데모, 또는 그 밖의 미디어를 다운로드하기 위한 네트워크 연결(105); CPU/GPU(100)에 의해서 현재 실행되는 프로그램 코드를 저장하기 위한 랜덤 엑서스 메모리(RAM; 101); 게임을 플레이하는 동안 사용자로부터 입력 명령어를 수신하기 위한 게임 컨트롤러(106); 및 디스플레이 장치(102)(예컨대, SDTV/HDTV 또는 컴퓨터 모니터)를 포함하는 것을 도시한다.

[0046] 도 1에 도시된 선행 기술 시스템은 몇 가지 제한을 겪는다. 첫번째, 광학 드라이브(104) 및 하드 드라이브(103)는 RAM(101)과 비교했을 때 훨씬 느린 접근 속도를 보인다. RAM(101)을 통해서 직접 동작할 때, 실제적으로 CPU/GPU(100)는 하드 드라이브(103) 또는 광학 드라이브(104)로부터 직접 데이터 및 프로그램 코드를 얻을 때 가능한 것보다 초당 보다 훨씬 많은 폴리건을 처리한다. 그 이유는 RAM(101)이 일반적으로 보다 높은 대역폭을 갖고 디스크 매커니즘의 긴 탐색 지연을 대체로 겪지 않기 때문이다. 하지만 오직 제한된 용량의 RAM이 이러한 선행 기술 시스템에 제공된다(예컨대, 256 ~ 512 Mbytes). 따라서, 비디오 게임의 다음 장면을 위한 데이터로 주기적으로 채워지는 연속 RAM(101)에서 "로딩..." 시퀀스가 자주 요구된다.

[0047] 어떤 시스템은 게임플레이와 동시에 프로그램 코드의 로딩을 오버랩하기 위해 시도하나, 이는 알려진 이벤트들

의 시퀀스가 있을 때만 행해질 수 있다(예컨대, 차가 도로를 따라 내려가면, 도로 측면 상의 빌딩들에 접근하기 위한 구조는 차가 운행하는 동안 로드될 수 있다). 복잡한 그리고/또는 신속한 화면의 변화를 위해, 이러한 형태의 오버랩핑은 보통 동작하지 않는다. 예컨대, 사용자가 전쟁 중에 있고 RAM(101)이 그 순간에 장면 내에 물체들을 나타내는 데이터로 완전히 채워지는 경우에 있어서, 사용자가 RAM(101) 내에 현재 로드되지 않은 물체를 보기 위해 장면들을 왼쪽으로 이동시킨다면, 새로운 객체들을 하드 드라이브(103) 또는 광학 미디어(104)로부터 RAM(101)으로 로드할 시간이 없기 때문에, 동작에는 불연속성이 생길 수 있을 것이다.

[0048] 도 1의 시스템의 또 다른 문제점은 하드 드라이브(103) 및 광학 미디어(104)의 저장 용량의 제한으로 인해 발생한다. 디스크 저장 장치가 상대적으로 큰 저장 공간(예컨대, 50 기가바이트 이상)으로 제조될 수 있음에도, 여전히 현재 비디오 게임들 내에서 직면하는 소정 시나리오들을 위한 저장 공간을 충분히 제공하지 않는다. 예컨대, 상기한 바와 같이, 축구 비디오 게임은 사용자가 그 세계를 통하여 12개의 팀들, 선수들 및 스타디움들 중에서 선택하는 것을 허락할 수 있다. 각 팀, 각 선수 및 각 스타디움을 위해 수많은 문서 맵들(maps) 및 환경 맵들이 그 세계 내에서 3D 외관을 특징화하기 위해 필요하다(예컨대, 각 팀은 고유의 운동 셔츠를 갖고, 각각 고유의 문서 맵을 요구한다).

[0049] 이러한 후자의 문제점을 다루기 위해 사용되는 하나의 기술은 그들이 사용자에 의해 선택되자마자 게임이 문서와 환경 맵들을 미리 컴퓨팅하는 것이다. 이는 신장되는 이미지들, 3D 맵핑(mapping), 쉐이딩(shading), 데이터 구조들을 조직하는 것 등을 포함하는, 여러 컴퓨터적으로 집약적인 프로세스들을 포함할 수 있다. 결과적으로, 비디오 게임이 이러한 연산들을 수행하는 동안에 사용자를 위한 지연이 있을 수 있다. 원칙적으로, 이러한 지연을 감소시키는 하나의 방법은 게임이 최초로 개발되었을 때, 팀, 선수 로스터(roster), 및 스타디움의 모든 변경을 포함하는 이러한 연산들을 전부 수행하는 것이다. 그 후에 게임의 출시된 버전은 사용자가 선택할 때 인터넷을 통해서 하드 드라이브(103)로 다운로드되는 스타디움 선택, 플레이어 로스터, 팀에 대해 선택된 이미 처리된 데이터를 인터넷에 하나 이상의 서버 또는 광학 미디어(104)에 저장된 이러한 미리 처리된 데이터 모두에 포함할 것이다. 실질적인 문제로서, 그러나 게임플레이 내에서 가능한 모든 치환들의 미리 로드된 이러한 데이터는 손쉽게 수 테라바이트(terabytes)일 수 있고, 이는 오늘날의 광학 미디어 장치들의 용량을 훨씬 초과한다. 더욱이, 주어진 팀, 선수 로스터, 스타디움 선택을 위한 데이터는 쉽게 수백 메가바이트(megabytes)의 데이터 이상이 될 수 있다. 소위 10Mbps의 홈 네트워크 연결로, 데이터를 국지적으로 계산하는 것보다 네트워크 연결(105)을 통하여 이러한 데이터를 다운로드하는 것이 더 오래 걸릴 수 있다.

[0050] 따라서, 도 1에 도시된 종래의 게임 구조는 사용자에게 복잡한 게임들의 주요 장면 변환들 사이의 현저한 지연들을 생기게 한다.

[0051] 도 1에 도시된 바와 같이, 종래 기술의 또 다른 문제점은 수년에 걸쳐, 비디오 게임들이 더 개선되고, 더 많은 CPU/GPU 프로세싱 파워를 요구하는 경향을 갖는다는 것이다. 따라서, RAM의 양이 무제한으로 있다고 가정하면, 비디오 게임 하드웨어 요구사항은 이러한 시스템 내에서 이용할 수 있는 최고 수준의 프로세싱 파워를 넘어설 것이다. 결과적으로, 사용자는 페이스를 유지하기 위해(또는 낮은 질의 수준에서 새로운 게임을 플레이하거나) 매 수년마다 게이밍 하드웨어를 업그레이드하는 것이 요구된다. 보다 개선된 비디오 게임을 위한 경향 중의 중요한 결론은, 그들의 비용이 보통 그들이 지원할 수 있는 최고의 수행 게임의 요구사항들에 의해 결정되기 때문에 집에서 사용하기 위한 비디오 게임 플레잉 머신들은 전형적으로 경제적으로 비효율적이라는 것이다. 예컨대, Xbox 360은 수백 메가바이트의 RAM과 높은 성능 CPU, GPU를 요구하는 "Gears of War"와 같은 게임을 플레이하기 위해 사용될 수 있고, 또는 Xbox 360은 매우 낮은 수행 CPU와 단지 수 킬로바이트(kilobytes)의 램을 요구하는 1970년대의 게임 팩 맨(Pac Man)을 플레이하기 위해 사용될 수도 있다. 실제로, Xbox 360은 동시에 진행되는 팩 맨 게임들을 한번에 호스팅하기 위해 충분한 컴퓨팅 파워를 갖는다.

[0052] 비디오 게임 기계들은 전형적으로 한 주의 시간들 대부분 동안 꺼져 있다. 13세 이상의 활동적인 게이머들에 대한 2006년 6월 Nielsen Entertainment 연구에 따르면, 평균적으로 활동적인 게이머들은 콘솔 비디오 게임들을 플레이하기 위해 주당 14시간을 쓰고, 또는 한 주의 전체 시간의 12%를 사용한다. 이는 평균 비디오 게임 콘솔이 88%의 시간 동안 동작하지 않음을 의미하고, 값비싼 자원의 비효율적인 사용을 의미한다. 이는 비디오 게임 콘솔들이 종종 소비자 가격을 내리기 위해 공급자들에 의해 보조금을 받는 것으로써 특별히 중요하다(보조금은 미래 비디오 게임 소프트웨어 구매로부터 로열티에 의해 다시 벌어질 수 있다는 기대와 함께).

[0053] 비디오 게임 콘솔들은 또한 거의 어떠한 소비자 전자 장치와 연관된 비용들을 초래한다. 예컨대, 시스템들의 전자공학 및 메카니즘은 동봉되어 하우징(housing)될 필요가 있다. 제조업자는 서비스 보증서를 제공할 필요가 있다. 시스템을 파는 소매업자들은 시스템의 어느 판매 및/또는 비디오 게임 소프트웨어의 판매의 이윤을 수집할

필요가 있다. 이러한 모든 요소들은 제조업자에 의해 보조금이 제공되거나, 소비자에게 전가되거나, 둘 다일 수 있는 비디오 게임 콘솔의 비용에 더해진다.

[0054] 더욱이, 불법 복제는 비디오 게임 산업에서 주요 문제점이다. 사실상 모든 주요 비디오 게이밍 시스템상에서 사용되는 보안 메카니즘(security mechanisms)은 연도에 걸쳐 "크랙(crack)" 되고, 비디오 게임의 공인되지 않은 불법 복제를 야기한다. 예컨대, Xbox 360 보안 시스템은 2006년 7월 내에 크랙되었고, 사용자는 이제 불법 복제물을 온라인으로 다운로드할 수 있다. 다운로드 가능한(예컨대, PC 또는 Mac을 위한 게임들) 게임들은 불법 복제에 부분적으로 약점이 있다. 불법 복제가 약하게 정책화된 세계의 어떤 지역에 있어서, 사용자들이 비용의 아주 작은 부분으로 합법적인 것만큼 쉽게 불법 복제물을 살 수 있기 때문에 본질적으로 독립형 비디오 게임 소프트웨어를 위한 어떠한 실행가능한 시장도 없다. 또한, 세계의 많은 지역에서 게임 콘솔의 비용이 수입의 높은 비율을 차지함에 따라, 침해가 제어된다고 하더라도, 얼마 안되는 사람들만이 기술분야의 게이밍 시스템을 살 여유가 있을 수 있다.

[0055] 더욱이, 사용되는 게임 시장은 비디오 게임 산업을 위한 수익을 감소시킨다. 사용자가 게임에 지루해질 때, 그들은 다른 사용자들에게 게임을 되팔 수 있는 상점에 게임을 팔 수 있다. 이러한 공인되지 않았으나 보통의 실행은 게임 발행자들의 수익을 현저하게 감소시킨다. 비슷하게, 50%의 주문상 판매의 감소는 보통 몇 년마다 플랫폼 변화가 있을 때 발생한다. 이는 사용자들이 더 새로운 버전 플랫폼이 막 출시되려고 할 때 더 오래된 플랫폼들을 위한 게임들을 사는 것을 멈추기 때문이다(예컨대, 플레이 스테이션 3가 막 출시되려할 때, 사람들은 플레이 스테이션 2 게임들을 사는 것을 멈춘다). 종합하면, 판매의 감소 및 새로운 플랫폼과 연관된, 증가하는 개발 비용들은 게임 개발자들의 적합성에 매우 중대한 불리한 영향을 가질 수 있다.

[0056] 또한 새로운 게임 콘솔들은 매우 비싸다. Xbox 360, Nintendo Wii, 및 소니 플레이스테이션 3는 모두 수백 달러에 팔린다. 고전력 PC 게이밍 시스템들은 8000달러 이상으로 비용이 들 수 있다. 이는 특별히 많은 시스템이 아이들에게 판매되었다는 사실 및 하드웨어는 몇 년 뒤에 못 쓰게 되는 것을 고려했을 때 사용자들을 위한 중요한 투자임을 나타낸다.

[0057] 예상되는 문제로 하나의 접근법은 게이밍 프로그램 코드 및 데이터가 서버상에서 호스팅되고, 디지털 브로드밴드(broadband) 네트워크에 걸쳐 스트리밍되는 압축된 비디오와 오디오로서 주문 클라이언트 머신(machine)으로 전송되는 온라인 게이밍이다. 핀란드의 G-Cluster(지금 일본의 SOFTBANK Broadmedia의 자회사)와 같은 소정 기업들은 현재 이러한 온라인 서비스들을 제공한다. 유사한 게이밍 서비스들은 DSL 및 케이블 텔레비전 공급자들에 의해 제공되고 호텔들 내의 네트워크와 같은 로컬 네트워크에서 사용가능하다. 이러한 시스템들의 주요 약점은 레이턴시의 문제이고, 즉, 이는 신호가 오퍼레이터의 "전파중계소(Head-end)" 내에 전형적으로 위치하는 게이머 서버를 왕복하는 데 걸리는 시간을 말한다. 빠른 액션 비디오 게임들(또한 "twitch" 비디오 게임들로 알려져 있는)은 사용자가 게임 컨트롤러를 가지고 액션을 수행하는 시간과 디스플레이 스크린이 사용자 동작의 결과를 업데이트해서 보여주는 시간 사이의 매우 낮은-레이턴시를 요구한다. 낮은-레이턴시는 사용자가 게임이 "즉각(instantly)" 응답하는 지각력(perception)을 갖도록 하기 위해 필요하다. 사용자들은 사용자 스킬 레벨과 게임 형태(type)에 의존하여 다른 레이턴시 간격에 만족할 수 있다. 예컨대, 100ms의 레이턴시는 느린 캐주얼 게임(서양주사위놀이(backgammon)와 같은) 또는 느린-액션 롤 플레잉 게임(role playing game)에 꽤 좋지만, 빠른 액션 게임에 있어서, 70 또는 80ms 초과의 레이턴시는 사용자가 게임 내에서 더 좋지 않게 수행하는 것을 야기시킬 수 있고, 따라서 이는 받아들일 수 없다. 예컨대, 빠른 반응 시간이 요구되는 게임 내에서는, 레이턴시가 50 에서 100ms로 증가할 때 정확도에서 가파른 경사가 있다.

[0058] 게임 또는 어플리케이션 서버가 가까이에 설치될 때, 제어된 네트워크 환경, 또는 사용자까지 네트워크 경로는 예측할 수 있고 그리고/또는 대역폭 피크를 견딜 수 있으며, 그것은 최대 레이턴시의 관점 및 레이턴시의 일관성의 관점에서 모두 레이턴시를 제어하는 것이 훨씬 쉽다(예컨대, 그래서 사용자는 네트워크를 통한 디지털 비디오 스트리밍으로부터 꾸준한 동작을 관찰한다). 이러한 수준의 제어는 케이블 TV 네트워크 전파 중계소와 케이블 TV 가입자의 가정까지의 사이, 또는 DSL 센트럴 오피스에서 DSL 가입자 가정으로, 또는 서버 또는 사용자로부터의 상업 오피스 LAN 환경 내에서 달성될 수 있다. 또한, 보증된 대역폭 및 레이턴시를 갖는 사업들 사이의 특별히-등급된 점-대-점(point-to-point) 개인 연결들을 얻는 것은 가능하다. 그러나 일반 인터넷과 연결된 서버 센터 내에 게임들을 호스팅하고 그 다음 브로드밴드 연결을 통해서 사용자에게 압축된 비디오를 스트림하는 게임 또는 어플리케이션 시스템에서, 레이턴시는 많은 요소에 의해 발생하고, 종래 시스템의 배치(deployment)에서 심각한 제한을 야기한다.

[0059] 전형적인 브로드밴드 연결 홈에서, 사용자는 브로드밴드 서비스를 위한 DSL 또는 케이블 모뎀을 가질 수 있다.

이러한 브로드밴드 서비스는 보통 사용자의 가정과 일반 인터넷 사이에서 25ms의 왕복 레이턴시(그리고 두배 이상)만큼 발생한다. 더욱이, 인터넷을 통해 라우팅 데이터로부터 서버 센터까지 발생한 왕복 레이턴시들이 존재한다. 인터넷을 통한 레이턴시는, 데이터들이 주어지고 그것이 라우팅되는 것처럼 발생하는 지연들에 기초하여 변화한다. 라우팅 지연(routing delays)에 추가하여, 왕복 레이턴시는 또한 인터넷의 대부분을 서로 연결하는 광섬유를 통해 전송되는 빛의 속도로 인해 발생한다. 예컨대, 각 1000마일당, 약 22ms는 광섬유 및 다른 오버헤드(overhead)를 통하여 빛의 속도에 기인한 왕복 레이턴시로 발생된다.

[0060] 추가적인 레이턴시는 인터넷을 통해 스트리밍되는 데이터의 전송률(data rate)에 기인하여 발생할 수 있다. 예컨대, 사용자가 "6Mbps DSL 서비스"로 팔리는 DSL 서비스를 갖고 있다면, 실제로 사용자는 아마도 최대 다운스트림의 5Mbps보다 작은 것을 갖을 것이고, 아마 DSLAM(Disital Subscriber Line Access Multiplexer)에서 피크 로드 타임 동안에 혼잡과 같은 여러 요소들로 인해 주기적으로 낮아지는 연결을 볼 수 있을 것이다. 비슷한 이슈는 케이블 모뎀 시스템 네트워크에서 그 밖의 장소 또는, 인접한 것을 통해 루프된(looped) 로컬 공유된 동축 케이블(local shared coaxial cable)에 혼잡(congestion)이 있다면, 그것보다 훨씬 작은 "6Mbps 케이블 모뎀 서비스"로서 판매되는 연결을 사용하는 케이블 모뎀의 전송률을 감소할 수 있다. 만약 4Mbps의 안정된 속도에 데이터 패킷(data packets)이 이러한 연결을 통해 서버 센터로부터 UDP(User Datagram Protocol) 포맷 내에서 한쪽 방향으로 스트리밍된다면, 모두 잘 동작될 때, 데이터 패킷은 추가적인 레이턴시를 발생시키지 않고 통과할 것이나, 혼잡(또는 다른 장애물들)이 있다면, 단 3.5Mbps만이 사용자에게 데이터를 스트리밍하는데 사용될 수 있고, 따라서 전형적인 상황에 있어서, 패킷들은 떨어질 것이고, 잃어버린 데이터를 야기하며, 또는 패킷들은 그들이 거기에 추가적인 레이턴시를 발생시키면서 전송될 수 있을 때까지, 혼잡 포인트에서 큐잉(queuing)할 것이다. 혼잡의 다른 포인트들은 지연된 패킷들을 잡기 위해 다른 큐잉 용량을 갖고, 그래서 어떤 경우에 있어서, 혼잡을 통해 만들 수 없는 패킷들은 즉시 떨어질 것이다. 다른 경우에 있어서, 수 메가비트의 데이터가 큐잉 업(up)되고 결국 전송될 것이다. 그러나, 거의 모든 경우에 있어서, 혼잡 포인트들에서의 큐들은 용량 제한을 갖고, 이러한 제한들이 초과되는 경우, 큐들은 오버 플로우될 수 있고 패킷들은 떨어질 것이다. 따라서, 추가적인 레이턴시의 발생(또는 더 불량한 패킷의 손실)을 피하기 위해, 게임 또는 어플리케이션 서버로부터 사용자에게 전송률 용량의 초과를 피하는 것이 필요하다.

[0061] 또한 레이턴시는 서버 내에서 비디오를 압축하고 클라이언트 장치 내에서 비디오를 신장하기 위해 필요한 시간에 의해 발생된다. 레이턴시는 서버 상에서 진행중인 비디오 게임이 디스플레이되기 위해 다음 프레임을 계산하는 동안 더 발생된다. 현재 이용가능한 비디오 압축 알고리즘들은 높은 전송률 또는 높은 레이턴시 중 하나를 겪는다. 예컨대, 움직임 JPEG은 낮은-레이턴시에 의해 특성화되는 인트라프레임-전용 손실이 많은 압축 알고리즘이다. 비디오의 각 프레임은 비디오의 서로 다른 프레임과 독립적으로 압축된다. 클라이언트 장치가 압축된 움직임 JPEG 비디오의 프레임을 수신할 때, 그것은 즉시 프레임을 신장할 것이고 그것을 디스플레이할 수 있으며, 매우 낮은-레이턴시를 초래한다. 그러나 각 프레임이 각각 압축되기 때문에, 알고리즘은 연속적인 프레임들 사이의 유사성을 활용하는 것이 불가능하고, 결과적으로 인트라프레임(intraframe)-전용 비디오 압축 알고리즘은 매우 높은 전송률을 겪는다. 예컨대, 60fps(초당 프레임)에서 640x480 움직임 JPEG 비디오는 40Mbps 이상의 데이터를 요구할 수 있다. 이러한 낮은 해상도 비디오 윈도우를 위한 이러한 높은 전송률은 많은 방송 어플리케이션에서 엄청 비쌀 수 있다(그리고 확실히 대부분의 소비자 인터넷 기반의 어플리케이션을 위해). 더욱이, 각 프레임이 독립적으로 압축되기 때문에, 손실이 많은 압축으로부터 야기될 수 있는 프레임 내의 아티팩트들은 연속적인 프레임들 내에서 다른 장소들 안에 나타나는 것처럼 보인다. 이는 비디오가 신장될 때, 움직이는 시각적 아티팩트(visual artifacts)로서 보는 사람에게 나타나는 결과를 초래한다.

[0062] 그들이 종래 기술 구성에서 사용된 것처럼, Microsoft Corporation의 MPEG2, H.264 또는 VC9과 같은 다른 압축 알고리즘들은 높은 압축 비율을 달성할 수 있으나 높은 레이턴시의 비용에서 이루어진다. 이러한 알고리즘들은 프레임의 인트라프레임-전용 압축을 수행한다. 이러한 프레임은 키 프레임(key frame)으로 알려져 있다(전형적으로"I" 프레임으로 불리는). 따라서, 이러한 알고리즘들은 전형적으로 I 프레임과 이전 프레임들 및 연속적인 프레임을 비교한다. 이전 프레임들과 연속적인 프레임들을 독립적으로 압축하는 것보다, 알고리즘은 I 프레임에서 이전 및 연속적인 프레임들까지 이미지 내에서 변화하는 것을 결정하는 것이 더 좋고, 따라서 I 프레임을 앞서는 변화들을 위한 "B" 프레임, 및 I 프레임에 뒤따르는 변화들을 위한 "P" 프레임으로 불리는 이러한 변화들을 저장한다. 이는 인트라프레임-전용 압축보다 훨씬 더 낮은 전송률을 초래한다. 그러나, 그것은 전형적으로 더 높은 레이턴시의 비용으로 나타난다. I 프레임은 전형적으로 B 또는 P 프레임보다 훨씬 더 크고(종종 10배 더 크다), 결과적으로, 주어진 전송률에서 전송하는데 비례하여 더 오래 걸린다. 예컨대, I 프레임들이 B 및 P 프레임들의 크기의 10배인 상황이라면, 매 단일 I 프레임마다 29 B 프레임 + 30 P 프레임 = 59 인터프레임(interframes)이 있고, 또는 각 "프레임의 그룹(Group of Frames; GOP)"마다 총 60 프레임들이 있다. 따라서,

60 fps에서, 매 초당 하나의 60-프레임 GOP가 존재한다.

[0063] 전송 채널이 2Mbps의 최대 전송률을 갖는 것을 상상해보라. 채널 내에서 가장 높은 품질의 데이터 스트림을 달성하기 위해, 압축 알고리즘은 2Mbps 데이터 스트림을 생산할 것이고, 상기 비율이 주어진다면, 이는 2메가비트(Mb)/(59+10)= 30,394 비트/인트라프레임과 303,935 비트/I 프레임을 초래할 것이다. 압축된 비디오 스트림이 신장 알고리즘에 의해 수신될 때, 비디오가 안정적으로 실행되도록 하기 위해, 각 프레임은 정기적인 간격(예컨대, 60 fps)에서 디스플레이되고 신장될 필요가 있다. 이러한 결과를 달성하기 위해, 어떤 프레임이 전송 레이턴시의 대상이 되면, 모든 프레임들은 적어도 그 레이턴시에 의해 지연되는 것을 필요로 하고, 따라서, 가장 나쁜-경우 프레임 레이턴시는 모든 비디오 프레임동안 레이턴시를 정의할 것이다. I 프레임들은 가장 크기 때문에, 가장 긴 전송 레이턴시를 발생시키고, 전체 I 프레임은 I 프레임이 신장되고 디스플레이되기 전에 수신될 것이다(또는 I 프레임에 의존적인 인터프레임). 채널 전송률이 2Mbps로 주어진다면, I 프레임을 전송하는 데는 303,935/2Mb = 145ms이 걸릴 것이다.

[0064] 전송 채널의 대역폭의 큰 비율을 사용하는 상기 설명한 것처럼 인터프레임(interframe) 비디오 압축 시스템은, 프레임의 평균 크기에 비례하는 I 프레임의 큰 크기로 인하여 긴 레이턴시의 대상이 될 것이다. 또는, 그것을 다른 방법으로 해결하기 위해, 종래 기술 인터프레임 압축 알고리즘들이 인트라프레임-전용 압축 알고리즘들보다 더 낮은 평균 프레임당 전송률을 달성하는 반면(예컨대, 2Mbps 대 40Mbps), 여전히 그들은 큰 I 프레임 때문에 여전히 높은 피크 프레임당 전송률을 겪는다(예컨대, 303,935 * 60 = 18.2Mbps). 명심하건대, 상기한 분석을 통해 P 및 B 프레임들은 모두 I 프레임들 보다 훨씬 더 작다는 것을 가정하라. 이것이 일반적으로 사실인 반면, 장면 변화, 많은 움직임, 또는 이전 프레임과 연관성이 없는 높은 이미지 복잡도를 갖는 프레임들에 대해서는 사실이 아니다. 이러한 상황에 있어서, P 또는 B 프레임들은 I 프레임만큼 커질 수 있다(P 또는 B 프레임이 I 프레임보다 더 크다면, 정교한 압축 알고리즘은 전형적으로 I 프레임을 "force"할 것이고 P 또는 B 프레임을 I 프레임으로 바꿀 것이다). 그래서, I 프레임-크기의 전송률의 피크는 디지털 비디오 스트림 내에서 어떠한 순간에도 발생할 수 있다. 따라서, 압축된 비디오로, 평균 비디오 전송률이 전송 채널들의 전송률 용량에 접근할 때(빈번한 경우, 비디오를 위한 높은 전송률 요구가 주어진다면), I 프레임 또는 큰 P 또는 B 프레임으로부터의 높은 피크 전송률은 높은 프레임 레이턴시를 야기한다.

[0065] 당연히, 상기 논의는 단지 GOP 내의 큰 B,P 또는 I 프레임들에 의해 생성된 압축 알고리즘 레이턴시를 특징짓는다. B 프레임들이 사용된다면, 레이턴시는 훨씬 더 높은 것이다. 이유는 B 프레임이 디스플레이될 수 있기 전에, B 프레임 및 I 프레임 후의 B 프레임들의 전부가 수신되기 때문이다. 따라서, 각 I 프레임 전에 5개 B 프레임이 있는 BBBBBIPPPPPBBBBBIPPPPP와 같은 시퀀스의 한 그룹의 그림(Group Of Picture ; GOP) 내에서, 첫 번째 B 프레임들은 차후 발생한 B 프레임들 및 I 프레임이 수신될 때까지 비디오 신장기(decompressor)에 의해 디스플레이될 수 없다. 그래서 비디오가 60fps(즉, 16.67ms/frame)에서 스트리밍된다면, 5개 B 프레임들 및 I 프레임은 수신하는데 16.67 * 6 = 100ms가 걸리고, 채널 대역폭이 아무리 빠르다고 하더라도, 이는 단 5 B 프레임들만을 갖는다. 30 B 프레임들을 갖는 압축된 비디오 시퀀스들은 매우 일반적이다. 그리고, 2Mbps와 같은 낮은 채널 대역폭에서, I 프레임의 사이즈에 의해 야기된 레이턴시 영향은 B 프레임들이 도착하는 것을 기다리기 때문에 레이턴시 영향에 크게 더해진다. 따라서, 2Mbps 채널 상에서, 엄청난 수의 B 프레임을 갖는다면, 종래 비디오 압축 기술을 사용하여 500ms의 레이턴시를 초과하는 것은 매우 쉽다. B 프레임들이 사용되지 않는다면(주어진 품질 수준을 위한 낮은 압축 비율의 비용에서), 상기한 바와 같이, B 프레임 레이턴시는 발생하지 않고, 피크 프레임 사이즈들에 의해 야기된 레이턴시만이 여전히 발생한다.

[0066] 그 문제가 많은 비디오 게임들의 중요한 특성으로 인해 악화된다. 상기한 GOP 구조를 사용하는 비디오 압축 알고리즘들은 수동적인 시청을 위해 의도된 라이브 동작 또는 그림 재료의 사용을 위해 매우 최적화되어왔다. 전형적으로, 단순히 카메라 또는 장면이 너무 갑자기 움직인다면, 비디오 또는 영화 재료가 (a) 전형적으로 시청하는데 불편하고 (b) 그것이 보여진다면, 보통 시청자는 카메라가 갑자기 움직일때, 액션을 가깝게 따라가지 못할 수 있기 때문에, 카메라(실제 카메라, 또는 컴퓨터-제작 애니메이션의 경우의 가상 카메라인지)와 장면들이 상대적으로 안정적이다(예컨대, 한 아이가 생일 케익 상의 초를 불어서 꺼뜨릴 때 및 갑자기 케이크로 돌아와 다시 불어서 초를 꺼뜨릴 때 카메라가 떨어진다면, 시청자들은 전형적으로 아이와 케이크에 포커스를 맞추고 카메라가 갑자기 움직일 때, 잠깐의 정지는 무시하게 된다). 비디오 인터뷰, 또는 비디오 원격 회의의 경우에 있어서, 카메라는 고정된 위치에 있을 수 있고 전혀 움직임이 없으며, 매우 작은 데이터 피크들을 초래한다. 그러나 3D 높은 액션 비디오 게임들은 일정한 동작에 의해 특징화된다(예컨대, 전체 프레임이 경기하는 동안 빠른 동작 내에 있을 때, 3D 레이싱을 생각하거나, 또는 가상 카메라가 항상 갑작스럽게 움직일 때, 첫 번째 사람을 쏜 사람을 생각하라). 이러한 비디오 게임들은 사용자가 이러한 갑작스런 동작들 동안 일어나는 것을 명확하게

보여줄 필요가 있는 크고 빈번한 피크들을 갖는 프레임 시퀀스를 초래한다. 이러한 경우처럼, 압축 아티팩트는 3D 높은 액션 비디오 게임들에서 매우 좋지 않다. 따라서, 많은 비디오 게임들의 비디오 출력은, 그들의 특성에 의해, 매우 높고 빈번한 피크들을 갖는 압축된 비디오 스트림을 생산한다.

[0067] 빠른-동작 비디오 게임의 사용자가 높은 레이턴시, 및 주어진 상기 모든 레이턴시의 원인에 대해 작은 인내력을 갖는다면, 인터넷에 비디오를 스트림하는 서버-호스트된 비디오 게임에 제한이 있다. 더욱이, 상호 대화의 높은 정도를 요구하는 어플리케이션들의 사용자들은 어플리케이션들이 일반 인터넷 및 스트림 비디오 상에 호스팅된다면 유사한 제한들을 겪는다. 클라이언트 장치로부터 서버까지의 루트 및 거리가 레이턴시를 최소화되도록 제어되기 위해서 그리고 피크들이 레이턴시 없이 수용될 수 있도록 하기 위해, 이러한 서비스들은 호스팅 서버가 전파 중계소 내에(케이블 브로드밴드의 경우) 또는 센트럴 오피스(central office)(DSL의 경우), 또는 상업 세팅 안의 LAN(또는 특수-등급된(specially-graded) 개인 연결 상의) 내에 직접 구성된 네트워크 구성을 요구한다. 적합한 대역폭을 갖는 LAN들(전형적으로 100Mbps-1Gbps의 속도를 갖는) 및 임대 전선은 전형적으로 피크 대역폭 요구사항을 지원할 수 있다(예컨대, 18Mbps 피크 대역폭은 100Mbps LAN 용량의 작은 부분이다).

[0068] 또한 피크 대역폭 요구사항들은 특별한 수용용량이 만들어진다면 가정용 브로드밴드 기반구조(infrastructure)에 의해 수용될 수 있다. 예컨대, 케이블 TV 시스템 상에서, 디지털 비디오 트래픽(digital video traffic)은 I 프레임들만큼 큰 피크들을 다룰 수 있는 전용 대역폭을 가질 수 있다. 그리고, DSL 시스템 상에서, 더 높은 속도의 DSL 모뎀이 높은 피크들을 허용하면서 공급될 수 있고, 또는 특수-등급된 연결이 더 높은 전송률을 다룰 수 있도록 공급될 수 있다. 그러나, 일반 인터넷에 부속된 전통적인 케이블 모뎀 및 DSL 기반구조는 압축된 비디오를 위한 피크 대역폭 요구사항들에 대해 훨씬 적은 오차 허용도를 갖는다. 그래서, 클라이언트 장치로부터 먼 거리에 있는 서버 센터들 내의 비디오 게임 또는 어플리케이션을 호스팅하고, 전통적인 가정용 브로드밴드 연결들을 통해 인터넷에 걸쳐 압축된 비디오 출력을 스트림하는 온라인 서비스는 특별히 매우 낮은-레이턴시를 요구하는 게임들과 어플리케이션(예컨대, 첫번째 사람 총잡이(shooters)와 다른 멀티-사용자, 인터랙티브 액션 게임들, 또는 빠른 반응 속도를 요구하는 어플리케이션)과 관련하여 중요한 레이턴시 및 피크 대역폭 제한들을 겪는다.

발명의 내용

해결하려는 과제

[0069] 본 발명은 상기한 점을 감안하여 이루어진 것으로, 비디오를 압축하기 위한 타일 기반 시스템 및 방법을 제공하는 것을 목적으로 한다.

과제의 해결 수단

[0070] 본 발명에 따른 이러한 목적은 각각의 이미지 시퀀스를 다수의 타일로 로컬적으로 분할하는 단계; 제1압축 포맷을 이용하여 이미지 시퀀스의 제1이미지 내의 제1의 정의된 위치의 타일 중 하나를 인코딩하고, 제2압축 포맷을 이용하여 상기 제1이미지 내의 나머지 타일을 인코딩하는 단계; 및 상기 제1압축 포맷을 이용하여 이미지 시퀀스의 제2이미지 내의 제2의 정의된 위치의 타일 중 하나를 인코딩하고, 제2압축 포맷을 이용하여 상기 제2이미지 내의 나머지 타일을 인코딩하는 단계를 포함하고, 상기 각각의 타일은 상기 각각의 이미지 시퀀스 내에 정의된 위치를 가지며, 연속적인 이미지들 사이에 상기 정의된 위치를 유지하고, 상기 제2압축 포맷은 상기 제1압축 포맷 및/또는 제2압축 포맷에 따라 인코딩된 미리 인코딩된 타일에 따른 것을 특징으로 하는 컴퓨터-실행 방법에 의해 달성된다.

발명의 효과

[0071] 상기와 같은 비디오를 압축하기 위한 타일 기반 시스템 및 방법을 제공할 수 있다.

도면의 간단한 설명

[0072] 도 1은 종래 기술의 비디오 게이밍 프로그램 아키텍처를 나타낸 도면,

도 2a와 2b는 일 실시예에 따른 하이 레벨 시스템 아키텍처를 나타낸 도면,

도 3은 클라이언트 및 서버 사이의 통신을 위한 실제적, 속도, 요구되는 전송률을 나타낸 도면,

도 4a는 일 실시예에 따른 호스팅 서비스 및 채택된 클라이언트를 나타낸 도면,

도 4b는 클라이언트 및 호스팅 서비스 사이의 통신과 관련된 전형적인 레이턴시를 나타낸 도면,

도 4c는 일 실시예에 따른 클라이언트 장치를 나타낸 도면,

도 4d는 다른 실시예에 따른 클라이언트 장치를 나타낸 도면,

도 4e는 도 4c에 클라이언트 서비스의 블럭도의 예를 나타낸 도면,

도 4f는 도 4d에 클라이언트 서비스의 블럭도의 예를 나타낸 도면,

도 5는 일 실시예에 따른 채택될 수 있는 비디오 압축의 형태의 예를 나타낸 도면,

도 6a는 다른 실시예에서 채택될 수 있는 비디오 압축의 형태의 예를 나타낸 도면,

도 6b는 낮은 복잡성 전송, 낮은 동작 비디오 시퀀스와 관련된 전송률에 피크를 나타낸 도면,

도 6c는 높은 복잡성 전송, 높은 동작 비디오 시퀀스와 관련된 전송률에 피크를 나타낸 도면,

도 7a,7b는 일 실시예에서 채택된 비디오 압축 기술의 예를 나타낸 도면,

도 8은 일 실시예에서 채택된 비디오 압축 기술의 추가적인 예를 나타낸 도면,

도 9a ~ 9c는 완화된 전송률 피크를 위한 일 실시예에서 채택된 기술의 예를 나타낸 도면,

도 10a,10b는 패킷내에 이미지 타일을 효율적으로 포장(pack)하기 위한 일 실시예를 나타낸 도면.

도 11a ~ 11d는 순방향 에러 보정 기술을 채택하는 일 실시예를 나타낸 도면,

도 12는 압축을 위한 멀티 코어 프로세싱 유닛을 사용하는 일 실시예를 나타낸 도면,

도 13a,13b는 다양한 실시예에 따라 호스팅 서비스와 사이에 지리적인 포지셔닝 및 통신을 나타낸 도면,

도 14는 클라이언트 및 호스팅 서비스 사이 통신과 관련된 전형적인 레이턴시를 나타낸 도면,

도 15는 호스팅 서비스 서버 센터 아키텍처의 예를 나타낸 도면,

도 16은 다수의 라이브 비디오 윈도우를 포함하는 사용자 인터페이스의 일 실시예의 스크린 샷의 예를 나타낸 도면,

도 17은 특정 비디오 윈도우의 선택에 따라 도 16의 사용자 인터페이스를 나타낸 도면,

도 18은 풀 스크린 크기의 특정 비디오 윈도우의 확대에 따른 도 17의 사용자 인터페이스를 나타낸 도면,

도 19는 멀티플레이어 게임의 스크린에 오버레이된 공동의 사용자 비디오 데이터의 예를 나타낸 도면,

도 20은 호스팅 서비스에 게임 플레이어를 위한 유저 페이지의 예를 나타낸 도면,

도 21은 3D 인터렉티브 광고의 예를 나타낸 도면,

도 22는 라이브 퍼포먼스의 표면 캡쳐로부터 텍스쳐 표면을 갖는 포토리얼 이미지를 생산하는 단계의 시퀀스의 예를 나타낸 도면,

도 23은 선형 미디어 컨텐츠의 선택을 허용하는 유저 인터페이스 페이지의 예를 나타낸 도면,

도 24는 연결 속도에 대한 웹 페이지가 라이브 되기 전 흐르는 시간을 나타내는 그래프,

도 25a~b는 클라이언트 장치에서 호스팅 서비스까지의 피드백 채널을 채용하는 본 발명의 실시예를 나타낸 도면,

도 26a~b는 성공적으로 수신되도록 마지막 알려진 타일/프레임에 기초하여 타일/프레임을 인코드하는 실시예를 나타낸 도면,

도 27a~b는 제1호스팅 서비스 또는 서버에서 제2호스팅 서비스 또는 서버로 게임 또는 어플리케이션의 상태가 포트되는 실시예를 나타낸 도면,

도 28은 차 데이터를 이용하여 게임 또는 어플리케이션의 상태가 포트되는 일 실시예를 나타낸 도면,

도 29는 클라이언트 장치에 임시 디코더를 채택하는 본 발명의 일 실시예를 나타낸 도면,

도 30은 본 발명의 일 실시예에 따라 "I 타일"이 어떻게 "R 프레임"에 걸쳐 분포되는지를 나타낸 도면,

도 31a~h는 라이브 스트림 및/또는 하나 또는 그 이상의 HQ 스트림을 생성하는 본 발명의 실시예를 나타낸 도면이다.

발명을 실시하기 위한 구체적인 내용

[0073] 다음의 상세한 설명에서 구체적인 세부사항은 본 발명 개시에 이해를 제공하기 위하여 장치 종류, 시스템 구성, 통신 방법과 같이 설정한다. 그러나, 관련 기술 분야의 통상의 기술을 가진 자가 이러한 구체적인 세부사항은 설명된 실시예를 실행하기 위하여 필요로 하지 않을지 모른다는 것을 인식할 것이다.

[0074] 도 2a와 2b는 가입 서비스 하에서 인터넷(206)(그 밖의 공공 또는 개인 네트워크)을 통해서 비디오 게임 및 소프트웨어 어플리케이션이 호스팅 서비스(210)에 의해서 호스트되고 사용자 구내(user premises; 211)("사용자 구내"는 만일 모바일 장치를 사용한다면 외부를 포함하는 사용자가 위치하는 장소를 의미한다)에서 클라이언트 장치(205)에 의해서 접속된다. 클라이언트 장치(205)는 인터넷과 유선 또는 무선으로 연결되는 마이크로소프트 윈도우(Microsoft Windows)와 같은 일반 목적 컴퓨터 또는 Linux기반 PC 또는 Apple Inc사의 매킨토시 컴퓨터이거나, 내부 또는 외부 디스플레이 장치(222)와 연결되거나 비디오 및 오디오를 모니터 또는 TV세트 (222)로 출력하는 셋톱 박스(인터넷과 유선 또는 무선으로 연결되고)와 같은 클라이언트 장치에 전용될 수 있으며, 인터넷과 유선 또는 무선으로 연결되는 아마 모바일 장치일 수도 있다.

[0075] 이러한 장치중 어떤 것은 그들 자신의 사용자 입력 장치(예컨대, 키보드, 버튼, 터치스크린, 트랙 패드, 또는 관성 감지 막대, 비디오 캡쳐 카메라 및/또는 동작 추적 카메라 등)를 가질 수 있거나, 그들은 유선 또는 무선으로 연결되는 내부 입력 장치(221)(예컨대, 키보드, 마우스, 게임 컨트롤러, 관성 감지 막대, 비디오 캡쳐 카메라 및/또는 동작 추적 카메라 등)를 사용한다. 보다 상세하게 아래에서 설명하는 것처럼, 호스팅 서비스(210)는 높은 전력의 CPU/GPU 프로세싱 성능을 포함하는 다양한 레벨의 성능의 서버를 포함한다. 호스팅 서버(210)에서 어플리케이션의 사용 또는 게임의 플레이 동안에, 홈 또는 오피스 클라이언트 장치(205)는 사용자로부터 키보드 및/또는 컨트롤러 입력을 수신하고, 그 후 그것은 인터넷(206)을 통해서 게임 또는 어플리케이션 소프트웨어(예컨대, 만일 사용자가 화면의 캐릭터를 오른쪽으로 움직이도록 지시하는 버튼을 누르면, 그 후 게임 프로그램은 오른쪽으로 캐릭터가 움직이는 것을 보여주는 비디오 이미지의 시퀀스를 만든다)를 위해 비디오 출력(비디오 이미지 시퀀스)의 연속적인 프레임을 생성하고 응답으로 게이밍 프로그램 코드를 실행하는 호스팅 서비스(210)로 컨트롤러 입력을 전송한다. 이러한 비디오 이미지의 시퀀스는 낮은-레이턴시 비디오 압축기를 사용하여 압축되고, 호스팅 서비스(210)은 낮은-레이턴시 비디오 스트림을 인터넷(206)을 통해서 전송한다. 홈 또는 오피스 클라이언트 장치는 압축된 비디오 스트림을 복호하고 모니터 또는 TV에 압축이 풀린 비디오 이미지를 랜더링(render)한다. 결과적으로, 클라이언트 장치(205)의 컴퓨팅 그리고 그래픽 하드웨어 요구는 현저하게 생략된다. 클라이언트(205)는 인터넷(206)으로 키보드/컨트롤러 입력을 전송하고, 인터넷(206)으로부터 수신된 압축된 비디오 스트림의 압축을 풀며, 복호화하기 위한 처리 능력(power)을 갖는 것이 오직 필요하고, 이는 사실상 개인용 컴퓨터가 그것의 CPU에서 소프트웨어를 구동할 능력이 있다(예컨대, 대략 2GHz로 동작하는 인텔의 Core Duo CPU는 윈도우 미디어 VC9과 H.264와 같은 압축기를 사용하여 엔코드된 720p HDTV의 압축을 풀 수 있다). 그리고, 어떤 클라이언트 장치의 경우에, 전용 칩은 또한 현대 PC를 위해 요구되는 것처럼 일반 목적 PC보다 낮은 가격 및 적은 전력 소모로 실시간으로 그러한 표준을 위한 비디오 압축을 푸는 것을 수행할 수 있다. 현저하게 컨트롤러 입력을 전송하고 비디오 압축을 푸는 것의 기능을 수행하기 위해서, 홈 클라이언트 장치(205)는 도 1에 도시한 선행 기술 비디오 게임 시스템과 같은 광학적 드라이브, 하드 드라이브, 또는 어떤 특별한 그래픽 처리 유닛(GPU)을 요구하지 않는다.

[0076] 게임 및 어플리케이션 소프트웨어가 보다 복잡하고 보다 포토리얼리즘이 되어가면서, 그들은 높은 성능 CPU, GPU, 더 많은 RAM, 더 크고 빠른 디스크 드라이브를 요구할 것이고, 호스팅 서비스(210)에서 컴퓨팅 파워는 연속적으로 업그레이드될 것이지만, 최종 사용자는 그것의 처리 요구가 기존 비디오 압축을 푸는 알고리즘으로 디스플레이 해상도 및 프레임 비율을 위해 고정되어 존재할 것이므로 홈 또는 오피스 클라이언트 플랫폼(205)을 업데이트하도록 요구되지 않을 것이다. 따라서, 하드웨어 제한 및 호환성 이슈는 오늘날 도 2a, 2b에서 도시된 시스템에서는 더 이상 존재하지 않는다.

[0077] 더욱이, 게임 및 어플리케이션 소프트웨어가 오직 호스팅 서비스(210)의 서버에서 실행되기 때문에, 사용자의 홈이나 오피스(여기서 "오피스"는 만약 어떤 자격을 갖춘 어떤 비거주의 세팅을 포함할 것이고, 예컨대 스쿨룸을 포함함)에서 결코 게임 또는 어플리케이션 소프트웨어(광학 미디어의 형태, 또는 다운로드된 소프트웨어와

같은 것 중 어느 것)의 복사가 없다. 이는 게임 또는 어플리케이션이 불법 복제되는 공산을 현저하게 완화시키고, 그뿐만 아니라 불법 복제된 게임 또는 어플리케이션에 의해서 사용될 수 있는 있는 가치있는 데이터베이스의 확률을 완화시킨다. 정말, 만약 전문 서버(specialized servers)가 홈 오피스에서 사용을 위해서는 별로 실효성 없는 게임 또는 어플리케이션 소프트웨어를 플레이하기 위해 요구된다면, 게임 또는 어플리케이션 소프트웨어의 불법 복제가 얻어졌음에도 불구하고, 그것은 홈 또는 오피스에서 동작할 수 없을 것이다.

[0078] 일 실시예에서, 호스팅 서비스(210)는 비디오 게임을 개발하여 호스팅 서비스(210)에 실행될 수 있는 게임을 디자인하는 게임 또는 어플리케이션 소프트웨어 개발자(이는 일반적으로 소프트웨어 개발 회사, 게임 또는 무비 스튜디오 또는 게임 또는 어플리케이션 소프트웨어 출판인으로 언급된다; 220)를 위해 소프트웨어 개발 툴(tools)을 제공한다. 그러한 툴은 개발자가 표준 PC 또는 게임 콘솔에서 통상적으로 이용할 수 없는 호스팅 서비스의 특징을 활용하는 것을 허용하게 한다(예컨대, 복잡한 지오메트리("지오메트리(geometry))" 그렇지않으면 여기서는 3D 데이터세트로 정의되는 폴리건, 텍스쳐, 리깅(rigging), 라이트닝(lighting), 동작 및 그 밖의 요소 및 파라미터로 언급된다)의 매우 큰 데이터베이스로 매우 빠른 접근).

[0079] 다른 비즈니스 모델은 이러한 아키텍쳐 하에서 가능하다. 하나의 모델 아래, 호스팅 서비스(210)는 도 2a에 도시된 것처럼, 최종 사용자로부터 가입 요금(subscription fee)을 수금하고 개발자(220)에게 로얄티를 지급한다. 도 2b에 도시된 택일적인 실행에서, 개발자(220)는 직접 사용자로부터 가입 요금을 수금하고 게임 또는 어플리케이션 컨텐츠 호스팅을 위해 호스팅 서비스(210)에 요금을 지불한다. 이러한 근본적인 이론은 온라인 게이밍 또는 어플리케이션 호스팅을 제공하기 위한 특정 비즈니스 모델로 제한되지는 않는다.

[0080] **압축된 비디오 특징(Compressed Video Characterisitics)**

[0081] 앞서 논의한 대로, 비디오 게임 서비스 또는 어플리케이션 소프트웨어의 서비스 온라인 제공으로 인한 하나의 현저한 문제는 레이턴시(latency)이다. 70 ~ 80 ms(사용자에 의해서 작동되는 입력 장치의 포인트에서 디스플레이 장치에 디스플레이되는 응답 포인트까지)의 레이턴시는 빠른 응답 시간을 요구하는 게임 및 어플리케이션을 위한 상한(upper limit)이다. 그러나, 이는 많은 실제적 물리적 제약으로 인하여 도 2a 및 2b에 도시된 아키텍쳐와 관련하여 성취하는 것은 굉장히 어렵다.

[0082] 도 3에 나타낸 바와 같이, 사용자가 인터넷 서비스에 가입한 때, 연결은 전형적으로 사용자의 홈 또는 오피스로 명목상 최대 전송률(301)로 평가된다. 제공자의 정책 및 전송 장비 능력에 따라, 최대 전송률은 엄격하게 더 빠르게 또는 느리게 강요되지만, 전형적으로 실제 이용가능한 전송률은 많은 다른 이유 중 하나에 의해서 더 느리다. 예컨대, DSL 센트럴 오피스 또는 로컬 케이블 모뎀 루프(loop)에는 너무 많은 네트워크 트래픽이 있거나, 떨어진 패킷 원인이 되는 케이블에 노이즈가 있거나, 제공자가 사용자마다 매달 최대 비트를 정할 것이다. 현재, 케이블 및 DSL 서비스를 위한 최대 다운스트림 전송률은 전형적으로 수백 킬로비트/초(Kbps)에서 30 Mbps 범위에 있다. 셀룰러 서비스는 전형적으로 수백 Kbps의 다운스트림 데이터로 제한된다. 그러나, 브로드밴드 서비스의 속도 및 브로드밴드 서비스에 가입한 다수의 사용자는 극적으로 시간 경과에 따라 증가될 것이다. 현재, 어떤 분석가는 미국 브로드밴드 가입자의 33%는 2Mbps 이상의 다운스트림 속도를 갖는 것으로 추정한다. 예컨대, 어떤 분석가는 2010년까지 미국 브로드밴드 가입자의 85% 이상은 2Mbps 이상의 전송률을 갖게 될 것으로 예상한다.

[0083] 도 3에 나타낸 바와 같이, 실제 이용가능한 최대 전송률(302)은 시간에 걸쳐 변동을 거듭할 것이다. 따라서, 낮은-레이턴시에서, 그것과 관련된 온라인 게임 또는 어플리케이션 소프트웨어는 특정 비디오 스트림에서 실제 이용가능한 전송률을 예상하기 어렵다. 만약 전송률(303)이 실제 이용가능한 최대 전송률(302)보다 늘어난 동작 및 어떤 다수의 복잡한 장면을 위해 주어진 해상도(예. 60fps에서 640 × 480)에서 주어진 다수의 초당 프레임(frames-per-second; fps)에서 주어진 수준의 품질을 유지하도록 요구된다면, 사용자의 비디오 스크린에 왜곡된/손실된 이미지 및 손실된 데이터를 초래하게 된다. 다른 서비스는 일시적으로 부가 패킷을 버퍼링하고(예, 큐업(queue up)), - 이용가능한 전송률에서 클라이언트에게 패킷을 제공해서, 많은 비디오 게임 및 어플리케이션을 위해 받아들일 수 없는 결과인, 레이턴시의 증가를 초래한다. 마지막으로, 어떤 인터넷 서비스 제공자는 서비스 공격의 부정(네트워크 연결을 못하도록 해커들에 의해서 사용되는 잘 알려진 기술 사용자)처럼, 악의 있는 공격으로 전송률이 증가되고, 특정 시간 주기 동안 사용자의 인터넷 연결을 끊어지는 것을 볼 것이다. 따라서, 여기서 설명된 실시예는 비디오 게임을 위해 요구된 전송률이 최대 이용가능한 전송률을 초과하지 않는 것을 보장하기 위한 단계를 거친다.

[0084] **호스팅 서비스 아키텍쳐(Hosting Service Architecture)**

[0085] 도 4a는 일 실시예에 따른 호스팅 서비스(210)의 아키텍쳐를 도시하고 있다. 호스팅 서비스(210)는 단일 서버 센터에 위치되거나 다수의 서버 센터에 분포될 수 있다(그 밖의 사용자보다 어떤 서버 센터로 낮은-레이턴시 경로를 가진 사용자에게 낮은-레이턴시 연결을 제공하기 위해, 하나 이상의 서버 센터 실패의 경우에 리던던시(redundancy)를 제공하기 위해서 그리고 여러 사용자 중에서 균형잡힌 로드를 제공하기 위해서). 호스팅 서비스(210)는 결국 매우 많은 사용자 베이스를 제공하는 수십만 또는 수백만의 서버(402)를 포함할 것이다. 호스팅 서비스 컨트롤 시스템(401)은 호스팅 서비스(201)를 전체적으로 제어하고, 라우터, 서버, 비디오 압축 시스템, 빌링(billing) 그리고 어카운팅(accounting) 시스템 등을 총괄한다. 일 실시예에서, 호스팅 서비스 컨트롤 시스템(401)은 시스템 통계, 서버 정보, 사용자 정보를 위한 데이터베이스를 저장하기 위해 사용되는 RAID 배열과 묶인 분산 처리 리눅스 기반 시스템에서 실행된다. 앞서 말한 상세한 설명에서, 만약 그 밖의 특정 시스템의 결과로 보지 않는다면, 호스팅 서비스(210)에 의해 실행된 다양한 동작은, 호스팅 서비스 컨트롤 시스템(401)에 의해서 시작되고 제어된다.

[0086] 호스팅 서비스(210)는 인텔, IBM, 휴렛 패커드, 및 그 밖의 회사에서 현재 이용가능한 것과 같은 다수의 서버(402)를 포함한다. 택일적으로, 서버(402)는 구성요소의 관습적인 구성으로 조립될 수 있거나, 결국 통합될 수 있어서 전체 서버는 단일 칩으로서 실행된다. 비록 이 다이어그램은 예시의 목적으로 작은 수의 서버(402)를 도시할지라도, 실제 배치에서는 적게는 한 서버(402) 또는 많게는 수백만 서버 이상이 있을 것이다. 서버(402)는 동일한 방법(구성 파라미터의 어떤 것의 예로서, 동일한 CPU 타입 및 성능을 갖고; GPU가 있거나 없고 만약 GPU가 있다면 동일한 GPU 타입이고 성능을 갖고; 동일한 수의 CPU 및 GPU를 갖고; 동일한 수량의 형태/스피드의 RAM을 갖고; 그리고 동일한 RAM 구성을 갖는 것)으로 모두 설정되거나, 또는 서버(402)의 다양한 서브셋(subset)은 동일한 구성(예컨대, 서버의 25%는 특정 방식으로 구성될 수 있고, 50%는 다른 방식으로 구성되고, 그리고 25%는 이미 다른 방식으로 구성된다)을 갖거나, 또는 모든 서버(402)가 다를 것이다.

[0087] 일 실시예에서, 서버(402)는 디스크가 없고, 예컨대 그 자체가 로컬 대용량 스토리지(그것은 광학 또는 자기적 스토리지 또는 플래시 메모리와 같은 반도체 기반 스토리지 또는 그 밖의 비슷한 기능을 수행하는 수단인 대용량 스토리지)를 갖는 것보다, 각 서버는 공유된 대용량 스토리지에 빠른 백프레인(backplane) 또는 네트워크 연결을 통해서 접근한다. 일 실시예에서, 이러한 빠른 연결은 기가바이트 이더넷을 사용하여 실행되는 장치 사이 연결을 갖는 일련의 RAID(Redundant Arrays of Independent Disks; 405)와 연결된 스토리지 영역 네트워크(Storage Area Network; 403)이다. 종래 기술에서 알려진 것처럼, SAN(403)은 현재 게임 콘솔 및 PC에서 사용되는 RAM에서 이용가능한 대역폭에 근접하거나 잠재적으로 초과하는- 극단적으로 높은 대역폭을 초래하는, 많은 RAID 배열(405)과 함께 결합되어 사용될 것이다. 그리고 자기 미디어와 같은, 회전 미디어에 기초한 RAID 배열은, 종종 현저한 탐색 시간 접근 레이턴시를 갖는 반면에, 반도체 스토리지에 기초한 RAID 배열은 보다 낮은 접근 레이턴시로 실행될 수 있다. 다른 구성에서, 서버(402)의 일부 또는 전부는 로컬로 그들 자신의 대용량 스토리지의 일부 또는 전부를 제공한다. 예컨대, 서버(402)는 그것의 동작 시스템 및 낮은-레이턴시 로컬 플래시 기반의 스토리지에서 비디오 게임 또는 어플리케이션의 복사와 그것의 동작 시스템과 같은 자주 접근하는 정보를 저장할 것이지만, 적은 빈도수에 기초하여 기하학적 구조 또는 게임 상태 정보에 기초한 RAID 배열(405)에 접근하기 위해 SAN을 이용할 것이다.

[0088] 더욱이 일 실시예에서, 호스팅 서비스(210)은 아래에서 상세히 설명되는 낮은-레이턴시 비디오 압축 로직(404)을 채택한다. 비디오 압축 로직(404)은 소프트웨어, 하드웨어, 또는 어떤 그들의 조합(아래에서 설명되는 어떤 실시예)에서 실행될 것이다. 비디오 압축 로직(404)은 시각적 재료뿐만 아니라 오디오 압축을 위한 로직을 포함한다.

[0089] 동작중에, 키보드, 마우스, 게임 컨트롤러 또는 그 밖의 입력장치(421)를 통해서 사용자 구내(211)에서 비디오 게임을 플레이하거나 어플리케이션을 사용하는 동안에, 클라이언트(415)에 있는 제어 신호 로직(413)은 사용자에 의해서 작동된 버튼 누름(그리고 사용자 입력의 그 밖의 형태)을 나타내는 제어 신호(406a, 406b)(전형적으로 UDP 패킷의 형태로)를 호스팅 서비스(210)로 전송한다. 기존 사용자로부터 제어 신호는 적절한 서버(또는 서버들, 만약 다수 서버가 사용자의 입력 장치에 응답하면)로 보내진다. 도 4a에서 도시된 것처럼, 제어 신호(406a)는 SAN을 통해서 서버(402)로 보내질 것이다. 택일적으로 또는 추가적으로, 제어 신호(406b)는 호스팅 서비스 네트워크(예컨대, 이더넷 기반 로컬 영역 네트워크)를 거쳐서 서버(402)로 직접 보내질 것이다. 그들이 어떻게 전송되는지에 상관없이, 서버 또는 서버들은 제어 신호(406a, 406b)에 응답하는 게임 또는 어플리케이션

소프트웨어를 실행한다. 비록 도 4a에 도시하지 않았지만, 방화벽(firewall) 및/또는 게이트웨이와 같은 다양한 네트워킹 요소는 호스팅 서비스(210)의 가장자리 및/또는 인터넷(410) 및 홈 또는 오피스 클라이언트(415) 사이의 사용자 구내(211)의 가장자리에서 들어오거나 나가는 트랙픽 처리를 할 것이다. 실행되는 게임 또는 어플리케이션 소프트웨어의 그래픽 그리고 오디오 출력- 예컨대, 새로운 연속적인 비디오 이미지는 여기서 설명된 것처럼 낮은-레이턴시 비디오 압축 기술에 따른 비디오 이미지 시퀀스를 압축하는 낮은-레이턴시 비디오 압축 로직(404)으로 제공되고 압축 비디오 스트림을 전송하며, 전형적으로 압축된 또는 비압축된 오디오를, 인터넷(410)을 거쳐(또는, 이하 설명되는 것처럼, 일반 인터넷을 통과하는 최적화된 높은 속도 네트워크 서비스를 거쳐) 클라이언트(415)로 되돌아가게 한다. 그 다음 클라이언트(415)에 낮은-레이턴시 비디오 신장(decompression) 로직(412)은 비디오 및 오디오 스트림을 신장하고 신장된 비디오 스트림을 랜더링하고, 전형적으로 신장된 오디오 시스템을 디스플레이 장치(422)에서 재생한다. 택일적으로, 오디오는 디스플레이 장치(422)와 분리 또는 분리되지 않는 스피커에서 재생될 수 있다, 그것에 주목하면, 입력 장치(421) 및 디스플레이 장치(422)가 도 2a 및 도 2b에서 독립적으로 도시되었음에도, 그들은 휴대용 컴퓨터 또는 모바일 장치와 같은 클라이언트 장치내에 통합될 것이다.

[0090] 홈 또는 오피스 클라이언트(415)(도 2a 및 도 2b에서 홈 또는 오피스 클라이언트(205)로서 미리 설명된)는 매우 제한된 컴퓨팅 또는 그래픽 성능을 가진 매우 가격이 저렴하고 낮은 전력 장치일 것이고, 훨씬 제한적이거나 로컬 대용량 스토리지를 갖지 않을 것이다. 대조적으로, SAN(403) 및 다수 RAID(405)와 결합된 각 서버(402)는 예외적으로 높은 성능 컴퓨팅 시스템일 수 있고, 실제로 만약 다수 서버가 병렬 처리 구성에서 협력하여 사용된다면, 견디려고 구입될 수 있는 많은 컴퓨팅 및 그래픽 처리 파워에 거의 제한이 없다. 그리고, 낮은-레이턴시 비디오 압축(404) 및 낮은-레이턴시 비디오 압축(412) 때문에, 사용자에게 지각적으로, 서버(402)의 컴퓨팅 파워는 사용자에게 제공될 것이다. 사용자가 입력 장치(421)에서 버튼을 누르면, 마치 게임 또는 어플리케이션 소프트웨어가 로컬로 동작되듯이, 디스플레이(422)에 이미지는 현저한 지연없이 지각적으로 버튼 누름에 응답하여 업데이트된다. 따라서, 매우 낮은 성능 컴퓨터 또는 오직 값싼 칩을 가진 홈 또는 오피스 클라이언트(415)는 낮은-레이턴시 비디오 신장 및 제어 신호 로직(413)을 실행하고, 사용자는 로컬로 이용가능한 것처럼 원거리 위치에서 효율적인 임의적 컴퓨팅 파워로 제공된다. 이는 사용자에게 가장 진보된, 프로세서-집중적인(전형적으로 새로운) 비디오 게임 및 최고 성능 어플리케이션을 제시한다.

[0091] 도 4c는 매우 기본적이고 비싸지 않은 홈 또는 오피스 클라이언트 장치(465)를 도시한다. 이 장치는 도 4a 및 4b로부터 홈 또는 오피스 클라이언트(415)의 실시예이다. 그것은 대략 2인치 길이이다. PoE(Power over Ethernet)를 갖는 이더넷 케이블과 인터페이스하는 이더넷 잭(462)을 갖고, 그것으로부터 파워 및 인터넷간의 연결을 얻는다. 이는 NAT(Network Address Translation)를 지지하는 네트워크 내에서 NAT를 동작할 수 있다. 오피스 환경에서, 많은 새로운 이더넷 스위치는 PoE를 갖추고 오피스에 이더넷 잭에 직접 PoE를 가져간다. 이러한 상황에서, 요구되는 모든 것은 월 잭(wall jack)에서 클라이언트(465)까지의 이더넷 케이블이다. 만약 이용가능 이더넷 연결이 파워를 전달하지 않으면(예컨대, 가정에서 DSL 또는 케이블 모뎀으로, 그러나 PoE는 아닌), PoE를 갖는 출력 이더넷 및 파워가 없는 이더넷 케이블을 허용하는 비싸지 않은 이용가능한 벽"브릭(bricks)"(예컨대 파워 서플라이)이 있다.

[0092] 클라이언트(465)는 블루투스 무선 인터페이스와 결합된 제어 신호 로직(413)(도 4a)을 포함하고, 이는 키보드, 마우스, 게임 컨트롤러 및/또는 마이크로폰 및/또는 헤드셋과 같은 블루투스 입력 장치(479)와 인터페이스된다. 또한, 클라이언트(465)의 일 실시예는 택일적으로 하나의 눈을 셔터(shutter)하기 위해 한 쌍의 셔터된 안경(전형적으로 적외선을 통해서)로 120fps 비디오 및 오디오를 지원할 수 있는 디스플레이 장치(468)와 결합되어서 120fps에서 비디오를 출력할 수 있다. 사용자에 의해 감지된 효과는 디스플레이 스크린의 "비약 전환(jumps out)"된 입체적 3D 이미지이다. 그러한 동작을 지원하는 디스플레이 장치(468)는 삼성 HL-T5076S이다. 각 눈을 위한 비디오 스트림이 분리됨으로써, 일 실시예에서 2개의 독립적인 비디오 스트림은 호스팅 서비스(210)에 의해서 압축되고, 프레임들은 제때에 끼워지며, 프레임들은 클라이언트(465) 내에 2개의 독립적인 신장 처리로 신장된다.

[0093] 또한, 클라이언트(465)는 낮은-레이턴시 비디오 신장 로직(412)을 포함하고, 이는 들어온 비디오 및 오디오 압축을 해제하고 HDMI(High-Definition Multimedia Interface)를 통해서, 비디오 및 오디오를 TV로 제공하는 SDTV(Standard Definition Television) 또는 HDTV(High Definition Television)와 플러그되는 커넥터(463), 또는 HDMI를 지원하는 모니터(468)를 통해서 출력한다. 만약 사용자의 모니터(468)가 HDMI를 지원하지 않는다면, HDMI to DVI (디지털 비주얼 인터페이스)가 사용될 수 있지만, 오디오는 손실될 것이다. HDMI 표준하에서, 디스플레이 성능(464)(예컨대 지원되는 해상도, 프레임 비율)은 디스플레이 장치(468)와 통신하고, 그 다음 이 정보

가 인터넷 커넥션(462)를 통해서 호스팅 서비스(210)로 전달되고 디스플레이 장치에 적절한 포맷으로 압축된 비디오를 스트림할 수 있다.

[0094] 도 4d는 홈 또는 오피스 클라이언트 장치(475)가 도 4c에서 도시된 홈 또는 오피스 클라이언트 장치(465)와 외부 인터페이스를 더 갖는 것을 제외하고는 동일하게 도시하고 있다. 또한, 클라이언트(475)는 전력을 얻기 위해 PoE 뿐만아니라 벽에 플러그되는 외부 전력 공급 어댑터(도시하지 않음)로부터 얻을 수 있다. 클라이언트(457)가 USB 입력을 사용하면, 비디오 카메라(477)은 클라이언트(475)에 압축된 비디오를 제공하고, 이는 이하 설명하는 사용에서 호스팅 서비스(210)을 위해 클라이언트(475)에 의해 업로드된다. 카메라(477)로 구성되는 것은 이하 설명하는 압축 기술을 이용하는 낮은-레이턴시 압축기이다.

[0095] 인터넷 연결을 위해 이더넷 컨넥터를 추가하여, 또한 클라이언트(475)는 인터넷을 위해서 802.11g 무선 인터페이스를 갖는다. 두 인터페이스는 NAT를 지원하는 네트워크내에서 NAT를 사용할 수 있다.

[0096] 또한, 비디오 및 오디오 출력을 위해 HDMI 커넥터를 갖추는 것에 부가하여, 클라이언트(475)는 또한 아날로그 출력(그리고 표준 어댑터 케이블은 VGA 출력을 제공하게 됨)을 포함하는 듀얼 링크 DVI-I 커넥터를 갖는다. 이는 또한 컴퍼지트 비디오 및 S-video를 위한 아날로그 출력을 갖는다.

[0097] 오디오에 대해, 클라이언트(475)는 왼쪽/오른쪽 아날로그 스테레오 RCA 잭을 갖고 디지털 오디오 출력을 위해 TOSLINK 출력을 갖는다.

[0098] 입력 장치(479)를 위한 블루투스 무선 인터페이스에 부가하여, 또한 입력 장치와 인터페이스하도록 USB 잭이 필요하다.

[0099] 도 4e는 클라이언트(465)의 내부 아키텍처의 일 실시예를 도시하고 있다. 다이어그램에 도시된 전체 또는 일부 장치는 커스텀 디자인 또는 규격품인, 커스텀 ASIC 또는 몇몇의 개별 장치, 필드 프로그래머블 로직 어레이(Field Programmable Logic Array)에서 실행될 수 있다.

[0100] PoE를 갖는 이더넷(497)은 이더넷 인터페이스(481)에 부착된다. 파워(499)는 PoE를 갖는 이더넷(497)으로부터 파생되고 클라이언트(465)에서 나머지 장치와 연결된다. 버스(480)는 장치 사이에서 통신을 위한 공통 버스이다.

[0101] 플래시(476; Flash)로부터 작은 클라이언트 제어 어플리케이션을 실행시키는 제어 CPU(483)(임베디드 RAM을 가진 100MHz에서 MIPS R4000 CPU 시리즈와 같은 어떤 작은 CPU가 대체로 적당하다)는 네트워크를 위해 프로토콜 스택을 실행하고 또한 호스팅 서비스(210)와 통신하며, 클라이언트(465)에서 모든 장치를 구성한다. 이는 또한 입력 장치(469)와 인터페이스를 다루고, 만약 필요하면 포워드 에러 보정에 의해서 보호되는 사용자 컨트롤러 데이터를 갖는 호스팅 서비스(210)로 패킷을 되돌려 보낸다. 또한, 제어 CPU(483)는 패킷 트래픽을 감시한다(예컨대, 만약 패킷들이 손실되거나 지연되면 또한 패킷은 그들의 도착을 타임스탬프(timestamp)한다). 이 정보는 호스팅 서비스(210)으로 되돌려 보내져서 일정하게 네트워크 연결을 감시하고 그에 맞춰 보낸 것을 조정한다. 플래시 메모리(476)는 처음에 제어 CPU(483)를 위한 제어 프로그램 및 특정 클라이언트(465) 유닛에게 고유한 시리얼 넘버를 초반에 로드한다. 이 시리얼 넘버는 클라이언트(465) 유닛을 유일하게 식별하기 위해 허용된다.

[0102] 블루투스 인터페이스(484)는 클라이언트(465) 내부, 안테나를 통해서 무선으로 입력 장치(469)와 통신한다.

[0103] 비디오 신장기(486)는 여기서 설명된 비디오 신장을 수행하기 위해서 구성된 낮은-레이턴시 비디오 신장기이다. 많은 비디오 신장 장치는 FPGA 또는 커스텀 ASIC으로 통합될 수 있는 디자인의 지적 재산(Intellectual Property; IP) 또는 규격화(off-the-shelf)되어 존재한다. H.264 디코더를 위한 IP를 제안하는 한 회사는 오스트레일리아 뉴 사우스 웨일즈, Ocean Logic of Manly 이다. IP 사용의 이점은 여기서 사용된 압축 기술이 압축 표준을 따르지 않는다는 것이다. 어떤 표준 신장기는 여기서 압축 기술을 수용하기 위해서 유연하게 구성되기에 충분하지만, 어떤 것은 그렇지 않다. 그러나 IP를 가지고, 요구대로 신장기를 다시 디자인하는데 유연성이 모두 갖춰져 있다.

[0104] 비디오 신장기의 출력은 비디오 출력 서브시스템(487)과 연결되고, 이는 HDMI 인터페이스(490)의 비디오 출력을 위한 비디오와 연결된다.

[0105] 오디오 신장 서브시스템(488)은 이용가능한 표준 오디오 신장기를 사용하거나, 그것은 IP로서 실행될 수 있거나, 오디오 신장기는 예컨대, Vorbis 오디오 시장기를 실행할 수 있는 제어 프로세서(483) 내에서 실행될 수 있다.

[0106] 오디오 신장을 실행하는 장치는 HDMI 인터페이스(490)의 오디오 출력을 위한 오디오와 결합되는 오디오 출력 서브시스템(489)과 결합된다.

[0107] 도 4f는 클라이언트(475)의 내부 아키텍처의 일 실시예를 도시하고 있다. 아키텍처는 벽에서 플러그되는 파워 서플라이 어댑터로부터 선택적인 외부 DC파워 및 추가된 인터페이스를 제외하고는 클라이언트(465)의 것과 동일하고, 만일 그렇게 사용되면, 이더넷 PoE(497)로부터 들어오는 파워를 대체한다. 클라이언트(465)와 공통된 목적은 이하 반복되지 않을 것이지만, 추가된 목적은 이하에서 설명된다.

[0108] CPU(483)는 부가 장치와 통신함과 더불어 부가 장치를 구성한다.

[0109] WiFi 서브시스템(482)운 안테나를 통하여 이더넷(497)과 택일적으로 접속하는 무선 인터넷을 제공한다. WiFi 서브시스템(485)은 캘리포니아, 산타클라라의 Atheros Communication을 포함하는 광범위한 범위의 제조자로부터 이용될 수 있다.

[0110] USB 서브시스템(485)는 유선 USB 입력장치(479)를 위한 블루투스 통신에 대해 택일성을 제공한다. USB 서브시스템은 꽤 표준적이고 상당히 FPGA 및 ASIC에서 이용가능하고, 뿐만아니라 종종 비디오 신장과 같은 그 밖의 기능을 수행하는 규격화된 장치로 설계된다.

[0111] 비디오 출력 서브시스템(487)은 클라이언트(465)내에서 보다 더 넓은 범위의 비디오 출력을 생산한다. HDMI(490)가 제공하는 비디오 출력에 부가하여, DVI-I(491), S-video(492), 및 컴포지트 비디오(493)을 제공한다. 또한, DVI-I(491) 인터페이스가 디지털 비디오를 위해 사용될 때, 디스플레이 성능(464)은 디스플레이 장치에서 제어 CPU(483)로 되돌아 전달됨으로써, 디스플레이 장치(478) 성능의 호스팅 서비스(210)를 알릴 수 있다. 비디오 출력 서브시스템(487)에 의해서 제공되는 모든 인터페이스는 상당히 표준적인 인터페이스이고 꽤 많은 형태로 이용가능하다.

[0112] 오디오 출력 서브시스템(489)은 디지털 인터페이스(494; S/PDIF 및/또는 Toslink)를 통해서 디지털인 오디오를 출력하고, 스테레오 아날로그 인터페이스(495)을 통해서 아날로그 형태로 오디오를 출력한다.

[0113] **왕복 레이턴시 분석(Round-Trip Latency Analysis)**

[0114] 물론 앞서 언급한 단락의 이점에서 알 수 있듯이, 입력 장치(421)를 사용하고 디스플레이 장치(420)에서 동작의 결과를 보여주는 사용자의 동작 사이의 왕복 레이턴시는 70-80ms 보다 크지 않다. 이러한 레이턴시는 사용자 구내(211)의 입력 장치(421)에서 호스팅 서비스(210)까지의 경로 그리고 디스플레이 장치(422)를 위해 사용자 구내(211)로 다시 되돌아오는 경로에 있는 모든 요소를 고려해야 한다. 도 4b는 신호가 이동해야하는 다양한 구성 및 네트워크를 도시하고, 상기 이러한 요소 및 네트워크는 실제적인 실행에서 기대될 수 있는 모범적인 레이턴시를 리스트하는 타임라인(timeline)이다. 도 4b는 간략하게 되어서 오직 임계 경로 라우팅(critical path routing)만이 도시됨을 주지해야 한다. 시스템의 그 밖의 특징을 위해 사용되는 그 밖의 데이터의 라우팅은 아래에서 설명된다. 양쪽 화살표(예컨대, 화살표(453))는 왕복 레이턴시를 가리키고 한쪽 화살표(예컨대, 화살표(457))는 편도 레이턴시를 가리키고, "~"는 대략적인 측정을 나타낸다. 이는 리스트된 레이턴시가 성취할 수 없는 실제-세상 상황이 있을 것임을 가리키고 있을 것이지만, 미국에서 많은 경우에, DSL 및 케이블 모뎀을 사용자 구내(211)와 연결하기 위해서 사용하면, 이러한 레이턴시는 다음 절에서 설명되는 상황하에서 성취될 수 있다. 또한, 인터넷에 대한 셀룰러 무선 연결은 확실히 도시된 시스템에서 동작할 것인 반면에, 대부분 현재 미국 셀룰러 데이터 시스템(EVDO와 같은)은 매우 높은 레이턴시를 발생시키고 도 4b에서 도시된 레이턴시를 달성할 수 없을 것이다. 그러나, 이러한 근본적인 이론은 이러한 수준의 레이턴시를 실행하는 성능이 있는 향후의 셀룰러 기술에서 실행될 것이다.

[0115] 사용자 구내(211)에서 입력 장치(421)로부터 시작되면, 일단 사용자는 입력 장치(421)을 작동하고, 사용자 제어 신호는 클라이언트(415)(이는 셋톱 박스와 같은 기본 장치일 것이거나, 그것은 PC 또는 모바일 장치와 같은 다른 장치에서 구동되는 소프트웨어 또는 하드웨어일지 모른다)로 보내지며, 패킷화되고, 그 패킷은 호스팅 서비스(210)로 도달하기 위해서 목적지 주소가 부여된다. 패킷은 또한 제어 신호가 들어올 때 사용자를 가리키는 정보를 포함한다. 그 다음 제어 신호 패킷(들)은 방화벽/라우터/NAT(Network Address Translation) 장치(443)를 통하여 WAN 인터페이스(442)로 보내진다. WAN 인터페이스(442)는 사용자의 ISP(Internet Service Provider)에 의해서 사용자 구내(211)에 제공되는 인터페이스 장치이다. WAN 인터페이스(442)는 케이블 또는 DSL 모뎀, WIMAX 트랜스시버, 파이버 트랜스시버, 셀룰러 데이터 인터페이스, 파워라인(powerline) 상에 인터넷 프로토콜 인터페이스, 또는 인터넷을 위한 그 밖의 많은 인터페이스일 것이다. 더욱이, 방화벽/라우터/NAT 장치(443)(그리고 잠재적인 WAN 인터페이스(442))는 클라이언트(415)와 통합될 것이다. 이러한 예는 어떤 표준(예컨대,

802.11g)을 통해서 무선으로 인터넷과 연결되고 전송되는 수단뿐만 아니라 홈 또는 오피스 클라이언트(415)의 기능을 실행하기 위한 소프트웨어를 포함하는, 모바일폰일 것이다

[0116] 그 다음 WAN 인터페이스(442)는 사용자 구내(211)와 연결된 WAN 트랜스포트와 일반 인터넷 또는 개인 네트워크 사이의 인터페이스를 제공하는 시설인 사용자의 인터넷 서비스 제공자(Internet Service Provider; ISP)를 위해 여기서는 "상호 접속 위치(point of presence)"로 불리는 것으로 제어 신호를 전송한다. 상호 접속 위치의 특징은 제공된 인터넷 서비스의 특성에 의존하여 다양하게 될 것이다. DSL의 경우, 그것은 전형적으로 DSLAM이 위치한 곳인 전화 회사 센트럴 오피스일 것이다. 케이블 모뎀의 경우, 그것은 전형적으로 케이블 멀티 시스템 오퍼레이트(Multi-System Office) 전파중계소(head end)가 될 것이다. 셀룰러 시스템의 경우, 전형적으로 셀룰러 타워와 관련된 제어 룸(control room)이 될 것이다. 그러나 상호 접속 위치의 특성이 무엇이든지 간에, 일반 인터넷(410)으로 제어 신호 패킷(들)이 전송될 것이다. 그 다음 제어 신호 패킷(들)은 가장 쉽게는 파이버 트랜시버 인터페이스를 통해서, 호스팅 서비스(210)를 위해서 WAN 인터페이스(441)로 전송될 것이다. 그 다음 WAN 인터페이스(441)는 제어 신호 패킷을 루팅 로직(409)(이는 이더넷 스위치 및 루팅 서버를 포함하는, 많은 다른 방법으로 실행될 수 있는)로 전송할 것이고, 이는 사용자의 주소를 평가하고 주어진 사용자를 위해 올바른 서버(402)로 제어 신호(들)를 전송한다.

[0117] 그 다음 서버(402)는 서버(402)에서 동작하는 게임 또는 어플리케이션 소프트웨어를 위한 입력으로써 제어 신호를 가져가고 게임 또는 어플리케이션의 다음 프레임을 처리하기 위해서 제어 신호를 사용한다. 다음 프레임이 생성되면, 비디오 및 오디오는 서버(402)로부터 비디오 압축기(404)로 출력될 것이다. 비디오 및 오디오는 서버(402)에서 압축기(404)로 다양한 수단을 통해서 출력될 것이다. 우선, 압축기(404)는 서버(402)의 일부로 구성될(built into) 것이고, 그래서 압축은 서버(402)내에서 국부적으로 실행될 것이다. 또는, 비디오 및/또는 오디오는 서버(402)와 비디오 압축기(404) 사이의 개인 네트워크 또는 SAN(403)과 같은 공유된 네트워크를 통한 이더넷 연결과 같은 네트워크 연결을 통해서 패킷화된 형태로 출력될 것이다. 또는, 비디오는 서버(402)로부터 DVI 또는 VGA 커넥터와 같은 비디오 출력 커넥터를 통해 출력될 것이고, 그 다음 비디오 압축기(404)에 의해서 캡쳐된다. 또한, 오디오는 디지털 오디오(예컨대, TOSLINK 또는 S/PDIF 커넥터를 통해서) 또는 비디오 압축기(404) 내에 오디오 압축 로직에 의해서 엔코드되고 디지털화되는 아날로그 오디오로써 서버(402)로부터 출력된다.

[0118] 비디오 압축기(404)가 비디오 프레임, 그리고 서버(402)로부터 프레임 시간 동안 생성된 오디오를 캡쳐하고, 그 다음 비디오 압축기는 비디오 및 오디오를 아래 설명되는 기술을 사용하여 압축할 것이다. 비디오 및 오디오가 압축되면, 그것은 사용자의 클라이언트(415)로 다시 되돌리기 위한 주소로 패킷화되고, 그것은 WAN 인터페이스(441)로 전송되며, 그 다음 비디오 및 오디오 패킷은 일반 인터넷(410)를 통해 전송되고, 사용자의 ISP 상호 접속 위치(441)로 비디오 및 오디오 패킷을 전송하며, 그 다음 사용자의 구내에서 WAN 인터페이스(442)로 비디오 및 오디오 패킷을 전송하고, 방화벽/라우터/NAT 장치(443)으로 비디오 및 오디오 패킷을 전송하며, 그 다음 클라이언트(415)로 비디오 및 오디오 패킷을 전송한다.

[0119] 클라이언트(415)는 비디오 및 오디오를 신장하고, 그 다음 디스플레이 장치(422)(또는 클라이언트의 일체형 디스플레이 장치)에서 비디오를 디스플레이하고, 디스플레이 장치(422) 또는 개별 앰프/스피커 또는 클라이언트에 일체형 앰프/스피커로 오디오를 전송한다.

[0120] 사용자가 지각적인 랙(lag)없이 설명된 전체 처리를 인식하는 것은 왕복 지연이 70 또는 80ms 이하일 필요가 있다. 설명된 왕복 경로에서 어떤 레이턴시 지연은 호스팅 서비스(210)의 제어하에 있고 그리고/또는 사용자 및 그 밖의 사람은 아니다. 그럼에도 불구하고, 많은 수의 실세계의 시나리오의 테스트와 분석에 기초할 때, 다음은 대략적인 측정이다.

[0121] 제어 신호(451)를 보내기 위한 편도 전송 시간은 전형적으로 1ms이하이고, 사용자 구내(452)를 통한 왕복 전송은 전형적으로 대략 1ms에서 쉽게 이용가능한 소비자 등급 이더넷 위에 방화벽/라우터/NAT 스위치를 이용하여 성취될 수 있다. 사용자 ISP는 그들의 왕복 지연(453)에서 넓게 변화하지만, DSL 및 케이블 모뎀 공급자를 갖으므로, 우리는 전형적으로 10 에서 25ms 사이로 볼 수 있다. 일반 인터넷에서 왕복 레이턴시는 크게 얼마나 많은 트래픽이 전송되었는지에 따라 변화할 수 있고 전송에서 어떤 실패가 있었는지에 따라 변화할 수 있으나, 전형적으로 일반 인터넷은 공정하게 최적으로 루트를 제공하고(그리고 이러한 이슈는 아래에서 설명된다), 레이턴시는 대체로 광섬유를 통하는 광의 속도, 목적지까지 주어진 거리를 통해서 결정된다. 아래에서 더 설명됨으로써, 우리는 사용자 구내(211)에서 떨어지는 호스팅 서비스(210)로 기대하는 최대한의 거리는 대충 1000마일로 확립된다. 1000 마일에서(왕복 2000 마일), 인터넷을 통하여 신호를 전송하는 실제적 시간은 대략적으로 22ms이다.

호스팅 서비스(210)을 위한 WAN 인터페이스(441)는 전형적으로 무시할 수 있는 레이턴시를 갖는 상업 등급 섬유의 고속 인터페이스이다. 따라서, 일반적인 인터넷 레이턴시(454)는 전형적으로 1 에서 10ms 사이이다. 호스팅 서비스(210)을 통한 편도 루팅(455) 레이턴시는 1ms 이하로 성취될 수 있다. 서버(402)는 전형적으로 한 프레임 시간(이는 60fps에서 16.7ms)보다 적은 시간에 게임 또는 어플리케이션을 위한 새로운 프레임을 연산할 것이고 16ms는 사용하기에 합리적인 최대 편도 레이턴시(456)이다. 여기서 설명된 비디오 압축 및 오디오 압축 알고리즘의 최적화된 하드웨어 실행에서, 압축(457)은 1ms내에 완료될 수 있다. 덜 최적화된 버전에서, 압축은 6ms 만큼 많이 걸릴지도 모른다(물론 최적화된 버전에 비해서 오래 걸리지만, 그러한 실행은 왕복 전체 레이턴시에 영향을 줄 것이고, 70 ~ 80ms가 유지되기 위하여 단축되도록(예컨대, 일반 인터넷을 통해서 허락된 거리가 줄어들 수 있다) 그 밖의 레이턴시가 요구될 것이다). 인터넷(454), 사용자 ISP(453), 및 사용자 구내 루팅(452)의 왕복 레이턴시는 이미 고려되었고, 그래서 남은 것은 비디오 신장(458) 레이턴시로, 비디오 신장(458)이 전용 하드웨어에서 실행되는지, 또는 만약 클라이언트 장치(415)(PC 또는 모바일 장치)의 소프트웨어에서 실행된다면, 그것은 디스플레이의 크기 및 신장하는 CPU의 성능에 매우 의존할 수 있게 된다. 전형적으로, 신장(458)은 1 에서 8ms 사이가 걸린다.

[0122] 따라서, 실행에서 보여지는 최악의 경우 레이턴시를 함께 더함으로서, 우리는 도 4a에서 도시하고 있는 시스템의 사용자에 의해 경험한 것으로 기대할 수 있는 최악의 경우 왕복 레이턴시를 결정할 수 있다. 그들은: 1+1+25+22+1+16+6+8 = 80ms이다. 그리고, 실제로 실행에서(아래 설명된 경고로), 이는 원형(prototype) 버전의 도 4a에 도시하고 있는 시스템을 사용하는 것, 미국 내에 클라이언트 장치, 홈 DSL 및 케이블 모뎀 커넥션처럼 규격화된 윈도우 PC를 사용하는 것으로 왕복 레이턴시가 도시된다. 물론, 최악의 경우보다 더 나은 시나리오는 훨씬 짧은 레이턴시를 초래할 수 있으나, 그들은 넓게 사용되는 상업 서비스를 개발하는데 필요로 하진 않는다.

[0123] 일반 인터넷상에 도 4b에 리스트된 레이턴시를 달성하기 위해서는, 매우 특별한 특성의 패킷 스트림을 생성하기 위해 도 4a로부터 클라이언트(415)에 있는 비디오 압축기(404) 및 비디오 신장기(412)가 요구되고, 그로인해 호스팅 서비스(210)에서 디스플레이 장치(422)까지 전체 경로를 통해서 생성된 패킷 시퀀스는 지연 또는 초과된 패킷 손실이 생기지 않고, 특히 WAN 인터페이스(442) 및 방화벽/라우터/NAT(443)을 통한 사용자의 인터넷 연결상에 사용자가 이용가능한 대역폭의 제한을 일관되게 떨어뜨린다. 더욱이, 비디오 압축기는 충분히 강력한 패킷 스트림을 생성하여서 그것은 보통 인터넷 및 네트워크 전송에서 일어나는 필연적인 패킷 손실 및 패킷 재주문(reordering)을 견딜 수 있다.

[0124] **<u>낮은-레이턴시 비디오 압축(Low-Latency Video Compression)</u>**

[0125] 앞서 언급한 목표를 성취하기 위해서, 일 실시예는 전송되는 비디오를 위해 피크 대역폭 요구 및 레이턴시를 감소하는 비디오 압축을 위한 새로운 접근에 이르게 한다. 이러한 실시예의 상세한 설명에 앞서, 현재 비디오 압축 기술의 분석은 도 5, 도 6a, 및 도 6b에 대하여 제공될 것이다. 물론, 만약 사용자가 이러한 기술에 의해서 요구되는 전송률을 다루기 위한 충분한 대역폭을 제공받는다면, 이러한 기술은 본질적인 이론과 일치되도록 채택될 것이다. 비디오 압축과 공시상태(synchrony) 및 동시에 실행된다는 것을 언급하는 것 외에 오디오 압축은 여기서 설명되지 않음을 주목하라. 종래 기술 오디오 압축 기술은 이 시스템을 위한 요구사항을 만족하도록 존재한다.

[0126] 도 5는 일련의 압축된 프레임(511 - 513)을 생성하기 위해 특정 압축 알고리즘을 사용한 압축기 로직(520)에 의해서 압축되는 각각 개별적인 비디오 프레임(501 - 503)에서 비디오 압축을 위한 특정 종래 기술을 도시하고 있다. 이 기술의 일 실시예는 이산 코사인 변환(discrete cosine transform; DCT)에 기초한 각 프레임이 Joint Picture Expert Group(JPEG) 압축 알고리즘에 따라 압축된 "움직임 JPEG"이다. 다양한 다른 형태의 압축 알고리즘이 채택될 것이지만, 반면 아직 이러한 근본적인 이론에 따른다(예컨대, JPEG-2000과 같은 웨이브렛-기반(wavelet-based) 압축 알고리즘).

[0127] 이러한 형태의 압축이 가진 하나의 문제는 각 프레임의 전송률을 줄인다는 것이지만, 그것은 전체 비디오 스트림의 전송률을 줄이기 위해 연속적인 프레임 사이의 유사성을 활용하지 않는다. 예컨대, 도 5에서 도시된 것처럼, 604×480×24비트/픽셀 = 604*480*24/8/1024=900킬로바이트/프레임(KB/프레임)으로 가정하면, 주어진 품질의 이미지의 경우, 움직임 JPEG는 오직 10의 비율로 압축하여, 90KB/프레임의 데이터 스트림을 초래한다. 60프레임/초에서, 이는 채널 대역폭 90KB*8비트*60프레임/초 = 42.2Mbps를 요구할 것이고, 이는 오늘날 미국에서 거의 모든 홈 인터넷 연결보다 훨씬 너무 높은 대역폭일 것이고, 많은 오피스 인터넷 연결에 대해서도 너무 높은 대역폭일 것이다. 실제로, 이러한 높은 대역폭에서 일정한 데이터 스트림을 요구하게 되면, 오피스 LAN환경에서

조차 그것은 오직 한명의 사용자에게 제공하게 될 것이고, 그것은 100Mbps 이더넷 LAN의 대역폭의 대부분의 비율을 소모하게 될 것이고 LAN을 지원하는 이더넷 스위치를 훨씬 부담되게 할 것이다. 따라서, 동영상을 위한 압축은 그 밖의 압축 기술과 비교할 때 비효율적이다(아래 설명되는 것처럼). 더욱이, JPEG 및 JPEG-2000과 같은 단일 프레임 압축 알고리즘은 정지 이미지(예컨대, 눈이 얼마나 빽빽한 나뭇잎을 나타낼 수 있는지 정확하게 모르기 때문에 장면에 있는 빽빽한 나뭇잎에 아티팩트는 아티팩트로 나타내지 않을 것이다)로 인식할 수 없을지 모르는 압축 아티팩트(artifact)를 생산하는 손실있는 압축 알고리즘을 사용한다. 그러나, 일단 장면이 움직이면, 장면의 한 영역에서 아티팩트는 정지 이미지로 인식할 수 없다는 사실에도 불구하고, 눈이 프레임마다 변화되는 아티팩트를 발견하기 때문에 아티팩트는 두드러질 수 있다. 이는 프레임의 시퀀스에서 "배경 노이즈(background noise)" 인식의 결과이고, 미미한 아날로그 TV 수신상태 동안 볼 수 있는 "스노우(snow)" 노이즈에서 나타나는 것과 비슷하다. 물론, 이러한 형태의 압축은 여전히 여기서 설명되는 어떤 실시예에서 사용할 것이지만, 일반적으로 말해서, 장면에서 배경 노이즈를 피하기 위해, 높은 전송률(예컨대, 낮은 압축 비율)는 주어진 지각적인 품질을 위해 요구된다.

[0128] H.264 또는 윈도우 미디어 VC9, MPEG2 및 MPEG4와 같은 다른 형태의 압축은 비디오 스트림을 압축하는데 모두 효율적인데, 이는 연속적인 프레임 사이의 유사성을 활용하기 때문이다. 이러한 기술은 모두 비디오를 압축하기 위해 동일한 일반적 기술에 따른다. 따라서, 비록 H.264 표준은 설명될지라도, 그러나 동일한 일반적인 이론은 그 밖의 압축 알고리즘에 적용된다. 많은 수의 H.264 압축기 및 신장기가 이용가능하고, H.264를 압축하기 위한 x264 오픈 소스 소프트웨어 라이브러리 및 H.264를 신장하기 위한 FFmpeg 오픈 소스 소프트웨어 라이브러리를 포함한다.

[0129] 도 6a 및 6b는 일련의 비압축된(uncompressed) 비디오 프레임(501 - 503, 559 - 561)이 압축 로직(620)에 의해서 일련의 "I 프레임"(611,671); "P 프레임"(612,613); 및 "B 프레임"(670)으로 압축되는 모범적인 종래 압축 기술을 도시하고 있다. 도 6a에서 수직축은 일반적으로 엔코드된 각 프레임(비록 프레임은 스케일로 도시되지 않았지만)의 결과 크기를 나타낸다. 설명된 것처럼, I 프레임, B 프레임, 및 P 프레임을 사용하는 비디오 코딩은 해당 기술 분야에 사람들에게 분명하게 이해될 것이고, I 프레임(611)은 완전히 비압축된 프레임(501)(앞서 설명된 것처럼 압축된 JPEG 이미지와 유사한)의 DCT 기반 압축이다. P 프레임(612,613)은 일반적으로 I 프레임(611)보다 현저하게 작다. 왜냐하면 그들은 선행 I 프레임 또는 P 프레임에서 데이터를 이용하기 때문이다; 즉, 그들은 선행 I 프레임 또는 P 프레임 사이의 변화를 가리키는 데이터를 포함한다. B 프레임(670)은 B 프레임이 선행 기준 프레임에 있는 잠재적인 프레임뿐만 아니라 다음 기준 프레임에 프레임을 사용하는 것을 제외하고는 P 프레임의 것과 비슷하다.

[0130] 다음 논의에서, 요청된 프레임 속도는 60프레임/초이고, 각 I 프레임은 대략 160Kb이고, 평균 P 프레임 및 B 프레임은 16Kb 이고 새로운 I 프레임은 매 초마다 생성된다. 이러한 세트의 파라미터로, 평균 전송률은 : 160Kb + 16Kb * 59 = 1.1Mbps 일 것이다. 이 전송률은 홈 및 오피스를 위한 현재 많은 광대역 인터넷 연결의 최대 전송률 이내로 떨어진다. 이 기술은 또한 오직 인트라프레임(intraframe) 엔코딩으로부터 배경 노이즈 문제를 피하려는 경향이 있고 왜냐하면 P 프레임 및 B 프레임은 프레임들 사이의 차이를 추적하고, 그래서 압축 아티팩트가 상기 설명한 배경 노이즈 문제를 줄이거나 프레임 투 프레임에서 사라지고 나타나지 않는 경향이다.

[0131] 앞서 말한 형태의 압축이 갖는 하나의 문제는 비록 평균 전송률이 대체적으로 낮더라도(예컨대, 1.1Mbps), 단일 I 프레임은 전송을 위해 몇 프레임 시간이 걸릴 것이다. 예컨대, 종래 기술을 사용하는 2.2Mbps 네트워크 연결(예컨대, 도 3a로 부터 최대 이용가능한 전송률(302)의 2.2Mbps 피크를 갖는 DSL 또는 케이블 모뎀)은 전형적으로 각 60 프레임인 160Kbps I 프레임으로 1.1Mbps에서 비디오를 스트림하는 것이 적당할 것이다. 이는 비디오를 신장하기 전에 비디오의 1초 큐 업(queue up)하는 신장기를 가짐으로써 달성될 수 있다. 1초에, 데이터의 1.1Mb는 전송될 것이고, 이는 쉽게 최대 이용가능한 전송률인 2.2Mbps까지 수용될 것이고, 이용가능한 전송률이 50% 이상까지 주기적으로 내려갈 것이다. 불행히도, 이러한 종래 기술 접근은 리시버에서 1초 비디오 버퍼 때문에 비디오를 위한 1초 레이턴시를 초래할 것이다. 그러한 지연은 많은 선행 어플리케이션(예컨대, 선형 비디오의 재생)에 적합하지만, 그것은 70 ~ 80ms의 레이턴시 이상을 견딜 수 없는 빠른 동작 비디오 게임을 위해서는 너무 오래 걸리는 레이턴시이다.

[0132] 만약 1초 비디오 버퍼를 줄이기 위한 시도가 있다면, 그것은 여전히 빠른 동작 비디오 게임을 위한 레이턴시에서 적당한 생략을 초래하지 않을 것이다. 어떤 경우에, 앞서 설명했던 것처럼, B 프레임의 사용은, I 프레임뿐만 아니라 I 프레임에 선행하는 모든 B 프레임의 수신상태를 필요로 할 것이다. 만약 우리가 P 프레임과 B 프레임 사이 대략적으로 나눠진 59개 비-I 프레임을 추측하면, 거기에는 최소 29개 B 프레임과 어떤 B 프레임이 디스플레이되기 전에 수신된 하나의 I 프레임이 있을 것이다. 따라서, 이용가능한 채널의 대역폭에도 불구하고,

각 1/60 초 기간의 29+1=30 프레임의 지연 또는 500ms의 레이턴시를 필요로 할 것이다. 분명하게도 그것은 너무 길다.

[0133] 따라서, 다른 접근은 B 프레임을 제거할 것이고 오직 I 프레임 및 P 프레임을 사용한다.(이것의 하나의 결과는 주어진 품질 수준을 위해 전송률이 증가될 것이지만, 이러한 예에서 일관성을 위하여 각 I 프레임은 160Kb이고, 평균 P 프레임은 16Kb 크기이며, 따라서 전송률은 여전히 1.1Mbps를 계속해서 추측하다) 이러한 접근은 오직 먼저 수신된 프레임에 의존하는 각 P 프레임의 디코딩 때문에, B 프레임에 의해서 소개된 피할 수 없는 레이턴시를 제거한다. 이러한 접근에 남아 있는 한 문제는 대부분 홈 및 많은 오피스에서 정형적인 것처럼, 낮은 대역폭 채널에 I 프레임은 평균 P 프레임에 비해 너무 크고, I 프레임의 전송은 실질적으로 레이턴시가 추가된다는 것이다. 이는 도 6b에 도시하고 있다. 비디오 스트림 전송률(624)은 I 프레임의 경우를 제외하고는 이용가능한 최대 전송률(621)보다 아래이고, 여기서 I 프레임(623)을 위해 요구된 피크 전송률은 이용가능한 최대 전송률(622)(그리고 정격된 최대 전송률(621)조차)을 초과한다. P 프레임에 의해서 요구되는 전송률은 이용가능한 최대 전송률보다 작다. 2.2Mbps에서 이용가능한 최대 전송률 피크가 그것의 2.2Mbps 피크 속도에서 꾸준히 남아있음에도, 그것은 I 프레임을 전송하기 위해서 160Kb/2.2Mb = 71ms가 걸릴 것이고, 만약 이용가능한 최대 전송률(622)이 50%까지(1.1Mbps) 떨어지면, 그것은 I 프레임을 전송하기 위해서 142ms가 걸릴 것이다. 그래서, I 프레임을 전송하는 레이턴시는 71 ~ 142ms 사이에서 다소 떨어질 것이다. 이 레이턴시는 도 6b에서 확인된 레이턴시에 첨가된 이 레이턴시, 최악의 경우에 70ms까지 추가되서, 이것은 사용자가 입력 장치(421)를 작동하는 시점에서 디스플레이 장치(422)에 이미지가 나타날 때까지 141 ~ 222ms의 종합 왕복 레이턴시를 초래할 것이고, 이는 너무 높다. 그리고 만약 이용가능한 최대 전송률이 2.2 Mbps 아래로 떨어지면, 레이턴시는 더 증가될 것이다.

[0134] 또한 거기에는 일반적으로 이용가능한 전송률(622)을 훨씬 초과한 피크 전송률(623)을 가진 ISP에게 심각한 결과인 "전파방해(jamming)"가 있다. 다른 ISP에서 기구는 다르게 동작할 것이지만, 이용가능한 전송률(622)보다 매우 높은 전송률로 패킷을 수신할 때는 DSL 및 케이블 모뎀 ISP 사이에 다음의 행동이 상당히 공통된다 : (a) 그것들을 큐잉함으로써 패킷을 지연하고(레이턴시를 유도함), (b) 일부 또는 모든 패킷을 떨어트리고, (c) 어떤 기간 동안 연결을 비활성화한다(대부분 그것과 관련된 ISP는 "denial of service"와 같은 악의있는 공격 때문에). 따라서, 도 6b에서 도시한 것과 같은 특징을 갖는 최대의 전송률에서 패킷 스트림을 전송하는 것은 실행가능한 선택이 아니다. 피크(623)는 호스팅 서비스(210)에서 큐 업될 것이고, 앞선 단락에서 설명된 받아들일 수 없는 레이턴시를 유도하는 이용가능한 최대 전송률보다 아래 전송률로 전송한다.

[0135] 더욱이, 도 6b에 도시하고 있는 비디오 스트림 전송률 시퀀스(624)는 잘 "제어된(tame)" 비디오 스트림 전송률 시퀀스이고, 매우 작은 동작을 갖고 매우 많은 변화를 하지 않는 비디오 시퀀스로 비디오를 압축한 결과 기대되는 분류의 전송률 시퀀스일 것이다(예컨대, 카메라는 고정된 위치에 있고 거의 움직임이 없으며, 비디오 전화 회의에서 공통될 것이다).

[0136] 도 6c에 도시된 비디오 스트림 전송률 시퀀스(634)는 움직임 그림 또는 비디오 게임 또는 어떤 어플리케이션 소프트웨어에서 만들어지는 훨씬 많은 움직임을 갖는 비디오에서 볼 수 있는 전형적인 시퀀스이다. I 프레임 피크(633)에 추가된 것을 주목하라, 또한 거기에는 많은 경우에 이용가능한 최대 전송률을 초과하고 꽤 큰 P 프레임 피크(635 및 636과 같은)가 있다. 비록 이러한 P 프레임 피크는 I 프레임 피크만큼 꽤 크지 않지만, 그들은 여전히 최대 전송률에 채널에 의해서 운반되기에 너무 크고, I 프레임 피크와 마찬가지로, P 프레임 피크는 느리게 전송되어야 한다(그것 때문에 증가하는 레이턴시).

[0137] 높은 대역폭 채널에서(예컨대, 100Mbps LAN, 또는 높은 대역 폭 100Mbps 개인 연결) 네트워크는 I 프레임 피크(633) 또는 P 프레임 피크(636)처럼 큰 피크를 견딜 수 있을 것이고, 이론상으로 낮은-레이턴시는 유지될 수 있겠다. 그러나, 그러한 네트워크는 빈번하게 많은 사용자들 사이에서 공유되고(예컨대, 오피스 환경에서),특히 만약 네트워크 트래픽이 개인 공유 커넥션에 전송된다면(예컨대, 데이터 센터에서 오피스까지) 그러한 "피키(peaky)" 데이터는 LAN의 성능에 영향을 줄 것이다. 우선, 이러한 예는 대체적으로 60fps에서 낮은 해상도 비디오 스트림 640×480 픽셀이라는 것을 명심해야 한다. 60fps에서 HDTV 스트림의 1920×1080는 쉽게 현대 컴퓨터 및 디스플레이에 의해서 다뤄지고, 60fps에서 2560×1440 해상도 디스플레이는 점차적으로 이용가능하다(예컨대, Apple, Inc의 30인치 디스플레이). 60fps에서 1920×1080인 높은 동작 비디오 시퀀스는 합리적인 품질 수준을 위해 H.264 압축을 사용하는 4.5Mbps를 요구할 것이다. 만약 우리가 10×명목상 전송률에서 I 프레임 피크를 추측하면, 45Mbps 피크를 초래할 것이나, 작을 뿐만아니라 여전히 상당한 P프레임 피크이다. 만약 몇몇 사용자가 동일한 100Mbps 네트워크로 비디오 스트림을 수신한다면(예컨대, 오피스와 데이터 센터 간의 개인 네트워크 연결), 몇몇 사용자의 비디오 스트림으로부터 피크를 어떻게 조정할 수 있는지, 네트워크의 대역폭을 압도하는지, 네트워크에 사용자를 지지하는 스위치의 백플레이트의 대역폭을 잠재적으로 압

도하는지를 아는 것은 쉽다. 기가비트 이더넷 네트워크에서 조차, 만약 충분한 사용자가 한번에 충분히 조정된다면, 그것은 네트워크 또는 네트워크 스위치를 극복할 수 있다. 그리고, 일단 2560×1440 해상도 비디오가 보다 흔해지면, 평균 비디오 스트림 전송률은 9.5Mbps일 것이고, 아마도 95Mbps 피크 전송률을 초래한다. 말할 필요없이, 데이터 센터 및 오피스 사이의 100Mbps연결(이는 오늘날 예외적으로 빠른 연결)은 완전히 단일 사용자로부터 피크 트래픽에 의해서 쇄도될 것이다. 따라서, 비록 LAN 및 개인 네트워크 연결이 피키 스트리밍 비디오를 보다 잘 견딜 수 있더라도, 높은 피크를 갖는 스트리밍 비디오는 바람직하지 않고 특별한 계획 및 오피스의 IT 부서에 의한 시설이 요구될 것이다.

[0138] 물론, 표준 선형 비디오 어플리케이션을 위해 이러한 이슈는 문제가 아니다. 왜냐하면 전송률은 최대 이용가능한 전송률(622) 아래 각 프레임을 위한 데이터 및 전송의 포인트에서 "부드럽게(smoothed)" 되고, 클라이언트에 버퍼는 신장되기 전에 I,P,B 프레임의 시퀀스를 저장한다. 따라서, 네트워크 상에 전송률은 비디오 스트림의 평균 전송률에 가깝게 남게된다. 불행히도, 이는 B 프레임이 사용되지 않았음에도 레이턴시를 유도하고, 이는 빠른 응답 시간을 요구하는 비디오 게임 및 어플리케이션과 같은 낮은-레이턴시 어플리케이션을 위해서는 받아들여지지 않는다. 높은 피크를 갖는 비디오 스트림을 완화하는 종래 기술 해결책은 "Constant Bit Rate(CBR)"로 불리는 엔코딩 기술을 사용하는 것이다. 비록 CBR이란 용어는 모든 프레임이 동일한 비트율(bit rate)(예컨대, 크기)을 갖도록 압축됨을 의미하는 것처럼 생각될지라도, 일반적으로보통 그것이 언급하는 것은 소정 수의 프레임(우리의 경우에, 1 프레임)을 가로지르는 최대 비트율이 허용되는 압축 패러다임. 예컨대 도 6c의 경우에, 만약 CBR 제약이 제한된 비트율을 엔코딩하는 데 적용된다면, 예컨대 70% 의 정격된 최대 전송률(621), 다음으로 압축 알고리즘은 각각의 프레임의 압축을 제한할 것이고 그래서 정격된 최대 전송률(621)은 적은 비트로 압축될 것이다. 이것의 결과는 주어진 품질 수준을 유지하기 위해 일반적으로 보다 많은 비트를 요구하는 프레임은 비트의 "부족(starved)" 그리고 프레임들의 이미지 품질은 최대 전송률(621)의 70% 비율보다 더 많은 비트를 요구하지 않는 그 밖의 프레임보다 더 나빠질 것이다. 이러한 접근은 (a) 작은 움직임 또는 장면 변화가 예상되고 (b) 사용자가 주기적으로 품질 감소를 받아들일 수 있는 어떤 형태의 압축된 비디오를 위한 받아들일 수 있는 결과를 생산할 수 있다. CBR와 어울리는 어플리케이션의 좋은 예는 비디오 원격 회의인데 이는 거의 피크가 없기 때문이고, 만약 품질이 분명하게 감소한다면(예컨대 만약 카메라에 찍히면, 높은 품질 이미지 압축을 위해 충분하게 비트 되지 않으면서 찍히는 동안에, 이는 감소된 이미지 품질을 초래할 수 있다), 그것은 대부분 사용자에게 받아들여질 수 있다. 불행하게도, CBR은 높은 복잡성의 장면을 갖거나 많은 움직임 및/또는 거기에 합리적인 일정한 수준의 품질이 요구되는 그 밖의 많은 어플리케이션에는 잘 어울리지 않는다.

[0139] 일 실시예에서 채택된 낮은-레이턴시 압축 로직(404)은 낮은-레이턴시 압축된 비디오를 스트리밍하여 문제의 범위로 전달하기 위한 몇 개의 다른 기술을 사용한다. 먼저, 낮은-레이턴시 압축 로직(404)은 오직 I 프레임 및 P 프레임을 생성하고, 그것 때문에 각 B 프레임을 디코드하기 위한 몇 개의 프레임 시간을 기다리기 위한 필요를 완화한다. 더욱이, 도 7a에 도시된 것처럼 일 실시예에서, 낮은-레이턴시 압축 로직(404)은 각각의 압축되지 않은 프레임(701 - 760)을 일련의 "타일(tile)" 및 개별적으로 I 프레임 또는 P 프레임과 같은 각각의 타일로 엔코드한다. 압축된 I 프레임 및 P 프레임의 그룹은 "R 프레임"(711 - 770)으로 여기서 언급한다. 도 7a에 도시된 특정한 예에서, 각 압축되지 않은 프레임은 16 타일의 4×4 매트릭스로 세분된다. 그러나, 이러한 근본적인 원리는 어떤 특정한 세분 계획에 제한되지 않는다.

[0140] 일 실시예에서, 낮은-레이턴시 압축 로직(404)은 다수의 타일로 비디오 프레임이 나눠지고, I 프레임(예컨대, 그 타일은 마치 그것이 전체 이미지의 크기에 $1/16^{th}$ 의 개별 비디오 프레임이고, 이러한 "미니(mini)" 프레임을 위해 사용되는 압축은 I 프레임 압축)으로써 각 프레임으로부터 하나의 타일을 엔코드하며(예컨대, 압축한다), P 프레임(예컨대, 각각의 "미니" $1/16^{th}$ 프레임을 위해 사용되는 압축은 P 프레임 압축이다)으로써 나머지 타일을 엔코드한다. I 프레임 및 P 프레임으로써 압축된 타일은 대체적으로 "I 타일" 그리고 "P 타일"로 언급될 것이다. 각각의 연속적인 비디오 프레임으로, I 타일로써 엔코드된 타일은 변화한다. 따라서, 주어진 프레임 시간에, 비디오 프레임에 있는 타일 중에 오직 하나의 타일은 I 타일이고, 타일들 중 나머지는 P 타일이다. 예컨대, 도 7a에서, 압축되지 않은 프레임(701)의 타일 0은 I 타일 I_0로 엔코드되고 나머지 1 - 15 타일은 R 프레임(711)을 생성하기 위해서 P 타일 P_1 에서 P_{15}로써 엔코드된다. 다음 압축되지 않은 비디오 프레임(702)에서, 압축되지 않은 프레임(701)의 타일 1은 I 타일 I_1 로 엔코드되고 나머지 타일 0 및 타일 2에서 15는 R 프레임(712)을 생성하기 위해서 P 파일 P_0, P_2 에서 P_{15}로써 엔코드된다. 따라서, 타일을 위해 I 타일 및 P 타일은 연속적인 프레임에 걸쳐 계속해서 제때에 인터리브된다(interleaved). 그 과정은 R 타일(770)이 I 타일(예컨대,

I_{15})로써 엔코드된 매트릭스에 마지막 타일이 생성될 때까지 계속된다. 다음으로 그 과정은 프레임(711)처럼 다른 R 프레임을 생성하는 것을 다시 시작한다(예컨대, 타일 0을 위해 I 타일을 엔코딩). 비록 도 7a에 도시되지 않았지만, 일 실시예에서, R 프레임의 비디오 시퀀스의 제 1 R 프레임은 오직 I 타일(예컨대, 차후의 P 프레임은 동작을 연상하기 위해 그것으로부터 기준 프레임 데이터를 갖는다)을 포함한다. 택일적으로, 일 실시예에서, 스타트업 시퀀스는 일반적으로 동일한 I 타일 패턴을 사용하지만, I 타일로 아직 엔코드되지 않은 타일을 위해 P 타일을 포함하지 않는다. 다른 말로, 어떤 타일은 제 1 I 타일이 도착할 때까지 어떤 데이터로 엔코드되지 않고, 그 때문에 도 9a에 비디오 스트림 전송률(934)에서 스타트업 피크를 피하고, 이는 아래에서 상세히 설명한다. 더욱이, 아래에 설명된 것처럼, 이러한 기본적인 이론을 준수하면서 다른 크기 및 형태는 타일을 위해 사용될 수 있다.

[0141] 클라이언트(415)에서 동작하는 비디오 신장 로직(412)은 마치 그것은 I 및 P 프레임의 별개 비디오 시퀀스처럼 각 타일을 신장하고, 그 다음에 디스플레이 장치(422)를 구동하는 프레임 버퍼을 위해 각 타일을 랜더링한다. 예컨대, R 프레임(711)에서 프레임(770)까지 I_0 및 P_0은 타일 1 등을 재구성하기 위해서 사용된다. 위에서 언급한 것처럼, I 프레임 및 P 프레임의 신장은 분야에서 잘 알려져 있고, I 타일 및 P 타일의 신장은 클라이언트(415)에서 동작하는 비디오 신장의 다수의 인스턴스(instance)를 가짐으로써 달성될 수 있다. 비록 멀티동작 프로세스가 클라이언트(415)에서 컴퓨터적인 부담이 증가하는 것처럼 보이지만, 그것은 실제적으로 그렇지 않은데 왜냐하면 타일 자체는 많은 추가적인 처리에 관하여 비율적으로 더 작고, 따라서 디스플레이된 픽셀의 수는 하나의 처리가 있었다면 통상적인 전체 크기 I 및 P 프레임을 사용하는 것과 동일하다.

[0142] 이러한 R 프레임 기술은 도 6b 및 도 6c 에 도시된 I 프레임과 관련된 전형적인 대역폭 피크를 현저히 완화한다, 왜냐하면 어떤 주어진 프레임은 전형적으로 I 프레임보다 작은 P 프레임으로 대부분 구성되기 때문이다. 예컨대, 전형적인 I 프레임은 160Kb라고 가정하면, 도 7a에 도시된 프레임 각각의 I 타일은 대략 이 크기의 1/16 또는 10Kb일 것이다. 비슷하게, 전형적인 P 프레임은 16Kb라고 가정하면, 도 7a에서 도시된 타일의 각각을 위한 P 프레임은 대략 1Kb일 것이다. R프레임의 최종 결과는 대략 10Kb + 15*1Kb = 25Kb이다. 그래서, 각 60-프레임 시퀀스는 25Kb*60 = 1.5Mbps이다. 그래서 60프레임/초에서, 이는 1.5Mbps의 대역폭을 유지할 수 있는 채널 성능을 요구할 것이나, 60-프레임 간격 내내 분배되는 I 타일에 기인하여 훨씬 더 낮은 피크를 갖는 것을 요구할 것이다.

[0143] I 프레임 및 P 프레임을 위해 동일하게 가정된 전송률을 갖는 이전 예를 주목하면, 평균 전송률은 1.1Mbps이다. 이는 이전 예에서는, 새로운 I 프레임은 오직 매 60 프레임시간당 한번만 소개되었고, 반면에 본 예에서는, I 프레임 사이클을 구성하는 16 타일은 16 프레임 시간(times)에서 완료되기 때문이고, I 프레임의 이러한 등가가 매 16 프레임 시간(times)에 유도됨에 따라, 근소하게 더 높은 평균 전송률을 초래한다. 보다 빈번한 I 프레임이 유도되는 것은 선형적으로 전송률을 증가시키지 않는다. 이는 P 프레임(또는 P 타일)은 주로 선행 프레임에서 다음 프레임 사이 차이를 엔코드한다는 사실에 기인한다. 그래서, 만약 선행 프레임이 다음 프레임과 상당히 다르다면, P 프레임은 매우 커질 것이다. 그러나 P 프레임이 실제 프레임 보다는 대체로 선행 프레임으로부터 유도되기 때문에, 엔코드된 프레임 결과는 적당한 수의 비트를 갖는 I 프레임보다 많은 에러(예컨대, 시각적 아티팩트)를 포함할 것이다. 그리고, 하나의 P 프레임이 다른 P 프레임을 따를 때, 발생할 수 있는 것은 긴 시퀀스 P 프레임이 있는 경우 더 악화되는 에러의 축적이다. 지금, 정교한 비디오 압축기는 시퀀스의 P 프레임 후에 이미지의 품질이 떨어진다는 사실을 발견할 것이고, 만약 필요하다면, 그것은 품질을 개선하기 위해서 이후 P 프레임에 보다 많은 비트를 할당할 것이나, 만약, 그것은 가장 효율적인 코스의 동작이고, I 프레임으로 P 프레임을 대체한다. 그래서, 긴 시퀀스의 P 프레임이 사용될 때,(예컨대, 59프레임, 앞의 예에서처럼) 특히 장면이 많은 복잡성 및/또는 움직임을 가지고 있을 때, 전형적으로 보다 많은 비트는 I 프레임에서 더 많이 제거됨에 따라 P 프레임을 위해 필요해진다.

[0144] 또는, 반대 시점에서 P프레임을 보기 위해서, I 프레임을 바짝 따르는 P 프레임은 I 프레임으로부터 추가 제거된 P 프레임보다 적은 비트를 요구하는 경향이 있다. 그래서, 도 7a에서 도시된 예에서, P 프레임은 I 프레임에서 제거된 15 프레임보다 더 추가되지 않고, 앞의 예에서처럼, P 프레임은 그것을 앞서는 I 프레임으로부터 제거된 59 프레임일 수 있다. 따라서, I 프레임이 보다 빈번함으로, P 프레임은 더 작다. 물론, 정확한 관련된 크기는 비디오 스트림의 특성에 기초되어 변화할 것이나, 도 7a의 예에서 만약 I 타일이 10Kb, 대체로 0.75Kb 크기 P 타일은, 10Kb + 15*0.75Kb = 21.25Kb를 초래하고, 또는 초당 60 프레임에서, 전송률은 21.25Kb*60 = 1.3 Mbps 일 것이나, 또는 1.1Mbps에서 59 P 프레임에 의해 뒤따르는 I 프레임을 갖는 스트림보다 약 16% 높은 전송률이다. 다시 한번, 비디오 압축에 대한 이러한 2가지 접근 사이에 관련 결과는 비디오 시퀀스에 의존하여 변화할 것이지만, 그러나 전형적으로, 우리는 R-프레임 사용은 I/P 프레임 시퀀스를 사용하는 것보다 주어진 수준의

품질을 위해 약 20% 이상 비트를 요구한다는 것을 경험적으로 발견한다. 그러나, 물론 R 프레임은 I/P 프레임 시퀀스보다 훨씬 적은 레이턴시를 갖고 사용할 수 있는 비디오 시퀀스를 만드는 피크를 극단적으로 줄인다.

[0145] R 프레임은 비디오 시퀀스, 채널의 신뢰성, 이용가능한 전송률의 특성에 의존하는 다양한 다른 방법으로 구성될 수 있다. 택일적 실시예에서, 다른 많은 수의 타일은 4×4 구성에서 16 이상 사용된다. 예컨대, 2 타일은 2×1 또는 1×2에서 사용될 것이고, 4 타일은 2×2, 4×1, 또는 1×4 구성에서 사용되거나, 6 타일은 3×2, 2×3, 6×1, 또는 1×6 구성에서 사용되거나, 또는 8 타일은 4×2(도 7b에서 도시된 것처럼),2×4, 8×1, 또는 1×8 구성에서 사용될 것이다. 타일은 스퀘어(square)일 필요없고, 비디오 프레임이 스퀘어일 필요없거나 직사각형일 필요조차 없다. 타일은 비디오 스트림 및 어플리케이션에 사용되는 적합한 형태이면 무엇이든지 분할될 수 있다.

[0146] 다른 실시예에서, I 및 P 타일의 사이클링은 다수의 타일로 고정되지 않는다. 예컨대 4×2 구성 8 타일에서, 16 사이클 시퀀스는 도 7b에서 도시된 것처럼 사용될 수 있다. 순차적으로 일어나는 압축되지 않은 프레임(721,722,723)은 각각 0-7인 8 타일로 분할되고, 각 타일은 개별적으로 압축된다. R 프레임(731)에서, 오직 0 타일은 I 타일로써 압축되고, 나머지 타일은 P 타일로 압축된다. 다음의 R 프레임(732) 동안 모든 8 타일은 P 타일로써 압축되고, 그러고나서 다음의 R 프레임(733) 동안 1 타일은 I 타일로써 압축되고 그 밖의 타일은 모두 P 타일로써 압축된다. 그리고, 16 프레임 동안에 시퀀스가 연속되고, 오직 모든 그 밖의 프레임으로 생성된 I 프레임으로, 그래서 마지막 I 타일은 15^{th} 프레임 시간(도 7b에 도시안됨)동안에 7 타일을 위해서 생성되고 16^{th} 프레임 시간 동안에 R 프레임(780)이 모든 P 타일을 사용하여 압축된다. 그러고나서, 시퀀스는 다시 I 타일로써 압축된 0 타일로 시작하고 그 밖의 타일은 P 타일로써 압축된다. 이전 실시예에서처럼, 제일 첫번째 프레임의 전체 비디오 시퀀스는 그런 시점에서 P 타일을 위한 기준을 제공하기 위해서 전형적으로 모두 I 타일일 것이다. I 타일 및 P 타일의 사이클링은 복수 개의 타일이 필요하지 않다. 예컨대, 8 타일로, I 타일을 갖는 각 프레임은 또다른 I 타일이 사용되기 전까지 모두 P 타일로 2 프레임까지 뒤따를 수 있다. 이미 또다른 실시예에서, 어떤 타일은 그 밖의 타일보다 자주 I 타일로 시퀀스될 것이다 예컨대, 스크린의 어떤 영역은 빈번한 I 타일로부터 요구되는 것보다 더 많은 동작을 가진 것으로 알려졌다, 반면 그 밖의 영역은 빈번한 I 타일로부터 보다 적게 요구되는 보다 정적이다(예컨대, 게임을 위한 스코어를 보이는 것). 더욱이, 비록 각 프레임은 도7a, 7b에서 단일 I 타일로 도시되었지만, 복수의 I 타일은 단일 프레임에 엔코드될 것이다(전송 채널의 대역폭에 의존하는). 반대로, 어떤 프레임 또는 프레임 시퀀스는 I 타일을 갖지 않고 전송될 수 있을 것이다(예컨대, 오직 P 타일).

[0147] 앞 단락 연구 접근의 좋은 이유는 모든 단일 프레임에 걸쳐 분포된 I 타일을 갖고 있지 않으면서 더 큰 피크를 초래하는 것으로 보이고, 시스템의 동작은 단순하지 않다는 것이다. 각 타일이 다른 타일과 분리되어 압축되고, 타일은 각 타일의 엔코딩이 작을수록 덜 효율적이 될 수 있는데, 왜냐하면 주어진 타일의 압축기는 다른 타일로부터 비슷한 이미지 특징 및 비슷한 동작을 이용할 수 없기 때문이다. 따라서, 일반적으로 16 타일로 화면을 분할하는 것은 8 타일로 화면을 분할하는 것보다 덜 효율적인 엔코딩이 될 것이다. 그러나, 만약 스크린이 8 타일되면 그것은 매 16 프레임 대신에 매 8 프레임으로 전체 I 프레임의 데이터가 유도되도록 하고, 그것은 총체적으로 매우 높은 전송률을 초래한다. 그래서, 전체 I 프레임을 매 8 프레임 대신에 매 16 프레임으로 유도함으로써, 전체 전송률은 감소된다. 또한, 16개 작은 타일을 사용하는 대신에 8개 더 큰 타일을 사용함으로써, 전체 전송률은 감소하게 되고, 이는 또한 큰 타일에 의해서 초래된 데이터 피크를 어느 정도까지 완화할 것이다.

[0148] 다른 실시예에서, 도 7a 및 7b에서 낮은-레이턴시 비디오 압축 로직(404)은 압축되기 위해서 비디오 시퀀스의 이미 알려진 특징에 기초한 셋팅에 의하여 선행 구성되거나, 자동적으로 진행중인 각 타일에서 이미지 품질의 분석에 기초하여 R 프레임에 다양한 타일을 위해 비트의 할당을 제어한다. 예컨대, 어떤 레이싱 비디오 게임에서 플레이어의 자동차 전면부(이는 대체로 장면에서 동작이 없음)는 스크린의 아래쪽 절반의 큰 영역을 차지하고, 반면에 스크린의 윗쪽 절반은 전체적으로 접근하는 도로, 빌딩, 풍경으로 채워지고, 이는 거의 항상 움직이고 있다. 만약 압축 로직(404)이 각 타일로 동일한 수의 비트를 할당하면, 그 다음 도 7b에 압축되지 않은 프레임(721)에서 스크린의 바닥 절반에 타일(타일 4 - 7)은, 일반적으로 도 7b에 압축되지 않은 프레임(721)에서 스크린 상부 절반에 타일(타일 0 - 3)보다 높은 품질로 압축될 것이다. 만약 이러한 특정 게임, 또는 게임의 특정 장면은 그러한 특징을 갖는다고 알려지면, 호스팅 서비스(210)의 오퍼레이터는 스크린의 하부에 타일보다 스크린의 상부에 타일을 위해 보다 많은 비트를 할당하기 위해서 압축 로직(404)을 구성한다. 또는, 압축 로직(404)은 프레임이 압축된 후에 타일의 압축의 품질을 측정할 수 있고(Peak-To-Noise Ratio(PSNR)과 같은 하나 또는 그 이상의 많은 압축 품질 성과지표를 사용함), 만약 그것은 어떤 특정 윈도우의 시간을 넘겨 결정된다면, 어떤 타일들은 일관되게 더 나은 품질 결과를 생성할 것이고, 다양한 타일이 비슷한 수준의 품질에 도달할 때까지 그

것은 점차적으로 보다 많은 비트를 낮은 품질 결과를 생성하는 타일에 할당한다. 택일적 실시예에서, 압축기 로직(404)은 특정 타일 또는 타일 그룹에 높은 품질을 달성하기 위해서 비트를 할당한다. 예컨대, 가장자리 보다 스크린의 중심에서 높은 품질을 갖도록 전체적으로 더 나은 지각있는 외관을 제공할 것이다.

[0149] 일 실시예에서, 비디오 스트림의 어떤 영역의 해상도를 향상하기 위해서, 비디오 압축 로직(404)은 대체적으로 보다 적은 복잡한 장면 및/또는 움직임을 갖는 비디오 스트림의 영역보다 보다 복잡한 장면 및/또는 움직임을 갖는 비디오 스트림의 영역을 엔코드하기 위해서 더 작은 타일을 사용한다. 예컨대, 도 8에 도시된 것처럼, 더 작은 타일은 R 프레임(811)의 한 영역에서 움직이는 캐릭터(805) 주변에 채택된다(잠재적으로 동일한 타일 사이즈를 갖고 일련의 R 프레임에 뒤따르게 된다(도시되지 않음)). 그 다음 캐릭터(805)가 이미지의 새로운 영역으로 움직일 때, 도시된 것처럼 더 작은 타일은 또 다른 R 프레임(812) 내에 이 새로운 영역 주변에서 사용된다. 위에서 언급한 것처럼, 이러한 근본적인 이론에 여전히 따르는 동안에는 다양한 다른 크기 및 형태는 "타일"로써 채택될 것이다.

[0150] 위에서 설명된 주기적인 I/P 타일이 본질적으로 비디오 스트림의 전송률에 피크를 줄여주는 동안에, 그들은 동영상(motion picture), 비디오 게임, 및 어떤 어플리케이션 소프트웨어로 발생되는 것처럼, 특히 빠르게 변화하거나 매우 복잡한 비디오 이미지의 경우에, 전체적으로 피크를 제거하지 않는다. 예컨대, 갑자기 장면 전환하는 동안에, 복잡한 프레임은 완전히 상이한 다른 복잡한 프레임이 뒤따를 것이다. 몇몇의 I 타일이 오직 몇 프레임 시간까지 장면 전환 전에 있을 수 있을지라도, 그들은 이러한 상황에 도움을 줄 수 없는데 왜냐하면 새로운 프레임의 재료가 사전 I 타일과 관련이 없기 때문이다. 그러한 상황에서(그리고 모든 것이 변화하는 것은 아닐지라도 많은 이미지 변화하는 그 밖의 상황에서), 비디오 압축기(404)는 I 타일처럼 보다 효율적으로 코드된 P 타일의 전부가 아니라면 얼마 정도인지, 그리고 어떤 결과가 그 프레임에 대한 전송률에서 매우 큰 피크인지를 결정할 것이다.

[0151] 앞에서 논의한 바와 같이, 대부분 소비자 등급 인터넷 연결(그리고 많은 오피스 연결)의 경우는 단순하고, 그것은 단순히 정격된 최대 전송률(621)에 따라, 도 6c에서 622로서 도시된 이용가능한 최대 전송률을 초과하는 "잼(jam)" 데이터로는 가능하지 않다. 정격된 최대 전송률(621)(예컨대, "6Mbps DSL")은 본질적으로 인터넷 연결의 구입을 고려하는 사용자에게 마켓팅 숫자(marketing number)이지만, 일반적으로 그것은 어떤 수준의 성능을 보장하지 않는다. 이러한 어플리케이션의 목적을 위해서, 그것은 무의미하고, 우리의 오직 관심은 비디오가 연결을 통해서 스트림되는 시간에 이용가능한 최대 전송률(622)이다. 결과적으로, 도 9a 및 9c에서, 우리가 피킹 문제(peaking problem)를 위한 해결책을 설명할 때, 정격된 최대 전송률은 그래프에서 생략되어 있고, 오직 이용가능한 최대 전송률(922)가 도시되어 있다. 비디오 스트림 전송률은 이용가능한 최대 전송률(922)을 초과하지 않아야 한다.

[0152] 이를 처리하기 위해서, 비디오 압축기(404)가 해야 하는 첫번째는 피크 전송률(941)을 결정하는 것이고, 이는 채널이 지속적으로 다룰 수 있는 전송률이다. 이 속도는 많은 기술에 의해서 결정된다. 그런 기술 중 하나는 호스팅 서비스(210)에서 도 4a 및 4b에서 클라이언트(415)로 증가하는 높은 전송률 테스트 스트림을 점차적으로 보낼 것이고, 클라이언트가 호스팅 서비스로 어떤 수준의 패킷 손실 및 레이턴시에 관하여 피드백을 제공할 것이다. 패킷 손실 및/또는 레이턴시는 날카로운 증가를 보이기 시작할 때, 그것은 이용가능한 최대 전송률(922)에 도달하고 있다는 표시이다. 그 후에, 호스팅 서비스(210)은 점차적으로 클라이언트(415)가 합리적인 주기의 시간 동안 테스트 스트림이 허용가능한 수준의 패킷 손실과 레이턴시가 거의 최소로 수신되는 때까지 테스트 스트림의 전송률을 줄일 수 있다. 시간이 지나면, 피크 전송률(941)이 변동하게 되고(예컨대, 만약 가정 내에 다른 사용자가 과도하게 인터넷 연결을 사용한다면), 클라이언트(415)는 패킷 손실 또는 레이턴시 증가의 여부, 이용가능한 최대 전송률(922)이 미리 확립된 피크 전송률(941) 이하로 떨어지는지의 여부, 그리고 그렇다면 피크 전송률(941)인가의 여부를 알도록 그를 끊임없이 모니터하는 것이 필요로 된다. 유사하게, 만약 시간이 지나고 클라이언트(415)가 패킷 손실 및 레이턴시가 최적의 수준에 남아있다는 것을 발견하면, 그것은 비디오 압축기가 천천히 이용가능한 최대 전송률이 증가되었는지를 보기 위해서 전송률을 증가하도록 요청할 수 있고(예컨대, 만약 가정 내에 다른 사용자가 과도한 인터넷 연결의 사용을 멈춘다면), 다시 패킷 손실 및/또는 높은 레이턴시가 이용가능한 전송률(922)이 초과되었다고 지시할 때까지 기다리며, 다시 낮은 수준은 피크 전송률(941)을 위해 찾게 되지만, 하나는 아마도 증가된 전송률을 테스트하기 전에 그 수준보다 더 높아진다. 그래서, 이러한 기술을 사용함으로써(그리고 이와같은 다른 기술) 피크 전송률(941)은 찾을 수 있고, 필요했던 것처럼 주기적으로 조정된다. 피크 전송률(941)은 사용자에게 비디오를 스트림하기 위해 비디오 압축기(404)에 의해서 사용될 수 있는 최대 전송률로 설정될 것이다. 피크 전송률을 결정하기 위한 로직은 사용자 구내(211) 및/또는 호스팅 서비스(210)에서 실행될 것이다. 사용자 구내(211)에서 클라이언트(415)는 피크 전송률을 결정하기 위한 연산을

수행하고 이 정보를 호스팅 서비스(210)로 되돌려 전송하고; 호스팅 서비스(210)에서, 호스팅 서비스에 서버(402)는 클라이언트(415)로부터 수신한 통계치(예컨대, 패킷 손실, 레이턴시, 최대 전송률 등)를 기초하여 피크 전송률을 결정하기 위한 연산을 수행한다.

[0153] 도 9a는 도 7a, 7b, 및 도 8에서 도시되고 사전에 설명된 순환적인 I/P 타일 압축 기술을 사용하여 생성된 실질적으로 복잡한 장면 및/또는 동작을 가지는 비디오 스트림 전송률(934)의 예를 도시하고 있다. 비디오 압축기(404)는 피크 전송률(941) 아래의 평균 전송률에서 압축된 비디오를 출력하기 위해서 구성되고, 대부분의 시간에, 비디오 스트림 데이터는 피크 전송률(941) 아래에 머무른다. 전송률(934)과 I/P/B 또는 I/P 프레임을 사용하여 만들어진 도 6c에 도시된 비디오 스트림 속도(634)의 비교는 순환적인 I/P 타일 압축이 보다 부드러운 전송률을 생산한다는 것을 보인다. 여전히, 프레임 2X 피크(952)(이는 2X 피크 전송률(942)에 근접한다) 그리고 프레임 4X 피크(954)(이는 4X 피크 전송률(944)에 근접한다), 그 전송률은 피크 전송률(941)을 초과하고, 이는 허용될 수 없다. 실제로는 빠르게 변하는 비디오 게임으로부터 높은 동영상으로, 피크 전송률(941)을 초과하는 피크는 2%의 프레임보다 적게 일어나고, 2X 피크 전송률(942)을 초과하는 피크는 거의 드물게 일어나며, 3X 피크 전송률(943)을 초과하는 피크는 거의 일어나지 않는다. 그러나, 그들이 일어날 때(예컨대, 장면 전환 동안), 그것들에 의해서 요구되는 전송률은 좋은 품질 비디오 이미지를 생산하기 위해서 필요하다.

[0154] 이러한 문제를 해결하기 위한 하나의 방법은 비디오 압축기(404)를 설치함으로서 그것의 최대 전송률 출력은 피크 전송률(941)이 된다. 불행하게도, 피크 프레임 동안에 결과 비디오 출력 품질은 형편없는데 왜냐하면 압축 알고리즘이 비트를 위해 "부족"하기 때문이다. 갑작스런 전환 또는 빠른 움직임이 있을 때 압축 아티팩트의 출현의 결과, 그때에, 사용자가 갑작스런 변화 또는 빠른 동작이 있는 때 항상 갑자기 생기는 아티팩트를 알아차리고, 그것들은 꽤 성가시게 될 수 있다.

[0155] 비록 인간 시각 시스템이 갑자기 변화하거나 빠른 움직임 동안에 나타나는 시각적 아티팩트에 꽤 민감하고, 그것은 그러한 상황에서 프레임의 속도에 감소를 검측하는데 매우 민감하지는 않다. 사실, 그러한 갑작스러운 변화가 일어날 때, 그것은 인간 시각 시스템이 변화를 추적하면서 미리 일어나고, 그것은 만약 프레임 속도가 분명하게 60fps에서 30fps로 떨어지고, 그 다음 즉시로 60fps 되돌아온다면 알아차리지 못한다. 그리고, 갑작스런 장면 변화와 같은 매우 극적인 변환의 경우에, 인간 시각 시스템은 프레임 속도가 20fps 또는 15fps까지 떨어지고, 그 다음 즉시로 60fps로 되돌아온다면, 이를 알아차리지 못한다. 프레임 속도 변화가 오직 빈번하지 않게 일어나는 동안에, 인간 관찰자에게, 그것은 비디오가 60fps에서 연속적으로 작동하고 있는 것으로 나타낸다.

[0156] 이러한 인간 시각 시스템의 특징은 도 9b에서 도시된 기술에 의해서 이용된다. 서버(402)(도 4a 및 4b에서)는 압축되지 않은 비디오 출력 스트림을 꾸준한 프레임 속도(일 실시예에서 60fps)로 출력한다. 타임라인은 각 프레임(961 - 970)의 $1/60^{th}$ 초당 출력을 보인다. 프레임(961)으로 시작하고, 각각의 압축되지 않은 비디오 프레임은, 제 1 프레임을 위해 압축된 프레임 1(981)을 생성하기 위해서, 한 프레임 시간보다 적은 시간에 프레임을 압축하는 낮은-레이턴시 비디오 압축기(404)로 출력된다. 압축된 프레임 1(981)을 위해 생산된 데이터는 앞서 설명된 것처럼, 많은 요소에 의존하여 더 크거나 더 작을 수 있을 것이다. 만약 데이터가 충분히 작다면 한 프레임 시간($1/60^{th}$초) 또는 피크 전송률(941)보다 적은 시간에 클라이언트(415)로 전송될 수 있고, 그 다음에 전송 시간(Xmit 시간)(991)(화살표의 길이는 전송 시간의 지속시간을 나타낸다). 다음 프레임 시간에, 서버(402)는 압축되지 않은 프레임 2(962)를 생산하고, 그것은 압축된 프레임 2(982)로 압축되며, 그것은 전송 시간(992) 동안에 클라이언트(415)로 전송되고, 이는 피크 전송률(941)의 프레임 시간 보다 작다.

[0157] 그 다음으로, 다음 프레임 시간에서, 서버(402)는 압축되지 않은 프레임 3(963)을 생성한다. 그것은 비디오 압축기(404)에 의해서 압축된 때, 결과 압축된 프레임 3(983)은 하나의 프레임 시간에 피크 전송률(941)로 전송될 수 있는 것보다 보다 큰 데이터이다. 그래서, 그것은 전송 시간(2X 피크)(993) 동안에 전송되고, 이는 모든 프레임 시간 및 다음 프레임 시간의 일부를 차지한다. 지금, 다음 프레임 시간 동안에, 서버(402)는 다른 압축되지 않은 프레임 4(964)를 생성하고 그것을 비디오 압축기(404)로 출력하지만 그 데이터는 무시되고, 974로 예시된다. 이는 왜냐하면 비디오 압축기(404)가 아직 이전 압축된 프레임을 전송하는 동안에 도착된 추가된 압축되지 않은 비디오 프레임을 무시하도록 구성된다. 물론 클라이언트(415)의 비디오 신장기는 프레임 4를 수신하는 것에 실패하지만, 그것은 단순히 디스플레이 장치(422)에 2 프레임 시간 동안에 프레임 3을 디스플레이하는 것을 계속한다(예컨대, 간략하게 60fps에서 30fps로 프레임 속도가 감소된다).

[0158] 다음 프레임 5에 대해, 서버(402)는 압축된 프레임 5(985)로 압축되는 압축되지 않은 프레임 5(965)을 출력하고 전송 시간(995) 동안에 1 프레임 내에 전송된다. 클라이언트(415)의 비디오 신장기는 프레임 5를 신장하고 디스플레이 장치(422)에 그것을 디스플레이한다. 다음으로, 서버(402)는 비디오 압축기(404)에 의해서 압축된 프레

임 6(986)으로 압축되는 압축되지 않은 프레임 6(966)을 출력하지만, 이 시간에 결과 데이터는 매우 크다. 압축된 프레임은 피크 전송률(941)에서 전송 시간(4X 피크)(996) 동안에 전송될 것이지만, 그것은 프레임을 전송하기 위해서는 거의 4 프레임 시간이 걸린다. 다음 3 프레임 시간 동안에, 비디오 압축기(404)는 서버(402)로부터 3개 프레임을 무시하고, 클라이언트(415)의 신장기는 4개 프레임의 시간 동안 디스플레이 장치(422)에 프레임 6을 유지한다(예컨대, 분명하게는 프레임 속도를 60fps에서 15fps로 줄인다). 그 다음 최종적으로, 서버(402)는 프레임 10(970)을 출력하고, 비디오 압축기(404)는 그것을 압축된 프레임 10(987)으로 압축하며, 그것은 전송 시간(997) 동안에 전송되고, 클라이언트(415)의 신장기가 프레임 10을 신장하고 그것을 디스플레이 장치(422)로 디스플레이하고 다시 한번 그 비디오는 60fps에서 다시 시작한다.

[0159] 비록 비디오 압축기(404)가 서버(402)에 의해서 생성된 비디오 스트림에서 비디오 프레임으로 떨어지더라도, 들어오는 오디오 형태가 무엇인지 상관없이, 그것은 오디오 데이터가 떨어지지 않고, 비디오 프레임이 떨어지고 그것을 클라이언트(415)로 전송된 때, 오디오 데이터를 압축하는 것을 계속하며, 이는 오디오 데이터를 신장하는 것을 계속하고 오디오를 재생하기 위해 사용자에 의해서 사용되는 어떤 장치로 오디오를 제공한다. 따라서, 오디오는 프레임이 떨어지는 주기 동안에도 줄지않고 계속된다. 압축된 비디오와 비교했을 때, 압축된 오디오는 대체로 작은 비율의 대역폭을 소모하고, 결과적으로 전체 전송률에 주된 영향을 갖지 않는다. 비록 그것은 어떤 전송률 다이어그램에 도시되지 않았지만, 피크 전송률(941)내에 압축된 오디오 스트림을 위해 제한된 전송률 성능 항상이 있다.

[0160] 도 9b에 도시된 예는 전송률 피크 동안에 프레임 속도가 얼마나 떨어지는 지를 도시하도록 선택되지만, 전에 설명된 것처럼 순환적인 I/P 타일 기술이 사용되는 때, 전송률 피크와 같은, 비디오 게임, 동영상, 및 어떤 어플리케이션 소프트웨어에서 일어나는 것과 같은 높은 복잡한 장면/높은 움직임 시퀀스에서 조차 연속적으로 떨어지는 프레임은 드물다. 결과적으로, 줄어든 프레임 속도는 빈번하지 않고 분명하게 인간 시각 시스템은 그것들을 감지하지 못한다.

[0161] 만약 프레임 속도 감소 메커니즘은 도 9a에서 설명된 비디오 스트림 전송률에 적용되도록 설명된다면, 결과 비디오 스트림 전송률은 도 9c에 도시된다. 이러한 예에서, 2x 피크(952)는 플랫된 2x 피크(953)로 줄어들게 되고, 4x 피크(955)는 플랫된 4x 피크(955)로 줄어들게 되어, 전체 비디오 스트림 데이터(934)는 피크 전송률(941)로 남거나 그 아래로 남는다.

[0162] 따라서, 앞서 설명된 기술을 사용하면, 높은 동작 비디오 스트림은 낮은-레이턴시로 일반적인 인터넷 및 소비자 등급 인터넷 연결을 통해서 전송할 수 있다. 더욱이, 오피스 환경, LAN(예컨대, 100Mbs 이더넷 또는 802.11g 무선) 또는 개인 네트워크(예컨대, 데이터 센터와 오피스 사이의 100Mbps 연결)에서, 높은 동작 비디오 스트림은 피크 없이 전송될 수 있으므로 다수의 사용자(예컨대, 4.5Mbps에서 60fps로 1920×1080을 전송하는 것)는 압도하는 네트워크 또는 네트워크 스위치 백플레이트, 겹치는 피크를 갖지 않고 공유된 개인 데이터 연결을 사용할 수 있다

[0163] **<u>전송률 조정(Data Rate Adjustment)</u>**

[0164] 일 실시예에서, 호스팅 서비스(210)는 처음 비디오 스트림을 위해 적절한 전송률을 결정하기 위해서 이용가능한 최대 전송률(622) 및 채널의 레이턴시를 평가하고 그에 응답하여 전송률을 동적으로 조정한다. 전송률을 조정하기 위해서, 호스팅 서비스(210)는 예컨대 이미지 해상도 그리고/또는 클라이언트(415)로 보내지는 비디오 스트림의 다수의 프레임/초를 수정할 것이다. 또한, 호스팅 서비스는 압축된 비디오의 품질 수준을 조정할 수 있다. 비디오 스트림의 해상도를 바꾸는 것은, 예컨대 1280×720 해상도에서 640×360으로, 클라이언트(415)에 있는 비디오 신장 로직(412)은 디스플레이 스크린에 동일한 이미지 크기를 유지하기 위해 이미지를 일정비율로 확대할 수 있다.

[0165] 일 실시예에서, 채널이 완전히 빠지는 어떤 상황에서, 호스팅 서비스(210)는 게임을 중단한다. 멀티플레이어 게임의 경우에, 호스팅 서비스는 그 밖의 사용자에게 사용자가 게임을 떠났는지 및/또는 그 밖의 사용자를 위해 게임을 중단하였는지를 보고한다.

[0166] **<u>떨어진 또는 지연된 패킷(Dropped or Delayed Packets)</u>**

[0167] 일 실시예에서, 만약 도 4a 또는 4b에서 비디오 압축기(404) 및 클라이언트(415) 사이에서 패킷 손실에 기인하

거나 압축되지 않은 프레임의 레이턴시 요구와 만나서 신장을 위해서 너무 늦게 도착한 명령의 순서로 수신된 패킷에 기인하여 데이터가 손실되면, 비디오 신장 로직(412)은 시각적 아티팩트를 완화할 수 있다. I/P 프레임 실행 스트리밍에서, 만약 손실/지연된 패킷이 있는 경우, 전체 스크린이 영향을 받고, 잠재적으로 한 주기의 시간 동안 완전히 화면이 멈추는 것을 초래하거나 그 밖의 스크린-와이드 시각적 아티팩트를 보인다. 예컨대, 만약 손실/지연된 패킷은 I 프레임의 손실에 원인이 되고, 그 다음에 신장기는 새로운 I 프레임이 수신될 때까지 따르는 모든 P 프레임을 위한 기준이 부족할 것이다. 만약 P 프레임이 손실되면, 그 다음에 그것을 뒤따르는 전체 스크린을 위해 P 프레임에 영향을 줄 것이다. I 프레임이 나타나기 전 얼마나 오랫동안 그것이 걸릴 것인지에 따라, 이는 더 길거나 혹은 더 짧은 시각적 영향을 가질 것이다. 도 7a 및 7b에 도시된 것처럼 인터리브된 I/P 타일을 사용하면, 그것은 오직 영향을 받은 패킷에 포함된 타일에만 영향을 줄 것이기 때문에 손실/지연된 패킷은 전체 스크린에 훨씬 적은 영향을 준다. 만약 각각의 타일의 데이터가 개별적인 패킷으로 보내진다면, 그 다음에 패킷이 손실되면, 그것은 오직 하나의 타일에 영향을 줄 것이다. 물론, 시각적 아티팩트의 지속기간은 I 타일 패킷이 손실되었는지에 의존할 것이고, 만약 P 타일이 손실되면, 얼마나 많은 프레임이 I 타일이 나타날 때까지 걸릴지에 의존한다. 그러나, 화면에서 다른 타일로 주어진 것은 매우 빈번하게 I 프레임으로 업데이트되고(잠재적으로는 매 프레임당), 화면에 하나의 타일이 영향을 받았을 때조차, 그 밖의 타일은 아닐지 모른다. 더욱이, 만약 어떤 이벤트가 즉시 몇개의 패킷의 손실을 초래한다면(예컨대, 데이터 흐름을 명백히 방해하는 DSL 라인 옆에 전력 급등(spike on power)), 그 다음 어떤 타일은 그 밖의 것보다 더 영향을 미칠 것이고, 그러나 왜냐하면 어떤 타일이 빠르게 새로운 I 타일로 업데이트될 것이고, 그들은 오직 간단한 영향을 받을 것이다. 또한, 스트리밍 I/P 프레임 실행으로, I 프레임은 가장 결정적인 프레임이지만, I 프레임은 매우 크므로, 그래서 만약 거기에 떨어진/지연된 패킷을 초래하는 이벤트가 있다면, 아마도 보다 작은 I 타일보다는 I 프레임이 영향을 받게 될 것이다(예컨대, 만약 I 프레임 중 어떤 부분이 손실되면, 그것은 I 프레임이 전혀 신장될 수 없다). 모든 이러한 이유때문에, I/P 타일을 사용하는 것이 I/P 프레임으로 패킷이 떨어진/지연된 때보다, 훨씬 적은 시각적 아티팩트를 초래한다.

[0168] 일 실시예는 TCP(transmission control protocol) 패킷 또는 UDP(user datagram protocol) 패킷에서 압축된 타일을 지능적으로 패키징함으로써 손실 패킷의 효과를 줄이려는 시도를 한다. 예컨대, 일 실시예에서, 타일은 가능하면 언제든지 패킷 경계와 정렬된다. 도 10a은 어떻게 타일이 이러한 특징을 실행하지 않고 일련의 패킷들(1001 - 1005)내에서 포장될 것인지를 도시하고 있다. 특히, 도 10a에서, 타일은 패킷 경계를 가로지르고 비효율적으로 패킹되어서 단일 패킷의 손실은 다수 프레임의 손실을 초래한다. 예컨대, 만약 패킷(1003 또는 1004)이 손실되면, 3개 타일이 손실되고, 이는 시각적 아티팩트를 초래한다.

[0169] 대조적으로, 도 10b는 패킷 손실의 효과를 줄이기 위해서 패킷 내에 지능적으로 타일을 포함하기 위한 타일 패킹 로직(1010)을 도시하고 있다. 첫번째, 타일 패킹 로직(1010)은 패킷 경계로 타일을 정렬한다. 따라서, 타일 T1,T3,T4,T7 및 T2는 대체적으로 패킷들(1001 - 1005)의 경계에서 정렬된다. 또한 타일 패킹 로직은 패킷 경계를 가로지르는 것이 없고, 가장 효율적인 가능한 방식으로 패킷 내에 타일을 고정하도록 시도한다. 각각의 타일의 크기에 기초하여, 타일 T1 및 T6는 하나의 패킷(1001)에 결합되고; 타일 T3 및 T5는 하나의 패킷(1002)에 결합되며; 타일 T4 및 T8은 하나의 패킷(1003)에 결합되고; 타일 T8는 패킷(1004)에 추가되며; 그리고 타일 T2는 패킷(1005)에 더해진다. 따라서, 이러한 구조 하에서, 단일 패킷 손실은 2 타일 이상의 손실을 초래하지 않을 것이다(도 10a에서 도시된 것처럼 3 타일 보다는).

[0170] 도 10b에 도시된 실시예에 추가적인 이점은 타일은 이미지 내에 그것들이 디스플레이되는 것과는 다른 차수(order)로 전송된다. 이 방식은, 만약 인접한 패킷이 전송되어서 간섭하는 동일한 이벤트로 손실되면, 그것은 화면에 서로 가까이 있지 않은 영역에 영향을 미칠 것이고, 디스플레이에서 거의 알 수 없는 아티팩팅을 생성한다.

[0171] 일 실시예는 채널 에러로부터 비디오 스트림의 어떤 부분을 보호하기 위해서 순방향 에러 보정(FEC) 기술을 채택한다. 종래에 알려진 것처럼, Reed-Solomon 및 Viterbi와 같은 FEC 기술은 통신 채널을 거쳐 전송된 데이터를 위해 에러 보정 데이터 정보를 생성하고 첨부한다. 만약 근본적인 데이터에서 에러가 발생하면(예컨대, I 프레임), 그 다음 FEC는 그 에러를 보정하게 될 것이다.

[0172] FEC 코드는 전송의 전송률을 증가하게 하고; 그래서 이상적으로, 그들은 오직 가장 필요한 곳에서 사용된다. 만약 데이터가 매우 인식가능한 시각적 아티팩트를 초래하지 않도록 보내진다면, 데이터를 보호하기 위해서 FEC 코드를 사용하지 않는 것이 바람직할 것이다. 예컨대 P 타일은 그것이 손실되는 경우에 스크린에서 $1/60^{th}$ 초 동안에 오직 시각의 아티팩트(예컨대, 스크린의 타일에서 업데이트 되지 않을 것이다)를 생성하는 I 타일을 즉시

앞서게 된다. 그러한 시각의 아티팩트는 거의 인간의 눈으로 감지되지 않는다. P 타일이 I 타일보다 더 뒤에 있음으로서, P 타일을 잃는 것은 점진적으로 보다 현저하게 된다. 예컨대, 만약 타일 사이클 패턴이 I 타일이 다시 이용가능해지기 전에 15 P 타일까지 I 타일이 뒤따르게 된다면, 그 다음 즉시 I 타일을 뒤따르는 P 타일이 손실되면, 그것은 15 프레임 시간동안(60fps에서 그것은 250ms 일 것이다) 잘못된 이미지를 보이는 타일을 초래하게 될 것이다. 인간의 눈은 250ms 동안 스트림의 혼란을 쉽게 감지할 수 있을 것이다. 그래서, P 타일이 새로운 I 타일로부터 더 뒤쪽에 있을수록(예컨대, 더 가까이 P 타일이 I 타일을 뒤따른다), 아티팩트를 보다 잘 인식할 수 있다. 앞서 논의한 것처럼, 일반적으로, P 타일이 I 타일을 더 가까이 뒤따를수록, P 타일을 위한 데이터는 보다 작아진다. 따라서, I 타일을 뒤따르는 P 타일은 손실되는 것을 보호하기 위해 보다 임계적(critical)이지 않고, 그들은 크기가 더 작아진다. 그리고 일반적으로, 보호가 필요한 데이터가 더 작을수록, 더 작은 FEC 코드가 그것을 보호하기 위해서 요구된다.

[0173] 그래서, 도 11a에 도시된 것처럼, 일 실시예의 비디오 스트림에서 I 타일의 중요성 때문에, 오직 I 타일은 FEC 코드로 제공된다. 따라서, FEC(1101)는 I 타일(1100)을 위해 에러 보정 코드를 포함하고, FEC(1104)는 I 타일(1103)을 위한 에러 보정 코드를 포함한다. 이러한 실시예에서, P 타일을 위해 생성되는 FEC는 없다.

[0174] 도 11b에서 도시된 일 실시예에서, 또한 만약 손실되면 FEC 코드가 시각의 아티팩트를 초래하기가 쉬운 P 타일을 위해 생성된다. 이러한 실시예에서, FEC(1105)는 첫번째 3 타일을 위해 에러 보정 코드를 제공하지만, 뒤따르는 P 타일을 위해서 제공하는 것은 아니다. 다른 실시예에서, FEC 코드는 데이터 크기가 가장 작은 P 타일을 위해 생성된다(이는 I 타일 후에 가장 곧바로 발생하는 P 타일을 자체 선택(self-select)하는 경향이 있고, 보호를 위해 가장 임계적이다).

[0175] 다른 실시예에서, 어떤 타일과 FEC 코드를 보내는 것보다는 다른 패킷에 개별 시간에서 타일을 두 번 전송한다. 만약 하나의 패킷이 손실/지연되면, 다른 패킷을 사용한다.

[0176] 도 11c에 도시된 일 실시예에서, FEC 코드(1111 및 1113)는 대체적으로 비디오와 동시에 호스팅 서비스로부터 전송되는 오디오 패킷(1110 및 1112)을 생성한다. 그것은 비디오 스트림에서 오디오의 완결성(integrity)을 유지하기 위해서 특히 중요한데 왜냐하면 왜곡된 오디오(예컨대, 클리킹(clicking) 또는 히싱(hissing))는 특히 즐겁지않은 사용자의 경험을 초래할 것이다. FEC 코드는 오디오 컨텐츠가 왜곡 없이 클라이언트 컴퓨터(415)에서 랜더링되는 것을 보장하는 것을 돕는다.

[0177] 다른 실시예에서, 오디오 데이터와 FEC 코드를 보내기보다는, 오디오 데이터를 다른 패킷의 각각 시간에 2번 전송한다. 만약 하나의 패킷이 손실/지연되면, 다른 패킷이 이용된다.

[0178] 부가적으로, 도 11d에 도시된 일 실시예에서, FEC 코드(1121 및 1123)는 대체적으로(예컨대, 버튼 누름) 클라이언트(415)에서 호스팅 서비스(210)로 업스트림 전송되는 사용자 입력 명령어(1120 및 1122)을 위해서 사용된다. 비디오 게임 또는 어플리케이션에서 버튼 누름 또는 마우스 움직임을 놓치는 것은 사용자에게 즐겁지 않은 경험을 초래할 수 있기 때문에 이것은 중요하다.

[0179] 다른 실시예에서, 사용자 입력 명령어 데이터와 FEC 코드를 전송하는 것보다는, 사용자 입력 명령 데이터는 다른 패킷에서 각각의 시간에서, 2번 전송된다. 만약 하나의 패킷이 손실/지연되면, 다른 패킷을 사용한다.

[0180] 일 실시예에서, 호스팅 서비스(210)는 FEC 코드를 사용할지를 결정하기 위해서 클라이언트(415)와 통신 채널의 품질을 평가하고, 만약 FEC 코드를 사용한다면, FEC가 적용될 곳이 비디오, 오디오, 및 사용자 명령어의 어느 부분인지를 평가한다. 채널의 "품질" 평가는 위에서 설명된 것처럼 패킷 손실, 레이턴시 등을 측정하는 것과 같은 기능을 포함할 것이다. 만약 채널이 특히 신뢰할 수 없다면, 그 다음 호스팅 서비스(210)는 모든 I 타일, P 타일, 오디오 및 사용자 명령어에 FEC를 적용할 것이다. 대조적으로, 만약 채널이 신뢰할만하다면, 그 다음 호스팅 서비스(210)는 오직 오디오 및 사용자 명령어에 FEC를 적용하거나, 오디오 또는 비디오에 FEC를 적용하지 않거나, 전혀 FEC를 사용하지 않을 것이다. 그 밖의 다양한 FEC의 응용의 변경은 여전히 이러한 근본적인 이론을 따르면서 채택될 수 있을 것이다. 일 실시예에서, 호스팅 서비스(210)는 연속적으로 채널의 상태와 그에 맞는 FEC 정책의 변화를 모니터링한다.

[0181] 다른 실시예에서, 도 4a 및 4b를 참고하면, 어떤 패킷이 손실/지연된 때, 타일 데이터의 손실을 초래하거나 만약 아마도 특정 나쁜 패킷 손실때문에, FEC는 타일 데이터 손실을 보정할 수 없고, 클라이언트(415)는 새로운 I 타일이 수신되기 전에 얼마나 많은 프레임이 떠나게 되었는지를 평가하고 클라이언트(415)에서 호스팅 서비스(210)까지 왕복 레이턴시를 비교한다. 만약, 왕복 레이턴시가 새로운 I 타일이 도착 예정 전에 다수의 프레임보다 적다면, 그 다음 클라이언트(415)는 새로운 I 타일을 요청하도록 호스팅 서비스(210)로 메세지를 보낸다. 이

러한 메세지는 비디오 압축기(404)로 라우트되고, 타일의 데이터가 손실되는 동안 P 타일을 생성하기 보다, I 타일을 생성한다. 도 4a 및 4b에서 도시된 시스템은 전형적으로 80ms보다 작은 왕복 레이턴시를 제공하기 위해서 설계되고, 이는 80ms(60fps에서 프레임은 16.67ms의 지속기간이고, 따라서 전체 프레임 시간에서 80ms 레이턴시는 83.33ms내에 보정된 타일을 초래할 것이며, 이는 5 프레임 시간 - 인식할 수 있을 만한 혼란, 그러나 예컨대 15 프레임 동안에 250ms 혼란보다는 훨씬 덜 인식할 수 있다). 압축기(404)가 그것의 보통 순환적인 차수를 벗어난 I 타일을 생성하고, 만약 I 타일이 이용가능한 대역폭을 초과하도록 프레임의 대역폭을 초래하면, 압축기(404)는 그 밖의 타일의 사이클을 지연시킬 것이므로 그 밖의 타일은 프레임 시간 동안(비록 하나의 타일이 정상적으로 그 프레임 동안에 I 타일을 수신할지라도) P 타일을 수신하고, 다음 프레임으로 시작하면 보통 사이클링이 계속될 것이며, 정상적으로 앞선 프레임에서 I 타일을 수신하게 될 타일은 I 타일을 수신할 것이다. 비록 이러한 동작이 간단히 R 프레임 사이클링의 상태를 지연할지라도, 그것은 정상적인 시각으로 인식할 수는 없을 것이다.

[0182] **비디오 및 오디오 압축기/신장기 실행(Video and Audio Compressor/Decompressor Implementation)**

[0183] 도 12는 멀티 코어 및/또는 멀티-프로세서(1200)가 병렬로 8 타일을 압축하기 위해서 사용되는 특정 실시예를 도시하고 있다. 일 실시예에서, 독립 프로세서로써 오픈 소스 x264 H.264 압축기를 실행하는 각각의 코어를 가지고, 2.66GHz 또는 더 높은 속도에서 동작하는 듀얼 프로세서, 쿼드 코어 Xeon CPU 컴퓨터 시스템이 사용된다. 그러나, 다양한 그 밖의 하드웨어/소프트웨어 구성은 이러한 근본적인 이론에 따르는 동안에 사용될 것이다. 예컨대, 각각의 CPU 코어는 FPGA에서 실행되는 H.264 압축기로 대체될 수 있다. 도 12에 도시된 예에서, 코어(1201 ~ 1208)는 8개 독립적인 쓰레드로서 I 타일 및 P 타일을 동시에 처리하기 위해서 사용된다. 종래에 잘 알려진 것처럼, 현재 멀티 코어 및 멀티 프로세서 컴퓨터 시스템은 마이크로소프트 윈도우 XP 프로페셔널 에디션(64비트 또는 32비트 에디션) 및 리눅스와 같은 멀티-쓰레딩 동작 시스템으로 통합된 때 멀티-쓰레딩이 본래적으로 가능할 수 있다.

[0184] 도 12에 도시된 실시예에서, 각각의 8개 코어는 오직 하나의 타일을 책임지고, 그것은 대체로 그 밖의 코어, 각각 동작하는 x264의 개별 인스턴스(instantiation)와 독립적으로 동작한다. Sendero Video Imaging IP Development Board from Microtronix of Oosterhout와 같은 DVI 캡쳐 카드에 기반한 PCI Express x1, 네덜란드는 640×480, 800×600, 또는 1280×720 해상도에서 압축되지 않은 비디오를 캡쳐하기 위해 사용되고, 카드의 FPGA는 시스템 RAM으로 DVI 버스를 통하여 캡쳐된 비디오를 전송하기 위해서 직접 메모리 접근(Direct Memory Access; DMA)을 사용한다. 타일은 4×2 배열(1205)에 정렬된다(비록, 정사각형 타일로써 그들이 도시되었어도, 이러한 실시예에서 그들은 160×240 해상도이다). x264의 각 인스턴스가 8개 160×240 타일 중의 하나를 압축하기 위해서 구성되고, 도 12에 도시된 7개 P 타일에 의해서 뒤따르는 I 타일을 압축하기 위해서, 그들은 초기 I 타일 압축 후에, 각각의 코어가 한 사이클로 들어가고, 각각의 프레임은 그 밖의 것과 상태를 벗어난 후에 동기화된다.

[0185] 각 프레임 시간에서, 앞서 설명된 기술을 사용하여, 결과 압축된 타일은 패킷 스트림으로 결합되고 나서 압축된 타일은 목적 클라이언트(415)로 전송된다.

[0186] 비록 도 12에 도시되지 않았지만, 만약 결합된 8 타일의 전송률이 특정 피크 전송률(941)을 초과한다면, 그 다음 모든 8개 x264 프로세스는 결합된 8 타일을 위한 데이터가 전송될 때까지 필요한 만큼의 프레임 시간 동안에 일시 중지된다.

[0187] 일 실시예에서, 클라이언트(415)는 FFmpeg의 8 인스턴스를 동작하는 PC에 소프트웨어처럼 실행된다. 수신 프로세스는 8 타일을 수신하고, 각각의 타일은 FFmpeg 인스턴스로 전송되고, 이는 타일을 신장하고 디스플레이 장치(422)에 적당한 타일 위치를 위해 그것을 랜더링한다.

[0188] 클라이언트(415)는 PC의 입력 장치 드라이버로부터 키보드, 마우스, 게임 컨트롤러의 입력을 수신하고 그것을 서버(402)로 전송한다. 그 다음 서버(402)는 수신된 입력 장치 데이터를 적용하고 인텔 2.16GHz Core Duo CPU를 사용하는 윈도우가 동작하는 PC, 서버(402)에서 동작하는 게임 또는 어플리케이션에 그것을 적용한다. 그 다음에 서버(402)는 새로운 프레임을 생성하고 그것을 DVI 출력을 통해서 출력하고, 마더보드 기반 그래픽 시스템, 또는 NVIDIA 8800GTX PCI 카드의 DVI 출력을 통해서 출력한다.

[0189] 동시에, 서버(402)는 디지털 오디오 출력(예컨대, S/PDIF)을 통해서 게임 또는 어플리케이션에 의해서 생산된 오디오를 출력하고, 이는 비디오 압축을 실행하는 듀얼 쿼드 코어 Xeon-기반 PC의 디지털 오디오 입력과 결합된

다. Vorbis 오픈 소스 오디오 압축기는 프로세스 쓰레드를 이용할 수 있는 어떤 코어를 사용하는 비디오와 동시에 압축된다. 일 실시예에서, 코어는 오디오 압축을 먼저 실행하여 그것의 타일을 압축하는 것을 완료한다. 그 다음 압축된 오디오는 압축된 비디오와 함께 전송되고, Vorbis 오디오 신장기를 사용하는 클라이언트(415)에서 신장된다.

[0190] **호스팅 서비스 서버 센터 배치(Hosting Service Server Center Distribution)**

[0191] 광섬유와 같은, 유리를 통하는 광은, 진공에서 광속도에 가깝게 이동하고 그래서 광섬유에서 광을 위한 정확한 전파속도가 결정될 수 있다. 그러나, 실제적으로, 라우팅 지연, 전송 비효율, 그 밖의 것에 시간을 허용하면, 우리는 인터넷에서 최적의 레이턴시는 광속도의 50%에 가까운 전송 속도를 반사하는 것을 관찰한다. 따라서, 최적의 1000마일 왕복 레이턴시는 대략 22ms이고, 최적의 3000 마일 왕복 레이턴시는 약 64ms 이다, 따라서, 미국 해안에 있는 단일 서버는 너무 멀리 떨어진 그 밖의 해안에(이는 3000마일 보다 훨씬 떨어질 수 있다) 클라이언트에게 요구되는 레이턴시로 제공하지 못할 것이다. 그러나, 도 13a에 도시된 것처럼, 만약 호스팅 서비스(210) 서버 센터(1300)가 미국의 중심에 위치하면(예컨대, 캔사스, 네브라스카 등), 미국 대륙의 어떤 위치까지의 거리가 1500 마일이거나 그 이하일 수 있고, 왕복 인터넷 레이턴시는 32ms보다 더 작다. 도 4b을 참조하면, 비록 ISP(453) 사용자에게 허용되는 최악의 레이턴시가 25ms일지라도 전형적으로 우리는 DSL 및 케이블 모뎀 시스템으로 10 ~ 15 ms에 가까운 레이턴시를 관찰한다. 또한, 도 4b는 사용자 구내(211)에서 호스팅 센터(210)까지 최대 거리로 1000마일로 가정한다. 따라서, 전형적인 ISP 사용자는 15ms의 왕복 레이턴시를 갖고 1500마일의 최대 인터넷 거리는 32ms의 왕복 레이턴시로, 사용자가 입력 장치(421)를 작동하고 디스플레이 장치(422)에서 응답을 보는 한 지점에서 전체 왕복 레이턴시는 1+1+15+32+1+16+6+8 =80ms이다. 그래서 80ms 응답 시간은 전형적으로 1500 마일의 인터넷 거리를 거쳐 달성될 수 있다. 이는 미국 대륙에서 어떤 사용자 구내가 중심부에 위치한 단일 서버 센터로 충분히 짧은 사용자 ISP 레이턴시를 갖고 접근하는 것을 허용한다.

[0192] 도 13b에 도시된 다른 실시예에서, 호스팅 서비스(210) 서버 센터들(HS1 - HS6)은, 매우 인기있는 센터와 근접하게 위치한 어떤 큰 호스팅 서비스 서버 센터로, 미국 전역에 전략적으로 위치한다(또는 그 밖의 지정학적인 지역). 일 실시예에서, 서버 센터들(HS1- HS6)은 인터넷 또는 개인 네트워크 또는 이들의 결합된 네트워크(1301)을 통해서 정보를 교환한다. 복수의 서버 센터로, 서비스는 높은 사용자 ISP 레이턴시(453)을 가지는 사용자에게 낮은-레이턴시를 제공한다.

[0193] 비록 인터넷상에서 거리가 확실히 인터넷을 통하는 왕복 레이턴시에 기여하는 요소일지라도, 때로는 그 밖의 요소는 대체적으로 레이턴시와 관련없는 활동을 시작한다. 때로는 패킷 스트림은 멀리 떨어진 위치로 인터넷을 통해서 라우트되고 다시 되돌아 오는데, 이는 긴 루프의 레이턴시를 초래한다. 때로는 적당히 동작하지 않는 경로에 라우팅 장비가 있는데, 이는 전송의 지연을 초래한다. 때로는 오버로드된 트래픽 경로가 있어서 이는 지연을 유도한다. 또한, 때로는 사용자의 ISP가 주어진 목적지로 전송하는 것을 막음으로써 실패가 생긴다. 따라서, 일반적인 인터넷은 보통 하나의 지점에서 다른 지점으로 공정하게 신뢰할 수 있고 최적의 경로 및 대체로 거리(특히, 사용자의 로컬 영역의 바깥쪽으로 라우팅을 초래하는 긴 거리 연결로)에 의해서 결정되는 레이턴시를 갖는 연결을 제공하는데, 이로 인해서는 신뢰성 및 레이턴시가 결코 보장되지 않으며 종종 사용자의 구내에서 주어진 목적지까지 일반적인 인터넷에서 달성될 수 없게 된다.

[0194] 일 실시예에서, 사용자 클라이언트(415)는 비디오 게임을 플레이하고 어플리케이션을 사용하기 위해서 처음에 호스팅 서비스(210)와 연결되고, 클라이언트는 시작시에 이용가능한 호스팅 서비스 서버 센터(HS1 - HS6)의 각각과 통신한다(예컨대, 앞서 살펴본 기술을 사용하여). 만약 레이턴시가 특정 연결을 위해서 충분히 낮으면, 그 연결은 사용된다. 일 실시예에서, 클라이언트는 모든 또는 하위 조직(subset)과 통신하고, 호스팅 서비스 서버 센터 중 가장 레이턴시가 낮은 연결이 선택된다. 클라이언트는 가장 낮은-레이턴시 연결을 갖는 서비스 센터를 선택할 것이고 서비스 센터는 가장 낮은-레이턴시 연결을 갖는 것 중 하나로 식별될 것이며 이러한 정보(예컨대, 인터넷 주소의 형태)를 클라이언트에 제공한다.

[0195] 만약 특정 호스팅 서비스 서버 센터가 오버로드되고 그리고/또는 사용자의 게임 또는 어플리케이션이 적게 로드된 호스팅 서비스 서버 센터에 대한 레이턴시를 견딜수 있다면, 클라이언트(415)는 다른 호스팅 서비스 서버 센터로 향하게 된다. 그러한 상황에서, 사용자가 동작하고 있는 게임 또는 어플리케이션은 사용자의 오버로드된 서버 센터에 있는 서버(402)에서 중지될 것이고, 게임 또는 어플리케이션 상태 정보는 다른 호스팅 서비스 서버 센터에 있는 서버(402)로 전송될 것이다. 게임 또는 어플리케이션은 재개될 것이다. 일 실시예에서, 호스팅 서비스(210)는 게임 또는 어플리케이션이 전송을 하기 위해서 당연히 중지 포인트(예컨대, 게임에서 레벨 사이에,

또는 사용자가 어플리케이션에서 "저장(save)" 명령을 시작한 후에)에 도달될 때까지 기다리게 될 것이다. 이미 다른 실시예에서, 호스팅 서비스(210)는 지정된 기간의 시간(예컨대, 1분) 동안에 사용자 활동이 중지할 때까지 기다릴 것이고 그 다음 그 시간에 전송을 시작할 것이다.

[0196] 앞서 살펴본 것처럼 일 실시예에서, 호스팅 서비스(201)는 그것의 클라이언트에게 보장된 레이턴시를 제공하도록 하기 위해서 도 14의 인터넷 바이패스 서비스(440)에 가입한다. 여기서 사용되는 것처럼, 인터넷 바이패스 서비스는, 보장되는 특징(예컨대, 레이턴시, 전송률 등)을 갖는 인터넷의 한 지점에서 다른 지점으로 개인 네트워크 경로를 제공하는 서비스이다. 예컨대, 만약 호스팅 서비스(210)가 샌프란시스코에서 제공하는 AT&T의 DSL 서비스를 사용하는 사용자로부터 많은 양의 트래픽을 수신한다면, AT&T의 샌프란시스코 기반의 센트럴 오피스로 라우팅하는 것보다는, 호스팅 서비스(210)는 호스팅 서비스(210)를 위해 샌프란시스코 기반의 센트럴 오피스 및 하나 이상의 서버 센터 사이의 서비스 공급자(아마도 AT&T 자체 또는 다른 공급자)로부터 높은-성능 개인 데이터 연결을 임대할 수 있다. 그 다음에, 만약 모든 호스팅 서비스 서버 센터(HS1 - HS6)로부터 AT&T DSL을 사용하는 샌프란시스코의 사용자에게 일반적인 인터넷을 통하여 라우트되면 너무 높은 레이턴시를 초래하므로, 그 대신에 개인 데이터 연결이 사용될 수 있다. 비록 개인 데이터 연결이 일반적인 인터넷을 통한 라우트보다 일반적으로 더 비싸지만, 사용자와 호스팅 서비스(210)간의 연결이 너무 작은 비율로 오랫동안 남아 있으므로 전체 비용의 영향은 낮을 것이고, 사용자는 보다 일관된 서비스를 경험할 것이다.

[0197] 서버 센터는 자주 정전 이벤트시에 2개 층의 백업 파워를 갖는다. 전형적으로 첫번째 층은 배터리로부터 파워를 백업하고(또는 택일적으로 즉시 이용가능한 에너지 소스, 발전기에 연결되고 동작이 계속되는 플라이휠과 같은), 이는 메인 파워가 끊어진 때 즉시로 전원을 공급하고 서버 센터 동작을 유지한다. 만약 정전이 단순하다면, 메인 파워는 빠르게 되돌아온다(예컨대, 1분 이내), 그 다음 배터리는 서버 센터 동작을 유지하기 위해 요청되는 것이다. 그러나 만약 정전이 오랜 시간동안 발생하면, 전형적으로 발전기(예컨대, 디젤-파워)가 보유한 연료만큼 오랫동안 작동될 수 있고 배터리를 위해 전달된다. 그러한 발전기는 서버 센터가 정상적으로 메인 파워로부터 얻을 수 있는 많은 전원만큼 생산하는 성능이 있기 때문에 굉장히 비싸다.

[0198] 일 실시예에서, 호스팅 서비스(HS1 - HS5)의 각각은 다른 것과 사용자 데이터를 공유하므로 한 서버 센터가 정전되면, 그것은 동작중인 게임 및 어플리케이션이 중지될 수 있고, 각 서버(402)에서 다른 서버 센터의 서버들(402)로 게임 또는 어플리케이션 상태 데이터를 전송하며, 새로운 서버(402)와 통신하도록 각 사용자의 클라이언트(415)에게 알릴 것이다. 이러한 사용이 가끔 일어나면, 사용자가 최적의 레이턴시를 제공할 수 없는 호스팅 서비스 서버 센터로 사용자를 전달하는 것이 허용될 것이고(예컨대, 사용자는 단순히 정전의 지속 기간 동안에 높은 레이턴시를 견뎌야 한다), 이는 전송하는 사용자를 위해 많은 넓은 범위 선택을 허용할 것이다. 예컨대, 미국 전역의 표준시간대(time zone)의 차이는, 동부해안에 사용자는 오후 11:30에 잠자리에 들어갈 것이고 반면에 서부해안에 사용자는 오후 8:30에 비디오 게임의 사용의 피크가 시작될 것이다. 만약 동시에 서부 해안의 호스팅 서비스 서버 센터가 정전에 있으면, 모든 사용자를 다루기 위한 그 밖의 호스팅 서비스 서버 센터에 충분한 서부 해안 서버(402)가 있지 않을 것이다. 그러한 상황에서, 어떤 사용자들은 이용가능한 서버(402)를 갖고 있는 동부 해안의 호스팅 서비스 서버 센터로 전송될 수 있고, 사용자에게 뒤따르는 결과는 오직 높은 레이턴시일 것이다. 사용자가 전원을 잃은 서버 센터로부터 전송된다면, 서버 센터는 그것의 서버 및 기구의 중단된 차례로 개시될 수 있고, 모든 기구는 배터리(또는 그 밖의 즉각적인 전원 백업)가 소진되기 전에 중단되게 된다. 이러한 방식으로, 서버 센터를 위한 발전기 비용은 피할 수 있다.

[0199] 일 실시예에서, 호스팅 서비스(210)의 무거운 로딩 시간 동안(사용자 로딩 피크에 기인하거나 하나 이상의 서버 센터가 작동하지 않게 된다)에 사용자는 그들이 사용하는 게임 또는 어플리케이션의 레이턴시 요구에 기초하여 그 밖의 서버 센터로 전송된다. 그래서 낮은-레이턴시를 요구하는 게임 또는 어플리케이션을 사용하는 사용자는 제한된 공급이 있을 때, 이용가능한 낮은-레이턴시 서버 연결로 특혜를 받게 될 것이다.

[0200] **호스팅 서비스 특징(Hosting Service Features)**

[0201] 도 15는 다음의 특징을 설명하는데 이용되는 호스팅 서비스(210)를 위한 서버 센터의 구성요소의 실시예를 도시하고 있다. 도 2a에 도시된 호스팅 서비스(210)처럼, 이 서버 센터의 구성요소는 그 밖의 제한이 없다면, 호스팅 서비스(201) 제어 시스템(401)에 의해서 제어되고 조정된다.

[0202] 사용자 클라이언트(415)로부터 인바운드 인터넷 트래픽(1501)은 인바운드 라우팅(1502)으로 이동된다. 전형적으로, 인바운드 인터넷 트래픽(1501)은 인터넷을 위해 고속 파이버 광연결을 통해서 서버 센터로 들어갈

것이지만, 적당한 대역폭, 신뢰성 및 낮은-레이턴시의 네트워크 연결 수단이 충분할 것이다. 인바운드 라우팅(1502)은 네트워크(네트워크는 이더넷 네트워크, 섬유 채널 네트워크 또는 어떤 그 밖의 전송 수단을 통해서 실행될 수 있다)의 시스템 스위치 및 적당한 어플리케이션/게임("어플(app)/게임(game)") 서버(1521 - 1525)로 각 패킷을 라우트하고 도착한 패킷을 취하는 스위치를 지지하는 라우팅 서버이다. 일 실시예에서, 특정 어플리케이션/게임 서버로 전달되는 어떤 패킷은 클라이언트로부터 수신된 데이터의 서브셋을 나타내고 그리고/또는 그 밖의 구성요소(예컨대, 게이트웨이 및 라우터와 같은 네트워킹 구성요소)에 의해서 데이터 센터 내에서 변환/변경될 것이다. 어떤 경우에, 예컨대, 만약 게임 또는 어플리케이션이 병렬로 한번에 다수의 서버에서 구동되고 있다면, 패킷은 서버(1521 -1525) 중 하나 이상으로 한번에 라우트될 것이다. RAID 배열(1511 - 1512)은 인바운드 라우팅 네트워크(1502)와 연결되고, 어플리케이션/게임 서버(1521 - 1525)는 RAID 배열(1511 - 1512)을 읽거나 쓸 수 있다. 더욱이, RAID 배열(1515)(이는 멀티 RAID 배열로써 실행될 것이다)은 또한 인바운드 라우팅(1502)과 연결되고 RADI 배열(1515)로부터 데이터는 어플리케이션/게임 서버(1521 - 1525)로부터 읽을 수 있다. 인바운드 라우팅(1502)은 루트(root)에 인바운드 인터넷 트랙픽(1501)을 갖는 트리 구조의 스위치를 포함하고; 모든 다양한 장치를 상호결합하는 메쉬 구조로; 또는 그 밖의 장치들 사이에 집중되는 트래픽으로부터 분리되고 교환하는 장치들 중에서 집중되는 트랙픽을 가진, 일련의 서브넷과 상호결합되는 것처럼; 종래 기술의 네트워크 아키텍처의 넓은 범위에서 실행될 수 있을 것이다. 네트워크 구성의 한 형태는 SAN이고, 비록 전형적으로 스토리지 장치를 위해 사용될지라도, 그것은 또한 장치들 사이에 일반적인 고속 데이터 변환을 위해 사용될 수 있다. 또한, 어플리케이션/게임 서버(1521 - 1525)는 각각이 인바운드 라우팅(1502)과 멀티 네트워크 연결을 갖게 될 것이다. 예컨대, 서버(1521-1525)는 RAID 배열(1511-1512)에 부착되는 서브넷과 네트워크 연결 및 그 밖의 장치에 부착되는 서브넷과 또 다른 네트워크 연결을 갖게 될 것이다.

[0203] 도 4a에 도시된 실시예에서 서버(402)와 관련하여 이미 설명된 것처럼, 어플리케이션/게임 서버(1521-1525)는 모두 동일하게, 일부 다르게, 모두 다르게 구성될 것이다. 일 실시예에서, 호스팅 서비스를 사용할 때, 각각의 사용자는, 전형적으로 적어도 하나의 어플리케이션/게임 서버(1521 - 1525)를 사용한다. 설명의 단순화를 위해서, 주어진 사용자는 어플리케이션/게임 서버(1521)을 사용하는 것으로 가정하지만, 멀티 서버가 한 사용자에 의해서 사용될 수 있고, 멀티 사용자가 단일 어플리케이션/게임 서버(1521 - 1525)를 공유할 수 있다. 앞서 설명된 것처럼 클라이언트(415)로부터 보내진 사용자의 제어 입력은, 인바운드 인터넷 트래픽(1501)으로 수신되고, 인바운드 라우팅(1502)을 통해서 어플리케이션/게임 서버(1521)로 라우트된다. 어플리케이션/게임 서버(1521)는 사용자의 제어 입력을 서버에서 동작하는 게임 또는 어플리케이션의 제어 입력으로써 사용되고, 그것과 관련된 다음 프레임의 비디오 및 오디오를 연산한다. 그 다음 어플리케이션/게임 서버(1521)는 압축되지 않은 비디오/오디오(1529)를 공유된 비디오 압축(1530)으로 출력한다. 어플리케이션/게임 서버는 하나 이상의 기가비트 이더넷 커넥션을 포함하는, 어떤 수단을 통해서 압축되지 않은 비디오를 출력할 것이나, 어떤 실시예에서는 비디오가 DVI 커넥션을 통해서 출력되고 오디오 및 그 밖의 압축 및 통신 채널 상태 정보는 범용 직렬 버스(USB) 커넥션을 통해서 출력된다.

[0204] 공유된 비디오 압축(1530)은 어플리케이션/게임 서버(1521 - 1525)로부터 압축되지 않은 비디오 및 오디오를 압축한다. 압축은 아마도 전체적으로 하드웨어로서 또는 소프트웨어가 동작하는 하드웨어로 실행될 것이다. 각 어플리케이션/게임 서버(1521 - 1525)를 위한 전용 압축기가 있을 것이나, 만약 압축기가 충분히 빠르다면, 주어진 압축기는 하나 이상의 어플리케이션/게임 서버(1521-1525)로부터 비디오/오디오를 압축하기 위해 사용될 수 있다. 예컨대, 60fps에서 비디오 프레임 시간은 16.67ms이다. 만약, 압축기가 1ms 이내에 하나의 프레임을 압축할 수 있다면, 압축기는 16 어플리케이션/게임 서버(1521-1525)만큼 많은 것으로부터의 하나의 서버에서 그 후에 다른 것으로부터 입력을 얻음으로써, 각각의 비디오/오디오 압축 처리의 상태를 저장하고 서버로부터 비디오/오디오 스트림 사이에 사이클되는 것처럼 콘텍스트(context)를 스위칭하는 압축기를 가지고, 비디오/오디오를 압축하기 위해서 사용될 수 있다. 이는 실질적으로 압축 하드웨어에서 비용 절약을 가져온다. 일 실시예에서, 다른 서버가 다른 시간에 프레임을 완료하기 때문에, 압축기 리소스는 각각의 압축기 처리의 상태를 저장하기 위해 공유된 저장 수단(예컨대, RAM, 플래시)을 갖는 공유된 풀(pool;1530)에 있고, 서버(1521 - 1525) 프레임이 완료되고 압축되기 위해 준비된 때, 제어 수단은 그 시간에 압축 리소스를 이용할 수 있을지 결정하고, 압축하기 위해서 압축되지 않은 비디오/오디오의 프레임 및 서버의 압축 처리의 상태를 갖는 압축 리소스를 제공한다.

[0205] 각 서버의 압축 처리의 상태의 일부는, P 타일을 위한 기준으로써 사용될 수 있고, 사전 프레임의 신장된 프레임 버퍼 데이터, 비디오 출력의 해상도; 압축의 품질; 타일링 구조; 타일 당 비트의 할당; 압축 품질, 오디오 포맷(예컨대, 스테레오, 서라운드 사운드, Dolby® , AC-3)와 같이 압축 자체에 대한 정보를 포함한다는 것을 주

목하라. 그러나 압축 처리의 상태는 또한 피크 전송률(941)에 관한 통신 채널 상태 정보를 포함하고, 사전 프레임(도 9b에서 도시된 것처럼)이 현재 출력되고 있는지(그리고 결과로써 현재 프레임이 무시될 수 있을 것이다), 그리고 잠재적으로 과도한 패킷 손실과 같은 압축을 고려해야 채널 특징이 존재하는지 여부는 압축을 위한 결정에 영향을 미친다(예컨대, I 타일의 주파수에 관하여 등). 피크 전송률(941) 또는 그 밖의 채널 특성은 시간에 따라 변하기 때문에, 클라이언트(415)로부터 보내진 각각의 사용자 모니터링 데이터를 지지하는 어플리케이션/게임 서버(1521-1525)에 의해서 결정됨으로써, 어플리케이션/게임 서버(1521-1525)는 공유된 하드웨어 압축(1530)으로 관련된 정보를 보낸다.

[0206] 또한, 공유된 하드웨어 압축(1530)은 앞서 설명했던 것과 같은 수단을 이용하고, 적당하다면 FEC 코드를 적용하며, 소정의 데이터를 복사하거나, 실현가능한 안정성 및 높은 품질로 신장된 그리고 클라이언트(415)에 의해서 수신된 비디오/오디오 데이터 스트림의 성능을 적당히 보장하기 위해 그 밖의 단계를 거쳐서 압축된 비디오/오디오를 패킷화한다.

[0207] 아래 설명되는 것과 같은, 어떤 어플리케이션은, 동시에 멀티 해상도(또는 다른 멀티 포맷)에서 이용될 수 있는 주어진 어플리케이션/게임 서버(1521 - 1525)의 비디오/오디오 출력을 요구한다. 따라서 만약 어플리케이션/게임 서버(1521 - 1525)가 공유된 하드웨어 압축(1530) 리소스를 통지하면, 어플리케이션/게임 서버(1521-1525)의 비압축 비디오 오디오(1529)는 동시에 다른 포맷, 다른 해상도, 및/또는 다른 패킷/에러 보정 구조로 압축될 수 있을 것이다. 어떤 경우에, 어떤 압축 리소스는 동일한 비디오/오디오를 압축하는 복수의 압축 처리 중에서 공유될 수 있다(예컨대, 많은 압축 알고리즘에서, 무언가에 의해서 이미지가 압축 적용되기 전에 복수의 크기로 스케일되는 단계가 있다. 만약 다른 크기의 이미지가 출력되도록 요구되면, 이 단계는 한 번에 몇 개의 압축 처리를 제공하는데 사용될 수 있다). 다른 경우로, 개별 압축 리소스는 각각의 포맷을 위해서 요구될 것이다. 어떤 경우에는, 주어진 어플리케이션/게임 서버(1521 - 1525)를 위해서 요청된 압축된 비디오/오디오(1539)의 모든 다양한 해상도 및 포맷은 한번에 아웃바운드 라우팅(1540)로 출력될 것이다. 일 실시예에서 압축된 비디오/오디오(1539)의 출력은 UDP 포맷이고, 그래서 패킷의 단방향 스트림이다.

[0208] 아웃바운드 라우팅 네트워크(1540)는 각각의 압축된 비디오/오디오 스트림을 아웃바운드 인터넷 트래픽(1599) 인터페이스(이는 전형적으로 인터넷과 섬유 인터페이서로 연결될 것이다)를 통해서 의도된 사용자 또는 그 밖의 목적지로 향하게 하고 그리고/또는 지연 버퍼(1515)로 되돌아가게 하며, 그리고/또는 인바운드 라우팅(1502)으로 되돌아가게 하고, 그리고/또는 개인 네트워크(도시되지 않음)를 통해서 비디오 분배를 위해 출력하는 일련의 라우팅 서버 및 스위치를 포함한다.(아래 설명되는 것처럼) 아웃바운드 라우팅(1540)은 주어진 비디오/오디오 스트림을 한 번에 복수의 목적지로 출력할 것이다. 일 실시예에서 이는 인터넷 프로토콜(IP) 멀티태스크를 사용하여 실행되고 한 번에 복수의 목적지로 스트림되도록 의도된 주어진 UDP스트림은 브로드캐스트되고, 브로드캐스트는 아웃바운드 라우팅(1540)에 있는 라우팅 서버 및 스위치에 의해서 반복된다. 브로드캐스트의 복수의 목적지는 인터넷을 통해서 복수의 사용자의 클라이언트(415)로 될 것이고, 인바운드 라우팅(1502) 및/또는 하나 이상의 지연 버퍼(1515)를 통해서 복수의 어플리케이션/게임 서버(1521 - 1525)로 될 것이다. 따라서, 주어진 서버(1521 - 1522)의 출력은 단일 또는 복수 포맷으로 압축되고, 그리고 각각의 압축된 스트림은 단일 또는 복수의 목적지로 향하게 된다.

[0209] 더욱이, 다른 실시예에서, 만약 복수 어플리케이션/게임 서버(1521 - 1525)는 동시에 한 명의 사용자(예컨대, 복잡한 장면의 3D 출력을 생성하기 위한 병렬 처리 구성에서)에 의해서 동시에 사용되고, 각각의 서버는 결과 이미지의 일부를 생성할 것이며, 복수 서버(1521 - 1525)의 비디오 출력은 결합된 프레임으로 공유된 하드웨어 압축(1530)에 의해서 결합될 수 있고, 마치 단일 어플리케이션/게임 서버(1521 - 1525)으로부터 온 것처럼, 그것은 순방향 포인트에서 앞서 설명된 것처럼 다루어진다.

[0210] 일 실시에서, 어플리케이션/게임 서버(1521 - 1525)에 의해서 생성된 모든 비디오의 복사는 최소 몇 분(일 실시예에서는 15 분)동안에 지연 버퍼(1515)에 기록된다. 이는 각 사용자에게 사전 동작 또는 이용을 검토하기 위하여 각각의 세션으로부터 비디오 "되감기(rewind)"를 허용한다(게임의 경우에). 따라서, 일 실시예에서, 사용자 클라이언트(415)로 라우트된 각각의 압축된 비디오/오디오 출력(1539) 스트림은 또한 지연 버퍼(1515)로 멀티캐스트된다. 비디오/오디오가 지연 버퍼(1515)에 저장된 때, 지연 버퍼(1515)의 디렉토리는 지연된 비디오/오디오가 발견될 수 있는 지연 버퍼(1515)의 위치 및 지연된 비디오/오디오의 소스를 어플리케이션/게임 서버(1521 - 1525)의 네트워크 어드레스 사이에서 상호 참조로 제공한다.

[0211] **라이브, 즉시-볼 수 있는, 즉시-플레이할 수 있는 게임(Live, Instantly-Viewable,Instantly-playable Games)**

[0212] 어플리케이션/게임 서버(1521 - 1525)는 사용자를 위해 주어진 어플리케이션 또는 비디오 게임을 동작하기 위해서 사용되는 것일 뿐만아니라, 그들은 호스팅 서비스(210) 및 그 밖의 특징을 통해서 네비게이션을 지원하는 호스팅 서비스(210)를 위한 사용자 인터페이스 어플리케이션을 만들기 위해서 사용되는 것이다. 이러한 사용자 인터페이스 어플리케이션, "게임 파인더(game finder)" 의 스크린 샷이 도 16에서 도시된다. 이러한 특정 사용자 인터페이스 스크린은 사용자에게 그 밖의 사용자에 의해서 라이브로 플레이되는(또는 지연되는) 15 게임을 지켜보도록 허용한다. 1600과 같은 "섬네일(thumbnail)" 비디오 윈도우 각각은, 한 사용자의 게임으로부터 비디오를 보여주는 움직임 내의 라이브 비디오 윈도우이다. 섬네일에서 보여지는 시야는 사용자가 보고 있는 시야와 동일한 것이나, 그것은 지연된 시야일 것이다(예컨대, 만약 사용자가 전투 게임을 플레이하고 있다면, 사용자는 어디에 숨었는지 일정 시간까지 다른 사용자에게 알려지길 원하지 않고, 10분가량 게임플레이의 어떤 시야가 지연되도록 선택할 것이다). 또한 시야는 어떤 사용자의 시야와 다르게 카메라의 시야일 것이다. 메뉴 선택을 통해서(이러한 일러스트는 도시안됨), 사용자는 동시에 다양한 기준에 기초한 보기를 위해서 게임을 선택할 것이다. 전형적인 선택(choice)의 작은 샘플링으로써, 사용자는 모든 게임 중 한 가지 종류를(모두 다른 플레이어에 의해서 플레이), 오직 최상위 등급 플레이어, 게임에서 주어진 수준에 있는 플레이어, 또는 하위 등급 플레이어(예컨대, 만약 플레이어가 기본을 배운다면), 동료("buddies")(또는 경쟁자) 플레이어, 대다수의 뷰어(viewer)를 가지고 있는 게임 등을 무작위로 선택할 것이다(도 16에서 도시된 것처럼).

[0213] 일반적으로, 각 사용자는 그의 또는 그녀의 게임 또는 어플리케이션을 그 밖의 사람들이 지켜볼 수 있을지를 결정할 것이고, 만약 그 밖의 사람들이 지켜볼 수 있게 된다면, 오직 지연된 것을 지켜보도록 할 것인지를 결정할 것이다.

[0214] 도 16에 도시된 사용자 인터페이스 스크린을 생성하는 어플리케이션/게임 서버(1521 - 1525)는 각각의 사용자가 누구의 게임인지를 요청하는 동안에 어플리케이션/게임 서버(1521 - 1525)로 메세지를 보냄으로써 15 비디오/오디오 공급(feeds)을 요구한다. 메세지가 인바운드 라우팅(1502) 또는 다른 네트워크를 통해서 보내진다. 메세지는 요청된 비디오/오디오의 크기와 포맷을 포함할 것이고, 사용자 인터페이스 화면을 지켜보는 사용자를 식별할 것이다. 주어진 사용자는 "프라이버시(privacy)" 모드를 고르도록 선택할 것이고 어떤 그 밖의 사용자가 그의 게임의 비디오/오디오를 지켜보는 것이 허용되지 않고(그의 시야 또는 다른 시야 모두), 또는 앞 단락에서 설명된 것처럼, 사용자는 그녀의 게임으로부터 비디오/오디오를 감상(viewing)하는 것을 허락하도록 선택할 것이지만, 감상하는 비디오/오디오는 지연된다. 사용자 어플리케이션/게임 서버(1521 - 1525)가 감상하고자 하는 비디오/오디오를 허락하기 위한 요청을 수신하고 받아들이면 요청 서버(requesting server)를 승인할 것이고, 그것은 또한 요청된 포맷 또는 스크린 크기(포맷 및 스크린 크기는 이미 생성된 것과는 다르다고 가정)에 추가적으로 압축된 비디오 스트림을 생성하기 위한 필요를 공유된 하드웨어 압축(1530)에 알릴 것이며, 그리고 또한 압축된 비디오를 위한 목적지(예컨대, 요청하는 서버)를 지시한다. 만약, 요청된 비디오/오디오가 오직 지연되면, 요청하는 어플리케이션/게임 서버(1521 - 1525)는 통지받게 될 것이고, 그것은 지연 버퍼(1515)에 있는 디렉토리에서 비디오/오디오의 위치, 지연된 비디오/오디오의 소스인 어플리케이션/게임 서버(1521 - 1525)의 네트워크 주소를 찾음으로써 지연 버퍼(1515)에서 지연된 비디오/오디오를 요구할 것이다. 일단 모든 이러한 요청이 생성되고 다뤄지면, 15개 섬네일-크기의 비디오 스트림까지는 아웃바운드 라우팅(1540)에서 인바운드 라우팅(1502)까지 사용자 인터페이스 스트림을 생성하는 어플리케이션/게임 서버(1521 - 1525)를 위해서 라우트될 것이고, 서버에 의해서 신장되고 디스플레이될 것이다. 지연된 비디오/오디오 스트림이 너무 큰 스크린 크기에 있다면, 어플리케이션/게임 서버(1521 - 1525)는 스트림을 신장할 것이고 섬네일 크기로 비디오 스트림을 크기를 줄일 것이다. 일 실시예에서, 오디오/비디오를 위한 요청은 요청을 적당한 어플리케이션/게임 서버(1521 - 1525)로 다시 향하게 하는 도 4a(도 15에는 도시되지 않음)의 호스팅 서비스 컨트롤 시스템과 비슷한 센트럴 "관리(management)" 서비스로 보내진다(그리고 관리된다). 더욱이, 일 실시예에서, 어떤 요청이 요구되지 않는데 이는 섬네일들이 그것을 허용하는 사용자들의 클라이언트로 "넣어진(pushed)" 때문이다.

[0215] 동시에 15 게임으로부터 모두 섞인 오디오는 불협화음의 소리를 만들 것이다. 사용자는 이 방식으로 모든 소리를 함께 선택할 것이나(아마도 보여지는 모든 행동에 의해서 생성되는 "소음(din)"의 감각을 얻는다), 또는 사용자는 한 번에 하나의 게임에서 오디오를 듣기 위해 선택할 것이다. 단일 게임의 선택은 주어진 게임에 옐로우 선택 상자(1601)를 움직여서 성취될 수 있다(옐로우 상자 움직임은 키보드에 화살표키를 사용함으로써, 마우스를 움직임으로써, 조이스틱을 움직임으로써, 또는 모바일 폰같은 다른 장치에 방향 버튼을 누름으로써 성취될 수 있다). 일단 단일 게임이 선택되면, 그 게임 플레이로부터 오직 오디오를 얻는다. 또한, 게임 정보(1602)가 도시된다. 예컨대, 이 게임의 경우에, 발행인 로고("EA"), 게임 로고, "Need for Speed Carbon" 및 오렌지색 수직의 바는 특정 순간에 게임을 관람하거나 플레이하는 사람의 수와 관련된 항목을 지시한다. 더욱이 Need for

Speed Game의 활동적으로 80개 다른 인스턴스 플레이하는 145 플레이어가 있고(예컨대, 그것은 개인 플레이어 게임 또는 멀티 플레이어 게임에 의해서 플레이될 수 있다), 거기에는 680명의 관찰자(사용자 중의 한 명)가 있다는 것을 나타내는 "stats"가 제공된다. 이러한 통계(그리고 그 밖의 통계)는 호스팅 서비스 210개 동작, 적당히 지불한 사용자 및 컨텐츠를 제공하는 출판인에게 지불되는 것의 로그를 유지하기 위해서 호스팅 서비스 컨트롤 시스템(401)에 의해서 수집되고 RAID 배열(1511 - 1512)에 저장된다. 어떤 통계는 서비스 컨트롤 시스템(401)에 의한 동작에 기인하여 기록되고, 어떤 것은 개인 어플리케이션/게임 서버(1521 - 1525)에 의해서 서비스 컨트롤 시스템(401)에 보고된다. 예컨대, 이 게임 파인더 어플리케이션을 구동하는 어플리케이션/게임 서버(1521 - 1525)는 게임이 관람되고 있을 때(그리고 그들이 관람을 시작할 때) 호스팅 서비스 컨트롤 시스템(401)에 메세지를 전달하여 얼마나 많은 게임이 관람 중인지를 업데이트할 것이다. 어떤 통계는 이러한 게임 파인더 어플리케이션과 같은 사용자 인터페이스 어플리케이션에 이용될 수 있다.

[0216] 만약, 사용자가 그들의 입력장치에 활성화 버튼을 클릭하면, 그들은 전체 화면 크기로 남아있는 동안에 옐로우 상자에 확대(zoom up)되는 섬네일 비디오를 볼 것이다. 이러한 효과는 도 17에서 프로세스가 도시된다. 비디오 윈도우(1700)가 크기가 커진다는 것을 주목하라. 이러한 효과를 실행하기 위해서, 어플리케이션/게임 서버(1521 - 1525)는 전체 화면 크기(사용자의 디스플레이 장치(422)의 해상도에서)를 위해 비디오 스트림의 카피를 갖도록 선택된 게임을 동작하는 어플리케이션/게임 서버(1521 - 1525)로부터 그것을 위해 라우트된 게임을 요청한다. 게임이 동작하는 어플리케이션/게임 서버(1521 - 1525)는 게임의 섬네일-크기의 카피가 더 이상 필요하지 않는다는 것을(만약 다른 어플리케이션/게임 서버(1521 - 1525)가 그러한 섬네일을 요구하지 않는다면) 공유된 하드웨어 압축기(1530)에 알리고, 그 다음 그것은 비디오를 확대하는 어플리케이션/게임 서버(1521 - 1525)로 비디오의 전체 화면 크기 카피를 보내도록 한다. 게임을 플레이하는 사용자는 사용자가 게임을 확대하는 것처럼 동일한 해상도 디스플레이 장치(422)를 갖거나 갖지 않을 수도 있다. 더욱이, 게임의 그 밖의 관찰자들은 사용자가 게임을 확대하는 것처럼 동일한 해상도로 디스플레이 장치를 가질 수도 있고 아닐 수도 있다(그리고 다른 오디오 재생 수단을 가질 것이고, 예컨대, 스테레오 또는 서라운드 사운드). 그래서, 공유된 하드웨어 압축(1530)은 적당하게 압축된 비디오/오디오 스트림이 이미 비디오/오디오 스트림을 요청하는 사용자의 요구에 맞도록 생성되었는지 여부, 만약 존재하다면 그것은 비디오를 확대하는 어플리케이션/게임 서버(1521 - 1525)로 스트림의 카피를 라우트하기 위해서 아웃바운드 라우팅(1540)에 알리고, 만약 그렇지 않다면 사용자를 위해 적당한 다른 비디오 카피를 압축하고 비디오를 확대하는 어플리케이션/게임 서버(1521 - 1525) 및 인바운드 라우팅(1502)으로 스트림을 되돌려 보내기 위해 아웃바운드 라우팅에게 알린다. 지금 선택된 비디오의 전체 화면 버전을 수신한 서버는, 그것을 신장할 것이고 점차로 전체 크기로 스케일을 확대할 것이다.

[0217] 도 18은 게임이 전체 스크린으로 확대된 후에 스크린이 어떤지를 도시하고 있고 화살표(1800)에 의해서 지시되는 이미지에 의해 지시되는 것처럼 게임은 사용자의 디스플레이 장치(422)의 전체 해상도에서 보여진다. 게임 파인더 어플리케이션이 동작하는 어플리케이션/게임 서버(1521 - 1525)는 더 이상 필요없는 섬네일을 제공하고 있는 다른 어플리케이션/게임 서버(1521 - 1525)에게 메세지를 보내고 그 밖의 게임이 더 이상 관람되고 있지 않은 호스팅 서비스 컨트롤 서버(401)에게도 메세지를 보낸다. 그 시점에서, 디스플레이는 오직 정보 및 사용자에 메뉴 컨트롤을 제공하는 스크린의 상부에 오버레이(overlay; 1801)를 생성할 것이다. 게임이 진행됨에 따라, 관중은 2,503 관람자까지 늘어난다. 그렇게 많은 관람자와 함께, 동일 또는 거의 같은 해상도(각각의 어플리케이션/게임 서버(1521 - 1525)가 조정을 맞추기 위한 비디오를 스케일할 수 있다)를 갖는 디스플레이 장치(422)를 갖는 많은 관람자가 있다.

[0218] 도시된 게임은 멀티 플레이어 게임이기 때문에, 사용자는 어떤 시점에 게임에 참여할지 결정할 것이다. 호스팅 서비스(210)는 다양한 이유로 게임에 참여하는 사용자를 허용하거나 허용하지 않을 것이다. 예컨대, 사용자가 게임을 플레이하기 위해 지불해야 할 것이고 선택이 아니며, 사용자는 특정 게임을 참여하기 위해 충분한 랭킹을 갖고 있지 않을 것이나, 또는 사용자의 인터넷 연결은 사용자가 플레이하기에 충분히 낮은-레이턴시를 갖지 못할 것이다(예컨대, 시청하는 게임을 위해서는 레이턴시 제약은 없고, 멀리 떨어져(실제로, 다른 대륙에서) 플레이하는 게임은 레이턴시 걱정 없이 볼 수 있지만, 게임을 플레이하기 위해서, 레이턴시는 (a) 게임을 즐기기 위해서 사용자에게 충분히 낮아야 하고, (b) 낮은-레이턴시 연결을 갖는 그 밖의 플레이어와 동일한 입장에 있고). 만약 사용자가 플레이하는 것이 허용되면, 사용자를 위해 게임 파인더 사용자 인터페이스를 제공해주는 어플리케이션/게임 서버(1521 - 1525)는 호스팅 서비스 컨트롤 서버(401)가 RAID 배열(1511 - 1512)에서 게임을 로드하여 특정 게임을 플레이하기 위해 적절히 구성된 어플리케이션/게임 서버(1521 -1525)를 시작하도록(예컨대, 탐색하고 시작하다) 요청할 것이다. 그리고 호스팅 서비스 컨트롤 서버(401)가 게임 파인더 어플리케이션을 호스팅하고 있는 어플리케이션/게임 서버로부터 비디오/오디오를 압축하는 것에서 현재 게임을 호스팅하는 어플리케이션/게임 서버로부터 비디오/오디오를 압축하는 것으로 스위치하기 위해 공유된 하드웨어 압축(1530)에게

명령할 것이다. 게임 파인더 어플리케이션/게임 서비스와 새로운 어플리케이션/게임 서버 호스팅 게임의 수직 동시성(sync)은 동조되지 않고, 결과적으로 2개 싱크 사이에는 시간 차이가 생기기 쉽다. 왜냐하면, 공유된 비디오 압축 하드웨어(1530)는 비디오 프레임을 완료하는 어플리케이션/게임 서버(1521 - 1525)에서 비디오를 압축하기 시작할 것이고, 새로운 서버로부터 첫번째 프레임은 원래의 서버의 전체 프레임 시간보다 더 곧바로 완료될 것이며, 이는 먼저 압축된 프레임의 전송이 완료되기 전일 것이다(예컨대, 도 9b의 전송 시간(992)를 생각해보라 : 만약 압축되지 않은 프레임 3(963)이 프레임 시간의 절반 일찍 완료되었다면, 그것은 전송 시간(992)에 영향을 미칠 것이다. 그러한 상황에서 공유된 비디오 압축 하드웨어(1530)는 새로운 서버로부터 첫번째 프레임을 무시할 것이고(예컨대, 프레임 4(964)가 무시된 것처럼(974)), 그리고 클라이언트(415)는 나머지 프레임 시간에 종래 서버로부터 마지막 프레임을 유지할 것이며, 공유된 비디오 압축 하드웨어(1530)은 게임을 호스팅하는 새로운 어플리케이션/게임 서버로부터 다음 프레임 시간 비디오를 압축하기 시작할 것이다. 외관상으로 사용자에게, 하나의 어플리케이션/게임 서버에서 그 밖의 것으로 전송은 한결같아 보일 것이다. 호스팅 서비스 컨트롤 서버(401)은 그 다음에 그것이 다시 필요할 때까지, 한가한 상태로 전환하도록 게임 파인더를 호스팅하는 어플리케이션/게임 게임 서버(1521 - 1525)에게 알릴 것이다.

[0219] 그 다음 사용자는 게임을 플레이할 수 있다. 그리고, 예외적인 것은 게임이 즉시로 지각적으로 플레이될 것이고(그것은 초당 기가비트율에서 RAID 배열(1511 - 1512)에서 어플리케이션/게임 게임 서버(1521 - 1525)로 로드될 것이기 때문에), 게임은 이상적인 드라이버, 레지스트리 구성(Window의 경우에)을 갖고, 게임의 동작과 경쟁하는 서버에서 동작하는 그 밖의 어플리케이션을 갖지 않는 게임을 위해 정확히 구성된 오퍼레이팅 시스템과 함께 게임을 위해 정확하게 적합한 서버로 로드될 것이다.

[0220] 또한, 사용자가 게임을 진행함에 따라, 게임의 각각의 세그먼트는 초당 기가비트의 속도(예컨대 8초에 1기가바이트가 로드)로 RAID 배열(1511 - 1512)에서 서버로 로드될 것이고, RAID 배열(1511 - 1512)의 광대한 저장 성능으로(그것은 많은 사용자들 사이에 공유된 리소스이고, 그것은 매우 큰 크기이며, 여전히 가격이 효율적임) 기하학적 셋업 또는 그 밖의 게임 세그먼트 셋업은 미리 연산되고 RAID 배열(1511 - 1512)에 저장되고 매우 빠르게 로드된다. 더욱이, 각각의 어플리케이션/게임 서버(1521 - 1525)의 하드웨어 구성 및 컴퓨터적인 성능이 알려졌기 때문에, 픽셀 및 버텍스 쉐이더(vertex shader)는 미리 연산될 수 있다.

[0221] 따라서, 게임은 거의 즉시 시작될 것이고, 그것은 이상적인 환경에서 작동할 것이며, 그리고 다음의 세그먼트는 거의 즉시로 로드될 것이다.

[0222] 그러나, 이러한 이점 너머에는, 사용자가 그 밖의 사람이 게임을 플레이하는 것을 볼 수 있을 것이고(앞서 설명된 게임 파인더를 통해서나, 그 밖의 수단), 게임이 흥미로운지, 그렇다면 그 밖의 것을 지켜보는 것으로부터 팁을 배운다. 그리고, 사용자가 용량이 큰 다운로드 및/또는 인스톨을 기다리는 것 없이도 즉시 게임을 데모할 수 있을 것이고, 사용자는 즉시 게임을 플레이할 수 있을 것이며, 아마도 평가기준에서 더 작은 요금, 또는 더 장기적으로 플레이할 수 있을 것이다. 그리고, 사용자는 홈에서 Window PC, 매킨토시, 텔레비전 세트에서 게임을 플레이할 수 있을 것이고, 여행중일 때나, 충분히 낮은-레이턴시 무선 연결로 모바일 폰으로 조차 게임을 플레이할 수 있다. 그리고, 이는 물리적으로 게임의 카피의 소유없이도 모두 달성될 수 있다.

[0223] 앞서 언급했던 것처럼, 사용자는 그 밖의 사람이 그의 게임 플레이를 볼 수 있도록 허락하지 않을 수 있고, 지연된 후에 그의 게임을 볼 수 있도록 할 수 있으며, 선택된 사용자에 의해서 그의 게임을 볼 수 있도록 할 수 있거나, 모든 사용자에 의해서 그의 게임을 볼 수 있도록 할 수 있다. 그럼에도 불구하고, 일 실시예에서 지연 버퍼(1515)에서 15분 동안, 비디오/오디오는 저장될 것이고, 사용자는 "다시 재생"할 수 있을 것이고 그의 예전 게임 플레이를 볼 수 있을 것이며, 일시정지, 천천히 그것을 되돌려 재생하기, 빠르게 순방향 재생하는 등, 그가 디지털 비디오 리코더(DVR)로 TV를 시청하면서 할 수 있는 것처럼 할 수 있을 것이다. 비록, 이러한 예에서, 사용자는 게임을 플레이하고, 만약 사용자가 어플리케이션을 이용한다면, 동일한 "DVR" 성능이 이용가능하다. 이는 사전 작업을 검토하는데 도움을 줄 수 있고 아래 설명되는 것처럼 그 밖의 어플리케이션에 도움이 될 수 있다. 더욱이, 만약 게임이 이용가능한 게임 상태 정보에 기초하여 다시 재생되는 성능을 갖도록 디자인되면, 카메라 시야가 변화될 수 있고, 그 다음 이러한 "3D DVR" 성능은 또한 지원될 것이지만, 그것은 그것을 지원하는 게임이 디자인될 것이 요구될 것이다. 지연 버퍼(1515)를 사용하는 "DVR" 성능은 게임 또는 어플리케이션이 사용된 때 생성되는 비디오를 위해서, 물론 제한적으로 어떤 게임 또는 어플리케이션과 함께 동작할 것이지만, 그것은 3D DVR 성능을 갖는 게임의 경우에 사용자는 "미리 플레이된 세그먼트의 3D에서 "플라이 쓰루(fly through)"를 제어할 수 있고, 결과 비디오를 지연 버퍼(1515)가 저장하고, 게임 세그먼트 기록의 게임 상태를 가진다. 따라서, 특정 "플라이 쓰루"는 압축된 비디오로서 기록될 것이지만, 게임 상태가 또한 기록될 것이기

때문에, 다른 플라이 쓰루는 게임의 동일한 세그먼트의 이후 날짜에서 가능하게 될 것이다.

[0224] 아래 설명되는 것처럼, 호스팅 서비스(210)에 사용자는 각각의 유저 페이지를 가질 것이고, 그들은 스스로에 대한 정보 및 그 밖의 데이터를 알려줄 수 있다. 사용자가 알려줄 수 있는 것들 중에서는 그들이 저장되는 게임 플레이로부터 비디오 세그먼트이다. 예컨대, 만약 사용자가 게임에서 특정 어려운 도전을 극복한다면, 사용자는 게임에서 그들의 위대한 성취를 이룬 지점 바로 전으로 "되돌려 감도록" 할 수 있고, 그 다음 그 밖의 사용자가 사용자의 유저 페이지에서 볼 수 있도록 어떤 지속 기간(예컨대, 30초)의 비디오 세그먼트를 저장하도록 호스팅 서비스(210)에 명령한다. 이를 실행하기 위해서, 어플리케이션/게임 서버(1521 - 1525)의 문제는 사용자가 지연 버퍼(1515)에 저장된 비디오를 재생하기 위해 RAID 배열(1511 - 1512)를 사용할 것이고 그 다음 사용자의 유저 페이지에 비디오 세그먼트를 색인에 올린다.

[0225] 만약 게임이 3D DVR의 성능을 갖는다면, 앞서 설명한 것처럼, 그 다음 3D DVR을 위해 요청되는 게임 상태 정보는 사용자에 의해서 기록될 수 있고 사용자의 유저 페이지에 이용할 수 있도록 만들어질 수 있다.

[0226] 게임이 "관객(spectator)"(예컨대, 사용자가 3D 월드를 통해서 여행을 할 수 있고 그것에 참여없이 동작을 관찰한다)을 갖도록 디자인된 이벤트에서, 활동적인 플레이어를 추가하고, 게임 파인더 어플리케이션은 사용자가 플레이어 뿐만아니라 관객으로써 게임에 참여할 수 있도록 할 것이다. 실행 관점에서, 사용자가 활동적인 플레이어 대신에 관객이라도 호스팅 시스템(210)에는 차이가 없다. 게임은 어플리케이션/게임 서버(1521 - 1525)으로 로드될 것이고 사용자는 게임을 조정할 것이다(예컨대, 그 세상을 보도록 가상 카메라를 제어하는). 사용자의 게임 경험이 유일한 차이일 것이다.

[0227] **복수의 사용자 협력(Multiple User Collaboration)**

[0228] 호스팅 서비스(210)의 다른 특징은 감상하기 위해서 매우 다른 장치를 사용함에도, 복수의 사용자가 라이브 비디오를 감상하는 동안에 협력할 수 있다는 것이다. 이는 게임을 플레이할 때 및 어플리케이션을 사용할 때 유용한다.

[0229] 많은 PC 및 모바일 폰은 비디오 카메라가 장착되어 있고 특히 이미지가 작을 때, 실시간 비디오 압축을 할 수 있는 성능이 있다. 또한, 작은 카메라는 텔레비젼에 부착되도록 이용될 수 있고, 그것은 비디오를 압축하기 위해서 소프트웨어 또는 많은 하드웨어 압축 장치를 사용하여 실시간 압축하는 것이 어렵지 않다. 또한, 많은 PC 및 모든 모바일 폰은 마이크로폰을 갖고 있고, 헤드셋은 마이크로폰으로 이용할 수 있다.

[0230] 로컬 비디오/오디오 압축 성능과 결합된(특히 여기서 설명된 낮은-레이턴시 비디오 압축 기술을 채용하는) 이러한 카메라 및/또는 마이크로폰은, 사용자 구내(211)에서 호스팅 서비스(210)까지 사용자가 비디오 및/또는 오디오를 입력 장치 제어 데이터와 함께 전송할 수 있을 것이다. 그러한 기술이 채택된 때, 다음 도 19에 도시된 성능치가 달성될 수 있고 : 사용자는 다른 사용자의 게임 또는 어플리케이션내에 스크린에 나타나는 그의 비디오 및 오디오(1900)를 가질 수 있다. 이러한 예는 자동차 경주에서 팀별 경쟁을 하는 멀티플레이어 게임이다. 한 사용자의 비디오/오디오는 오직 그들의 팀에게만 선택적으로 볼 수 있고/들을 수 있다. 그리고, 거기에는 효과적으로 레이턴시가 없기 때문에, 상기 설명된 기술을 사용하는 것은 사용자가 인식할 수 있는 지연없이 실시간으로 서로 간에 대화 또는 동작을 만들 수 있을 것이다.

[0231] 비디오/오디오 통합은 사용자의 카메라/마이크로폰에서 인바운드 인터넷 트래픽(1501)에 도착한 압축된 비디오 및/또는 오디오를 가짐으로써 달성될 수 있다. 인바운드 라우팅(1502)은 비디오 및/또는 오디오를 비디오 및/또는 오디오를 보고/듣는 것이 허용되는 어플리케이션/게임 게임 서버(1521 - 1525)로 전송한다. 비디오 및/또는 오디오를 사용하도록 선택된 각각의 어플리케이션/게임 게임 서버(1521 - 1525)의 사용자는 그것을 압축하고 1900에 의해서 도시된 것처럼, 게임 또는 어플리케이션 내에 나타내기 원하는 것처럼 통합한다.

[0232] 도 19의 예는 그러한 협력이 얼마나 게임에서 사용되는지를 도시하지만, 그러한 협력은 어플리케이션을 위해서 막대한 파워있는 도구일 수 있다. 큰 건물이 뉴욕에 있는 부동산 개발업자를 위해 시카고의 건축가에 의해 뉴욕시를 위해 설계되는 상황을 고려하면, 결정은 여행중이어서 공교롭게도 마이애미 공항에 있는 재정투자자를 포함하고, 투자자 및 부동산 개발자를 만족시키기 위해서, 결정은 가까이에 있는 빌딩과 어떻게 조화되는지에 대하여 빌딩의 디자인 요소에 대해 만들어지는 것이 필요하다. 건설 회사는 시카고에서 PC에 부착된 카메라가 있는 높은 해상도 모니터를 보유하고, 부동산 개발자는 뉴욕에 카메라가 있는 랩탑을 보유하며, 투자자는 마이애미에서 카메라가 있는 모바일 폰을 보유한다. 건설 회사는 매우 현실적인 3D 랜더링 성능이 있는 강력한 건축 디자인 어플리케이션을 호스트하기 위한 호스팅 서비스(210)를 사용할 수 있고, 디자인 하에 빌딩의 데이터베이

스 뿐만아니라 뉴욕 시에 있는 빌딩의 많은 데이터베이스를 사용할 수 있다. 건축 디자인 어플리케이션은 하나 실행될 것이나, 어플리케이션/게임 서버(1521 - 1525)의 여러 개에서는 많은 컴퓨터 파워가 요구된다. 개별적인 위치에 3 사용자 각각은 호스팅 서비스(210)에 연결될 것이고, 각각은 건축 디자인 어플리케이션의 비디오 출력을 동시에 볼 수 있을 것이지만, 그것은 각각의 사용자가 가지고 있는 주어진 장치 및 네트워크 연결 특성(예컨대 건축 회사는 상업 인터넷 연결 20Mbps를 통해서 2560×1440 60fps 디스플레이, 뉴욕에 부동산 개발자는 그의 랩탑에서 6 Mbps DSL 연결을 통해서 1280×720 60fps 이미지를 볼 수 있고, 투자자는 그녀의 모바일 폰에 250Kbps 셀룰러 데이터 연결을 거쳐 320×180 60fps 이미지를 볼 수 있을 것이다)때문에 공유된 하드웨어 압축(1520)에 의해서 적당한 치수로 만들어질 것이다. 각각의 단체는 그 밖의 단체의 소리를 들을 것이고(회의 전화는 어플리케이션/게임 서버(들) (1521 - 1525)에 어떤 많은 이용가능한 회의 전화 소프트웨어 패키지에 의해서 다뤄질 것이다), 사용자 입력 장치의 버튼 동작에 따라, 사용자는 그들의 로컬 카메라를 사용하여 스스로를 나타내는 비디오를 만들 수 있을 것이다. 모임이 진행됨에 따라, 건축가는 각각의 단체의 디스플레이 장치의 해상도에 맞게 모든 단체가 볼 수 있는 동일한 비디오 및 매우 포토리얼리즘 3D 랜더링으로 빌딩을 회전하는 것처럼 보이게 하고, 다른 빌딩이 있는 영역의 옆으로 그것을 날라온 것처럼 보이게 할 수 있을 것이다. 어떤 단체에 의해서 사용되는 로컬 장치 중 아무것도 그러한 리얼리즘을 갖는 3D 애니메이션을 다룰 수 있는 성능이 없다는 것은 문제되지 않고, 뉴욕시의 주변 빌딩을 랜더링하기 위해서 요구되는 광대한 데이터베이스를 저장하거나 다운로드하는 것은 물론이고 문제되지 않는다.

[0233] 각 사용자의 시점에서, 거리가 떨어졌음에도 불구하고, 개별 로컬 장치임에도 불구하고, 그들은 단순히 믿을 수 없는 리얼리즘의 정도로 원활할 경험을 갖게 될 것이다. 그리고, 하나의 단체가 그들의 감정 상태를 더 잘 전달하기 위해서 그들의 얼굴을 원할 때, 그들은 그것을 전달할 수 있다. 더욱이 만약, 부동산 개발자 또는 투자자가 건축 프로그램을 제어하고 싶거나 그들 자신의 입력 장치(이는 키보드, 마우스, 키패드, 또는 터치 스크린)를 사용하고 싶을 때, 그들은 그것을 제어할 수 있고 지각할 수 있는 레이턴시없이 응답할 것이다(그들의 네트워크 연결은 비합리적인 레이턴시를 갖지 않는다고 가정하면). 예컨대, 모바일 폰의 경우에, 만약 모바일 폰이 공항에서 WiFi 네트워크와 연결되면, 그것은 매우 낮은-레이턴시를 가질 것이다. 그러나 만약 미국에서 오늘날 이용가능한 셀룰러 데이터 네트워크를 사용한다면 그것은 아마도 눈에 띄는 랙을 겪을 것이다. 여전히 대부분의 모임의 목적은, 투자자는 셀룰러 레이턴시가 받아들여질 수 있음에도, 건축가가 빌딩 플라이 바이(fly-by)를 제어하는 것을 지켜보거나 비디오 화상 회의로 이야기하기 위함이다.

[0234] 마지막으로, 콜레보레이티브(collaborative) 전화 회의의 끝에는, 부동산 개발자 및 투자가가 그들의 의견(comment)을 만들 것이고 호스팅 서비스를 종료할 것이며, 건축 회사는 지연 버퍼(1515)에 저장된 회의 비디오를 "되감기(rewind)" 및 모임 동안 만들어진 빌딩의 3D 모델에 적용되는 표면상의 표현 및/또는 동작, 의견을 검토할 것이다. 만약 그들이 저장하고 싶은 특별한 세그먼트가 있다면, 비디오/오디오의 세그먼트는 아카이브 스토리지(archival storage) 및 후에 재생을 위해서 지연 버퍼(1515)에서 RAID 배열(1511 - 1512)로 이동될 수 있다.

[0235] 또한, 비용 관점에서, 만약 건축가가 15분 전화 회의를 위해서 컴퓨터 사용 성능 및 뉴욕시의 거대한 데이터베이스를 사용할 필요가 있다면, 그들은 큰 데이터베이스의 비싼 카피를 구비하는 것 및 매우 성능있는 워크스테이션을 소유하는 것보다는 오직 리소스를 사용하는 시간에 대해서만 지불할 필요가 있다.

[0236] **비디오-리치 커뮤니티 서비스(Video-rich Community Service)**

[0237] 호스팅 서비스(210)는 인터넷의 비디오-리치 커뮤니티 서비스를 설치하기 위해 전례가 없는 기회를 가능하게 한다. 도 20은 호스팅 서비스(210)에서 게임 플레이어를 위한 모범적인 유저 페이지를 도시하고 있다. 게임 파인더 어플리케이션과 마찬가지로, 유저 페이지는 어플리케이션/게임 서버(1521 - 1525) 중 하나에서 실행되는 어플리케이션이다. 이 페이지에 있는 모든 섬네일 및 비디오 윈도우는 지속적으로 움직이는 비디오를 도시하고 있다(만약 세그먼트가 짧고, 그들은 반복된다면).

[0238] 비디오 카메라를 사용하거나 동영상을 업로드함으로써, 사용자(어느 사용자의 이름은 "KILLHAZARD")는 다른 사용자가 볼 수 있는 스스로의 비디오(2000)를 게시할 수 있다. 비디오가 RAID 배열(1511 - 1512)에 저장된다. 또한, 다른 사용자가 KILLHAZARD의 유저 페이지에 왔을 때, 만약 그때에 KILLHAZARD가 호스팅 서비스(210)를 사용한다면, 그가 하고 있는 무엇이든지 라이브 비디오(2001)(그를 지켜보기 위해 그의 유저 페이지를 관람하는 사용자를 그가 허용한다고 가정하면)로 보여질 것이다. 이는 KILLHAZARD가 활성화되는지, 만약 활성화된다면, 그가 사용하는 어플리케이션/게임 서버(1521 - 1525)인지 여부에 따라, 서비스 컨트롤 시스템(401)로부터 요청되

는 유저 페이지 어플리케이션을 호스팅하는 어플리케이션/게임 서버(1521 - 1525)에 의해서 달성될 것이다. 그 다음, 같은 방법이 게임 파인더 어플리케이션에 의해서 사용되고, 적당한 해상도로 압축된 비디오 스트림과 포맷은 유저 페이저 어플리케이션이 동작하는 어플리케이션/게임 서버(1521 - 1525)로 전송되고 그것은 디스플레이될 것이다. 만약 사용자가 KILLHAZARD의 라이브 게임 플레이의 윈도우를 선택하면, 그 다음 그들의 입력 장치로 적당히 클릭하고, 윈도우는 확대될 것이다(다시 게임 파인더 어플리케이션과 같은 방법을 사용하여, 관람하는 사용자의 인터넷 연결의 특성에 적절하고, 관람하는 사용자의 디스플레이 장치(422)의 해상도에서, 라이브 비디오가 스크린을 채울 것이다).

[0239] 선행 기술 접근의 이상의 주요 장점은 사용자가 유저 페이지를 보는 것으로 사용자가 소유하지 않는 라이브 게임하는 것을 볼 수 있도록 하고, 게임이 플레이되는 로컬 컴퓨터 또는 게임 콘솔을 가지고 있지 않아도 매우 잘 할 수 있다. 그것은 사용자가 유저 페이지에 도시된 "동작중인(in action)" 플레이되는 게임을 볼 수 있는 좋은 기회를 제공하고, 그것은 관람하는 사용자가 시도해 보거나 더 잘하고자 하는 게임에 대해서 배울 수 있는 기회를 제공한다.

[0240] KILLHAZARD의 친구(buddies ; 2002)로부터 카메라 녹화하거나 업로드된 비디오 클립은 또한 유저 페이지에 표시되며, 각 비디오 클립 아래에 있는 텍스트는 친구가 온라인 게임을 플레이하는지 여부를 나타낸다(예컨대, six_shot은 "Eragon" 게임을 플레이하고 있고 MrSnuggles99은 오프라인, 등). 메뉴 항목(도시되지 않음)을 클릭하여 친구의 비디오 클립이 기록되고 업로드된 비디오를 보는 것에서 그들의 게임 순간에 동작하는 호스팅 서비스(210)에서 게임을 플레이하는 친구의 라이브 비디오로 스위치된다. 따라서, 이는 친구들을 그룹화하는 게임 파인더가 될 것이다. 만약 친구의 게임이 선택되고 그것이 클릭되면, 그것은 전체 화면으로 확대될 것이고, 사용자가 전체 화면으로 플레이되는 게임을 라이브로 볼 수 있을 것이다.

[0241] 반복해서, 친구의 게임을 관람하는 사용자는 게임의 카피 또는, 게임을 플레이하기 위해서 로컬 컴퓨팅/게임 콘솔 리소스를 소유하지 않아도 된다. 게임을 관람하는 것은 효과적으로 즉시적인 것이다.

[0242] 앞서 설명된 것처럼, 사용자가 호스팅 서비스(210)에서 게임을 플레이할 때, 사용자가 게임을 "되감기(rewind)"할 수 있고 그가 저장하고자 하는 비디오 세그먼트를 찾을 수 있으며, 그의 유저 페이지에 비디오 세그먼트를 저장할 수 있다. 이는 "브래그 크립(Brag Clips)"이라고 불린다. 비디오 세그먼트(2003)는 그가 플레이한 사전 게임으로부터 KILLHAZARD에 의해 저장되는 모든 브래그 클립(2003)이다. 번호(2004)는 브래그 클립이 얼마나 많은 시간 관람되었는지를 도시하고, 브래그 클립이 보여질 때, 사용자가 그들을 평가하기 위한 기회를 갖게 되고, 다수의 오렌지색 키홀 형태의 아이콘(2005)은 얼마나 높은 등급을 나타내는지 보여준다. 브래그 클립(2003)은 사용자가 유저 페이지를 볼 때, 페이지에 남은 비디오에 따라 지속적으로 루프한다. 사용자가 브래그 클립(2003) 중 하나를 선택하고 클릭하면, 그것은 재생하기(play), 일시정지(pause), 되감기(rewind), 고속 감기(fast-forward), 스텝 쓰루(stepped through) 등이 되도록 클립을 허용하는 DVR 컨트롤에 따라 브래그 클립(2003)을 나타내도록 확대한다.

[0243] 브래그 클립(2003) 재생은 사용자가 브래그 클립을 기록하고, 그것을 신장하며 다시 그것을 플레이할 때 RAID 배열(1511 - 1512)에 저장된 압축된 비디오 세그먼트를 로딩하는 어플리케이션/게임 서버(1521 - 1525)에 의해서 구현된다.

[0244] 또한, 브래그 클립(2003)은 그러한 기능을 지원하는 게임에서 "3D DVR" 비디오 세그먼트(예컨대, 카메라 시점을 사용자가 바꿀 수 있도록 허락하고 리플레이될 수 있는 게임에서 게임 상태 시퀀스)일 수 있다. 이 경우의 게임 상태 정보가 게임 세그먼트가 기록된 때 사용자가 만드는 특정한 "플라이 쓰루"의 압축된 비디오 기록을 추가하여 저장된다. 유저 페이지가 보여지고 있을 때, 모든 섬네일 및 비디오 윈도우는 지속적으로 루프될 것이고, 3D DVR 브래그 클립(2003)은 사용자가 게임 세그먼트의 "플라이 쓰루"를 기록했을 때, 압축된 비디오처럼 기록된 브래그 클립(2003)을 지속적으로 루프할 것이다. 그러나 사용자가 3D DVR 브래그 클립(2003)을 선택하고 그것을 클릭할 때, 압축된 비디오 브래그 클립이 플레이되도록 허용하는 DVR 컨트롤을 추가하고, 사용자는 게임 세그먼트를 위한 3D DVR 기능을 그들에게 주는 버튼을 클릭할 수 있을 것이다. 그들은 자신의 게임 세그먼트 동안에 카메라 "플라이 쓰루"를 제어할 수 있게 될 것이고, 만약 그들이 원한다면(그리고 유저 페이지를 가진 사용자가 그것을 허용하면), 그들은 압축된 비디오 형태로 택일적인 브래그 클립 "플라이 쓰루"를 기록할 수 있을 것이고 그 다음 유저 페이지의 다른 관찰자에게 이용가능하도록 할 것이다(즉시, 또는 유저 페이지의 소유자가 브래그 클립을 검토하기 위한 기회를 갖은 후에).

[0245] 이러한 3D DVR 브래그 클립(2003) 성능은 다른 어플리케이션/게임 서버(1521 - 1525)에서 기록된 게임 상태 정보를 리플레이하는 게임을 활성화하도록 할 수 있다. 게임이 거의 즉시 활성화될 수 있기 때문에(앞서 설명된

것처럼), 브래그 클립 세그먼트에 의해서 기록된 게임 상태로 제한된 그 게임을 활성화하는 것은 어렵지 않고, 그 다음 압축된 비디오를 지연 버퍼(1515)로 기록하는 동안에 카메라로 "플라이 쓰루"하도록 사용자를 허용한다. 일단 사용자가 실행을 완료하면, 그 "플라이 쓰루" 게임은 비활성화된다.

[0246] 사용자의 시점에서, 3D DVR 브래그 클립(2003)을 가진 활성화된 "플라이 쓰루"는 선형 브래그 클립(2003)의 DVR 컨트롤을 제어하는 노력이 필요 없게 된다. 그들은 어떻게 게임을 플레이하는지 또는 그 게임에 대해서 아무것도 모를 것이다. 그들은 단지 다른 사람에 의해서 기록된 게임 세그먼트 동안에 3D 세상을 살펴보는 가상 카메라 오퍼레이터이다.

[0247] 또한 사용자는 마이크로폰으로부터 기록되거나 업로드된 것을 브래그 클립으로 그들 자신의 오디오로 다중으로 녹음할 수 있을 것이다. 이러한 방법에서, 브래그 클립은 게임으로부터 캐릭터 및 동작을 사용하는 커스텀 애니메이션(custom animation)을 만드는데 사용할 수 있다. 이러한 애니메이션 기술은 흔히 "머시니마(machinima)"로써 알려진다.

[0248] 사용자가 게임을 진행함에 따라, 그들은 기술 수준이 서로 다르게 달성할 것이다. 플레이되는 게임은 서비스 제어 시스템(401)에 업적을 보고하고, 이러한 기술 수준은 유저 페이지에 게재된다.

[0249] **인터랙티브 애니메이션 광고(Interactive Animated Advertisements)**

[0250] 온라인 광고가 텍스트에서 변화되어 왔고, 여전히 이미지로, 비디오로, 지금은 인터랙티브 세그먼트로, 전형적으로 Adobe Flash 같은 씬 클라이언트(thin client)는 애니메이션을 사용하여 실행해 왔다. 씬 클라이언트가 애니메이션을 사용하는 이유는 사용자가 일반적으로 그들에게 판매된 서비스 또는 상품이 가지는 특권에 대해 지연되는 조금의 인내심을 갖는다는 것이다. 또한, 씬 클라이언트는 계속 매우 낮은 성능 PC으로 동작하고, 광고주는 인터랙티브 광고가 적당히 작용할 수 있다는 높은 정도의 확신을 가질 수 있다. 불행하게도, Adobe Flash와 같은 애니메이션 씬 클라이언트는 인터랙티비티(interactivity) 및 경험의 지속 기간(다운로드 시간을 완화하기 위해서)의 정도로 제한이 있다.

[0251] 도 21은 사용자가 전시장에서 자동차 주위를 회전하는 동안에, 자동차의 외부 및 내부 색을 선택할 수 있고, 실시간 레이 트레이싱(ray tracing)이 자동차가 어떻게 보이는지를 보여주는 인터랙티브 광고를 도시하고 있다. 그 다음 사용자가 그 자동차를 운전하기 위한 아바타(avatar)를 선택한 후에, 사용자는 레이스 트랙에 또는 모나코와 같은 이국적인 현지로 드라이브를 위해서 자동차를 가져갈 수 있다. 사용자는 더 큰 엔진, 또는 더 나은 타이어를 선택할 수 있고, 변경된 구성으로 자동차가 가속하고 노면에 착 붙어서 달리는 성능에 어떠한 영향을 주는지를 볼 수 있다.

[0252] 물론, 그 광고는 효과적인 정교한 3D 비디오 게임이다. 그러나 이러한 광고가 PC 또는 비디오 게임 콘솔에서 플레이될 수 있으려면 그것은 아마도 100MB의 다운로드가 필요할 것이고, PC의 경우에 그것은 특별한 드라이버의 설치가 필요할 수 있으며, 적절한 CPU나 GPU의 컴퓨팅 성능이 부족하다면 동작하지 않을 수도 있다. 따라서, 이러한 광고는 종래 기술 구성에서는 비현실적이다.

[0253] 호스팅 서비스(210)에서, 이러한 광고는 거의 즉시로 시작하고, 완벽하게 실행되며, 사용자의 클라이언트(415) 성능이 무엇인지는 문제되지 않는다. 그래서 그들은 씬 클라이언트 인터랙티브 광고보다 더 빠르고, 그 경험이 보다 풍부하며, 매우 신뢰할 수 있다.

[0254] **실시간 애니메이션 동안 스트리밍 지오메트리(Streaming Geometry During Real-time Animation)**

[0255] RAID 배열(1511 - 1512) 및 인바운드 라우팅(1502)은 실시간 애니메이션(예컨대, 복잡한 데이터 베이스로 플라이-쓰루) 동안에 어플리케이션에서 또는 게임 플레이의 도중에 즉시 지오메트리를 확실하게 전달하기 위해서 RAID 배열(1511 - 1512) 및 인바운드 라우팅(1502)에 의존하는 비디오 게임 및 어플리케이션을 디자인하도록 너무 낮은 레이턴시를 갖고 너무 빠른 전송률을 제공할 수 있다.

[0256] 도 1에 도시된 비디오 게임 시스템과 같은 종래 기술의 시스템으로, 거대한 스토리지 장치가 이용가능하고, 특히 실제 가정에서 장치는, 요구되는 지오메트리가 다소 예상 가능한 상황을 제외하고는 너무 느려서 게임 플레이하는 동안에 지오메트리를 스트림할 수 없다. 예컨대, 특별한 도로에서 드라이빙하는 게임은, 시야로 들어오는 빌딩을 위한 지오메트리가 합리적으로 잘 예상될 수 있고 거대한 스토리지 장치는 곧 다가올 지오메트리가

위치한 위치를 미리 찾을 수 있다.

[0257] 그러나 예상할 수 없는 변화를 가진 복잡한 장면에서(예컨대, 주변에 복잡한 캐릭터를 갖는 전투 장면에서) 만약 PC 또는 비디오 게임 시스템에 RAM이 현 시야의 물체를 위한 지오메트리로 완전히 채워진다면, 사용자는 갑자기 그들의 캐릭터뒤쪽의 것을 보기 위해 그들의 캐릭터를 돌리고, 만약 지오메트리가 RAM에 미리 로드되지 않는다면, 그것을 디스플레이되는데 지연이 있을 것이다.

[0258] 호스팅 서비스(210)에서 RAID 배열(1511 - 1512)은 기바비트 이더넷 속도를 초과하여 데이터를 스트림할 수 있고, SNA 네트워크를 가지며, 그것은 10 기가비트 이더넷 또는 그 밖의 네트워크 기술보다 초당 10기가비트율을 달성하는 것이 가능하다. 초당 10기가비트는 초당 적어도 1기가바이트의 데이터를 로드할 것이다. 60fps 프레임 시간에서(16.67ms), 대략 170 메가비트(21MB)의 데이터가 로드될 수 있다. 물론 RAID 구성에서조차 회전 미디어는 여전히 한 프레임 시간보다 더 큰 레이턴시가 일어날 것이나, 플래시 기반의 RAID 스토리지는 결국 회전 미디어 RAID 배열만큼 클 것이고 이러한 높은 레이턴시가 일어나지는 않을 것이다. 일 실시예에서, 대규모 RAM 라이트-쓰루 캐싱(write-through caching)은 매우 낮은-레이턴시 접근을 제공할 것이다.

[0259] 따라서, 충분히 높은 네트워크 속도로, 그리고 충분히 낮은 충분한 레이턴시를 갖는 대량 스토리지, 지오메트리는 CPU 및/또는 GPU가 3D 데이터를 처리할 수 있는 만큼 빠르게 어플리케이션/게임 게임 서버(1521 - 1525)로 스트림될 수 있다. 그래서, 앞서 주어진 예에서, 사용자가 갑작스럽게 그들이 캐릭터를 회전하고 뒤쪽을 보기위해서, 캐릭터의 회전이 완료되기 전에 뒤에 있는 모든 캐릭터를 위한 지오메트리가 로드될 수 있고, 따라서, 사용자에게 그것은 라이브 액션처럼 생생하게 포로리얼리즘 세상에 그 또는 그녀가 있는 것처럼 보일 것이다.

[0260] 앞서 논의한 것처럼, 포토리얼리즘 컴퓨터 애니메이션에 마지막 한계 중의 하나는 인간의 얼굴이고, 인간의 눈의 민감도가 완벽하지 않기 때문에, 포토리얼 페이스로부터 근소한 에러는 관찰자로부터 부정적인 반응을 초래할 수 있다. 도 22는 Contour™ Reality Capture Technology(코-펜딩(co-pending) 어플리케이션의 주제 : "실행자의 동작을 캡처하는 방법 및 장치" 10/942,609, 2004년 9월15일 출원; " 실행자의 표현을 캡처하기 위한 방법 및 장치", 10/942,413, 2004년9월15일 출원; "동작 캡처 시스템 내에 마커 식별을 향상하기 위한 방법 및 장치" 11/066,954, 2005년 2월 25일 출원; "셔터 동기화를 사용하여 동작 캡처를 수행하는 방법 및 장치", 1/077,628, 2005년 3월 10일 출원; "캡처 표면에 임의의 패턴을 사용하여 동작을 캡처하는 방법 및 장치" 11/255,854, 2005년 10월 20일 출원; "형광체 응용 기술을 사용하여 동작 캡처를 수행하는 방법 및 시스템", 11/449,131, 2006년 6월 7일 출원; "형광 램프 섬광효과를 이용하여 동작 캡처를 수행하는 시스템 및 방법", 11/449,043, 2006년 6월 7일 출원; "정지 동작 애니메이션 캐릭터의 3차원 캡처를 위한 방법 및 시스템", 11/449,127, 2006년 6월 7일 출원, 이러한 각각의 것은 현재 CIP 출원의 양수인으로 양수되었다)를 사용하여 어떻게 라이브 성능을 매우 부드러운 캡처 표면에 초래하고, 그 후에 표면에 트랙된 높은 폴리건-카운트에 관하여 도시하고 있다(예컨대, 폴리건 동작은 얼굴의 동작을 정확하게 따른다). 마침내, 라이브 퍼포먼스의 비디오가 텍스쳐 표면(textured surface)을 생성하기 위해서 트랙된 표면(tracked surface)에 맵핑되고 포토리얼 결과가 생성된다.

[0261] 비록 현재 GPU 기술이 다수의 트랙된 표면에서 폴리건 및 표면 텍스쳐 및 라이트를 실시간으로 랜더링할 수 있고, 만약 폴리건 및 텍스쳐가 매 프레임 시간에 변한다면(이는 가장 포토리얼한 결과를 생성할 것이다), 그것은 빠르게 현대 PC 및 비디오 게임 콘솔의 이용가능한 RAM을 소모할 것이다.

[0262] 상기 설명된 스트리밍 지오메트리 기술을 사용하면, 그것은 실제적으로 어플리케이션/게임 게임 서버(1521 - 1525)로 연속적으로 지오메트리를 공급하게 되므로 그들은 연속적으로 포토리얼 얼굴을 생기있게 할 수 있고, 얼굴로 비디오 게임을 창조하게되면 거의 살아서 동작하는 얼굴과 구별할 수 없다.

[0263] **인터랙티브 기능을 가진 선형 컨텐츠의 통합(Integration of Linear Content with Interactive Features)**

[0264] 영화, TV 프로그램 및 오디오 자료 ("선형 컨텐츠"는 많은 형태로 홈 및 오피스 사용자에게 널리 이용될 수 있다) 선형 컨텐츠는 CD, DVD, HD-DVD 및 Blue-ray 미디어와 같은 물리적 미디어에서 획득될 수 있다. 그것은 또한 위성 및 케이블 TV 방송에서 DVRs로 기록될 수 있다. 그리고, 그것은 위성 및 케이블 TV와 케이블 TV에 주문형 비디오 (VOD)를 통해 유료보기(pay-per-view; PPV) 컨텐츠로써 이용할 수 있다.

[0265] 점점 선형 컨텐츠가 다운로드 및 스트리밍 컨텐츠로써 인터넷을 통해서 이용할 수 있다 오늘날에는 선형 미디어와 관련된 모든 특징을 경험하기 위한 하나의 장소가 있지 않다. 예컨대, DVD 및 기타 비디오 광학 미디어는 일

반적으로 특작 단편영화(featurettes)를 "구성하는(making of)" 감독의 논평처럼, 그 밖의 곳에서는 이용할 수 없는 인터랙티브 특징을 갖는다. 온라인 음악 사이트는 일반적으로 CD에서 이용하지 않는 표지 미술 및 노래 정보를 가지고 있지만, 모든 CD가 온라인으로 이용할 수 있는 것은 아니다. 텔레비전 프로그래밍과 관련된 웹 사이트는 종종 그 밖을 특징, 블로그, 때로는 배우 또는 스태프로부터 논평을 보유한다.

[0266] 더욱이, 많은 동영상 또는 스포츠 이벤트는, 자주 출시되는 비디오 게임이 있고(동영상의 경우에), 자주 선형 미디어와 함께 하거나(스포츠의 경우에) 실세계 이벤트와 면밀하게 연결될 것이다(예컨대, 선수의 거래 등).

[0267] 호스팅 서비스(210)는 관련된 컨텐츠의 개별 형태와 함께 연결되는 선형 컨텐츠의 전달을 위해 매우 적합하다. 물론, 제공하는 동영상이 더 이상 그 전달하는 고도의 인터랙티브 비디오 게임을 요구하지는 않을 것이고, 호스팅 서비스(210)는 홈 또는 오피스의 광범위한 장치 또는 모바일 장치로 선형 컨텐츠를 전달할 수 있다. 도 23은 선형 컨텐츠의 선택을 보여주는 호스팅 서비스(210)를 위한 모범적인 사용자 인터페이스 페이지를 도시하고 있다.

[0268] 하지만, 대부분의 선형 컨텐츠 전달 시스템과는 달리, 호스팅 서비스(210)는 또한 관련된 인터랙티브 구성요소(예컨대, 웹 사이트 상에 Adobe Flash 애니메이션(아래 설명하는 것처럼), HD-DVD에서 인터랙티브 오버레이, 및 DVD에서 메뉴 및 기능)를 전달할 수 있다. 따라서, 클라이언트 장치(415)는 그것의 기능을 이용할 수 있는 것에 관하여 더 이상 제한을 소개하지 않는다.

[0269] 또한, 호스팅 시스템(210)은 실시간으로 동적인 비디오 게임 컨텐츠와 선형 컨텐츠를 함께 연결할 수 있다. 예컨대, 사용자가 해리 포터 영화에 나오는 퀴디치(Quidditch) 경기를 보고 있다면, 그녀는 버튼을 클릭할 수 있고 영화는 정지될 것이며 즉시 그녀는 해리 포터 비디오 게임의 퀴디치 세그먼트로 이동되어질 것이다. 퀴디치 경기를 플레이한 후에, 다른 버튼을 클릭하고, 영화는 즉시 다시 시작할 것이다.

[0270] 포토리얼 그래픽 및 생산 기술로 포토그래픽하게 캡처한 비디오는 라이브 액션 캐릭터와 구별할 수 없고, 여기서 설명된 것처럼 사용자가 라이브 액션 영화에 퀴디치 게임에서 호스팅 서비스에 있는 비디오 게임에 퀴디치 게임으로 이동을 만들 때, 사실상 두 장면은 구별할 수 없다. 이는 선형 컨텐츠와 인터랙티브 컨텐츠(예컨대, 비디오 게임)의 감독들이 두 세계의 차이를 구별할 수 없게 되는 새로운 창조적인 옵션을 제공한다.

[0271] 도 14에 도시된 호스팅 서비스 아키텍처를 활용하면, 3D 영화에서 버추얼 카메라의 제어는 관찰자에게 제공될 수 있다. 예컨대, 열차 내에서 이루어지는 장면에서, 그것은 이야기 진행 동안에 열차 주변을 둘러보고, 사용자가 버추얼 카메라를 제어하는 것을 허용하는 것이 가능할 것이다. 이는 원래 영화만큼이나 실시간으로 장면을 랜더링할 수 있는 컴퓨터 파워의 적당한 수준만큼 그 열차에 있는 모든 3차원 객체의 ("자산(assets)")으로 이용할 수 있다.

[0272] 심지어 비-컴퓨터에 의해 생성된 엔터테인먼트에서, 제공될 수 있는 매우 흥미로운 인터랙티브 기능이 있다. 예컨대, 2005년 영화 "오만과 편견"은 화려한 고대 영국 저택에 대한 많은 장면이 있다. 특정 맨션 장면에서, 사용자가 비디오를 일시 중지할 수 있고 그 다음 저택 또는 그 주변 지역의 투어를 위해 카메라를 제어할 수 있다. 이를 실행하기 위해서, Apple사의 종래 기술인, 그것의 위치를 추적하는 물고기 눈 렌즈를 갖는 카메라가 저택을 통과하여 운반될 수 있겠다. QuickTime VR이 실행된다. 다양한 프레임이 변환되고 이미지가 왜곡되지 않으며, 영화와 함께 RAID 배열(1511 -1512)에 저장되며, 가상 투어를 위해 사용자가 선택을 할 때, 다시 재생된다.

[0273] 농구 경기와 같은 스포츠 이벤트, 라이브 스포츠 이벤트는 그들이 일반 TV에서처럼, 사용자가 시청하기 위해 호스팅 서비스(201)을 통해서 스트림될 것이다. 사용자가 특정 경기를 시청한 후에, 그 게임의 비디오 게임(결국 실제 플레이어만큼 포토리얼하게 보이는 농구 플레이어)은 동일한 위치에서 시작하는 플레이어로 시작하고, 사용자(아마도 각각은 하나의 플레이어의 제어)는 그 플레이어보다 더 잘할 수 있는지를 확인하기 위해서 이를 다시 실행할 수 있다.

[0274] 여기서 설명된 호스팅 서비스(210)는 미래 세계를 지원하기 위해 매우 적합하다 왜냐하면 그것이 홈 또는 대부분의 오피스 세팅에서 설치되는 것이 비현실적인 컴퓨팅 파워 및 대용량 스토리지 리소스를 견딜 수 있도록 해주기 때문이고, 또한 홈 세팅에서, 항상 구세대의 PC와 비디오 게임이 있는 것에 비하여 그것은 컴퓨팅 자원을 항상 최신의 것으로 하며, 최신의 컴퓨팅 하드웨어가 이용될 수 있고, 호스팅 서비스(210)에서 이 모든 컴퓨팅의 복잡성은 사용자로부터 숨겨지며, 비록 그들이 볼 때 사용자의 입장에서 매우 정교한 시스템을 사용 중일지라도, 사용자의 관점에서 그것은 텔레비전에 채널 변경만큼 단순하다. 더욱이, 사용자가 모든 컴퓨팅 파워에 접근할 수 있고, 컴퓨팅 파워에 대한 경험은 어떤 클라이언트(415)로부터 가져올 수 있을 것이다.

[0275] **멀티플레이어 게임(Multiplayer Games)**

[0276] 게임이 멀티플레이어 게임인 경우, 인바운드 라우팅(1502) 네트워크를 통해 어플리케이션/게임 게임 서버(1521 - 1525)와 인터넷과 연결된 네트워크, 호스팅 서비스(210)에서 동작하지 않는 게임 머신을 연결할 수 있다. 일반적인 인터넷상에서 컴퓨터로 멀티플레이어 게임을 할 경우, 어플리케이션/게임 게임 서버(1521 - 1525)는 인터넷에 매우 빠르게 접근할 수 있다는 장점을 가지나(가정에서 서버로 게임을 진행하는 경우와 비교해서), 더 느린 연결로 플레이하는 다른 컴퓨터의 성능에 의해 제한될 수 있고, 또한 잠재적으로 최소한 공통 분모를 할당하도록 설계된 인터넷의 게임 서버, 이는 대체로 느린 소비자 인터넷 연결의 홈 컴퓨터일 것이라는 사실에 의해서 제한될 수 있다.

[0277] 그러나 멀티플레이어 게임이 전적으로 호스팅 서비스(210) 서버 센터에서 진행될 경우, 다른 세상이 달성될 수 있다. 사용자를 위해 게임을 호스팅하는 각각의 어플리케이션/게임 게임 서버(1521 - 1525)는 매우 높은 스피드, 매우 낮은-레이턴시와 광대하고 매우 빠른 스토리지 배열을 가진 멀티플레이어 게임을 위해 센트럴 컨트롤을 호스팅하는 어떤 서버뿐만 아니라 다른 어플리케이션/게임 게임 서버(1521 - 1525)와 연결될 수 있을 것이다. 예컨대, 만약 기가비트 이더넷이 인바운드 라우팅(1502) 네트워크로 사용된다면, 어플리케이션/게임 게임 서버(1521 - 1525)는 각각의 서버와 통신하고 잠재적으로 1ms레이턴시 또는 그보다 적은 레이턴시로 초당 기가바이트에서 멀티플레이어 게임을 위한 센트럴 컨트롤을 호스팅하는 어떤 서버와 통신할 것이다. 더욱이, RAID 배열(1511 - 1512)은 매우 빠르게 대응할 수 있고 초당 기가비트율로 데이터를 전송한다. 예컨대, 만약 사용자가 외양 및 군장(accoutrements)으로 캐릭터를 커스터마이즈(customize)하면 그 캐릭터는 독특한 행동 및 많은 지오메트리를 가지게 되고, 홈 PC 또는 게임 콘솔을 동작하는 게임 클라이언트로 제한된 종래 기술 시스템으로는, 만약 그 캐릭터가 다른 사용자의 시야로 들어오면, 그 사용자는 길고 느린 다운로드 완료까지를 기다려서 모든 지오메트리 및 행동(behaviors) 데이터를 그들의 컴퓨터로 로드한다. 호스팅 서비스(210)내에서, 동일한 다운로드는 초당 기가비트의 속도로 RAID 배열(1511 - 1512)에서 제공되는 기가비트 이더넷을 거칠 수 있다. 홈 사용자는 8Mbps 인터넷 연결을 가진다면(이는 오늘날의 표준에서 매우 빠르다), 기가비트 이더넷은 100배 더 빠르다. 그래서 빠른 인터넷 연결을 통해 1분 더 걸리는 것은 기가비트 이더넷을 통해서는 1초 이하가 걸릴 것이다.

[0278] **탑 플레이어 그룹화 및 토너먼트(Top Player Groupings and Tournaments)**

[0279] 호스팅 서비스(210)는 토너먼트에 매우 적합하다. 왜냐하면 어떤 게임이 로컬 클라이언트에서 동작하지 않으면, 거기에는 사용자들에게 부정행위를 할 기회가 있지 않다. 또한, UDP 스트림을 멀티캐스트하기 위한 출력 라우팅(1540)의 성능때문에, 호스팅 서비스(210)는 한번에 청중인 수천 명의 사람에게 메이저 토너먼트를 중계할 수 있다.

[0280] 사실 어떤 비디오 스트림은 너무 인기가 있어서 수많은 사용자가 동일한 스트림을 수신하려고 할 때(예컨대, 메이저 토너먼트의 시청), 그것은 많은 클라이언트 장치(415)에게 대용량 배포를 위한 Akamai 또는 Limelight와 같은 컨텐츠 전달 네트워크(CDN)로 비디오 스트림을 보다 효율적으로 보낼 것이다.

[0281] 비슷한 수준의 효율성은 CDN이 탑 플레이어를 그룹화하는 게임 파인더 페이지를 보여주는데 이용할 때 확보될 수 있다.

[0282] 메이저 토너먼트에서, 생방송 유명 아나운서가 어떤 경기에서 실황 방송을 제공하는데 사용될 수 있다. 비록 많은 수의 사용자가 메이저 토너먼트를 시청하더라도 상대적으로 적은 수가 토너먼트를 플레이할 것이다. 유명 아나운서의 오디오는 토너먼트에서 플레이하는 사용자에게 호스팅하고 토너먼트에서 어떤 게임의 관찰자 모드 카피를 호스팅하는 어플리케이션/게임 서버(1521 - 1525)로 라우트될 수 있고, 오디오는 게임 오디오 외에 다중으로 녹음될 수 있다. 유명 아나운서의 비디오는 게임에 오버레이될 수 있고, 게다가 아마도 관찰자 시점에서 오버레이될 수 있다.

[0283] **웹페이지 로딩의 가속(Acceleration of Web Page Loading)**

[0284] 월드 와이드 웹(World Wide Web)의 주요 전송 프로토콜, 하이퍼텍스트 트랜스퍼 프로토콜(HTTP)은, 오직 비지니스가 고속 인터넷 연결을 갖고, 온라인이었던소비자가 다이얼업 모뎀 또는 ISDN을 사용한 시대에서 이해되고 정

의되었다. 동시에 빠른 연결의 "금본위제(gold standard)"는 동시에 1.5Mbps 데이터를 전송하는 T1 라인이다(예컨대, 양방향으로 같은 데이터의 양을 갖는).

[0285] 오늘날, 상황은 완전히 달라졌다. 많은 발전이 있는 세계에서 DSL 또는 케이블 모뎀을 통하는 평균 가정 연결 속도는 T1라인보다 다운스트림 전송률이 상당히 높다. 사실상 세상의 일부분에서는, 파이버 투 더 커브(fiber to the curb)는 가정으로 50에서 100Mbps만큼 빠른 전송률을 가져다 준다.

[0286] 불행하게도, HTTP는 이러한 극적인 속도 개선을 효과적인 장점으로써 설계되지 않았다(그것은 실현되지도 않았다). 웹사이트는 원격 서버에 파일의 집합체이다. 간단하게, HTTP가 첫번째 파일을 요청하고, 첫번째 파일이 다운로드되는 것을 기다린 후, 두번째 파일을 요청해서 그 파일이 다운로드 되는 것을 기다리는 것이다. 사실, HTTP는 하나의 이상의 "개방 연결(open connection)"을 허락하는데, 예컨대 한번에 요청되는 하나의 파일보다 많은 것을 허용하지만, 표준 합의(agreed-upon standard) 때문에(그리고 과부하로부터 웹서버를 보호하는) 오직 소수의 개방 연결만이 허용된다. 게다가, 웹 페이지의 경로가 구성되어져 있기 때문에, 브라우저는 즉시 다운로드되도록 이용할 수 있는 복수의 동시적인 페이지를 종종 인식하지 못한다(예컨대, 페이지의 어구를 파싱(parsing)한 후에, 다운로드할 필요가 있는 이미지와 같은 새로운 파일들이 나타나게 된다). 따라서, 웹사이트 상의 파일들은 HTTP에서 이용되는 요청-응답 프로토콜 때문에 본질적으로 한번에 하나씩 로드되고, 각 파일이 로드되는 것과 관련하여 100ms레이턴시가 있다(미국에 있는 전형적인 웹 서버의 접근).

[0287] 상대적으로 낮은 속도의 연결 상태로는 파일 자체의 다운로드 시간이 웹페이지를 기다리는 시간보다 우위를 정하기 때문에 많은 문제를 야기하지는 않는다. 그러나, 복잡한 웹페이지로 연결 속도가 증가함에 따라, 특히 문제가 일어나기 시작한다.

[0288] 도 24에서 도시된 예에서는, 전형적인 상업 웹사이트를 도시하고 있다(이 특정 웹사이트는 메이저 운동화 브랜드의 것이다). 이 웹사이트는 54개의 파일이 있다. 그 파일들은 HTML, CSS, JPEG, PHP, JavaScript 및 플래시 파일, 및 비디오 컨텐츠를 포함한다. 전체 1.5Mbytes는 페이지가 활성화 전에 로드되어야 한다(즉, 사용자가 그것을 클릭할 수 있고, 사용할 수 있는 상태). 많은 양의 파일이 있어야 하는 이유는 많다. 첫째로, 그것은 복잡하고 세련된 웹페이지이고, 다른 이유는 그 페이지에 접근하는 사용자의 정보에 기초하여 동적으로 정보를 수집한다(즉, 사용자가 어떤 나라인지, 어떤 언어인지, 사용자가 전에 물품을 구매한 적이 있는지 등등) 그리고 이러한 요소들에 의존하여, 다른 파일들이 다운로드된다. 여전히, 이것은 전형적인 상업 웹페이지이다.

[0289] 도 24는 연결 속도가 증가함에 따라 웹페이지가 열리기 전에 경과되는 시간의 전체량을 도시하고 있다. 1.5Mbps 연결 속도(2401)로, 전통적인 웹 브라우저를 갖는 전통적인 서버를 사용하면, 웹페이지가 열리기까지 13.5초가 걸린다. 12Mbps 연결 속도(2402)로, 로드 시간은 6.5초까지 줄어들거나 약 2배 정도 빨라진다. 그러나 96Mbps 연결 속도(2403)로, 로드 시간은 단지 5.5초까지 줄어든다. 그 이유는 그러한 높은 다운로드 속도에서는, 파일 자체를 다운로드 하는 시간은 미미하지만, 파일당 레이턴시는 대략 각각 100ms로 여전히 남아있고, 이는 54파일*100ms = 5.4초의 레이턴시가 된다. 따라서, 가정에 얼마나 빠른 연결이 있는지는 상관없이, 이 웹사이트는 활성화되기까지는 항상 최소 5.4초는 걸릴 것이다. 다른 요소는 서버-사이드 큐잉이다; 모든 HTTP 요청은 큐잉의 뒤에 더해지고, 따라서 바쁜 서버에서 이것은 웹 서버에서 얻을 수 있는 모든 작은 아이템 때문에 현저한 영향을 가질 것이며, HTTP는 그것의 순서를 기다리기 위해 필요를 요청한다.

[0290] 이러한 이슈를 해결하기 위한 한 방법은 HTTP를 버리거나 재정의하는 것이다. 또는, 웹사이트 소유자가 여러 개의 파일을 단일 파일로 통합시키도록 하는 것이다(예컨대, Adobe Flash 포맷으로). 그러나 실질적인 문제로는 회사뿐만 아니라, 많은 그 밖의 것들이 웹사이트 구축에 상당한 투자를 하는 것이다. 더욱이, 몇몇 가정은 12 - 100Mbps 속도로 연결되어 있다면, 대다수의 가정은 낮은 속도로 연결되어 있고, HTTP는 낮은 속도에서도 잘 운영된다.

[0291] 한 가지 대안은 어플리케이션/게임 서버(1521 - 1525) 상의 웹브라우저를 호스트하는 것이고, RAID 배열(1511 - 1512) 상의 웹 서버의 파일들을 호스트하는 것이고(또는 잠재적으로 RAM에서 또는 웹브라우저를 호스팅하는 어플리케이션/게임 서버(1521 - 1525)에 로컬 스토리지에서, 인바운드 라우팅(1502)(또는 로컬 스토리지)을 통해서 매우 빠른 상호 연결로), HTTP를 사용하여 파일당 100ms의 레이턴시를 보이는 것보다, HTTP를 사용하여 파일당 최소한의 레이턴시가 있게 될 것이다. 그 다음에, HTTP를 통해서 웹페이지에 접근하는 그녀의 가정에 사용자를 갖는 대신에, 사용자는 클라이언트(415)를 통해 웹페이지에 접근할 수 있다. 그 다음에, 1.5Mbps 연결로(왜냐하면 이 웹사이트는 그것의 비디오를 위해 많은 대역폭을 요구하지 않기 때문에), 그 웹페이지는 라인당 1초 이하(2400)로 활성화될 수 있을 것이다. 본질적으로 어플리케이션/게임 서버(1521 - 1525)상에서 동작하는 웹브라우저가 페이지를 디스플레이하기 전에는 레이턴시가 없을 것이며, 클라이언트(415)가 웹브라우저로부터 비

디오 출력을 디스플레이하기 전에는 눈에 띄는 레이턴시가 없을 것이다. 사용자의 마우스가 웹페이지 상에 이동하고 그리고/또는 타이핑할 때, 사용자의 입력 정보는 어플리케이션/게임 서버(1521 - 1525)에서 동작하는 웹브라우저로 보내질 것이고, 웹 브라우저는 그에 따라 응답할 것이다.

[0292] 이러한 접근의 단점은 만약 압축기가 지속적으로 비디오 데이터를 전송하고 있으면, 웹페이지가 고정되어 있더라도, 대역폭이 사용된다는 것이다. 이는 웹페이지가 변할 때만 오직 데이터를 전송하도록 하고, 만약 데이터를 전송한다면 변화하는 페이지의 일부에 데이터를 전송하도록 압축기를 구성함으로써 해결될 수 있다. 반면, 계속적으로 변화하는 몇몇의 플래시 배너가 있는 웹사이트가 있는데, 그러한 웹사이트들은 화나게 하는 경향이 있고, 보통 웹페이지는 움직이는(비디오 클립) 무언가의 원인이 없다면 고정적이다. 그러한 웹사이트의 경우, 전통적인 웹서버보다 호스팅 서비스(210)를 이용하여 더 적은 데이터가 전송될 것이고 그 이유는 실제 나타난 이미지들만이 전송되기 때문이고, 씬 클라이언트가 실행할 수 있는 코드가 없고, 롤오버 이미지 같은, 결코 보이지 않는 거대한 물체도 없다.

[0293] 따라서, 레거시(legacy) 웹페이지를 호스트하기 위해 호스팅 서비스(210)을 사용하면, 웹페이지 소모 시간은 텔레비전에서 채널을 변경하는 것처럼 웹페이지를 여는 곳을 줄여줄 수 있다 : 웹페이지는 효과적으로 즉시 활성화된다.

[0294] **게임 및 어플리케이션의 가능한 디버깅(Facilitating Debugging of Games and Applications)**

[0295] 앞서 언급한 것처럼, 실시간 그래픽을 가진 비디오 게임 및 어플리케이션은 매우 복잡한 응용 프로그램(application)이고 전형적으로 그것들이 버그를 포함하여 시장으로 출시된다. 비록 소프트웨어 개발자가 버그에 대하여 사용자로부터 피드백을 받는다 할지라도, 고장 후에 머신을 되돌려주는 것을 의미하고, 그것은 게임 또는 실시간 어플리케이션이 고장 또는 부적절하게 동작되는 원인이 무엇인지 명확하게 감정하기 어렵다.

[0296] 게임 또는 어플리케이션은 호스팅 서비스(210)에서 동작할 때, 게임 또는 어플리케이션의 비디오/오디오 출력은 계속적으로 지연 버퍼(1515)에 기록된다. 더욱이, 감시 프로세스는 어플리케이션/게임 서버(1521 - 1525)가 안정적으로 동작하는지를 호스팅 서비스 컨트롤 시스템(401)으로 정기적으로 보고하도록 각각의 어플리케이션/게임 서버(1521 - 1525)를 운영한다. 만약 감시 프로세스가 보고하는 것을 실패하면, 그 다음 서버 컨트롤 시스템(401)은 어플리케이션/게임 서버(1521 - 1525)와 통신을 시도할 것이고, 만약 성공한다면 머신 상태가 이용가능한지 수집한다. 지연 버퍼(1515)에 의해서 기록된 비디오/오디오와 함께 이용가능한 어떤 정보는 소프트웨어 개발자에게 보내질 것이다.

[0297] 따라서, 게임 또는 어플리케이션 소프트웨어 개발자는 호스팅 서비스(210)로부터 고장의 통지를 얻고, 고장을 일으킨 프레임마다 기록을 얻는다. 이 정보는 버그를 밝혀내고 고치는데 있어서 굉장히 가치가 있다.

[0298] 뿐만 아니라, 어플리케이션/게임 서버(1521 - 1525)가 고장 났을 때, 서버는 가장 최근의 재시작 시점에서 재시작하게 되고, 기술적인 어려움에 대한 사과(apologiaing)의 메세지를 사용자에게 제공한다.

[0299] **리소스 공유 및 비용 절감(Resource Sharing and Cost Savings)**

[0300] 도 4a 및 4b에서 도시된 시스템은 사용자와 게임 및 어플리케이션 개발자에게 다양한 이점을 제공한다. 예컨대, 전형적으로 홈과 오피스 클라이언트 시스템(예컨대, PC 또는 게임 콘솔)은 오직 일주일에 적은 비율의 시간만큼만 사용한다. Nielsen Entertainment에 의해서 2006년 10월 5일에 배포된 "활동적인 게이머 벤치마크 연구(Active Gamer Benchmark Study)"(http://prnewswire.com/cgi-bin/stories.pl?ACCT=104&STORY=/www/story/10-05-2006/0004446115&EDATE=)에 따르면, 활동적인 게이머는 일주일에 평균 14시간을 비디오 게임 콘솔을 하는데 보내고, 핸드헬드 컴퓨터를 일주일에 약 17시간 사용하는 것으로 드러났다. 그 자료는 또한 모든 게임활동(콘솔, 핸드헬드(hand-held), PC 게임을 포함)의 경우 활동적 게이머들은 일주일에 평균 13시간을 소비하는 것을 언급했다. 콘솔 비디오 게임을 하는 시간이 높은 비율을 차지하는 것을 참작해 볼 때, 일주일은(24시간*7일) 168시간이고. 이는 활동적인 게이머의 집에서 비디오 게임 콘솔이 오직 일주일의 시간 중 17/168 = 10%만 사용되는 것이다. 또는 90%의 시간은 비디오 게임 콘솔이 쓰이지 않는 것이다. 비디오 게임 콘솔의 높은 비용을 가정하면 제조사들이 그러한 장치에 보조금을 주는 사실은 비싼 자원을 비효율적으로 사용하고 있는 것이다. 비지니스 내에 PC는 또한 일주일에 적은 시간동안만 사용되고, 특히 휴대불가능한 데스크톱 PC는 종종 Autodesk Maya와 같이 최상의 어플리케이션을 요구한다. 비록 몇몇의 비지니스가 모든 시간과 휴일에도 이루어

진다 하더라도 몇몇의 PC는(예컨대, 저녁에 일을 하기 위해 집으로 가져온 휴대용 PC)모든 시간과 휴일에도 사용되고, 대부분의 비지니스 활동은 월요일부터 금요일, 근무 시간대인 오전 9시부터 오후 5시에 집중되는 경향이 있고, 휴일과 쉬는 시간(점심시간 같은)에는 사용을 덜하고, 대부분 PC 사용이 일어나기 때문에 데스크톱 PC 활용은 이러한 작동 시간에 따르는 경향이 있다. 만약, 우리가 PC는 일주일에 5일 9시 ~ 5시까지 계속적으로 활용된다고 가정하면 PC는 일주일의 시간 중 40/168 = 24%에 해당하는 시간만큼 사용된다. 고성능 데스크톱 PC는 사무를 위한 매우 값비싼 투자이고, 이는 매우 낮은 수준의 활용을 초래한다. 데스크톱을 이용해 가르치는 학교는 일주일에 더 적은 시간동안 컴퓨터를 사용하고, 비록 가르치는 시간이 다양하다고 하더라도 대부분의 교육은 월 ~ 금요일까지 낮시간 동안 이루어진다. 따라서, 일반적으로 PC와 비디오 게임 콘솔은 일주일에 적은 시간 동안만 활용되는 것이다.

[0301] 주목할 것은, 휴일이 아닌 월 ~ 금요일의 낮시간 동안 오피스에서 또는 학교에서 많은 사람들이 컴퓨터를 활용하며, 이 사람들은 일반적으로 이 시간에는 비디오 게임을 하지 않는다. 그리고 비디오 게임을 할 때는 일반적으로 이와 다른 시간대인 저녁, 주말, 공휴일일 것이다.

[0302] 도 4a가 도시하는 호스팅 서비스의 형태를 보면, 활용 패턴이 자원을 매우 효율적으로 활용하는 결과로서, 위에 2개 단락에서 설명되었다. 특히 만약 사용자가 정교한 3D 비디오 게임과 같은 복잡한 어플리케이션의 실시간 응답을 요구한다면, 주어진 시간에 호스팅 서비스(210)에 의해 제공될 수 있는 사용자의 수는 제한이 있을 것이다. 그러나, 대부분의 시간 동안 사용되지 않는 비지니스를 위해 사용되는 PC 또는 가정에서 비디오 콘솔과는 다르게, 서버(402)는 다른 시간에 다른 사용자에 의해서 재사용될 수 있다. 예컨대, 고성능 듀얼 CPU와 듀얼 GPU와 큰 용량의 RAM을 가진 고성능 서버(402)는 평일 오전 9시에서 오후 5시 사이에 오피스와 학교에서 활용될 수 있으나, 저녁, 주말 휴일에 정교한 비디오 게임을 하는 게이머에 의해서 활용된다. 비슷하게, 저성능 어플리케이션이 셀러론 CPU, GPU가 없거나 저성능 GPU, 제한된 RAM을 가진 저성능 서버(402)에 의해 근무 시간에 활용될 수 있으며, 저성능 게임은 비근무시간 동안 저성능 서버(402)를 활용할 수 있다.

[0303] 게다가 여기서 설명된 호스팅 서비스의 배치와 함께 리소스는 수백만은 아니더라도 수천의 사용자 사이에서 효율적으로 공유된다. 일반적으로 온라인 서비스는 주어진 시간에 서비스를 이용하는 전체 사용자의 오직 적은 비율만을 가지고 있다. 만약 앞서 실은 Nielsen 비디오 게임 사용 통계치를 고려해본다면, 원인을 쉽게 알 수 있다. 만약 활동적인 게이머가 일주일에 17시간 동안 콘솔게임을 할 경우, 우리는 게임의 피크 사용 시간대가 비근무시간대인 저녁(오전 5시 ~ 12시, 7시간*5일 = 35 시간/주) 그리고 주말(오전 8시 ~ 오전 12시, 16시간*2=32 시간/주)이라고 가정하면, 게임시간인 17시간에 65(35+32)시간의 피크 타임이 있는 것이다. 시스템의 로드하는 정확한 피크 사용자는 많은 이유로 어림잡기 어렵다 : 몇몇 사용자들은 비피크 시간동안 게임을 하고 사용자들이 집단을 형성할 때가 분명 낮 시간일 것이며, 피크 시간대는 게임의 유형에 영향을 끼친다(예컨대, 어린이들은 초저녁에 게임을 하는 경향이 있다). 그러나 게이머들이 게임을 하는 평균 시간은 게이머가 게임을 하는 낮 시간보다 훨씬 적고, 주어진 시간에 호스팅 서비스(210)는 오직 적은 수의 사용자만이 사용한다. 이 분석의 목적을 위해, 우리는 피크 로드가 12.5%라고 가정할 것이다. 따라서, 컴퓨팅, 압축, 대역폭 리소스의 12.5%만이 주어진 시간에 사용되고, 하드웨어 비용의 12.5%만이 리소스의 재사용에 기인한 퍼포먼스 게임의 주어진 수준을 플레이하기 위해 주어진 사용자에게 지원하는 결과를 초래한다.

[0304] 게다가 몇몇의 게임과 어플리케이션이 다른 것들보다 좀더 많은 컴퓨팅 파워를 요구한다면, 리소스는 사용자가 하고 있는 게임, 실행하고 있는 어플리케이션에 기초하여 동적으로 배치될 것이다. 그래서 저성능 게임 또는 어플리케이션을 선택한 사용자는 저성능(덜 비싼)서버(402)로 할당할 것이며, 고성능 게임 또는 어플리케이션을 선택한 사용자는 고성능 서버(402)(더 비싼)로 할당할 것이다. 실제로, 주어진 게임 또는 어플리케이션은 게임 또는 어플리케이션의 저성능과 고성능 부분이 있고, 사용자는 게임 또는 어플리케이션의 필요에 따라 최저 가격의 서버(402)에서 운영하는 사용자를 유지하기 위해 게임 또는 어플리케이션의 부분 사이에서 하나의 서버(402)와 다른 서버(402)를 교환할 수 있다. 단일 디스크보다 훨씬 빠른 RAID 배열(405)은 저성능 서버(402)에서도 이용가능하며 이는 디스크 전송 속도를 빠르게 하는 장점을 가지고 있다. 따라서, 플레이하고 있는 게임 또는 사용하고 있는 어플리케이션 모두에 따라 서버(402)마다의 평균 비용은 고성능의 게임 또는 어플리케이션의 가장 비싼 서버(402)의 가격보다는 훨씬 낮지만, 저성능 서버(402)조차도 RAID 배열(405)로부터 디스크 성능 이점을 획득할 것이다.

[0305] 더욱이, 호스팅 서비스(210)의 서버(402)는 디스크가 없는 PC 마더보드 또는 네트워크 인터페이스외의 주변 인터페이스에 지나지 않고, 오직 SAN(403)과 빠른 네트워크 인터페이스를 갖는 단일 칩을 위해서 다운 통합될 것이다. 또한, RAID 배열(405)은 디스크보다 더 많은 사용자들이 공유하게 될 것이고 활동적인 사용자당 디스크 비용은 단일 디스크 드라이브보다 가격이 저렴할 것이다. 모든 이러한 장치는 환경적인 제어 서버 룸 환경에서

랙이 있을 것이다. 만약 서버(402)가 실패하면, 그것은 호스팅 서비스(210)에서 쉽게 수리되거나 대체될 수 있다. 대조적으로 홈이나 오피스에서 PC 또는 게임 콘솔이 견고하고 치거나 떨어뜨리는 것으로부터의 합리적인 기계적 손실을 견뎌내야 하는 독립적 장치는 하우징(덮개)이 있어야 하고, 최소한 하나의 디스크 드라이브를 가지고 있어야 하며, 불리한 환경조건을 견뎌내야 하고(예컨대 다른 장치와 함께 과열된 AV 캐비넷에 넣어져야 하는 경우) 서비스 보증이 요구되며, 포장이 되어 배송되어야 하므로 유통마진을 받는 소매업자에 의해서 판매된다. 게다가 비록 저성능 게임 또는 장치가 대부분의 시간동안 운용되더라도(또는 게임 또는 어플리케이션의 섹션) PC 또는 게임 콘솔은 장래에 어떤 시점에 사용될 컴퓨터적으로 강렬히 기대했던 게임 또는 어플리케이션의 최적의 성능을 보일 수 있도록 환경 설정이 이뤄져야 한다. 그리고, 만약 PC 또는 콘솔이 실패하면, 그것을 수리하는 것은 값비싸고 시간을 소모하는 과정이다(제조업자, 사용자 그리고 소프트웨어 개발자에게 악영향을 끼친다).

[0306] 따라서, 도 4a가 도시하는 시스템은 주어진 컴퓨터 성능 수준을 경험하기 위해 가정, 사무실 또는 학교에서 사용자가 로컬 컴퓨터의 자원과 비교할 수 있는 기회를 제공하며 도 4a가 도시하는 아키텍처를 통해 컴퓨팅 성능을 제공하는 것은 훨씬 저렴할 것이다.

[0307] **업그레이드 필요성 제거(Eliminating The Need to Upgrade)**

[0308] 더욱이, 사용자들은 더 이상 새로운 게임 또는 고성능의 새로운 어플리케이션을 다루기 위해 PC나 콘솔을 업그레이드를 해야하는 걱정을 할 필요가 없다. 서버(402)가 게임 또는 어플리케이션의 어떤 타입을 요구하는 지와 상관없이, 호스팅 서비스(210)에서 사용자는 어떤 게임이나 어플리케이션을 거의 즉시 사용할 수 있으며(즉, RAID 배열(405) 또는 서버(402)의 로컬 스토리지를 빠르게 로딩한다), 적절히 최신의 업데이트가 되고 버그를 수정한다(즉 소프트웨어 개발자는 주어진 게임 또는 어플리케이션에 맞도록 서버(402)를 이상적인 환경으로 설정하도록 선택할 수 있고, 최적의 드라이버와 함께 서버(402)의 환경설정을 하며 시간이 지나면 개발자는 즉시 호스팅 서비스(210)의 게임 또는 어플리케이션의 모든 복사에 대한 업데이트를 제공하고 버그를 고친다). 실제로, 사용자가 호스팅 서비스(210)를 사용하기 시작한 후에 사용자는 게임과 어플리케이션에 더 좋은 경험을 제공하고 시작했음을 알게되고(예컨대, 업데이트나 버그 수정을 통해) 그것은 아마도 사용자가 1년 전에는 존재하지도 않았던 컴퓨팅 기술(예컨대, 높은 성능 GPU)을 활용하는 서비스(210)에서 이용할 수 있도록 만들어진 새로운 게임이나 어플리케이션을 사용자가 1년 후 발견하는 경우가 될 것이다. 따라서 사용자가 1년 후에 게임을 하거나 어플리케이션을 작동하기 전에 그 기술을 구입하는 것은 불가능한 것이다. 게임을 하고 기기를 운용하는 컴퓨팅 자원은 사용자에게 보이지 않기 때문에(즉, 사용자의 지각으로, 사용자는 텔레비전 채널을 바꾸는 것만큼이나 사용자는 거의 즉시 작동하는 게임이나 어플리케이션을 간단히 선택한다) 사용자의 하드웨어는 업그레이드를 인식하지 못한 상태에서 "업그레이드"가 될 것이다.

[0309] **백업의 필요성 제거(Eliminating the Need for Backups)**

[0310] 비지니스, 학교 그리고 가정에서 사용자가 겪는 주요한 다른 문제는 백업이다. 로컬 PC 또는 비디오 게임에 저장된 정보(예컨대, 콘솔의 경우 사용자의 게임 성취도 및 순위)는 디스크가 에러가 나거나 의도치 않는 삭제로 잃을 수 있다. 많은 어플리케이션에 사용할 수 있는 PC를 위한 매뉴얼이나 자동 백업이 제공되며 게임 콘솔은 백업을 위한 온라인 서버로 업로드를 할 수 있으나 로컬 백업은 전형적으로 안전하고 조직화된 어딘가에 저장되는 다른 로컬 디스크에 복사되는 것이며(또는 데이터가 없어지지 않는 다른 저장 장치), 온라인 서비스를 통한 백업은 전형적으로 저비용의 인터넷 연결을 통한 늦은 업스트림 속도 때문에 종종 제한을 받는다. 도 4a의 호스팅 서비스(210)과 함께 RAID 배열(405)에 저장된 데이터는 잘 알려진 RAID 환경설정 선행기술을 이용하여 환경설정을 할 수 있는데 이는 서버 센터에 실패한 경우 어떤 데이터도 잃지 않을 것이며, 서버 센터의 기술자는 실패한 디스크에 대해 공지를 하게 될 것이고 업데이트를 자동적으로 실행한 디스크로 대체될 것이며 RAID 배열은 한번 더 실패를 견디게 될 것이다. 게다가, 디스크 드라이버의 모든 것이 가까이 있고 SAN(403)을 통해 그것들 사이에 빠른 로컬 네트워크와 함께 있으며 서버 센터 또는 떨어진 곳에 다시 위치되어 저장될 수 있는 2차적인 저장장소에 규칙적 기초를 바탕으로 백업된 디스크 시스템의 모든 것을 서버센터에 배치하는 것은 어렵지 않다. 호스팅 서비스(210)를 보는 사용자의 관점에서 데이터는 항상 간단하게 보호되고 결코 백업에 대해 생각할 필요가 없게 된다.

[0311] 데모에 접근(Access to Demos)

[0312] 사용자들은 종종 게임이나 기기를 구입하기 전에 시도해보길 원한다. 앞서 설명한 것처럼 게임과 어플리케이션을 시범해볼 수 있는 선행 기술이 있으나(데모(demo)라는 언어의 뜻은 데모(명사는 아님)라고 불리기도 하는 데몬스트레이션(demonstration) 버전을 시도해보는 것이다) 데모 게임과 어플리케이션 그러나 그것들 각각은 제한 및/또는 불편을 겪는다. 호스팅 서비스(210)를 이용할 때 사용자들은 데모를 시도하는 것이 쉽고 편리하다. 실제로, 모든 사용자가 사용자 인터페이스(아래 설명하는 것처럼)를 통해 데모를 선택하고 시도한다. 데모는 적절한 서버(402)에서 거의 즉시 로드될 것이고, 다른 어느 게임이나 어플리케이션과 같이 운용될 것이다. 데모가 고성능 서버(402)를 요구한다든지 저성능 서버(402)를 요구한다든지 사용자의 관점에서 홈 또는 오피스 클라이언트(415)를 사용하는 것과는 상관없이 데모는 작동하게 될 것이다. 게임이나 어플리케이션 데모 소프트웨어 제작사는 어떤 데모가 얼마나 오랫동안 사용이 허용되는지 정확하게 제어할 수 있을 것이며 물론 데모는 게임이나 어플리케이션의 전체 버전의 접근성을 확보할 수 있는 기회를 사용자에게 제공하는 사용자 인터페이스 요소를 포함하고 있어야 한다.

[0313] 데모가 원가 이하 또는 무료로 제공되기 쉽기 때문에 몇몇 사용자들은 데모를 반복적으로 사용할 것이다(특히 반복적으로 플레이하며 즐기는 게임 데모). 호스팅 서비스(210)는 주어진 사용자가 데모를 사용하는 것을 제한하는 다양한 기술을 적용할 수 있다. 가장 간단한 접근은 각 사용자에게 사용자 아이디를 설정하고 주어진 사용자 아이디가 데모를 실행하도록 허용된 횟수를 제한하는 것이다.

[0314] 그러나 사용자는 다수의 사용자 아이디를 설정할 것이고 그것이 무료라면 더욱 그렇다. 이러한 문제를 다루는 한가지 기술은 주어진 클라이언트(415)가 데모를 실행하는 데 허용된 횟수를 제한하는 것이다. 만약 클라이언트가 독립형 장치라면 그 장치는 시리얼 넘버를 가지고 있을 것이며 호스팅 서비스(210)는 데모가 그와 같은 시리얼 넘버의 클라이언트에 의해서 다뤄질 수 있도록 횟수를 제한할 수 있다. 만약 클라이언트(415)가 PC 또는 다른 장치의 소프트웨어로서 운용된다면 시리얼 넘버는 호스팅 서비스(210)에 의해서 할당될 수 있고, PC에 저장되어 데모 사용을 제한하는데 이용될 것이다, 그러나 사용자가 PC를 다시 프로그램화했다고 가정한다면 시리얼 넘버는 지워지거나 변경될 수 있으며, 다른 대책은 호스팅 서비스(210)가 PC 네트워크 확장카드, MAC 어드레스(및/또는 하드 드라이브 시리얼 넘버 등과 같은 다른 기계적 특별한 식별자)를 기록하고 데모 사용을 제한하는 것이다. 그러나, 네트워크 확장카드의 MAC 주소가 변경된다고 가정한다면 이는 실패할 염려가 없는 방법이 아니다. 데모의 사용횟수를 제한하는 다른 접근은 주어진 IP 주소로 실행될 수 있게 하는 것이다. 비록 IP 주소가 케이블 모뎀과 DSL 제공자에 의해 주기적으로 재할당되더라도 이는 실제로는 매우 자주 일어나는 것은 아니며 주택지의 DSL 또는 케이블 모뎀 접근을 위한 지역 내의 IP가 결정이 되면 데모 사용의 적은 숫자는 전형적으로 주어진 가정에 설정될 수 있다. 또한 같은 IP 주소를 공유하는 NAT 라우터 하에 가정에는 다수 기계 장치가 있을 수 있지만 전형적인 주거환경에서는 그러한 장치는 제한된 수만큼 있을 것이다. 만약 IP 주소가 한 기업에 있다면 데모의 대다수가 한 기업을 위해 설정될 수 있다. 그러나 마침내 이전에 언급한 모든 접근법의 결합이야말로 PC의 데모 수를 제한하는 최선의 방법이다. 비록 숙달된 사용자가 데모 사용을 반복적으로 하는 것을 제한할 수 있는 확고하고 기술적이며 실패 없는 방법이 없을지라도 많은 장애물을 생성하여 대다수의 PC 사용자가 데모 시스템을 남용하는 것은 노력을 하지 않도록 충분한 제지책을 창조할 수 있으며, 사용자들은 새로운 게임과 어플리케이션을 시도하려는 경향이 있었던 것과 같이 오히려 데모를 사용한다.

[0315] 학교, 비지니스 그리고 다른 단체에서의 이점(Benefits to Schools, Businesses and Other Institutions)

[0316] 도 4a가 도시되는 시스템을 활용하는 비지니스, 학교 그리고 기타 단체에서 중요한 이점이 축적된다. 오피스와 학교는 PC를 설치, 유지 그리고 업그레이드하는데, 특히 Maya와 같이 고성능 어플리케이션을 운영하는 PC에 관한 한 상당한 비용이 든다. 앞에서 언급한 바와 같이 PC는 일반적으로 일주일에 일부 시간만이 활용되는 것이 일반적이고 주어진 수준의 성능에 맞는 PC의 가격은 서버 센터 환경에서 보다 오피스 또는 학교 환경에서 훨씬 높다.

[0317] 큰 사업체나 학교의 경우(예컨대, 큰 대학교들), 서버 센터를 구축하고 LAN 단계 연결을 통해 원거리 접근하여 컴퓨터를 유지하는 독립체들과 같은 IT 부서에서 실용적일 수 있다. 오피스 사이에 LAN이나 개인 높은 대역폭 연결 원거리 컴퓨터 접근을 위한 다수의 해결책이 존재한다. 예컨대, 마이크로소프트 윈도우 터미널 서버 또는 RealVNC사의 VNC와 같은 가상의 네트워크 컴퓨팅 기기 또는 Sun Microsystems의 씬 클라이언트를 통해서, 사용자들은 그래픽 응답시간이나 사용자 경험의 일정 범위의 품질로 PC 또는 서버와 원거리 접근을 확보할 수 있게 된다. 게다가 그러한 자체-관리 서버센터는 전형적으로 하나의 사업체 또는 학교 등 그러한 곳에서 전용되며 전

혀 다른 어플리케이션(예컨대 엔터테인먼트와 사무기기)은 일주일에 다른 시간대에 같은 컴퓨터 자원을 활용하게 될 때 가능한 중복 사용을 활용할 수 없다. 따라서, 많은 사업체와 학교는 스스로 각각의 사용자에게 LAN-속도 네트워크 연결을 가지는 서버센터를 구축할 만한 규모, 자원 또는 기술이 부족하다. 사실, 학교와 사업체의 많은 비율이 가정에서와 같은 인터넷 연결(예컨대, DSL, 케이블 모뎀)을 가지고 있다.

[0318] 그러나 이러한 조직은 정기적 근거 또는 주기적 근거로 여전히 매우 높은 고성능 컴퓨팅에 대한 필요성이 있을 수 있다. 예컨대, 소규모 건축 회사는 상대적으로 아주 소수의 건축가를 보유할 것이고, 디자인 작업을 할 때 대체로 현대 컴퓨팅 요구를 갖지만, 주기적으로 매우 높은-성능의 3D 컴퓨팅을 요구할 것이다(예컨대, 고객을 위해 새로운 건축 디자인의 3차원 플라이-쓰루를 창조하는 때). 이 시스템은 도 4a와 같이 그러한 조직을 위해 매우 적합하다. 그 조직은 가정에 제공되는 네트워크 연결(예컨대, DSL, 케이블 모뎀)과 동일한 종류 이상의 어떤 필요도 있지않고, 전형적으로 매우 저렴하다. 그들은 클라이언트(415)처럼 저렴한 PC를 활용할 수 있거나 PC를 한꺼번에 제공하고, 단지 제어 신호 로직(413) 및 낮은-레이턴시 비디오 신장(412)을 실행하는 저렴한 전용 장치를 활용할 수 있다. 이러한 기능은 PC 내에 민감한 구성 요소 손상이나 PC 도난에 문제가 있는 학교를 위해 매력적이다.

[0319] 이러한 배열은 그러한 조직을 위한 많은 문제를 해결한다(그리고 이러한 많은 장점은 또한 일반 목적 컴퓨팅을 하는 가정 사용자에 의해서 공유될 수 있다). 운영 비용(이는 궁극적으로 실행 가능한 사업을 하기 위해서 사용자에게 몇 가지 형태로 통과해야한다)은 매우 낮을 수 있는데, 이는 (a) 컴퓨팅 리소스가 그 밖의 주중에 다른 피크 사용 시간을 갖는 어플리케이션과 공유된다, (b) 조직은 필요할 때만 높은 성능 컴퓨팅 리소스에 접근을 얻을 수 있다(그리고 비용을 초래하다), (c) 조직은 높은 성능 컴퓨팅 리소스를 유지하기 위해 백업을 위한 리소스를 제공하지 않아도 되기 때문이다.

[0320] **불법 복제의 제거(Elimination of Piracy)**

[0321] 게다가, 게임, 어플리케이션, 인터랙티브 영화 등은, 오늘날처럼 더 이상 불법 복제할 수 없게 된다. 왜냐하면 게임이 서비스 센터에서 실행되고 사용자들은 근본적인 프로그램 코드에 접근하는 것이 제공되지 않고, 거기에는 아무런 불법 복제도 있지 않다. 사용자가 소스 코드를 복제했을지라도, 사용자가 표준 게임 콘솔 또는 홈 컴퓨터에서 그 코드를 실행할 수 없을 것이다. 이는 표준 비디오 게이밍이 이용될 수 없는 중국과 같은 세상의 장소에서 시장을 연다. 사용하는 게임의 재판매도 가능하지 않다.

[0322] 게임 개발자를 위해, 오늘날의 경우처럼 거기에는 적은 시장 불연속성이 있다. 호스팅 서비스(210)는 완전히 새로운 세대의 기술은 사용자와 개발자가 업그레이드하고 게임 개발자는 하드웨어 플랫폼을 적시에 납품하는데 의존하게 되는 현재 상황과는 대조적으로 점차로 게이밍 요구의 변화에 따라 시간에 걸쳐 업데이트될 수 있을 것이다.

[0323] **스트리밍 인터랙티브 비디오(Streaming Interactive Video)**

[0324] 상기 설명은 새로운 근본적인 개념의 일반적인 인터넷 기반, 낮은-레이턴시 스트리밍 인터랙티브 비디오(이는 함축적으로 여기서 사용되는 것처럼, 비디오와 함께 오디오를 포함한다)에 의해서 할 수 있는 넓은 범위의 어플리케이션을 제공한다. 인터넷을 통해 스트리밍 비디오를 제공하는 종래 기술 시스템은 오직 높은 레이턴시 상호작용을 갖으면서 실행될 수 있는 어플리케이션으로 활성화될 수 있다. 예컨대, 선형 비디오를 위한 기본적인 재생 컨트롤(예컨대, 정지, 되감기, 빨리 감기)은 적당히 높은 레이턴시로 동작하고, 그것은 선형 비디오 공급 중에서 선택하는 것이 가능하다. 그리고, 앞서 명시된 바와 같이, 일부 비디오 게임의 특성은 그들이 높은 레이턴시로 플레이될 수 있도록 허락한다. 하지만 스트리밍 비디오를 위한 종래 기술 접근의 높은 레이턴시(또는 낮은 압축 비율)는 심각하게 스트리밍 비디오의 잠재적인 어플리케이션을 제한하거나 전문 네트워크 환경을 위해 그들의 배포(deployment)를 좁게하고, 그러한 환경에서조차, 종래 기술은 실질적으로 네트워크에 부담을 유도한다. 여기서 설명된 기술은 인터넷을 통한, 특히 소비자 등급 인터넷 연결을 통해 가능할 수 있는, 낮은-레이턴시 스트리밍 인터랙티브 비디오를 갖을 수 있는 어플리케이션의 넓은 범위를 위한 문을 연다.

[0325] 사실, 강력한 서버들 사이에 매우 빠른 네트워킹 및 임의의 양의 빠른 스토리지, 임의의 양의 컴퓨팅 파워를 갖는 강화된 사용자 경험을 제공하기에 충분하면서 도 4c의 클라이언트(465)만큼 작은 클라이언트 장치는, 새로운 기준의 컴퓨팅을 가능하게 할 수 있다. 더욱이, 대역폭 요구 사항이 시스템 성장의 컴퓨팅 파워만큼 성장하지 않았기 때문에(예컨대, 대역폭 요구는 디스플레이 해상도, 품질 및 프레임 속도에만 얽매여 있기 때문에), 일단

광대역 인터넷 연결성은 유비쿼터스(예컨대, 널리보급된 낮은-레이턴시 무선 커버리지(coverage)를 통해서), 신뢰할 수 있고, 모든 사용자의 디스플레이 장치(422)의 필요와 만날 수 있는 충분히 높은 대역폭이며, 씩 클라이언트(thick client)(윈도우, 리눅스, OSK 등이 구동되는 PC 또는 모바일 폰와 같은)인지 또는 씬 클라이언트(thin client)(Adobe Flash 또는 Java와 같은)인지에 관한 의문은 전형적인 소비자 및 비지니스 어플리케이션을 위해 필요하다.

[0326] 스트리밍 인터랙티브 비디오의 출현은 컴퓨팅 아키텍처의 구조에 대한 가정을 다시 생각하도록 한다. 이러한 예는 도 15에 도시된 호스팅 서비스(210) 서버 센터 실시예이다. 지연 버퍼 및/또는 그룹 비디오(1550)를 위한 비디오 경로는 어플리케이션/게임 서버(1521 - 1525)의 멀티캐스트된 스트리밍 인터랙티브 비디오가 경로(1552)를 경유하여 실시간 또는 경로(1551)를 경유하는 선택가능 지연 후에 어플리케이션/게임 서버(1521 - 1525)로 되돌아오는 피드백 루프이다. 이는 종래 기술 서버 또는 로컬 컴퓨팅 아키텍처를 통해 불가능하거나 실행할 수 없는 넓은 범위의 실질적인 어플리케이션이 가능할 수 있다(예컨대, 도 16,17 및 20에 도시된 것과 같은). 하지만, 보다 일반적인 아키텍처 특징으로서, 피드백 루프(1550)가 제공하는 것은 어플리케이션이 요구하는 것처럼 비디오가 불명확하게 루프백될 수 있기 때문에, 스트리밍 인터랙티브 비디오 수준의 순환이다. 이는 넓은 범위 어플리케이션의 가능성은 전에는 결코 이용가능하지 않도록 할 수 있다.

[0327] 또 다른 주요 아키텍처 기능은 비디오 스트림은 단방향 UDP 스트림이라는 것이다. 이는 효과적으로 임의의 정도의 멀티캐스팅의 스트리밍 인터랙티브 비디오를 가능하게 할 수 있다(반대로, 양방향 스트림, 즉, TCP/IP 스트림과 같은 양방향 스트림은, 다수의 사용자가 증가됨으로서 뒷면-및-앞에서 네트워크에 점점 더 많은 트래픽 로그잼(logjam)을 만들 수 있게 될 것이다). 멀티캐스팅은 서버 센터내에 중요한 성능이다 왜냐하면 그것은 일대다(one-to-many) 또는 다대다(many-to-many)로 통신 하기 위해서 인터넷 사용자(그리고 사실 그 세계의 인구)의 증가하는 필요에 응답하는 시스템을 허용하도록 하기 때문이다. 다시 말해서, 도 16에서처럼, 여기서 논의하는 예로 스트리밍 인터랙티브 비디오 재귀 및 멀티캐스팅은 큰 빙산의 일각이다.

[0328] **비-트랜싯 피어링(Non-Transit Peering)**

[0329] 일 실시예에서, 호스팅 서비스(210)는 사용자에게 인터넷 서비스를 제공하는 하나 또는 그 이상의 인터넷 서비스 공급자(ISP)에 하나 또는 그 이상의 피어링 연결을 가지며, 이러한 방식으로 호스팅 서비스(210)는 ISP 네트워크 내에 유지되는 비-트랜싯 라우트를 통해 사용자와 통신할 수 있다. 예컨대, 만약 호스팅 서비스(210) 및 WAN 인터페이스(441)가 컴캐스트 케이블 커뮤니케이션즈 인코포레이티드(Comcast Cable Communications, Inc.'s)의 네트워크에 직접 연결되고, 사용자 구내(211)에 컴캐스트 케이블 모뎀에 의한 광대역 서비스가 제공되면, 호스팅 서비스(210)와 클라이언트(415)간 라우트는 컴캐스트 네트워크 내에 완전히 구축될 것이다. 이러한 잠재적인 장점은 낮은 통신 비용(2개 또는 그 이상의 ISP 네트워크간 IP 트랜싯 비용을 피할 수 있으므로), 잠재적으로 보다 높은 신뢰성 있는 연결(혼잡하거나 ISP 네트워크간 다른 트랜싯 방해가 있는 경우), 및 낮은 레이턴시(혼잡하거나, 불충분한 라우트 또는 ISP 네트워크간 다른 지연이 있는 경우)를 포함한다.

[0330] 본 실시예에서, 클라이언트(415)가 세션의 시작시에 호스팅 서비스(210)를 최초 접촉할 때, 호스팅 서비스(210)는 사용자 구내(211)의 IP 주소를 수신한다. 다음에, 또 다른 ISP를 통한 IP 트랜싯 없이 사용자 구내(211)로 라우트할 수 있는 호스팅 서비스(210)에 연결된 특정 ISP로 그 IP 주소가 할당되었는지를 보기 위해, 예컨대 ARIN(American Registry for Internet Numbers)으로부터 이용가능한 IP 주소 테이블을 이용한다. 예컨대, IP 주소가 76.21.0.0과 76.21.127.255 사이이면, 그 IP 주소는 컴캐스트 케이블 커뮤니케이션즈 인코포레이티드로 할당된다. 이러한 예에서, 호스팅 서비스(210)가 컴캐스트, AT&T 및 Cox ISPs에 연결을 유지하면, 특정 사용자에게 적절한 라우트를 제공하기에 가장 적합한 ISP로서 컴캐스트를 선택한다.

[0331] **피드백을 이용한 비디오 압축**

[0332] 일 실시예에서, 피드백은 성공적인(또는 비성공적인) 타일 및/또는 프레임 전송을 표시하기 위해 클라이언트 장치에서 호스팅 서비스로 제공된다. 이후, 클라이언트로부터 제공된 피드백 정보는 호스팅 서비스에서 비디오 압축 동작을 조절하기 위해 이용된다.

[0333] 예컨대, 도 25a~b는 피드백 채널(2501)이 클라이언트 장치(205A)와 호스팅 서비스(210) 사이에 구축된 본 발명의 일 실시예를 나타낸다. 피드백 채널(2501)은 패킷화된 성공적으로 수신된 타일/프레임의 확인 응답 및/또

는 비성공적으로 수신된 타일/프레임의 표시를 송신하기 위해 클라이언트 장치(205)에 의해 이용된다.

[0334] 일 실시예에서, 각각의 타일/프레임을 성공적으로 수신한 후, 클라이언트는 확인 응답 메시지를 호스팅 서비스(210)로 전송한다. 본 실시예에서, 호스팅 서비스(210)는 만약 특정 시간 주기 후 확인 응답을 수신하지 못하거나 클라이언트 장치(205)가 송신된 것과 다른 다음의 타일/프레임을 수신했다는 확인 응답을 수신하면 패킷 손실을 검출한다. 선택적으로 또는 추가로, 클라이언트 장치(205)는 그 패킷 손실을 검출하고 그러한 패킷 손실에 영향을 받은 타일/프레임의 표시와 함께 호스팅 서비스(210)로 그 패킷 손실의 표시를 전송한다. 본 예에서, 성공적으로 전송된 타일/프레임의 연속적인 확인 응답은 필요치 않다.

[0335] 패킷 손실이 검출되었는지에 상관없이, 도 25a~b에 나타낸 실시예에서, 이미지(도 24a에는 도시하지 않음)를 위한 최초의 I-타일 세트를 생성한 후, 패킷 손실이 검출될 때까지 인코더는 거의 P-타일만을 생성한다. 도 25a에서, 2510과 같은 각각의 프레임은 4개의 수직 타일로 도시되어 있다. 그러한 프레임은 2×2, 2×4, 4×4 등과 같이 각기 다른 구성으로 타일되고, 또 그러한 프레임은 각각의 타일들이 없이 완전체(즉, 1개의 큰 타일로)로 인코드될 수 있다. 상기한 프레임 타일링 구성의 예들은 본 발명의 이러한 실시예를 나타낼 목적으로 제공된다. 본 발명의 기초가 되는 원리는 소정의 특정 프레임 타일링 구성으로 한정하지 않는다.

[0336] 단디 P-타일만을 전송하는 것은 상술한 모든 이유에 대한 채널의 대역폭 필요조건을 감소시킨다(즉, P-타일이 보통 I-타일보다 작은). 패킷 손실이 피드백 채널(2501)을 통해 검출되면, 도 25b에 나타낸 바와 같이, 클라이언트 장치(205) 상의 디코더(2502)의 상태를 재초기화하기 위해, 인코더(2500)에 의해 새로운 I-타일이 생성된다. 나타낸 바와 같이, 일 실시예에서, I-타일은 각각의 개별 인코드된 프레임에 의해 소비된 대역폭을 한정하기 위해 다수의 인코드된 프레임에 걸쳐 확산된다. 예컨대, 도 25에서, 각각의 프레임은 2개의 타일을 포함하며, 단일의 I-타일은 4개의 성공적인 인코드된 프레임 내의 각기 다른 위치로 전송된다.

[0337] 인코더(2500)는 본 실시예와 관련하여 기술한 기술들을 여기에 기술한 다른 인코딩 기술들과 결합한다. 예컨대, 검출된 패킷 손실에 따라 I-타일을 생성하는 것 외에, 상기 인코더(2500)는 I-타일이 이미지 시퀀스를 적절하게 랜더하는데 효과적인 다른 상황으로 I-타일을 생성한다(갑작스런 장면 전환에 따른 것과 같은).

[0338] 도 26a는 클라이언트 장치(205)와 호스팅 서비스(210)간 피드백 채널(2601)에 따른 본 발명의 다른 실시예를 나타낸다. 검출된 패킷 손실에 따른 새로운 I-타일/프레임의 생성이 아니라, 본 실시예의 인코더(2600)는 P-타일/프레임의 의존성을 조절한다. 최초의 내용과 같이, 본 예를 기술하는 구체적인 세부사항은 본 발명의 기초가 되는 원리들을 추구할 필요가 없다는 것을 염두해 두자. 예컨대, 본 예가 P-타일/프레임을 이용하여 기술되었지만, 본 발명의 기초가 되는 원리들이 소정의 특정 인코딩 포맷으로 한정되지 않는다.

[0339] 도 26a에서, 인코더(2600)는 다수의 압축되지 않은(비압축) 타일/프레임(2605)을 다수의 P-타일/프레임(2606)으로 인코드하여, 그 P-타일/프레임을 통신 채널(예컨대, 인터넷)을 통해 클라이언트 장치(205)로 전송한다. 클라이언트 장치(205) 상의 디코더(2602)는 다수의 압축복원된(신장된) 타일/프레임(2607)을 생성하기 위해 P-타일/프레임(2606)을 디코드한다. 인코더(2600)의 페이스트 상태(들)(2611)는 호스팅 서비스(210) 상의 메모리 장치(2610) 내에 저장되고, 디코더(2602)의 페이스트 상태(들)(2621)은 클라이언트 장치(205) 상의 메모리 장치(2620) 내에 저장된다. 디코더의 그러한 "상태"는 MPEG-2 및 MPEG-4와 같은 비디오 코딩 시스템의 기술과 관련되어 잘 알려져 있다. 일 실시예에서, 메모리들 내에 저장된 그러한 페이스트 "상태"는 이전 P-타일/프레임으로부터의 결합된 데이터를 포함한다. 그러한 메모리들(2611, 2621)은 도 26a에 나타낸 바와 같이 인코더(2600) 및 디코더(2602)로부터 분리되는 것이 아니라 각각 인코더(2600) 및 디코더(2602) 내에 통합될 것이다. 더욱이, RAM(random access memory)을 포함한 다양한 타입의 메모리가 예로서 한정하지 않고 사용될 수 있다.

[0340] 일 실시예에서, 패킷 손실이 일어나지 않으면, 인코더(2600)는 이전 P-타일/프레임에 따라 각각의 P-타일/프레임을 인코드한다. 따라서, 도 26a에 사용된 기호로 나타낸 바와 같이, P-타일/프레임(4)은 P-타일/프레임(3; 기호 4_3을 이용하여 나타낸)을 따르고, P-타일/프레임(5)은 P-타일/프레임(4; 기호 5_4를 이용하여 나타낸)을 따르며, P-타일/프레임(6)은 P-타일/프레임(5; 기호 6_5를 이용하여 나타낸)을 따른다. 이러한 예에서, P-타일/프레임(4_3)은 인코더(2600)와 디코더(2602)간 전송 동안 손실된다. 그러한 손실은 상술한 것들을 포함하지만 한정하지 않는 다양한 방식으로 인코더(2600)로 전달될 것이다. 예컨대, 디코더(2606)가 성공적으로 타일/프레임을 수신 및/또는 디코드할 때마다 이러한 정보는 디코더(2602)에서 인코더(2600)로 전달될 것이다. 만약 인코더(2600)가 시간 주기 후 특정 타일/프레임이 수신 및/또는 디코드되었다는 표시를 수신하지 못하면, 인코더(2600)는 타일/프레임이 성공적으로 수신되지 못한 것으로 추정한다. 선택적으로, 또는 추가로, 디코더(2602)는 특정 타일/프레임이 성공적으로 수신되지 못할 경우 인코더(2600)에 통지할 것이다.

[0341] 일 실시예에서, 손실 타일/프레임이 어떻게 검출되었는지에 상관없이 인코더(2600)는 디코더(2602)에 의해 성공적으로 수신된 것으로 알려진 마지막 타일/프레임을 이용하여 다음 타일/프레임을 인코드한다. 도 26a에 나타낸 예에서, 타일/프레임(5, 6)은 타일/프레임(4)의 손실로 인해 디코더(2602)에 의해 적절하게 디코드될 수 없기 때문에 "성공적으로 수신된" 것으로 고려하지 않는다(즉, 타일/프레임(5)의 디코딩은 타일/프레임(4)에 의존하고 타일/프레임(6)의 디코딩은 타일/프레임(5)에 의존한다). 따라서, 도 26a에 나타낸 예에서, 인코더(2600)는 디코더(2602)가 적절하게 디코드할 수 없는 타일/프레임(6)이 아니라 타일/프레임(3; 마지막 성공적으로 수신된 타일/프레임)에 따라 타일/프레임(7)을 인코드한다. 비록 도 26a에는 도시하지 않았지만, 추가의 패킷 손실이 검출되지 않는다는 것을 가정하여, 이후 계속해서 타일/프레임(8)이 타일/프레임(7)에 따라 인코드되고, 타일/프레임(9)이 타일/프레임(8)에 따라 인코드될 것이다.

[0342] 상술한 바와 같이, 인코더(2600) 및 디코더(2602) 모두는 각각의 메모리(2610, 2620) 내에 페이스트 인코더 및 디코더 상태(2611, 2621)를 유지한다. 따라서, 타일/프레임(7)을 인코딩할 때, 인코더(2600)는 메모리(2610)로부터 타일/프레임(3)과 관련된 이전 인코더 상태를 회복한다. 유사하게, 디코더(2602)와 관련된 메모리(2620)는 적어도 마지막 알려진 양호한 디코더 상태(예에서 P-타일/프레임(3)과 관련된 상태)를 저장한다. 결과적으로, 디코더(2602)는 타일/프레임(7)이 디코드될 수 있도록 타일/프레임(3)과 관련된 페이스트 상태 정보를 회복한다.

[0343] 상술한 기술들의 결과로서, I-타일/프레임이 항상 필요치 않기 때문에 비교적 작은 대역폭을 이용하여 실시간, 낮은 레이턴시, 인터렉티브 비디오가 인코드될 수 있다(스트림의 시작시에 디코더 및 인코더를 초기화하는 것을 제외하고). 더욱이, 디코더에 의해 생성된 비디오 이미지가 손실 타일/프레임(4) 및 타일/프레임(5, 6)으로부터 야기되는 원하지 않는 왜곡을 일시적으로 포함할 수 있지만, 이러한 왜곡은 아주 짧은 기간 동안 보일 것이다. 더욱이, 만약 타일이 사용되면(전체 비디오 프레임이 아니라), 그러한 왜곡은 회복된 비디오 이미지의 특정 영역으로 한정될 것이다.

[0344] 본 발명의 일 실시예에 따른 방법은 도 26b에 기술된다. 2650에서, 타일/프레임이 앞서 생성된 타일/프레임에 따라 생성된다. 2651에서, 손실 타일/프레임이 검출된다. 일 실시예에서, 그러한 손실 타일/프레임은 상술한 바와 같이 인코더에서 디코더로 전달된 정보에 따라 검출된다. 2652에서, 디코더에서 성공적으로 수신 및/또는 디코드된 알려진 타일/프레임에 따라 다음 타일/프레임이 생성된다. 일 실시예에서, 인코더는 메모리로부터 성공적으로 수신 및/또는 디코드된 타일/프레임과 관련된 그러한 상태를 로딩함으로써 다음 타일/프레임을 생성한다. 유사하게, 그러한 디코더가 새로운 타일/프레임을 수신하면, 메모리로부터 성공적으로 수신 및/또는 디코드된 타일/프레임과 관련된 그러한 상태를 로딩함으로써 그 타일/프레임을 디코드한다.

[0345] 일 실시예에서, 다음 타일/프레임은 디코더에서 성공적으로 수신 및/또는 디코드된 마지막 타일/프레임에 기초하여 생성된다. 다른 실시예에서, 생성된 그러한 다음 타일/프레임은 I 타일/프레임이다. 또 다른 실시예에서, I 프레임으로서 또는 앞서 성공적으로 수신된 타일/프레임에 기초하여 다음 타일/프레임을 생성할지의 선택은 많은 타일/프레임이 채널의 레이턴시 및/또는 손실이 얼마나 있는지에 기초한다. 비교적 작은 수(예컨대, 1 또는 2)의 타일/프레임이 손실되고 왕복 레이턴시(round-trip latency)가 비교적 낮은(예컨대 1 또는 2 프레임 타임) 상황에서, 마지막 성공적으로 수신된 타일/프레임과 새롭게 생성된 것간의 차이가 비교적 작기 때문에 P 타일/프레임을 생성하기에 가장 적합할 것이다. 만약 몇개의 타일/프레임이 손실되고 왕복 레이턴시가 높으면, 마지막 성공적으로 수신된 타일/프레임과 새롭게 생성된 것간의 차이가 많기 때문에 I 타일/프레임을 생성하기에 가장 적합할 것이다. 일 실시예에서, 타일/프레임 손실 임계치 및/또는 레이턴시 임계치가 I 타일/프레임 또는 P 타일/프레임을 전송할지를 결정하기 위해 설정된다. 손실 타일/프레임의 수가 타일/프레임 손실 임계치 이하 및/또는 왕복 레이턴시가 레이턴시 임계치 이하이면, 새로운 I 타일/프레임이 생성되거나, 아니면 새로운 P 타일/프레임이 생성된다.

[0346] 일 실시예에서, 인코더는 항상 마지막 성공적으로 수신된 타일/프레임에 따라 P 타일/프레임을 생성하려 하며, 만약 인코딩 프로세스에서 P 타일/프레임이 I 타일/프레임보다 작을 것이라는 것을 인코더가 결정하면(예컨대, 만약 1/8번째 압축된 타일/프레임을 갖고 그 압축된 크기가 앞서 압축된 평균 I 타일/프레임의 1/8번째의 크기보다 크다면), 인코더는 P 타일/프레임을 압축하는 것을 포기하고 대신 I 타일/프레임을 압축할 것이다.

[0347] 만약 손실 패킷이 가끔 발생하면, 드롭된 타일/프레임을 리포트하기 위해 피드백을 이용하는 상술한 시스템은 통상 인코더(2600)가 짧은 시간에 타일/프레임을 압축한다고 가정하면 손실 패킷에 의해 손상된 타일/프레임이 클라이언트 장치(205)와 호스팅 서비스(210)간 1회 왕복 시간으로 개략적으로 교체되기 때문에 사용자에 이르기까지 비디오 스트림에 있어 매우 적은 손상을 야기한다. 그리고, 압축된 새로운 타일/프레임이 압축되지 않은

비디오 스트림의 마지막 프레임에 기초하기 때문에, 비디오 스트림은 압축되지 않은 비디오 스트림 이후로 떨어지지 않는다. 그러나, 만약 새로운 타일/프레임을 포함하는 패킷 또한 손실되면, 이는 적어도 2회 왕복의 지연을 야기하고 또 다른 새로운 타일/프레임을 요청 및 송신을 야기하며, 많은 실제 상황에서는 비디오 스트림에 대한 현저한 손상을 야기할 것이다. 그 결과, 드롭된 타일/프레임 이후 송신된 새롭게 인코드된 타일/프레임이 호스팅 서비스(210)에서 클라이언트 장치(205)로 성공적으로 송신되는 것이 매우 중요하다.

[0348] 일 실시예에서, 앞서 기술되고 도 11a, 11b, 11c 및 11d에 도시된 것과 같은 순방향 에러 보정(FEC) 코딩 기술은 새롭게 인코드된 타일/프레임의 손실 가능성을 감소시키기 위해 사용된다. FEC 코딩이 타일/프레임을 전송할 때 이미 사용되었다면, 좀더 강한 FEC 코드가 새롭게 인코드된 타일/프레임을 위해 사용된다.

[0349] 드롭 패킷의 하나의 잠재적인 원인은 예컨대 큰 대역폭을 이용하여 사용자 구내(211)에서 몇몇 다른 사용자의 대역폭 연결이 시작될 때 채널 대역폭의 갑작스런 손실이다. 만약 새롭게 생성된 타일/프레임 또한 드롭된 패킷으로 인해 손실되면(FEC가 사용되었더라도), 호스팅 서비스(210)가 두번째 새롭게 인코드된 타일이 드롭된 것을 클라이언트로 통지하는 일 실시예에서, 비디오 압축기(404)는 다음의 새롭게 인코드된 타일/프레임을 인코드할 때 전송률을 감소시킨다. 다른 실시예들은 다른 기술들을 이용하여 전송률을 감소시킨다. 예컨대, 일 실시예에서, 이러한 전송률 감소는 압축 비율을 증가시켜 그 인코드된 타일/프레임의 품질을 낮춤으로써 달성된다. 다른 실시예에서, 전송률은 비디오의 프레임 비율(예컨대, 60fps 내지 30fps)을 낮추고 그에 따라 데이터 전송률을 느리게함으로써 감소된다. 일 실시예에서, 전송률을 감소시키기 위한 기술 모두(예컨대, 프레임 비율을 감소시키고 압축 비율을 증가시키는)가 이용된다. 이러한 낮은 비율의 데이터 전송이 드롭된 패킷을 성공적으로 감소시키면, 앞서 기술한 채널 전송률 검출 및 조정 방법에 따라, 호스팅 서비스(210)는 계속해서 낮은 전송률로 인코딩하고, 이후 채널 허용에 따라 전송률의 상향 또는 하향을 점진적으로 조정할 것이다. 드롭된 패킷 및/또는 레이턴시와 관련된 피드백 데이터의 연속적인 수신은 현재의 채널 조건에 기초하여 호스팅 서비스(210)가 전송률을 동적으로 조정할 수 있게 한다.

[0350] **온라인 게이밍 시스템의 상태 관리**

[0351] 본 발명의 일 실시예는 서버간 액티브 게임의 현재 상태를 효율적으로 저장 및 포트하기 위한 기술들을 채택한다. 여기에 기술된 실시예들이 온라인 게이밍과 관련되지만, 본 발명의 기초가 되는 원리들은 다양한 다른 타입의 어플리케이션(예컨대, 디자인 어플리케이션, 워드 프로세서, 이메일 또는 인스턴트 메시징 등)에 이용될 것이다. 도 27a는 본 실시예를 실시하기 위한 예시의 시스템 구성을 나타내고, 도 27b는 예시의 방법을 나타낸다. 상기 방법 및 시스템 구성이 동시에 기술되며, 도 27b에 나타낸 방법은 소정의 특정 시스템 구성으로 한정하지 않는다.

[0352] 도 27b의 2751에서, 사용자는 클라이언트 장치(205)로부터 호스팅 서비스(210a) 상의 새로운 온라인 게임을 시작한다. 응답으로, 2752에서, 게임의 "클린(clean)" 이미지(2702a)가 저장소(예컨대, 하드 드라이브, 게임을 실행하는 서버에 직접 연결되거나, 또는 네트워크를 통해 서버에 연결되어 있든지간에)에서 호스팅 서비스(210a)의 메모리(예컨대, RAM)로 로딩된다. "클린" 이미지는 소정 게임 플레이 시작(예컨대, 처음 게임이 실행될 때) 전에 게임에 대한 데이터 및 런타임 프로그램 코드를 포함한다. 다음에, 사용자는 2753에서 게임을 플레이함으로써, "클린" 이미지를 비-클린 이미지(예컨대, 도 27a에서 "상태 A"로 나타낸 실행 게임)로 변경한다. 2754에서, 게임이 사용자 또는 호스팅 서비스(210a)에 의해 일시정지되거나 또는 종료된다. 2755에서, 호스팅 서비스(210a) 상의 상태 관리 로직(2700a)은 게임의 "클린" 이미지와 현재 게임 상태("상태 A")간 차를 결정한다. 예컨대 유닉스 동작 시스템에 이용가능한 잘 알려진 "diff" 유틸리티에 사용된 것을 포함한 2개의 이진 이미지간 차를 계산하기 위해 여러 공지된 기술들이 사용된다. 물론, 본 발명의 기본이 되는 원리들은 차 계산을 위한 소정의 특정 기술들로 한정하지 않는다.

[0353] 차들이 어떻게 계산되는지에 상관없이, 일단 차가 있으면, 그 차 데이터는 저장장치(스토리지; 2705a) 내에 로컬적으로 저장 및/또는 다른 호스팅 서비스(210b)로 전송된다. 만약 다른 호스팅 서비스(210b)로 전송되면, 그 차 데이터는 새로운 호스팅 서비스(210b)의 저장장치(도시하지 않음)에 저장될 것이다. 어떤 경우이든, 그러한 차 데이터는 다음에 호스팅 서비스에 대한 사용자 로그인이 확인되어 게임을 시작하도록 호스팅 서비스의 사용자 계정과 연관된다. 일 실시예에서, 곧바로 전송되는 것이 아니라, 그 차 데이터는 다음에 사용자가 게임을 플레이하려고 시도할 때까지 새로운 호스팅 서비스로 전송되지 않는다(그리고 다른 호스팅 서비스는 게임을 호스팅하기 위한 최선의 선택으로서 확인된다).

[0354] 도 27b에 나타낸 방법으로 되돌아 가서, 2757에서, 사용자는 사용자가 초기에 게임을 플레이한 동일한 클라이언트 장치(205) 또는 다른 클라이언트 장치(도시하지 않음)가 되는 클라이언트 장치로부터 게임을 재시작한다.

응답으로서, 2758에서, 호스팅 서비스(210b) 상의 상태 관리 로직(2700b)은 저장장치로부터 게임의 "클린" 이미지 및 차 데이터를 회복한다. 2759에서, 상태 관리 로직(2700b)은 오리지널 호스팅 서비스(210a) 상의 게임 상태("상태 A")를 회복하기 위해 클린 이미지 및 차 데이터를 조합한다. 예컨대 유닉스 동작 시스템에 이용가능한 잘 알려진 "patch" 유틸리티에 사용된 것을 포함한 차 데이터를 이용하여 이진 이미지의 상태를 재형성하기 위해 여러 공지된 기술들이 사용된다. PC 백업과 같은 잘 알려진 백업 프로그램에 이용된 차 계산 기술들 또한 사용될 것이다. 본 발명의 기본이 되는 원리들은 이진 이미지를 재형성하기 위해 차 데이터를 이용하기 위한 소정의 특정 기술들로 한정하지 않는다.

[0355] 또한, 2760에서, 플랫폼-의존 데이터(2710)는 최종 게임 이미지(2701b) 내에 통합된다. 그러한 플랫폼-의존 데이터(2710)는 목적지 서버 플랫폼에 유일한 소정 데이터를 포함한다. 예로서 한정하지 않으며, 그러한 플랫폼-의존 데이터(2710)는 새로운 플랫폼의 매체 접속 콘트롤(MAC: Medium Access Control) 주소, TCP/IP 주소, 시각, 하드웨어 일련 번호(예컨대, 하드 드라이브 및 CPU에 대한), 네트워크 서버 주소(예컨대, DHCP/Wins 서버), 및 소프트웨어 일련 번호(들)/활성 코드(들)(동작 시스템 일련 번호(들)를 포함한)/활성 코드(들))를 포함한다.

[0356] 클라이언트/사용자와 관련된 다른 플랫폿-의존 데이터는 다음을 포함한다(이들로 한정하지 않음):

[0357] 1. 사용자의 스크린 해상도. 사용자가 게임을 재개할 때, 사용자는 각기 다른 해상도를 갖는 각기 다른 장치를 사용하게 될 것이다.

[0358] 2. 사용자의 콘트롤러 구성. 게임을 재개할 때, 사용자는 게임 콘트롤러에서 키보드/마우스로 전환할 것이다.

[0359] 3. 할인률이 만료되었는지의 여부(예컨대, 사용자가 홍보기간 동안 게임을 플레이하고 이제 좀더 높은 비용의 정상기간 동안 게임을 플레이하는지의 여부) 또는 사용자 또는 장치가 일정한 나이 제한을 갖는지의 여부(예컨대, 사용자의 부모가 아이가 성인물을 볼 수 없게 하기 위해 설정을 변경하거나, 또는 게임을 플레이하는 장치(예컨대, 공공 도서관의 컴퓨터)가 성인물이 플레이될 수 있는지를 일정한 제한을 갖는지의 여부)와 같은 사용자 자격.

[0360] 4. 사용자의 랭킹. 사용자는 소정 리그의 멀티플레이어 게임을 플레이하도록 허용되지만, 몇몇 다른 사용자들은 사용자의 랭킹을 초과하기 때문에, 그러한 사용자는 하위 리그로 강등될 것이다.

[0361] 상기한 예의 플랫폼-의존 데이터(2710)는 본 발명의 이러한 실시예의 설명을 목적을 위해 제공된다. 본 발명의 기본이 되는 원리들은 소정 특정 세트의 플랫폼-의존 데이터로 한정하지 않는다.

[0362] 도 28은 제1호스팅 서비스의 상태 관리 로직(2700a)이 어떻게 실행되고 있는 게임(2701a)으로부터 차 데이터(2800)를 추출하는지를 그래픽적으로 나타낸다. 다음에, 제2호스팅 서비스의 상태 관리 로직(2700b)은 실행된 게임(2701b)의 상태를 재생성하기 위해 차 데이터(2800) 및 플랫폼-의존 데이터(2710)와 클린 이미지(2702b)를 결합한다. 일반적으로 도 28에 나타낸 바와 같이, 차 데이터의 크기는 전체 게임 이미지(2701a)의 크기보다 상당히 작고, 따라서 상당한 양의 저장 공간 및 대역폭이 차 데이터만을 저장/전송함으로써 보존된다. 도 28에 나타내지 않았지만, 플랫폼-의존 데이터(2710)는 그것이 최종 게임 이미지(2701b)에 통합될 때 몇몇 차 데이터를 오버라이트할 것이다.

[0363] 온라인 비디오 게이밍 실행이 상기 기술되었지만, 본 발명의 기본이 되는 원리들은 비디오 게임으로 한정하지 않는다. 예컨대, 상기 상태 관리 기술들은 소정 타입의 온라인-호스트 어플리케이션의 콘텍스트 내에서 실행될 것이다.

[0364] **<u>클라이언트 디코더를 유지하기 위한 기술들</u>**

[0365] 본 발명의 일 실시예에서, 호스팅 서비스(210)는 사용자가 호스팅 서비스(210)에 연결을 요청할 때마다 새로운 디코더를 클라이언트 장치(205)로 전송한다. 따라서, 본 실시예에서, 클라이언트 장치에 의해 사용된 디코더는 항상 업테이트되어 클라이언트 장치에서 실행된 하드웨어/소프트웨어에 아주 잘 맞는다.

[0366] 도 29에 나타낸 바와 같이, 본 실시예에서, 클라이언트 장치(205)에 영구적으로 설치된 어플리케이션은 디코더를 포함하지 않는다. 오히려, 그것은 클라이언트 장치(205)가 호스팅 서비스(210)에 연결될 때마다 임시 디코더(2900)의 다운로드 및 설치를 관리하는 클라이언트 다운로더 어플리케이션(2903)이다. 그러한 다운로더 어플리케이션(2903)은 하드웨어, 소프트웨어, 펌웨어, 또는 그들의 소정 조합으로 실시될 것이다. 새로운 온라

인 세션에 대한 사용자 요청에 따라, 다운로더 어플리케이션(2903)은 네트워크(예컨대, 인터넷)를 통해 클라이언트 장치(205)와 관련된 정보를 전송한다. 그러한 정보는 클라이언트 장치 및/또는 클라이언트 장치의 하드웨어/소프트웨어 구성(예컨대, 프로세서, 동작 시스템 등)을 확인하는 식별 데이터를 포함할 것이다.

[0367] 이러한 정보에 기초하여, 호스팅 서비스(210) 상의 다운로더 어플리케이션(2901)은 클라이언트 장치(205) 상에서 이용되는 적절한 임시 디코더(2900)를 선택한다. 다음에, 호스팅 서비스 상의 다운로더 어플리케이션(2901)은 그 임시 디코더(2900)를 전송하고, 클라이언트 장치 상의 다운로더 어플리케이션(2903)은 클라이언트 장치(205) 상의 디코더를 확인 및/또는 설치한다. 다음에, 인코더(2902)는 여기에 기술된 소정 기술들을 이용하여 오디오/비디오 컨텐츠를 인코드하고 그 컨텐츠(2910)를 디코더(2900)로 전송한다. 일단 새로운 디코더(2900)가 설치되면, 현재 온라인 세션에 대한 컨텐츠를 디코드한다(즉, 여기에 기술된 하나 또는 그 이상의 오디오/비디오 신장 기술을 이용하여). 일 실시예에서, 그 세션이 종료될 때, 디코더(2900)는 클라이언트 장치(205)로부터 삭제된다(예컨대, 제거된다). 일 실시예에서, 다운로더 어플리케이션(2903)은 전송률이 채널 상에서 달성가능한(예컨대, 데이터를 다운로드하는데 얼마다 오래 걸리는지를 결정함으로써) 전송률, 채널 상의 패킷 손실률, 및 채널의 레이턴시와 같은 채널 평가(channel assessment)를 행하여 임시 디코더(2900)가 다운로드됨에 따라 채널을 특성화한다. 다운로더 어플리케이션(2903)은 채널 평가를 기술하는 채널 특성화 데이터를 생성한다. 다음에, 이러한 채널 특성화 데이터는 클라이언트 장치(205)에서 호스팅 서비스 다운로더(2901)로 전송되며, 상기 호스팅 서비스 다운로더는 클라이언트 장치(205)로 미디어를 전송하기 위해 채널을 어떻게 이용하는 것이 최선인지를 결정하기 위해 채널 특성화 데이터를 이용한다.

[0368] 클라이언트 장치(205)는 통상 임시 디코더(2900)의 다운로딩 동안 호스팅 서비스(205)로 메시지를 다시 보낼 것이다. 이들 메시지는 패킷이 에러 없이 또는 에러를 갖고 수신되는지의 여부를 나타내는 확인 통지 메시지를 포함할 수 있다. 또한, 그 메시지는 전송률(패킷이 수신되는 비율에 기초하여 계산된), 패킷 에러율(에러가 있는 것으로 수신 리포트된 패킷의 퍼센테이지에 기초한), 및 채널의 왕복 레이턴시(다운로더(2901)가 전송되는 주어진 패킷에 대한 피드백을 수신하기 전 걸리는 시간에 기초한)에 대해 다운로더(2901)로 피드백을 제공한다.

[0369] 예로서, 만약 전송률이 2Mbps로 결정되면, 다운로더는 전송률이 5Mbps로 결정된 경우의 해상도(예컨대, 60fps에서 1280×720)보다 작은 인코더(2902)에 대한 비디오 윈도우 해상도(60fps에서 640×480)를 선택할 것이다. 다른 순방향 에러 보정(FEC) 또는 패킷 구조는 패킷 손실률에 따라 선택될 것이다.

[0370] 만약 패킷 손실이 매우 낮으면, 압축된 오디오 및 비디오가 소정 에러 보정 없이 전송될 것이다. 만약 패킷 손실이 중간이면, 압축된 오디오 비디오가 에러 보정 코딩 기술(예컨대, 앞서 기술되고 도 11a, 11b, 11c 및 11d에 도시된 것과 같은)에 의해 전송될 것이다. 만약 패킷 손실이 매우 높으면, 적절한 품질의 시청각 스트림이 전송될 수 있는 것으로 결정되고, 클라이언트 장치(205)는 호스팅 서비스가 통신 채널(즉, "링크")을 통해 이용가능하지 않다는 것을 사용자에게 통지하거나, 또는 낮은 패킷 손실을 갖는 호스팅 서비스로 다른 라우트를 구축하려 할 것이다(이하 기술하는 바와 같이).

[0371] 만약 레이턴시가 낮으면, 압축된 오디오 및 비디오가 낮은 레이턴스로 전송될 수 있고 세션이 확립될 수 있다. 만약 레이턴시가 너무 높으면(예컨대, 80ms 이상), 낮은 레이턴시를 필요로 하는 게임에 있어, 클라이언트 장치(205)는 호스팅 서비스가 이용가능하나 입력된 사용자에 대한 응답 시간이 느리거나 또는 "지연"되는 링크를 통해서는 이용가능하지 않거나, 또는 사용자가 낮은 레이턴시를 갖는 호스팅 서비스로 다른 라우트를 구축할 수 있다는 것을 사용자에게 통지할 것이다(이하 기술하는 바와 같이).

[0372] 클라이언트 장치(205)는 장애가 감소되었는지(예컨대, 패킷 손실이 낮고, 레이턴시가 낮거나, 또는 심지어 전송률이 높을 지라도)를 확인하기 위해 네트워크에 걸친 또 다른 라우트를 통해 호스팅 서비스(210)에 연결하려 할 것이다. 예컨대, 호스팅 서비스(210)는 지리적으로 다수의 위치(예컨대, 로스앤젤레스의 호스팅 센터 및 덴버의 호스팅 센터)로부터 인터넷에 연결되고, 아마도 그곳에서 로스앤젤레스의 혼잡으로 인해 높은 패킷 손실이 있으나, 덴버에서는 혼잡이 없다. 또한, 호스팅 서비스(210)는 다수의 인터넷 서비스 공급자(예컨대, AT&T 및 Comcast)를 통해 인터넷에 연결될 것이다.

[0373] 클라이언트 장치(205)와 서비스 공급자 중 하나(예컨대, AT&T)간 혼잡 또는 다른 문제 때문에, 패킷 손실 및/또는 높은 레이턴시 및/또는 한정된 전송률이 야기될 것이다. 그러나, 만약 클라이언트 장치(205)가 또 다른 서비스 공급자(예컨대, Comcast)를 통해 호스팅 서비스(210)에 연결되면, 혼잡 문제 및/또는 낮은 패킷 손실 및/또는 낮은 레이턴시 및/또는 높은 전송률 없이 연결될 수 있다. 따라서, 클라이언트 장치(205)가 임시 디코더(2900)를 다운로딩하는 동안 특정 임계치(특정 기간 동안 드롭된 패킷의 특정 수) 이상의 패킷 손실, 특정 임계치 이상의 레이턴시 및/또는 특정 임계치 이하의 전송률을 경험하면, 일 실시예에서, 최선의 연결이 얻어질 수

있는지를 결정하기 위해 대체 라우트를 통해 호스팅 서비스(210)에 재연결(통상 다른 IP 주소 또는 다른 도메인 네임으로 연결함으로써)을 시도한다.

[0374] 만약 대체 연결 옵션을 다 써버린 후 받아들일 수 없는 장애가 그 연결에 여전히 있으면, 클라이언트 장치(205)의 인터넷에 대한 연결이 장애에 처해지거나, 또는 적적절한 레이턴시를 달성하기 위해 호스팅 서비스(210)로부터 아주 멀리 떨어진다. 그와 같은 경우, 클라이언트 장치(205)는 호스팅 서비스가 링크를 통해 이용가능하지 않거나 또는 단지 장애가 있는 상태로 이용가능하고, 및/또는 단지 소정 타입의 낮은 레이턴시 게임/어플리케이션만이 이용가능하다는 것을 사용자에게 통지한다.

[0375] 호스팅 서비스(210)와 클라이언트 장치(205)간 링크 특성의 이러한 평가 및 잠재적인 개선은 임시 디코더가 다운로드되는 동안 이루어지며, 이는 클라이언트 장치(205)가 임시 디코더(2900)를 독립적으로 다운로딩하고 링크 특성을 평가하는데 걸리는 필요한 시간을 감소시킨다. 그럼에도 불구하고, 다른 실시예에서, 링크 특성의 평가 및 잠재적인 개선은 임시 디코더(2900)를 독립적으로 수행하는 클라이언트 장치(205)에 의해 수행된다(예컨대, 디코더 프로그램 코드가 아닌 더미 테스트 데이터를 이용하여). 거기에는 바람직한 실시를 해야 하는 많은 이유가 있다. 예컨대, 몇몇 실시예에서, 클라이언트 장치(205)는 하드웨어에 있어 부분적으로 또는 전체적으로 실시된다. 따라서, 이들 실시예에서는 소프트웨어 디코더를 본질적으로 다운로드할 필요가 없다.

[0376] **표준-기반 타일 크기를 이용한 압축**

[0377] 상술한 바와 같이, 타일-기반 압축이 사용될 경우, 본 발명의 기본이 되는 원리들은 소정의 특정 타일 크기, 형태, 또는 방위로 한정하지 않는다. 예컨대, MPEG-2 및 MPEG-4와 같은 DCT-기반 압축 시스템에 있어서, 타일은 매크로블럭(통상 16×16 픽셀의 블럭을 나타내는 비디오 압축에 사용된 요소)의 크기가 될 것이다. 본 실시예는 타일을 대상으로 작업하기 위한 아주 미세한 레벨의 입도(granularity)를 제공한다.

[0378] 더욱이, 타일 크기에 상관없이, 다양한 타입의 타일링 패턴이 사용될 것이다. 예컨대, 도 30은 다수의 I-타일이 각각의 R 프레임(3001~3004)에 이용된 실시예를 나타낸다. 풀(full) I-프레임이 모든 4개의 R 프레임에 생성되도록 I-타일이 각각의 R 프레임에 걸쳐 분산되는 회전 패턴이 이용된다. 이러한 방식으로 I-타일을 분산시키는 것은 패킷 손실의 결과를 감소시킬 것이다(디스플레이의 작은 영역으로 손실을 제한).

[0379] 타일은 또한 기본이 되는 압축 알고리즘의 인테그랄 네이티브(integral native) 구조의 크기가 될 것이다. 예컨대, 만약 H.264 압축 알고리즘이 사용되면, 일 실시예에서, 타일은 H.264 "슬라이스"의 크기로 설정된다. 이는 여기에 기술된 기술들이 H.264 및 MPEG-4와 같은 다양한 다른 표준 압축 알고리즘의 콘텍스트를 쉽게 통합시킬 수 있게 한다. 일단 타일 크기가 네이티브 압축 구조로 설정되면, 상술한 것과 동일한 기술들이 실시될 것이다.

[0380] **스트림 되감기 및 플레이백 동작을 위한 기술**

[0381] 도 15와 연결지어 앞서 기술한 바와 같이, 어플리케이션/게임 서버(1521~1525)에 의해 생성된 압축되지 않은 비디오/오디오 스트림(1529)은 다수의 압축된 비디오/오디오 스트림(1539)을 동시에 형성하는 다수의 해상도로 공유된 하드웨어 압축(1530)에 의해 압축될 것이다. 예컨대, 어플리케이션/게임(1521)에 의해 생성된 비디오/오디오 스트림은 공유된 하드웨어 압축(1530)에 의해 1280×720×60fps로 압축되어 아웃바운드 인터넷 트래픽(1599)으로서 아웃바운드 라우팅(1540)를 통해 사용자에게 전송된다. 그러한 동일한 비디오/오디오 스트림은 도 16의 섬네일 컬렉션 중 어느 하나의 섬네일로서 디스플레이되도록 어플리케이션/게임 서버(1522)로 경로(1552)를 통해(또는 지연 버퍼(1515)를 통해) 공유된 하드웨어 압축(1530)에 의해 섬네일 크기(예컨대, 200×113) 아래로 동시에 스케일될 것이다. 섬네일(1600)이 도 17의 중간 크기(1700)를 거쳐 도 18의 크기(1800; 1280×720×60fps)로 확대될 경우, 섬네일 스트림을 신장하기 보다는 오히려, 어플리케이션/게임 서버(1522)는 어플리케이션/게임 서버(1521)의 사용자로 보내지는 1280×720×60fps 스트림의 카피를 신장하고 섬네일 크기에서 1280×720 크기로 확대됨에 따라 보다 높은 해상도 비디오를 스케일할 수 있다. 이러한 접근방식은 2배로 압축된 1280×720 스트림을 재활용하는 장점을 갖는다. 그러나, 이하의 몇가지 단점을 갖는다: (a) 만약 사용자의 인터넷 연결의 데이터 처리율이 어플리케이션/게임 서버(1522)의 "관람" 사용자가 본 변경 이미지 품질의 결과를 변경하고, 심지어 사용자의 인터넷 연결이 변경되지 않을 지라도 사용자에게 보내진 그러한 압축된 비디오 스트림은 이미지 품질이 변할 것이고, (b) 어플리케이션/게임 서버(1522)는 전체 1280×720 이미지를 신장한 후 확대 동안 좀더 작은 크기(예컨대, 640×360)로 디스플레이하기 위해 그 이미지를 스케일하기 위한 프로세싱

리소스를 사용(그리고 가능한 리샘플링 필터를 제공)해야 하며, (c) 만약 프레임이 한정된 인터넷 연결 대역폭 및/또는 손실/손상된 패킷으로 인해 드롭되고, 관람 사용자가 지연 버퍼(1515)에 기록된 비디오를 "되감기" 및 "일시정지"하면, 관람 사용자는 지연 버퍼에서 놓친 드롭된 프레임을 찾을 것이며(이는 특히 사용자가 프레임마다 "단계"를 밟을 경우 특히 뚜렷해질 것이다), (d) 만약 관람 사용자가 지연 버퍼에 기록된 비디오에서 특정 프레임을 찾기 위해 되감기하면, 어플리케이션/게임 서버(1522)는 지연 버퍼에 기록된 비디오 스트림에서 프레임을 찾기 전에 I 프레임 또는 I 타일을 찾은 다음, 원하는 프레임에 도달될 때까지 모든 P 프레임/타일을 신장해야 할 것이다. 이러한 동일한 한정은 사용자에게 "관람" 및 비디오/오디오 스트림 라이브를 제공할 뿐만 아니라, 사용자가 기록보관된 (예컨대, "Brag Clip) 비디오/오디오 스트림의 카피를 보는 것도 제공할 것이다.

[0382] 본 발명의 다른 실시예는 하나 이상의 크기 및/또는 구조의 비디오 스트림을 압축함으로써 이들 문제를 처리한다. 하나의 스트림("라이브" 스트림)은 여기에 기술한 바와 같이 네트워크 연결(예컨대, 데이터 대역폭, 패킷 신뢰도) 및 사용자의 로컬 클라이언트 성능(예컨대, 신장 성능, 디스플레이 해상도)의 특성에 기초하여 말단 사용자에 대한 최적의 스트림으로 압축한다. 다른 스트림(여기서 "HQ" 스트림이라 부르는)은 높은 품질로, 하나 또는 그 이상의 해상도로, 그리고 비디오 플레이백할 수 있는 구조로 압축되며, 그와 같은 HQ 스트림은 서버 센터(210) 내로 라우트되어 저장된다. 예컨대, 일 실시예에서, HQ 압축 스트림은 RAID 디스크 어레이(1515)에 저장되어, 일시정지, 되감기, 및 다른 플레이백 기능(예컨대, 시청을 위해 다른 사용자에게 분배되는 "Brag Clips")과 같은 기능을 제공하기 위해 사용된다.

[0383] 도 31a에 나타낸 바와 같이, 본 발명의 일 실시예는 적어도 2개의 포맷으로 비디오 스트림을 압축할 수 있는 인코더(3100)를 포함한다. 즉 그 어느 하나는 I-타일 또는 I-프레임(3110)을 주기적으로 포함하고, 어느 하나는 스트림의 파괴로 인해 또는 I-타일 또는 I-프레임이 I-타일 또는 I-프레임보다 작을 것이라(상술한 바와 같이) 결정되기 때문에 필요하지 않은 이상 I-타일 또는 I-프레임(3111)을 포함하지 않는다. 예컨대, 비디오 게임을 플레이하는 동안 사용자에 전송된 "라이브" 스트림(3111)은 단지 P-프레임만을 이용하여 압축될 것이다(I-타일 또는 I-프레임이 반드시 필요하거나 상술한 바와 같이 작지 않은 이상). 또한, 본 실시예의 인코더(3100)는 일 실시예에서 I-타일 또는 I-프레임을 주기적으로 포함하는 두번째 포맷(또는 유사한 타입의 이미지 포맷)으로 라이브 비디오 스트림(3111)을 동시에 압축한다.

[0384] 상술한 실시예는 I-타일, I-프레임, P-타일 및 P-프레임을 채택하지만, 본 발명의 기초가 되는 원리는 소정의 특정 압축 알고리즘으로 한정하지 않는다. 예컨대, 프레임이 이전 또는 다음 프레임에 의존하는 소정 타입의 이미지 포맷이 P-타일 또는 P-프레임 대신 사용될 수 있다. 유사하게, 이전 또는 다음 프레임에 의존하는 소정 타입의 이미지 포맷이 상술한 I-타일 또는 I-프레임 대신 대체될 수 있다.

[0385] 상기 언급한 바와 같이, HQ 스트림(3110)은 주기적인 I-프레임(예컨대, 일 실시예에서 모두 12개 프레임 정도)을 포함한다. 이는 사용자가 언제든지 저장된 비디오 스트림을 특정 지점으로 빠르게 되감기를 원할 경우, I-타일 또는 F-프레임이 필요하기 때문에 중요하다. 단지 P-프레임의 압축된 스트림인(즉, I-프레임이 존재하는 시퀀스의 제1프레임이 없는) 경우에는, 디코더를 시퀀스(긴 시간이 걸리는)의 제1프레임으로 되돌리고 사용자가 되감기를 원하는 지점까지 P 프레임을 신장할 필요가 있다. HQ 스트림(3110)에 저장된 12개의 I-프레임인 경우에는, 사용자는 특정 지점으로 되감을 것을 결정할 수 있고 HQ 스트림의 가장 가까운 이전 I-프레임은 원하는 프레임 전에 12개 프레임 이상은 없다. 심지어 디코더의 최대 디코드 비율이 실시간(예컨대 60frame/sec 스트림에 대해 $1/60^{th}$ 초)일 지라도, 12(프레임)/60(frames/sec)=I-프레임으로부터 1/5초 떨어진다. 그리고, 많은 경우에, 디코더는 실시간보다 좀더 빠르게 동작할 수 있어, 예컨대 2x 실시간에서, 디코더는 단지 "되감기"에 대해 $1/10^{th}$ 초 지연이 있는 6개 프레임으로 12개 프레임을 디코드할 것이다. 물론, 만약 가장 가까운 이전 I-프레임이 되감기 지점(되감기에 걸리는 시간 1hour/10=6minute) 전의 많은 수의 프레임이면 고속 디코더(예컨대, 10× 실시간)는 수용할 수 없는 지연을 가질 것이다. 다른 실시예에서, 주기적인 I-타일이 사용되고, 이러한 경우 사용자가 되감기 지점 전의 가장 가까운 이전 I-타일을 찾고, 이후 모든 타일이 되감기 지점에 걸쳐 디코드되는 그러한 타일의 디코딩을 개시할 것이다. 비록 주기적인 I-타일 또는 I-프레임이 I-프레임을 완전히 제거하는 것보다 효과적인 압축이 낮을 지라도, 호스팅 서비스(210)는 통상 H1 스트림을 관리하기 위한 그 이상의 충분한 로컬적으로 이용가능한 대역폭 및 저장 용량을 갖는다.

[0386] 다른 실시예에서, 인코더(3100)는 앞서 기술한 바와 같이 P-타일 또는 P-프레임이 이후에 오지만, B-타일 또는 B-프레임이 앞서는 주기적인 I-타일 또는 I-프레임을 갖는 HQ 스트림을 인코드한다. 앞서 기술한 바와 같은 B-프레임은 I-프레임을 앞서는 프레임이고 적시에 역방향으로 실행되는 I-프레임과 다른 프레임에 기초한다. B-타일은 I-타일에 선행하고 역방향으로 실행되는 I-타일과 다른 프레임에 기초한 상대적인 타일이다. 본 실

시예에서, 원하는 되감기 지점이 B-프레임(또는 B-타일을 포함하는)이면, 디코더는 가장 가까운 이후 I-프레임 또는 I-타일을 찾고 원하는 되감기 지점이 디코드될 때까지 적시에 역방향으로 디코드하고, 다음에 비디오 플레이백이 순방향 지점으로 진행함에 따라 디코더는 순방향의 연속 프레임의 B-프레임, I-프레임 및 P-프레임(또는 이들에 상대적인 타일)을 디코드할 것이다. I 및 P 타입 외에 B-프레임 또는 B-타일 채택의 장점은 종종 주어진 압축 비율에서 보다 높은 품질이 달성될 수 있다는 것이다.

[0387] 또 다른 실시예에서, 인코더(3100)는 모든 I-프레임으로서 HQ 스트림을 인코드한다. 이러한 접근방식의 장점은 모든 되감기 지점이 I-프레임이고, 그 결과 되감기 지점에 도달하기 위해 다른 프레임들이 디코드될 필요가 없다는 것이다. I, P 또는 I, P, B 스트림 인코딩과 비교하여 압축 전송률이 매우 높다는 장점이 있다.

[0388] 각각의 경우 스트림이 적시에 프레임에 걸친 순방향과 다른 프레임 순서로 플레이백되기 때문에, 주기적인 I-프레임 또는 I-타일에 의해 통상 다른 비디오 플레이백 동작들(예컨대, 고속 또는 저속 되감기, 고속 또는 저속 감기 등)이 좀더 실용적으로 달성되고, 결과적으로 디코더는 특정, 종종 임의의 시퀀스의 프레임을 찾아 디코드해야 한다. 예컨대, 초고속 감기의 경우(예컨대, 100× 속도), 디스플레이된 각각의 연속 프레임은 이전 프레임 이후의 100개 프레임이다. 심지어 10× 실시간으로 동작하고 1개 프레임 시간에 10개 프레임을 디코드하는 디코드의 경우에, 그것은 100× 고속 감기를 달성하기에는 너무 느린 10x가 될 것이다. 반면, 상술한 바와 같이 주기적인 I-프레임 또는 I-타일의 경우, 디코더는 다음에 디스플레이해야 하는 프레임에 대한 가장 가까운 적용가능한 I-프레임 또는 I-타일을 찾고 타겟 프레임의 지점에 대해 개재되는 프레임 또는 타일만을 디코드할 수 있다.

[0389] 다른 실시예에서, I-프레임은 일정한 주기로 HQ 스트림으로 인코드되고 그러한 I-프레임 비율보다 빠른 고속 감기 및 되감기를 위해 사용자가 이용가능하게 이루어진 속도 멀티플라이어(multiplier)는 그 I-프레임 주기에 정확히 배수이다. 예컨대, 만약 그 I-프레임 주기가 8개 프레임이면, 사용자가 이용가능하게 이루어진 고속 감기 또는 되감기 속도는 1X, 2X, 3X, 4X, 8X, 16X, 64X, 128X 및 256X가 된다. 그 I-프레임 주기보다 빠른 속도의 경우, 디코더는 우선 속도에 앞서는 다수의 프레임인 가장 가까운 I-프레임 앞으로 점프하고(예컨대, 만약 현재 디스플레이된 프레임이 I-프레임 전 3개 프레임이면, 128X로, 디코더는 128+3개 프레임 앞으로 점프한다), 이후 각각의 연속 프레임에 대해 디코더는 매번 I-프레임 상에 정확하게 하달되는 선택된 속도(예컨대, 128X의 선택된 속도)로 정확한 다수의 프레임을 점프할 것이다. 따라서, I-프레임 주기보다 빠른 주어진 그러한 모든 속도는 정확한 배수의 I-프레임 주기이고, 디코더는 원하는 프레임을 찾기 위해 어떤 선행하는 또는 다음의 프레임을 결코 디코드할 필요가 없고, 단지 디스플레이된 프레임마다 하나의 I-프레임을 디코드해야 할 것이다. 예컨대, I-프레임 주기보다 느린 속도(예컨대, 1X, 2X, 3X, 4X)의 경우, 또는 비배수의 I-프레임 주기인 것보다 빠른 속도의 경우, 각각의 디스플레이된 프레임에 대해, 디코더는, 디코드된 형태로(RAM 또는 다른 고속 스토리지에서) 여전히 이용가능한 디코드되지 않은 I-프레임 또는 이미 디코드된 프레임이면, 프레임을 디스플레이하기 위해 프레임이 적어도 추가의 새롭게 디코드된 프레임을 요구하는 어떤 것이든 찾은 후, 원하는 프레임이 디코드되어 디스플레이될 때까지 필요에 따라 개재의 프레임을 디코드한다. 예컨대, 4X 고속 감기에, 8X I-프레임 주기를 갖는 I, P 인코드된 시퀀스에 있어서, 만약 현재 프레임이 I-프레임 다음에 오는 1개 프레임인 P-프레임이면, 디스플레이되는 원하는 프레임은 이후의 4개 프레임으로, 즉 이전 I-프레임 다음에 오는 5^{th} P-프레임이 될 것이다. 만약 현재 디스플레이된 프레임(금방 디코드된)이 시작 지점으로 사용되면, 디코더는 원하는 프레임을 디스플레이 하기 위해 4개 이상의 P-프레임을 디코드해야 한다. 만약 이전 I-프레임이 사용되면, 디코더는 원하는 프레임을 디스플레이하기 위해 6개 프레임(I-프레임 및 연속하는 5개 P-프레임)을 디코드해야 한다(분명히, 이러한 경우는 디코드하기 위한 추가의 프레임을 최소화하기 위해 현재 디스플레이된 프레임을 사용하는 장점이 있다). 다음에, 4개 프레임 앞의 디코드될 다음 프레임은 I-프레임 다음에 오는 1^{st} P-프레임이 될 것이다. 이 경우, 현재 디코드된 프레임이 시작 지점으로 사용되면, 디코더는 4개 이상의 프레임(2개 P-프레임, I-프레임 및 P-프레임)을 디코드해야 한다. 그러나, 만약 다음 I-프레임이 대신 사용되면, 디코더는 단지 I-프레임 및 연속의 P-프레임을 디코드해야 한다(분명히, 이러한 경우는 디코드하기 위한 추가의 프레임을 최소화하기 위해 시작 지점으로 다음 I-프레임을 사용하는 장점이 있다). 따라서, 본 예에서, 디코더는 시작 지점으로 현재 디코드된 프레임을 사용하는 것과 시작 지점으로 다음 I-프레임을 사용하는 것 중 선택할 것이다. HQ 비디오 스트림 플레이백 모드(고속 감기, 되감기 또는 스텝) 및 속도에 상관없이, 일반적인 원리에 따라, 디코더는 I-프레임 또는 앞서 디코드된 프레임이면 어떤 프레임이라도 시작할 수 있고, 그러한 플레이백 모드 및 속도로 디스플레이된 각각의 연속 프레임에서 원하는 프레임을 디스플레이하기 위해 최소의 새롭게 디코드된 프레임을 필요로 한다.

[0390] 도 31b에 나타낸 바와 같이, 호스팅 서비스(210)의 일 실시예는 HQ 스트림(3110)을 리플레이(replay)하기 위해 사용자 요청을 관리하기 위한 스트림 리플레이 로직(3112)을 포함한다. 스트림 리플레이 로직(3112)은 비디오 플레이백 명령(예컨대, 일시정지, 되감기, 특정 지점으로 플레이백 등)을 포함하는 클라이언트 요청을 수신하고, 그 명령을 해석하고, 특정 지점(I-프레임 또는 앞서 디코드된 프레임으로 시작하는, 그리고 적절하다면 특정 지점으로 순방향 또는 역방향 진행하는)으로부터 HQ 스트림(3110)을 디코드한다. 일 실시예에서, 디코드된 HQ 스트림은 신장되어(여기에 기술한 기술들을 이용하여) 클라이언트 장치(205)로 전송되도록 인코더(3100; 즉시 하나 이상의 스트림을 인코딩할 수 있으면, 잠재적으로 그러한 동일한 인코더 3100, 또는 독립된 인코더 3100))에 제공된다. 다음에, 클라이언트 장치 상의 디코더(3102)는 상술한 바와 같이 스트림을 디코드하여 제공한다.

[0391] 일 실시예에서, 스트림 리플레이 로직(3112)은 HQ 스트림을 디코드하지 않고 이후 인코더(3100)가 그 스트림을 재인코드하게 한다. 오히려, 간단히 HQ 스트림(3110)을 특정 지점으로부터 클라이언트 장치(205)로 곧바로 스트림한다. 다음에, 클라이언트 장치(205) 상의 디코더(3102)는 HQ 스트림을 디코드한다. 통상 여기에 기술한 플레이백 기능이 실시간 비디오 게임을 플레이함에 따라 동일한 낮은 레이턴시 필요조건을 갖지 않으므로(예컨대, 플레이어가 액티브하게 플레이하지 않고 단순히 게임플레이 전에 시청만하는 경우), 일반적으로 높은 품질의 HQ 스트림이 본래부터 갖고 있는 추가된 레이턴시는 수용가능한 말단 사용자가 경험하게 한다(예컨대, 높은 레이턴시이지만 높은 품질의 비디오를 갖는).

[0392] 예로서 한정하지 않으며, 만약 사용자가 비디오 게임을 플레이하면, 인코더(3100)는 사용자의 연결 및 로컬 클라이언트를 위해 최적화된 본질적인 모든 P-프레임의 라이브 스트림을 제공한다(예컨대, 640×360 해상도로 약 1.4Mbps). 동시에, 인코더는 또한 호스팅 서비스(210) 내의 HQ 스트림(3110)과 같은 비디오 스트림을 압축하여, 로컬 디지털 비디오 디코더 RAID 어레이 상에 예컨대 1280×720의 12개 프레임마다 I-프레임을 갖는 HQ 스트림을 저장한다. 만약 사용자가 "일시정지" 버튼을 누르면, 클라이언트의 마지막 디코드된 프레임으로 일시정지되어 스크린이 정지할 것이다. 이후 사용자가 "되감기" 버튼을 누르면, 스트림 리플레이 로직(3112)은 상술한 바와 같이 가장 가까운 I-프레임 또는 이용가능한 이미 디코드된 프레임으로부터 시작하는 DVR RAID로부터 HQ 스트림(3110)을 읽을 것이다. 스트림 리플레이 로직(3112)은 개재되는 P 또는 B 프레임을 신장하고, 필요에 따라 플레이백 시퀀스가 원하는 되감기 속도로 백워드(backward)되도록 프레임을 재시퀀스하고, 이후 1280×720에서 640×360로 디스플레이하기 위해 원하는 디코드된 의도된 크기로 크기를 조절하며(공지된 종래의 이미지 스케일링 기술을 이용하여), 그리고 라이브 스트림 인코더(3100)는 640×360 해상도로 재시퀀스된 스트림을 재신장하여 사용자에게 전송한다. 만약 사용자가 다시 일시정지하고, 이후 주의깊게 시퀀스를 지켜보기 위해 비디오를 단일-스텝 쓰루(single-step through)하면, DVR RAID 상의 HQ 스트림(3110)은 단일 스텝핑에 이용가능한 모든 프레임을 가질 것이다(비록 오리지널 라이브 스트림이 여기에 기술한 소정의 많은 이유에 따라 드롭된 프레임을 가질 지라도). 더욱이, 비디오 플레이백의 품질이 HQ 스트림의 모든 지점에서 상당히 높아지고, 반면 거기에는 예컨대 대역폭이 손상되어 압축된 이미지 품질의 일시적인 감소를 야기하는 라이브 스트림의 지점이 존재한다. 짧은 시간에, 또는 이동하는 이미지의 손상된 이미지 품질이 사용자에게 수용가능한 반면, 만약 사용자가 특정 프레임(또는 느리게 단일-스텝)에서 정지하여 주의깊게 프레임을 눈여겨 본다면, 손상된 품질은 수용불가능해질 것이다. 또한 사용자에게는 고속 감기 능력, 또는 HQ 스트림 내의 지점(예컨대, 2분 전)을 확인함으로써 특정 지점으로 점프하는 능력이 제공된다. 모든 이들 동작은 단지 P-프레임만이 있는 또는 드물게(또는 예상치 못하게) I-프레임을 갖는 라이브 비디오 스트림의 높은 품질에 있어 그리고 대부분에 있어 비실용적이다.

[0393] 일 실시예에서, HQ 스트림을 백하는 한, 사용자가 비디오 스트림에 걸쳐 순방향 및 역방향으로 스윕(sweep)하게 하는 "스크러버(scrubber)"(즉, 좌-우측 슬라이더 콘트롤)를 갖는 애플 퀵타임(Apple QuickTime) 또는 어도비 플레시 비디오 윈도우(Adobe Flash video window)와 같은 비디오 윈도우(도시하지 않음)가 사용자에게 제공된다. 마치 라이브 스트림에 걸친 "스크러빙(scrubbing)"인 것처럼 사용자에게 보일지라도, 실제로는 다음에 라이브 스트림으로 크기가 재조정되고 재압축되는 저장된 HQ 스트림(3110)에 걸친 스트러빙이다. 또한, 앞서 언급한 바와 같이, 만약 HQ 스트림을 동시에 또는 다른 시간에 어떤 누군가가 지켜볼 경우, 라이브 스트림의 해상도보다 높은(또는 낮은) 해상도가 목격될 수 있지만 HQ 스트림이 동시에 인코드되고, 그 품질이 잠재적으로 HQ 스트림의 품질까지 시청자의 라이브 스트림의 품질만큼 높아질 것이다.

[0394] 따라서, 라이브 스트림(낮은 레이턴시, 대역폭 및 패킷 에러-허용 조건에 적합한 방식으로 여기에 기술한 바와 같은) 및 높은 품질, 스트림 플레이백 동작 조건을 갖는 HQ 스트림 모두를 동시에 인코딩함으로써, 사용자에게 양쪽 장면의 원하는 구성이 제공된다. 그리고, 실제로는 다르게 인코드되는 2개의 다른 스트림이 존재하는 것

이 사용자에게 효과적으로 비춰진다. 사용자의 시각으로부터, 그러한 경험은, 크게 변할 수 있으면서 비교적 낮은 대역폭 인터넷 연결에 의해 동작함에도 불구하고, 낮은 레이턴시에 크게 반응하며, 여전히 디지털 비디오 레코딩(DVR) 기능성은 유연한 동작 및 유연한 속도와 함께 매우 높은 특성이다.

[0395] 상술한 기술의 결과로서, 사용자는 라이브 스트림 또는 HQ 스트림의 소정의 한계로부터 손해보지 않고 온라인 게임플레이 또는 다른 온라인 상호작용 동안 라이브 및 HQ 비디오 스트림 모두의 이점을 수용한다.

[0396] 도 31c는 상기 동작들을 수행하기 위한 시스템 구성의 일 실시예를 나타낸다. 나타낸 바와 같이, 본 실시예에서, 인코더(3100)는 일련의 "라이브(Live)" 스트림(3121L, 3122L, 3125L) 및 대응하는 일련의 "HQ" 스트림(3121H1~H3, 3122H1~H3, 3125H1~H3)을 각각 인코드한다. 각각의 HQ 스트림(H1)은 풀 해상도로 인코드되고, 반면 각각의 인코더(H2, H3)는 인코딩 전에 보다 작은 크기로 비디오 스트림으로 스케일된다. 예컨대, 만약 비디오 스트림이 1280×720 해상도이면, H1은 1280×720 해상도로 인코드되고, 반면 H2는 640×360으로 스케일되어 그 해상도로 인코드되며, H3은 320×180으로 스케일되어 그 해상도로 인코드된다. 다양한 해상도로 다수의 동시 HQ 인코딩을 제공하는 소정 수의 동시의 Hn 스케일러/인코더가 사용될 수 있다.

[0397] 각각의 라이브 스트림은 상술한 바와 같이(예컨대, 도 25~26의 피드백 신호 2501 및 2601의 설명 참조) 인바운드 인터넷 커넥션(3101)을 통해 수신된 채널 피드백 신호(3161, 3162, 3165)에 따라 동작한다. 라이브 스트림은 아웃바운드 라우팅 로직(3140)을 통해 인터넷(또는 다른 네트워크)을 통해 전송된다. 라이브 압축기(3121L~3125L)는 채널 피드백에 기초한 압축된 비디오 스트림(스케일링, 드롭핑 프레임 등을 포함하는)을 채택하기 위한 로직을 포함한다.

[0398] HQ 스트림은 신호 경로(3151)를 통해 내부 지연 버퍼(예컨대, RAID 어레이(3115)) 또는 다른 데이터 저장장치로 인바운드 라우팅 로직(3141, 1502)에 의해 라우트되고 및/또는 추가의 처리를 위해 어플리케이션/게임 서버 및 인코더(3100)로 신호 경로(3152)를 통해 피드백된다. 상술한 바와 같이, HQ 스트림(3121Hn~3125Hn)은 실질적으로 요청에 따라 말단 사용자에게 스트림된다(예컨대, 도 31b 및 관련된 문장 참조).

[0399] 일 실시예에서, 인코더(3100)는 도 15에 나타낸 공유된 하드웨어 압축 로직(1530)으로 실시된다. 다른 실시예에서, 몇몇 또는 모든 인코더 및 스케일러는 개별 서브시스템이다. 본 발명의 기본이 되는 원리들은 소정의 특정 공유의 스케일링 또는 압축 리소스 또는 하드웨어/소프트웨어 구성으로 한정하지 않는다.

[0400] 도 31c의 구성의 장점은 풀-사이즈 비디오 윈도우보다 작은 것을 필요로 하는 어플리케이션/게임 서버(3121~3125)가 풀-사이즈 윈도우를 처리 및 신장할 필요가 없다는 것이다. 또한, 중간의 윈도우 사이즈를 필요로 하는 어플리케이션/게임 서버(3121~3125)는 원하는 윈도우 사이즈에 가까운 압축된 스트림을 수신할 수 있고, 이후 원하는 윈도우 사이즈로 스케일 업 또는 다운할 수 있다. 또한, 만약 다수의 어플리케이션/게임 서버(3121~3125)가 다른 어플리케이션/게임 서버(3121~3125)로부터 동일한 사이즈의 비디오 스트림을 요청하면, 인바운드 라우팅(3141)은 공지된 바와 같은 IP 멀티캐스트 기술을 실시하고, 각각의 어플리케이션/게임 서버로 독립된 스트림을 요청하지 않고 즉시 다수의 어플리케이션/게임 서버(3121~3125)로 그 요청된 스트림을 브로드캐스트(broadcast)할 수 있다. 만약 어플리케이션/게임 서버가 브로드캐스트를 수신하는 어플리케이션/게임 서버가 비디오 윈도우의 사이즈를 변경하면, 다른 비디오 사이즈의 브로드캐스트로 바꿀 수 있다. 따라서, 임의의 다수의 사용자가 어플리케이션/게임 서버 비디오 스트림을 동시에 볼 수 있고, 이들 각각은 비디오 스케일링의 유연성을 갖고 항상 원하는 윈도우 크기에 가깝게 스케일된 비디오 스트림의 이점을 갖는다.

[0401] 도 31c에 나타낸 접근방식의 한가지 장점은 호스팅 서비스(210)의 많은 실제 실시에 있어서 모든 크기의 모든 압축된 HQ 스트림이 즉시 단독으로 보여지는 경우가 없다는 것이다. 인코더(3100)가 공유된 리소스(예컨대, 소프트웨어 또는 하드웨어에서 실시되는 스케일러/압축기)로서 실시될 경우, 이러한 낭비가 감소된다. 그러나, 거기에는 수반된 대역폭으로 인해 공통 공유된 리소스에 많은 수의 비압축 스트림을 연결하는데 실질적인 문제가 있다. 예컨대, 각각의 1080p60 스트림은 거의 3Gbps이며, 이는 동등한 기가비트 이더넷(Gigabit Ethernet)을 훨씬 초과한다. 다음의 다른 실시예들은 이러한 문제를 처리한다.

[0402] 도 31d는 각각의 어플리케이션/게임 서버(3121~3125)가 (1) 채널 피드백(3161~3165)에 기초한 압축된 비디오 스트림을 채택하는 라이브 스트림 압축기(3121L~3125L), 및 (2) 상술한 바와 같이 풀-해상도 HQ 스트림을 출력하는 HQ 스트림 압축기로 할당된 2개의 압축기를 갖춘 호스팅 서비스(210)의 다른 실시예를 나타낸다. 그 중에서도 특히, 클라이언트(205)와 양방향 통신을 이용하는 라이브 압축기는 동적이면서 적합한 반면, HQ 스트림은 단방향이면서 비적합하다. 스트림간 다른 차이는 라이브 스트림 품질이 비디오 자료의 특성 및 채널 조건에 따라 급격하게 변한다는 것이다. 몇몇 프레임은 품질이 좋지 않고, 거기에는 드롭된 프레임이 존재한다. 또

한, 라이브 스트림은 드물게 나타나는 I-프레임 또는 I-타일과 함께 거의 완전히 P-프레임 또는 P-타일이 존재한다. 통상 HQ 스트림은 라이브 스트림보다 더 높은 전송률을 가지며, 소정 프레임의 드롭없이 안정한 높은 품질을 제공한다. HQ 스트림은 모드 I-프레임이 존재하거나, 또는 빈번한 그리고/또는 규칙적인 I-프레임 또는 I-타일을 갖는다. HQ 스트림은 또한 B-프레임 또는 B-타일을 포함한다.

[0403] 일 실시예에서, 공유된 비디오 스케일링 및 재압축(3142; 이하 상세히 기술)은 단지 앞서 기술한 바와 같이 라우팅하기 위해 인바운드 라우팅(3141)으로 보내지 전에 하나 또는 그 이상의 다른 해상도로 스케일 및 재압축되는 소정의 HQ 비디오 스트림(3121H1~3125H1)만을 선택한다. 또 다른 HQ 비디오 스트림은 앞서 기술한 바와 같이 라우팅하기 위해 인바운드 라우팅(3141)으로 풀 사이즈로 통과하거나, 또는 전혀 통과하지 않는다. 일 실시예에서, HQ 스트림이 스케일 및 재압축되고 그리고/또는 HQ 스트림이 조금이라도 통과하는 결정은 특정 해상도(또는 스케일된 또는 풀 해상도에 가까운 해상도)로 그 특정 HQ 스트림을 요청하는 어플리케이션/게임 서버(3121~3125)가 있는지에 기초하여 결정된다. 이러한 수단을 통해, 스케일 및 재압축된(또는 잠재적으로 조금이라도 통과되는) HQ 스트림만이 실제 필요한 HQ 스트림이다. 호스팅 서비스(210)의 많은 어플리케이션에서, 이는 스케일링 및 압축 리소스의 크나큰 감소를 제공한다. 또한, 주어진 그러한 모든 HQ 스트림은 압축기(3121H1~3125H1)에 의해 풀 해상도로 적어도 압축되고, 공유된 비디오 스케일링 및 재압축(3142)으로 라우트되어야 하는 대역폭은 수용된 비압축 비디오가 존재하는 것보다 크게 감소된다. 예컨대, 3Gbps 비압축 1080p60 스트림은 10Mbps로 압축되어 여전히 매우 높은 품질을 유지한다. 따라서, 기가비트 이더넷 연결의 경우, 오히려 하나의 압축된 3Gbps를 수행할 수 없고, 수십개의 10Mbps 비디오 스트림을 수행할 수 있어 어느정도 품질의 명백한 감소를 제공한다.

[0404] 도 31f는 좀더 많은 수의 HQ 비디오 압축기(HQ 3121H1~3131H1)와 함께 공유된 비디오 스케일링 및 재압축(3142)을 상세한 나타낸다. 어플리케이션/게임 서버(3121~3125)로부터 특정 사이즈로 스케일된 특정 비디오 스트림의 요청마다 내부 라우팅(3192)은 통상 HQ 비디오 압축기(HQ 3121H1~3131H1)로부터 서브세트의 압축된 HQ 스트림을 선택한다. 이러한 선택된 서브세트의 스트림 내의 스트림은 만약 요청된 스트림이 스케일되면 신장기(3161~3164)를 통해 라우트되거나, 또는 만약 요청된 스트림이 풀 해상도이면 비스케일된 비디오 경로(3196)으로 라우트된다. 스케일되는 스트림들은 신장기(3161~3164)에 의해 비압축된 비디오로 신장되고, 다음에 스케일러(3171~3174)에 의해 요청된 사이즈로 각각 스케일된 후, 압축기(3181~3184)에 의해 각각 압축된다. 만약 특정 HQ 스트림이 하나 이상의 해상도로 요청되면, 내부 라우팅(3192)은 그 스트림을 하나 또는 그 이상의 신장기(3161~3164)로 그리고 아웃바운드 라우팅(3193; 풀 해상도이고 하나의 요청된 사이즈이면)으로 멀티캐스트(종래 기술자들에게 잘 알려진 IP 멀티캐스팅을 이용하여)한다는 것을 염두해 두자. 다음에, 모든 요청된 스트림이 스케일되었는지(압축기(3181~3184)로부터) 또는 되지 않았는지(내부 라우팅(3192)으로부터)의 여부는 아웃바운드 라우팅(3193)으로 보내진다. 다음에, 라우팅(3193)은 각각의 요청된 스트림을 그 요청한 어플리케이션/게임 서버(3121~3125)로 보낸다. 일 실시예에서, 만약 하나 이상의 어플리케이션/게임 서버가 동일한 해상도로 그 동일한 스트림을 요청하면, 아웃바운드 라우팅(3193)은 요청하는 모든 어플리케이션/게임 서버(3121~3125)으로 그 스트림을 멀티캐스트한다.

[0405] 공유된 비디오 스케일링 및 재압축(3142)의 현재의 바람직한 실시예에서, 그 라우팅은 기가비트 이더넷 스위치를 이용하여 실시되고, 신장, 스케일링, 및 압축은 각 기능을 수행하는 각 개별의 특화된 반도체 장치에 의해 실시된다. 그러한 동일한 기능이 초고속 프로세서에 의해 또는 좀더 높은 레벨의 통합된 하드웨어로 실시된다.

[0406] 도 31e는 호스팅 서비스(210)의 다른 실시예를 나타내며, 앞서 기술한 지연 버퍼(3115)의 기능은 공유된 비디오 지연 버퍼, 스케일링 및 신장 서브시스템(3143)으로 실시된다. 도 31g에는 서브시스템(3143)이 상세히 나타나 있다. 서브시스템(3143)의 동작은 3191을 제외하고는 도 31f에 나타낸 서브시스템(3142)과 유사하고, 우선 어플리케이션/게임 서버(3121~3125)로부터의 요청마다 HQ 비디오 스트림이 라우트되고, 다음에 지연되도록 요청된 HQ 스트림은 현재의 바람직한 실시예의 RAID 어레이로서 실시된(그러나 충분한 대역폭 및 용량의 소정 저장 매체로 실시되는) 지연 버퍼(3194)를 통해 라우트되며, 지연되도록 요청되지 않은 스트림은 비지연 비디오 경로(3195)를 통해 라우트되는지를 선택한다. 다음에, 지연 버퍼(3194) 및 비지연 비디오(3195) 모두의 출력은 요청된 스트림이 스케일되었는지의 여부에 기초하여 내부 라우팅(3192)에 의해 라우트된다. 스케일된 스트림은 신장기(3161~3164), 스케일러(3171~3174) 및 압축기(3181~3184)를 통해 아웃바운드 라우팅(3193)으로 라우트되고, 비스케일된 비디오(3193) 또한 아웃바운드 라우팅(3193)으로 보내지며, 다음에 아웃바운드 라우팅(3193)은 그 비디오를 앞서 도 31f의 서브시스템(3142)에 기술한 바와 같은 동일한 방식으로 어플리케이션/게임 서버로 유니캐스트 또는 멀티캐스트 모드로 보낸다.

[0407] 도 31h에는 비디오 지연 버퍼, 스케일링 및 신장 서브시스템(3143)의 다른 실시예가 나타나 있다. 본 실시예에서, 각 개별 지연 버퍼(HQ 3121D~HQ 3131D)는 각각의 HQ 스트림을 위해 제공된다. 개별 압축된 비디오 스트림을 지연하기 위해 사용될 수 있는 RAM 및 플레쉬 ROM의 급격한 비용 감소를 제공하고, 이는 공유된 지연 버퍼(3194)를 갖춘 것보다 좀더 비용이 저렴하고 그리고/또는 좀더 유연해지게 한다. 또, 또 다른 실시예에서, 단일의 지연 버퍼(3197; 점선으로 나타낸)는 높은 성능의 집합 리소스(예컨대, 고속 RAM, 플레쉬 또는 디스크)로 개별적으로 모든 HQ 스트림에 대한 지연을 제공할 수 있다. 시나리오에 있어서, 각각의 지연 버퍼(HQ 3121D~3131D)가 HQ 비디오 소스로부터 스트림을 가변적으로 지연하거나, 또는 비지연의 스트림을 통과시킬 수 있다. 다른 실시예에서, 각각의 지연 버퍼는 다른 지연량을 갖는 다수의 스트림을 제공할 수 있다. 모든 지연 또는 비지연은 어플리케이션/게임 서버(3121~3125)에 의해 요청된다. 이러한 모든 경우, 지연 및 비지연된 비디오 스트림은 내부 라우팅(3192)으로 보내지고, 도 31g와 관련하여 앞서 기술한 바와 같은 나머지의 서브시스템(3143)을 통해 진행한다.

[0408] 상기 실시예들에서, 도 31n과 관련된 것은 라이브 스트림이 양방향 연결이고 최소 레이턴시로 특정 사용자에게 맞추어져 있다는 것을 알아 두자. HQ 스트림은 단방향 연결을 이용하며, 유니캐스트 및 멀트캐스트이다. 그러한 멀티캐스트 기능이 기가비트 이더넷 스위치로 실시되는 것과 같이 단일 유닛으로 이들 도면에 도시되어 있는 반면, 대규모 시스템에서 그러한 멀티캐스트 기능은 트리(tree)의 다수 스위치를 통해 실시될 것이다. 덧붙여, 상위 랭크의 비디오 게임 플레이어로부터의 비디오 스트림의 경우는 플레이어의 HQ 스트림을 수백만의 사용자가 동시에 지켜보는 경우가 될 것이다. 그와 같은 경우에는 멀티캐스트된 HQ 스트림을 브로드캐스팅하는 연속 스테이지의 많은 수의 개별 스위치들이 존재할 것이다.

[0409] 쌍방의 원인 분석의 목적을 위해, 그리고 사용자에게 피드백을 제공하기 위해(예컨대, 실행하고 있는 게임플레이가 얼마나 인기가 있는지를 사용자에게 알려주기 위해), 일 실시예에서, 호스팅 서비스(210)는 각각의 어플리케이션/게임 서버(3121~3125)에 많은 동시 시청자의 트랙을 유지시킬 것이다. 이는 특정 비디오 스트림을 위한 어플리케이션/게임 서버에 의한 다수의 액티브 요청의 러닝 카운트(running count)를 유지함으로써 달성될 수 있다. 따라서, 100,000 동시 시청자를 갖고 있는 게이머는 자신의 게임플레이가 얼마나 인기가 있는지를 알 수 있고, 이는 게임플레이어가 게임을 좀더 잘 수행하게 하고 시청자의 관심을 더 잘 끌게하기 위한 자극을 야기할 것이다. 비디오 스트림(예컨대, 챔피언쉽 비디오 게임 경기)의 시청자수가 아주 많을 경우, 이는 비디오 게임 경기 동안 해설하는 해설자에게도 효율적이므로, 멀티캐스트를 지켜보는 사용자의 일부 또는 모두가 해설을 들을 수 있다.

[0410] 어플리케이션/게임 서버 상에서 진행되는 어플리케이션 및 게임에는 그러한 어플리케이션 및/또는 게임이 특정 특성(예컨대, 해상도 및 지연량)의 특정 비디오 스트림에 대한 요청을 제공할 수 있는 어플리케이션 프로그램 인터페이스(API; Application Program Interface)가 제공될 것이다. 또한, 도 4a의 호스팅 서비스 컨트롤 시스템에 또는 어플리케이션/게임 서버 상에서 운용되는 동작 환경에 제공된 이들 API는 그와 같은 다양한 이유의 요청을 거절할 수 있다. 예컨대, 요청된 비디오 스트림은 소정 라이센싱 권리의 제한(예컨대, 단지 단일 시청자만이 시청할 수 있고, 다른 시청자들에게는 방송하지 않도록)을 둘 수 있는데, 거기에는 가입 제한(예컨대, 시청자가 그러한 스트림을 시청하기 위한 권리에 대해 비용을 지불해야 하고)이 있고, 나이 제한(예컨대, 시청자가 스트림을 시청하기 위해서는 18세가 되어야 하고)이 있고, 프라이버시 제한(예컨대, 어플리케이션을 이용하고 게임을 플레이하는 사람이 선택된 시청자 수 또는 시청자 부류(예컨대, 자신의 "친구들")를 시청할 수 있도록 제한하거나, 또는 전혀 시청할 수 없게 하며)이 있으며, 또한 자료 요청의 제한(예컨대, 만약 사용자가 자신의 위치가 노출되는 비밀 게임을 플레이할 경우)이 있다. 또한 스트림의 시청을 제한해야 하는 소정 다수의 또 다른 제한이 있다. 이들의 경우, 어플리케이션/게임 서버에 의한 요청은 상기한 거절 이유에 의해 거절될 수 있고, 일 실시예에서 대안으로서 그러한 요청이 받아들여질 수 있을 것이다(예컨대, 가입에 대한 비용 지불이 유지되어 있어야 한다).

[0411] 소정의 상기 선행 실시예의 지연 버퍼에 저장된 HQ 비디오 스트림은 호스팅 서비스(210) 외부의 다른 목적지로 수송될 것이다. 예컨대, 유튜브(YouTube)로 보내기 위한 특정 관심있는 비디오 스트림이 어플리케이션/서버(통상 사용자의 요청에 의해)에 의해 요청될 수 있다. 그와 같은 경우, 그러한 비디오 스트림은 적절한 설명 정보(예컨대, 플레이하는 사용자의 이름, 게임, 시간, 점수 등)와 함께 유튜브에서 동의된 포맷으로 인터넷을 통해 전송될 것이다. 이것은 그와 같은 해결을 요청하는 모든 게임/어플리케이션 서버(3121~3125)로 해설 오디오를 각각의 스트림으로 멀티캐스팅함으로써 실시될 것이다. 게임/어플리케이션 서버는 종래 기술자들에게 잘 알려진 오디오 혼합 기술들을 이용하여 그러한 해설 오디오를 사용자 구내(211)로 보내진 오디오 스트림에 합병한다. 거기에는 다수의 해설자(예컨대, 각기 다른 관전 포인트, 또는 다른 언어를 갖는)가 있을 것이고,

사용자는 그들 중에서 선택할 것이다.

[0412] 유사한 방식으로, 각각의 독립된 오디오 스트림들이 지연 버퍼로부터 또는 실시간으로 비디오 스트리밍으로부터 오디오를 혼합하거나 교체함으로써, 호스팅 서비스(210)에서 혼합되거나 또는 특정 비디오 스트림(또는 각 개별 스트림)의 대체자로 제공될 것이다. 그와 같은 오디오는 해설 또는 서술되거나, 또는 비디오 스트림의 캐릭터의 음성을 제공할 것이다. 이는 사용자가 쉽게 머시니마(Machinima; 비디오 게임 비디오 스트림으로부터 사용자가 만든 애니메이션)를 만들 수 있게 할 것이다.

[0413] 본 문서에 걸쳐 기술된 비디오 스트림들은 어플리케이션/게임 서버의 비디오 출력으로부터 캡처되고, 이후 다양한 방식으로 스트림 및/또는 지연과 재사용 또는 분배되는 것으로 나타내고 있다. 그러한 동일한 지연 버퍼가 비어플리케이션/게임 서버 소스로부터 오고 적절한 제한으로 플레이백 및 분배에 대한 동일한 정도의 유연성을 제공하는 비디오 자료를 유지하기 위해 사용될 수 있다. 그와 같은 소스는 텔레비전 방송국(CNN과 같은 비방송, 또는 방송, 그리고 HBO와 같은 유료, 또는 무료)으로부터의 라이브 피드(live feed)를 포함한다. 또한, 그와 같은 소스는 미리 저장된 영화 또는 텔레비젼 쇼, 가정용 영화, 광고 및 원격 제공된 라이브 비디오를 포함한다. 라이브 피드는 게임/어플리케이션 서버의 라이브 출력과 같이 처리될 것이다. 미리 저장된 자료는 지연 버퍼의 출력과 같이 처리될 것이다.

[0414] 일 실시예에서, 여기서 예시하는 다양한 기능의 모듈은 및 관련 단계는 그 단계를 수행하기 위한 주문형 반도체(ASIC) 또는 프로그램된 컴퓨터 구성 요소 및 사용자 정의 하드웨어 구성 요소의 어떤 조합에 의한 하드웨어 로직을 포함하는 특별한 하드웨어 구성요소에 의해서 수행될 수 있다.

[0415] 일 실시예에서, 모듈은 텍사스인스트루먼트의 TMS320x 아키텍처(예, TMS320C6000, TMS320C5000, ... 등)와 같은 프로그램 가능한 디지털 신호 프로세서("DSP")로 구현될 수 있을 것이다. 다양한 다른 DSP는 여전히 이러한 근본적인 이론을 따르는 동안에 사용될 수 있을 것이다.

[0416] 실시예는 위에 정한 내용으로 여러 단계를 포함할 수 있다. 그 단계는 기계-실행가능한(machine-executable) 명령어에 의해서 범용 또는 특수 용도의 프로세서가 어떤 단계를 수행할 수 있도록 구현될 수 있다. 이러한 기본 원칙에 관련된 컴퓨터 메모리, 하드 드라이브, 입력 장치와 같은 이러한 기본 이론에 관련되지 않는 다양한 구성 요소는 적절한 측면을 불분명하게 하는 것을 피하기 위해서 도면에서 빠지게 된다.

[0417] 개시된 주제의 문제는 또한 기계-실행가능한 명령어를 저장하기 위해서 기계-판독가능한 매체로서 제공될 수 있다. 기계-판독가능한 매체는 플래시 메모리, 광학 디스크, CD-ROM, DVD-ROM RAM, EPROM, EEPROM, 자기 또는 광학 카드, 전파 매체(propagation media), 전자적 명령어(electronic instructions)를 저장하기에 적합한 그 밖의 형태의 기계-판독가능한 미디어 등을 포함하되 이에 제한되지 않는다. 예컨대, 본 발명은 통신 링크(예컨대, 모뎀, 네트워크 연결)를 경유하는 그 밖의 전파 매체 또는 캐리어 웨이브(carrier wave)에 포함된 데이터 신호의 방식에 의해서 원격 컴퓨터(예컨대, 서버)에서 요청하는 컴퓨터(예컨대, 클라이언트)로 전송될 수 있는 컴퓨터 프로그램처럼 다운로드될 수 있을 것이다.

[0418] 또한, 이는 개시된 주제의 문제의 요소가 시퀀스의 오퍼레이터를 수행하기 위해서 컴퓨터 프로그램하는데 사용될 수 있는 저장된 명령어를 가진 기계적으로 판독가능한 미디어를 포함하는 컴퓨터 프로그램 제품으로써 제공될 수 있을 것이다(예컨대, 프로세서 또는 그 밖의 전자 장치). 택일적으로, 오퍼레이션은 하드웨어 및 소프트웨어의 조합으로 수행될 수 있을 것이다. 기계-판독가능한 매체는 플로피 디스켓, 광학 디스크, 자기-광학 디스크, ROM, RAM, EPROM, EEPROM, 자기 또는 광학 카드, 전파 매체 또는 그 밖의 형태의 미디어/전기적 명령어를 저장하기에 적합한 기계-판독가능한 미디어를 포함할 수 있으나 이에 국한되지 않는다. 예컨대, 개시된 주제의 문제 요소는 컴퓨터 프로그램 제품처럼 다운로드될 수 있을 것이고, 이러한 프로그램은 통신 링크를 경유하여 그 밖의 전파 매체 또는 캐리어 웨이브에 포함된 데이터 신호의 방식에 따라 처리를 요청하기 위한 원격 컴퓨터 또는 전자 장치로 전송될 수 있을 것이다(예컨대, 모뎀 또는 네트워크 연결).

[0419] 추가적으로, 비록 개시된 주제의 문제가 특정 실시예와 결합되어 설명되었어도, 다수의 변형 및 치환은 본 개시의 범위 내에 잘 설명되어 있다. 따라서, 명세서 및 도면은 제한적인 센스보다는 예시적인 센스로 간주될 것이다.

부호의 설명

[0420] 205 : 클라이언트 장치, 206 : 인터넷,

210 : 호스팅 서비스, 211 : 사용자 구내,

220 : 소프트웨어 개발자, 401 : 호스팅 서비스 컨트롤 시스템,

402 : 서버, 403 : 스토리지 영역 네트워크,

404 : 비디오 압축 로직, 413 : 제어 신호 로직,

415 : 클라이언트, 422 : 디스플레이 장치,

462 : 이더넷 잭, 479 : 블루투스 입력 장치,

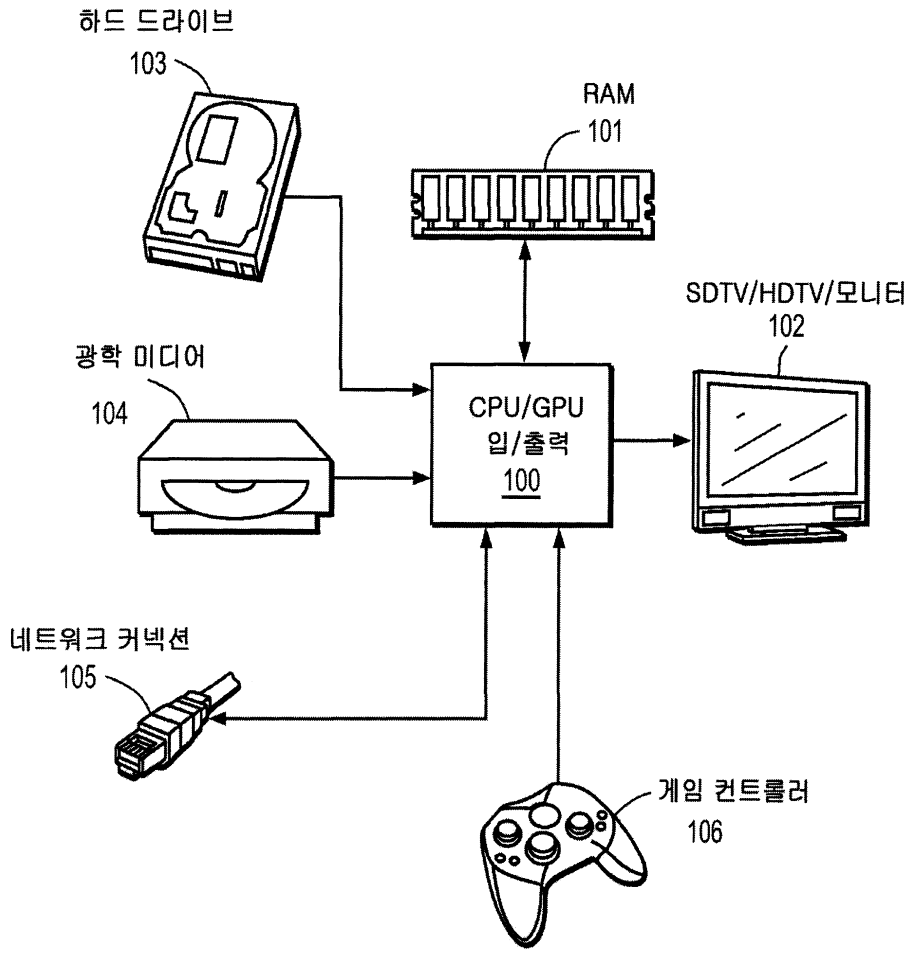
2500,2600,2902,3100 : 인코더, 2502,2602,2900,3102 : 디코더,

2900 : 디코더 로직, 2901 : 다운로더,

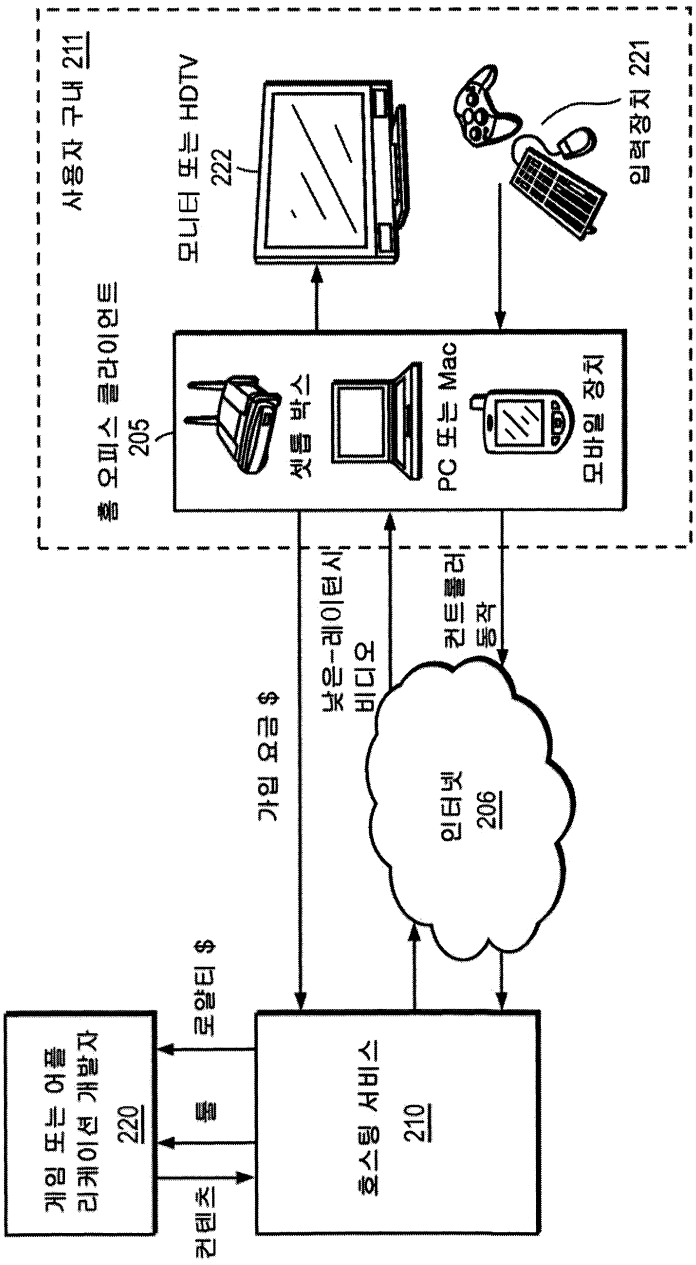
2903 : 클라이언트 다운로더.

도면

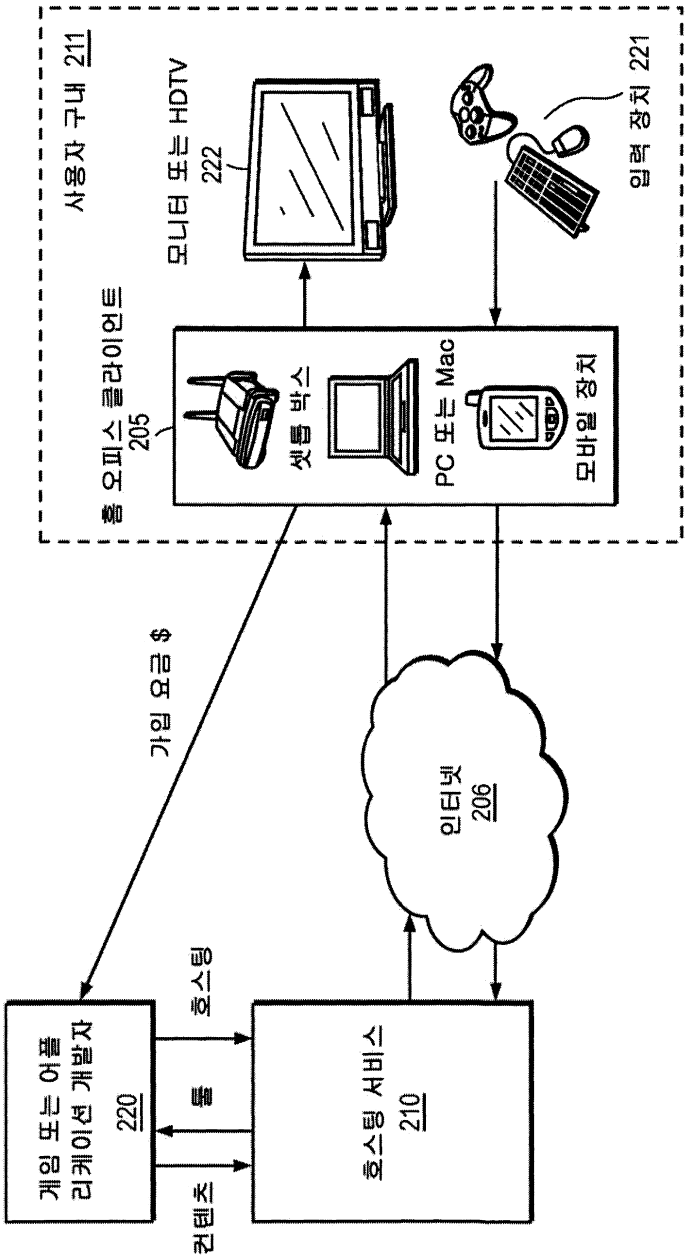
도면1



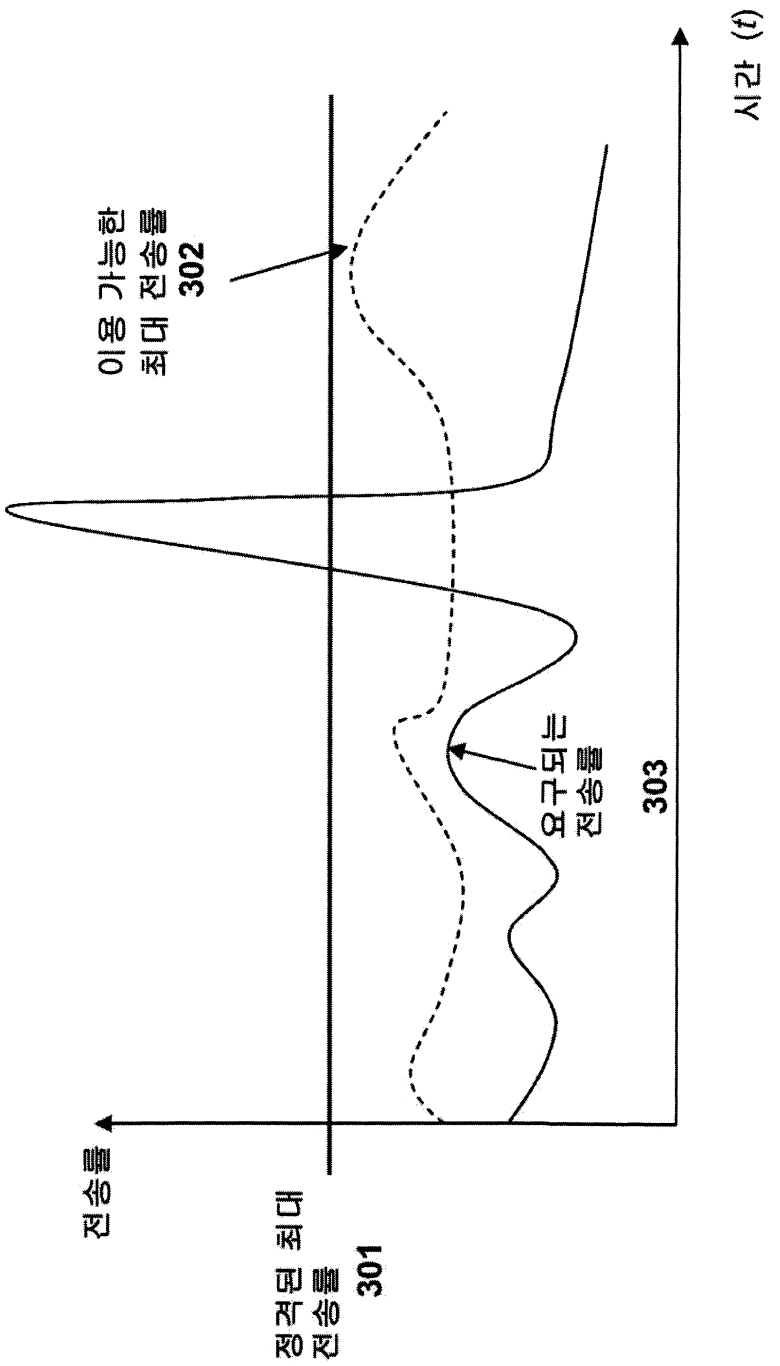
도면2a


사용자 구내 211
홈 오피스 클라이언트
205
모니터 또는 HDTV
222
입력장치 221
셋톱 박스
PC 또는 Mac
모바일 장치
낮은-레이턴시
비디오
컨트롤러
동작
가입 요금 $
인터넷
206
로얄티 $
툴
컨텐츠
게임 또는 어플
리케이션 개발자
220
호스팅 서비스
210

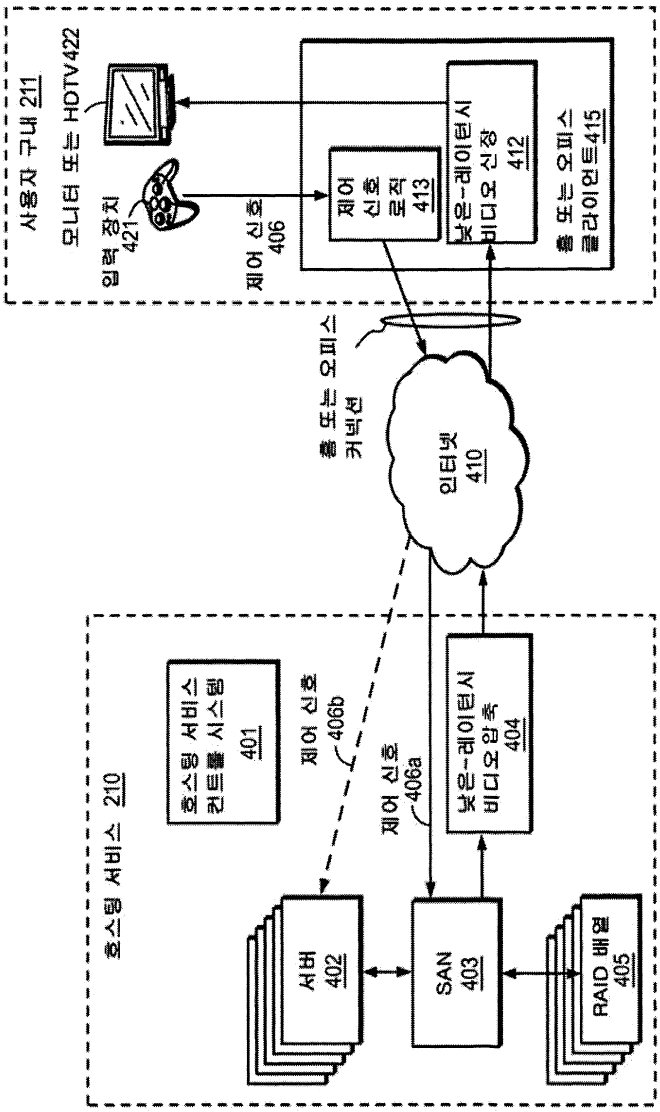
도면2b


사용자 구내 211
홈 오피스 클라이언트
205
셋톱 박스
PC 또는 Mac
모바일 장치
모니터 또는 HDTV
222
입력 장치 221
가입 요금 $
인터넷
206
게임 또는 어플
리케이션 개발자
220
호스팅
툴
컨텐츠
호스팅 서비스
210

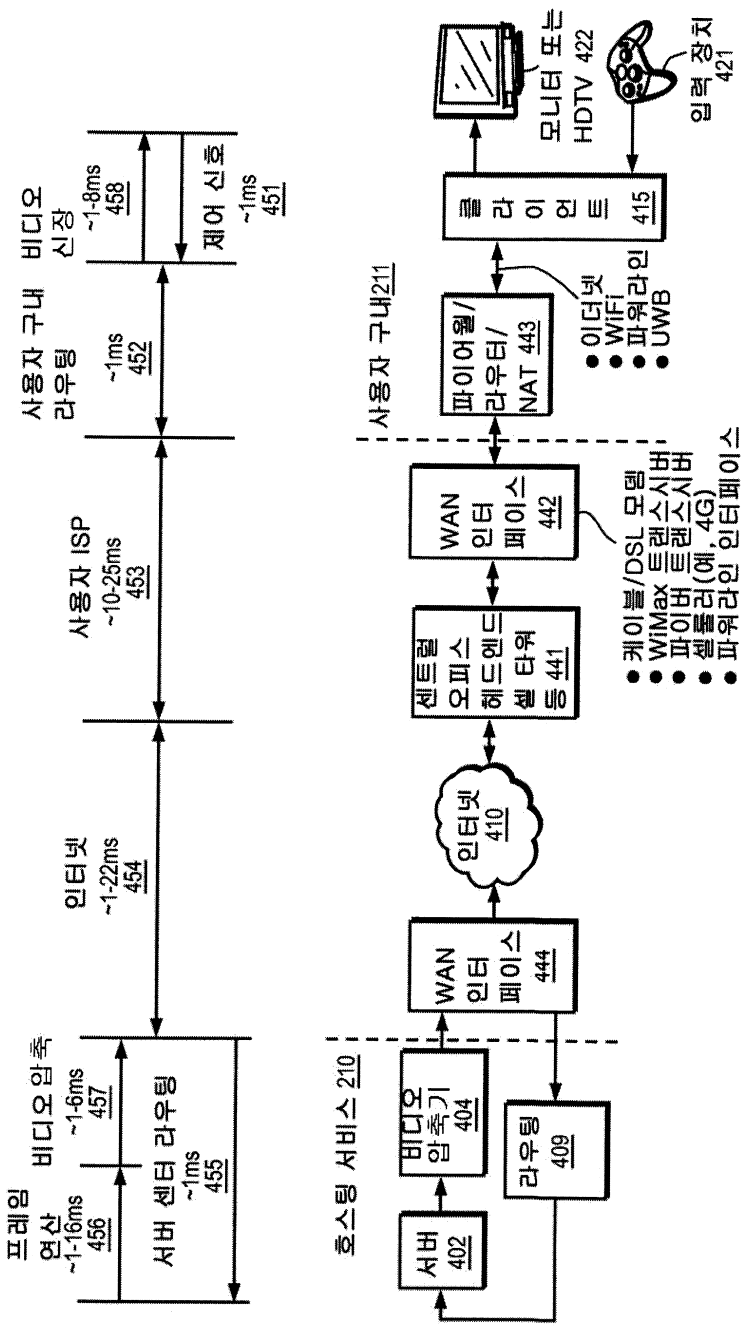
도면3



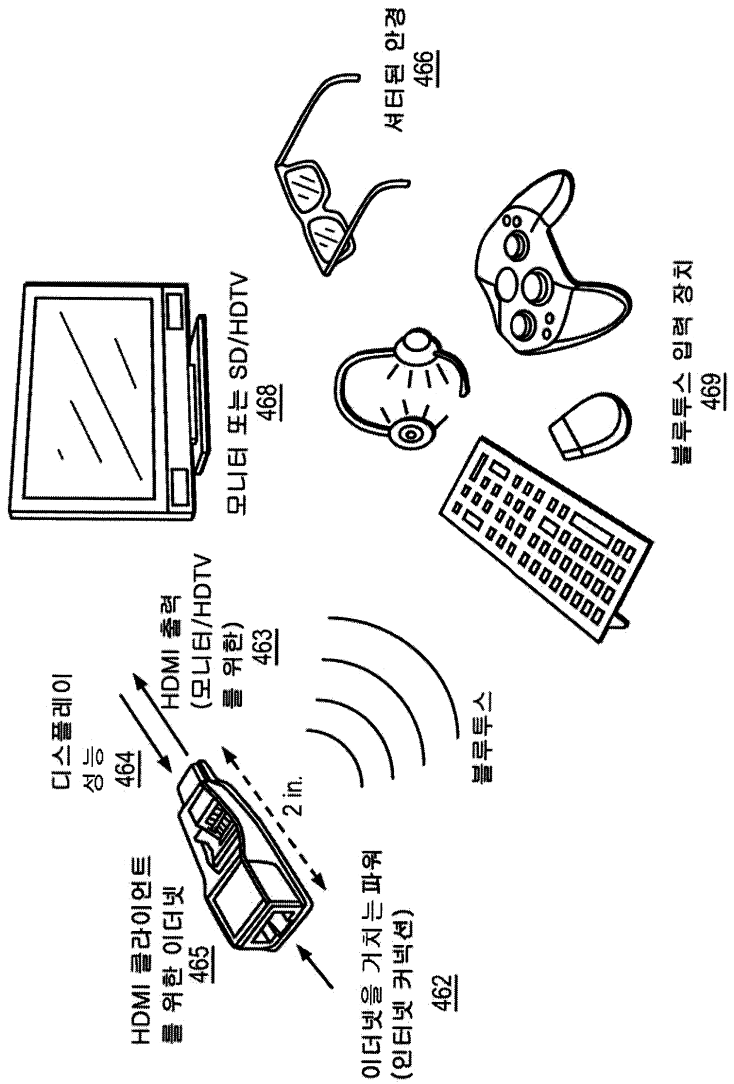
도면4a


사용자 구내 211
모니터 또는 HDTV422
입력 장치 421
제어 신호 406
제어 신호 로직 413
낮은-레이턴시 비디오 신장 412
홈 또는 오피스 클라이언트415
홈 또는 오피스 커넥션
인터넷 410
호스팅 서비스 210
호스팅 서비스 컨트롤 시스템 401
제어 신호 406b
제어 신호 406a
낮은-레이턴시 비디오압축 404
서버 402
SAN 403
RAID 배열 405

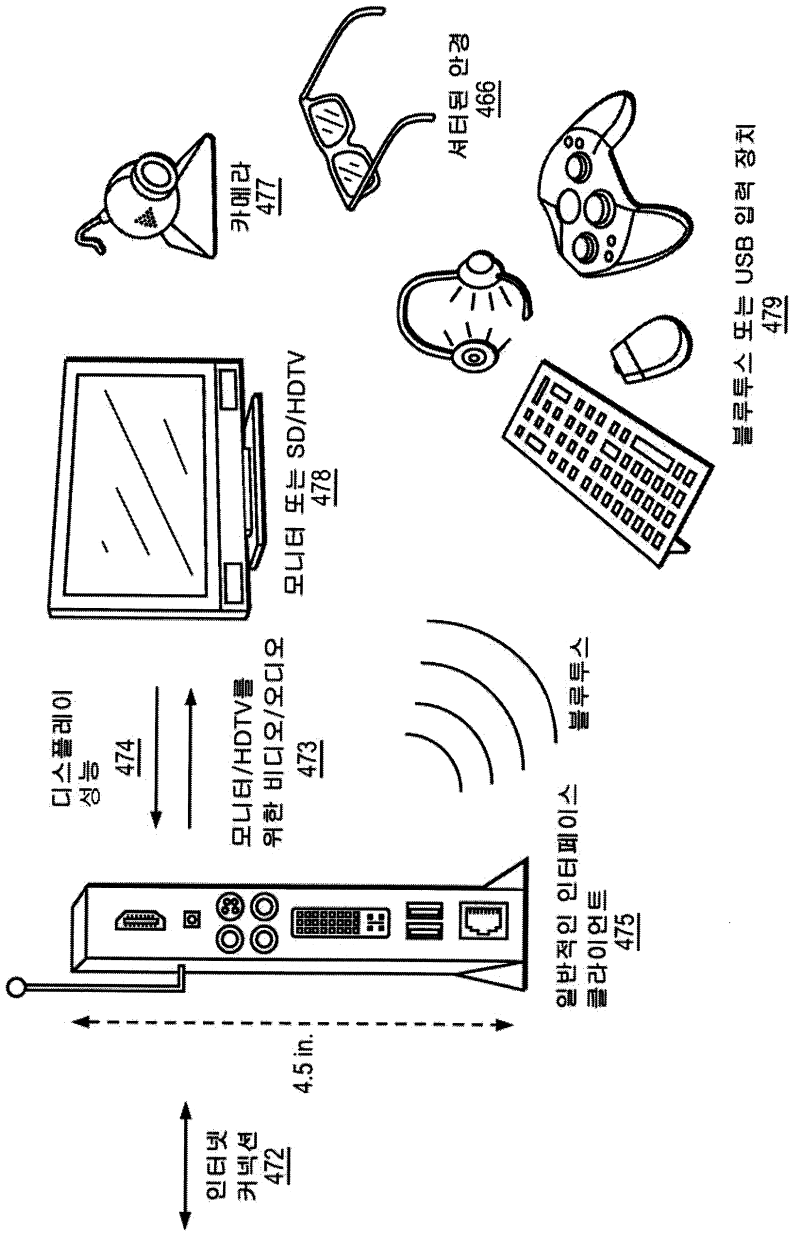
도면4b


모니터 또는 HDTV 422
입력 장치 421
클라이언트 415
파이어월/라우터/NAT 443
● 인터넷
● WiFi
● 파워라인
● UWB
사용자 구내211
WAN 인터페이스 442
● 케이블/DSL 모뎀
● WiMax 트랜스시버
● 파이버 트랜스시버
● 셀룰러(예, 4G)
● 파워라인 인터페이스
센트럴 오피스 헤드엔드 셀 타워 등 441
인터넷 410
WAN 인터페이스 444
비디오 압축기 404
라우팅 409
서버 402
호스팅 서비스 210
사용자 구내 비디오 신장 ~1-8ms 458
제어 신호 ~1ms 451
사용자 구내 라우팅 ~1ms 452
사용자 ISP ~10-25ms 453
인터넷 ~1-22ms 454
비디오 압축 ~1-6ms 457
서버 센터 라우팅 ~1ms 455
프레임 연산 ~1-16ms 456

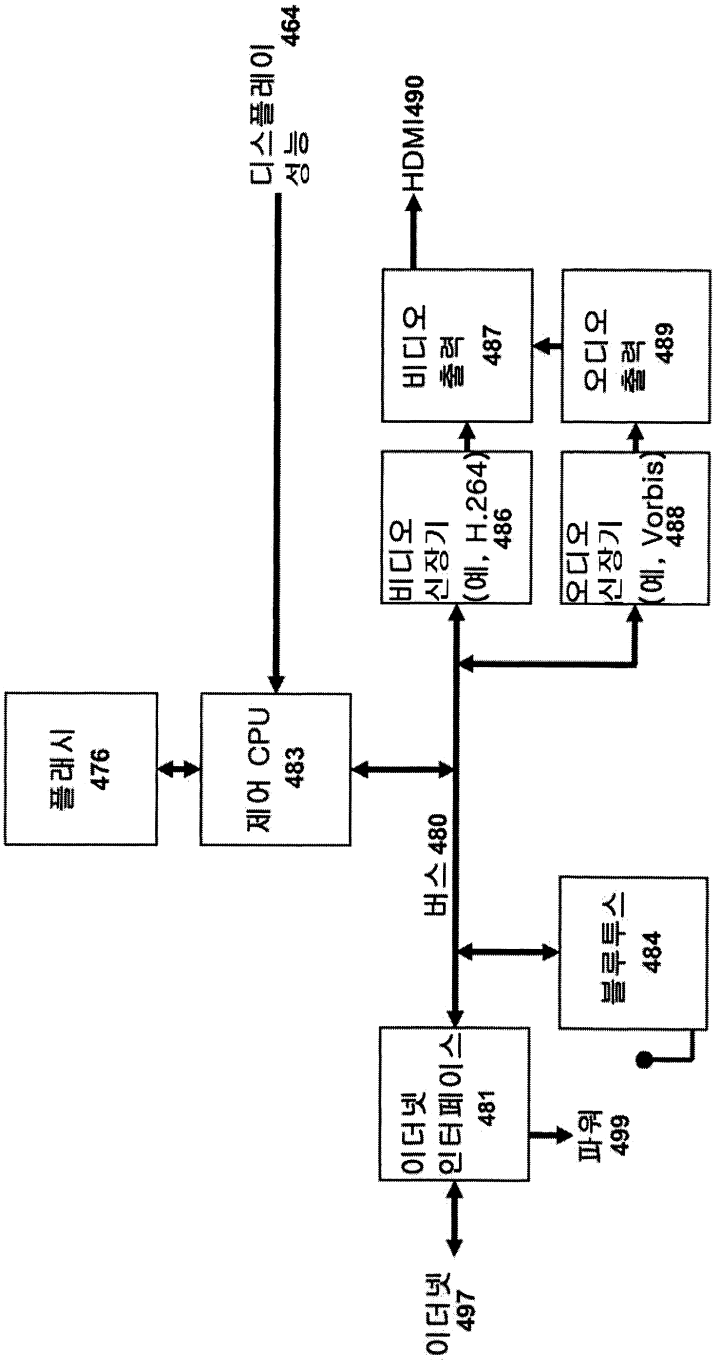
도면4c


셔터된 안경
466
모니터 또는 SD/HDTV
468
블루투스 입력 장치
469
디스플레이
성능
464
HDMI 출력
(모니터/HDTV
를 위한)
463
2 in.
블루투스
HDMI 클라이언트
를 위한 이더넷
465
이더넷을 거치는 파워
(인터넷 커넥션)
462

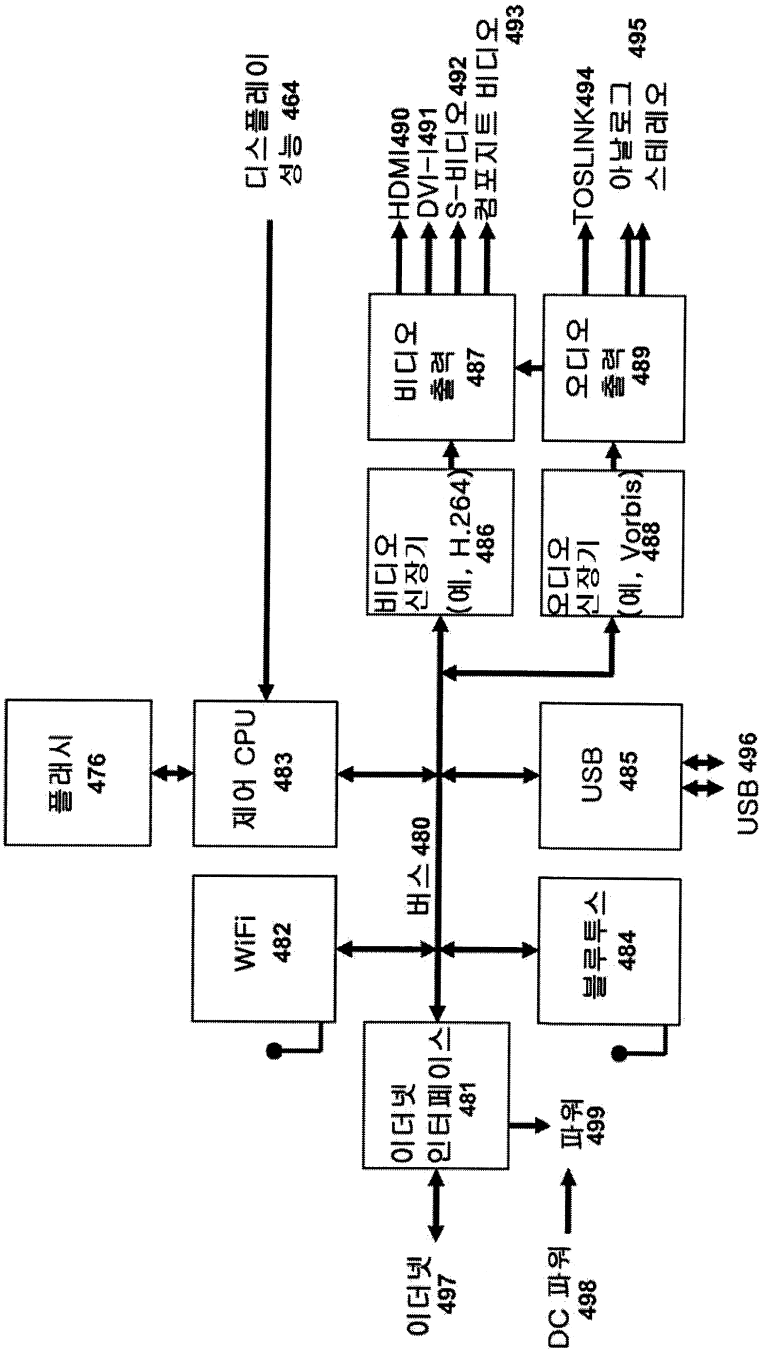
도면4d


인터넷
커넥션
472
4.5 in.
일반적인 인터페이스
클라이언트
475
블루투스
모니터/HDTV를
위한 비디오/오디오
473
디스플레이
성능
474
모니터 또는 SD/HDTV
478
블루투스 또는 USB 입력 장치
479
셔터된 안경
466
카메라
477

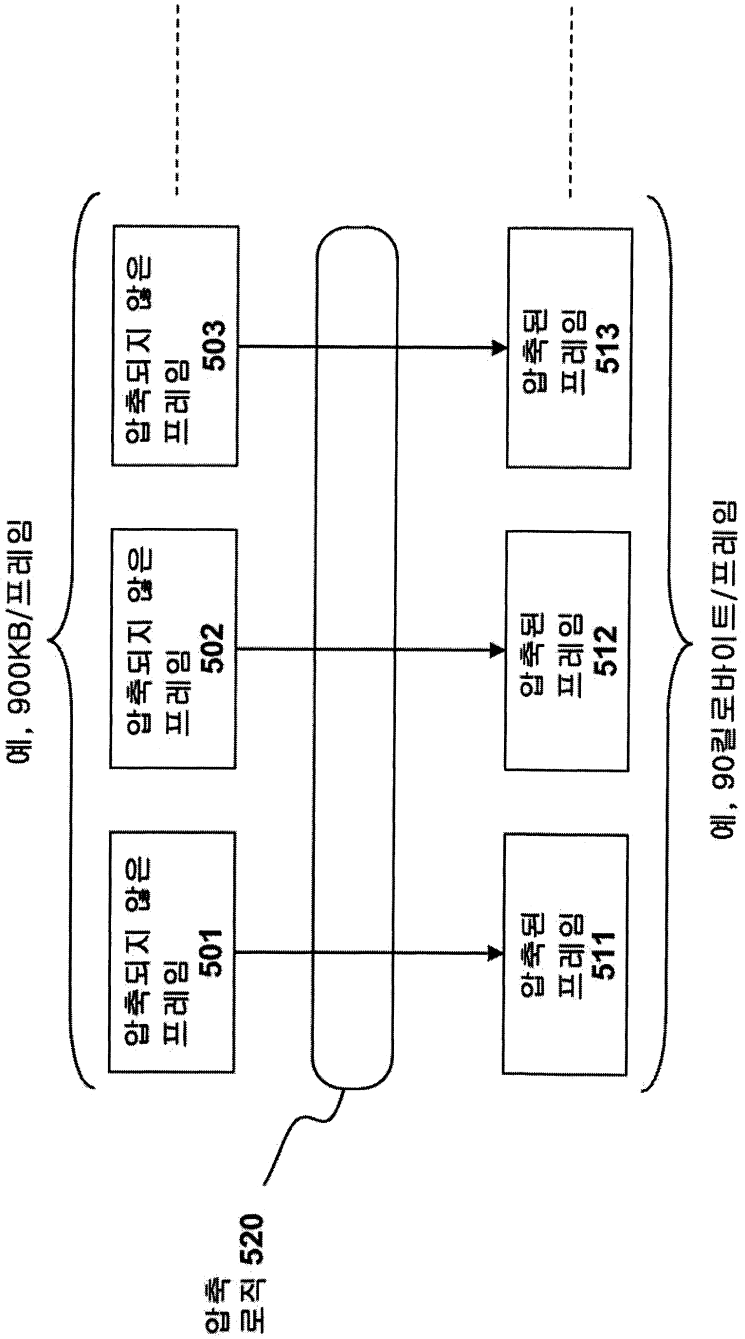
도면4e


디스플레이 성능
464
HDMI490
비디오 출력
487
오디오 출력
489
비디오 신장기 (예, H.264)
486
오디오 신장기 (예, Vorbis)
488
플래시
476
제어 CPU
483
버스 480
블루투스
484
이더넷 인터페이스
481
파워
499
이더넷
497

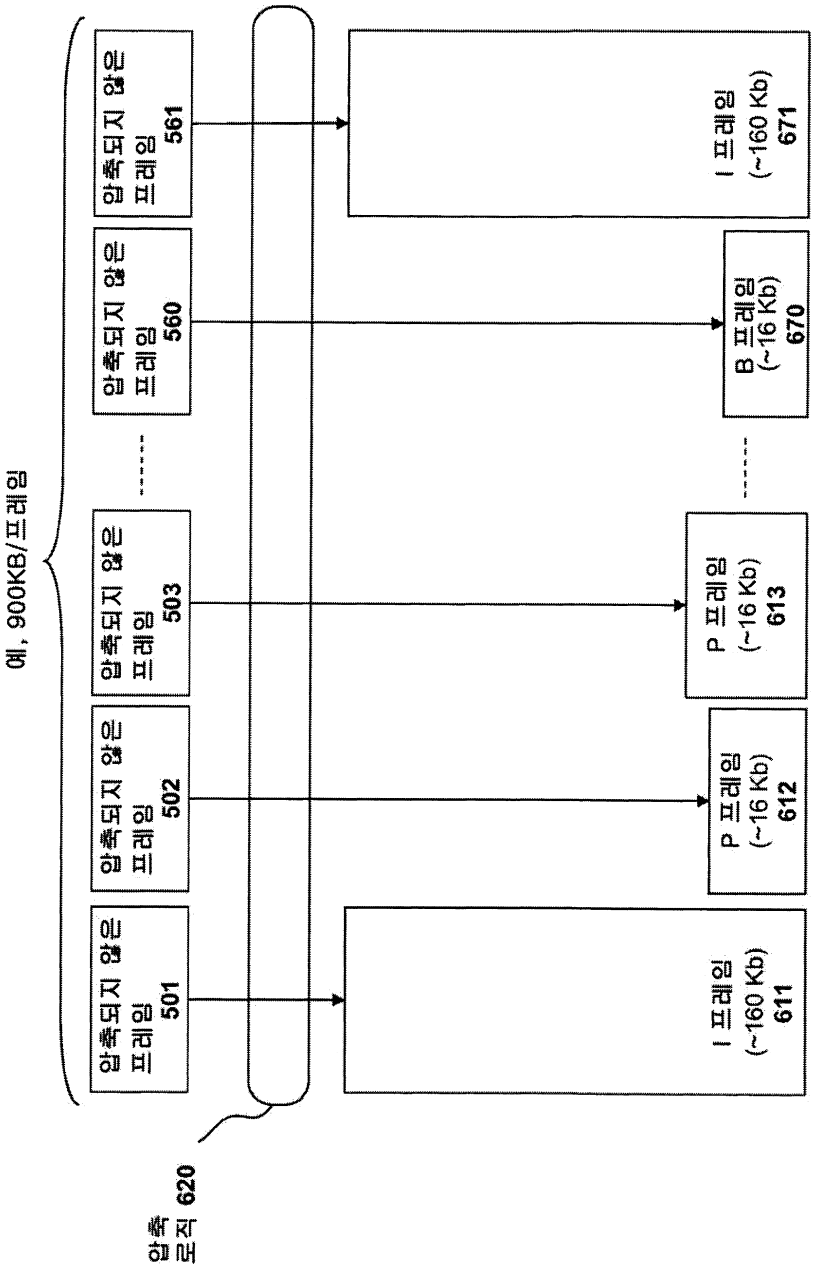
도면4f


디스플레이 성능 464
HDMI490
DVI-I491
S-비디오 492
컴포지트 비디오 493
TOSLINK494
아날로그 스테레오 495
비디오 출력 487
오디오 출력 489
비디오 신장기 (예, H.264) 486
오디오 신장기 (예, Vorbis) 488
플래시 476
제어 CPU 483
USB 485
USB 496
버스 480
WiFi 482
블루투스 484
이더넷 인터페이스 481
파워 499
이더넷 497
DC 파워 498

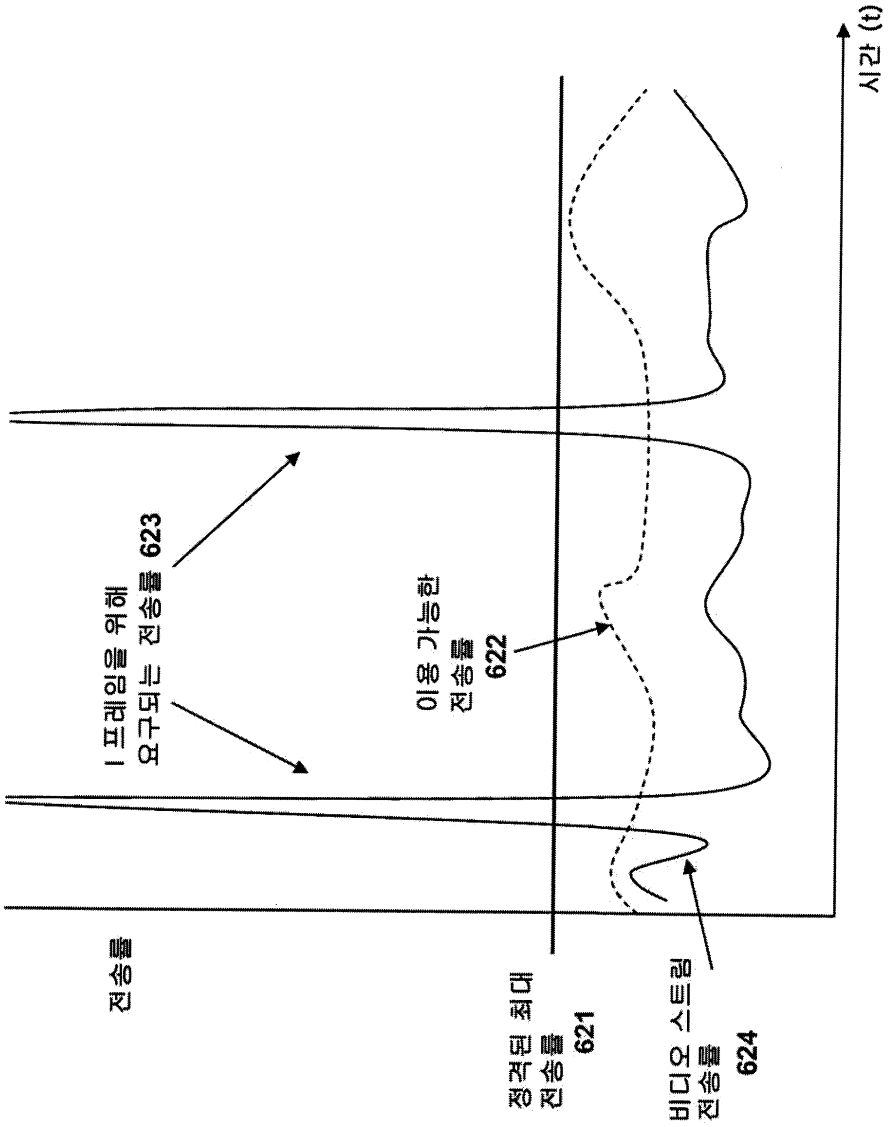
도면5


예, 900KB/프레임
압축되지 않은 프레임 503
압축되지 않은 프레임 502
압축되지 않은 프레임 501
압축 로직 520
압축된 프레임 513
압축된 프레임 512
압축된 프레임 511
예, 90킬로바이트/프레임

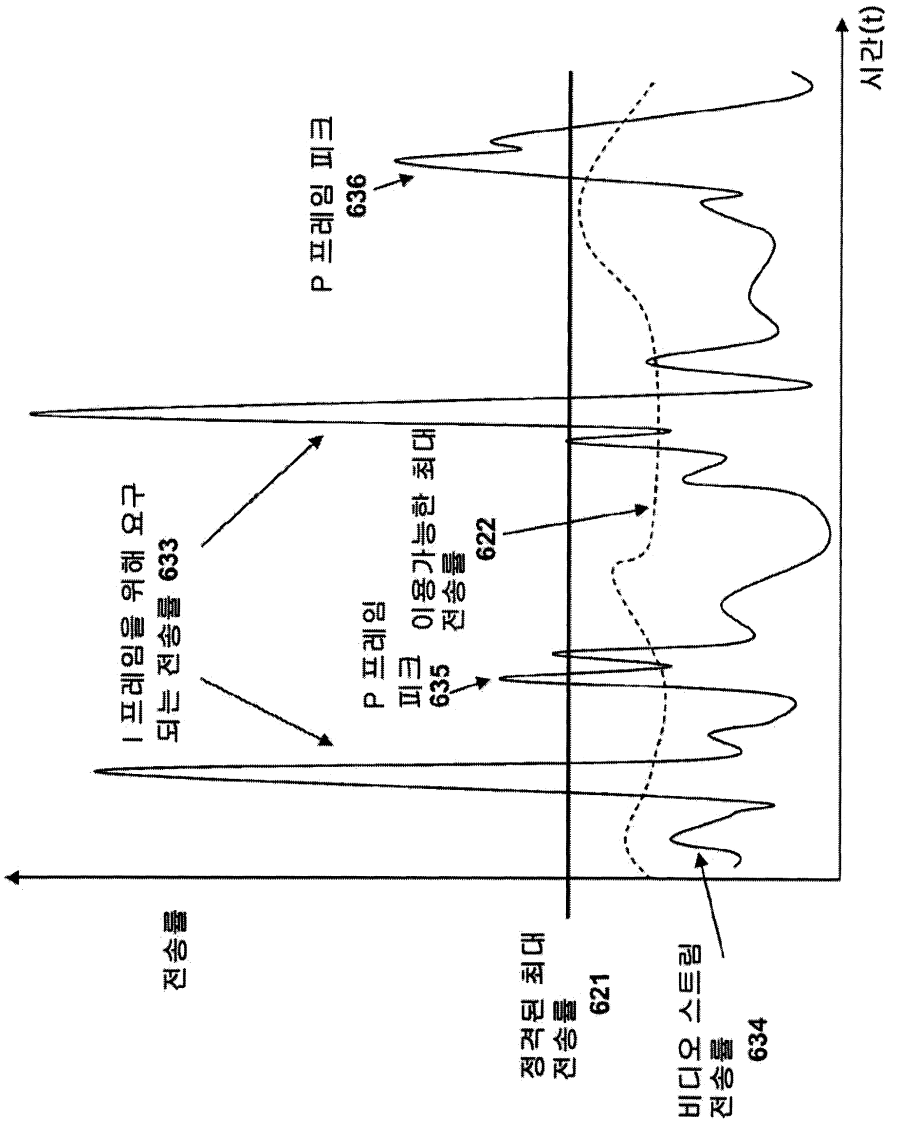
도면6a


예, 900KB/프레임
압축되지 않은 프레임 501
압축되지 않은 프레임 502
압축되지 않은 프레임 503
압축되지 않은 프레임 560
압축되지 않은 프레임 561
압축 로직 620
I 프레임 (~160 Kb) 611
P 프레임 (~16 Kb) 612
P 프레임 (~16 Kb) 613
B 프레임 (~16 Kb) 670
I 프레임 (~160 Kb) 671

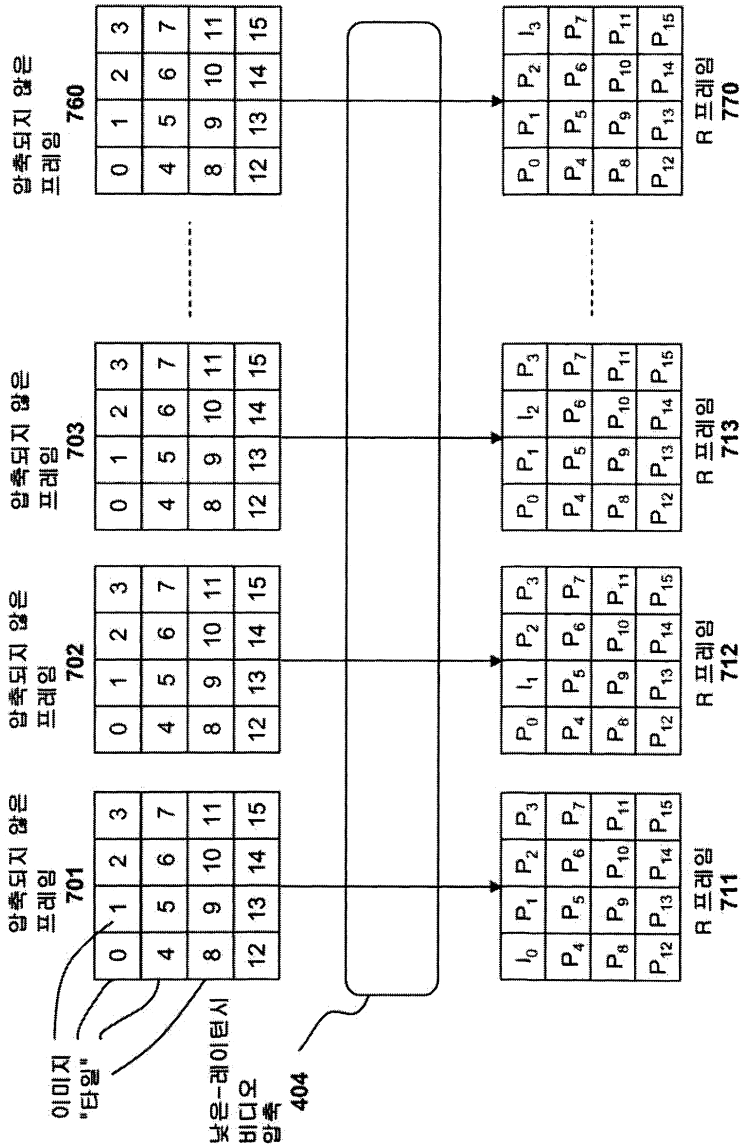
도면6b


시간 (t)
전송률
I 프레임을 위해
요구되는 전송률 623
이용 가능한
전송률
622
정적된 최대
전송률
621
비디오 스트림
전송률
624

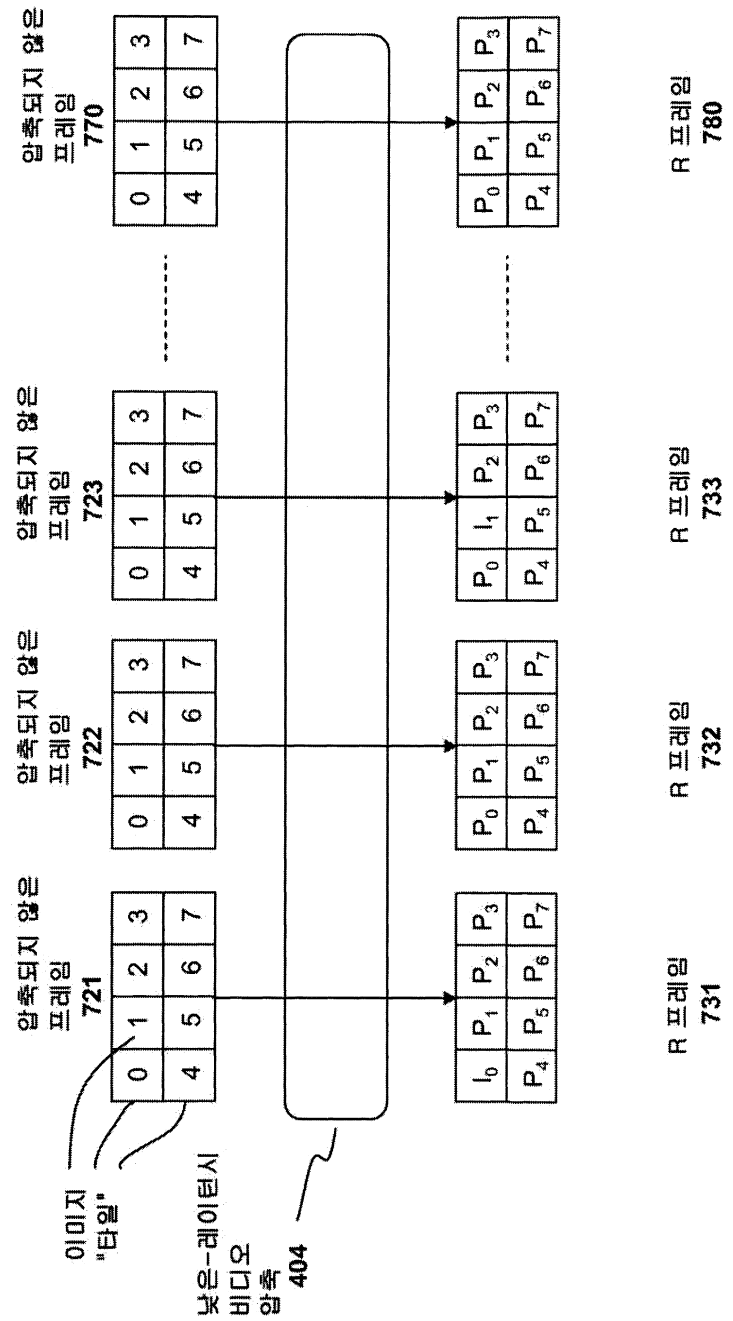
도면6c


시간(t)
P 프레임 피크
636
I 프레임을 위해 요구
되는 전송률 633
P 프레임
피크 635
이용가능한 최대
전송률
622
전송률
정적된 최대
전송률
621
비디오 스트림
전송률
634

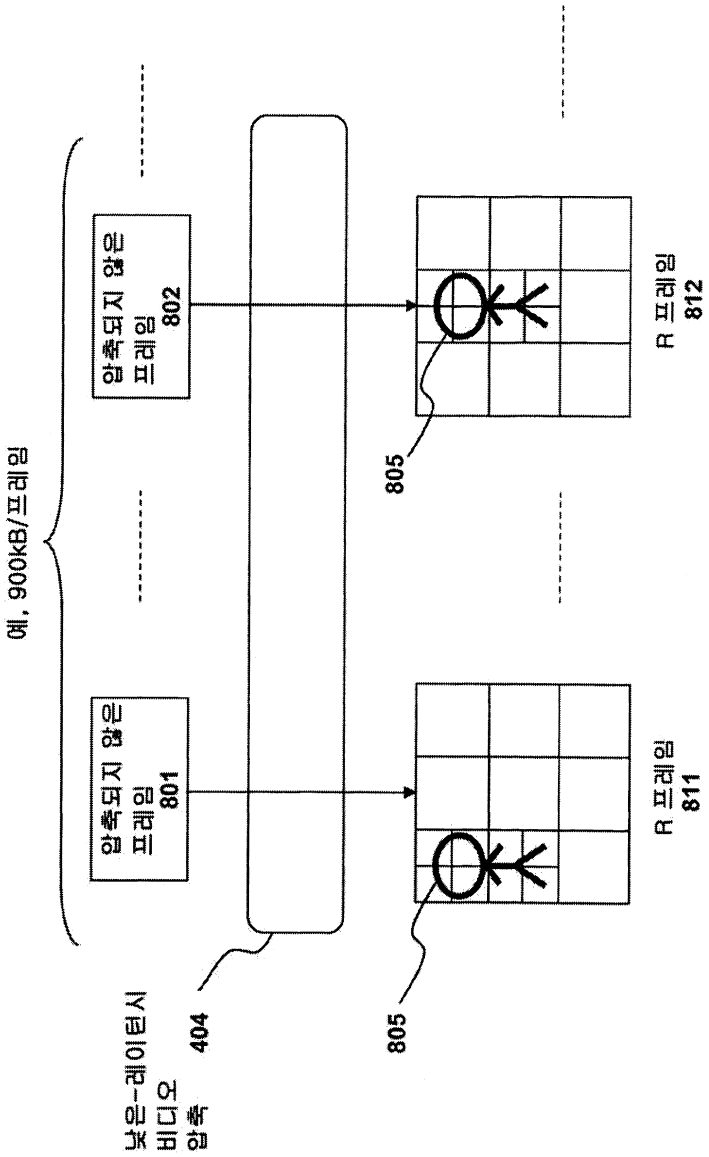
도면7a


압축되지 않은 프레임 760
압축되지 않은 프레임 703
압축되지 않은 프레임 702
압축되지 않은 프레임 701
R 프레임 770
R 프레임 713
R 프레임 712
R 프레임 711
이미지 "타일"
낮은-레이턴시 비디오 압축 404

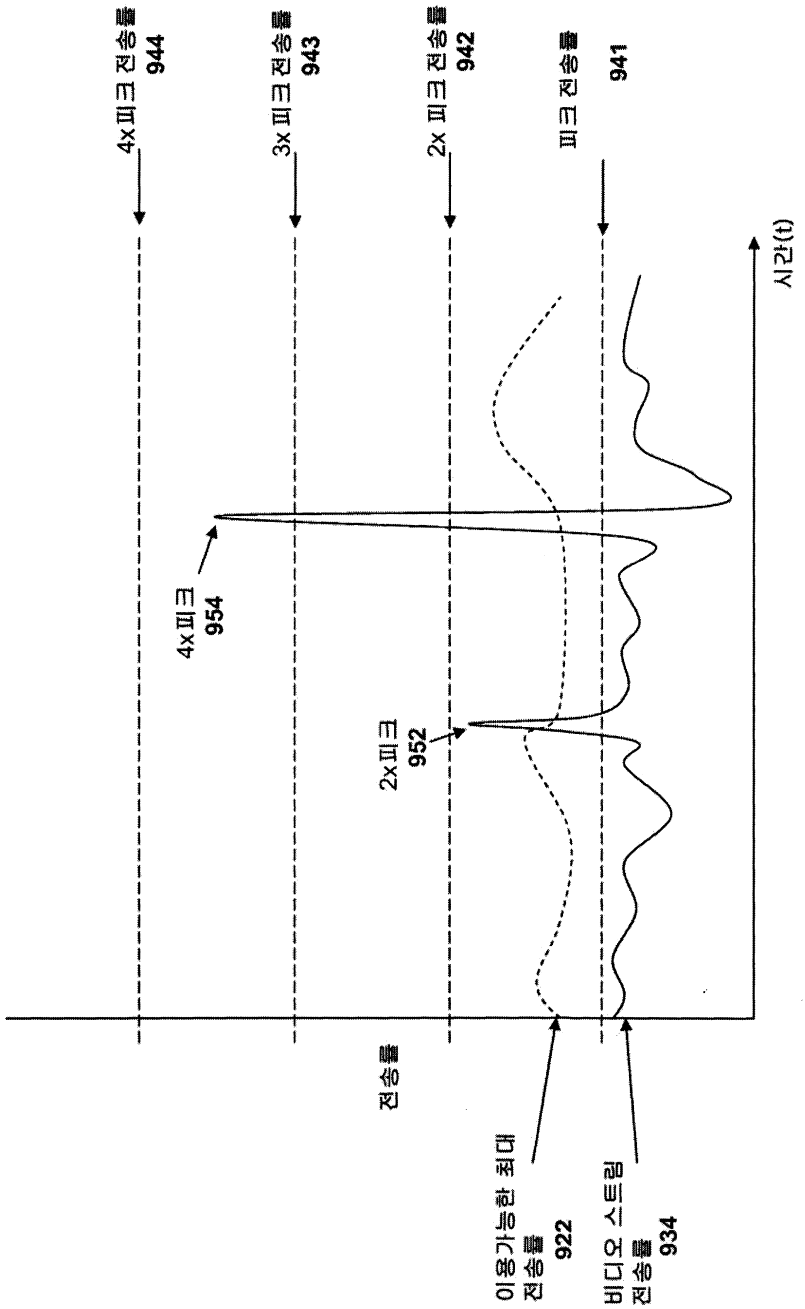
도면7b


압축되지 않은 프레임
721
압축되지 않은 프레임
722
압축되지 않은 프레임
723
압축되지 않은 프레임
770
R 프레임
731
R 프레임
732
R 프레임
733
R 프레임
780
이미지 "타일"
낮은-레이턴시 비디오 압축
404

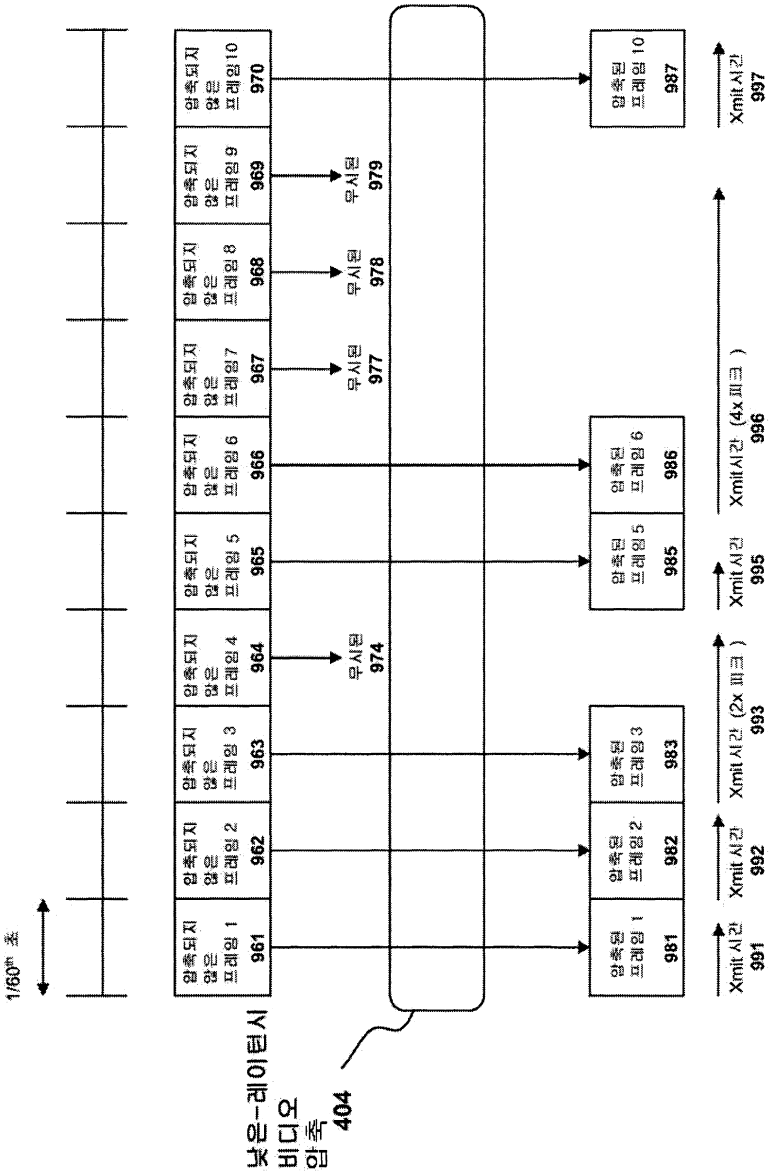
도면8


예, 900kB/프레임
압축되지 않은 프레임 802
압축되지 않은 프레임 801
805
805
R 프레임 812
R 프레임 811
낮은-레이턴시 비디오 압축
404

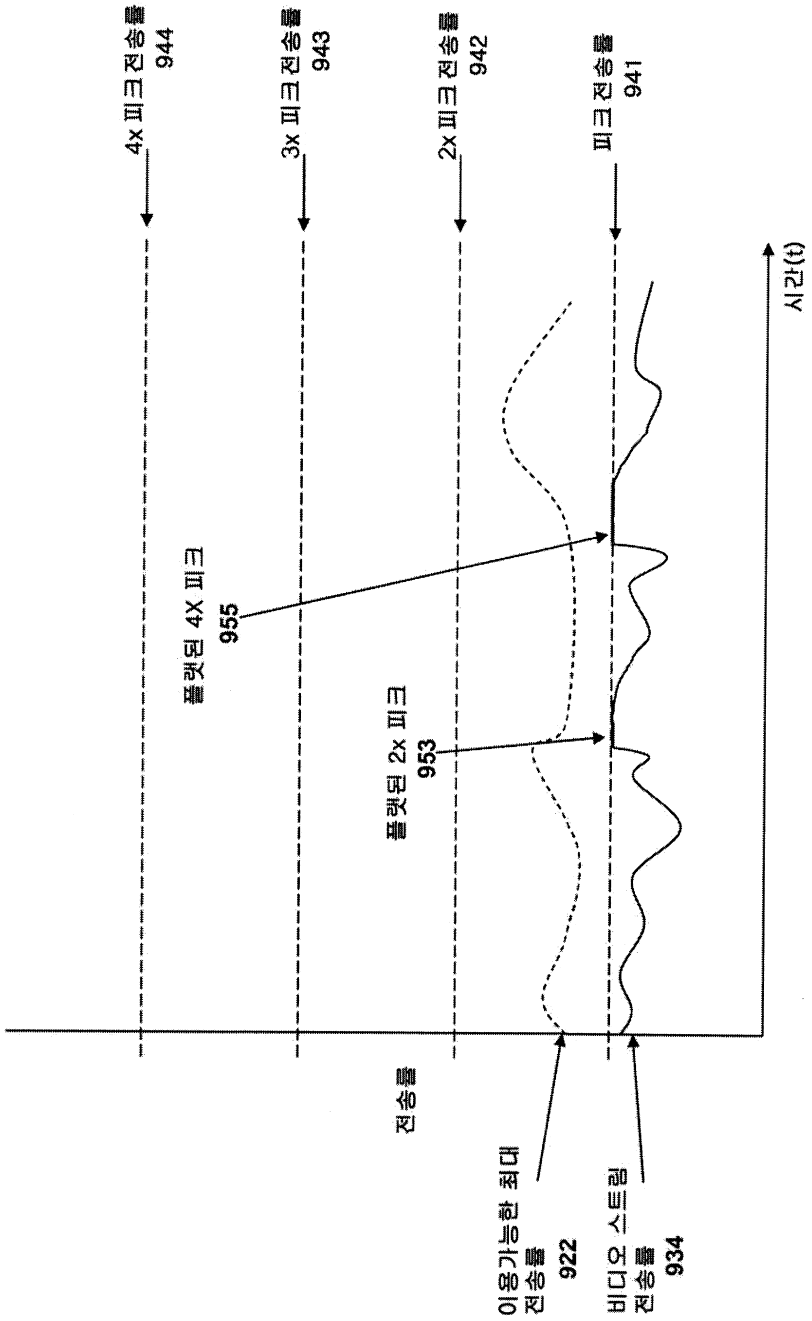
도면9a


4x 피크 전송률
944
3x 피크 전송률
943
2x 피크 전송률
942
피크 전송률
941
시간(t)
4x 피크
954
2x 피크
952
전송률
이용가능한 최대
전송률
922
비디오 스트림
전송률
934

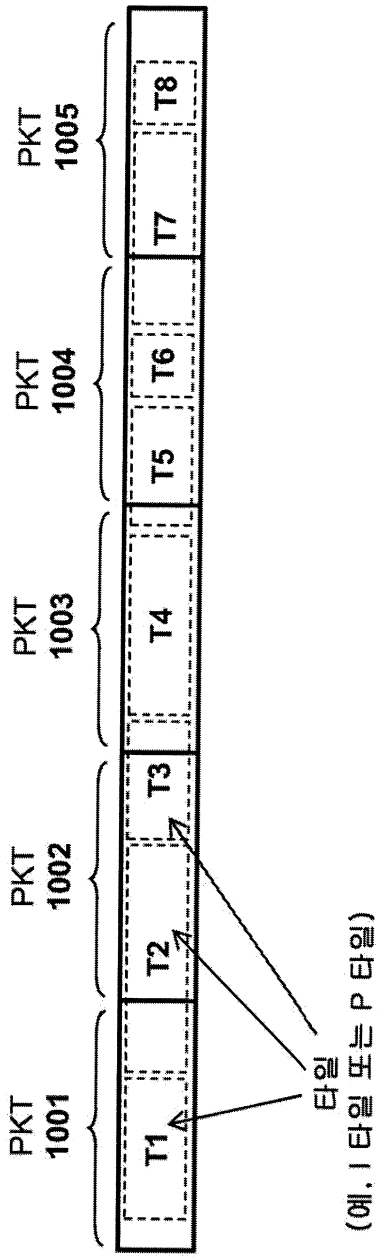
도면9b


1/60th 초
압축되지 않은 프레임 1 961
압축되지 않은 프레임 2 962
압축되지 않은 프레임 3 963
압축되지 않은 프레임 4 964
압축되지 않은 프레임 5 965
압축되지 않은 프레임 6 966
압축되지 않은 프레임 7 967
압축되지 않은 프레임 8 968
압축되지 않은 프레임 9 969
압축되지 않은 프레임10 970
무시됨 974
무시됨 977
무시됨 978
무시됨 979
낮은-레이턴시 비디오 압축 404
압축된 프레임 1 981
압축된 프레임 2 982
압축된 프레임 3 983
압축된 프레임 5 985
압축된 프레임 6 986
압축된 프레임 10 987
Xmit 시간 991
Xmit 시간 992
Xmit 시간 (2x 피크) 993
Xmit 시간 995
Xmit 시간 (4x 피크) 996
Xmit 시간 997

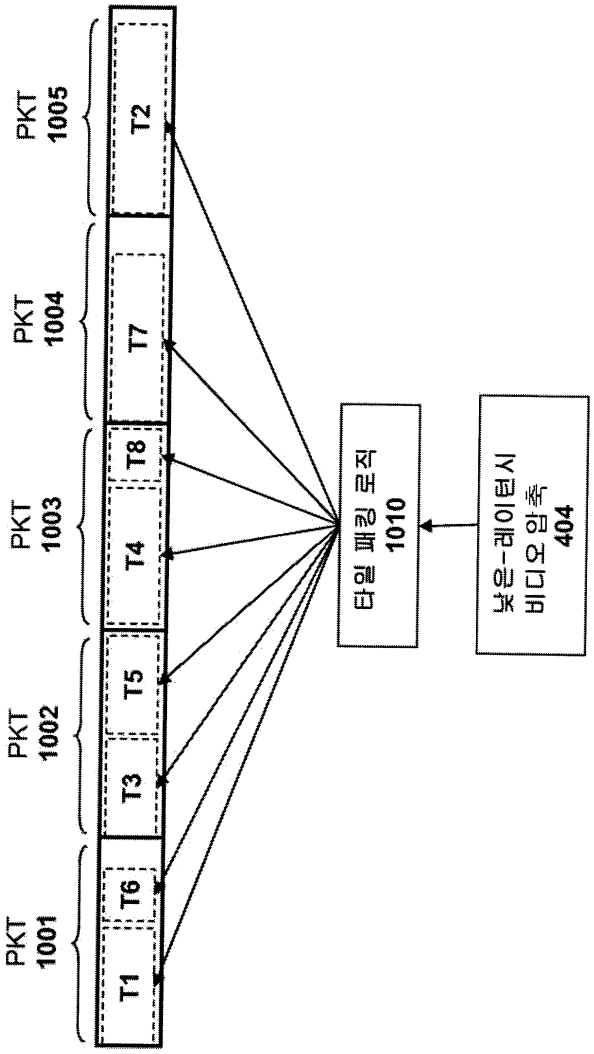
도면9c


4x 피크전송률
944
3x 피크전송률
943
2x 피크전송률
942
피크전송률
941
시간(t)
플랫된 4X 피크
955
플랫된 2x 피크
953
전송률
이용가능한 최대
전송률
922
비디오 스트림
전송률
934

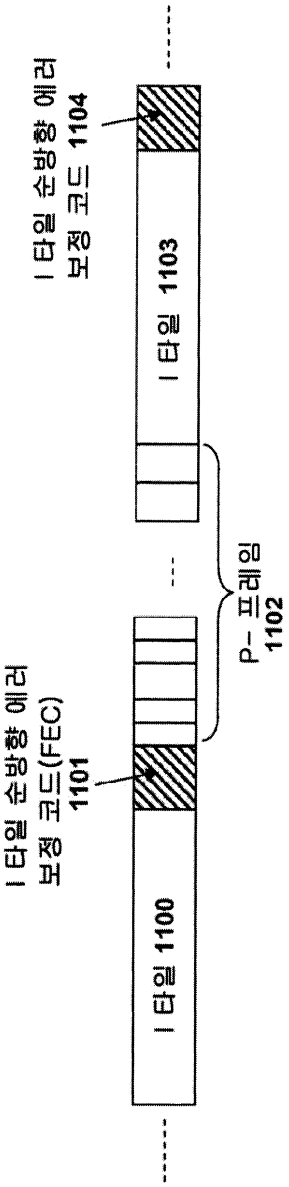
도면10a


PKT 1001
PKT 1002
PKT 1003
PKT 1004
PKT 1005
T1
T2
T3
T4
T5
T6
T7
T8
타일
(예, I 타일 또는 P 타일)

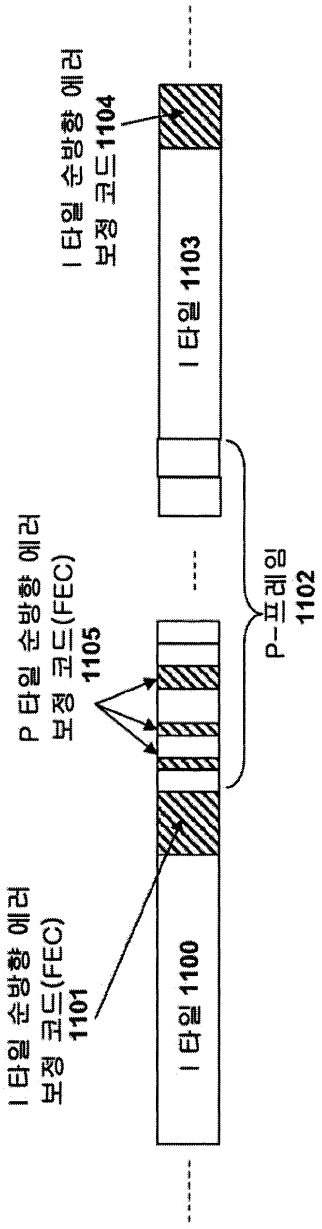
도면10b


PKT 1001
PKT 1002
PKT 1003
PKT 1004
PKT 1005
T1
T6
T3
T5
T4
T8
T7
T2
타일 패킹 로직 1010
낮은-레이턴시 비디오 압축 404

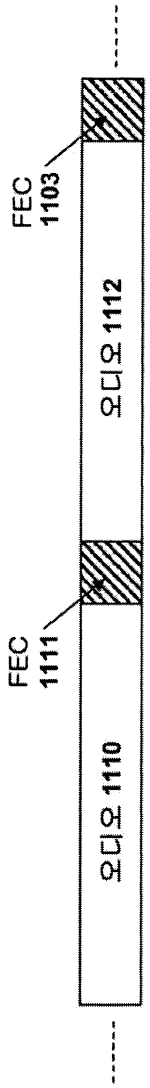
도면11a


I 타일 순방향 에러
보정 코드 1104
I 타일 1103
P- 프레임
1102
I 타일 순방향 에러
보정 코드(FEC)
1101
I 타일 1100

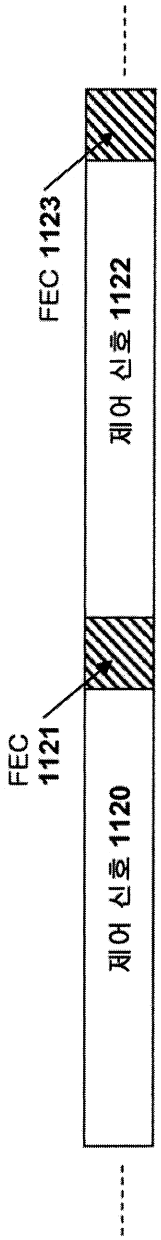
도면11b


I 타일 순방향 에러
보정 코드1104
I 타일 1103
P 타일 순방향 에러
보정 코드(FEC)
1105
P-프레임
1102
I 타일 순방향 에러
보정 코드(FEC)
1101
I 타일 1100

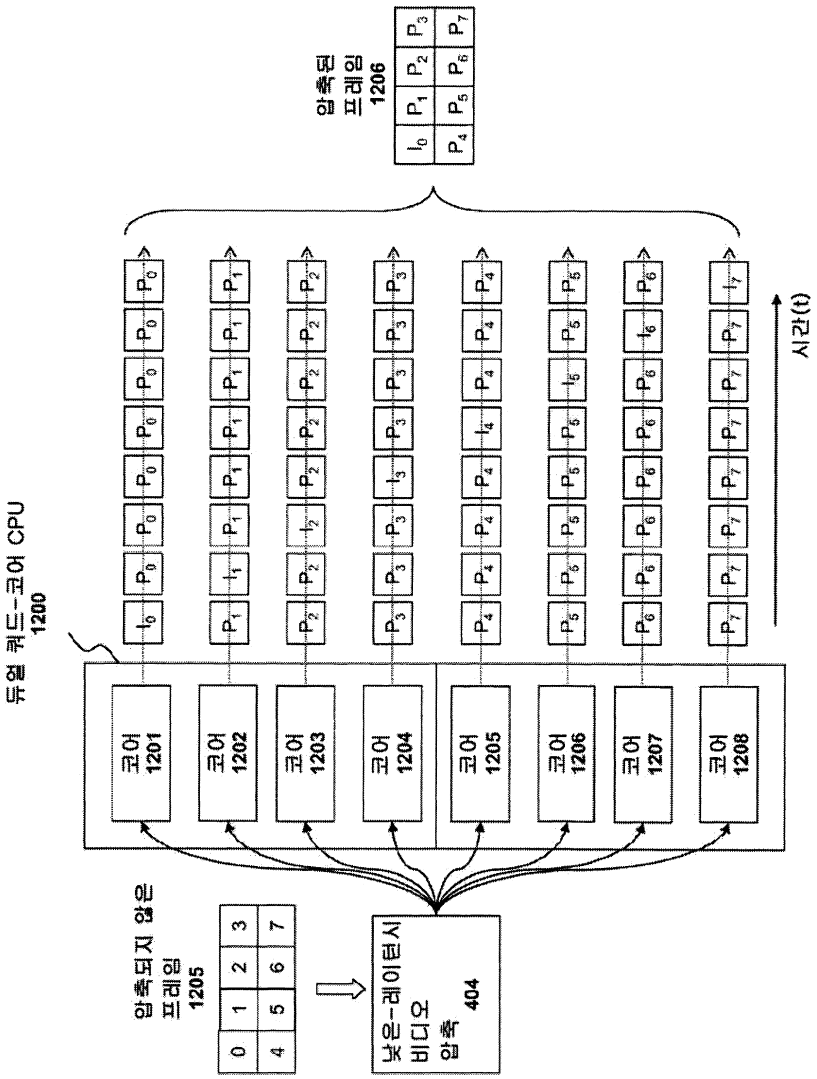
도면11c



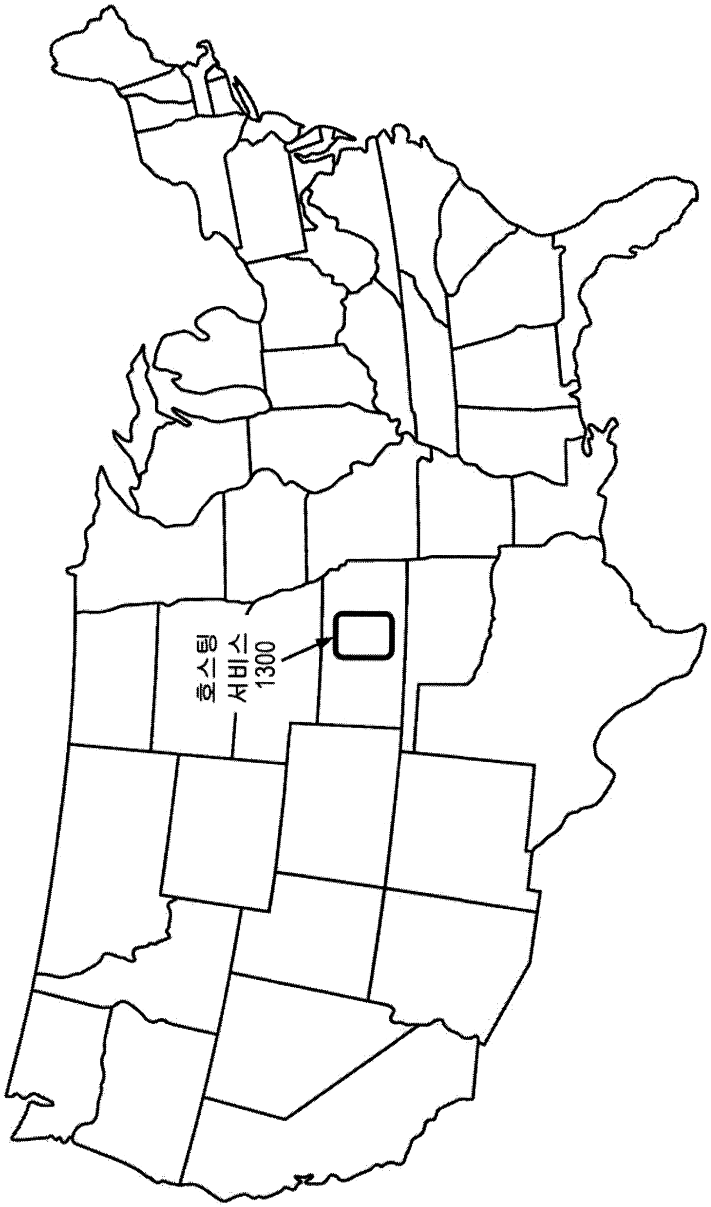
도면11d



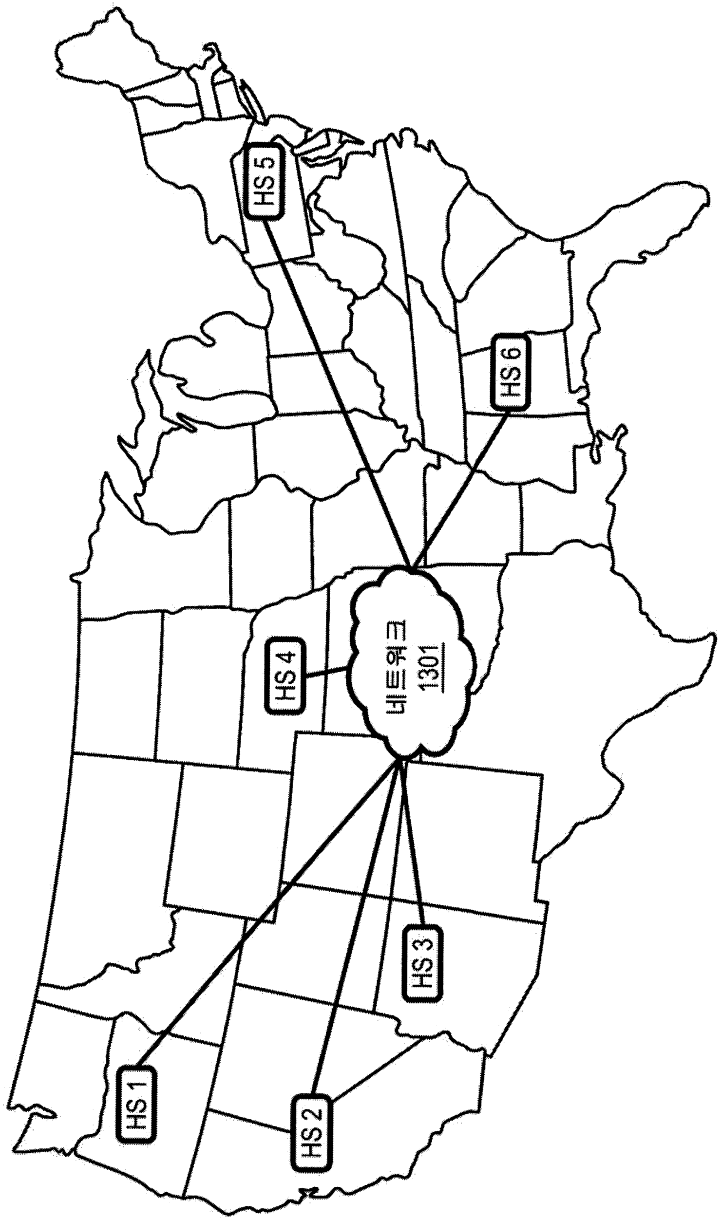
도면12


듀얼 쿼드-코어 CPU
1200
코어
1201
코어
1202
코어
1203
코어
1204
코어
1205
코어
1206
코어
1207
코어
1208
시간(t)
압축된
프레임
1206
압축되지 않은
프레임
1205
낮은-레이턴시
비디오
압축
404

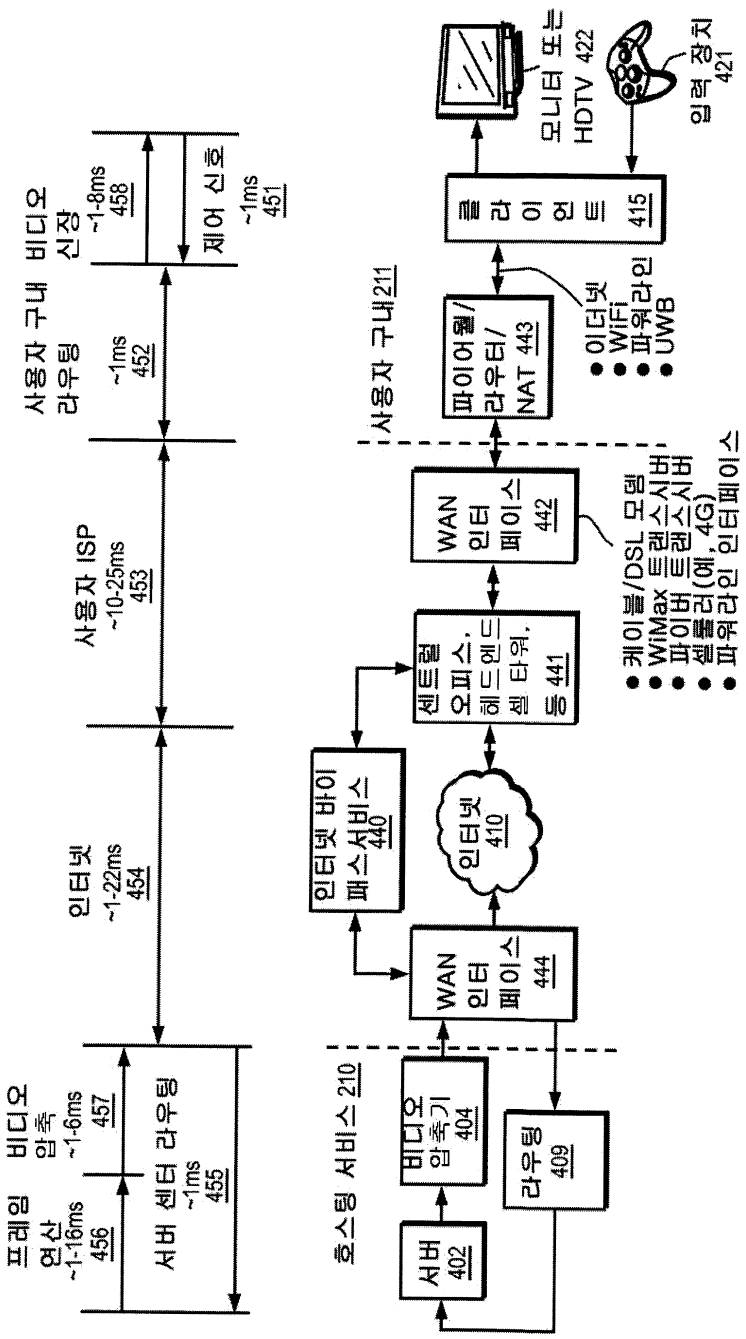
도면13a



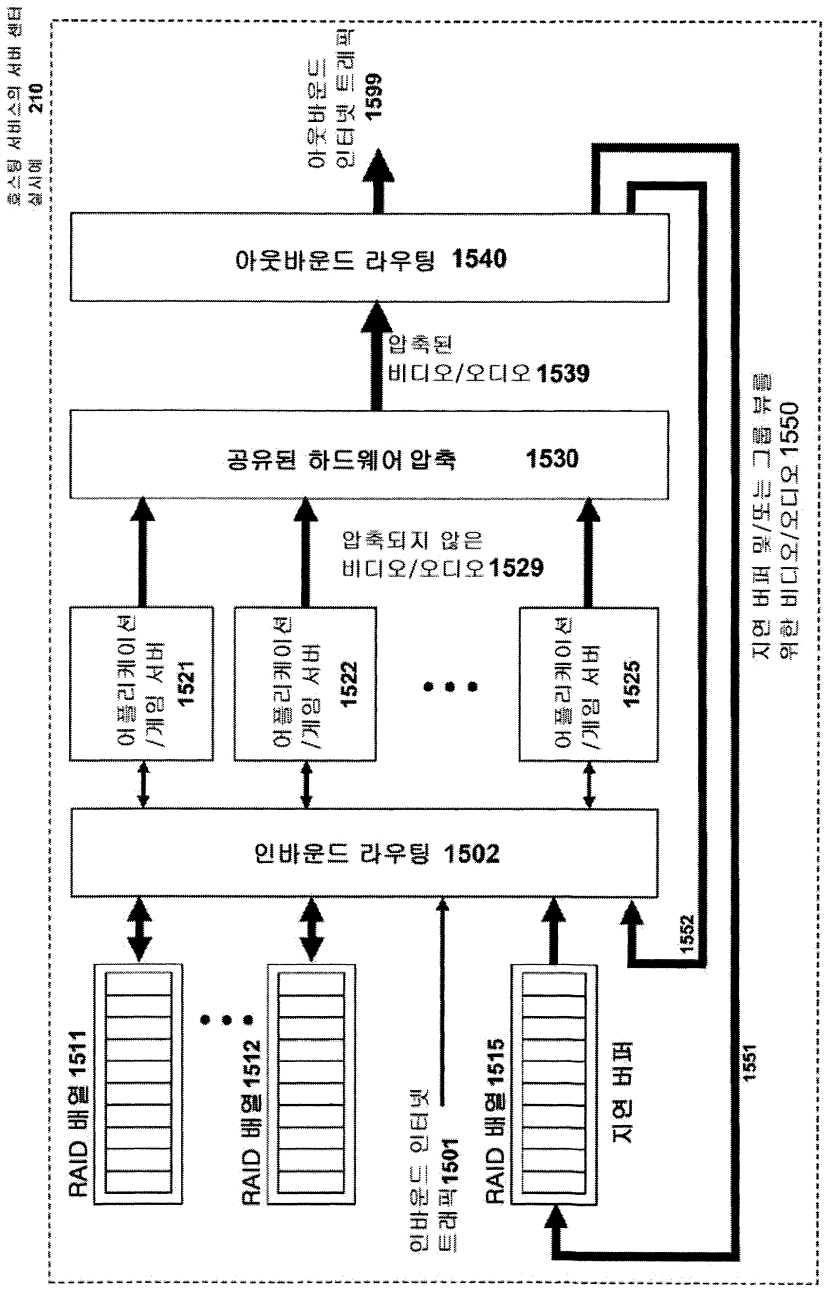
도면13b



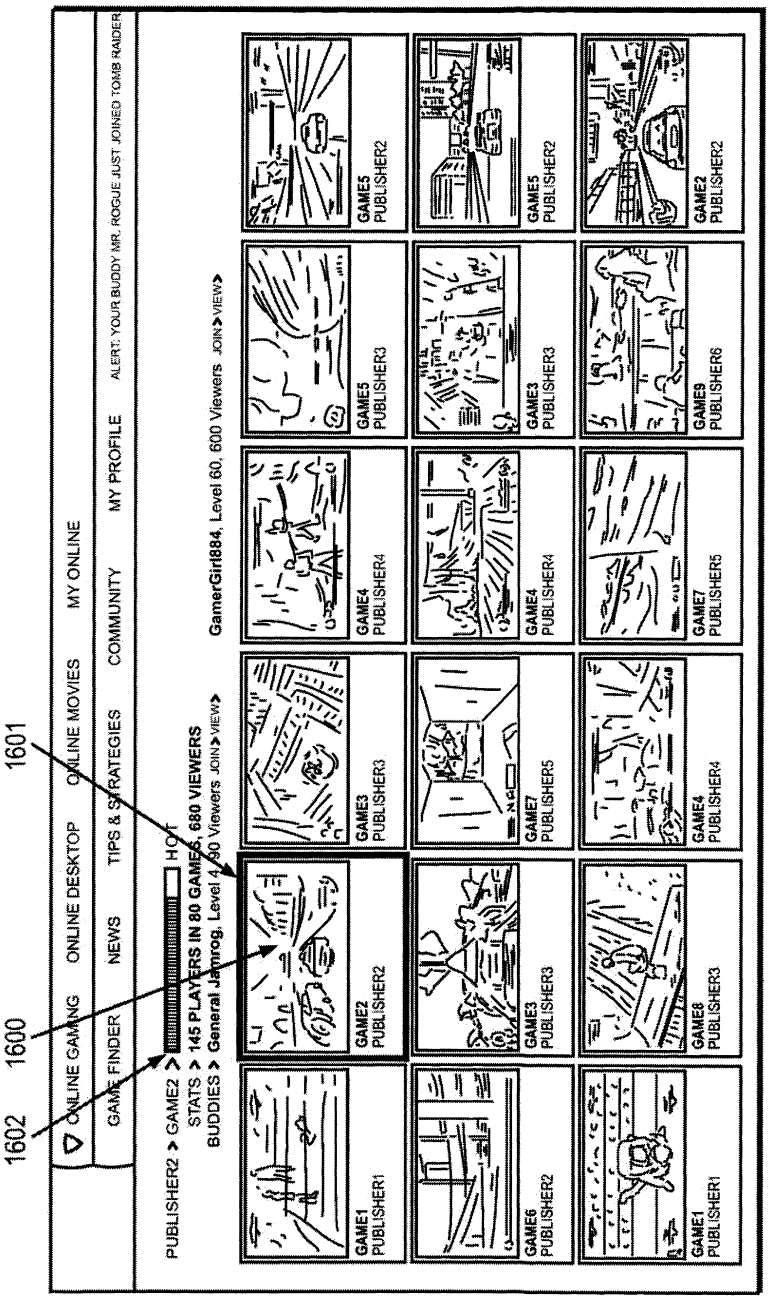
도면14


프레임 연산 ~1-16ms 456
비디오 압축 ~1-6ms 457
서버 센터 라우팅 ~1ms 455
인터넷 ~1-22ms 454
사용자 ISP ~10-25ms 453
사용자 구내 라우팅 ~1ms 452
사용자 구내 비디오 신장 ~1-8ms 458
제어 신호 ~1ms 451
호스팅 서비스 210
서버 402
비디오 압축기 404
라우팅 409
WAN 인터페이스 444
인터넷 410
인터넷 백본 패스서비스 440
센트럴 오피스, 헤드엔드 셀 타워, 등 441
WAN 인터페이스 442
사용자 구내 211
케이블/DSL 모뎀
WiMax 트랜스시버
파이버 트랜스시버
셀룰러(예, 4G) 인터페이스
파워라인
파이어월/라우터/NAT 443
인터넷
WiFi
파워라인
UWB
클라이언트 415
모니터 또는 HDTV 422
입력 장치 421

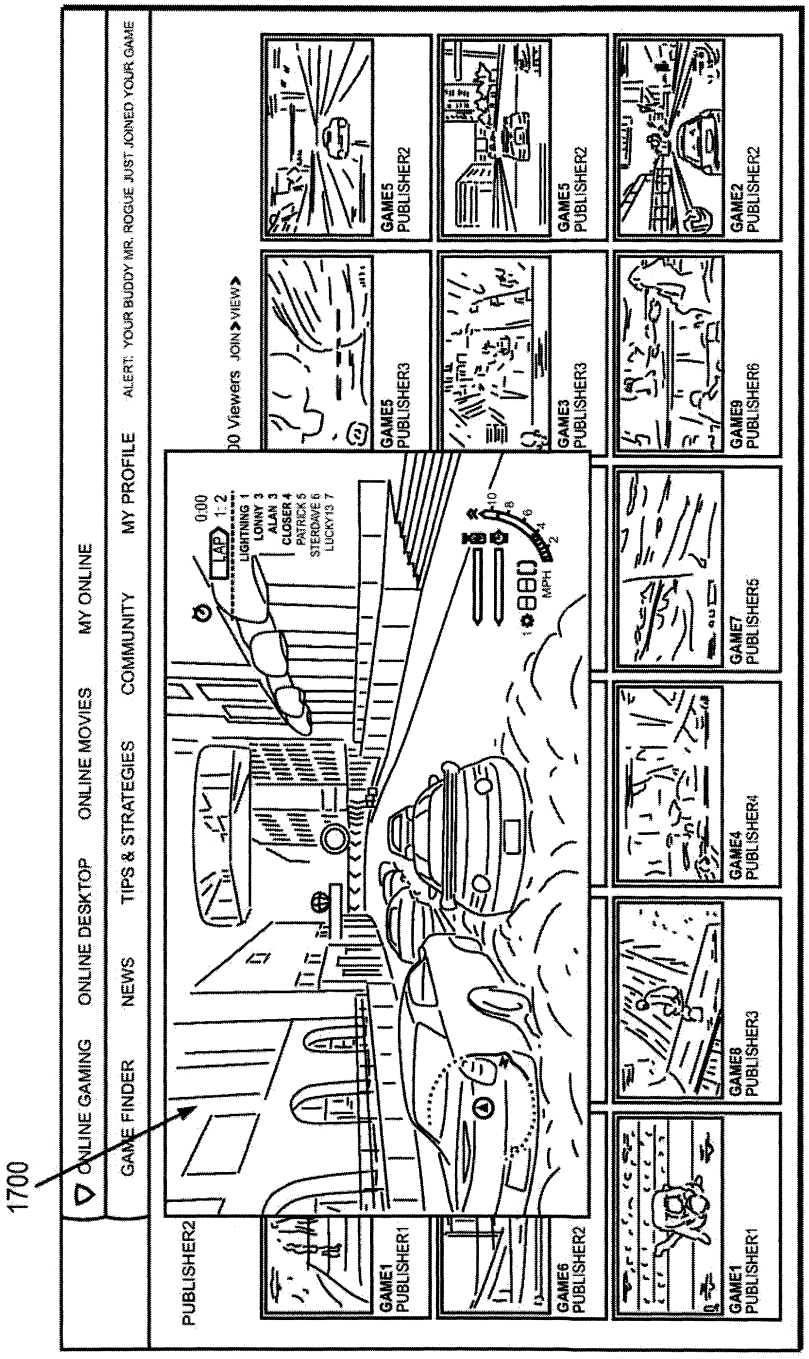
도면15


호스팅 서비스의 서버 센터 실시예 210
아웃바운드 인터넷 트래픽 1599
아웃바운드 라우팅 1540
압축된 비디오/오디오 1539
공유된 하드웨어 압축 1530
압축되지 않은 비디오/오디오 1529
어플리케이션 /게임 서버 1521
어플리케이션 /게임 서버 1522
어플리케이션 /게임 서버 1525
인바운드 라우팅 1502
RAID 배열 1511
RAID 배열 1512
RAID 배열 1515
지연 버퍼
인바운드 인터넷 트래픽 1501
지연 버퍼 및/또는 그룹 뷰를 위한 비디오/오디오 1550
1551
1552

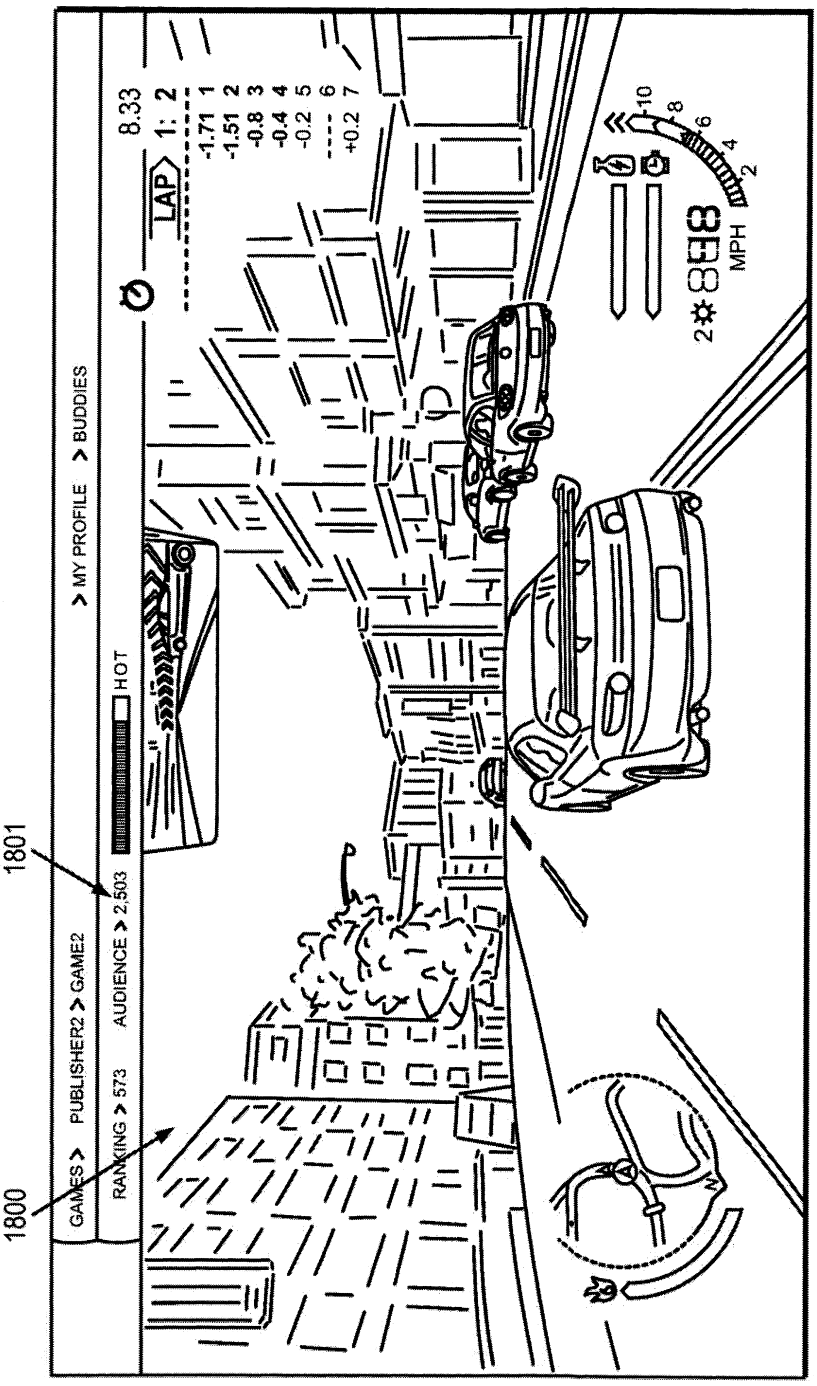
도면16


1602
1600
1601
ONLINE GAMING
ONLINE DESKTOP
ONLINE MOVIES
MY ONLINE
GAME FINDER
NEWS
TIPS & STRATEGIES
COMMUNITY
MY PROFILE
ALERT: YOUR BUDDY MR. ROGUE JUST JOINED TOMB RAIDER
PUBLISHER2 > GAME2 >
HOT
STATS > 145 PLAYERS IN 80 GAMES, 680 VIEWERS
BUDDIES > General Jamrog, Level 4, 60 Viewers JOIN > VIEW >
GamerGirl884, Level 60, 600 Viewers JOIN > VIEW >
GAME1 PUBLISHER1
GAME2 PUBLISHER2
GAME3 PUBLISHER3
GAME4 PUBLISHER4
GAME5 PUBLISHER3
GAME5 PUBLISHER2
GAME6 PUBLISHER2
GAME3 PUBLISHER3
GAME7 PUBLISHER5
GAME4 PUBLISHER4
GAME3 PUBLISHER3
GAME5 PUBLISHER2
GAME1 PUBLISHER1
GAME8 PUBLISHER3
GAME4 PUBLISHER4
GAME7 PUBLISHER5
GAME9 PUBLISHER6
GAME2 PUBLISHER2

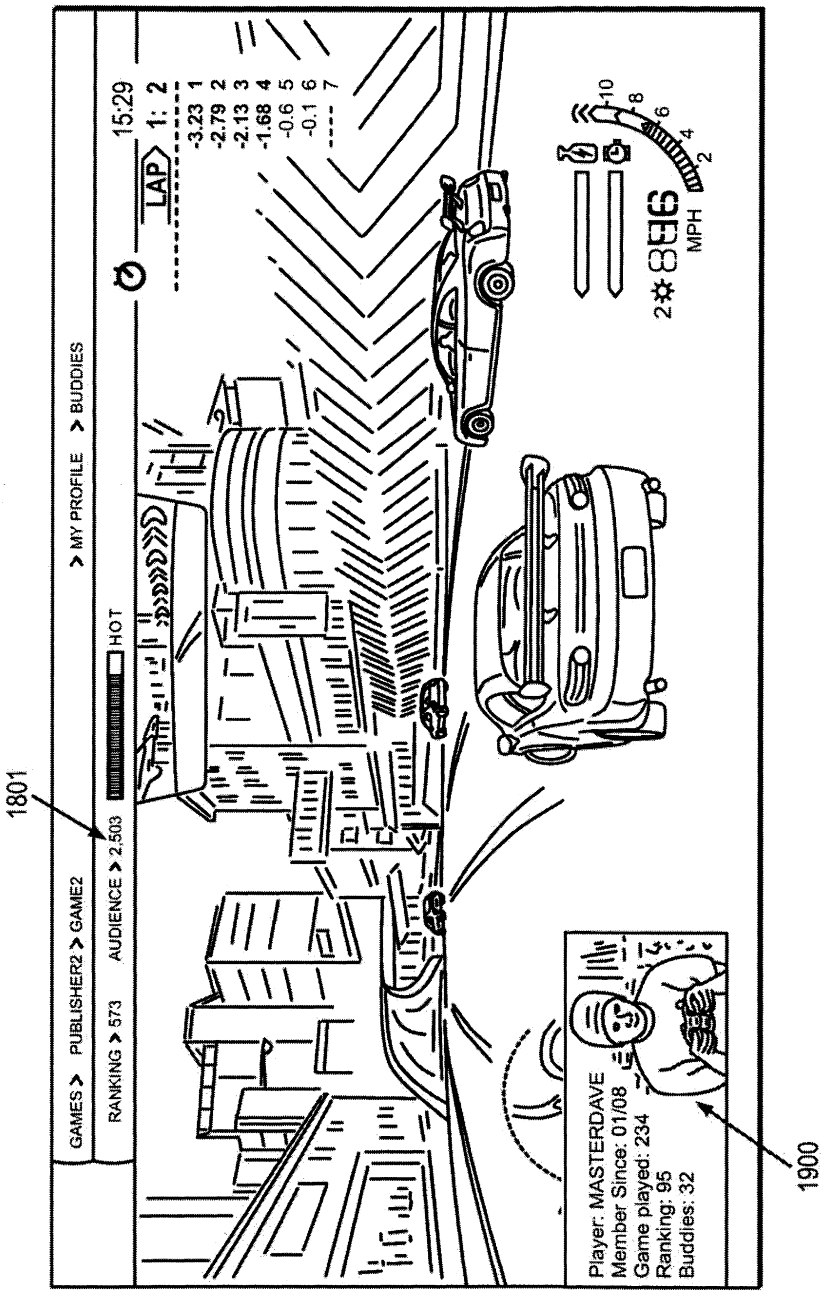
도면17


1700
ONLINE GAMING
ONLINE DESKTOP
ONLINE MOVIES
MY ONLINE
GAME FINDER
NEWS
TIPS & STRATEGIES
COMMUNITY
MY PROFILE
ALERT: YOUR BUDDY MR. ROGUE JUST JOINED YOUR GAME
PUBLISHER2
LAP
LIGHTNING 1
LONNY 3
ALAN 3
CLOSER 4
PATRICK 5
STERDAVE 6
LUCKY13 7
MPH
GAME1 PUBLISHER1
GAME8 PUBLISHER3
GAME4 PUBLISHER4
GAME7 PUBLISHER6
GAME6 PUBLISHER2
GAME5 PUBLISHER3
GAME3 PUBLISHER3
GAME9 PUBLISHER6
GAME5 PUBLISHER2
GAME5 PUBLISHER2
GAME2 PUBLISHER2

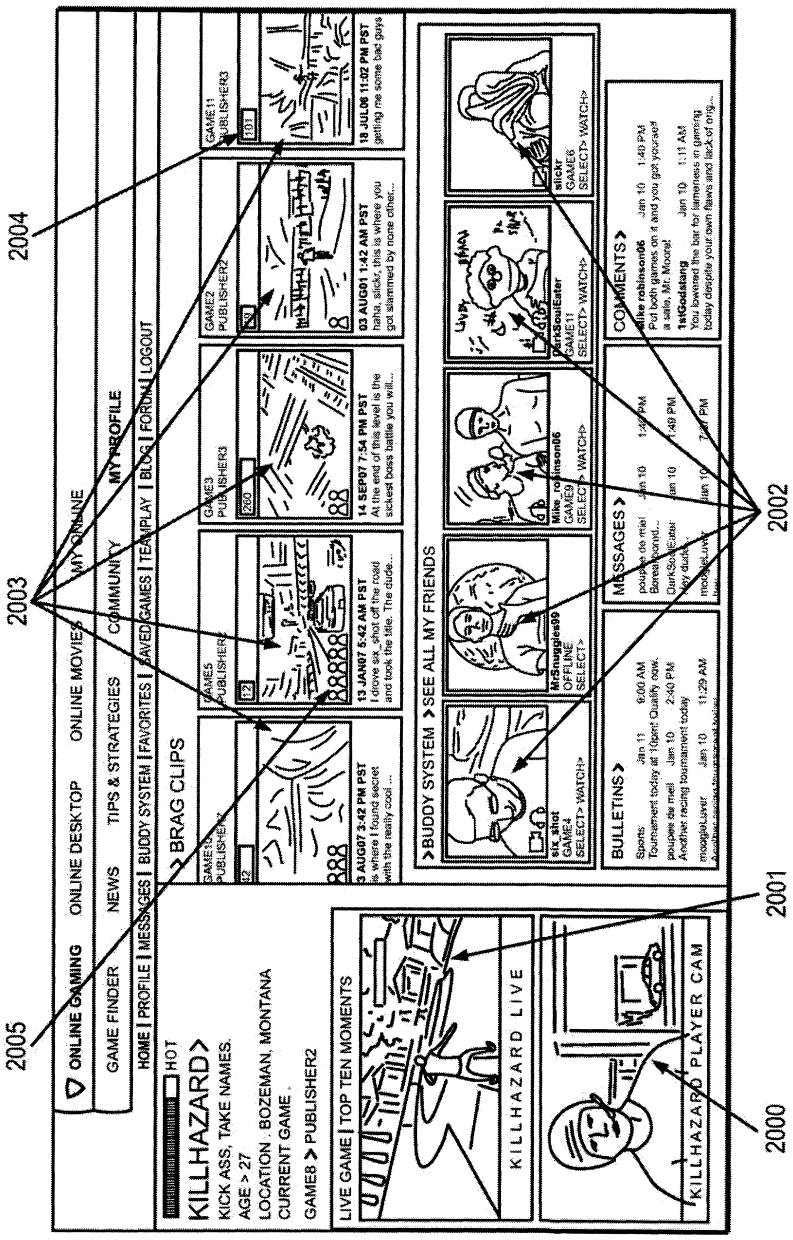
도면18


1801
1800
GAMES > PUBLISHER2 > GAME2
RANKING > 573
AUDIENCE > 2,503
HOT
> MY PROFILE
> BUDDIES
LAP 1: 2
8.33
-1.71 1
-1.51 2
-0.8 3
-0.4 4
-0.2 5
---- 6
+0.2 7
MPH

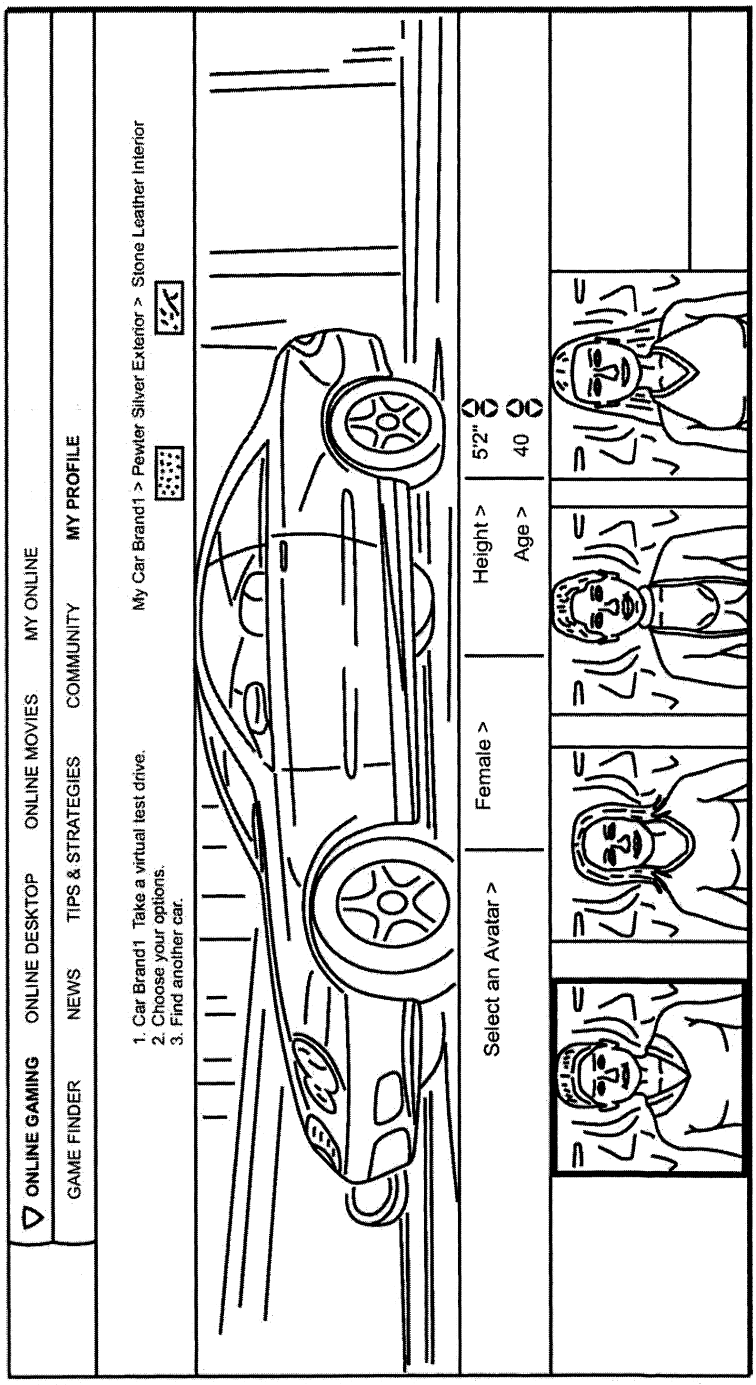
도면19


1801
GAMES > PUBLISHER2 > GAME2
> MY PROFILE > BUDDIES
RANKING > 573
AUDIENCE > 2,503
HOT
LAP 1: 2
15:29
-3.23 1
-2.79 2
-2.13 3
-1.68 4
-0.6 5
-0.1 6
7
MPH
1900
Player: MASTERDAVE
Member Since: 01/08
Game played: 234
Ranking: 95
Buddies: 32

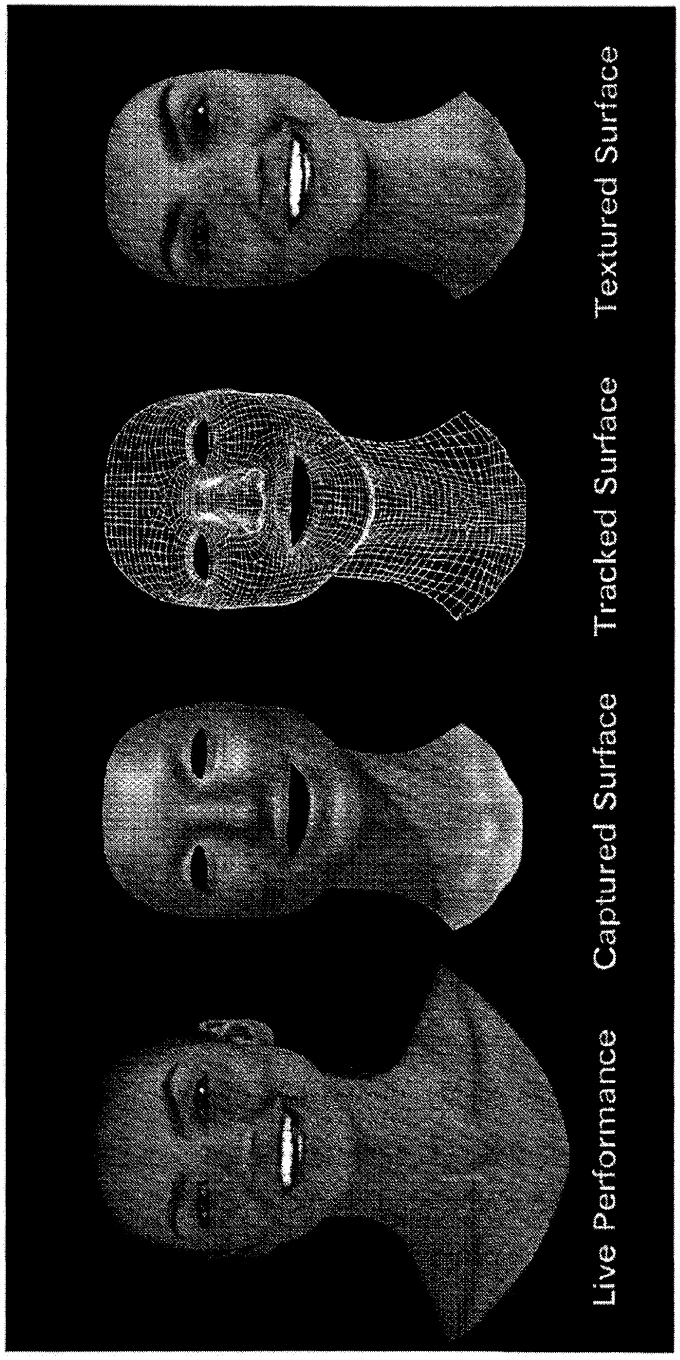
도면20


ONLINE GAMING
ONLINE DESKTOP
ONLINE MOVIES
MY ONLINE
GAME FINDER
NEWS
TIPS & STRATEGIES
COMMUNITY
MY PROFILE
HOME | PROFILE | MESSAGES | BUDDY SYSTEM | FAVORITES | SAVED GAMES | TEAMPLAY | BLOG | FORUM | LOGOUT
HOT
KILLHAZARD>
KICK ASS, TAKE NAMES.
AGE > 27
LOCATION . BOZEMAN, MONTANA
CURRENT GAME .
GAME8 > PUBLISHER2
>BRAG CLIPS
LIVE GAME | TOP TEN MOMENTS
KILLHAZARD LIVE
KILLHAZARD PLAYER CAM
>BUDDY SYSTEM >SEE ALL MY FRIENDS
BULLETINS >
MESSAGES >
COMMENTS >
2000
2001
2002
2003
2004
2005

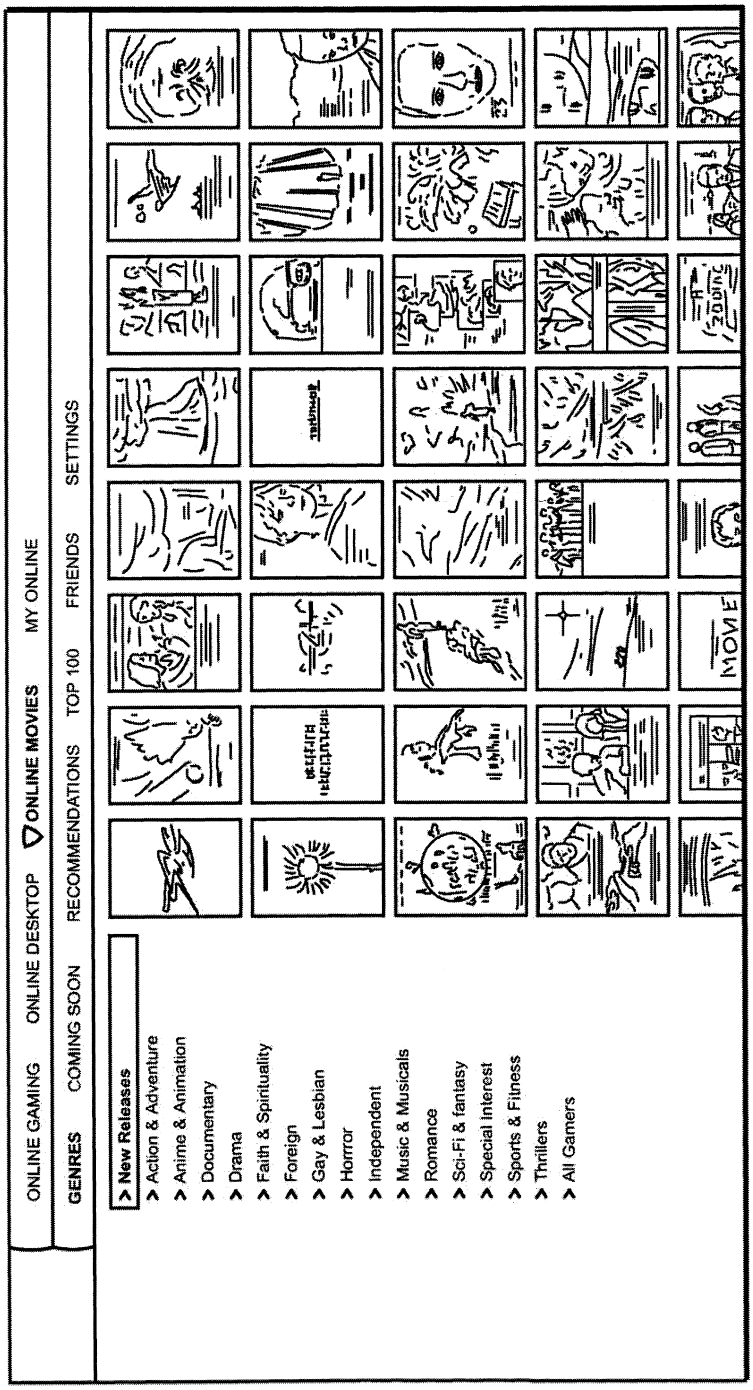
도면21


ONLINE GAMING
ONLINE DESKTOP
ONLINE MOVIES
MY ONLINE
GAME FINDER
NEWS
TIPS & STRATEGIES
COMMUNITY
MY PROFILE
1. Car Brand1 Take a virtual test drive.
2. Choose your options.
3. Find another car.
My Car Brand1 > Pewter Silver Exterior > Stone Leather Interior
Select an Avatar >
Female >
Height >
5'2"
Age >
40

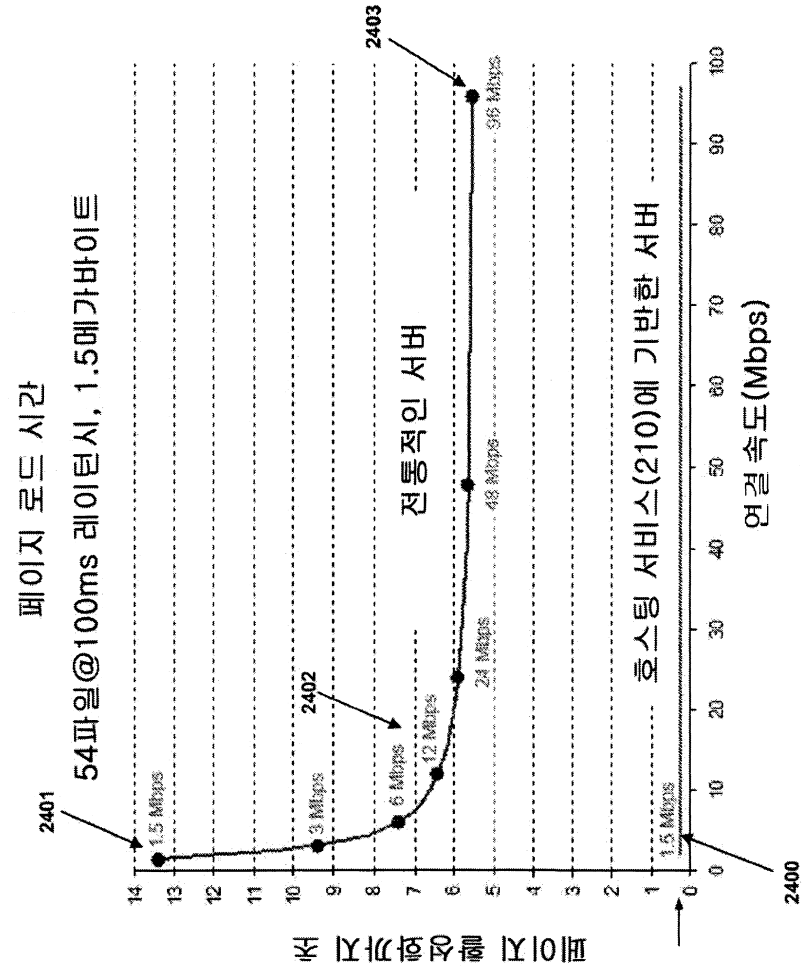
도면22


Live Performance
Captured Surface
Tracked Surface
Textured Surface

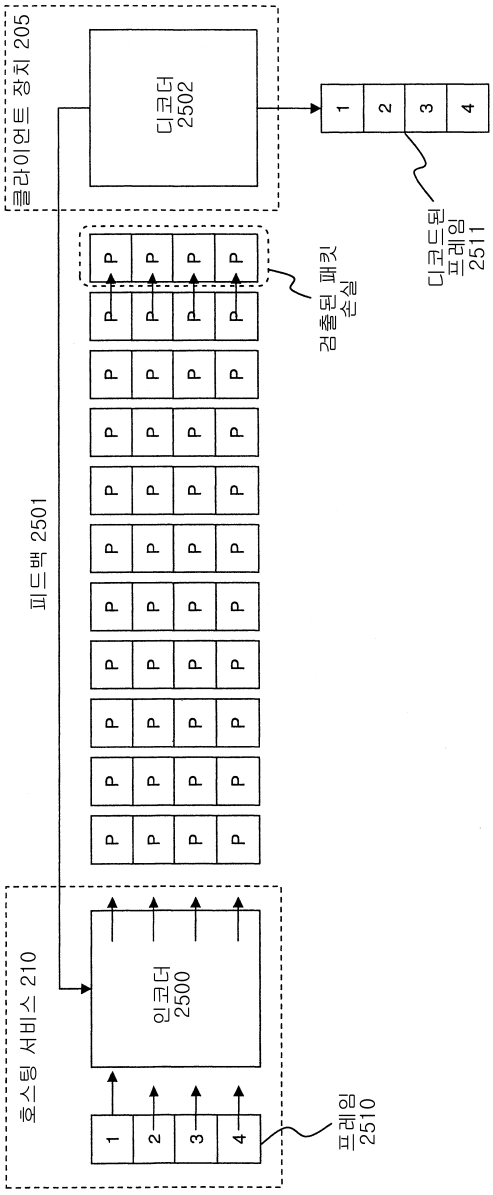
도면23


ONLINE GAMING
ONLINE DESKTOP
ONLINE MOVIES
MY ONLINE
GENRES
COMING SOON
RECOMMENDATIONS
TOP 100
FRIENDS
SETTINGS
> New Releases
> Action & Adventure
> Anime & Animation
> Documentary
> Drama
> Faith & Spirituality
> Foreign
> Gay & Lesbian
> Horrror
> Independent
> Music & Musicals
> Romance
> Sci-Fi & fantasy
> Special interest
> Sports & Fitness
> Thrillers
> All Gamers

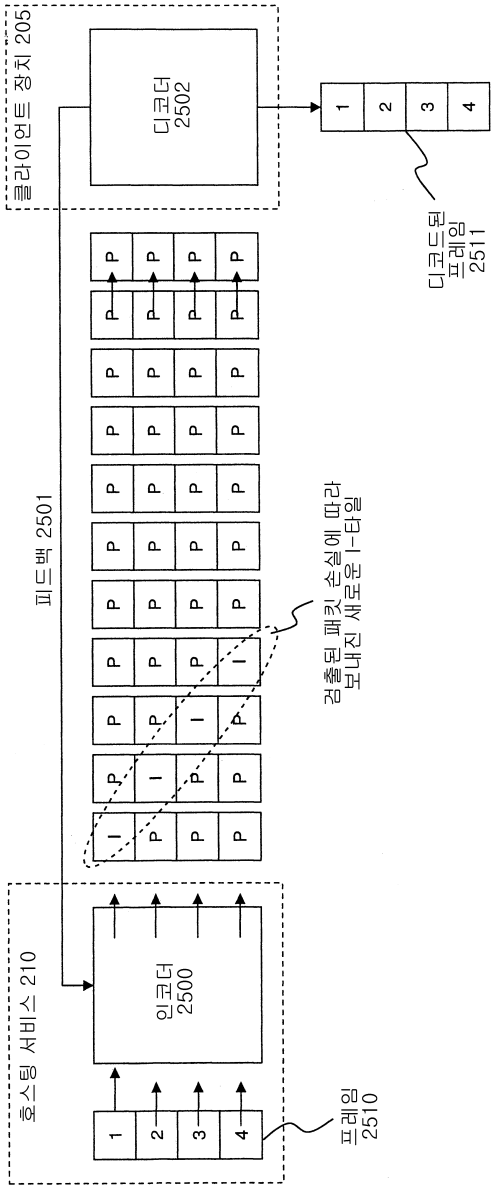
도면24


페이지 로드 시간
54파일@100ms 레이턴시, 1.5메가바이트
2401
2402
2403
2400
1.5 Mbps
3 Mbps
6 Mbps
12 Mbps
24 Mbps
48 Mbps
96 Mbps
전통적인 서버
호스팅 서비스(210)에 기반한 서버
페이지 활성화까지 초
연결 속도(Mbps)

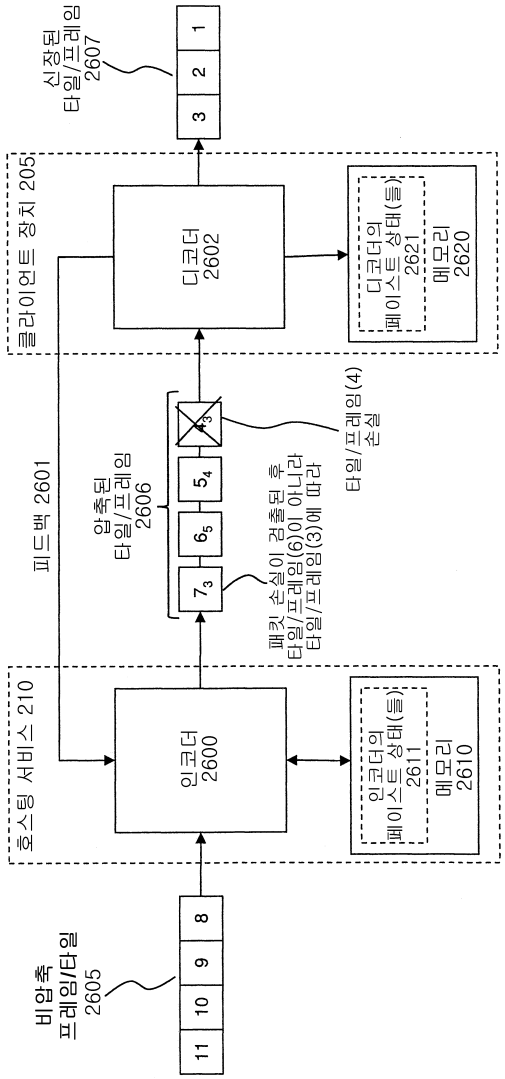
도면25a


클라이언트 장치 205
디코더
2502
1
2
3
4
디코드된
프레임
2511
검출된 패킷
손실
P
피드백 2501
호스팅 서비스 210
인코더
2500
1
2
3
4
프레임
2510

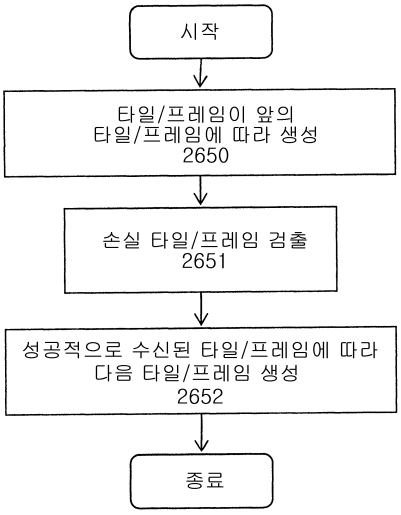
도면25b


클라이언트 장치 205
디코더
2502
1
2
3
4
디코드된
프레임
2511
P
I
피드백 2501
검출된 패킷 손실에 따라
보내진 새로운 I-타일
호스팅 서비스 210
인코더
2500
1
2
3
4
프레임
2510

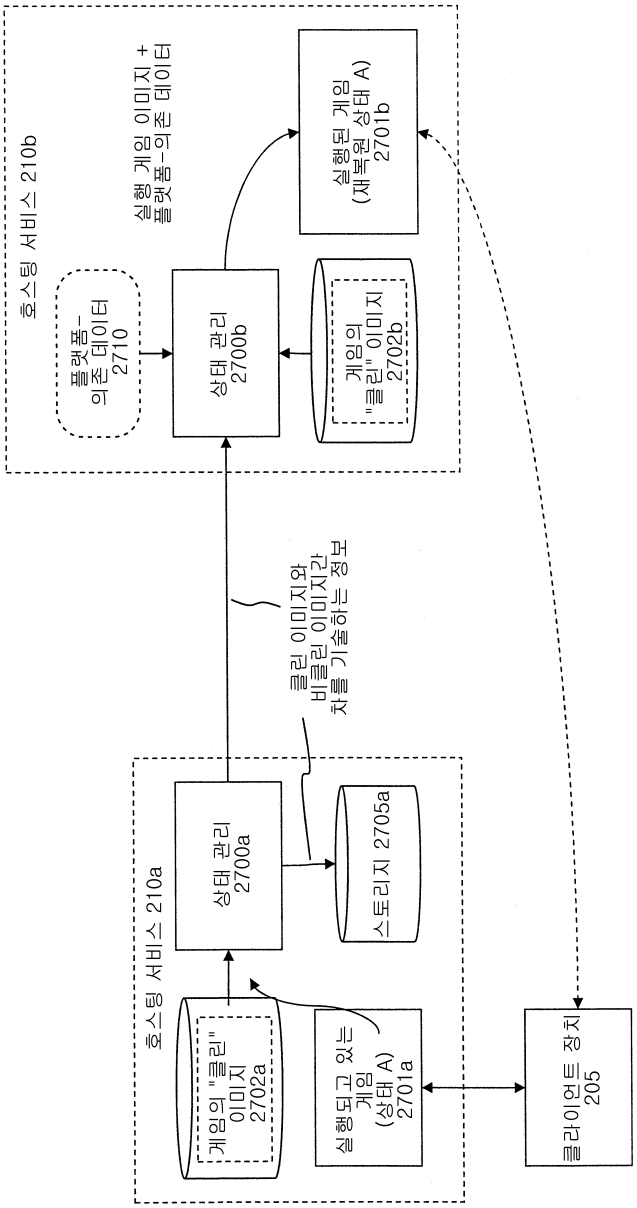
도면26a


신장된
타일/프레임
2607
1
2
3
클라이언트 장치 205
디코더
2602
디코더의
페이스트 상태(들)
2621
메모리
2620
피드백 2601
압축된
타일/프레임
2606
5_4
6_5
7_3
4_3
타일/프레임(4)
손실
패킷 손실이 검출된 후
타일/프레임(6)이 아니라
타일/프레임(3)에 따라
호스팅 서비스 210
인코더
2600
인코더의
페이스트 상태(들)
2611
메모리
2610
비압축
프레임/타일
2605
8
9
10
11

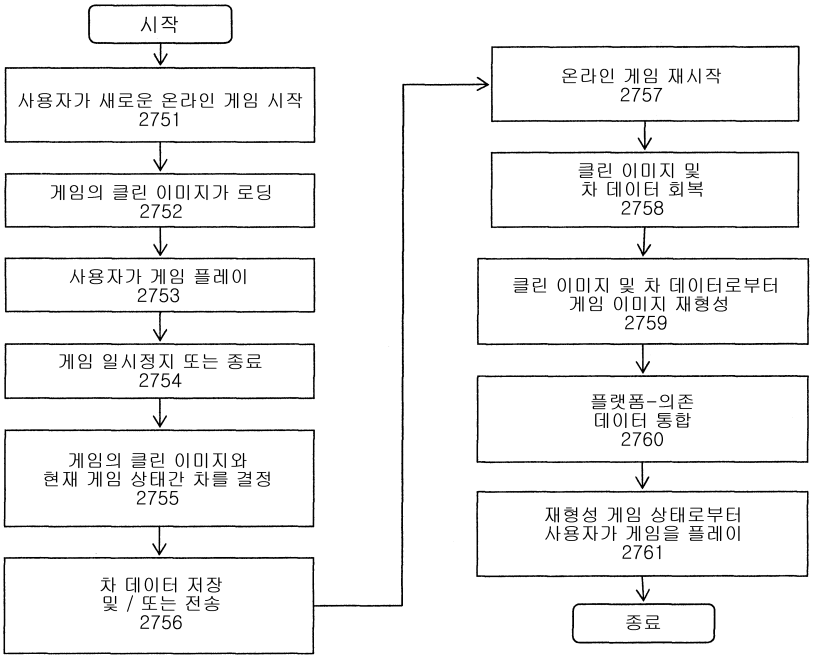
도면26b


시작
타일/프레임이 앞의
타일/프레임에 따라 생성
2650
손실 타일/프레임 검출
2651
성공적으로 수신된 타일/프레임에 따라
다음 타일/프레임 생성
2652
종료

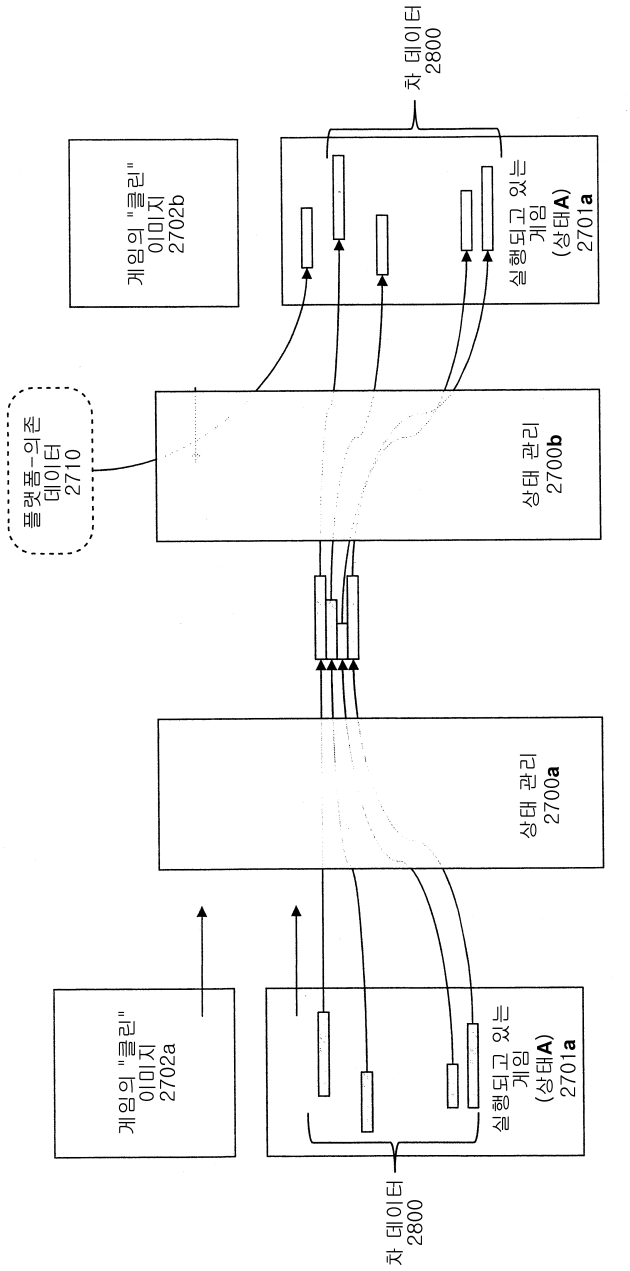
도면27a


호스팅 서비스 210b
실행 게임 이미지 +
플랫폼-의존 데이터
실행된 게임
(재복원 상태 A)
2701b
플랫폼-
의존 데이터
2710
상태 관리
2700b
게임의
"클린" 이미지
2702b
클린 이미지와
비클린 이미지간
차를 기술하는 정보
호스팅 서비스 210a
상태 관리
2700a
스토리지 2705a
게임의 "클린"
이미지
2702a
실행되고 있는
게임
(상태 A)
2701a
클라이언트 장치
205

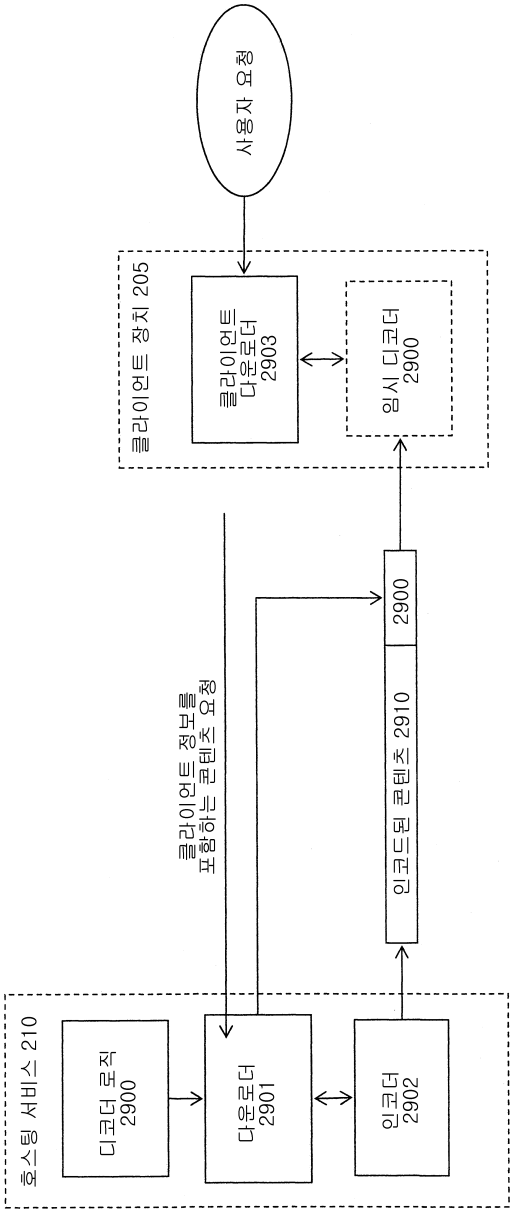
도면27b


시작
사용자가 새로운 온라인 게임 시작
2751
게임의 클린 이미지가 로딩
2752
사용자가 게임 플레이
2753
게임 일시정지 또는 종료
2754
게임의 클린 이미지와
현재 게임 상태간 차를 결정
2755
차 데이터 저장
및 / 또는 전송
2756
온라인 게임 재시작
2757
클린 이미지 및
차 데이터 회복
2758
클린 이미지 및 차 데이터로부터
게임 이미지 재형성
2759
플랫폼-의존
데이터 통합
2760
재형성 게임 상태로부터
사용자가 게임을 플레이
2761
종료

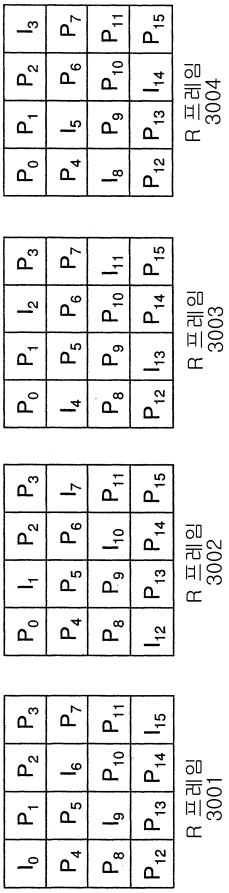
도면28


차 데이터
2800
게임의 "클린"
이미지
2702b
실행되고 있는
게임
(상태A)
2701a
플랫폼-의존
데이터
2710
상태 관리
2700b
상태 관리
2700a
게임의 "클린"
이미지
2702a
실행되고 있는
게임
(상태A)
2701a
차 데이터
2800

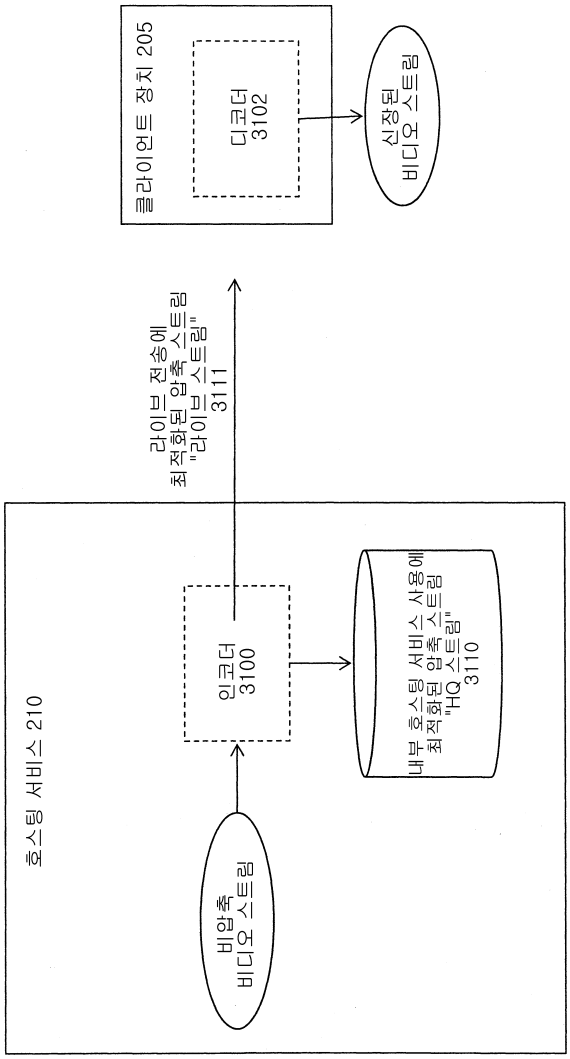
도면29


사용자 요청
클라이언트 장치 205
클라이언트 다운로더 2903
임시 디코더 2900
2900
인코드된 콘텐츠 2910
클라이언트 정보를 포함하는 콘텐츠 요청
호스팅 서비스 210
디코더 로직 2900
다운로더 2901
인코더 2902

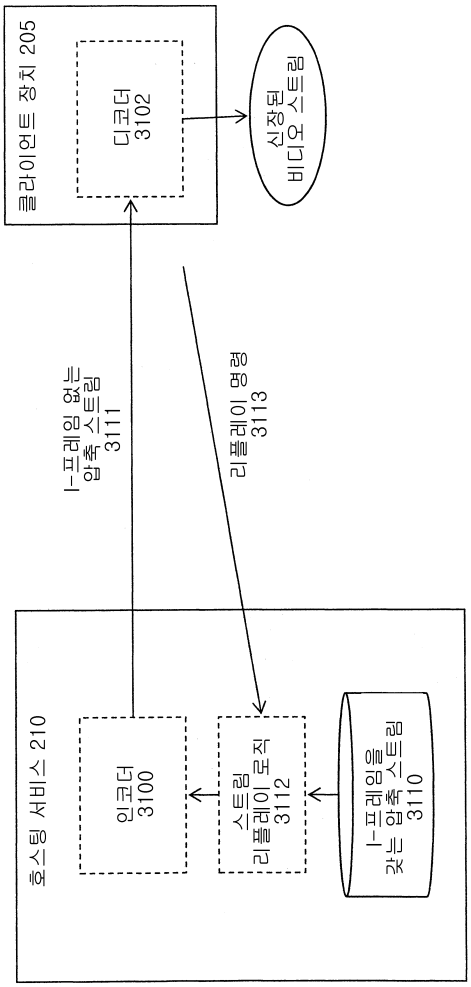
도면30



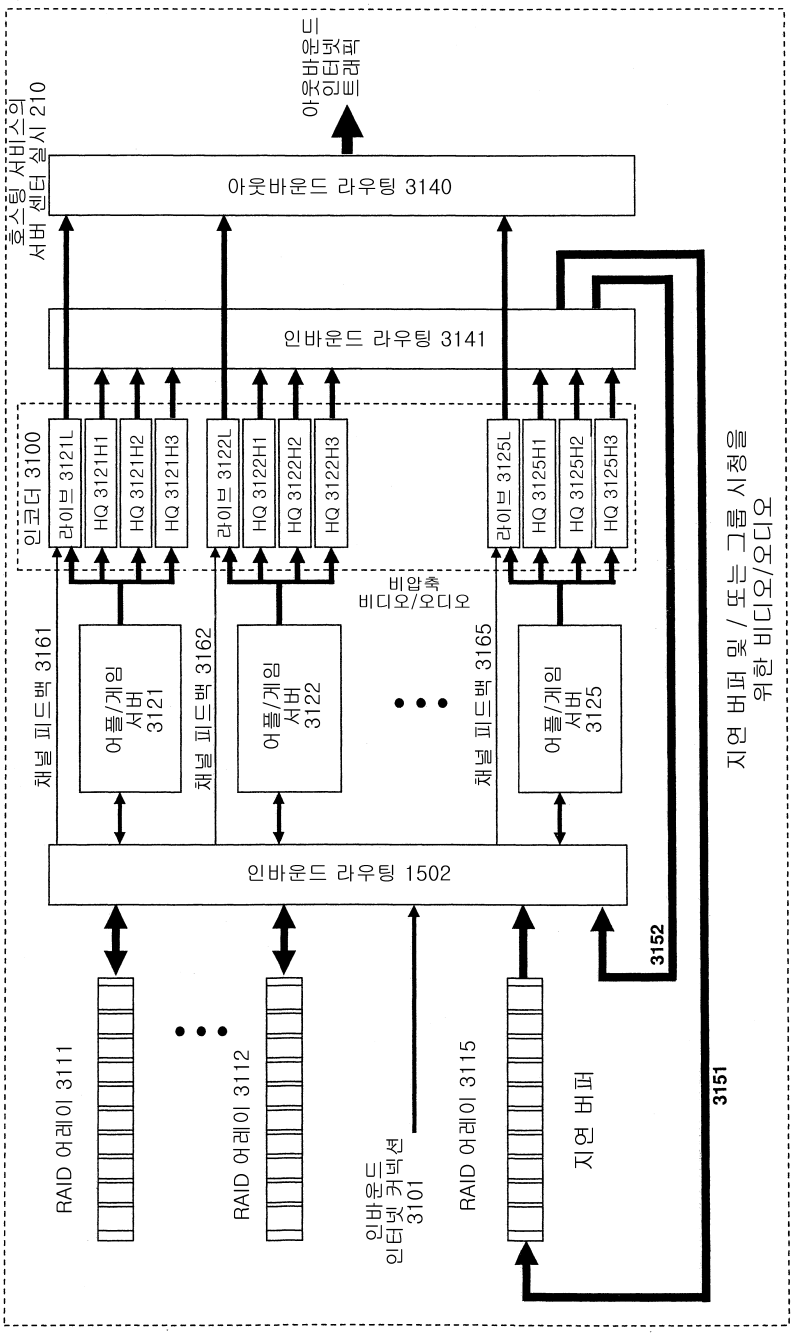
도면31a


클라이언트 장치 205
디코더
3102
신장된
비디오 스트림
라이브 전송에
최적화된 압축 스트림
"라이브스트림"
3111
호스팅 서비스 210
인코더
3100
내부 호스팅 서비스 사용에
최적화된 압축 스트림
"HQ 스트림"
3110
비압축
비디오 스트림

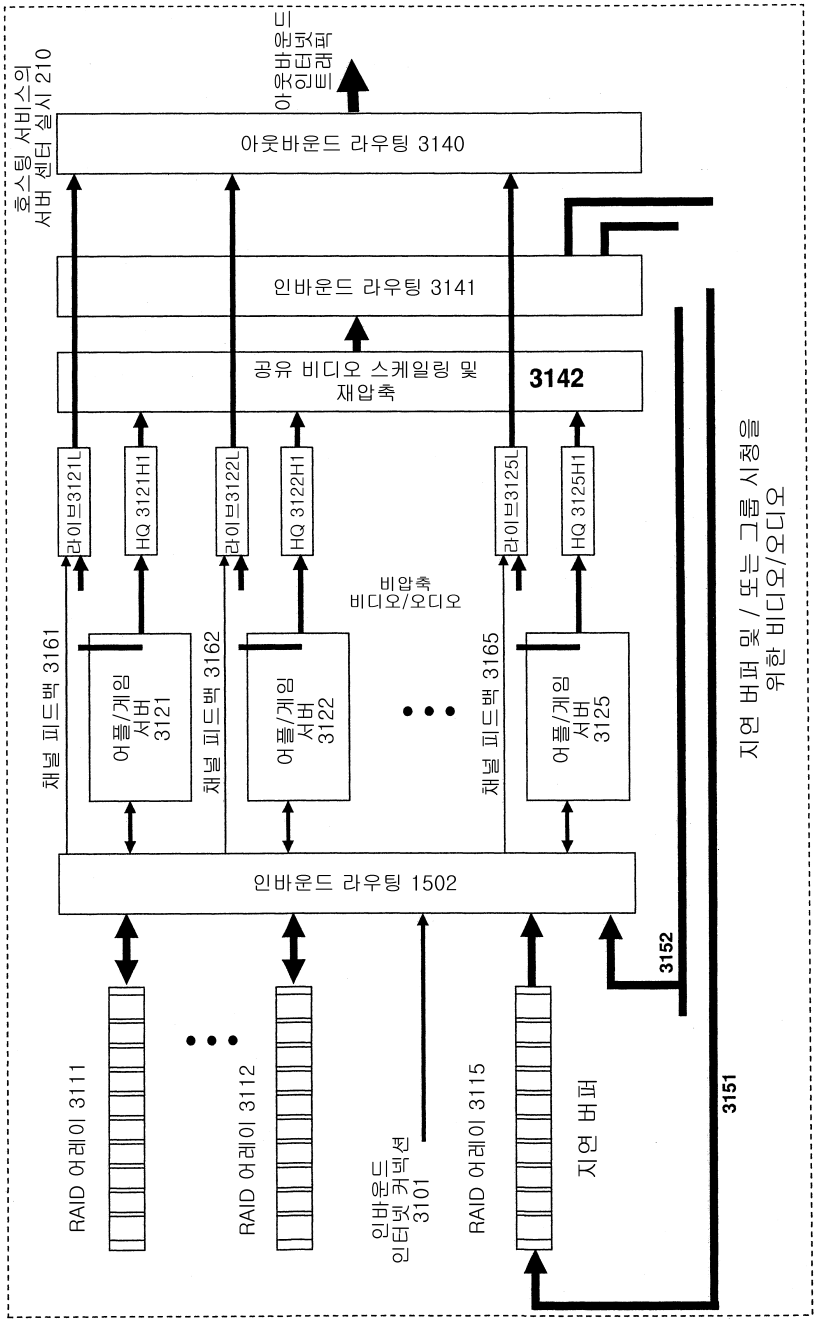
도면31b


클라이언트 장치 205
디코더
3102
신장된
비디오 스트림
I-프레임 없는
압축 스트림
3111
리플레이 명령
3113
호스팅 서비스 210
인코더
3100
스트림
리플레이 로직
3112
I-프레임을
갖는 압축 스트림
3110

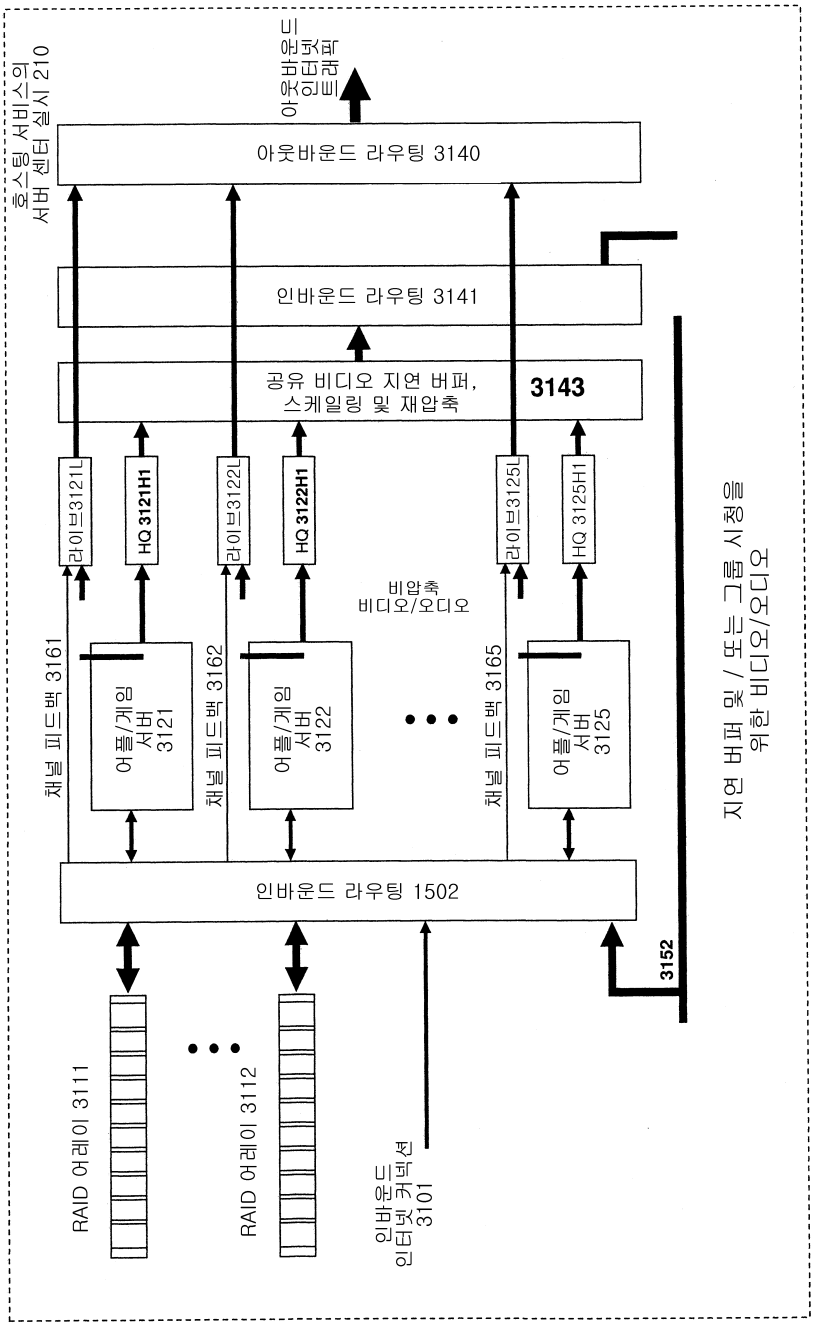
도면31c


호스팅 서비스의 서버 센터 실시 210
아웃바운드 인터넷 트래픽
아웃바운드 라우팅 3140
인바운드 라우팅 3141
인코더 3100
라이브 3121L
HQ 3121H1
HQ 3121H2
HQ 3121H3
라이브 3122L
HQ 3122H1
HQ 3122H2
HQ 3122H3
라이브 3125L
HQ 3125H1
HQ 3125H2
HQ 3125H3
비압축 비디오/오디오
채널 피드백 3161
채널 피드백 3162
채널 피드백 3165
어플/게임 서버 3121
어플/게임 서버 3122
어플/게임 서버 3125
인바운드 라우팅 1502
3152
3151
지연 버퍼 및 / 또는 그룹 시청을 위한 비디오/오디오
RAID 어레이 3111
RAID 어레이 3112
인바운드 인터넷 커넥션 3101
RAID 어레이 3115
지연 버퍼

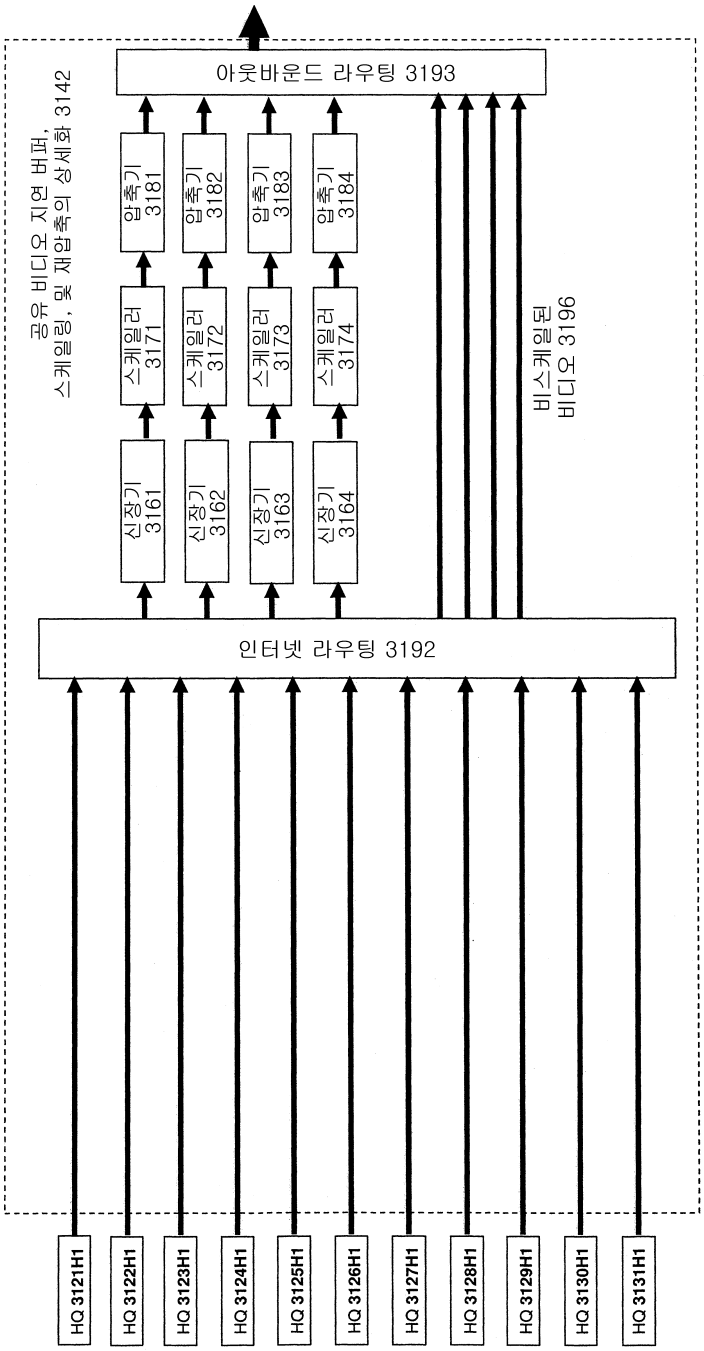
도면31d


호스팅 서비스의
서버 센터 설시 210
아웃바운드 인터넷 트래픽
아웃바운드 라우팅 3140
인바운드 라우팅 3141
공유 비디오 스케일링 및 재압축
3142
라이브3121L
HQ 3121H1
라이브3122L
HQ 3122H1
라이브3125L
HQ 3125H1
비압축 비디오/오디오
채널 피드백 3161
채널 피드백 3162
채널 피드백 3165
어플/게임 서버 3121
어플/게임 서버 3122
어플/게임 서버 3125
인바운드 라우팅 1502
RAID 어레이 3111
RAID 어레이 3112
RAID 어레이 3115
인바운드 인터넷 커넥션 3101
지연 버퍼
3152
3151
지연 버퍼 및 / 또는 그룹 시청을 위한 비디오/오디오

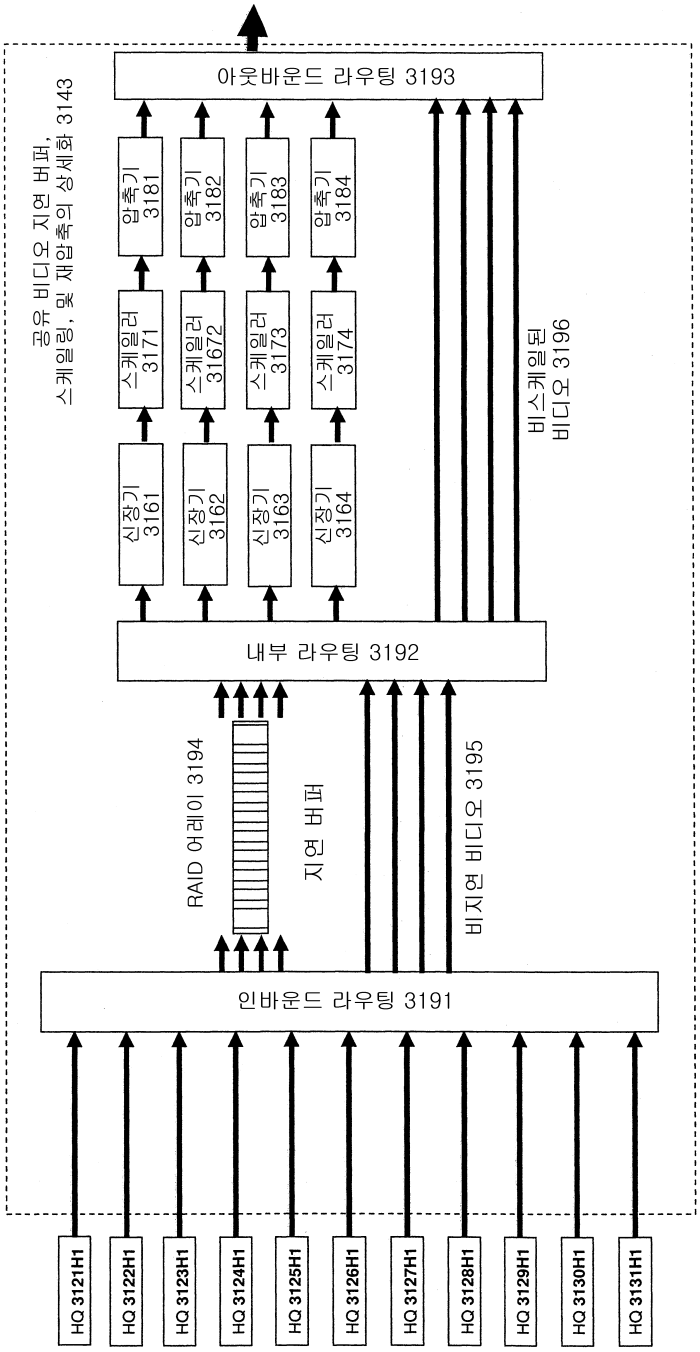
도면31e


호스팅 서비스의
서버 센터 실시 210
아웃바운드
인터넷
아웃바운드 라우팅 3140
인바운드 라우팅 3141
공유 비디오 지연 버퍼,
스케일링 및 재압축
3143
라이브3121L
HQ 3121H1
라이브3122L
HQ 3122H1
비압축
비디오/오디오
라이브3125L
HQ 3125H1
지연 버퍼 및 / 또는 그룹 시청을
위한 비디오/오디오
채널 피드백 3161
채널 피드백 3162
채널 피드백 3165
어플/게임
서버
3121
어플/게임
서버
3122
어플/게임
서버
3125
인바운드 라우팅 1502
3152
RAID 어레이 3111
RAID 어레이 3112
인바운드
인터넷 커넥션
3101

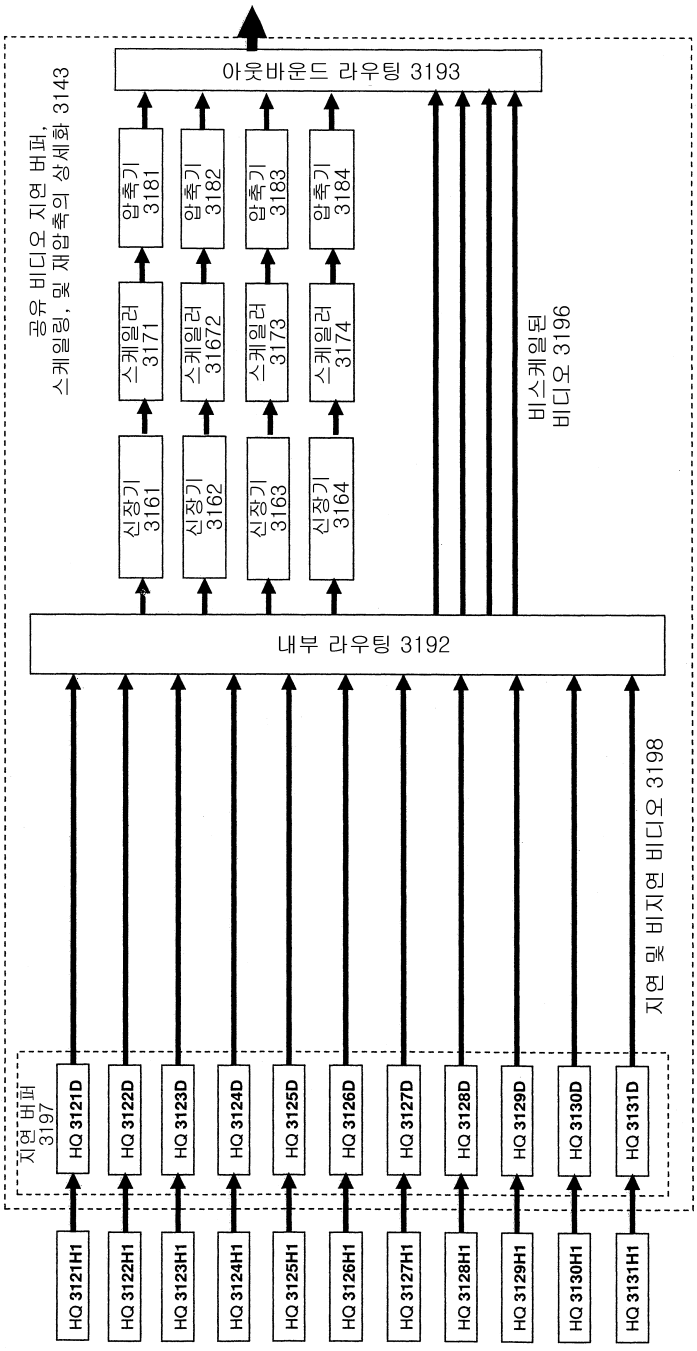
도면31f


아웃바운드 라우팅 3193
공유 비디오 지연 버퍼,
스케일링, 및 재압축의 상세화 3142
압축기 3181
압축기 3182
압축기 3183
압축기 3184
스케일러 3171
스케일러 3172
스케일러 3173
스케일러 3174
신장기 3161
신장기 3162
신장기 3163
신장기 3164
비스케일된
비디오 3196
인터넷 라우팅 3192
HQ 3121H1
HQ 3122H1
HQ 3123H1
HQ 3124H1
HQ 3125H1
HQ 3126H1
HQ 3127H1
HQ 3128H1
HQ 3129H1
HQ 3130H1
HQ 3131H1

도면31g


공유 비디오 지연 버퍼,
스케일링, 및 재압축의 상세화 3143
아웃바운드 라우팅 3193
압축기 3181
압축기 3182
압축기 3183
압축기 3184
스케일러 3171
스케일러 31672
스케일러 3173
스케일러 3174
신장기 3161
신장기 3162
신장기 3163
신장기 3164
비스케일된 비디오 3196
내부 라우팅 3192
RAID 어레이 3194
지연 버퍼
비지연 비디오 3195
인바운드 라우팅 3191
HQ 3121H1
HQ 3122H1
HQ 3123H1
HQ 3124H1
HQ 3125H1
HQ 3126H1
HQ 3127H1
HQ 3128H1
HQ 3129H1
HQ 3130H1
HQ 3131H1

도면31h


아웃바운드 라우팅 3193
공유 비디오 지연 버퍼,
스케일링, 및 재압축의 상세화 3143
압축기 3181
압축기 3182
압축기 3183
압축기 3184
스케일러 3171
스케일러 31672
스케일러 3173
스케일러 3174
신장기 3161
신장기 3162
신장기 3163
신장기 3164
비스케일된
비디오 3196
내부 라우팅 3192
지연 및 비지연 비디오 3198
지연 버퍼 3197
HQ 3121D
HQ 3122D
HQ 3123D
HQ 3124D
HQ 3125D
HQ 3126D
HQ 3127D
HQ 3128D
HQ 3129D
HQ 3130D
HQ 3131D
HQ 3121H1
HQ 3122H1
HQ 3123H1
HQ 3124H1
HQ 3125H1
HQ 3126H1
HQ 3127H1
HQ 3128H1
HQ 3129H1
HQ 3130H1
HQ 3131H1