(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2022-0138403
(43) 공개일자 2022년10월12일

(51) 국제특허분류(Int. Cl.)
H04N 13/344 (2018.01) *G06T 17/20* (2006.01)
G06T 19/00 (2011.01) *H04N 13/139* (2018.01)
H04N 13/15 (2018.01) *H04N 13/161* (2018.01)
H04N 13/194 (2018.01) *H04N 13/279* (2018.01)
H04N 13/383 (2018.01) *H04N 19/182* (2014.01)
H04N 19/517 (2014.01)
(52) CPC특허분류
H04N 13/344 (2018.05)
G06T 17/20 (2013.01)
(21) 출원번호 10-2022-7031146
(22) 출원일자(국제) 2021년02월12일
심사청구일자 없음
(85) 번역문제출일자 2022년09월07일
(86) 국제출원번호 PCT/US2021/017967
(87) 국제공개번호 WO 2021/163570
국제공개일자 2021년08월19일
(30) 우선권주장
16/792,080 2020년02월14일 미국(US)

(71) 출원인
밸브 코포레이션
미국 워싱턴 (우편번호 98004) 벨뷰 엔이 포쓰 스트리트 10400 스위트 1400
(72) 발명자
레이비, 아론
미국 98004 워싱턴 벨뷰 노스이스트 4번 스트리트 10400
(74) 대리인
양영준

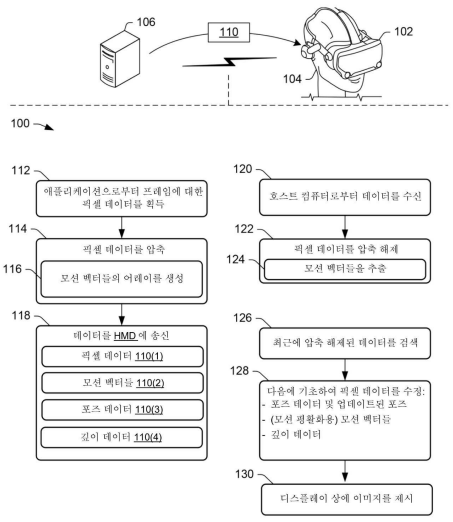
전체 청구항 수 : 총 20 항

(54) 발명의 명칭 **분산 시스템에서 모션 평활화**

(57) 요 약

저더(judder)를 완화하는 방식으로 객체를 움직이거나 애니메이션하는 모션을 처리하기 위해, 헤드 마운트 디스플레이(HMD: head-mounted display)와 같은 디스플레이 또는 디스플레이 시스템을 위한 모션 평활화(smooting) 기술들이 본원에 설명된다. 디스플레이 시스템은, 비디오 게임과 같은 그래픽 기반 애플리케이션이 디스플레이 시스템 상에서 렌더링하기 위한 프레임들을 출력하는 호스트 컴퓨터와 분리되어 있지만, 통신 가능하게 연결될 수 있다. 호스트 컴퓨터는 디스플레이 시스템으로의 전송을 위해 압축된 픽셀 데이터를 나타내는 모션 벡터들을 생성할 수 있다. 모션 벡터들은 프레임의 픽셀 데이터를 수정하기 위해서, 디스플레이 시스템에 의해서, 사용될 수 있다. 프레임에 대한 수정된 픽셀 데이터는, 움직이거나 애니메이션되는 객체의 저더를 완화하는 방식으로 디스플레이 시스템 상에 렌더링되기 위해 "모션 평활화"된다.

대 표 도 - 도1



(52) CPC특허분류
G06T 19/006 (2013.01)
H04N 13/139 (2018.05)
H04N 13/15 (2018.05)
H04N 13/161 (2018.05)
H04N 13/194 (2018.05)
H04N 13/279 (2018.05)
H04N 13/383 (2018.05)
H04N 19/182 (2015.01)
H04N 19/517 (2015.01)

명 세 서

청구범위

청구항 1

헤드 마운트 디스플레이(HMD: head-mounted display)로서,

하나 또는 둘 이상의 디스플레이 패널들;

프로세서; 및

컴퓨터 실행 가능 명령어들을 저장하는 메모리를 포함하되, 상기 컴퓨터 실행 가능 명령어들은, 상기 프로세서에 의해서 실행될 때, 상기 HMD로 하여금,

상기 HMD에 통신 가능하게 결합된 호스트 컴퓨터로부터, 압축된 픽셀 데이터를 나타내는 모션 벡터들의 어레이를 수신하고,

픽셀 데이터를 획득하기 위해서 상기 압축된 픽셀 데이터를 압축 해제하고,

수정된 픽셀 데이터를 얻기 위해 상기 픽셀 데이터를 수정하고 - 상기 픽셀 데이터는,

상기 HMD의 예측된 포즈(pose), 및

상기 모션 벡터들의 어레이에 적어도 부분적으로 기초하여 수정됨 -,

상기 수정된 픽셀 데이터에 적어도 부분적으로 기초하여 상기 하나 또는 둘 이상의 디스플레이 패널들 상에 이미지를 제시하게 하는, HMD.

청구항 2

제1항에 있어서, 상기 컴퓨터 실행 가능 명령어들은, 상기 프로세서에 의해서 실행될 때, 상기 HMD로 하여금 추가로,

복수의 정점들을 갖는 바둑판 모양화된 메시(tessellated mesh)를 포함하는 렌더 메시(render mesh)를 생성하고,

상기 모션 벡터들의 어레이 중에서 넌-제로(non-zero) 모션 벡터들에 적어도 부분적으로 기초하여, 상기 복수의 정점들 중 정점들을 이동된 정점들로서 상기 렌더 메시 내의 상이한 위치들로 이동시키게 하며,

상기 모션 벡터들의 어레이에 적어도 부분적으로 기초하여 상기 픽셀 데이터를 수정하는 것은, 상기 수정된 픽셀 데이터를 획득하기 위해서 상기 렌더 메쉬의 상기 이동된 정점들에 따라 상기 픽셀 데이터의 픽셀 값들을 이동시키는 것을 포함하는, HMD.

청구항 3

제2항에 있어서, 상기 정점들은,

상기 넌-제로 모션 벡터들의 방향들로, 그리고

상기 넌-제로 모션 벡터들의 크기들에 대응되는 양들만큼 이동된, HMD.

청구항 4

제1항에 있어서, 상기 컴퓨터 실행 가능 명령어들은, 상기 프로세서에 의해서 실행될 때, 상기 HMD로 하여금 추가로, 인코딩된 데이터 스트림으로부터 상기 모션 벡터들의 어레이를 추출하게 하는, HMD.

청구항 5

제1항에 있어서,

상기 호스트 컴퓨터는 상기 HMD에 무선으로 결합되고,

상기 모션 벡터들의 어레이는 상기 호스트 컴퓨터로부터 무선으로 수신되는, HMD.

청구항 6

제1항에 있어서, 상기 컴퓨터 실행 가능 명령어들은, 상기 프로세서에 의해서 실행될 때, 상기 HMD로 하여금 추가로,

모션 벡터들의 수정된 어레이를 획득하기 위해 상기 모션 벡터들의 어레이에 필터를 적용하게 하며,

상기 모션 벡터들의 어레이에 적어도 부분적으로 기초하여 상기 픽셀 데이터를 수정하는 것은, 상기 모션 벡터들의 수정된 어레이에 적어도 부분적으로 기초하여 상기 픽셀 데이터를 수정하는 것을 포함하는, HMD.

청구항 7

방법으로서,

압축된 픽셀 데이터를 나타내는 모션 벡터들의 어레이를 호스트 컴퓨터로부터, 헤드 마운트 디스플레이(HMD)에 의해서, 수신하는 단계;

픽셀 데이터를 획득하기 위해서 상기 압축된 픽셀 데이터를, 상기 HMD에 의해서, 압축 해제하는 단계;

수정된 픽셀 데이터를 획득하기 위해서 상기 모션 벡터들의 어레이에 적어도 부분적으로 기초하여 상기 픽셀 데이터를, 상기 HMD에 의해서, 수정하는 단계; 및

상기 수정된 픽셀 데이터에 적어도 부분적으로 기초하여 상기 HMD의 하나 또는 둘 이상의 디스플레이 패널들 상에 이미지를 제시하는 단계를 포함하는, 방법.

청구항 8

제7항에 있어서, 상기 픽셀 데이터를 수정하는 단계 전에,

상기 픽셀 데이터를 상기 HMD의 메모리에, 캐시(cache)된 픽셀 데이터로서 캐시하는 단계;

상기 캐시된 픽셀 데이터가 상기 HMD에 이용 가능한 상기 가장 최근에 압축 해제된 픽셀 데이터를 나타낸다고 결정하는 단계; 및

상기 HMD의 상기 메모리로부터 상기 캐시된 픽셀 데이터를 검색하는 단계를 더 포함하는 방법.

청구항 9

제7항에 있어서,

상기 방법은, 복수의 정점들을 갖는 바둑판 모양화된 메시(tessellated mesh)를 포함하는 렌더 메시(render mesh)를, 상기 HMD에 의해서, 생성하는 단계; 및

상기 모션 벡터들의 어레이 중에서 넌-제로(non-zero) 모션 벡터들에 적어도 부분적으로 기초하여, 상기 복수의 정점들 중 정점들을 이동된 정점들로서 상기 렌더 메시 내의 상이한 위치들로 이동시키는 단계를 더 포함하되,

상기 모션 벡터들의 어레이에 적어도 부분적으로 기초하여 상기 픽셀 데이터를 수정하는 단계는, 상기 수정된 픽셀 데이터를 획득하기 위해서 상기 렌더 메쉬의 상기 이동된 정점들에 따라 상기 픽셀 데이터의 픽셀 값들을 이동시키는 단계를 포함하는, 방법.

청구항 10

제7항에 있어서, 인코딩된 데이터 스트림으로부터 상기 모션 벡터들의 어레이를 추출하는 단계를 더 포함하는 방법.

청구항 11

제7항에 있어서, 상기 방법은, 상기 이미지를 제시하는 단계 전에,

상기 픽셀 데이터를 생성하기 위해서 상기 호스트 컴퓨터 상에서 실행되는 애플리케이션에 의해 사용된 상기 HMD의 예측된 포즈를 나타내는 포즈 데이터를, 상기 호스트 컴퓨터로부터, 상기 HMD에 의해서, 수신하는 단계를

더 포함하되,

상기 픽셀 데이터를 수정하는 단계는 또한 상기 HMD의 상기 예측된 포즈 및 업데이트된 포즈 예측 간의 비교에 적어도 부분적으로 기초하는, 방법.

청구항 12

제7항에 있어서, 상기 방법은, 상기 이미지를 제시하는 단계 전에,

상기 호스트 컴퓨터 상에서 실행되는 애플리케이션에 의해서 출력된 Z-버퍼 데이터를 나타내는 깊이 데이터를, 상기 호스트 컴퓨터로부터, 상기 HMD에 의해서, 수신하는 단계를 더 포함하되,

상기 픽셀 데이터를 수정하는 단계는 추가로 상기 깊이 데이터에 적어도 부분적으로 기초하는, 방법.

청구항 13

제7항에 있어서,

상기 방법은, 모션 벡터들의 수정된 어레이를 획득하기 위해 상기 모션 벡터들의 어레이에 필터를, 상기 HMD에 의해서, 적용하는 단계를 더 포함하되,

상기 모션 벡터들의 어레이에 적어도 부분적으로 기초하여 상기 픽셀 데이터를 수정하는 단계는 상기 모션 벡터들의 수정된 어레이에 기초하여 상기 픽셀 데이터를 수정하는 단계를 포함하는, 방법.

청구항 14

디스플레이 시스템으로서,

하나 또는 둘 이상의 디스플레이 패널들;

프로세서; 및

컴퓨터 실행 가능 명령어들을 저장하는 메모리를 포함하되, 상기 컴퓨터 실행 가능 명령어들은, 상기 프로세서에 의해서 실행될 때, 상기 디스플레이 시스템으로 하여금,

압축된 픽셀 데이터를 나타내는 모션 벡터들의 어레이를 호스트 컴퓨터로부터 수신하고,

픽셀 데이터를 획득하기 위해서 상기 압축된 제1 픽셀 데이터를 압축 해제하고,

수정된 픽셀 데이터를 획득하기 위해서 상기 모션 벡터들의 어레이에 적어도 부분적으로 기초하여 상기 픽셀 데이터를 수정하고,

상기 수정된 픽셀 데이터에 적어도 부분적으로 기초하여 상기 하나 또는 둘 이상의 디스플레이 패널들 상에 이미지를 제시하게 하는, 디스플레이 시스템.

청구항 15

제14항에 있어서, 상기 컴퓨터 실행 가능 명령어들은, 상기 프로세서에 의해서 실행될 때, 상기 디스플레이 시스템으로 하여금, 상기 픽셀 데이터를 수정하는 것 전에,

상기 픽셀 데이터를 상기 메모리에, 캐시된 픽셀 데이터로서 캐시하고,

상기 캐시된 픽셀 데이터가 상기 HMD에 이용 가능한 상기 가장 최근에 압축 해제된 픽셀 데이터를 나타낸다고 결정하고,

상기 메모리로부터 상기 캐시된 픽셀 데이터를 검색하게 하는, 디스플레이 시스템.

청구항 16

제14항에 있어서, 상기 컴퓨터 실행 가능 명령어들은, 상기 프로세서에 의해서 실행될 때, 상기 디스플레이 시스템으로 하여금,

복수의 정점들을 갖는 바둑판 모양화된 메시(tessellated mesh)를 포함하는 렌더 메시(render mesh)를 생성하고,

상기 모션 벡터들의 어레이 중에서 넌-제로(non-zero) 모션 벡터들에 적어도 부분적으로 기초하여, 상기 복수의 정점들 중 정점들을 이동된 정점들로서 상기 렌더 메시 내의 상이한 위치들로 이동시키게 하며,

상기 모션 벡터들의 어레이에 적어도 부분적으로 기초하여 상기 픽셀 데이터를 수정하는 것은, 상기 수정된 픽셀 데이터를 획득하기 위해서 상기 렌더 메쉬의 상기 이동된 정점들에 따라 상기 픽셀 데이터의 픽셀 값들을 이동시키는 것을 포함하는, 디스플레이 시스템.

청구항 17

제14항에 있어서,

상기 호스트 컴퓨터는 상기 디스플레이 시스템에 무선으로 결합되고,

상기 모션 벡터들의 어레이는 상기 호스트 컴퓨터로부터 무선으로 수신되는, 디스플레이 시스템.

청구항 18

제14항에 있어서, 상기 컴퓨터 실행 가능 명령어들은, 상기 프로세서에 의해서 실행될 때, 상기 디스플레이 시스템으로 하여금, 상기 이미지를 제시하는 것 전에,

상기 호스트 컴퓨터 상에서 실행되는 애플리케이션에 의해서 출력된 Z-버퍼 데이터를 나타내는 깊이 데이터를, 상기 호스트 컴퓨터로부터 수신하고,

상기 픽셀 데이터를 수정하는 것은 추가로 상기 깊이 데이터에 적어도 부분적으로 기초하는, 디스플레이 시스템.

청구항 19

제14항에 있어서, 상기 컴퓨터 실행 가능 명령어들은, 상기 프로세서에 의해서 실행될 때, 상기 디스플레이 시스템으로 하여금,

모션 벡터들의 수정된 어레이를 획득하기 위해 상기 모션 벡터들의 어레이에 필터를 적용하게 하며,

상기 모션 벡터들의 어레이에 적어도 부분적으로 기초하여 상기 픽셀 데이터를 수정하는 것은 상기 모션 벡터들의 수정된 어레이에 기초하여 상기 픽셀 데이터를 수정하는 것을 포함하는, 디스플레이 시스템.

청구항 20

제14항에 있어서, 상기 디스플레이 시스템은 가상 현실(VR: virtual reality) 헤드셋 또는 증강 현실(AR: augmented reality) 헤드셋 중 적어도 하나를 포함하는, 디스플레이 시스템.

발명의 설명

기 술 분 야

[0001] 관련 출원의 교차 참조

[0002] 본원은, 2020년 2월 14일자로 출원되고, 발명의 명칭이 "MOTION SMOOTHING IN A DISTRIBUTED SYSTEM"인, 동시계류 중이고 공동으로 양도된 미국 특허 출원 제16/792,080호에 대한 우선권을 주장하는 PCT 출원이며, 이 미국 출원은 그 전체가 본원에 인용되어 포함된다.

배 경 기 술

[0003] 가상 현실(VR: virtual reality) 시스템은 비디오 게임 산업 내외 모두에서 사용된다. VR 시스템은 올인원(all-in-one)(또는 독립형) VR 헤드셋으로서, 또는 분산 시스템으로서 설정될 수 있다. 분산 설정에서, VR 헤드셋은 호스트 컴퓨터에 물리적으로 연결되거나 무선으로 연결될 수 있다. 이러한 분산 설정에서, 호스트 컴퓨터는 일반적으로 프레임들을 출력하는 비디오 게임과 같은 그래픽 기반 애플리케이션을 실행하고, VR 헤드셋은 호스트 컴퓨터로부터 프레임들이 스트리밍될 때 이들을 디스플레이한다. 이러한 유형의 설정은, "씬 클라이언트(thin-client)" 디바이스와 매우 유사하게 기능하는 경량 VR 헤드셋에 고품질 이미지를 디스플레이하기 위해서 호스트 컴퓨터의 고성능 컴퓨팅 용량을 활용한다. 또한, VR 헤드셋에 내장된 것과 같은 VR 시스템들용 디스플레이들은 통상적으로 VR 애플리케이션들에 적합한 최소 재생률로 작동된다. 예를 들어, 90 헤르츠(Hz)는 VR 디스플레이의

일반적인 재생률이다. "라이브 렌더링(live rendering)" 시나리오에서, 그래픽 기반 애플리케이션은 디스플레이의 재생률과 매칭되는 프레임 레이트로 렌더링하도록 프레임들을 출력한다. 이 시나리오에서, 프레임들이 호스트 컴퓨터로부터 VR 헤드셋으로 충분한 데이터 전송 속도로 전송되고 있다고 가정하면, 애플리케이션에 의해 출력된 새로운 프레임(본원에서 "실제 프레임(actual frame)"이라고 함)은 매 화면 리프레시에 디스플레이될 수 있다. 이러한 라이브 렌더링 시나리오는 종종 애플리케이션이 "프레임 레이트에 도달하는" 것으로 지칭된다.

[0004] 실제로, 애플리케이션은 다양한 이유로 인해 항상 프레임 레이트에 도달하는 것은 아니다. 예를 들어, 애플리케이션은 간헐적으로 프레임을 드랍할 수 있고/있거나, 애플리케이션은 일시적으로 더 느린 레이트(예를 들어, 이상적인 프레임 레이트가 초당 90 프레임인 경우 초당 45 프레임)로 프레임들을 출력할 수 있다. 또한, 분산 시스템에서, 네트워크 정체로 인해, 호스트 컴퓨터로부터 VR 헤드셋으로 데이터가 전송되는 속도에 레이턴시(latency)가 도입될 수 있다. 이러한 상황에서, "회전 전용 재투영(rotation-only re-projection)"이라는 기술을 사용하여, 누락된 프레임들을, 사용자의 머리 회전을 처리하는 방식으로 재투영된 프레임으로 대체할 수 있어, 마치 누락된 프레임이 없는 것처럼 사용자에게 보이게 할 수 있다. 예를 들어, 재투영이 없으면, 애플리케이션으로부터의 부족한 프레임 레이트, 또는 VR 헤드셋에 프레임들이 늦게 도착하는 것은 게임 내 끊김 현상(in-game stuttering)이나 장애 현상(hitching)을 야기할 수 있다. 사용자가 가상 환경에 완전히 몰입되는 VR 애플리케이션에서, 만약 프레임들이 누락되고, 누락된 프레임들을 보상하기 위한 재투영이 없으면, 사용자는 메스꺼워질 수 있다. 따라서, 재투영은 프레임들이 누락되었을 때 더 나은 사용자 경험을 제공하는 기술이다. 애플리케이션이 이상적인 프레임 레이트의 절반(예를 들어, 초당 90프레임이 이상적인 프레임 레이트인 경우 초당 45프레임)으로 프레임을 출력하는 일례를 고려한다. 이러한 예에서, 다른 모든 프레임은 가장 최근에 렌더링된 실제 프레임으로부터 픽셀 데이터를 사용하여 재투영되어 사용자의 현재 머리 방향으로 재투영된 장면과 정합하도록 (예를 들어, 회전 및 재투영 계산을 통해) 장면을 변환하는 재투영된 프레임을 생성할 수 있다. 이를 통해, 재투영된 프레임들이 누락된 프레임들을 보상하기 위해서 사용되는 경우에도, 사용자의 머리 회전을 고려하여 예상되는 방식으로 장면이 움직이는 것처럼 사용자에게 보인다.

[0005] 회전 전용 재투영은 게임 내 끊김 현상 또는 장애 현상을 방지하지만, 적어도 낮은 지속성 디스플레이를 사용하는 VR 시스템(예를 들어, 디스플레이가 프레임 시간의 극히 일부 동안 조명되는 경우)에서 머리의 회전 동안 그 자체의 원치 않는 시각적 아티팩트를 생성한다. 예를 들어, 회전 전용 재투영은 머리 회전을 처리하지만, 장면에서 프레임 사이에서 이동하거나 움직이는 가상 객체를 처리하지 않는다. 이로 인해 "저더(judder)"라는 원치 않는 시각적 아티팩트가 이동하거나 애니메이션되는 객체들과 관련하여 발생될 수 있다. 저더로 인해 사용자는 이동하는 객체(예를 들어, 화면을 가로지르는 총알이나 공)가 두 개의 위치들 사이에서 프레임 간 튀는(또는 그 자체로부터 분리되는) 것처럼 보이는 경우에 "이중 고스팅 효과"를 인지한다. 따라서, 재투영이 사용되는 동안 사용자가 자신의 머리를 회전시키면, 장면에서 이동하거나 애니메이션되는 객체가 저더링될 것이다.

[0006] 이러한 시스템 및 다른 시스템을 개선시키고 향상시키기 위한 기술적 솔루션들이 본원에서 제공된다.

도면의 간단한 설명

[0007] 상세한 설명은 수반된 도면을 참조하여 설명된다. 도면에서, 참조 번호의 맨 왼쪽 숫자(들)는 이 참조 번호가 처음으로 나타나는 도면을 식별한다. 여러 도면에서의 동일한 도면 부호의 사용은 유사하거나 동일한 구성 요소 또는 특징부를 나타낸다.

도 1은, 본원에 개시된 실시형태에 따라, HMD에 통신 가능하게 결합된 호스트 컴퓨터 상에서 실행되는 애플리케이션에 의해 프레임들이 출력될 때 헤드 마운트 디스플레이(HMD: head-mounted display) 상에 프레임들을 렌더링하기 위한 예시적인 프로세스의 흐름도이다. 도 1은 장면에서 객체들을 움직이거나 애니메이션하는 것을 처리하는 재투영 동안 구현될 수 있는 모션 평활화(smoothing) 기술을 예시한다.

도 2는 본 명세서에 개시된 실시형태에 따른, 호스트 컴퓨터 및 HMD의 각각의 렌더링 워크로드를 도시하는 2개의 예시적인 타임라인들을 예시하는 도면이다.

도 3은 HMD와 같은 디스플레이 상에 프레임들을 렌더링하기 위해 재투영이 사용되고 있을 때 장면에서 이동하거나 애니메이션되는 객체를 처리하기 위한 예시적인 모션 평활화 기술을 예시하는 다이어그램이다.

도 4는 예시적인 렌더 메쉬 및 재투영된 프레임들을 위한 모션 평활화에 렌더 메쉬가 사용될 수 있는 방법을 나타낸 도면이다.

도 5는 본원에 개시된 실시형태에 따른, 모션 평활화 기술의 일부로서 그래픽 처리 유닛(GPU: graphics

processing unit)에 의해 생성된 모션 벡터들을 사용하여 재투영된 프레임들을 생성하기 위한 예시적인 프로세스의 흐름도이다.

도 6은 본원에 개시된 실시형태에 따른, 픽셀 데이터가 모션 벡터 추정을 위해 GPU에 입력되기 전에, 이전에 렌더링된 프레임들의 픽셀 데이터를 정렬하기 위한 예시적인 프로세스의 흐름도이다.

도 7은 본원에 개시된 실시형태에 따른, 이전에 렌더링된 프레임들의 일부 픽셀 데이터를 배제하고, 픽셀 데이터의 잔여 부분을 모션 벡터 추정을 위해 GPU에 입력으로서 제공하기 위한 예시적인 프로세스의 흐름도이다.

도 8은 본원에 개시된 실시형태에 따른, 이전에 렌더링된 프레임들의 픽셀 데이터를 정렬하고, 이전에 렌더링된 프레임들의 픽셀 데이터의 패널 마스크 부분을 배제하고, 픽셀 데이터의 잔여 부분을 모션 벡터 추정을 위해 GPU에 입력으로서 제공하기 위한 예시적인 프로세스의 흐름도이다.

도 9은 본원에 개시된 실시형태에 따른, 모션 평활화 기술의 일부로서 재투영된 프레임을 생성하기 위해 사용되기 전에 GPU로부터 출력되는 모션 벡터들을 임계화하기 위한 예시적인 프로세스의 흐름도이다.

도 10은 본원에 개시된 실시형태에 따른, 모션 평활화 기술의 일부로서 재투영된 프레임을 생성하기 위해 사용되기 전에 GPU로부터 출력되는 모션 벡터들을 감쇠하기 위한 예시적인 프로세스의 흐름도이다.

도 11은 본원에 개시된 실시형태에 따른, 모션 평활화 기술의 일부로서 재투영된 프레임을 생성하기 위해 사용되기 전에 GPU로부터 출력되는 모션 벡터들을 감쇠하기 위해서 사용될 수 있는 감쇠 텍스처를 생성하기 위한 예시적인 프로세스의 흐름도이다.

도 12는 본원에 개시된 실시형태에 따른, 모션 평활화 기술의 일부로서 재투영된 프레임을 생성하기 위해 모션 벡터들의 결과 세트가 사용되기 전에 색상 변화가 거의 또는 전혀 없는 영역에 대응되는 모션 벡터들을 제로화(zeroing out)하기 위한 예시적인 프로세스의 흐름도이다.

도 13은 본원에 개시된 실시형태에 따른, 모션 평활화 기술의 일부로서 재투영된 프레임을 생성하기 위해 모션 벡터들의 결과 세트가 사용되기 전에 하나 또는 둘 이상의 필터들을 사용하여 모션 벡터 필드를 "클린업"하기 위한 예시적인 프로세스의 흐름도이다.

도 14는 본원에 개시된 실시형태에 따른, 픽셀 데이터가 모션 벡터 추정을 위해 GPU에 입력되기 전에 이전에 렌더링된 프레임들을 회전시키기 위한 예시적인 프로세스의 흐름도이다.

도 15는 본원에 개시된 실시형태에 따른, 이전에 렌더링된 프레임들의 크로마 데이터 및 루마 데이터에 기초하여 생성되는 모션 벡터들의 어레이들 사이에서 선택하기 위한 예시적인 프로세스의 흐름도이다.

도 16은 본원에 개시된 실시형태에 따른, 모션 벡터들의 다수 어레이들을 획득하고, 어레이들 간의 차이들을 결정하고, 모션 평활화를 위해 결정된 차이들에 기초하여 모션 벡터들의 최종 어레이를 생성하기 위한 예시적인 프로세스의 흐름도이다.

도 17은 본원에 개시된 실시형태에 따른, 이미지 영역의 상이한 부분들에 대해 상이한 해상도들에서 모션 벡터들의 다수의 어레이들을 획득하기 위한 예시적인 프로세스의 흐름도이다.

도 18a 및 도 18b는 본 명세서에 개시된 실시형태에 따른, HMD 및 호스트 컴퓨터를 활용하는 시스템의 2개의 대안적인 설정을 예시한다.

도 19는 본 명세서에 개시된 기술들이 구현될 수 있는 웨어러블 디바이스, 예컨대 VR 헤드셋의 예시적인 컴포넌트들을 예시한다.

도 20은 본원에 개시된 기술들이 구현될 수 있는 호스트 컴퓨터의 예시적인 컴포넌트들을 예시한다.

발명을 실시하기 위한 구체적인 내용

[0008] 본원에서는 무엇보다도 분산 시스템의 디스플레이 상에 프레임들을 렌더링할 때 장면에서 객체를 이동하거나 애니메이션하는 것을 처리하는 모션 평활화 기술들이 설명된다. 본원에 설명된 모션 평활화 기술들은, 이동하거나 애니메이션되는 객체에 대한 위에서 언급된 저더 아티팩트와 같은, 이동하거나 애니메이션되는 객체에 대한 원치 않는 시각적 아티팩트를 완화한다. 헤드 마운트 디스플레이(HMD)는, 디스플레이 상에 이미지들을 렌더링할 때 개시된 모션 평활화 기술들을 구현할 수 있는 예시적인 유형의 디스플레이, 또는 디스플레이 시스템이다. HMD는 사용자를 가상 현실(VR) 환경 또는 증강 현실(AR: augmented reality) 환경에 몰입시키기 위해 사용자에

의해 착용될 수 있다. HMD와 같은 디스플레이 또는 디스플레이 시스템은 호스트 컴퓨터와 분리되어 있지만, 통신 가능하게 결합될 수 있다. 애플리케이션(예: 비디오 게임)은 호스트 컴퓨터 상에서 실행되고, 일련의 프레임들의 개별 프레임들에 대한 픽셀 데이터를 생성한다. 픽셀 데이터는 프레임별로 HMD로 송신되고, HMD의 하나 또는 둘 이상의 디스플레이 패널들은 수신된 픽셀 데이터에 기초하여 이미지들을 렌더링한다. 이러한 이미지는 HMD에 포함된 광학기기를 통해 사용자에 의해 보여져서, 사용자가 VR 또는 AR 환경에 몰입된 것처럼 사용자가 이미지들을 인지하게 한다.

[0009] 언급된 바와 같이, HMD와 같은 디스플레이 시스템은 프레임 레이트에 도달하지 못하는 애플리케이션을 보상하기 위해 "재투영"이라는 기술을 활용할 수 있다. 예를 들어, 재투영된 프레임은 이상적인 프레임 레이트를 달성하기 위해 실제 프레임들 사이에서 렌더링될 수 있고, 각각의 재투영된 프레임은 호스트 컴퓨터 상에서 실행되는 애플리케이션에 의해서 출력된 최근 렌더링된 실제 프레임(예를 들어, 가장 최근에 렌더링된 실제 프레임)으로부터 픽셀 데이터를 사용하여 생성될 수 있다. 재투영된 프레임에서, 이전의 실제 프레임에서 렌더링된 장면은 사용자의 머리 회전을 처리하는 방식으로 (예를 들어, 회전 및 재투영 계산을 통해) 변환된다.

[0010] 프레임들 사이에서 이동하거나 애니메이션되는 객체들의 모션을 추가로 처리하는 방식으로 프레임들에 대한 픽셀 데이터를 수정하기 위해 분산 시스템에서 사용 가능할 수 있는 모션 평활화 기술들이 본원에 설명된다. 본원에 설명된 모션 평활화 기술들은 모션 벡터들의 형태로 다수 프레임들에 걸쳐 객체들의 모션(예를 들어, 방향 및 크기)을 추정하기 위해 컴퓨터 비전 알고리즘을 사용한다. 예를 들어, 모션 벡터들의 어레이는, 픽셀 데이터가 HMD와 같은 디스플레이 시스템에 전송되기 전에 픽셀 데이터를 압축한 결과로서(예를 들어, 부산물로서) 호스트 컴퓨터에 의해 생성될 수 있다. 따라서, 모션 벡터들의 어레이는 일련의 프레임들에서 적어도 프레임들의 일부에 대해 압축된 픽셀 데이터를 나타낼 수 있다. 예시적인 예에서, 호스트 컴퓨터 상의 그래픽 처리 장치(GPU)의 비디오 인코더(예: 비디오 인코드 칩)는, 주어진 프레임에 대해 압축된 픽셀 데이터를 나타내는 모션 벡터들의 어레이를 생성하기 위해 다수의 이전에 렌더링된 프레임들의 픽셀 데이터를 분석할 수 있다. 모션 벡터들은 HMD로 전송될 수 있고, 압축 해제된 픽셀 데이터를 획득하기 위해서 모션 벡터들을 사용하는 것뿐만 아니라, 모션 벡터들은 또한 객체들을 이동하거나 애니메이션하는 것을 처리하는 방식으로 압축 해제된 픽셀 데이터를 수정하기 위해서 HMD에 의해서 사용될 수 있다. 달리 말하면, 호스트 컴퓨터로부터 수신된 모션 벡터들은, 이동하는 객체의 저더가 완화되도록 렌더링될 (예: 재투영된) 프레임에서 객체가 위치되어야 하는 곳을 (이전에 렌더링된 프레임에서 객체의 모션으로부터) 외삽에 의해 추정하기 위해서 HMD 상에서 이용될 수 있다.

[0011] 예시적인 모션 평활화 프로세스에서, 연관된 픽셀 데이터는 호스트 컴퓨터 상에서 압축될 수 있다. 호스트 컴퓨터 상의 그래픽 처리 장치(GPU)의 비디오 인코더는 픽셀 데이터를 압축하기 위해서 사용될 수 있다. 픽셀 데이터를 압축한 결과로서, 비디오 인코더는 압축된 픽셀 데이터를 나타내는 모션 벡터들의 어레이를 생성할 수 있다. 호스트 컴퓨터는 모션 벡터들의 어레이를 HMD와 같은 디스플레이 시스템에 송신할 수 있다. 일부 실시형태에서, 호스트 컴퓨터는 HMD에 무선으로 결합될 수 있지만, 유선 연결도 본원에서 고려된다. HMD는 모션 벡터들의 어레이를 호스트 컴퓨터로부터 수신할 수 있다. HMD는 픽셀 데이터(또는, 압축 해제 또는 디코딩 알고리즘을 사용하여 픽셀 데이터의 근사치)를 획득하기 위해서, 압축된 제1 픽셀 데이터를 압축 해제할 수 있다. HMD는 수정된 픽셀 데이터를 획득하기 위해 호스트 컴퓨터로부터 수신된 모션 벡터들의 어레이에 적어도 부분적으로 기초하여 픽셀 데이터를 추가로 수정할 수 있다. 이러한 수정된 픽셀 데이터는 장면에서 이동하거나 애니메이션되는 객체들에 대한 임의의 저더를 제거(그렇지 않으며, 완화)하기 위해 "모션 평활화"된다. 다음으로, 모션-평활화된 프레임은 수정된 픽셀 데이터에 적어도 부분적으로 기초하여 HMD의 디스플레이 패널(들) 상에 이미지를 제시함으로써 렌더링될 수 있다.

[0012] 본원에 설명된 모션 평활화 기술들은, 프레임이 누락된 경우에도, 객체가 예상된 방식으로 장면 내에서 움직이거나 애니메이션되는 보다 사실적이고 충실도가 높은 보기 경험을 제공한다. 언급된 바와 같이, 애플리케이션이 프레임 레이트에 도달하지 못하거나 네트워크 정체가 발생되거나, 또는 가능하게는 다른 이유로 인해 프레임이 누락될 수 있다. 따라서, 본원에 설명된 모션 평활화 기술들은, 프레임들이 때때로 누락될 수 있는 경우 분산 시스템에서 충실도가 높은 사용자 경험을 유지함으로써 이러한 발생을 보상한다. 모션 평활화 기술들은 또한, 프레임들이 누락되지 않은(즉, 프레임들이 HMD에서 적시에 수신되는) 라이브 렌더링 시나리오에서도, 프레임들의 전송에서 내재하는 레이턴시를 보상한다. 또한, 무선 구현예에서, 본원에 설명된 기술들은, 무선 HMD에 더 많은 컴포넌트들을 추가할 필요 없이, 무선 HMD에서 모션 평활화 보정을 가능하게 하며, 이로써, 오늘날 시장에 나와 있는 많은 올인원(또는 독립형) 헤드셋과 비교하여, 너무 뜨거워지지 않고, 착용이 더 편안한 더 경량인 헤드셋을 허용한다. 또한, 본원에 설명된 기술들 및 시스템들은, 데이터를 압축하고, 호스트 컴퓨터로부터 HMD로 전송하기 위해 이미 생성된 모션 벡터들을 활용할 수 있으며, 이는 HMD 상에서 재투영된 프레임들의 모션 평

활화를 위해 모션 벡터들을 사용하는 데 추가 전송 비용이 없다는 것을 의미한다.

[0013] 호스트 컴퓨터로부터 수신된 모션 벡터들의 어레이는 미래 프레임(예: 재투영된 프레임)을 외삽에 의해 추정하기 위해서 HMD에 의해서 사용될 수 있으며, 이는 수정된 제2 픽셀 데이터가, 제1 픽셀 데이터를 프레임 버퍼로 출력한 후, HMD 상의 프레임 버퍼에 출력되기 때문이라는 것이 이해될 것이다. 이러한 방식으로, 본 명세서에 설명된 실시예들에서, 제1 프레임은 재투영된 프레임 전에 렌더링된다. 이러한 외삽 기술은 프레임들 간 보간과 대조될 수 있고, 본원에 설명된 기술들 및 시스템들이, 프레임들 간 보간을 위해 모션 벡터들을 사용하는 것과 대조적으로, 미래 프레임을 외삽에 의해 추정하기 위해 호스트 컴퓨터로부터 수신된 모션 벡터들을 사용하는 것과 관련되는 것이 이해될 것이다.

[0014] 또한, 본원에서는 본원에 개시된 기술 및 프로세스를 구현하도록 구성된 디스플레이 시스템(예를 들어, HMD)을 포함하는 시스템과 같은 시스템뿐만 아니라, 본원에 개시된 기술 및 프로세스를 구현하기 위한 컴퓨터 실행 가능 명령을 저장하는 비일시적 컴퓨터 판독 가능 매체를 개시한다. 본원에 개시된 기법 및 시스템이, 예로서, 비디오 게임 애플리케이션 및 구체적으로는 VR 게이밍 애플리케이션의 맥락에서 논의되지만, 본원에 설명되는 기법 및 시스템이, 비-VR 애플리케이션(예를 들어, AR 애플리케이션), 및/또는 비-게이밍 애플리케이션, 예컨대 산업 기계 애플리케이션, 방위 애플리케이션, 로봇 애플리케이션 등을 포함하지만, 이에 한정되지 않는, 다른 애플리케이션에 장점을 제공할 수 있다는 점이 이해될 것이다. 또한, HMD가 이미지를 표시하는 디스플레이 시스템의 예로 제공되지만, 호스트 컴퓨터로부터 비디오 콘텐츠를 스트리밍하는 핸드헬드 디스플레이 디바이스, 및 상대적으로 큰 벽걸이형 또는 빌보드 디스플레이 시스템 등과 같은 다른 유형들의 디스플레이 시스템들이 본원에 설명된 모션 평활화 기술들로부터 이익을 얻을 수 있다는 것이 이해될 것이다.

[0015] 본원에 설명되는 방법은 하드웨어, 소프트웨어, 펌웨어, 또는 이들의 조합(즉, 로직)으로 구현될 수 있는 동작들의 시퀀스를 표현하는 논리적 흐름 그래프의 블록들의 모음으로서 예시된다. 소프트웨어의 경우, 블록은, 하나 또는 둘 이상의 프로세서들에 의해 실행될 때 언급된 작업을 수행하는 컴퓨터 실행 가능 명령어들을 나타낸다. 일반적으로, 컴퓨터 실행 가능 명령어는, 특정 기능들을 수행하거나 특정된 추상적 데이터 유형을 구현하는 루틴, 프로그램, 오브젝트, 컴포넌트, 데이터 구조 등을 포함한다. 작동이 설명되는 순서는 제한으로서 해석되게 하려는 것이 아니고, 임의의 수의 설명된 블록들은 임의의 순서로 그리고/또는 병렬로 결합되어서 프로세스를 구현할 수 있다.

[0016] 도 1은, 본원에 개시된 실시형태에 따라, HMD(102)에 통신 가능하게 결합된 호스트 컴퓨터(106) 상에서 실행되는 애플리케이션에 의해 프레임들이 출력될 때 헤드 마운트 디스플레이(HMD)(102)(사용자(104)에 의해서 착용됨) 상에 프레임들을 렌더링하기 위한 예시적인 프로세스(100)의 흐름도이다. 도 1은 장면에서 객체들을 움직이거나 애니메이션하는 것을 처리하는 재투영 동안 구현될 수 있는 모션 평활화 기술을 예시한다.

[0017] 도 1의 상부에서, HMD(102)는 사용자(104)에 의해서 착용된 것으로서 도시되고, 호스트 컴퓨터(106)는 HMD(102)에 통신 가능하게 결합된 것으로서 도시된다. 예를 들어, 호스트 컴퓨터(106)는, 개인용 컴퓨터(PC: personal computer), 랩탑 컴퓨터, 데스크탑 컴퓨터, 개인용 정보 단말기(PDA: portable digital assistant), 모바일 폰, 태블릿 컴퓨터, 셋톱 박스, 게임 콘솔, 서버 컴퓨터, 웨어러블 컴퓨터(예를 들어, 스마트 워치 등) 또는 데이터를 송신/수신할 수 있는 기타 전자 장치를 이에 한정됨 없이 포함하는 임의의 유형의 컴퓨팅 디바이스 및/또는 임의의 수의 컴퓨팅 디바이스로서 구현될 수 있다.

[0018] 호스트 컴퓨터(106) 및 HMD(102)는 애플리케이션(예를 들어, 비디오 게임)을 실행하고 디스플레이 상에 연관된 이미지를 렌더링하기 위한 분산 시스템을 집합적으로 나타낸다. 일부 실시형태에서, 호스트 컴퓨터(106)는 HMD(102)를 착용한 사용자(104)의 가정과 같은, HMD(102)와 동일한 환경에 콜로케이팅될 수 있다. 대안적으로, 호스트 컴퓨터(106)는, HMD(102)의 지리적 위치에 대해 원격 지리적 위치에 위치되는 서버 컴퓨터 형태의 호스트 컴퓨터(106)와 같이, HMD(102)에 대해 원격으로 위치될 수 있다. 원격 호스트 컴퓨터(106) 구현예에서, 호스트 컴퓨터(106)는 인터넷과 같은 광역 네트워크를 통해 HMD(102)에 통신 가능하게 커플링될 수 있다. 로컬 호스트 컴퓨터(106) 구현예에서, 호스트 컴퓨터(106)는 HMD(102)가 있는 환경(예를 들어, 가정)에 콜로케이팅될 수 있고, 이에 의해 호스트 컴퓨터(106) 및 HMD(102)는 직접적으로 또는 중간 네트워크 디바이스를 통해 근거리 통신망(LAN: local area network)을 통해 함께 통신 가능하게 커플링될 수 있다.

[0019] HMD(102)와 호스트 컴퓨터(106)는 함께 통신 가능하게 연결됨으로써, 대응되는 이미지(들)를 HMD(102)의 디스플레이 패널(들) 상에 제시하기 위해서 사용되는 픽셀 데이터를 생성하여 주어진 프레임을 렌더링하기 위해 협력 방식으로 함께 작동하도록 구성된다. 호스트 컴퓨터(106) 및 HMD(102)는 무선으로 그리고/또는 유선 연결을 통해 함께 통신 가능하게 결합될 수 있다. 예를 들어, 디바이스들(102/106)은 Wi-Fi, 블루투스, 무선 주파수(RF),

및/또는 임의의 다른 적절한 무선 프로토콜을 사용하여 데이터를 교환할 수 있다. 추가적으로 또는 대안적으로, 디바이스들(102/106)은 그들 사이의 데이터 전달을 위한 유선 연결을 용이하게 하기 위해 하나 또는 둘 이상의 물리적 포트들을 포함할 수 있다.

[0020] HMD(102)가 개시된 모션 평활화 기술을 구현할 수 있는 예시적인 "디스플레이 시스템"으로서 본원에 제시되지만, 다른 유형 및/또는 구현의 "디스플레이 시스템"이 본원에 설명된 모션 평활화 기술을 구현할 수 있다는 것이 이해될 것이다. 따라서, HMD가 단지 본원에서 설명되는 기술을 구현하기 위한 예시적인 유형의 디스플레이라는 것이 이해된다고 하더라도, 본원에서 "HMD"에 대한 어떠한 참조도 "디스플레이"또는 "디스플레이 시스템"이라는 용어로 대체될 수 있다는 것이 이해될 것이다. 일부 실시예에서, HMD(102)는 VR 시스템에서 사용되기 위한, 예컨대 VR 게이밍 시스템과 함께 사용되기 위한 VR 헤드셋을 나타낼 수 있다. 그러나, HMD(102)는, 추가적으로 또는 대안적으로, AR 애플리케이션에서 사용되기 위한 AR 헤드셋, 또는 게임과 관련되지 않은 VR 및/또는 AR 애플리케이션(예를 들어, 산업용 애플리케이션)을 위해 사용 가능한 헤드셋으로서 구현될 수 있다. AR에서, 사용자(104)는 현실 세계 환경 상에 오버레이된 가상 객체를 보는 반면, VR에서, 사용자(104)는 전형적으로 현실 세계 환경을 보지 않고, HMD(102)의 디스플레이 패널(들) 및 광학기기(예를 들어, 렌즈)를 통해 지각되는 바와 같은 가상 환경에 완전히 몰입된다. 일부 VR 시스템에서, 사용자(104)의 현실 세계 환경의 패스 쓰루(pass-through) 이미지가 가상 이미지와 함께 디스플레이되어 VR 시스템에서 증강 VR 환경을 생성할 수 있으며, 이에 의해 VR 환경은 현실 세계 이미지로 증강된다(예를 들어, 가상 세계에 오버레이됨)는 것이 이해될 것이다. 본원에 설명되는 실시예가 주로 VR-기반 HMD(102)에 관한 것이지만, HMD(102)가 VR 애플리케이션에서의 구현예에 한정되지 않는다는 것이 이해될 것이다.

[0021] 또한, HMD(102)는 단일의 디스플레이 패널 또는 다수의 디스플레이 패널, 예를 들어, 스테레오 쌍의 디스플레이 패널의 좌측 디스플레이 패널 및 우측 디스플레이 패널을 포함할 수 있다. HMD(102)의 하나 또는 둘 이상의 디스플레이 패널들은 HMD(102)를 착용한 사용자(104)에 의해서 가시적인 일련의 이미지 프레임들(본원에서, "프레임들"로 지칭됨)을 제시하기 위해서 사용될 수 있다. HMD(102)가 임의의 수의 디스플레이 패널(예를 들어, 2개 초과의 디스플레이 패널, 한 쌍의 디스플레이 패널, 또는 단일 디스플레이 패널)을 포함할 수 있다는 것이 이해될 것이다. 따라서, 본원에서 단수로 사용되는 바와 같은 용어 "디스플레이 패널"은 2-패널 HMD(102)의 한 쌍의 디스플레이 패널들 중 어느 하나의 디스플레이 패널을 지칭할 수 있거나, 또는 디스플레이 패널은 임의의 수의 디스플레이 패널들을 갖는 HMD(102)(예를 들어, 단일-패널 HMD(102) 또는 다중-패널 HMD(102))의 단일 디스플레이 패널을 지칭할 수 있다. 2-패널 HMD(102)에서, 스테레오 프레임 버퍼는, 예를 들어, HMD(102)의 양 디스플레이 패널들 상에서 2160 x 1200 픽셀(예를 들어, 디스플레이 패널당 1080 x 1200 픽셀)을 렌더링할 수 있다.

[0022] 또한, HMD(102)의 디스플레이 패널(들)은 임의의 적합한 유형의 디스플레이 기술, 예컨대, 디스플레이 패널(들) 상에서의 프레임들의 제시 동안 광을 방출하기 위해 발광 요소들(예를 들어, 발광 다이오드(LED: light emitting diode)들)을 활용하는 발광 디스플레이를 활용할 수 있다. 일 예로서, HMD(102)의 디스플레이 패널(들)은 액정 디스플레이(LCD: liquid crystal display), 유기 발광 다이오드(OLED: organic light emitting diode) 디스플레이, 무기 발광 다이오드(ILED: inorganic light emitting diode) 디스플레이, 또는 HMD 응용분야를 위한 임의의 다른 적합한 유형의 디스플레이 기술을 포함할 수 있다.

[0023] HMD(102)의 디스플레이 패널(들)은 90 헤르츠(Hz) 재생률과 같은 임의의 적절한 재생률에서 작동될 수 있으며, 이 재생률은 고정 재생률 또는 일 재생률 범위에 걸쳐 동적으로 변하는 가변 재생률일 수 있다. 디스플레이의 "재생률"은 디스플레이가 스크린을 리드로잉(redraw)하는 초당 횟수이다. 초당 디스플레이되는 프레임들의 수는, 고정 재생률을 사용하는 경우, 디스플레이의 재생률에 의해 제한될 수 있다. 따라서, 일련의 프레임들은, 모든 스크린 리프레시에 따라 일련의 프레임들 중 단일 프레임이 디스플레이되도록 디스플레이 상에 이미지들로서 프로세싱(예를 들어, 렌더링)되고 디스플레이될 수 있다. 즉, HMD(102) 상에 일련의 이미지들을 제시하기 위해, HMD(102)의 디스플레이 패널(들)은, 모든 스크린 리프레시에서 픽셀들을 조명하면서, 일련의 프레임들에서 디스플레이의 재생률로 프레임으로부터 프레임으로 전이될 수 있다. 일부 실시예에서, 프레임 레이트는 스로틀링될 수 있고/있거나, 애플리케이션이 타겟 프레임 레이트에 도달하는 데 실패할 수 있고/있거나, 네트워크 혼잡은 데이터 전송에 레이턴시를 도입할 수 있다. 이러한 시나리오에서, 재투영된 프레임들(본원에서 때때로 "팬텀 프레임(phantom frame)들"로서 지칭됨)은 애플리케이션-렌더링된 프레임들(본원에서 때때로 "실제 프레임들"로서 지칭됨) 사이에 삽입될 수 있다.

[0024] 일반적으로, 호스트 컴퓨터(106) 상에서 실행되는 애플리케이션(들)은 그래픽 기반 애플리케이션(들)(예를 들어, 비디오 게임)일 수 있다. 애플리케이션(들)은, HMD(102)의 디스플레이 패널(들) 상에 이미지를 제시하기 위해서 사용될 수 있는 일련의 프레임들을 출력하도록 구성된다. 예를 들어, 애플리케이션(들)은 일련의 프레임

들에 대한 픽셀 데이터를 생성할 수 있고, 픽셀 데이터는 HMD(102)의 디스플레이 패널(들) 상에 대응되는 이미지들을 제시하기 위해서 사용될 수 있다. 일부 실시형태에서, 프레임들이 HMD(100)의 디스플레이 패널(들) 상에 렌더링되기 전에 타겟에 렌더링될 수 있도록 오프 스크린(off-screen) 렌더링이 활용된다. 이와 같이, 본원에서 사용되는 "렌더링"은 디스플레이 자체 상에 렌더링하기 전에 디스플레이 이외의 타겟에 대해 렌더링하는 것 및/또는 디스플레이 상에 렌더링하는 것(예: 다른 타겟에 대한 오프 스크린 렌더링 없이 또는 이후)을 포함할 수 있다.

[0025] 이제 프로세스(100)를 참조하면, 112에서, 호스트 컴퓨터(106)의 로직은 호스트 컴퓨터(106) 상에서 실행되는 애플리케이션으로부터 일련의 프레임들 중 제1 프레임에 대한 픽셀 데이터를 획득(또는 수신)할 수 있다. 픽셀 데이터는, HMD(102)의 디스플레이 패널(들)의 픽셀의 어레이 내 개별 픽셀에 대한 픽셀 값을 포함할 수 있다. 실행 애플리케이션으로부터 획득된 픽셀 데이터는, 일부 실시형태에서, 픽셀당 값(예를 들어, 컬러 값)의 2차원 어레이를 포함할 수 있다. 일부 실시형태에서, 픽셀 데이터는 깊이 값, 휘도 값 등과 같은 추가 데이터 또는 메타데이터를 더 포함한다. 일부 실시형태에서, 픽셀 데이터는, 컬러 및 알파 값들(예를 들어, 적색 채널에 대한 하나의 컬러 값, 녹색 채널에 대한 하나의 컬러 값, 청색 채널에 대한 하나의 컬러 값, 및 하나 또는 둘 이상의 알파 채널들에 대한 하나 또는 둘 이상의 값들)의 단일 세트에 의해 표현되는 각각의 픽셀에 대한 데이터를 포함할 수 있다.

[0026] 일부 실시형태에서, 호스트 컴퓨터(106)는 먼저 HMD(102)로부터 헤드 트래킹 데이터를 수신할 수 있으며, 이는 프레임에 대한 픽셀 데이터를 생성하기 위한 HMD(102)의 예측된 포즈를 결정하기 위해서 사용될 수 있다. 즉, 호스트 컴퓨터(106)는 HMD(102)로부터 수신한 머리 추적 데이터를 사용하여, HMD(102)의 디스플레이 패널(들)의 발광 소자가 애플리케이션에 의해 렌더링될 프레임을 위해 비춰질 시간에 HMD(102)가 있게 될 예측 포즈를 나타내는 포즈 데이터를 생성할 수 있다. 예를 들어, HMD(102)의 머리 추적 시스템은 HMD(102)의 최대 6 자유도(예를 들어, 3차원(3D) 위치, 롤링, 피칭, 및 요잉)를 추적하도록 구성될 수 있으며, 이는 HMD(102)의 예측 포즈를 결정하기 위해(예를 들어, HMD(102)의 미래 포즈로 귀결되는 예측된 머리 움직임을 처리) 호스트 컴퓨터(106)에 머리 추적 데이터로서 전송될 수 있다. 따라서, 예측 포즈를 나타내는 포즈 데이터는 프레임을 렌더링하기 위해 애플리케이션에 입력으로서 제공될 수 있고, 애플리케이션은 포즈 데이터에 기초하여 픽셀 데이터를 출력할 수 있다. 예를 들어, 애플리케이션은 포즈 데이터를 수신하는 함수를 호출할 수 있고, 요청된 포즈 데이터(프레임에 대한 타겟 조명 시간으로 예측됨)는, 애플리케이션이 포즈 데이터에 따라 프레임을 렌더링할 수 있도록 애플리케이션에 제공될 수 있으며, 이 프레임은 장면을 렌더링하기 위해서 사용되는 가상 카메라 포즈에 대응된다.

[0027] 일부 실시형태에서, 애플리케이션은 프레임에 대한 깊이 데이터(예를 들어, Z-버퍼 데이터) 및/또는 추가 픽셀 데이터(때때로 본원에서 "범위를 벗어난 픽셀 데이터" 또는 "추가 픽셀 데이터"로서 지칭됨)를 생성할 것을 추가로 지시받을 수 있고, 이에 응답하여, 블록 112에서, 호스트 컴퓨터(106)의 로직은 위에서 언급된 픽셀 데이터에 추가하여 프레임과 연관된 깊이 데이터 및/또는 여분의 픽셀 데이터를 애플리케이션으로부터 획득할 수 있다. 예를 들어, 애플리케이션에 의해 출력되는 깊이 버퍼(또는 Z 버퍼)로부터의 깊이 데이터는 장면에서 가려진 객체를 나타낼 수 있다. 따라서, 깊이 데이터는, 무엇보다도, 장면에 있는 객체의 시차(parallax)를 조정하기 위해서 사용될 수 있다(예를 들어, 세계 공간(world space)에서 멀리 떨어져 있는 배는 머리 움직임으로, 근접한 객체가 동일한 머리 움직임으로 움직이는 것만큼 많이 움직이지 않을 수 있다). 장면의 가상 객체에 대응되는 픽셀의 깊이를 아는 것은 HMD(102) 상의 재투영 동안 이러한 시차를 조정하는 방법을 아는 데 도움이 된다. 깊이 정보를 아는 것은 또한, HMD(102)의 회전에 기초할 뿐만 아니라 공간에서 HMD(102)의 병진에 기초하여 장면을 워핑(warp)하는 것을 허용한다. 깊이 데이터가 없으면, 시스템은 약 2 미터의 평균 깊이를 취할 수 있다. 그러나, 장면의 평균 깊이에 대한 더 양호한 추정을 제공하는 임의의 적절한 깊이 해상도는 HMD(102) 상에 재투영을 수행할 때 유리할 수 있고, 사용 가능한 대역폭에 포함될 수 있는 깊이 데이터의 해상도가 높을수록, HMD(102) 상에 대해 더 양호한 재투영 조정이 이루어질 수 있다. 그러나, 저해상도 깊이 데이터를 전송하는 것에는 더 낮은 레이턴시와 같은 이점이 있을 수 있다. 깊이 데이터의 해상도가 HMD(102)에 대한 재투영 조정을 개선하기 위해 적합하기만 하다면, 깊이 데이터를 포함하는 것이 유리할 수 있다.

[0028] 애플리케이션에 의해 출력되는 여분의 픽셀 데이터는 HMD(102)의 디스플레이 패널(들)의 픽셀 어레이의 경계 외부에 있는 여분의 픽셀 값을 포함할 수 있다. 예를 들어, HMD(102)의 디스플레이 패널(들)이 2160 x 1200 픽셀의 어레이를 갖는 경우, 블록 112에서 획득된 픽셀 데이터는 2160 x 1200의 픽셀의 어레이의 픽셀 값들에 대응될 수 있는 반면, 추가 픽셀 데이터는 2160 x 1200 어레이의 경계 외부에 있는 픽셀들에 대응될 수 있다. 따라서, 픽셀 데이터 및 여분의 픽셀 데이터는, 예를 들어, 2400 x 1400 픽셀의 더 큰 어레이와 같은 더 많은 수의 픽셀 값을 구성할 수 있다. 타겟보다 큰 뷰에서 렌더링하는 능력은 시야(FOV)의 어떠한 유효한 감소 없이 디스

플레이 영역의 주변에 패널 마스크를 적용하는 것을 허용할 수 있다.

[0029] 114에서, 호스트 컴퓨터(106)의 로직은 블록 112에서 애플리케이션으로부터 획득된 픽셀 데이터를 압축할 수 있다. 블록 114에서 픽셀 데이터를 압축하는 예시적인 이유는 호스트 컴퓨터(106)와 HMD(102) 사이의 데이터 연결의 대역폭 제한 때문이다. 예를 들어, 원하는 프레임 레이트(예: 90 Hz 이상의 프레임 레이트)로 프레임을 표시하기 위해서 할당된 시간의 양 내에 비압축 픽셀 데이터(예: 파일)를 송신하기에 충분한 대역폭이 없는 경우가 많다. 압축이 종단간(end-to-end) 그래픽 파이프라인에 약간의 레이턴시를 도입하나(픽셀 데이터를 압축하지 않는 것과 비교하여), 픽셀 데이터를 압축하기 위한 이 추가 시간은 포즈 예측에서 처리될 수 있으며, 압축을 위해 도입된 레이턴시를 처리하기 위해 조정이 HMD(102) 상에서 이루어질 수 있다.

[0030] 일부 실시형태에서, 블록 114에서 수행되는 압축은 픽셀 데이터의 포맷을 변경하는 것을 포함한다. 예를 들어, 블록 114에서의 압축은 HEVC(High Efficiency Video Coding) 및/또는, 종종 H.265 및/또는 MPEG-H 파트 2라고 지칭되는 HEVC의 확장과 같은 비디오 압축 표준을 활용할 수 있다. HEVC는 블록 114에서 활용될 수 있는 비디오 압축 표준의 일례이지만, 블록114에서 픽셀 데이터를 압축하기 위해 다른 적절한 비디오 압축 표준이 활용될 수 있다는 것이 이해될 것이다. HEVC는, 대역폭 제약이 있는 시스템에서 압축되지 않은 픽셀 데이터보다 적은 데이터를 보내는 것을 목표로 픽셀 데이터를 압축하기 위해 모션 추정을 활용하나, 압축된 데이터는 수신단(예: HMD(102))에서 압축되지 않은 원본 데이터를 근사화하기에 충분하다. 블록 114에서 압축 작동에 활용되는 모션 추정의 일부로서, 모션 벡터의 어레이가 생성될 수 있으며, 이는 서브블록 116에 표시된다. 일반적으로, "모션 벡터"는 방향에 대한 X 및 Y 성분 및 크기(통상적으로 2D 화살표의 길이로 표시됨)를 갖는 2차원(2D) 화살표이다. 모션 벡터의 크기는 X 및 Y 성분 방향 모두에서 픽셀 수와 같은 임의의 적절한 측정 단위로 특정될 수 있다. 일부 실시예에서, 호스트 컴퓨터(106)의 GPU의 비디오 인코더(예를 들어, 비디오 인코드 칩)는 블록 112에서 획득된 제1 픽셀 데이터에 기초하고, 하나 또는 둘 이상의 다른 프레임들(예: 이전에 렌더링된 프레임)의 픽셀 데이터에 기초하여, 서브블록 116에서 모션 벡터들의 어레이를 생성할 수 있다. 서브-블록 116에서 모션 벡터들의 어레이를 생성하기 위해, 비디오 인코더는 압축 알고리즘에 따라, 분석해야 할 각각의 프레임의 픽셀 데이터 사이에서 픽셀 값(예를 들어, 휘도 값) 마다 비교할 수 있다. 추가적으로 또는 대안적으로, 비디오 인코더는 압축 알고리즘에 따라, 분석해야 할 각각의 프레임의 픽셀 데이터 사이에서 매크로블록들(예를 들어, 16픽셀의 블록(즉, 4 x 4픽셀 매크로블록), 64픽셀의 블록(즉, 8 x 8픽셀 매크로블록), 4096픽셀의 블록(즉, 64 x 64픽셀 매크로블록 등))을 비교할 수 있다. 호스트 컴퓨터(106)의 GPU는, 압축 작동(들)의 일부로서, 임의의 적절한 해상도에서 한 쌍의 프레임들 사이의 픽셀 데이터의 부분들을 비교할 수 있다.

[0031] HEVC를 포함하여 많은 비디오 압축 알고리즘에는, I-프레임, P-프레임 및 B-프레임과 같은 다양한 유형의 프레임이 있을 수 있다. I-프레임들은, 이들이 압축 해제 동안 프레임을 재구성하기에 충분한 데이터를 포함한다는 점에서, 독립적일 수 있다. 이 때문에, I-프레임들은, P-프레임들 및 B-프레임들에 비해, 상대적으로 많은 양의 데이터를 포함하고 가장 압축 가능하지 않다. P-프레임들은 이전 프레임들(예: 참조 프레임)로부터의 데이터를 사용하여 원본 픽셀 데이터를 압축하고, 다음으로, 압축 해제하고, 복원한다. P-프레임들은, P-프레임이 이전 프레임과의 차이를, 예를 들어, 모션 벡터들의 형태로 인코딩하기 때문에, I-프레임들에 비해 상대적으로 더 작다. B-프레임의 "B"는 양방향을 의미하며, 이는 B-프레임이 선행 프레임 및 퓨쳐 프레임을 모두 사용하여 원본 픽셀 데이터를 압축하고, 다음으로 압축 해제하고, 복원한다는 것을 의미한다. I-프레임들이 상대적으로 크기 때문에, I-프레임들을 전송하는 인스턴스들은 본 개시내용에서 감소되거나 최소화될 수 있다. 그러나, 일부 시나리오에서, 예를 들어, 호스트 컴퓨터(106)와 HMD(102) 사이의 무선 연결이 일시적으로 손실되고, 후속적으로 재설정되는 경우, I-프레임을 전송하는 것이 유리할 수 있다. 어느 경우든, 일반적인 일련의 프레임들은 일 I 프레임으로 시작되고 그 뒤에 일련의 P 프레임들이 올 수 있다. 블록 114에서 P-프레임들의 압축은, 이미지의 블록들이 참조 프레임에서 이들의 위치에 대해 새로운 프레임에서 이동된 위치의 관점에서 참조 프레임(예를 들어, I-프레임, 및/또는 선행 P-프레임 등)과의 차이를 인코딩하는 것을 포함할 수 있다. 이러한 차이들은 서브-블록 116에서 생성된 모션 벡터들에 의해 표현될 수 있다.

[0032] 또한, 114에서, 도 6의 프로세스(600)를 참조하여 더 상세히 설명되는 바와 같이, 호스트 컴퓨터(106)의 로직은 압축을 위해서 사용된 프레임들에 대한 픽셀 데이터가 호스트 컴퓨터(106)의 GPU에 대한 입력으로서 제공되기 전에 이러한 프레임들을 정렬(또는 달리 수정)할 수 있다. 이것은 프레임들 사이의 HMD(102)의 움직임으로 인해 수행될 수 있으며, 이는 객체(정지된 객체 및 움직이는 객체 모두)로 하여금 다수의 연속적인 프레임들을 렌더링하는 과정에 걸쳐 위치들 사이를 이동하게 할 수 있다. 한 프레임을 다른 프레임과 정렬하거나 이 반대로 함으로써, 또는 두 프레임들을 서로 정렬하기 위해 이들을 조정함으로써, 장면 내의 특정 정적 객체들을 나타내는 픽셀 데이터는 두 프레임들 사이에서 일반적으로 동일한 위치로 이동될 수 있어, 정적 객체들에 대응되는 픽셀

데이터가 GPU의 비디오 인코더에 의해 움직이는 객체들로 오인되지 않는다. 블록 114에서 수행된 정렬은, 헤드로 움직임으로 인한 모션 벡터의 델타(또는 차이)를 감소시키기 위해 프레임들 중 하나 또는 둘 모두에 대한(예를 들어, 프레임(들)의 픽셀 데이터에 대한) 회전 재투영 수정을 포함할 수 있다. 모션 벡터들의 델타들(또는 차이들)의 이러한 감소는 본원에 설명된 바와 같이, 호스트 컴퓨터(106)로부터 HMD(102)로 모션 벡터들을 전송할 때 감소된 대역폭의 추가 이점을 제공할 수 있다. 따라서, 수정된 픽셀 데이터는, 서브-블록 116에서 모션 벡터들을 생성할 때, 블록 114에서 압축을 위해 GPU에 대한 입력으로서 제공될 수 있다.

[0033] 118에서, 호스트 컴퓨터(106)는 압축된 픽셀 데이터(110(1))를 포함할 수 있는 데이터(110)를 HMD(102)에 송신할 수 있다. P-프레임들과 관련하여 위에서 언급된 바와 같이, 대부분의 시간에, 압축된 픽셀 데이터(110(1))는 서브-블록 116에서 생성된 모션 벡터들(110(2))의 어레이의 형태로 전송될 수 있다. 즉, 일련의 P-프레임들에 대해, 모션 벡터들(110(2))의 어레이는 압축된 픽셀 데이터를 나타낼 수 있다. 그러나, I-프레임이 HMD(102)로 송신되는 경우에, 압축된 픽셀 데이터(110(1))는 모션 벡터들을 포함하지 않을 수 있다. 블록 118 에서 송신되는 데이터(110)는 프레임에 대한 픽셀 데이터(110(1))를 생성하기 위해 애플리케이션에 의해 사용되는 포즈 데이터(110(3))를 더 포함할 수 있다. 일부 실시형태에서, 블록 118에서 송신되는 데이터(110)는 포즈 데이터(110(3))를 생략할 수 있고/있거나, 데이터(110)는 깊이 데이터(110(4)), 여분의 픽셀 데이터(예를 들어, HMD(102)의 디스플레이 패널(들)의 경계 외부), 시차 폐색 데이터, 및/또는 /또는 큐브 맵 데이터(예를 들어, 빠른 대규모 헤드 움직임에 대해 HMD(102)가 어떠한 데이터를 갖지 않는 어두운 픽셀을 제시하는 것 외에 다른 옵션을 갖도록)를 포함할 수 있다. 데이터(110)는 블록 118에서 HMD(102)로 구현에 따라 다양한 방식으로, 예를 들어, 무선으로, 유선 연결을 통해, 광역 네트워크 등을 통해 송신될 수 있다. 일부 실시형태에서, 데이터(110)의 일부 또는 전부는, 인코딩된 데이터 스트림의 일부로서 모션 벡터들(110(2))과 같은 압축된 픽셀 데이터(110(1))와 함께 대역내 또는 대역외로 송신될 수 있다. 예를 들어, 포즈 데이터(110(3)), 깊이 데이터(110(4)), 및/또는 추가 데이터는 모션 벡터들(110(2))과 같은 압축된 픽셀 데이터(110(1))로부터 대역외로 송신될 수 있다.

[0034] 120에서, HMD(102)는 호스트 컴퓨터(106)로부터, 모션 벡터들(110(2))(수신 시 압축된 픽셀 데이터를 나타냄), 포즈 데이터(110(3)) 및/또는 깊이 데이터(110(4)) 등과 같은 압축된 픽셀 데이터(110(1))를 포함하지만 이에 한정되지 않는 데이터(110)를 수신할 수 있다.

[0035] 122에서, HMD(102)의 로직(예: 소프트웨어, 하드웨어 및/또는 펌웨어 등)은, 예를 들어, 모션 벡터들(110(2)) 및 참조 프레임(예: 이전 프레임의 픽셀 데이터 110(1))을 사용함으로써 압축된 픽셀 데이터(110(1))를 압축 해제하여, 애플리케이션에 의해 출력된 프레임에 대한 픽셀 데이터, 또는 적어도 원본 픽셀 데이터의 근사치를 얻을 수 있다. 블록 122에서 압축된 픽셀 데이터를 압축 해제하는 것은 HEVC 알고리즘과 같은 압축 해제 또는 디코딩 알고리즘을 사용할 수 있다. 또한, 서브블록 124에 의해 도시된 바와 같이, 예를 들어, 적어도, 모션 벡터들(110(2))이 (예를 들어, P-프레임들에 대한) 압축된 픽셀 데이터(110(1))를 나타내는 경우, HMD(102)의 로직은 인코딩된 데이터 스트림(예를 들어, HEVC 스트림)과 같은 데이터 스트림으로부터 모션 벡터들(110(2))을 추출할 수 있다. 압축 해제된 픽셀 데이터(110(1)), 모션 벡터들(110(2)), 및 가능하게는 추가 데이터(예를 들어, 포즈 데이터(110(3), 깊이 데이터(110(4)) 등))는 HMD(102)의 메모리에 캐시(cache)될 수 있어, 이것이, 예를 들어, 재투영된 프레임을 렌더링하고/하거나 미래에 데이터를 압축 해제하는 데 사용되기 위해, 나중에 액세스될 수 있다. 프레임들이 호스트 컴퓨터(106)로부터 연속적으로 수신됨에 따라 블록 120 내지 블록 124가 반복될 수 있다는 것이 이해될 것이다. 데이터(110)가 일정 기간 동안 HMD(102)의 메모리에 유지(예를 들어, 캐시됨)될 수 있고, 그 후에 데이터(110)는 퓨처 데이터(110)를 저장하기 위한 공간을 만들기 위해 폐기될 수 있다는 것이 이해될 것이다.

[0036] 126에서, HMD(102)의 로직은, HMD(102)의 디스플레이 패널(들) 상에 이미지를 제시하는데 사용되기 위해 압축 해제된 픽셀 데이터(110(1))와 같은 최근에 압축 해제된 데이터(110)를 검색할 수 있다. 예를 들어, 데이터(110)가 블록 120에서 수신되고 블록 122에서 압축 해제될 때, 블록 126에서의 검색 작동은 HMD(102)의 메모리에 저장된 캐시된 픽셀 데이터(110(1))가 HMD(10)에 이용 가능한 가장 최근에 압축 해제된 픽셀 데이터를 나타내는 것으로 결정할 수 있고, HMD(102)의 로직은 메모리로부터 가장 최근에 수신된 모션 벡터들(110(2))을 압축 해제하는 것으로부터 획득된 픽셀 데이터(110(1))와 같은 가장 최근에 압축 해제된 데이터를 검색할 수 있다. 예시적인 일 실시예에서, HMD(102)가 90Hz의 리프레시 레이트에서 작동되고 있다면, 프레임은 대략 11.11 밀리초마다 한 번씩 디스플레이될 것이다. 이러한 맥락에서, HMD(102)가 마지막 프레임이 디스플레이된 이후 마지막 11.11 밀리초 동안에 새로운 데이터를 수신하고 압축 해제했다면, 새로운 압축 해제된 픽셀 데이터(110(1))는 다음 프레임을 렌더링하기 위해 사용될 수 있다. 반면에, HMD(102)가 마지막 프레임이 표시된 이후 마지막

11.11 밀리초 동안 새로운 데이터를 수신하고 압축 해제하지 않은 경우, 가장 최근에 렌더링된 프레임으로부터의 픽셀 데이터(110(1))가 다음 프레임을 렌더링하기 위해서 사용될 수 있다.

[0037] 128에서, HMD(102)의 로직은 프레임에 대한 수정된 픽셀 데이터를 획득하기 위해 HMD(102)의 예측된 포즈에 적어도 부분적으로 기초하여 프레임에 대한 픽셀 데이터를 수정할 수 있다. 예를 들어, HMD(102)의 로직은, 블록 126에서 검색된 픽셀 데이터를 생성하기 위해 애플리케이션에 의해 사용된 HMD(102)의 원래 예측된 포즈와, 프레임을 위한 조명 시간에 가까운 시간에 HMD(102)의 로직에 의해 예측된 업데이트된 포즈 간의 비교에 기초하여 픽셀 데이터에 조정을 적용할 수 있다. 이 비교는 애플리케이션이 프레임을 렌더링했을 때의 원래 포즈 예측과 디스플레이 패널(들) 상에 프레임을 렌더링하기 전에 HMD(102)에서 업데이트된 포즈 예측 간의 델타(또는 차이)를 밝힐 수 있고, 블록 118에서 적용된 조정은 이 델타를 보상하기 위한 회전 계산을 포함할 수 있다(예를 들어, 두 포즈 결정들 사이의 델타에 기초하여, 픽셀 데이터를 일 방향 또는 다른 방향으로 시프트 및/또는 회전함으로써).

[0038] 또한, 128에서, HMD(102)의 로직은 또한, 블록 124에서 추출되고 블록 126에서 검색된 모션 벡터들(110(2))의 어레이에 적어도 부분적으로 기초하여 프레임에 대한 픽셀 데이터를 수정할 수 있다. 블록 128에서의 수정의 결과로서, 재투영된 프레임에 대한 수정된 픽셀 데이터가 획득된다. 본 개시내용이 "실제" 프레임과 재투영된(또는 "팬텀" 프레임) 사이를 구별하지만, 이러한 구별은 실제 프레임이 HMD(102) 상에 대한 제시 전에 블록 128에서 종종 조정/수정되지 않는다는 것을 의미하지 않는다. 즉, HMD 측에 제시된 프레임들은 "실제" 및 "팬텀" 프레임들 모두에 대해 블록 128에서 수행될 수 있는 수정 작동(들)으로 인해 합성(즉, 호스트 컴퓨터(106) 상에서 실행되는 애플리케이션에 의해 출력된 원본 프레임들과 동일하지 않음)된 것으로 간주될 수 있다. 이러한 의미에서, 블록 128에서 수정되는 모든 프레임은 본원에서 사용되는 "재투영된" 프레임으로서 간주될 수 있다. 픽셀 데이터가 수정되는지 여부 및/또는 이것이 블록 128에서 어느 정도 수정되는지는 시스템의 작동 성능에 의존된다. 예를 들어, 애플리케이션에 의해 출력된 모든 프레임이 HMD(102)에서 적시에 수신되는 라이브 렌더링 시나리오에서 예정된 대로 모든 것이 진행되는 경우, 원래 포즈 예측과 업데이트된 포즈 예측 간의 델타는 0에 가까울 수 있고(0은 아니더라도), 이 경우, 블록 128에서 적용된 수정(있는 경우)은 최종 출력에 거의 또는 전혀 영향을 미치지 않을 수 있다. 그러나, 응용 프로그램이 프레임 레이트에 도달하지 못하거나, 또는 프레임이, 네트워크 혼잡으로 인해 늦게 도착하거나 전송 중 드롭되는 시나리오에서는, 블록 128에서 픽셀 데이터(110(1))에 적용된 수정이 원래 포즈 예측과 업데이트된 포즈 예측 사이의 더 큰 델타를 보상하기 위해 중요할 수 있다. 아래에서 더 상세히 설명되는 바와 같이, 모션 벡터들(110(2))에 기초한 재투영된 프레임에 대한 픽셀 데이터의 수정은 렌더 메시(render mesh)를 활용할 수 있다. 예를 들어, 렌더 메쉬는 재투영된 프레임에 대해 생성될 수 있다. 렌더 메시는 복수의 정점들을 갖는 바둑판 모양(tessellated) 메시를 포함할 수 있고, 렌더 메시의 정점들은 이동되는 정점들로서 렌더 메시 내의 다양한 위치로 이동될 수 있다(예를 들어, (i) 넌-제로 모션 벡터들(110(2))의 방향들로, (ii) 넌-제로(non-zero) 모션 벡터들(110(2))의 크기들에 대응되는 양들만큼 정점들을 이동시킴으로써). 따라서, 블록 128에서의 픽셀 데이터의 수정은, 예를 들어, 재투영된 프레임의 수정된 픽셀 데이터를 획득하기 위해서, 렌더 메시의 이동된 정점들에 따라 블록 126에서 검색된 픽셀 데이터의 픽셀 값들을 이동시킴으로써, 렌더 메시의 이동된 정점들에 따를 수 있다. 예를 들어, 픽셀 값은, 렌더 메시 내에서 이동된 정점에 따라 수정된 픽셀 데이터 내의 새 위치로 왼쪽으로 4픽셀, 위쪽으로 4픽셀 이동될 수 있었다. 일부 실시형태에서, 모션 벡터들(110(2))은 수정된 모션 벡터들(110(2))을 픽셀 데이터에 적용하기 전에 수정될 수 있다. 예를 들어, 아래에서 더 자세히 설명되는 바와 같이, 필터(예를 들어, N x N 스칼라 중간값 필터(median filter), M x M 평균-최대 블러(blur) 필터 등)가 모션 벡터들(110(2))의 어레이에 적용되어 모션 벡터들의 수정된 어레이를 획득하고, 모션 벡터들의 수정된(예를 들어, 필터링된) 어레이는 블록 128에서 픽셀 데이터를 수정하기 위해 사용될 수 있다.

[0039] 또한, 128에서, HMD(102)의 로직은 또한, 블록 126에서 검색된 깊이 데이터(110(4))에 적어도 부분적으로 기초하여 프레임에 대한 픽셀 데이터를 수정할 수 있다. 깊이 데이터(110(4))는, 무엇보다도, 장면에 있는 객체의 시차를 조정하기 위해서 블록 128에서 사용될 수 있다(예를 들어, 세계 공간(world space)에서 멀리 떨어져 있는 배는 머리 움직임으로, 근접한 객체가 동일한 머리 움직임으로 움직이는 것만큼 많이 움직이지 않을 수 있다). 장면의 가상 객체에 대응되는 픽셀의 깊이를 아는 것은 HMD(102) 상의 재투영 동안 이러한 시차를 조정하는 방법을 아는 데 도움이 된다. 깊이 정보를 아는 것은 또한, HMD(102)의 회전에 기초할 뿐만 아니라 공간에서 HMD(102)의 병진에 기초하여 장면을 워핑(warp)하는 것을 허용한다. 깊이 데이터(110(4))가 호스트 컴퓨터(106)로부터 수신되지 않으면, HMD(102)의 로직은 약 2미터의 평균 깊이를 가정할 수 있다.

[0040] 130에서, 이미지는, 수정된 픽셀 데이터에 적어도 부분적으로 기초하여 HMD(102)의 디스플레이 패널(들) 상에

제시될 수 있다. 예를 들어, HMD(102)의 로직은 수정된 픽셀 데이터를 프레임 버퍼(들)로 출력할 수 있고, 이미지(들)로 하여금 프레임 버퍼에 출력된 수정된 픽셀 데이터에 기초하여 HMD(102)의 디스플레이 패널(들) 상에 제시되게 할 수 있다. 이것은 수정된 픽셀 데이터를 HMD(102)의 디스플레이 패널(들)에 스캔 아웃하는 것, 및 디스플레이 패널(들) 상의 픽셀들을 조명하기 위해 디스플레이 패널(들)의 발광 소자를 조명하는 것을 포함할 수 있다. 한 쌍의 디스플레이 패널들을 갖는 HMD(102)의 경우, 이러한 수정된 픽셀 데이터는 한 쌍의 디스플레이 패널들 상에 디스플레이될 한 쌍의 이미지들을 나타내는 프레임에 대응될 수 있고, 스테레오 프레임 버퍼로 이에 상응되게 출력되고 스캔 아웃될 수 있다. 재투영된 프레임에 대응되는 결과적인 이미지는 모션 벡터들(110(2))에 따른 픽셀 데이터의 수정에 의해서 "모션 평활화"된다.

[0041] 도 2는 본 명세서에 개시된 실시형태에 따른, 호스트 컴퓨터(106) 및 HMD(102)의 각각의 렌더링 워크로드를 도시하는 2개의 예시적인 타임라인들(200(1) 및 200(2))을 예시하는 도면이다. 도 2의 실시예는 호스트 컴퓨터(106)와 연관된 제1 타임라인(200(1))에 관하여 3개의 예시적인 프레임, 즉 프레임 "F", 프레임 "F+1" 및 프레임 "F+2"를 도시한다. 이러한 제1 타임라인(200(1))은, 호스트 컴퓨터(106)의 GPU(들)를 사용하여 호스트 컴퓨터(106) 상에서 실행되는 애플리케이션에 의해 어떻게 프레임들이 연속적으로 렌더링될 수 있는지를 예시한다. 여기서, 애플리케이션은 제1 타임라인(200(1)) 상에서 좌측으로부터 우측으로, 시퀀스로, 프레임 F를, 다음으로 프레임 F+1을, 다음으로 프레임 F+2를 렌더링한다. 제1 타임라인(200(1)) 상의 타원은, 애플리케이션이 계속 실행됨에 따라, 이것이 임의의 수의 프레임에 대해 계속될 수 있음을 나타낸다. 제1 타임라인(200(1))은 또한, 수평 타임라인(200(1))에 수직으로 배향된 수직 라인에 의해, 애플리케이션이 타겟 프레임 레이트(예를 들어, 수직 라인들이 약 11.11 밀리초로 구분되는 90 Hz의 프레임 레이트)를 목표로 하고 있다는 것을 의미한다. 도 2의 실시예에서, 호스트 컴퓨터(106) 상에서 실행되는 애플리케이션은 프레임 F 및 프레임 F+1에 걸쳐 타겟 프레임 레이트에 도달하게 되지만, 애플리케이션은 프레임 F+2에 대한 타겟 프레임 레이트에 도달하는 데 실패한다. 예를 들어, 이것은, 프레임 F+2의 장면이 많은 수의 움직이는 객체 또는 복잡한 텍스처를 포함하고/하거나, 다른 이유 때문에, 애플리케이션이 프레임 F+2를 렌더링하기 위해서 할당된 시간보다 더 오래 걸리는 경우일 수 있다. 호스트 컴퓨터(106)는 타임라인(200(1)) 상의 각각의 프레임을 렌더링하기 위해 HMD(102)의 예측된 포즈를 결정하도록 HMD(102)의 움직임에 관한 헤드 트래킹 데이터(208)를 수신할 수 있다.

[0042] HMD(102)와 연관된 도 2의 제2 타임라인(200(2))은 개별 프레임들에 대한 HMD(102)의 렌더링 워크로드들(202(a), 202(b), 및 202(c))을 도시한다. 주어진 프레임에 대한 HMD(102)의 개별 렌더링 워크로드(202)는 최종 이미지(들)가 HMD(102)의 디스플레이 패널(들) 상에 제시되기 전에 픽셀 데이터(110(1))에 적용되는 조정을 나타낼 수 있다. 이러한 조정들은 HMD(102) 상에 최종 이미지(들)를 렌더링하기 전에 픽셀 데이터(110(1))에 적용되는, 기하학적 왜곡, 색수차, 머리 움직임 등에 대한 조정들을 한정됨 없이 포함할 수 있다. 프레임 F 및 프레임 F+1의 경우, 이러한 조정들은 호스트 컴퓨터(106) 상에서 실행되는 애플리케이션에 의해 생성된 픽셀 데이터에 적용되고, 이러한 조정들 중 적어도 일부는 HMD(102)의 원래 예측된 포즈와 HMD(102)의 업데이트된 포즈 예측 사이의 델타를 처리함으로써 호스트 컴퓨터(106)로부터 수신된 포즈 데이터(110(3))를 활용할 수 있다. 따라서, 두 번째 타임라인(200(2))의 프레임 F 및 프레임 F+1은, 이들이 라이브 렌더링 상황 동안에 실시간으로 애플리케이션으로부터 출력되는 픽셀 데이터(110(1))의 수정된 버전들이라는 점에서 "실제" 프레임들의 수정된 버전들을 나타내기 위한 것이다. 대조적으로, 프레임 F+2를 렌더링하기 위해, HMD(102)는 선행 프레임의 포즈 예측 및 HMD(102)에 의해서 수행된 업데이트된 포즈 예측에 기초하여 재투영된(또는 "팬텀") 프레임을 생성하기 위해 선행 프레임(예: 프레임 F+1에 대한 픽셀 데이터 110(1))에 대한 이전에 수신된 픽셀 데이터(110(1))를 사용할 수 있다. 또한, 렌더링 워크로드(202(c))는, 본원에 설명된 바와 같이, 프레임 F+1에 대한 데이터(110)와 함께 수신된 모션 벡터들(110(2))에 기초한 프레임 F+1로부터 "모션 평활화" 프레임 F+2로 픽셀 데이터를 수정하는 것을 포함할 수 있다. 어느 경우든, 렌더링 워크로드(202)의 결과는 프레임 버퍼(예를 들어, 스테레오 프레임 버퍼)에 출력될 수 있는 수정된 픽셀 데이터의 생성이다. 본원에서 "실제" 프레임과 "팬텀" 프레임 사이의 구분은 실제 프레임이 HMD(102) 상에서 조정되지 않는다는 것을 의미하지 않고, 이러한 의미에서, HMD 측 상에서 생성된 프레임들은 모두 효과적으로 합성된다(즉, 호스트 컴퓨터(106) 상에서 실행되는 애플리케이션에 의해 출력된 원래 프레임과 동일하지 않음).

[0043] 도 2의 제2 타임라인(200(2))은 또한, 각각의 프레임에 대한 스캔 아웃 시간(204(a), 204(b), 204(c))뿐만 아니라, 각각의 프레임에 대한 조명 시간(206(a), 206(b) 및 206(c))을 도시한다. 주어진 프레임에 대한 스캔 아웃 시간(204) 동안, (수정된 픽셀 데이터의) 픽셀 값들의 서브세트들이 디스플레이 포트(예를 들어, 고화질 멀티미디어 인터페이스(HDMI: high-definition multimedia interface))를 통해 HMD(102)의 디스플레이 패널(들)에 스캔 아웃되고, 주어진 프레임에 대한 조명 시간(206) 동안, HMD(102)의 디스플레이 패널(들)의 발광 소자는 디스플레이 패널(들)의 픽셀로 하여금 조명하게 하도록 조명된다. 도 2는, HMD(102)의 재생률로 디스플레이 패

널(들)의 발광 소자들로부터 광을 동시에 방출하기 위해 LCD 패널과 함께 사용될 수 있는 글로벌 플래싱 유형의 디스플레이 구동 방식의 실시예를 예시한다. 예시적인 실시예에서, HMD(102)가 90 Hz 재생률에서 작동되면, 각각의 프레임에 대한 조명 시간(206)은 대략 11.11 밀리초만큼 분리될 수 있다.

[0044] 도 3은 HMD(102)와 같은 디스플레이 상에 프레임을 렌더링하기 위해 재투영이 사용되고 있을 때 장면에서 이동하거나 애니메이션되는 객체를 처리하기 위한 예시적인 모션 평활화 기술을 예시하는 다이어그램이다. 도 3의 예는 일련의 프레임들(300)로서 시퀀스로 렌더링될 3개의 예시적인 프레임들(300(1), 300(2), 및 300(3))을 도시한다. 도 3의 실시예에서, 객체(302)는 프레임들(300(1)-300(3))의 코스에 걸쳐 좌측 방향(즉, 우측으로부터 좌측으로)으로 장면을 가로질러 이동하고 있는 것으로 도시된다. 프레임들(300(2) 및 300(3))에서 객체(302)의 점선 윤곽선은 이전에 렌더링된 프레임(300)에서 객체(302)가 위치되었던 위치를 나타낸다. 여기서, 프레임(300(1))이 먼저 렌더링된 후에, 프레임(300(2))이 두 번째로 렌더링되고, 다음으로, 프레임(300(3))이 세 번째로 렌더링된다.

[0045] 일련의 프레임들(300)의 프레임들(300) 중 적어도 일부는, 이들이 프레임(300)에 대한 픽셀 데이터에 기초하여 HMD(102)에 대응되는 이미지를 제시하기에 충분한 시간 내에 비디오 게임 애플리케이션과 같은 애플리케이션, 또는 임의의 다른 유형의 그래픽 기반 애플리케이션으로부터 출력된다는 점에서 "실제" 프레임일 수 있다. 애플리케이션은 개별 프레임들(300)을 렌더링하기 위해 픽셀 데이터(110(1))를 프레임 버퍼에 출력하는 그래픽 파이프라인에서 실행될 수 있다.

[0046] 런타임 동안, HMD(102)의 헤드 트래킹 모듈은 사용자(104)의 머리 포즈에 따른 일련의 프레임들(300)에서 다음 프레임(300)을 렌더링하는 방법에 관해 애플리케이션에 알리기 위해 애플리케이션을 실행하는 호스트 컴퓨터(106)에 제공되는 HMD(102)의 위치 및 배향(포즈)에 대한 데이터를 생성할 수 있다. 이것은 애플리케이션이, 사용자(104)로 하여금 자신이 객체들(움직이는 객체(302)와 같은 이동 객체 및 정적 객체 모두)을 포함하는 가상 환경을 둘러보고 있다고 믿게 하는 방식으로 HMD(102) 상에 이미지를 렌더링하기 위한 픽셀 데이터(110(1))를 출력하는 것을 허용한다. 정적 객체 및 이동 객체는 모두, 애플리케이션이 프레임 레이트에 도달되는 경우에, 사용자(104)의 머리 움직임과 함께 예상되는 방식으로 장면 내에서 움직이는 것으로 인지된다. 본원에 설명된 모션 평활화 기술은, 움직이는 객체들에 대해 유사한 시각적 인지가 달성되도록, 프레임 레이트에 도달하지 못하는 애플리케이션을 보상하는 방식이다.

[0047] 도 3의 실시예에서, 제1 프레임(300(1))은 애플리케이션으로부터 수신되는 제1 "실제"프레임을 나타낼 수 있고, 제2 프레임(300(2))은 애플리케이션으로부터 수신되고 제1 프레임(300(1)) 이후에 렌더링되는 제2 "실제" 프레임을 나타낼 수 있고, 제3 프레임(300(3))은 제2 프레임(300(2))과 연관된 픽셀 데이터(110(1)(2))로부터 생성되는 재투영된 프레임을 나타낼 수 있다. 그러므로, 도 3의 "제3 프레임(300(3))"은 본원에서 때때로 "재투영된 프레임(300(3))"으로 지칭된다. 도 3의 실시예에서, 제2 프레임(300(2))(프레임 2)을 압축하기 위해, 호스트 컴퓨터(106)의 로직은, 제1 프레임(300(1))과 연관된 제1 픽셀 데이터(110(1)(1)) 및 제2 프레임(300(2))과 연관된 제2 픽셀 데이터(110(1)(2))를 입력으로서 호스트 컴퓨터(106)의 그래픽 처리 장치(GPU)(304)에 제공할 수 있다.

[0048] GPU(304)의 비디오 인코더(예를 들어, 비디오 인코드 칩)는 GPU(304)에 입력된 제1 픽셀 데이터(110(1)(1)) 및 제2 픽셀 데이터(110(1)(2))에 기초하여 모션 벡터들의 어레이(110(1)(2))를 생성할 수 있다. 모션 벡터들의 어레이(110(2)(2))를 생성하기 위해, GPU(304)의 비디오 인코더는 입력으로서 제공된 각각의 프레임(300)의 픽셀 데이터(110(1)) 사이의 픽셀당 값들(예를 들어, 휘도 값들)과 비교될 수 있다. 추가적으로 또는 대안적으로, GPU(304)의 비디오 인코더는 입력으로서 제공된 각각의 프레임(300)의 픽셀 데이터(110(1)) 사이에서 매크로블록들(예를 들어, 16픽셀들의 블록(즉, 4 x 4 픽셀 매크로블록), 64픽셀들의 블록(즉, 8 x 8 픽셀 매크로블록))을 비교할 수 있다. 이러한 방식으로, GPU(304)는 임의의 적절한 해상도의 한 쌍의 프레임들(300) 사이에서 픽셀 데이터(110(1))의 부분들을 비교할 수 있다. 일부 실시형태에서, 입력 픽셀 데이터(110(1)(1) 및 110(1)(2))는 다운 샘플링된 프레임들을 GPU(304)에 입력하기 위해 더 낮은 해상도로 다운 샘플링된다. 일부 실시형태에서, 모션 벡터들(110(2)(2))은 HEVC 알고리즘과 같은 임의의 적절한 압축 알고리즘에 따라 생성된다.

[0049] GPU(304)로부터 출력된 모션 벡터들의 어레이(110(2)(2))는 본원에서 때때로 "모션 벡터 필드"로 지칭된다. 이러한 모션 벡터 필드(110(2)(2))는 임의의 적절한 해상도에서 출력될 수 있고/있거나, 임의의 적절한 해상도로 다운 샘플링/업 샘플링될 수 있다. 예를 들어, 모션 벡터들의 어레이(110(2)(2))는 픽셀당 단일 모션 벡터, 픽셀들의 그룹당 단일 모션 벡터(예를 들어, 4 x 4 매크로블록, 8 x 8 매크로블록, 픽셀들의 임의의 형상 패치 등에 대한 하나의 모션 벡터), 또는 심지어 주어진 프레임(300)에 대한 모든 픽셀들을 위한 단일 모션 벡터를 포

함할 수 있다.

[0050] GPU(304)의 비디오 인코더에 의한 입력 픽셀 데이터(110(1))의 비교에 기초하여, 제2 프레임(300(2))의 일 부분이 제1 프레임(300(1))의 일 부분과 유사한(예를 들어, 임계 휘도 값 내에서) 경우, 및 각각의 프레임(300)의 유사한 부분이 거리(예를 들어, X 및 Y 성분 방향들에서 픽셀들의 수)만큼 오프셋된 경우, 이는 모션 벡터들의 어레이(110(2)(2))에 포함되는 모션 벡터에 의해서 표현될 수 있다. 프레임들(300(1) 및 300(2))에서 객체(302)에 대응되는 픽셀 값들이 유사하다(예를 들어, 픽셀 데이터(110(1))의 픽셀 값에 기초하여 일부 유사성 메트릭을 만족하는 정합 부분들)고 GPU(304)의 비디오 인코더에 의해서 결정되는 일 실시예를 고려한다. 이러한 객체(302)에 대한 모션 벡터는 객체(302)의 움직임과 동일한 방향을 가리키는 방향을 가질 수 있거나, 또는 모션 벡터는 객체(302)의 움직임 방향과 반대 방향을 가리킬 수 있다. 다시 말해서, 모션 벡터는 후속 프레임(300(2))에서 객체(302)의 위치로부터 오프셋되는 이전 프레임(300(1))에서 객체(302)의 위치를 다시 참조하는 방향을 가리킬 수 있다. 이와 같이, 어레이의 모션 벡터(110(2)(2))는 제2 프레임(300(2))의 좌표로부터 제1 프레임(300(1))의 좌표까지의 오프셋을 제공한다. 오프셋은 객체(302)와 같은 이동하거나 애니메이션되는 객체에 대해 제1 프레임(300(1)) 내의 이미지로부터 제2 프레임(300(2)) 내의 이미지로의 변환을 설명한다.

[0051] 전술된 바와 같이, 도 3의 제3 프레임(300(3))은 재투영된 프레임을 나타낼 수 있으며, 이는 제3 프레임(300(3))에 대한 픽셀 데이터(110(1)(3))가 이전에 렌더링된 실제 프레임(이 경우, 제2 프레임(300(2)))과 연관된 픽셀 데이터(110(1)(2))로부터 파생될 수 있음을 의미한다. 즉, 재투영된 프레임(300(3))에 대한 픽셀 데이터(110(1)(3))는 실시간으로 호스트 컴퓨터(106)로부터 수신되지 않으나, 애플리케이션 생성 프레임들의 픽셀 데이터(110(1))로부터 생성되고, 애플리케이션이 프레임 레이트에 도달되지 않을 때 그리고/또는 네트워크 정체가 발생될 때, 누락된 프레임들의 간격들을 "채우기" 위해서 사용된다. 이 경우, 재투영된 프레임(300(3))에 대한 픽셀 데이터(110(1)(3))는 제2 프레임(300(2))과 연관된 픽셀 데이터(110(1)(2))로부터 생성되는 데, 이는 제2 프레임(300(2))이 도 3의 실시예에서 재투영된 프레임(300(3)) 이전에 가장 최근에 렌더링된 프레임(300)이기 때문이다. 일부 실시형태에서, 회전 변환 및 재투영 변환은, 재투영된 프레임(300(3))에 대한 제3 픽셀 데이터(110(1)(3))를 생성하기 위해서 제2 프레임(300(2))과 연관된 제2 픽셀 데이터(110(1)(2))를 수정하기 위해 계산되고 사용되며, 이는 제2 프레임(300(2))이 렌더링된 이후 HMD(102)의 회전을 처리하는 방식으로 제2 프레임(300(2))에서 렌더링된 장면을 효과적으로 회전, 병진, 및/또는 이와 달리 이동시킨다. 예를 들어, 사용자(104)는 제2 프레임(300(2))이 렌더링된 시각 이후로 자신의 머리를 회전시킬 수 있으며, 이는, 이러한 머리 움직임에 따라 장면이 제시되도록, 재투영된 프레임(300(3))에 대한 제3 픽셀 데이터(110(1)(3))의 생성에서 처리된다.

[0052] 도 3의 모션 평활화 기술은 다음으로, 재투영된 프레임(300(3))에 대한 수정된 제3 픽셀 데이터(110(1)(3)')를 획득하기 위해서, 제2 프레임(300(2))과 연관된 호스트 컴퓨터(106)로부터 수신된 모션 벡터들의 어레이(110(2)(2))에 적어도 부분적으로 기초하여 제3 픽셀 데이터(110(1)(3))를 수정한다. 일부 실시형태에서, 제3 픽셀 데이터(110(1)(3))의 수정된 제3 픽셀 데이터(110(1)(3)')로의 이러한 수정은 특정 픽셀들 또는 픽셀들의 그룹들에 대응되는 넌-제로 모션 벡터들(110(2)(2))에 기초하여 제3 픽셀 데이터(110(1)(3))의 픽셀 값들을 상이한 위치들로 이동시키는 것을 포함한다. 이동은 일 방향으로 그리고 일정 양만큼 될 수 있다(예를 들어, 수평(+/-) 방향 및 수직(+/-) 방향으로 다수의 픽셀을 이동). 다음으로, 재투영된 프레임(300(3))은 수정된 제3 픽셀 데이터(110(1)(3)')에 적어도 부분적으로 기초하여 디스플레이(예를 들어, HMD(102)의 디스플레이 패널(들)) 상에 렌더링된다. 따라서, 모션 벡터(110(2)(2))에 기초하여 수정된 재투영된 프레임(300(3))(예를 들어, 모션 벡터(110(2)(2))는, 이전에 수신된(그리고/또는 이전에 렌더링된) 실제 프레임(300(1) 및 300(2))의 픽셀 데이터(110)로부터 제2 프레임(300(2))에 대한 대응되는 픽셀 데이터(110(1)(2))를 압축하는 것의 결과(예: 부산물)로서 생성됨)은 예상 위치에서 객체(302)를 렌더링하기 위해 "모션 평활화"된다.

[0053] 도 4는, 예시적인 렌더 메쉬(400), 및 렌더 메쉬(400)가 재투영된 프레임을 위한 모션 평활화에서 사용될 수 있는 방법을 예시하는 다이어그램이다. 예를 들어, 렌더 메쉬(400)는 호스트 컴퓨터(106)의 GPU(304)에 의해 출력된 모션 벡터들(110(2))의 어레이에 기초하여 도 3의 재투영된 프레임(300(3))에 대한 제3 픽셀 데이터(110(1)(3))를 수정하기 위해서 HMD(102)에 의해서 사용될 수 있다. 이러한 실시예에서, HMD(102)의 로직은 렌더 메쉬(400)를 생성할 수 있고, 렌더 메쉬(400)의 정점들(402)은 모션 벡터들(110(2))의 어레이에 따라 이동될 수 있다. 예를 들어, 정점들(402)은 넌-제로 모션 벡터들(404)의 방향들로, 그리고 넌-제로 모션 벡터들(404)의 크기들에 대응되는 양들만큼 이동될 수 있다. 예를 들어, 도 3에 도시된 좌향 이동 객체(302)를 일례로 들면, 모션 벡터(404)는 렌더 메쉬(400)의 컨텍스트에서 적용되어 정점(402)을 좌향(또는 음의 X) 방향으로 특정 개수의 픽셀들(모션 벡터(404)의 크기에 대응됨)만큼 이동시킬 수 있다.

[0054] 렌더 메쉬(400)는 복수의 정점들(402(1), 402(2),…, 402(N))(집합적으로 402)을 갖는 바둑판 모양 메쉬로서 도시된다. 렌더 메쉬(400)의 바둑판 모양화는 임의의 적절한 기하학적 패턴으로 될 수 있다. 도 4의 예시적인 렌더 메시(400)는, 정사각형들(때때로 "쿼드(quad)들"로 지칭됨), 육각형들(예: 하니콤들 패턴을 위해) 등을 포함하지만 이에 한정되지 않는 임의의 적절한 기하학적 형상이 렌더 메시(400)를 위해서 사용될 수 있으나, 삼각형들(406)의 반복 패턴으로서 도시된다. 이 실시예에서, 정사각형(또는 쿼드)의 하부 좌측 코너로부터 정사각형(또는 쿼드)의 상부 오른쪽 코너까지의 대각선은 특정 배향을 갖는 반복 삼각형들(406)의 렌더 메시(400)를 생성하기 위해서 사용된다. 렌더 메쉬(400)에 대한 상이한 배향은, 도 4에 도시된 배향 대신에, 정사각형(또는 쿼드)의 상부 좌측 코너로부터 정사각형(또는 쿼드)의 하부 우측 코너까지의 대각선을 사용하여 정사각형들(또는 쿼드들)을 삼각형들(406)로 분할함으로써 생성될 수 있다. 일부 실시형태에서, 예를 들어, 정사각형(또는 쿼드)의 하부 좌측 코너로부터 정사각형(또는 쿼드)의 상부 우측 코너까지의 대각선을 사용하여 하나 걸러 정사각형 마다 분할하고, 정사각형(또는 쿼드)의 상부 좌측 코너로부터 정사각형(또는 쿼드)의 하부 우측 코너까지의 대각선을 사용하여 사이에 있는 정사각형들을 분할함으로써, 이러한 상이한 배향들의 혼합이 또한 단일 렌더 메쉬(400)에 대해 사용될 수 있다. 일부 실시형태에서, HMD(102)의 로직은, 호스트 컴퓨터(106)로부터 수신된 모션 벡터 필드(110(2))에 기초하여, 이러한 다수의 배향들 중 어느 배향을 렌더 메시(400)를 생성하는 데 사용할지를 동적으로 결정하도록 구성될 수 있다. 이것은, 가장 평활하게 보이는 모션 평활화된 이미지로 귀결되는, 렌더 메쉬(400)에서 기하학적 형상들(예를 들어, 삼각형들(406))에 대한 최상의 배향을 선택하기 위해 수행될 수 있다.

[0055] 렌더 메쉬(400)는 또한 임의의 적절한 해상도에서 생성될 수 있다. 예를 들어, 가장 높은 해상도의 렌더 메쉬(400)는 픽셀당 두 개의 인접된 삼각형들(406)일 수 있으며, 각각의 정사각형(또는 쿼드)은 단일 픽셀에 매핑된다. 더 낮은 해상도는 픽셀들의 일 그룹, 예를 들어, 16개의 픽셀들의 일 그룹당 두 개의 인접된 삼각형들(406)일 수 있다. 대안적으로, 픽셀들은 임의의 적절한 해상도에서 렌더 메쉬(400)의 정점들(402)에 매핑될 수 있다. 예를 들어, 각각의 정점(402)은 가장 높은 해상도에서 단일 픽셀과 연관될 수 있거나, 각각의 정점(402)은 더 낮은 해상도에서 픽셀들의 일 그룹, 예를 들어, 16개의 픽셀들의 일 그룹과 연관될 수 있다. 일부 실시형태에서, 렌더 메쉬(400)의 해상도는 모션 벡터들(110(2))의 어레이의 해상도와 동일한 해상도이어서, 모션 벡터들(110(2))의 어레이에서 단일 모션 벡터(404)는 정점(402)에 또는 정사각형(또는 쿼드)에 매핑된다(예를 들어, 2개의 인접된 삼각형들(406)). 렌더 메쉬(400)와 모션 벡터들(110(2))의 어레이 사이의 정합 해상도의 달성은 다양한 방식으로, 예를 들어, GPU(304)로부터, 렌더 메쉬(400)의의 해상도와 정합하는 특정 해상도에서 모션 벡터들(110(2))의 어레이를 요청함으로써, 렌더 메쉬(400)로서의 해상도와 정합하도록 모션 벡터들(110(2))의 어레이를 다운 샘플링 또는 업 샘플링함으로써, 또는 호스트 컴퓨터(106)로부터 수신되는 모션 벡터들(110(2))의 어레이의 해상도와 정합하는 해상도로 렌더 메시(400)를 생성함으로써 이루어질 수 있다.

[0056] 도 4는 4개의 넌-제로 모션 벡터들(404(1), 404(2), 404(3), 및 404(4))이 렌더 메시(400)의 4개의 정점들(402)에 대응되는 실시예를 도시한다. 이러한 4개의 모션 벡터들(404(1)-(4))은 도 3에 도시된 이동 객체(302)에 기초하여 GPU(304)가 검출한 모션 벡터들에 대응될 수 있다. 따라서, 예시적인 모션 벡터들(404(1)-(4))은, 언급된 바와 같이, 방향성이 객체(302)의 방향성 모션의 방향성과 반대일 수 있으나(예를 들어, 오른쪽 방향으로), 객체(302)의 방향 모션에 대응되는 좌향 방향으로 가리킬 수 있다. 모션 벡터(404)의 방향성은 원하는 방향으로 픽셀 데이터(110(1))를 수정하기 위해 모션 평활화 알고리즘에서 고려될 수 있다. 모션 벡터들의 필드(110(2))가 4개의 예시적인 모션 벡터들(404(1)-(4))을 포함하고, 모션 벡터들(110(2))의 어레이에서 모든 잔여 모션 벡터들이 제로 벡터인 기본 예를 고려한다. 이 실시예에서, 재투영된 프레임(300(3))에 대한 픽셀 데이터(110(1)(3))는, 넌-제로 모션 벡터들(404(1)-(4))에 대응되는 정점들(402)을 이동된 정점(408(1), 408(2), 408(3) 및 408(4))인 렌더링 메시(400) 내의 다른 위치들(도 4의 하부에 도시됨)로 이동시킴으로써 넌-제로 모션 벡터들(404(1)-(4))에 기초하여 수정될 수 있다. 도 4의 하부는 모션 평활화가 적용된 후의 렌더 메쉬(400)를 도시한 것으로, 이동된 정점들(408(1)-(4))은 모션 평활화 전의 정점들(402)의 위치들에 비해 렌더 메쉬(400) 내의 상이한 위치들에 있다. 모션 벡터들(404(1)-(4))이 이러한 방식으로 적용될 때, 이동된 정점들(408(1)-(4))은, 예를 들어, 렌더 메쉬(400)에서 기하학적 형상들(예를 들어, 삼각형들(406)) 중 특정 형상들을 신장하거나 둘러쌈으로써, 렌더 메쉬(400)의 하나 또는 둘 이상의 부분들로 하여금 왜곡되게 한다. 도 4의 예에서, 도 4의 하부에 도시된 바와 같이, 삼각형들(406) 중 일부는 모션 평활화의 결과로서 신장되어 신장된 삼각형들(410)을 생성한다. 이동된 정점들(408(1)-(4))에 대응되는 (재투영된 프레임(300(3))에 대한 픽셀 데이터(110(1)(3))의) 픽셀 값들은 렌더 메쉬(200) 내에서 이동된 정점들(408(1)-(4))의 위치들에 대응되는 상이한 픽셀의 위치들에서 렌더링된다. 이동된 정점들(206(1)-(4))과 비이동된 정점들(402) 사이의 픽셀 위치들은 (예를 들어, 이동된 정점들(408(1)-(4))과 비이동된 정점들(402) 사이의 픽셀 값들을 보간함으로써와 같이, 그라디언

트를 적용함으로써) 블렌딩될 수 있다. 일부 실시형태에서, 깊이 버퍼는 재투영된 프레임(300(3))에 대한 수정된 픽셀 데이터(110(1)(3)')의 프레임 버퍼에 출력되는 픽셀 값들의 최종 세트를 결정하기 위해서 활용될 수 있다. 즉, 모션 벡터들(404(1)-(4))을 렌더 메쉬(400)에 적용한 것의 결과로서, 이동된 정점들(408(1)-(4))에 대응되는 이미지 내 위치에 다수의 픽셀 값들이 있을 수 있다. 이 경우, "더 가까운"(더 작은) 깊이 값과 연관된 픽셀 값들은 어는 것이든지, "더 먼"(더 큰) 깊이 값과 연관된 해당 위치에서 다른 픽셀 값을 렌더링하는 것 대신 렌더링될 수 있다.

[0057] 도 5는 본원에 개시된 실시형태들에 따른, 모션 평활화 기술의 일부로서 그래픽 처리 유닛(GPU)에 의해 생성된 모션 벡터들을 사용하여 재투영된 프레임들을 생성하기 위한 예시적인 프로세스(500)의 흐름도이다. 논의를 위해, 프로세스(500)는 이전 도면들을 참조하여 설명된다.

[0058] 502에서, 호스트 컴퓨터(106)의 로직은 이전에 렌더링된 프레임들(300)과 연관된 픽셀 데이터(110(1))를 입력으로서 GPU(304)에 제공할 수 있다. 예를 들어, 2개의 가장 최근에 렌더링된 프레임들(300)과 연관된 픽셀 데이터(110(1))는, 압축된 픽셀 데이터(110(1))의 전송 이전에 픽셀 데이터(110(1))를 압축하기 위한 압축 알고리즘의 일부로서, GPU(304)에 입력으로서 제공될 수 있다. 이러한 프레임들(300)은 애플리케이션(예: 비디오 게임 애플리케이션)으로부터 수신된 실제 프레임들, 예를 들어, 과거에 렌더링된 제1 프레임(300(1)) 및 제1 프레임(300(1)) 이후에 렌더링된 제2 프레임(300(2))일 수 있다. 따라서, 제2 프레임(300(2))은 호스트 컴퓨터(106) 상에서 실행되는 애플리케이션에 의해 출력된 가장 최근에 렌더링된 프레임을 나타낼 수 있고, 제1 프레임(300(1))과 제2 프레임(300(2))은 애플리케이션에 의해서 일련의 프레임들(300)로 연속적으로 출력될 수 있지만, 블록 502에서 입력으로서 제공되는 픽셀 데이터(110(1))가 연속적으로 렌더링되는 프레임들에 대한 픽셀 데이터(110(1))일 필요는 없다. 예를 들어, 중간 프레임(들)(300)은 제1 프레임(300(1))과 제2 프레임(300(2)) 사이에 렌더링될 수 있고, 블록 502에서 입력으로서 제공되는 픽셀 데이터(110(1))는 제1 프레임(300(1)) 및 제2 프레임(300(2))과 관련될 수 있다.

[0059] 504에서, 모션 벡터들(110(2))의 어레이는 GPU(304)로부터 수신될 수 있다. 블록 504에서 수신되는 모션 벡터들(110(2))의 어레이는 제1 프레임(300(1))과 연관된 제1 픽셀 데이터(110(1)(1)) 및 제2 프레임(300(2))과 연관된 제2 픽셀 데이터(110(1)(2))에 적어도 부분적으로 기초하여(예를 들어, 제1 픽셀 데이터(110(1)(1))와 제2 픽셀 데이터(110(1)(2)) 간의 비교에 기초하여) GPU(304)의 비디오 인코더에 의해 생성될 수 있다. GPU(304)의 비디오 인코더는, 예를 들어, 비교되는 픽셀 값들 간의 차이가 임계 차이보다 더 작은지를 결정함으로써, 픽셀 값들(또는 픽셀 값들의 그룹들) 간의 유사성을 찾는 적절한 컴퓨터 비전 및/또는 비디오 인코딩 알고리즘(예: HEVC)을 사용하도록 구성될 수 있다. 이러한 유사성 메트릭 내의 어는 것이나 두 개의 프레임들(300) 사이의 정합 픽셀 데이터(110(1))인 것으로 간주될 수 있다.

[0060] 506에서, HMD(102)의 로직은 제2 프레임(300(2))의 제2 픽셀 데이터(110(1)(2))에 적어도 부분적으로 기초하여 재투영된 프레임(300(3))에 대한 제3 픽셀 데이터(110(1)(3))를 생성할 수 있다. 이 경우, 제2 프레임(300(2))은 재투영된 프레임(300(3)) 직전에 렌더링되는 프레임을 나타낸다. 예를 들어, 블록 504와 블록 506 사이에서, HMD(102)는 제2 프레임(300(2))에 대한 압축된 제2 픽셀 데이터(110(1)(2))를 수신하였고, 픽셀 데이터(110(1)(2))를 압축 해제하였고, HMD(102)의 업데이트된 포즈 예측을 처리하기 위해 픽셀 데이터(110(1)(2))를 수정하였고, 제2 프레임(300(2))에 대해 HMD(102) 상에 이미지를 제시하였고, 블록 506에서, HMD(102)는 재투영된 프레임(300(3))을 다음 프레임으로서 렌더링하기 위해 준비하고 있다.

[0061] 508에서, HMD(102)의 로직은 모션 벡터의 어레이(110(2)(2))에 적어도 부분적으로 기초하여 제3 픽셀 데이터(110(1)(3)')를 수정하여, 재투영된 프레임(300(3))에 대한 수정된 제3 픽셀 데이터(110(1)(3)')를 획득할 수 있다. 서브블록 510 및 512에 의해 도시된 바와 같이, 재투영된 프레임(300(3))에 대한 픽셀 데이터(110(1)(3))의 수정은 렌더 메쉬(400)를 활용할 수 있다.

[0062] 따라서, 510에서, HMD(102)의 로직은 재투영된 프레임(300(3))에 대한 렌더 메쉬(400)를 생성할 수 있다. 렌더 메쉬(400)는 복수의 정점들(402)을 갖는 바둑판 모양화된 메쉬를 포함할 수 있다. 일부 실시형태에서, 렌더 메쉬(400)의 해상도는, 모션 벡터(404)와 렌더 메쉬(400)의 "요소"(예를 들어, 렌더 메쉬(400)의 정점들(402), 렌더 메쉬(400)의 정사각형들(또는 쿼드들) 등과 같은 요소들) 사이에 일대일 대응이 있도록, 모션 벡터들의 어레이(110(2)(2))의 해상도와 정합할 수 있다. 모션 벡터 필드(110(2))와 렌더 메쉬(400) 사이의 정합 해상도를 획득하는 것은, GPU(304)가 모션 벡터 필드(110(2)(2))를 특정 해상도에서 출력하도록 요청하는 것, 모션 벡터 필드(110(2)(2))의 해상도를 다운 샘플링 또는 업 샘플링하는 것, 및/또는 GPU(304)에 의해서 출력된 모션 벡터 필드(110(2)(2))의 해상도와 정합하는 해상도에서 렌더 메쉬(400)를 생성하는 것과 같은 본원에 설명된 기술들

중 어느 하나를 포함할 수 있다.

[0063] 512에서, 로직은 렌더 메쉬(400)의 (복수의 정점들(402) 중) 정점들(402)을 이동된 정점들(408)인 렌더 메쉬(400) 내의 상이한 위치들로 이동시킬 수 있다. 정점들(402)은 (i) 넌-제로 모션 벡터들(404)의 방향으로, 그리고 (ii) 넌-제로 모션 벡터들(404)의 크기에 대응되는 양만큼 이동될 수 있다. 따라서, 블록 508에서 제3 픽셀 데이터(110(1)(3))의 수정은, 예를 들어, 제3 픽셀 데이터(110(1)(3))의 픽셀 값들을 이동된 정점들(408)에 따라 이동시켜 재투영된 프레임(300(3))의 수정된 제3 픽셀 데이터(110(1)(3)')를 획득함으로써, 렌더 메쉬(400)의 이동된 정점들(408)에 따라 될 수 있다. 예를 들어, 제3 픽셀 데이터(110(1)(3))의 픽셀 값은 렌더 메쉬(400) 내의 이동된 정점(408)에 따라 수정된 제3 픽셀 데이터(110(1)(3)') 내의 새로운 위치로, 좌측으로 4개의 픽셀들, 및 상향으로 4개의 픽셀들 이동될 수 있다.

[0064] 일부 실시형태에서, 다수의 모션 벡터 필드들(110(2))은 블록 502에서 입력된 이전에 렌더링된 프레임들(300)의 상이한 세트들에 기초하여 블록 504에서 수신될 수 있고, 추가 모션 관련 파라미터들은 재투영된 프레임에 대해 블록 508에서 모션 평활화에서 사용되도록 다수의 모션 벡터 필드들(110(2))에 기초하여 결정될 수 있다. 예를 들어, 블록 504에서 수신된 모션 벡터들의 어레이(110(2)(2))로 귀결되는 이전의 두 개의 프레임들(300(1) 및 300(2))에 추가하여, 도 5의 알고리즘은, 일부 실시형태에서, 예를 들어 제1 프레임(300(1)) 및 제1 프레임(300(1)) 이전에 렌더링된 "제0" 프레임(300(0))을 입력으로서 GPU(304)에 제공하고 해당 상이한 쌍의 입력 프레임들(300)에 기초하여 모션 벡터들의 추가 어레이(110(2))를 수신함으로써, 하나 또는 둘 이상의 추가 프레임들로 되돌아갈 수 있다. 다음으로, 모션 벡터들의 다수 어레이들은 프레임 간 이동하는 객체의 가속과 같은 모션 관련 파라미터들을 결정하기 위해 비교될 수 있고, 이러한 모션 관련 파라미터들은, 예를 들어 제3 픽셀 데이터(110(1)(3))에 적용되는 최종 모션 벡터(404)의 크기를 수정(예를 들어, 증가/감소)하여 모션 평활화 조정 관점에서 픽셀 값을 다소 이동시킴으로써, 블록 508에서 적용될 수 있다.

[0065] 514에서, HMD(102)의 로직은, 수정된 제3 픽셀 데이터(110(1)(3)')에 적어도 부분적으로 기초하여 디스플레이(예를 들어, HMD(102)의 디스플레이 패널(들)) 상에, 재투영된 프레임(300(3))을 렌더링할 수 있다. 블록 514에서 렌더링되는 결과적인 재투영된 프레임(300(3))은 GPU(304)로부터 수신된 모션 벡터들(110(2))에 따라 제3 픽셀 데이터(110(1)(3))의 수정으로 인해 "모션 평활화"된다. GPU(304)가 하나 또는 둘 이상의 GPU들(304)을 나타낼 수 있다는 것이 이해될 것이다. 예를 들어, 다수의 GPU들(304)은 HMD(102)의 스테레오 디스플레이 패널들 상에, 주어진 프레임(300)을 렌더링하기 위해 활용될 수 있고, 이러한 GPU들(304)에 입력되는 프레임(300)의 픽셀 데이터(110)는 이에 상응되게 파티션될 수 있다(예를 들어, 픽셀 데이터(110(1))의 좌측 절반부는 제1 GPU(304)에 입력으로서 제공될 수 있고, 픽셀 데이터(110(1))의 우측 절반부는 제2 GPU(304)에 입력으로서 제공될 수 있다).

[0066] 도 6은 본원에 개시된 실시형태에 따른, 픽셀 데이터가 모션 벡터 추정을 위해 GPU에 입력되기 전에, 이전에 렌더링된 프레임들의 픽셀 데이터를 정렬하기 위한 예시적인 프로세스(600)의 흐름도이다. 논의를 위해, 프로세스(600)는 이전 도면들을 참조하여 설명된다. 또한, 도 5 및 도 6에서 오프 페이지 참조 "A"에 의해 도시된 바와 같이, 프로세스(600)는 도 5의 블록 504에서의 작동들 이전에 수행되는 작동들을 나타낼 수 있고, 프로세스(500)는, 일부 실시형태에서, 블록 504 내지 블록 514의 작동들을 계속할 수 있다.

[0067] 602에서, 로직(예를 들어, 호스트 컴퓨터(106)의 로직)은, HMD(102)로부터 수신된 회전 데이터에 기초하여, 이전에 렌더링된 프레임들(300)(이들의 픽셀 데이터(110(1))가 GPU(304)에 입력으로서 제공될 것임)을 렌더링하는 사이에 HMD(102)가 제1 배향에서 제2 배향으로 회전되었다는 것을 결정할 수 있다. 예를 들어, 사용자(104)는, 제1 프레임(300(1))과 제2 프레임(300(2))을 렌더링하는 시간들에 대응될 수 있는 시간(t1)과 시간(t2) 사이에서 자신의 머리를 오른쪽 방향으로 회전했을 수 있다.

[0068] 604에서, 이전에 렌더링된 프레임들(300)은 해당 프레임들(300)에 대한 픽셀 데이터(110(1))가 호스트 컴퓨터(106)의 GPU(304)에 입력으로서 제공되기 전에 정렬될 수 있다. 이것은 프레임들(300) 사이에서 HMD(102)의 움직임으로 인해 이루어지며, 이는 객체(고정 및 움직이는 객체 모두)로 하여금 두 개의 프레임들(300(1) 및 300(2))을 렌더링하는 과정에서 위치들 사이를 이동하게 할 수 있다. 하나의 프레임(300(1))을 다른 프레임(300(2))과 정렬하거나 이와 반대로 정렬함으로써, 장면 내의 특정 정적 객체들을 나타내는 픽셀 데이터(110(1))는 일반적으로 두 개의 프레임들(300(1) 및 300(2)) 사이의 동일한 위치로 이동할 수 있어, 정적 객체들에 대응되는 픽셀 데이터(110(1))가 GPU(304)의 비디오 인코더에 의해 움직이는 객체들로 오인되지 않는다. 블록 604에서의 정렬은, (i) 제1 프레임(300(1))의 장면을 제2 프레임(300(2))을 렌더링하는 시간에 HMD(102)의 제2 배향과 방향과 정렬하는 수정된 제1 픽셀 데이터(610(1)(1))를 획득하기 위해 제1 픽셀 데이터(110(1)(1))

(제1 프레임(300(1))과 연관됨), 또는 (ii) 제2 프레임(300(1))의 장면을 제1 프레임(300(1))을 렌더링하는 시간에 HMD(102)의 제1 배향과 방향과 정렬하는 수정된 제2 픽셀 데이터를 획득하기 위해 제2 픽셀 데이터(110(1)(2))(제2 프레임(300(2))과 연관됨) 중 적어도 하나를 수정하는 것을 포함할 수 있다. 도 6의 하부의 도면은 전자의 경우를 도시하며, 여기서, 제1 픽셀 데이터(110(1)(1))가 HMD(102)의 제2 배향과 제1 프레임(300(1))의 장면을 정렬하는 수정된 제1 픽셀 데이터(610(1)(1))를 획득하기 위해 수정된다. 그러나, 제1 픽셀 데이터(110(1)(1)) 또는 제2 픽셀 데이터(110(1)(2))가 정렬 목적을 위해 수정될 수 있다는 것이 이해될 것이다. 블록 604에서의 정렬은 헤드 움직임으로 인한 모션 벡터들(110(2))의 델타를 감소시키기 위해 프레임들의 픽셀 데이터 중 하나 또는 둘 모두에 대한 회전 재투영 수정을 포함할 수 있다. 모션 벡터들(110(2))의 델타의 이러한 감소는, 본 명세서에 설명된 바와 같이 호스트 컴퓨터(106)로부터 HMD(102)로 모션 벡터들(110(2))을 전송할 때 감소된 대역폭의 추가 이점을 제공할 수 있다.

[0069] 606에서, 호스트 컴퓨터(106)의 로직은, 이전에 렌더링된 프레임(300) 및 다른 프레임(300)의 원래 픽셀 데이터(110(1)) 중 하나의 수정된 픽셀 데이터를 GPU(304)에 대한 입력으로서 제공할 수 있다. 도 6의 다이어그램에 도시된 실시예는 수정된 제1 픽셀 데이터(610(1)(1)) 및 원래의 제2 픽셀 데이터(110(1)(2))를 입력으로서 GPU(304)에 제공하는 것을 도시한다. 언급된 바와 같이, 프로세스(600)는 블록 606으로부터 프로세스(500)의 블록 504로 계속될 수 있다(오프 페이지 참조 "A"로 도시됨). 따라서, 모션 벡터들(110(2))의 어레이는 블록 606에서 입력으로서 GPU(304)에 제공되는 픽셀 데이터에 기초하여 블록 504에서 GPU(304)로부터 수신될 수 있고, 도 5의 모션 평활화 알고리즘의 잔여 작동들은 모션 평활화되는 재투영된 프레임(300(3))을 렌더링하기 위해 수행될 수 있다.

[0070] 도 7은 본원에 개시된 실시형태에 따른, 이전에 렌더링된 프레임들의 일부 픽셀 데이터를 배제하고, 픽셀 데이터의 잔여 부분을 모션 벡터 추정을 위해 GPU에 입력으로서 제공하기 위한 예시적인 프로세스(700)의 흐름도이다. 논의를 위해, 프로세스(700)는 이전 도면들을 참조하여 설명된다. 또한, 도 5 및 도 7에서 오프 페이지 참조 "A"에 의해 도시된 바와 같이, 프로세스(700)는 도 5의 블록 504에서의 작동들 이전에 수행되는 작동들을 나타낼 수 있고, 프로세스(500)는, 일부 실시형태에서, 블록 504 내지 블록 514의 작동들을 계속할 수 있다.

[0071] 702에서, 로직(예를 들어, 호스트 컴퓨터(106)의 로직)은, HMD(102)로부터 수신된 회전 데이터에 기초하여, 이전에 렌더링된 프레임들(300)(이들은 GPU(304)에 입력으로서 제공될 것임(예: 픽셀 데이터 압축용))을 렌더링하는 사이에 HMD(102)가 제1 배향에서 제2 배향으로 회전되었다는 것을 결정할 수 있다. 이전에 렌더링된 프레임들(300)은 도 3에 도시된 제1 프레임(300(1)) 및 제2 프레임(300(2))일 수 있다.

[0072] 704에서, 로직(예를 들어, 호스트 컴퓨터(106)의 로직)은 이용 가능한 픽셀 데이터(110(1))의 일 부분을 입력으로서 GPU(304)에 제공할 수 있다. GPU(304)에 입력으로서 제공되는 이용 가능한 픽셀 데이터(110(1))의 일 부분은, 예를 들어, 제1 픽셀 데이터(110(1)(1))의 일 부분(706(1)) 및 제2 픽셀 데이터(110(1)(2))의 부분(706(2))을 포함할 수 있으며, 이들 각각은 HMD(102)의 디스플레이 패널의 하나 또는 둘 이상의 에지들에서 픽셀들의 세브세트 이외의 픽셀들에 대응된다. 예를 들어, 도 7에 도시된 바와 같이, GPU(304)에 입력으로서 제공되는 픽셀 데이터(110(1))의 부분들(706)은 디스플레이 패널의 좌우 에지들에서 픽셀 데이터(110(1))의 잔여 부분을 배제한다(도 7에서 흑색으로 도시됨). 다시 말해서, 디스플레이 패널의 좌우 에지들에서의 픽셀 데이터(110(1))는 GPU(304)에 제공되지 않으므로, GPU(304)는 디스플레이 패널의 좌우 에지들 사이의 픽셀 데이터(110(1))의 중앙부(706)에 모션 추정 노력을 집중시킨다. 양 및/또는 음의 수직 방향으로 HMD(102) 회전의 경우, 픽셀 데이터(110(1))의 배제된 부분은 이미지의 상부 에지 및 하부 에지일 수 있다. 어느 경우든, 이미지의 에지(들)에서 픽셀 데이터(110(1))의 일 부분을 제외하는 것, 및 픽셀 데이터(110(1))의 잔여 부분(706)을 GPU(304)에 배타적으로 제공하는 것은, 이미지의 에지(들)에서 효과적으로 제로 모션 벡터들(404)로 귀결되며, 이는, 그렇지 않으면 이동 객체(302)의 특성이 아닌 이상치(outlier) 모션 벡터들(404)로 귀결될 수 있는 이미지의 에지들 근처에서 원치 않는 시각적 아티팩트가 나타나는 상황에서 유용할 수 있다. 언급된 바와 같이, 프로세스(700)는 블록 704로부터 프로세스(500)의 블록 504로 계속될 수 있으며, 이 경우 모션 벡터들(110(2))의 어레이는 블록 704에서 GPU(304)에 입력으로서 제공되는 픽셀 데이터에 기초하여 GPU(304)로부터 수신되고, 도 5의 모션 평활화 알고리즘은 모션 평활화되는 재투영된 프레임(300(3))을 렌더링하기 위해 수행될 수 있다.

[0073] 도 8은 본원에 개시된 실시형태에 따른, 이전에 렌더링된 프레임들의 픽셀 데이터를 정렬하고, 이전에 렌더링된 프레임들의 픽셀 데이터의 패널 마스크 부분을 배제하고, 픽셀 데이터의 잔여 부분을 모션 벡터 추정을 위해 GPU에 입력으로서 제공하기 위한 예시적인 프로세스(800)의 흐름도이다. 논의를 위해, 프로세스(800)는 이전 도면들을 참조하여 설명된다. 또한, 도 5 및 도 8에서 오프 페이지 참조 "A"에 의해 도시된 바와 같이, 프로세스(800)는 도 5의 블록 504에서의 작동들 이전에 수행되는 작동들을 나타낼 수 있고, 프로세스(500)는, 일부 실시

형태에서, 블록 504 내지 블록 514의 작동들을 계속할 수 있다.

[0074] 802에서, 로직(예를 들어, 호스트 컴퓨터(106)의 로직)은, HMD(102)로부터 수신된 회전 데이터에 기초하여, 이전에 렌더링된 프레임들(300)(이들은 GPU(304)에 입력으로서 제공될 것임)을 렌더링하는 사이에 HMD(102)가 제1 배향에서 제2 배향으로 회전되었다는 것을 결정할 수 있다. 이전에 렌더링된 프레임들은 도 3에 도시된 제1 프레임(300(1)) 및 제2 프레임(300(2))일 수 있다.

[0075] 804에서, 로직(예를 들어, 호스트 컴퓨터(106)의 로직)은, 제1 프레임(300(1))의 장면을 HMD(102)의 제2 배향과 정렬시키는 수정된 제1 픽셀 데이터(808(1))를 획득하기 위해 제1 프레임(300(1))과 연관된 제1 픽셀 데이터(110(1)(1))를 수정할 수 있다. 도 8의 실시예에서, 제1 픽셀 데이터(110(1)(1))의 일 부분은 각각의 프레임에서 HMD(102)의 디스플레이 패널의 주변에서 렌더링된 패널 마스크(흑색으로 도시됨)에 대응되는 데이터를 나타낸다. 따라서, 수정된 제1 픽셀 데이터(808(1))의 패널 마스크 부분(814(1))(즉, 도 8에서 검은색으로 표시된 부분)은 수정된 제1 픽셀 데이터(808(1))에서 패널 마스크에 대응되는 데이터를 나타낸다. 언급된 바와 같이, 애플리케이션은, 프레임들을 렌더링할 때 추가 픽셀 데이터를 생성하도록 지시받을 수 있으며, 추가 픽셀 데이터는 패널 마스크에 대응된다. 이러한 방식으로, 패널 마스크를 적용할 때 FOV의 유효한 감소가 없다.

[0076] 806에서, 로직(예를 들어, 호스트 컴퓨터(106)의 로직)은, 제2 프레임의 장면을 HMD(102)의 제1 배향과 정렬시키는 수정된 제2 픽셀 데이터(808(2))를 획득하기 위해 제2 프레임(300(2))과 연관된 제2 픽셀 데이터(110(1)(2))를 수정할 수 있다. 다시, 제2 픽셀 데이터(110(1)(2))의 일 부분은 각각의 프레임에서 HMD(102)의 디스플레이 패널의 주변에서 렌더링된 패널 마스크(흑색으로 도시됨)에 대응되는 데이터를 나타낸다. 따라서, 수정된 제2 픽셀 데이터(808(2))의 패널 마스크 부분(814(2))(즉, 도 8에서 검은색으로 표시된 부분)은 수정된 제2 픽셀 데이터(808(2))에서 패널 마스크에 대응되는 데이터를 나타낸다.

[0077] 810에서, 로직(예: 호스트 컴퓨터(106)의 로직)은 수정된 제1 픽셀 데이터(808(1))의 패널 마스크 부분(814(1))을 수정된 제2 픽셀 데이터(808(2))의 패널 마스크 부분(814(2))과 조합하여, 패널 마스크(816)에 대응되는 공통 값을 갖는 픽셀들의 서브세트를 결정할 수 있다. 이것은 일종의 벤 다이어그램으로 생각될 수 있으며, 이 경우 패널 마스크(816)에 대응되는 픽셀들의 서브세트는 수정된 제1 픽셀 데이터(808(1)) 및 수정된 제2 픽셀 데이터(808(2))에서 수정된 패널 마스크의 조합된 버전이다.

[0078] 812에서, 로직(예: 호스트 컴퓨터(106)의 로직)은, GPU(304)에 대한 입력으로서, 제1 픽셀 데이터(110(1)(1))의 특정 부분(예를 들어, 중앙부) 및 제2 픽셀 데이터(110(1)(2))의 특정 부분(예를 들어, 중앙부)을 제공할 수 있으며, 이들 각각은 패널 마스크(816)에 대응되는 픽셀들의 세브세트 이외의 픽셀들에 대응한다. 이를 통해 GPU(304)는 패널 마스크(816) 픽셀들에 의해 커버되는 각각의 이전에 렌더링된 프레임에서 픽셀 데이터(110(1))의 일 부분에 대한 모션 추정을 무시할 수 있고, GPU(304)는 모션 추정 노력을 각각의 프레임(300)에 대한 픽셀 데이터(110(1))의 중앙의 비패널 마스크 부분에 집중시킬 수 있다. 언급된 바와 같이, 프로세스(800)는 블록 812로부터 프로세스(500)의 블록 504로 계속될 수 있으며, 이 경우 모션 벡터들(110(2))의 어레이는 블록 812에서 GPU(304)에 입력으로서 제공되는 픽셀 데이터에 기초하여 GPU(304)로부터 수신되고, 도 5의 모션 평활화 알고리즘은 모션 평활화되는 재투영된 프레임(300(3))을 렌더링하기 위해 수행될 수 있다.

[0079] 도 9은 본원에 개시된 실시형태에 따른, 모션 평활화 기술의 일부로서 재투영된 프레임을 생성하기 위해 사용되기 전에 GPU로부터 출력되는 모션 벡터들을 임계화하기 위한 예시적인 프로세스(900)의 흐름도이다. 논의를 위해, 프로세스(900)는 이전 도면들을 참조하여 설명된다.

[0080] 902에서, 로직(예를 들어, 호스트 컴퓨터(106)의 로직)은 이전에 렌더링된 프레임들(300)과 연관된 픽셀 데이터(110(1))를 입력으로서 GPU(304)에 제공할 수 있다. 블록 902에서 수행된 작동(들)은 프로세스(500)의 블록 502와 관련하여 설명된 것과 유사할 수 있다.

[0081] 904에서, 로직(예를 들어, 호스트 컴퓨터(106)의 로직)은 GPU(304)로부터 모션 벡터들(110(2))의 어레이를 수신할 수 있다. 블록 904에서 수행된 작동(들)은 프로세스(500)의 블록 504와 관련하여 설명된 것과 유사할 수 있다.

[0082] 906에서, 로직(예를 들어, 호스트 컴퓨터(106)로부터 모션 벡터들(110(2))을 수신한 후 HMD(102)의 로직)은 모션 벡터들(110(2))의 어레이에서 개별 모션 벡터들의 크기들(또는 길이들)을 제1 임계 크기와 비교하여, 제1 임계 크기보다 더 큰 크기를 갖는 모션 벡터들(110(2))의 제1 서브세트를 결정할 수 있다. 제1 임계 크기는, 가상 객체(302)의 이동 또는 애니메이션 이외의 것인 프레임들 간의 변화를 나타낼 수 있는, 큰 크기의 이상치 모션 벡터들(110(2))의 이러한 서브세트의 영향을 완화시키기 위해서 활용될 수 있다. 비정상적으로 큰 크기의 모션

벡터들은 다양한 이유들로 발생될 수 있다. GPU(304)는 제2 프레임(300(2))의 상부 우측에서의 픽셀이 제1 프레임(300(1))의 하부 좌측에 있는 픽셀과 충분히 유사하다는 것을 찾고, 이러한 모션 벡터(404)가 프레임들(300) 사이의 가상 객체(302)의 이동 또는 애니메이션을 나타내지 않더라도, 어레이에서 다른 모션 벡터들(404)에 비해 상대적으로 큰 크기를 갖는 결과적인 모션 벡터(404)를 출력할 수 있다. 일부 경우에, 비디오 게임은, 프레임들(300) 사이에서 장면들이 급격하게 변경되며 큰 모션 벡터들(404)이 GPU(304)의 비디오 인코더에 의해 생성되게 하는 상이한 위치로 사용자가 순간이동(teleport)하는 것을 허용할 수 있다. 이러한 경우 및 다른 경우에, 이러한 큰 모션 벡터들(404)을 임계치화(threshold)하는 것이 유용할 수 있다.

[0083] 907에서, 다수의 이전에 렌더링된 프레임들(300)이 제1 프레임(300(1)) 및 제2 프레임(300(2))을 포함할 때, 로직(예를 들어, HMD(102)의 로직)은 제1 프레임(300(1))이 렌더링된 제1 시간과 제2 프레임(300(2))이 렌더링된 제2 시간 사이의 시간 기간(또는 갭)을 결정할 수 있다.

[0084] 909에서, 블록 906에서의 비교에 사용되는 제1 임계치 크기는 제1 시간과 제2 시간 사이의 시간 기간에 적어도 부분적으로 기초하여 선택될 수 있다. 예시적인 예에서, 사용자의 관점으로부터 단위 시간당 이동의 디그리(degree)로 측정된 제1 임계 크기는 두 개의 프레임들(300(1) 및 300(2)) 사이의 11.11 ms 시간당 6 디그리의 이동이다. 따라서, 두 개의 프레임들(300(1) 및 300(2))이 시간 상 더 길게 떨어져 있을수록, 제1 임계 크기가 더 커지고, 이에 따라 모션 벡터들(110(2))은 더 길어지는 것(크기에서)이 허용된다. 이것은, 특정 속도를 초과하는 모션이 있으면 모션 평활화가 그다지 효과적이지 않을 수 있고, 실제로 (예를 들어, 장면의 일부를 "스위미(swimmy)"하게 보이게 함으로써) 부정적인 시각적 아티팩트를 유발할 수 있기 때문이다.

[0085] 908에서, 로직(예를 들어, HMD(102)의 로직)은 제1 임계 크기보다 더 큰 크기를 갖는 것으로 결정된 모션 벡터들(110(2))의 제1 서브세트의 크기들을 감소시켜, 크기들이 제1 임계 크기에서 캡핑(capping)된다. 이것은 모션 벡터들(110(2))의 제1 수정된 어레이를 생성한다. 다시 말해서, 제1 임계 크기를 초과하는 해당 모션 벡터들(110(2))의 경우, 로직은 이러한 모션 벡터들(110(2))의 크기를 제1 임계 크기로 캡핑(또는 제한)하도록 구성되어, 모션 벡터들(110(2))의 제1 수정된 어레이가 제1 임계 크기 이하인 크기를 포함하고, 제1 임계 크기보다 더 큰 어떠한 크기도 포함하지 않는다. 일부 실시형태에서, 모션 벡터들(110(2))의 크기들을 제1 임계 크기에서 캡핑하고 모션 평활화를 위해 모든 넌-제로 모션 벡터들을 사용하는 대신, 로직은 제1 임계 크기를 충족하거나 초과하는 크기들을 갖는 모션 벡터들(110(2))의 제1 서브세트를 폐기하여 이들이 모션 평활화에서 사용되지 않을 수 있다.

[0086] 910에서, 로직(예를 들어, HMD(102)의 로직)은 모션 벡터들(110(2))의 어레이에서 개별 모션 벡터들의 크기들(또는 길이들)을 제2 임계 크기와 비교하여 제2 임계 크기보다 작은 크기를 갖는 모션 벡터들(110(2))의 제2 서브세트를 결정할 수 있다. 제2 임계 크기는, 사용자의 일정한 머리 움직임 및/또는, 정확하지만, 절대 제로 모션의 정도까지는 정확하지 않은 머리 트랙킹에 의해 종종 야기되는 프레임들 간의 변화들을 나타낼 수 있는, 작은 크기의 이상치 모션 벡터들(404)의 이러한 서브세트의 영향을 완화시키기 위해서 활용될 수 있다. 이 때문에, GPU(304)의 출력은 제로 길이 모션 벡터(404)를 거의 제공하지 않는다. 오히려, GPU(304)의 출력이 상술된 바와 같이, 주변 머리 모션 및/또는 트래킹 지터(tracking jitter)로 인해 상당한 양의 노이즈를 갖는 것이 더 일반적이다. 다시 말해서, 두 개의 연속적인 프레임들 사이의 픽셀들이 100% 일치하는 경우는 거의 없다. 예시적인 예에서, 모션의 픽셀들로 측정된 제2 임계 크기는 모션의 2개의 픽셀들 임계치이다. 이러한 예에서, 크기들(또는 길이들)이 모션의 2개의 픽셀들 미만인 임의의 모션 벡터들은 작은 크기의 이상치 벡터들로 간주된다.

[0087] 912에서, 로직(예를 들어, HMD(102)의 로직)은 제2 임계 크기보다 작은 크기를 갖는 것으로 결정된 모션 벡터들(110(2))의 제2 세브세트(예를 들어, 모션의 2개의 픽셀들 미만의 길이들을 갖는 작은 크기의 이상치 모션 벡터들)의 크기들을 제로의 길이/크기로 감소시킬 수 있다. 이것은 제2 임계 크기보다 더 작은 크기를 갖는 어떠한 모션 벡터도 포함하지 않는 모션 벡터들(110(2))의 제2 수정된 어레이를 생성한다. 이것은 본원에서 때때로 모션 벡터들(110(2))에 작은 데드 존(dead-zone)을 적용하는 것으로 지칭된다.

[0088] 914에서, 로직(예를 들어, HMD(102)의 로직)은 제2 임계치보다 작은 크기를 갖는 어떠한 모션 벡터(404)도 없이 제1 임계 크기에서 캡핑된 크기를 갖는 모션 벡터들(110(2))의 제2 수정된 어레이에 기초하여 재투영된 프레임에 대한 픽셀 데이터(110(1)(3))를 수정할 수 있다. 일부 실시형태에서, 예를 들어, 모션 벡터들의 제1 서브세트가 제1 임계 크기에서 캡핑되는 대신 폐기될 때, 모션 벡터들(110(2))의 폐기된 제1 서브세트 이외의 모션 벡터들(110(2))의 잔여 서브세트는 블록 914에서 재투영된 프레임에 대한 픽셀 데이터(110(1)(3))를 수정하기 위해서 사용된다. 픽셀 데이터(110(1)(3))의 수정은 본원에 설명된 기술들(예를 들어, 프로세스(500)의 블록 508

내지 블록 512를 참조하여 설명된 기술들) 중 어느 하나를 포함할 수 있다.

[0089] 916에서, 로직(예를 들어, HMD(102)의 로직)은, 재투영된 프레임(300(3))에 대한 수정된 픽셀 데이터(110(1)(3)')에 적어도 부분적으로 기초하여 디스플레이(예를 들어, HMD(102)의 디스플레이 패널(들)) 상에, 재투영된 프레임(300(3))을 렌더링할 수 있다. 블록 916에서 수행된 작동(들)은 프로세스(500)의 블록 514와 관련하여 설명된 것과 유사할 수 있다.

[0090] 도 10은 본원에 개시된 실시형태에 따른, 모션 평활화 기술의 일부로서 재투영된 프레임을 생성하기 위해 사용되기 전에 GPU로부터 출력되는 모션 벡터들을 감쇠하기 위한 예시적인 프로세스(1000)의 흐름도이다. 논의를 위해, 프로세스(1000)는 이전 도면들을 참조하여 설명된다.

[0091] 1002에서, 로직(예를 들어, 호스트 컴퓨터(106)의 로직)은 이전에 렌더링된 프레임들(300)과 연관된 픽셀 데이터(110(1))를 입력으로서 GPU(304)에 제공할 수 있다. 블록 1002에서 수행된 작동(들)은 프로세스(500)의 블록 502와 관련하여 설명된 것과 유사할 수 있다.

[0092] 1002에서, 로직(예를 들어, 호스트 컴퓨터(106)의 로직)은 GPU(304)로부터 모션 벡터들(110(2))의 어레이를 수신할 수 있다. 블록 1004에서 수행된 작동(들)은 프로세스(500)의 블록 504와 관련하여 설명된 것과 유사할 수 있다.

[0093] 1006에서, 로직(예를 들어, 호스트 컴퓨터(106)로부터 모션 벡터들(110(2))을 수신한 후 HMD(102)의 로직)은 (예를 들어, 해당 모션 벡터들(110(2))의 품질에 대한 신뢰도가 낮은 경우) 모션 벡터들(110(2)) 중 개별적인 것들의 크기들을 감쇠(예를 들어, 축소, 감소 등)할 목적으로 모션 벡터들(110(2))의 어레이에 오버레이된 감쇠 텍스처를 생성할 수 있다. 감쇠 텍스처는, 감쇠 텍스처가 복수의 텍셀들(예를 들어, 텍셀들의 그리드)을 포함하도록 임의의 적절한 해상도에서 생성될 수 있다. 감쇠 텍스처의 해상도는, 다수의 모션 벡터들이 감쇠 텍스처의 개별 텍셀 내에 있도록, 모션 벡터들(110(2))의 어레이의 해상도보다 더 낮은 해상도일 수 있다. 일 예에서, 감쇠 텍스처는, 사용자의 시야(FOV) 중앙에서의 텍셀이 약 6디그리 x 6 디그리(수평 및 수직)이도록, 해상도를 가질 수 있다. 텍셀당 해상도(수평 및 수직)는, 텍셀당 디그리들의 수가 감쇠 텍스처에 걸쳐 일정하지 않을 수 있기 때문에, 대략적일 수 있다. 예를 들어, HMD(102)의 FOV의 영역은 대략 100 디그리 x 100 디그리(예를 들어, 플러스 또는 마이너스 약 10 디그리)에 걸쳐 있을 수 있다. 감쇠 텍스처는 비선형 투영 매트릭스에 기초하여 생성될 수 있으므로, 해상도는 감쇠 텍스처의 텍셀마다 가변되지만, 해상도는 대략 6-10 디그리 x 6-10 디그리인 텍셀들을 제공한다. 또한, 감쇠 텍스처의 각각의 텍셀에는 (예를 들어, 0.0 및 1.0의 범위 내에서) 감쇠 값이 할당될 수 있다. 주어진 텍셀에 할당된 감쇠 값은 해당 텍셀에서 모션 벡터들(404)의 크기들이 감소(축소)되는 양을 제어한다. 예를 들어, 1.0의 감쇠 값은 감쇠가 없는 것에 해당할 수 있고, 이에 따라, 1.0의 감쇠 값을 모션 벡터(404)에 적용하는 것은 모션 벡터(404)가 크기의 관점에서, 있는 그대로(예를 들어, 감소되지 않고, 축소되지 않는 등) 남아 있는 것을 의미한다. 그러나, 0.0의 감쇠 값은 전체 감쇠에 해당할 수 있고, 이에 따라 0.0의 감쇠 값을 모션 벡터(404)에 적용하는 것은 모션 벡터(404)가 크기 관점에서 제로로 감소되는 것을 의미한다. 0.0과 1.0 사이의 감쇠 값은 부분 감쇠에 해당할 수 있고, 이에 따라, 예를 들어, 0.5의 감쇠 값을 모션 벡터(404)에 적용하는 것은 모션 벡터가 원래 길이(크기)의 50%(절반)로 감소되는 것을 의미한다. 따라서, 감쇠 텍스처는 모션 벡터(404)의 크기를 감소(축소)시킬 수 있지만, 모션 벡터(404)의 크기를 증가(늘림)시킬 수는 없다.

[0094] 도 10은 2개의 연속적으로 렌더링된 프레임들(300) 사이의 화면의 하단에서의 휘도(또는 색상)의 급격한 변화에 기초하여 블록 1006에서 생성될 수 있는 예시적인 감쇠 텍스처(1007)를 도시한다. 예를 들어, 비디오 게임을 플레이하는 동안, HMD(102)의 사용자는 다음 렌더링된 프레임(300) 상의 화면의 하부 절반부에 큰 청색 메쉬가 나타나게 하는 핸드헬드 제어기 상의 버튼을 선택할 수 있다. 이러한 예시적인 시나리오에서, GPU(304)는, 이러한 프레임(300) 및 이전에 렌더링된 프레임(300)을 입력으로서 처리할 때, 화면의 하부 절반부에 있는 큰 청색 메쉬에서의 픽셀들의 휘도 값들과 최상으로 정합하는 휘도 값들로 장면의 상부에 렌더링되는 스카이에서 픽셀들을 찾을 수 있다. 이것은, 갑자기 크기가 매우 크고 장면에서 실제 모션을 나타내지 않는 장면의 하부에 모션 벡터들(404)을 갖는 모션 벡터 필드(110(2))를 GPU(304)가 출력하는 것으로 귀결된다. 예시적인 감쇠 텍스처(1007)에서, 백색 텍셀들은 1.0의 감쇠 값을 나타낼 수 있는 반면, 흑색 텍셀들은 0.0의 감쇠 값을 나타낼 수 있고, 회색 텍셀들의 음영들은 0.0과 1.0 사이(경계값 제외)의 감쇠 값들을 나타낼 수 있다.

[0095] 1008에서, 로직(예를 들어, HMD(102)의 로직)은 블록 1006에서 생성된 감쇠 텍스처를 사용하여 모션 벡터들(110(2))의 어레이에서 개별 모션 벡터들(404)의 크기들을 감소(예를 들어, 스케일 다운)시킬 수 있다(예를 들어, 감쇠 텍스처의 감쇠 값들을 사용하여 감쇠 값들에 대응되는 양들만큼 모션 벡터들(110(2))의 크기들을 감소

시킴). 이것은, 감쇠 텍스처에서 1.0 미만의 적어도 하나의 감쇠 값을 취하여, 모션 벡터들(110(2))의 수정된 어레이를 생성한다. 목적은 프레임들(300) 사이의 임계량보다 더 크게 변경된 모션 벡터들(404)을 스케일 다운하는 것이며, 이는 이러한 모션 벡터들(404)이 장면 내의 실제 모션을 나타내지 않고, 모션을 나타내는 것으로 오인되지 않아야 한다는 것을 나타낼 가능성이 크다. 감쇠 텍스처는 화면의 영역들의 모션 벡터들이 단일 프레임에서 (크기의 측면에서) 극단적인 이러한 영역들을 찾기 위해서 사용되며, 이는 모션 벡터들이 실제 모션을 나타내는 것으로 신뢰될 수 없거나, 또는 만약 이들이 장면 내에서 실제 모션을 나타내는 경우, 모션 평활화가 어차피 눈에 띄는 영향을 미치기에는 이러한 모션이 너무 빠르고, 해당 모션 벡터들을 사용하여 모션 평활화를 시도하는 것보다 제로 모션 벡터를 갖는 것이 바람직하다는 것을 나타낸다. 현실적으로, 프레임 간 모션이 있을 때마다, 디스플레이의 재생률이 통상적으로 90 Hz 정도이고 프레임들 사이의 짧은 시간 내에 무엇이든 매우 멀리 이동하기가 어렵기 때문에, 모션은 통상적으로 이전 프레임과 크게 다르지 않다. 따라서, 감쇠 텍스처에 의해 감쇠된 이러한 극단적인 모션 벡터들(404)은 종종 실제 모션을 나타내지 않는다.

[0096] 서브블록 1009에 의해 도시된 바와 같이, 일부 실시형태에서, 모션 벡터들의 크기들을 감소시키기 위해 블록 1008에서 감쇠 텍스처를 사용하는 단계는 감쇠 텍스처를 사용하기 전에 감쇠 텍스처에 최소 N x N 필터를 적용하여 모션 벡터들의 크기들을 감소시키는 단계를 포함할 수 있다. 여기서, N은 임의의 적절한 수일 수 있다. N = 3은 최소 3 x 3 필터(즉, 3 x 3 텍셀 블록)를 사용한다. N = 3의 예를 사용하여, 블록 1009에서 감쇠 텍스처에 최소 3 x 3 필터를 적용할 때, 로직(예를 들어, HMD(102)의 로직)은, 감쇠 텍스처(1007)로부터 텍셀 기준으로, 해당 텍셀을 중심으로 한 각각의 3 x 3 텍셀 블록(예를 들어, 대응되는 텍셀 및 이의 8개의 인접 텍셀들을 포함하는 텍셀 블록)에서 9개의 감쇠 값들 중 최소 값을 페칭할 수 있다. 블록 1009에서, 이러한 최소 N x N 필터를 적용하는 것은, 감쇠 텍스처의 임의의 대응되는 텍셀(넌-제로 감쇠 값을 할당 받음, 그러나 제로의 감쇠 값이 할당된 주변 텍셀을 가짐)이 제로의 감쇠 값을 해당 대응되는 텍셀의 대응되는 모션 벡터들에 적용할 것이기 때문에, 훨씬 더 많은 모션 벡터들(404)을 제로화하는 것으로 귀결되며, 이로써, 감쇠 텍스처(1007)에서 넌-제로 감쇠 값이 대응되는 텍셀에 할당되었음에도 불구하고, 대응되는 텍셀에서 모션 벡터를 제로로 감소시킨다. 다시 말해서, 감쇠 텍스처의 주어진 텍셀에서 모션 벡터들(404)은, 인접 텍셀에 제로의 감쇠 값이 할당되는 경우, 제로화되며, 이는 인접 텍셀이 수많은 큰 크기의 모션 벡터들을 갖는다는 것을 의미한다. 도 10은 감쇠 텍스처(1007)에 최소 N x N 필터를 적용한 결과인 예시적인 감쇠 텍스처(1011)를 도시한다. 결과적인 감쇠 텍스처(1011)는 감쇠 텍스처(1007)보다 더 많은 흑색 텍셀들을 포함하며, 이는 결과적인 감쇠 텍스처(1011)를 사용하여 더 많은 모션 벡터들이 제로화되는 것을 의미한다.

[0097] 1010에서, 로직(예를 들어, HMD(102)의 로직)은, 아마도 감쇠 텍스처에서 1.0 미만의 감쇠 값에 대해 감쇠된 크기들의 일부를 갖는 모션 벡터들(110(2))의 수정된 어레이에 기초하여 재투영된 프레임에 대한 픽셀 데이터(110(1)(3))를 수정할 수 있다. 픽셀 데이터(110(1)(3))의 수정은 본원에 설명된 기술들(예를 들어, 프로세스(500)의 블록 508 내지 블록 512를 참조하여 설명된 기술들) 중 어느 하나를 포함할 수 있다.

[0098] 1012에서, 로직(예를 들어, HMD(102)의 로직)은, 재투영된 프레임(300(3))에 대한 수정된 픽셀 데이터(110(1)(3)')에 적어도 부분적으로 기초하여 디스플레이(예를 들어, HMD(102)의 디스플레이 패널(들)) 상에, 재투영된 프레임(300(3))을 렌더링할 수 있다. 블록 1012에서 수행된 작동(들)은 프로세스(500)의 블록 514와 관련하여 설명된 것과 유사할 수 있다.

[0099] 도 11은 본원에 개시된 실시형태에 따른, 모션 평활화 기술의 일부로서 재투영된 프레임을 생성하기 위해 사용되기 전에 GPU로부터 출력되는 모션 벡터들을 감쇠하기 위해서 사용될 수 있는 감쇠 텍스처를 생성하기 위한 예시적인 프로세스(1100)의 흐름도이다. 따라서, 프로세스(1100)는 프로세스(1000)의 블록 1006에서 감쇠 텍스처를 생성하기 위해 수행될 수 있는 작동들의 서브프로세스를 나타낼 수 있다. 논의를 위해, 프로세스(1100)는 이전 도면들을 참조하여 설명된다.

[0100] 1102에서, 호스트 컴퓨터(106)로부터 모션 벡터들(110(2))의 어레이를 수신한 후, HMD(102)의 로직은 모션 벡터들(110(2))의 다수의 이전에 획득된 어레이에 적어도 부분적으로 기초하여 감쇠 텍스처의 각각의 텍셀에 대한 델타 벡터들의 세트를 결정할 수 있다. 예를 들어, 모션 벡터들(110(2))의 어레이의 이력은 일련의 프레임들이 처리됨에 따라 유지될 수 있고, 모션 벡터들(110(2))의 마지막 두 개의 어레이들(예를 들어, 모션 벡터들(110(2))의 가장 최근의 두 개의 어레이들)이 비교되어 블록 1102에서 델타 벡터들의 세트(예를 들어, 두 개의 어레이들에서 한 쌍의 대응되는 모션 벡터들 사이의 차이(예를 들어, 크기의 측면에서)를 나타내는 차이 벡터들)를 결정할 수 있다.

[0101] 1104에서, 로직(예를 들어, HMD(102)의 로직)은 임계 크기보다 더 큰 크기를 갖는 감쇠 텍스처의 각각의 텍셀에

서 델타 벡터들의 백분율을 계산할 수 있다. 예를 들어, 블록 1104에서, 로직은 감쇠 텍스처의 주어진 텍셀에서 델타 벡터들을 평가하고, 텍셀에서 "이상치" 벡터들을 임계 크기보다 더 큰 델타(예를 들어, 크기)를 갖는 해당 델타 벡터들로서 식별하고, 텍셀에서 이상치 벡터들의 백분율을 계산할 수 있다. 프레임당(또는 11.11 ms 시간당) (사용자의 관점으로부터) 약 3 디그리의 이동/모션의 임계치와 같은 임의의 적절한 임계치는 블록 1104에서 "이상치" 벡터들을 식별할 목적을 위해서 사용될 수 있다. 다시 말해서, 블록 1104에서 사용되는 임계치는, 도 9의 프로세스(900)의 블록 907 및 블록 909를 참조하여 전술한 바와 같이, 제1 프레임(300(1))이 렌더링된 제1 시간과 제2 프레임(300(2))이 렌더링된 제2 시간 사이의 시간 기간(또는 갭)에 기초할 수 있다.

[0102] 1106에서, 로직(예를 들어, HMD(102)의 로직)은, 감쇠 텍스처의 각각의 텍셀에 대해, "이상치" 델타 벡터들인 텍셀에서 델타 벡터들의 백분율이 25%와 같은 임계 백분율을 충족하거나 초과하는지 여부를 결정할 수 있다. "이상치" 델타 벡터들인 텍셀에서 델타 벡터들의 백분율이 블록 1106에서 임계 백분율을 충족하거나 초과하면, 프로세스(1100)는 블록 1106으로부터 블록 1108로의 "예(YES)" 경로를 따를 수 있다.

[0103] 1108에서, 로직(예를 들어, HMD(102)의 로직)은 임계 백분율을 충족하거나 초과하는 다수의 "이상치" 델타 벡터들을 갖는 텍셀들에 대해 감쇠 값을 제로로 계산할 수 있다. 즉, 로직은 주어진 텍셀에 대한 감쇠 값을, 해당 텍셀에 대한 델타 벡터들의 특정 양(예를 들어, 25% 이상)이 최대 길이보다 더 큰 경우에, 제로로 계산할 수 있다. 프로세스(1000)에 설명된 바와 같이, 제로의 감쇠 값을 주어진 텍셀 내의 모션 벡터들(404)에 적용하는 것은 이러한 모션 벡터들(404)의 크기들을 제로로 감소시킨다(예를 들어, 해당 텍셀 내의 모션 벡터들을 완전히 감쇠시키도록).

[0104] 1106에서, "이상치" 델타 벡터들의 임계 백분율 미만을 갖는 텍셀의 경우, 프로세스(1100)는 블록 1106으로부터 블록 1110으로의 "아니오(NO)" 경로를 따를 수 있으며, 블록 1110에서, 로직은 블록 1104에서 계산된 백분율에 적어도 부분적으로 기초하는 감쇠 값으로서 해당 텍셀들에 대한 감쇠 값을 계산할 수 있다. 예를 들어, 주어진 텍셀에 대한 임계 크기보다 큰 크기들을 갖는 델타 벡터들의 백분율이 임계 백분율(예를 들어, 25%)보다 더 작은 경우, 감쇠 값은 임계 백분율보다 더 작은 값들의 범위 내에서(예를 들어, 0%-25%의 범위 내에서) 계산된 백분율에 선형으로 매핑되는 값으로서 설정될 수 있다. 블록 1110에서 계산된 감쇠 값은 (감쇠 텍스처가 모션 벡터 필드(110(2))에 적용될 때) 제로화되지 않은 잔여 벡터들에서 노이즈를 감소시키기 위해서 효과적이다.

[0105] 1112에서, 계산된 감쇠 값은 이전에 렌더링된 프레임(300)에 기초하여 해당 텍셀에 대한 기존 감쇠 값과 비교될 수 있다. 감쇠 값이 블록 1112에서 기존 값보다 더 작으면, 프로세스(1100)는 블록 1112로부터 블록 1114로의 "예(YES)" 경로를 따를 수 있으며, 블록 1114에서, 텍셀에 대한 감쇠 값이 블록 1108 또는 블록 1110에서 계산된 감쇠 값으로 설정된다. 예를 들어, 이전 프레임에서 감쇠 텍스처의 해당 텍셀에 대한 감쇠가 없었던 경우, 기존 감쇠 값은 1.0과 같을 수 있었고, 블록 1108 또는 블록 1110에서 계산된 새로운 감쇠 값이 1.0 미만인 경우, 이러한 예에서, 감쇠 값은 해당하는 새롭게 계산된 감쇠 값으로 설정되어 주어진 텍셀에서 모션 벡터들을 감쇠시킨다.

[0106] 1112에서, 계산된 감쇠 값이 논의되는 텍셀에 대한 기존 값보다 작지 않은 경우, 프로세스(1100)는 블록 1112로부터 블록 1116으로의 "아니오(NO)" 경로를 따를 수 있으며, 블록 1116에서, 기존 감쇠 값은 감쇠 없음에 대응되는 최대 감쇠 값(예를 들어, 1.0)을 향해 시간이 지남에 따라 증가된다. 예를 들어, 일부 상황은 프레임 간 변경되는 불규칙한 모션 벡터들을 야기하기 때문에, 새로운 감쇠 값들은 감쇠 텍스처의 텍셀들에 대한 이전(구) 감쇠 값들에 대하여 "블렌딩"될 수 있으며, 이는, 계산된 감쇠 값이 해당 기간 동안에 걸쳐 텍셀에 대한 기존 감쇠 값보다 작지 않다고 가정하면서, 감쇠 값이 일정 기간(예를 들어, 1초 동안)에 걸쳐 점진적으로 무감쇠를 향해 증가되는 것을 의미한다.

[0107] 도 12는 본원에 개시된 실시형태에 따른, 모션 평활화 기술의 일부로서 재투영된 프레임을 생성하기 위해 모션 벡터들의 결과 세트가 사용되기 전에 색상 변화가 거의 또는 전혀 없는 영역에 대응되는 모션 벡터들을 제로화하기 위한 예시적인 프로세스(1200)의 흐름도이다. 논의를 위해, 프로세스(1200)는 이전 도면들을 참조하여 설명된다.

[0108] 1202에서, 로직(예를 들어, 호스트 컴퓨터(106)의 로직)은 이전에 렌더링된 프레임들(300)과 연관된 픽셀 데이터(110(1))를 입력으로서 GPU(304)에 제공할 수 있다. 블록 1202에서 수행된 작동(들)은 프로세스(500)의 블록 502와 관련하여 설명된 것과 유사할 수 있다.

[0109] 1204에서, 로직(예를 들어, 호스트 컴퓨터(106)의 로직)은 GPU(304)로부터 모션 벡터들(110(2))의 어레이를 수신할 수 있다. 블록 1204에서 수행된 작동(들)은 프로세스(500)의 블록 504와 관련하여 설명된 것과 유사할 수

있다.

[0110] 1206에서, 로직(예를 들어, 호스트 컴퓨터(106)로부터 모션 벡터들(110(2))을 수신한 후 HMD(102)의 로직)은 GPU(304)에 입력된 다수의 이전에 렌더링된 프레임들(300) 사이의 (RGB 벡터들의 측면에서) 컬러 픽셀 값들을 비교하여 모션 벡터들(110(2))의 어레이에서 모션 벡터당 색상 차이 값을 결정할 수 있다. 이러한 색상 차이 값들은 임계 색상 차이와 비교되어, 이들의 색상 차이 값들이 임계 색상 차이보다 더 작기 때문에 프레임들 간의 색상 변화가 거의 또는 전혀 없는 것과 연관된 모션 벡터들(110(2))의 서브세트를 결정할 수 있다. 일부 실시예에서, 로직은 RGB 픽셀 값을 벡터로서 취급한다. 그 후, 이전에 렌더링된 프레임들(300) 사이의 대응되는 RGB 벡터들이 색상 차이(예를 들어, 델타 벡터)를 결정하기 위해 비교된다. 모션 벡터들(110(2))의 어레이에서 각각의 모션 벡터(404)는 픽셀들의 블록(예를 들어, 8 x 8 픽셀 블록)에 대응할 수 있기 때문에, 이러한 색상 비교는 프레임들(300) 사이의 8 x 8 픽셀 블록에서 RGB 픽셀 값들의 일부 또는 전부를 비교함으로써 모션 벡터별로 기준으로 수행될 수 있다. 일부 실시형태에서, 로직(예를 들어, HMD(102)의 로직)은, 프레임들(300) 사이의 색상 변화의 양(예를 들어, 색상 차이 값)을 결정하기 위해 단일 모션 벡터(404)에 대응되는 각각의 8 x 8 픽셀 블록에서 픽셀들의 매 두 번째 행 및 픽셀의 매 두 번째 열에 대한 색상 값들을 비교하며, 색상 변화의 양은 특정 모션 벡터(404)와 연관된다. 따라서, 주어진 모션 벡터(110(2))에 대응되는 픽셀 블록과 관련하여, 프레임들 간의 색상 비교는 프로세싱 리소스들을 절약하기 위해 대응되는 픽셀 블록들에서 픽셀들의 일 부분(예를 들어, 25%)을 비교하는 것을 포함할 수 있다.

[0111] 1208에서, 로직(예를 들어, HMD(102)의 로직)은 색상 차이 값들이 임계 색상 차이보다 더 작은 모션 벡터들(110(2))의 세브세트의 크기들을 제로의 길이/크기로 감소시킬 수 있다. 이것은, 이전에 렌더링된 프레임들 사이에 색상 변화가 거의 또는 전혀 없는(때로는 제로 모션 색상 차이로 지칭됨) 장면의 영역에서 어떠한 모션 벡터들도 포함하지 않는 모션 벡터들(110(2))의 수정된 어레이를 생성한다. 이것은, 예를 들어, 텍스트가 단색 배경(solid-colored background)의 전방에 있을 때, 에지들 주변에서 텍스트가 흐릿하거나 구불구불하게 표시되는 것을 방지하는 데 도움을 줄 수 있다. 이것은 또한, 장면 내 조명들이나 기타 사용자 인터페이스 요소들이 점차 더 밝아지거나 더 어두워질 때마다, 시각적으로 산만해지는 아티팩트들이 나타나는 것을 방지하는 데 도움을 줄 수 있다.

[0112] 1210에서, 로직(예를 들어, HMD(102)의 로직)은 모션 벡터들(110(2))의 수정된 어레이에 기초하여 재투영된 프레임에 대한 픽셀 데이터(110(1)(3))를 수정할 수 있으며, 이는, 이전에 렌더링된 프레임들(300) 사이에서 색상이 실질적으로 동일한 영역들에서 더 이상 모션 벡터들(404)을 포함하지 않는다. 픽셀 데이터(110(1)(3))의 수정은 본원에 설명된 기술들(예를 들어, 프로세스(500)의 블록 508 내지 블록 512를 참조하여 설명된 기술들) 중 어느 하나를 포함할 수 있다.

[0113] 1212에서, 로직(예를 들어, HMD(102)의 로직)은, 재투영된 프레임(300(3))에 대한 수정된 픽셀 데이터(110(1)(3)')에 적어도 부분적으로 기초하여 디스플레이(예를 들어, HMD(102)의 디스플레이 패널(들)) 상에, 재투영된 프레임(300(3))을 렌더링할 수 있다. 블록 1212에서 수행된 작동(들)은 프로세스(500)의 블록 514와 관련하여 설명된 것과 유사할 수 있다.

[0114] 도 13은 본원에 개시된 실시형태에 따른, 모션 평활화 기술의 일부로서 재투영된 프레임을 생성하기 위해 모션 벡터들의 결과 세트가 사용되기 전에 하나 또는 둘 이상의 필터들을 사용하여 모션 벡터 필드를 "클린업"하기 위한 예시적인 프로세스(1300)의 흐름도이다. 논의를 위해, 프로세스(1300)는 이전 도면들을 참조하여 설명된다.

[0115] 1302에서, 로직(예를 들어, 호스트 컴퓨터(106)의 로직)은 이전에 렌더링된 프레임들(300)과 연관된 픽셀 데이터(110(1))를 입력으로서 GPU(304)에 제공할 수 있다. 블록 1302에서 수행된 작동(들)은 프로세스(500)의 블록 502와 관련하여 설명된 것과 유사할 수 있다.

[0116] 1304에서, 로직(예를 들어, 호스트 컴퓨터(106)의 로직)은 GPU(304)로부터 모션 벡터들(110(2))의 어레이를 수신할 수 있다. 블록 1304에서 수행된 작동(들)은 프로세스(500)의 블록 504와 관련하여 설명된 것과 유사할 수 있다.

[0117] 1306에서, 로직(예를 들어, 호스트 컴퓨터(106)로부터 모션 벡터들(110(2))을 수신한 후 HMD(102)의 로직)은 모션 벡터들(110(2))의 어레이에 N x N 스칼라 중간값 필터를 적용하여 모션 벡터들(110(2))의 제1 수정된 어레이를 획득할 수 있다. 여기서 N은 5 x 5 스칼라 중간값 필터를 적용할 수 있는 임의의 적절한 수, 예를 들어, N = 5일 수 있다. N = 5인 예에서, 5 x 5 스칼라 중간값 필터는, 선택된 모션 벡터를 둘러싸는 모션 벡터들(110

(2))의 5 x 5 세트를 고려함으로써 모션 벡터들(110(2))의 어레이의 각각의 모션 벡터에 적용된다. 선택된 모션 벡터의 경우, 선택된 벡터를 둘러싸는 5 x 5 영역에는 25개의 모션 벡터들이 있다. 선택된 모션 벡터에 적용된 스칼라 중간값 필터는 (25개의 x 성분 값들 중에서) 중간 x 성분 값 및 (25개의 y 성분 값들 중에서) 중간 y 성분 값을 계산한 후, 중간 x 성분 값과 중간 y 성분 값을 결합하여 스칼라 중간값 벡터를 획득하고, 선택된 모션 벡터는 스칼라 중간값 벡터로 대체될 수 있다. 이러한 프로세스는 모션 벡터들(110(2))의 어레이의 각각의 벡터에 대해 반복된다. 또한, N에 대한 임의의 적절한 값이 사용될 수 있고, N = 5는 단지 일례일뿐이다. 블록 1306에서 스칼라 중간값 필터를 적용하는 것은 모션 벡터 필드(110(2))에서 노이즈를 감소시킬 수 있고 이상치 모션 벡터들을 제거하는 것을 돕는다. 블록 1306에서 스칼라 중간값 필터를 사용하는 것은 또한, 더욱 계산 집약적인 더 복잡한 거리 알고리즘에 의존하는 벡터 중간값 필터와 같은 더 복잡한 필터들을 적용하는 것보다 계산적으로 더 빠르다.

[0118] 1308에서, 로직(예를 들어, HMD(102)의 로직)은 모션 벡터들(110(2))의 제2 수정된 어레이를 획득하기 위해서 모션 벡터들(110(2))의 제1 수정된 어레이에 M x M 평균-최대 블러 필터를 적용할 수 있다. 여기서, M은, 3 x 3 평균-최대 블러 필터를 적용할 수 있는 임의의 적절한 수, 예를 들어, M = 3일 수 있다. 평균-최대 블러 필터는 2개의 개별 필터들(평균 벡터 필터 및 최대 길이 벡터 필터)를 독립적으로 적용한 후, 해당하는 2개의 독립적으로 적용된 필터들로부터 결과적인 벡터들의 벡터 평균을 취한다. M = 3을 사용하는 예에서, 3 x 3 평균-최대 블러 필터는 선택된 모션 벡터를 둘러싸는 모션 벡터들(110(2))의 3 x 3 세트를 고려함으로써 모션 벡터들(110(2))의 어레이의 각각의 모션 벡터에 적용된다. 선택된 모션 벡터의 경우, 선택된 벡터를 둘러싸는 3 x 3 영역에는 9개의 모션 벡터들이 있다. 먼저, 평균 벡터 필터가 선택된 모션 벡터에 적용되어, (9개의 x 성분 값들 중에서) 평균 x 성분 값 및 (9개의 y 성분 값들 중에서) 평균 y 성분 값을 계산한 후에, 평균 x 성분 값과 평균 y 성분 값을 결합하여 평균 벡터를 획득한다. 이어서, 최대 길이 벡터 필터가 선택된 모션 벡터에 독립적으로 적용되어, (9개의 모션 벡터들 중에서) 가장 긴 벡터를 결정한다. 그 후, 평균 벡터와 가장 긴 벡터의 벡터 평균을 취함으로써 결과적인 벡터가 결정되고, 선택된 모션 벡터는 결과적인 벡터로 대체될 수 있다. 이러한 프로세스는 모션 벡터들(110(2))의 어레이의 각각의 벡터에 대해 반복된다. 또한, M에 대한 임의의 적절한 값이 사용될 수 있고, M = 3은 단지 일례일뿐이다. 블록 1308에서 평균-최대 블러 필터를 적용하는 것은 모션 벡터들의 그룹들의 서브영역들에서 불연속성을 제거하는 평활화된 모션 벡터 필드를 제공한다. 일부 실시예에서, 블록 1308에서 평균-최대 블러 필터를 적용하는 단계는 본원에 설명된 바와 같이 모션 벡터 필드(110(2))가 모션 평활화를 위해 사용되기 전에 모션 벡터 필드(110(2))에 대한 최종 "클린업" 단계이다. 예를 들어, 모션 벡터들의 임계치화, 감쇠 텍스처의 사용, 색상 차이가 거의 또는 전혀 없는 영역들에서 벡터들의 제로화 등과 같이 본원에 설명된 다른 기술들과 조합하여 사용되는 경우, 이러한 다양한 작동들은 평균-최대 블러 필터를 적용하기 이전에 수행될 수 있다.

[0119] 1310에서, 로직(예를 들어, HMD(102)의 로직)은 필터링된 모션 벡터들을 갖는 모션 벡터들(110(2))의 수정된 어레이에 기초하여 재투영된 프레임에 대한 픽셀 데이터(110(1)(3))를 수정할 수 있다. 픽셀 데이터(110(1)(3))의 수정은 본원에 설명된 기술들(예를 들어, 프로세스(500)의 블록 508 내지 블록 512를 참조하여 설명된 기술들) 중 어느 하나를 포함할 수 있다.

[0120] 1312에서, 로직(예를 들어, HMD(102)의 로직)은, 재투영된 프레임(300(3))에 대한 수정된 픽셀 데이터(110(1)(3)')에 적어도 부분적으로 기초하여 디스플레이(예를 들어, HMD(102)의 디스플레이 패널(들)) 상에, 재투영된 프레임(300(3))을 렌더링할 수 있다. 블록 1312에서 수행된 작동(들)은 프로세스(500)의 블록 514와 관련하여 설명된 것과 유사할 수 있다.

[0121] 도 9 내지 도 13의 프로세스들에서 설명된 다양한 기술들(GPU(304)에 의해 출력되는 모션 벡터 필드를 수정하기 위한 다양한 기술들)이 임의의 조합 및/또는 임의의 순서로 사용될 수 있다는 것이 이해될 것이다. 예를 들어, 도 9를 참조하여 설명된 기술들은 도 10 및 도 11을 참조하여 설명된 기술들과 결합하여 사용될 수 있고, 이러한 기술들의 일부 또는 전부는 도 12를 참조하여 설명된 기술들과 결합하여 사용될 수 있고, 이러한 기술들의 일부 또는 전부는 도 13을 참조하여 설명된 기술과 결합하여 사용될 수 있다. 즉, 최적화된 모션 벡터 필드를 획득하는 강력한 기술은 모션 벡터 필드의 강력한 수정에서 도 9 내지 도 13을 참조하여 설명된 기술의 전부를 함께 활용할 수 있다. 또한, 호스트 컴퓨터(106) 및 통신 가능하게 결합된 HMD(102)를 포함하는 분산 시스템에서, 도 9 내지 도 13의 프로세스들에서 설명된 다양한 기술들의 일부 또는 전부(GPU(304)에 의해 출력된 모션 벡터 필드를 수정하기 위한 다양한 기술들)는 호스트 컴퓨터(106)에서 수행될 수 있다는 것이 이해될 것이다. 예를 들어, 호스트 컴퓨터(106)는, 모션 벡터를 임계치화하도록(예를 들어, 감쇠 텍스처를 사용하여), 그리고/또는 모션 평활화에 사용되기 위해 모션 벡터들의 수정된 어레이를 HMD(102)에 전송하기 전에 모션 벡터들에 필

터를 적용하도록 구성될 수 있다. 이 시나리오에서, 모션 벡터들(110(2))의 수정된 어레이는 압축된 픽셀 데이터(110(1))로부터 대역외로 전송될 수 있고/있거나, 모션 벡터들(110(2))의 수정된 어레이의 제로 모션 벡터들은 HMD(102)로의 모션 벡터들의 전송 이전에 모션 벡터 데이터를 압축하기 위해 런-렝스(run-length) 인코딩될 수 있다.

[0122] 도 14는 본원에 개시된 실시형태에 따른, 픽셀 데이터가 모션 벡터 추정을 위해 GPU에 입력되기 전에 이전에 렌더링된 프레임들을 회전시키기 위한 예시적인 프로세스(1400)의 흐름도이다. 논의를 위해, 프로세스(1400)는 이전 도면들을 참조하여 설명된다.

[0123] 1402에서, 로직(예를 들어, 호스트 컴퓨터(106)의 로직)은 GPU(304)에 입력으로서 제공될 픽셀 데이터(110(1))을 갖는 다수의 이전에 렌더링된 프레임들(300) 중 제1 프레임(300(1))을 제1 회전된 프레임(1404(1))을 획득하기 위한 회전량만큼 회전시킬 수 있다.

[0124] 1406에서, 로직(예를 들어, 호스트 컴퓨터(106)의 로직)은 다수의 이전에 렌더링된 프레임들(300) 중 제2 프레임(300(2))을 제2 회전된 프레임(1404(2))을 획득하기 위한 회전량만큼 회전시킬 수 있다.

[0125] 1408에서, 로직(예를 들어, 호스트 컴퓨터(106)의 로직)은 회전된 프레임들(1404(1) 및 1404(2))의 픽셀 데이터를 입력으로서 GPU(304)에 제공할 수 있다. 프로세스(1400)는 블록 1408로부터 프로세스(500)의 블록 504로 계속될 수 있다(오프 페이지 참조 "A"에 의해서 도시됨). 따라서, 모션 벡터들(110(2))의 어레이는 블록 1408에서 입력으로서 GPU(304)에 제공되는 픽셀 데이터에 기초하여 블록 504에서 GPU(304)로부터 수신될 수 있고, 도 5의 모션 평활화 알고리즘의 잔여 작동들은 모션 평활화되는 재투영된 프레임(300(3))을 렌더링하기 위해 수행될 수 있다.

[0126] 프로세스(1400)는, 이동하거나 애니메이션되는 객체를 진정으로 나타내지 않는 "이상한" 모션 벡터들을 야기하는 것으로부터 장면에서 수평 및/또는 수직 아키텍처 에지의 효과를 완화시킬 수 있다. 다수의 이전에 렌더링된 프레임들(300(1) 및 300(2))은 블록 1402 및 블록 1406에서 동일한 회전량만큼 회전될 수 있거나, 또는 각각의 프레임(300(1) 및 300(2))은 블록 1402 및 블록 1406에서 각각 상이한 회전량만큼 회전될 수 있다. 일부 실시형태에서, 블록 1402 및 블록 1406에서 각각의 프레임의 회전량은 미리 정의된다(예를 들어, 입력 프레임들(300(1) 및 300(2))을 45 디그리로 회전). 프레임들(300(1) 및 300(2))을 45 디그리 회전하는 것은, 장면에서 수평 및 수직 에지로 하여금 대각적으로 배향되게 할 수 있고, 결과적으로, 이러한 대각선 에지들은 많은 "이상한" 모션 벡터들이 GPU(304)의 비디오 인코더에 의해 생성되게 하지 않을 수 있다. 일부 실시예에서, 블록 1402 및 블록 1406에서 각각의 프레임의 회전량은 각각의 프레임(300)에 대한 임의의 회전량일 수 있다. 이것은 45 디그리로 정렬될 수 있는 임의의 에지들을 일시적으로 은닉할 수 있다.

[0127] 도 15는 본원에 개시된 실시형태에 따른, 이전에 렌더링된 프레임들의 크로마 데이터 및 루마 데이터에 기초하여 생성되는 모션 벡터들의 어레이들 사이에서 선택하기 위한 예시적인 프로세스(1500)의 흐름도이다. 논의를 위해, 프로세스(1500)는 이전 도면들을 참조하여 설명된다. 또한, 도 5 및 도 15에서 오프 페이지 참조 "B"에 의해 도시된 바와 같이, 프로세스(1500)는 도 5의 블록 506에서의 작동들 이전에 수행되는 작동들을 나타낼 수 있고, 프로세스(500)는, 일부 실시형태에서, 블록 506 내지 블록 514의 작동들을 계속할 수 있다.

[0128] 1502에서, 로직(예를 들어, 호스트 컴퓨터(106)의 로직)은 이전에 렌더링된 프레임들(300)과 연관된 루마 데이터를 입력으로서 GPU(304)에 제공할 수 있다. 예를 들어, 제1 프레임(300(1))과 연관된 제1 픽셀 데이터(110(1)(1))는 제1 루마 데이터, 제1 크로마 데이터, 및/또는 추가 유형들의 데이터를 포함할 수 있다. 유사하게, 제2 프레임(300(2))과 연관된 제2 픽셀 데이터(110(1)(2))는 제2 루마 데이터, 제2 크로마 데이터 등과 같은 유사한 유형들의 데이터를 포함할 수 있다. 따라서, 제1 루마 데이터 및 제2 루마 데이터는 블록 1502에서 입력으로서 GPU(304)에 제공될 수 있다.

[0129] 1504에서, 로직(예를 들어, 호스트 컴퓨터(106)의 로직)은, GPU(304)로부터, 제1 루마 데이터 및 제2 루마 데이터에 적어도 부분적으로 기초하여 GPU(304)의 비디오 인코더에 의해 생성된 모션 벡터들(110(2))의 제1 어레이를 수신할 수 있다.

[0130] 1506에서, 로직(예를 들어, 호스트 컴퓨터(106)의 로직)은 제1 프레임(300(1))과 연관된 제1 크로마 데이터 및 제2 프레임(300(2))과 연관된 제2 크로마 데이터를 GPU(304)에 입력으로서 제공할 수 있다.

[0131] 1508에서, 로직(예를 들어, 호스트 컴퓨터(106)의 로직)은, GPU(304)로부터, 제1 크로마 데이터 및 제2 크로마 데이터에 적어도 부분적으로 기초하여 GPU(304)의 비디오 인코더에 의해 생성된 모션 벡터들(110(2))의 제2 어

레이를 수신할 수 있다.

[0132] 1510에서, 로직(예를 들어, 모션 벡터들(110(2))의 다수의 어레이의 수신 후 HMD(102)의 로직, 또는 호스트 컴퓨터(106)의 로직)은 모션 벡터들(110(2))의 제1 어레이 또는 모션 벡터들(110(2))의 제2 어레이의 중 하나를 모션 벡터들(110(2))의 제1 어레이 및 모션 벡터들(110(2))의 제2 어레이의 분석에 기초하여 모션 벡터들의 선택된 어레이로서 선택할 수 있다. 예를 들어, 선택 알고리즘은, 일례로서, 가장 적게 불규칙한 것을 선택하기 위해서 모션 벡터들(110(3))의 각각의 필드에서 모션 벡터들(110(2))의 크기들 및/또는 방향들에 기초할 수 있다. 예를 들어, 밝은 섬광(예를 들어, 가상 폭발)이 제2 프레임(300(2))에서 발생할 때, 루마 데이터는 GPU(304)가 일부 임계 크기를 초과하는 크기들을 갖는 모션 벡터들(110(2))을 출력하게 할 수 있는 반면, 동일한 이전에 렌더링된 프레임들(300)에 대한 크로마 데이터는 GPU(304)가 극단적으로 큰 크기를 갖는 이러한 모션 벡터들을 포함하지 않는 모션 벡터들(110(2))을 출력하게 할 수 있다. 이 경우, 크로마 데이터로부터 생성된 모션 벡터들(110(2))의 어레이는 블록 1510에서 선택될 수 있다. 이것은 선택 알고리즘의 일 예일뿐이고, 다른 것이 채용될 수 있다. 언급된 바와 같이, 프로세스(1500)는 블록 1510으로부터 프로세스(500)의 블록 506으로 계속될 수 있으며, 블록 506에서, 블록 1510으로부터의 모션 벡터들(110(2))의 선택된 어레이에 기초하여 재투영된 프레임(300(3))에 대한 픽셀 데이터(110(1)(3))가 수정된다.

[0133] 도 16은 본원에 개시된 실시형태에 따른, 모션 벡터들의 다수 어레이들을 획득하고, 어레이들 간의 차이들을 결정하고, 모션 평활화를 위해 결정된 차이들에 기초하여 모션 벡터들의 최종 어레이를 생성하기 위한 예시적인 프로세스(1600)의 흐름도이다. 논의를 위해, 프로세스(1600)는 이전 도면들을 참조하여 설명된다. 또한, 도 5 및 도 16에서 오프 페이지 참조 "B"에 의해 도시된 바와 같이, 프로세스(1600)는 도 5의 블록 506에서의 작동들 이전에 수행되는 작동들을 나타낼 수 있고, 프로세스(500)는, 일부 실시형태에서, 블록 506 내지 블록 514의 작동들을 계속할 수 있다.

[0134] 1602에서, 로직(예를 들어, 호스트 컴퓨터(106)의 로직)은 이전에 렌더링된 프레임들(300)과 연관된 제1 픽셀 데이터(110(1)(1))를 입력으로서 GPU(304)에 제공할 수 있다. 블록 1602에서 수행된 작동(들)은 프로세스(500)의 블록 502와 관련하여 설명된 것과 유사할 수 있다.

[0135] 1604에서, 로직(예를 들어, 호스트 컴퓨터(106)의 로직)은 블록 1602에서 입력으로서 제공된 제1 픽셀 데이터(110(1)(1))에 기초하여 GPU(304)로부터 모션 벡터들(110(2))의 제1 어레이를 수신할 수 있다.

[0136] 1606에서, 로직(예를 들어, 호스트 컴퓨터(106)의 로직)은 이전에 렌더링된 프레임들(300)과 연관된 제2 픽셀 데이터(110(1)(2))를 입력으로서 GPU(304)에 제공할 수 있다.

[0137] 1608에서, 로직(예를 들어, 호스트 컴퓨터(106)의 로직)은, GPU(304)로부터, 제2 픽셀 데이터(110(1)(2))에 적어도 부분적으로 기초하여, 모션 벡터들(110(2))의 제2 어레이를 수신할 수 있다.

[0138] 1610에서, 로직(예를 들어, 호스트 컴퓨터(106)로부터 모션 벡터들(110(2))의 어레이들의 수신 후 HMD(102)의 로직, 또는 호스트 컴퓨터(106)의 로직)은 제1 어레이(110(2))와 제2 어레이(110(2)) 사이에서 모션 벡터들(110(2))의 방향들 및/또는 크기들의 차이를 결정할 수 있다. 예를 들어, 제1 어레이(110(2))의 모션 벡터(404)와 제2 어레이(110(2))의 대응되는 모션 벡터(404) 사이를 비교하여, 모션 벡터들이 모션 벡터들의 방향 또는 크기 중 어느 하나 또는 둘 모두의 측면에서 상이한지, 및, 상이한 경우에, 서로 얼만큼 상이한지 그 양을 결정할 수 있다.

[0139] 1612에서, 로직(예를 들어, 호스트 컴퓨터(106)의 로직, 또는 HMD(102)의 로직)은, 블록 1610에서 결정된 바와 같은, 모션 벡터들(110(2))의 제1 어레이 또는 모션 벡터들(110(2))의 제2 어레이 사이의 차이들에 적어도 부분적으로 기초하여, 모션 벡터들(110(2))의 최종 어레이를 생성할 수 있다. 예를 들어, 모션 벡터들(110(2))의 최종 어레이는, 제1 어레이(110(2)) 또는 제2 어레이(110(2))의 모션 벡터들의 개별 모션 벡터들을 감쇠시키는 것으로부터, 또는 각각의 어레이(110(2))에서 대응되는 모션 벡터들의 평균을 나타내는 평균 모션 벡터들을 감쇠시키는 것으로부터 기인되는 모션 벡터들의 세트를 나타낼 수 있다. 언급된 바와 같이, 프로세스(1600)는 블록 1612로부터 프로세스(500)의 블록 506으로 계속될 수 있으며, 블록 506에서, 블록 1612에서 생성된 모션 벡터들(110(2))의 최종 어레이에 기초하여 재투영된 프레임(300(3))에 대한 픽셀 데이터(110(1)(3))가 수정된다.

[0140] 프로세스(1600)가 사용될 수 있는 방법에 대한 예시적인 예로서, 모션 벡터들(110(2))의 제1 어레이가 원래("y-up") 배향으로 입력 프레임들(300(1) 및 300(2))의 픽셀 데이터(110(1))에 기초하여 블록 1604에서 수신되고, 모션 벡터들(110(2))의 제2 어레이가 45 디그리 회전된 입력 프레임들(300(1) 및 300(2))의 픽셀 데이터(110(1))에 기초하여 블록 1608에서 수신되는 일례를 고려한다. 입력 이미지 프레임들(300)을 회전시키는 이러

한 개념은 도 14를 참조하여 설명되었다. 따라서, 이러한 예시적인 예에서, 제1 프레임(300(1)) 및 제2 프레임(300(2))에 대한 픽셀 데이터(110(1))는 모션 벡터들(110(2))의 제1 어레이를 생성하기 위해서 블록 1602에서 입력으로서 제공될 수 있고, 제1 회전된 프레임(1404(1)) 및 제2 회전된 프레임(1404(2))에 대한 픽셀 데이터(110(1))는 모션 벡터들(110(2))의 제2 어레이를 생성하기 위해서 블록 1606에서 입력으로서 제공될 수 있다. 이러한 예시적인 실시예에서, 모션 벡터들(110(2))의 두 개의 어레이들은 (예를 들어, 제2 어레이(110(2))를 역방향으로 45 디그리 회전시킴으로써) 재정렬된 후에 비교되어 양 어레이들(110(2))에서 대응되는 벡터들 간의 각도 차이들(방향) 및/또는 길이 차이들(크기)를 고려함으로써 불일치를 결정할 수 있다. 실제 모션(예를 들어, 장면에서 이동하거나 애니메이션되는 객체들)이 회전된 입력 프레임들(300)에 대한 모션 벡터들의 매우 유사한 어레이들을 생성할 수 있는 한편, 부정확하게 검출된 모션이 모션 벡터들의 매우 상이한 어레이들을 생성할 수 있는 경우일 수 있다. 이러한 예시적인 예에서, 모션 벡터들(110(2))의 최종 어레이는 2개의 정규화된 벡터들의 내적(dot product)에 의해 각각의 어레이(110(2))에서 2개의 대응되는 벡터들의 평균의 길이를 감쇠시킴으로써 블록 1612에서 생성될 수 있다. 일부 실시형태에서, 도 10 및 도 11을 참조하여 설명된 것과 같은 감쇠 텍스처는 최종 어레이를 생성하기 위해 블록 1612에서 활용될 수 있다. 예를 들어, 프로세스(1100)의 블록 1102에서 결정된 델타 벡터들의 세트는 블록 1604에서 수신된 모션 벡터들(110(2))의 제1 어레이와 블록 1608에서 수신된 모션 벡터들(110(2))의 제2 어레이 사이의 델타들에 기초할 수 있고, 결과적인 감쇠 텍스처는 블록 1612에서 사용되어 제1 어레이(110(2)), 제2 어레이(110(2)), 또는 두 개의 어레이들(110(2)) 사이의 벡터들의 조합(예를 들어, 평균)을 감쇠시킬 수 있다.

[0141] 프로세스(1600)가 사용될 수 있는 방법에 대한 또 다른 예시적인 예로서, 다수의 이전에 렌더링된 프레임들(300)의 입력 텍스처들에 대해 제1 밉(mip) 레벨(밉맵(mipmap) 레벨)이 생성되고, 그리고 이러한 제1 밉 레벨에 대한 대응되는 픽셀 데이터(110(1))가 모션 벡터들(110(2))의 제1 어레이를 생성하기 위해 블록 1602에서 GPU(304)에 입력으로서 제공되는 일례를 고려한다. 한편, 다수의 이전에 렌더링된 프레임들(300)의 입력 텍스처에 대해 제2 밉 레벨이 생성될 수 있고, 이러한 제2 밉 레벨에 대한 대응되는 픽셀 데이터(110(1))는 모션 벡터들(110(2))의 제2 어레이를 생성하기 위해 블록 1606에서 GPU(304)에 입력으로서 제공될 수 있다. 이는 모션 벡터들의 임의의 개수의 대응되는 어레이들을 생성하기 위해 임의의 수의 밉 레벨들에 대해 수행될 수도 있다. 일부 실시형태에서, 모션 벡터들(110(2))의 3개 또는 4개의 어레이들이 블록 1610 이전에 수신되도록 입력 텍스처의 3개 또는 4개의 밉 레벨들이 생성될 수 있다. 일부 실시형태에서, 각각의 밉 레벨은 이전 밉 레벨의 폭 및 높이의 절반(예를 들어, 면적의 25%)이다. 블록 1610에서, 이러한 예시적인 예에서, 상이한 밉 레벨들로부터 생성된 어레이들(110(2))의 쌍들 간의 차이가 결정될 수 있고, 블록 1612에서, 모션 벡터들(110(2))의 최종 어레이가, 결정된 차이들에 기초하여 생성될 수 있다. 예를 들어, 로직은 상이한 밉 레벨들에 대해 모션 벡터들(110(2))의 다수의 어레이들에 걸쳐 변칙들을 검출할 수 있고, 변칙적인 모션 벡터들은 감쇠될 수 있다(예를 들어, 제로로 감소됨). 모션 벡터들(110(2))의 상이한 어레이들을 생성하기 위해 입력 프레임들(300)의 상이한 밉 레벨을 사용하는 이러한 예는, 장면에 반복 패턴(예를 들어, 벽지)의 넓은 영역이 있고, GPU(304)의 비디오 인코더가 이와 달리, 이동하거나 애니메이션되는 객체들이 없더라도, 모션 벡터 해상도와 관련된 반복 패턴의 빈도로 인해 다수의 모션이 있다고 생각하여 다수의 큰 크기의 모션 벡터들을 생성할 수 있는 상황에 도움이 될 수 있다. 따라서, 프로세스(1600)는, 상이한 밉 레벨들에 대한 모션 벡터들(110(2))의 어레이들을 생성하기 위해서 사용될 때, 반복 패턴의 상이한 빈도들에 걸쳐 변칙들의 검출을 도울 수 있고, 검출될 때, 감쇠 텍스처는 해당 모션 벡터들(110(2))을 감쇠(예를 들어, 제로화)하기 위해서 사용될 수 있다.

[0142] 프로세스(1600)가 사용될 수 있는 방법에 대한 또 다른 예로서, 다수의 어레이들(110(2))이 서로 다른 해상도들에서 획득되고, 블록 1610에서, 제1 어레이의 단일 모션 벡터(110(2))를 제2 어레이의 다수의 대응되는 모션 벡터들(110(2))과(예를 들어, 제2 어레이의 다수의 대응되는 모션 벡터들(110(2))의 방향 및/또는 크기 측면에서 평균과) 비교하여 차이들이 결정되는 일례를 고려한다. 상이한 해상도들에서 모션 벡터들(110(2))의 어레이들을 사용하는 것은 블록 1612에서 모션 벡터들의 최종 어레이를 생성하는 데 유용한 차이들을 산출할 수 있다.

[0143] 도 17은 본원에 개시된 실시형태에 따른, 이미지 영역의 상이한 부분들에 대해 상이한 해상도들에서 모션 벡터들의 다수의 어레이들을 획득하기 위한 예시적인 프로세스(1700)의 흐름도이다. 논의를 위해, 프로세스(1700)는 이전 도면들을 참조하여 설명된다. 또한, 도 5 및 도 17에서 오프 페이지 참조 "B"에 의해 도시된 바와 같이, 프로세스(1700)는 도 5의 블록 506에서의 작동들 이전에 수행되는 작동들을 나타낼 수 있고, 프로세스(500)는, 일부 실시형태에서, 블록 506 내지 블록 514의 작동들을 계속할 수 있다.

[0144] 1702에서, 로직(예를 들어, 호스트 컴퓨터(106)의 로직)은 이전에 렌더링된 프레임들(300)과 연관된 픽셀 데이터(110(1))를 입력으로서 GPU(304)에 제공할 수 있다. 블록 1702에서 수행된 작동(들)은 프로세스(500)의 블록

502와 관련하여 설명된 것과 유사할 수 있다.

[0145] 1704에서, 로직(예를 들어, 호스트 컴퓨터(106)의 로직)은 GPU(304)로부터 모션 벡터들의 제1 어레이(110(2)(1))를 수신할 수 있으며, 모션 벡터들의 제1 어레이(110(2)(1))는 제1 해상도에서 수신되거나, 또는 이와 달리 제1 해상도로 업 샘플링 또는 다운 샘플링된다.

[0146] 1706에서, 로직(예를 들어, 호스트 컴퓨터(106)의 로직)은 모션 벡터들의 제1 어레이(110(2)(1))에 적어도 부분적으로 기초하여 디스플레이에 걸쳐 있는 이미지 영역(1705)의 제1 부분(1705(A))에서 모션의 부재를 검출할 수 있고, 모션 벡터들의 제1 어레이(110(2)(1))에 적어도 부분적으로 기초하여 이미지 영역(1705)의 제2 부분(1705(B))에서 객체(302)의 모션을 검출할 수 있다. 예를 들어, 넌-제로 모션 벡터들은 이미지 영역(1705)의 우측 절반부에서 검출될 수 있는 한편, 이미지 영역(1705)의 좌측 절반부는 임의의 넌-제로 모션 벡터들 없이 제로 값의 모션 벡터들을 포함할 수 있다.

[0147] 1708에서, 로직(예를 들어, 호스트 컴퓨터(106)의 로직)은 (제1 프레임(300(1))과 연관된) 제1 픽셀 데이터(110(1)(1))의 제1 부분 및 (제2 프레임(300(2))과 연관된) 제2 픽셀 데이터(110(1)(2))의 제1 부분을 GPU(304)에 입력으로서 제공할 수 있다. 제1 픽셀 데이터(110(1)(1))의 제1 부분 및 제2 픽셀 데이터(110(1)(2))의 제1 부분 각각은 블록 1706에서 모션의 부재가 검출된 이미지 영역(1705)의 제1 부분(1705(A))에 대응할 수 있다.

[0148] 1710에서, 로직(예를 들어, 호스트 컴퓨터(106)의 로직)은, GPU(304)로부터, 제1 픽셀 데이터(110(1)(1))의 제1 부분 및 제2 픽셀 데이터(110(1)(2))의 제1 부분에 적어도 부분적으로 기초하여 GPU(304)의 비디오 인코더에 의해 생성된 모션 벡터들의 제2 어레이(110(2)(2))를 수신할 수 있다. 이러한 모션 벡터들의 제2 어레이(110(2)(2))는 모션 벡터들의 제1 어레이(110(2)(1))의 제1 해상도보다 더 높은 해상도인 제2 해상도에서 생성될 수 있다. 모션의 부재가 검출된 이미지 영역(1705)의 제1 부분(1705(A))에 대해 더 높은 해상도의 모션 벡터 필드(110(2)(2))를 획득하는 것은, 더 높은 해상도에서의 모션 벡터 필드(110(2)(2))가 대규모 이동이 검출되지 않았던 이미지 영역(1705)의 제1 부분(1705(A))에서 소규모 이동을 검출하는 데 도움이 될 수 있다는 개념에 기초할 수 있다.

[0149] 1712에서, 로직(예를 들어, 호스트 컴퓨터(106)의 로직)은 (제1 프레임(300(1))과 연관된) 제1 픽셀 데이터(110(1)(1))의 제2 부분 및 (제2 프레임(300(2))과 연관된) 제2 픽셀 데이터(110(1)(2))의 제2 부분을 GPU(304)에 입력으로서 제공할 수 있다. 제1 픽셀 데이터(110(1)(1))의 제2 부분 및 제2 픽셀 데이터(110(1)(2))의 제2 부분 각각은 블록 1706에서 객체(302)의 모션이 검출된 이미지 영역(1705)의 제2 부분(1705(B))에 대응할 수 있다.

[0150] 1714에서, 로직(예를 들어, 호스트 컴퓨터(106)의 로직)은, GPU(304)로부터, 제1 픽셀 데이터(110(1)(1))의 제2 부분 및 제2 픽셀 데이터(110(1)(2))의 제2 부분에 적어도 부분적으로 기초하여 GPU(304)의 비디오 인코더에 의해 생성된 모션 벡터들의 제3 어레이(110(2)(3))를 수신할 수 있다. 이러한 모션 벡터들의 제3 어레이(110(2)(3))는 모션 벡터들의 제2 어레이(110(2)(2))의 제2 해상도보다 더 낮은 해상도인 제3 해상도에서 생성될 수 있다. 모션이 검출된 이미지 영역(1705)의 제2 부분(1705(B))에 대해 상대적으로 더 낮은 해상도의 모션 벡터 필드(110(2)(3))는, 더 높은 해상도 모션 벡터 필드가, 대규모 이동이 제1 모션 벡터 필드(110(2)(1))의 제1 해상도에서 이미 검출된 이미지 영역(1705)의 제2 부분(1705(B))에서 이동을 검출할 필요가 없을 수 있다는 개념에 기초할 수 있다.

[0151] 1716에서, 로직(예를 들어, 호스트 컴퓨터(106)로부터 모션 벡터들(110(2))의 어레이들의 수신 후 HMD(102)의 로직, 또는 호스트 컴퓨터(106)의 로직)은 모션 평활화를 위해 모션 벡터들의 제2 어레이(110(2)(2)) 및 모션 벡터들의 제3 어레이(110(2)(3))를 사용할 수 있다. 프로세스(1700)는 블록 1716으로부터 프로세스(500)의 블록 506으로 계속될 수 있으며, 블록 506에서, 블록 1716으로부터의 모션 벡터들(110(2))에 기초하여 재투영된 프레임(300(3))에 대한 픽셀 데이터(110(1)(3))가 수정된다. 일 예에서, 모션 평활화를 위해 블록 1716에서 모션 벡터들의 다수의 어레이들을 사용하는 단계는 프로세스(1600)의 블록 1612를 참조하여 설명된 작동들을 포함할 수 있다.

[0152] 도 18a 및 도 18b는 본 명세서에 개시된 실시형태에 따른, HMD(102) 및 호스트 컴퓨터(106)를 활용하는 시스템의 2개의 대안적인 설정을 예시한다. 도 1에 도시된 바와 같이, 예시적인 구현예는 사용자(104)가 착용한 HMD(102)와 함께 환경에 호스트 컴퓨터(106)가 콜로케이팅되는 경우이다. 예를 들어, 호스트 컴퓨터(106)가 HMD(102)와 동일한 방에 위치되는지 또는 다른 방에 위치되는지에 관계없이, 사용자(104)가 집에서 HMD(102)를

사용하는 동안 호스트 컴퓨터(106)는 사용자(104)의 집에 위치될 수 있다. 대안적으로, 모바일 컴퓨팅 디바이스(예를 들어, 태블릿 또는 랩톱) 형태의 호스트 컴퓨터(106)는 사용자(104)의 등에 있는 배낭에 넣어 운반될 수 있고, 이에 의해 더 큰 이동성을 허용한다. 예를 들어, 사용자(104)는, 이러한 시스템을 사용하는 동안 공공 공원에 위치될 수 있다.

[0153] 도 18a는, 호스트 컴퓨터(106)가 HMD(102)에 대해 지리적으로 원격 위치에 위치된 하나 또는 둘 이상의 서버 컴퓨터들을 나타내는 대안적인 구현예를 도시한다. 이러한 경우에, HMD(102)는, 무선 AP(WAP: wireless AP), 기지국 등과 같은 액세스 포인트(AP: access point)(1800)를 통해 호스트 컴퓨터(들)(106)에 통신 가능하게 결합될 수 있다. 예시적인 실시예에서, 데이터는, 예를 들어, 인터넷을 통해 데이터를 스트리밍함으로써, AP(1800)를 통해 호스트 컴퓨터(106)와 HMD(102) 사이에서 교환(예를 들어, 스트리밍)된다.

[0154] 도 18b는, 호스트 컴퓨터(106)가 랩톱 또는 태블릿 컴퓨터와 같은 중간 컴퓨팅 디바이스(1802)를 통해 HMD(102)에 통신 가능하게 결합된 또 다른 대안적인 구현예를 도시한다. 도 18a와 도 18b 사이의 차이점은, 도 18a의 AP(1800)가 렌더링을 수행하지 않는 데이터 라우팅 디바이스로서 단순히 작동될 수 있는 반면, 도 18b의 중간 컴퓨팅 디바이스(1802)는 렌더링 워크로드의 일 부분을 수행할 수 있다는 것이다. 즉, 호스트 컴퓨터(106)와 HMD(102) 사이에서 렌더링 워크로드를 분기하는 대신, 렌더링 워크로드는 두 개보다 더 많은 디바이스들, 예를 들어, 세 개의 디바이스들(호스트 컴퓨터(106), 중간 컴퓨팅 디바이스(1802), 및 HMD(102)) 사이에서 분할될 수 있다. 도 18b의 시나리오에서, 호스트 컴퓨터(106)는, 본원에 설명된 바와 같이, 픽셀 데이터(110(1))를 생성할 수 있고, 중간 컴퓨팅 디바이스(1802)는 픽셀 데이터(110(1))를 수정하기 위해 제1 세트의 렌더링 작동들을 수행할 수 있고, HMD(102)는 수정된 픽셀 데이터를 수정하기 위해 최종 세트의 렌더링 작동들을 수행할 수 있다.

[0155] 도 19는 본원에 개시된 실시형태에 따른 VR 헤드셋과 같은 HMD(1900)의 예시적인 컴포넌트들이 내장될 수 있다는 것을 예시한다. HMD(102)는 사용자(104)에 의해서 (예를 들어, 사용자(104)의 머리에) 착용될 디바이스로서 구현될 수 있다. 일부 실시형태에서, HMD(102)는, 예를 들어, 사용자(104)의 머리 둘레에 맞도록 크기가 설정된 고정 메커니즘(예를 들어, 조정 가능한 밴드)을 사용하여 사용자(104)가 자신의 머리 상에 HMD(102)를 고정시키는 것을 허용함으로써 머리 장착 가능할 수 있다. 일부 실시형태에서, HMD(102)는 근안(near-eye 또는 near-to-eye) 디스플레이(들)를 포함하는 가상 현실(VR) 또는 증강 현실(AR) 헤드셋을 포함한다. 그러므로, 용어 "웨어러블 디바이스", "웨어러블 전자 디바이스", "VR 헤드셋", "AR 헤드셋", 및 "헤드 마운트 디스플레이(HMD)"는 디바이스(102)를 지칭하기 위해 본원에서 상호 교환 가능하게 사용될 수 있다. 그러나, 이러한 유형의 디바이스들은 단지 HMD(102)의 예라는 점이 이해될 것이며, HMD(102)가 다양한 다른 폼팩터들로 구현될 수 있다는 점이 이해될 것이다.

[0156] 예시된 구현예에서, HMD(102)는 하나 또는 둘 이상의 프로세서들(1900) 및 메모리(1902)(예를 들어, 컴퓨터 판독 가능 매체들(1902))를 포함한다. 일부 구현예에서, 프로세서(들)(1900)는 중앙 처리 유닛(CPU: central processing unit), 그래픽 처리 유닛(GPU)(들)(1900(1)), CPU와 GPU(1900(1)) 모두, 마이크로프로세서, 디지털 신호 프로세서, 또는 당업계에 알려진 다른 처리 유닛들 또는 컴포넌트들을 포함할 수 있다. 대안적으로 또는 추가로, 본원에서 기능적으로 설명된 것은 적어도 부분적으로 하나 또는 둘 이상의 하드웨어 로직 컴포넌트들에 의해 수행될 수 있다. 예를 들어, 그리고 한정됨 없이, 사용될 수 있는 예시적인 유형들의 하드웨어 로직 컴포넌트들은 필드 프로그래머블 게이트 어레이(FPGA: feld-programmable gate array)들, 애플리케이션 특정 집적 회로(ASIC: application-specific integrated circuit)들, 애플리케이션 특정 표준 제품(ASSP: application-specific standard product)들, 시스템 온 칩 시스템(SOC: system-on-a-chip system)들, 복합 프로그래밍가능 로직 디바이스(CPLD: complex programmable logic device)들 등을 포함한다. 추가적으로, 프로세서(들)(1900) 각각은, 프로그램 모듈, 프로그램 데이터, 및/또는 하나 또는 둘 이상의 운영 체제들을 또한 저장할 수 있는 자체 로컬 메모리를 구비할 수 있다.

[0157] 메모리(1902)는, 컴퓨터 판독 가능 명령어들, 데이터 구조들, 프로그램 모듈들 또는 다른 데이터와 같은 정보의 저장을 위한 임의의 방법 또는 기술로 구현된 휘발성 및 비휘발성 메모리, 착탈식 및 비착탈식 매체를 포함할 수 있다. 이러한 메모리는, 원하는 정보를 저장하는 데 사용될 수 있으며 컴퓨팅 디바이스에 의해서 액세스될 수 있는 RAM, ROM, EEPROM, 플래시 메모리 또는 다른 메모리 기술, CD-ROM, 디지털 다목적 디스크(DVD: digital versatile disk)들 또는 다른 광학 기억 장치, 자기 카세트들, 자기 테이프, 자기 디스크 저장 또는 다른 자기 저장 디바이스, RAID 저장 시스템, 또는 임의의 다른 매체를 포함하나, 이에 한정되지 않는다. 메모리(1902)는 컴퓨터 판독 가능 저장 매체(CRSM: computer-readable storage media)로서 구현될 수 있으며, 이는 메모리(1902)에 저장된 명령어들을 실행하기 위해 프로세서(들)(1900)에 의해 액세스 가능한 임의의 이용 가능한 물리적 매체일 수 있다. 하나의 기본 구현예에서, CRSM은 랜덤 액세스 메모리(RAM: random access memory)

및 플래시(Flash) 메모리를 포함할 수 있다. 다른 구현예에서, CRSM은, 읽기 전용 메모리(ROM: read-only memory), 전기적으로 지울 수 있는 프로그래밍 가능한 읽기 전용 메모리(EEPROM: electrically erasable programmable read-only memory), 또는 원하는 정보를 저장하기 위해서 사용될 수 있고, 프로세서(들)(1900)에 의해서 액세스될 수 있는 임의의 다른 유형(tangible)의 매체를 포함할 수 있으나, 이에 한정되지 않는다.

[0158] 일반적으로, HMD(102)는 본원에 설명된 기술들, 기능, 및/또는 작동들을 구현하도록 구성된 로직(예를 들어, 소프트웨어, 하드웨어, 및/또는 펌웨어 등)을 포함할 수 있다. 컴퓨터 판독 가능 매체(1902)는, 본원에 설명된 기술들, 기능, 및/또는 작동들을 수행하기 위해 프로세서(들)(1900) 상에서 실행되도록 구성될 수 있는 다양한 모듈들, 예컨대 명령어, 데이터스토어들 등을 포함하는 것으로 도시된다. 몇몇 예시적인 기능 모듈들은 컴퓨터 판독가능 매체들(1902)에 저장되고 프로세서(들)(1900) 상에서 실행 가능한 것으로서 도시되지만, 동일한 기능은 대안적으로 하드웨어, 펌웨어에서, 또는 시스템 온 칩(SOC), 및/또는 다른 로직으로서 구현될 수 있다.

[0159] 운영 체제 모듈(1904)은 다른 모듈들을 위하여 HMD(102) 내의 그리고 이에 결합된 하드웨어를 관리하도록 구성될 수 있다. 부가적으로, 일부 예시들에서, HMD(102)는 메모리(1902)에 저장되거나 또는 그렇지 않으면 HMD(102)에 액세스 가능한 하나 또는 둘 이상의 애플리케이션들(1906)을 포함할 수 있다. 이러한 구현예에서, 애플리케이션(들)(1906)은 게이밍 애플리케이션(1908)을 포함한다. 그러나, HMD(102)는 임의의 수 또는 유형의 애플리케이션들을 포함할 수 있고, 본원에 도시된 특정 실시예에 한정되지 않는다. HMD(102)의 다른 로직과 조합된 합성기(1910)는 모션 평활화된 재투영된 프레임들(300)을 렌더링하기 위해 본원에 설명된 모션 평활화 기술들을 수행하도록 구성될 수 있다.

[0160] 일반적으로, HMD(102)는 입력 디바이스들(1912) 및 출력 디바이스들(1914)을 갖는다. 입력 디바이스들(1912)은 컨트롤 버튼들을 포함할 수 있다. 일부 구현예에서, 하나 또는 둘 이상의 마이크로폰들은 사용자 음성 입력과 같은 오디오 입력을 수신하기 위한 입력 디바이스들(1912)로서 기능할 수 있다. 일부 구현예에서, 하나 또는 둘 이상의 카메라들 또는 다른 유형들의 센서들(예를 들어, 관성 측정 유닛(IMU: inertial measurement unit)은 사용자(104)의 손 및/또는 머리 모션과 같은 제스처 입력을 수신하기 위한 입력 디바이스들(1912)로서 기능할 수 있다. 일부 실시형태들에서, 추가적인 입력 디바이스들(1912)은 키보드, 키패드, 마우스, 터치 스크린, 조이스틱 등의 형태로 제공될 수 있다. 다른 실시형태에서, HMD(102)는 키보드, 키패드, 또는 다른 유사한 형태들의 기계적 입력부를 생략할 수 있다. 대신에, HMD(102)는 입력 디바이스(1912), 네트워크 인터페이스(무선 또는 유선 기반), 전력, 및 프로세싱/메모리 기능들의 비교적 단순한 형태들로 구현될 수 있다. 예를 들어, 하나 또는 둘 이상의 입력 컴포넌트들의 제한된 세트(예를 들어, 구성을 시작하기 위한 전용 버튼, 전원 온/오프 등)는, 이후에 HMD(102)가 사용될 수 있도록 채용될 수 있다. 일 구현예에서, 입력 디바이스(들)(1912)는 볼륨 증가/감소를 위한 기본 볼륨 제어 버튼(들)뿐만 아니라 전원 및 리셋 버튼과 같은 제어 메커니즘들을 포함할 수 있다.

[0161] 출력 디바이스들(1914)은 디스플레이(들)(1916)를 포함할 수 있으며, 이는 하나 또는 다수의 디스플레이 패널들(예를 들어, 스테레오 쌍의 디스플레이 패널들)을 포함할 수 있다. 출력 디바이스들(1914)은, 광 요소(예를 들어, LED), 햅틱 감각을 생성하기 위한 진동기, 및/또는 스피커(들)(예를 들어, 헤드폰들) 등을 이에 한정됨 없이 더 포함할 수 있다. 예를 들어, 또한, 전원이 켜져 있을 때와 같은 상태를 나타내는 간단한 라이트 요소(예: LED)가 있을 수 있다.

[0162] HMD(102)는 네트워크에 대한 무선 연결을 용이하게 하기 위해 안테나(1920)에 결합된 무선 유닛(1918)을 더 포함할 수 있다. 무선 유닛(1918)은 Wi-Fi, 블루투스, 무선 주파수(RF) 등과 같은 다양한 무선 테크놀로지들 중 하나 이상을 구현할 수 있다. HMD(102)가 네트워크, 연결된 주변 디바이스(호스트 컴퓨터(106), 예를 들어, PC, 게임 콘솔 등을 포함함) 또는 다른 무선 네트워크들과 통신하는 플러그-인 네트워크 디바이스에 대한 유선 연결을 용이하게 하기 위한 물리적 포트들을 더 포함할 수 있다는 것이 이해될 것이다.

[0163] HMD(102)는, 하나 또는 둘 이상의 광학 요소들을 사용하여 전자 디스플레이(들)(1916)로부터 사용자의 눈(들)으로 광을 지향시키는 광학 서브시스템(1922)을 더 포함할 수 있다. 광학 서브시스템(1922)은, 예를 들어 애퍼처들, 렌즈들(예를 들어, 프레넬(Fresnel) 렌즈들, 볼록 렌즈들, 오목 렌즈들 등), 필터들 등을 이에 한정됨 없이 포함하여, 다양한 유형들 및 조합들의 상이한 광학 요소들을 포함할 수 있다. 일부 실시형태에서, 광학 서브시스템(1922) 내의 하나 또는 둘 이상의 광학 요소들은 반사 방지 코팅물과 같은 하나 또는 둘 이상의 코팅물들을 가질 수 있다. 광학 서브시스템(1922)에 의한 이미지 광의 확대는 전자 디스플레이(들)(1916)가 물리적으로 더 작고, 더 가벼운 것을, 그리고 더 큰 디스플레이들보다 더 적은 전력을 소모하는 것을 허용한다. 또한, 영상 광의 확대는 디스플레이된 콘텐츠(예를 들어, 영상들)의 시야(FOV: field of view)를 증대시킬 수 있다. 예를 들어, 디스플레이된 콘텐츠의 FOV는, 디스플레이된 콘텐츠가 사용자의 FOV의 거의 전부(예를 들어, 120 내지 150

디그리 대각선) 및, 일부 경우에서는, 전부를 사용하여 제시되도록 된다. AR 애플리케이션들은 더 좁은 FOV(예를 들어, 약 40 디그리의 FOV)를 가질 수 있다. 광학 서브시스템(1922)은 하나 또는 둘 이상의 광학 에러들, 예를 들어, 한정됨 없이, 배럴 왜곡, 핀쿠션 왜곡, 종방향 색수차, 횡방향 색수차, 구면 수차, 코마 수차(comatic aberration), 상면 만곡, 및 비점 수차 등을 보정하도록 설계될 수 있다. 일부 실시형태에서, 디스플레이를 위해 전자 디스플레이(들)(1916)에 제공된 콘텐츠는 사전 왜곡되고, 광학 서브시스템(1922)은 콘텐츠에 기초하여 생성된 전자 디스플레이(들)(1916)로부터 이미지 광을 수신할 때 왜곡을 보정한다.

[0164] HMD(102)는 모션, 포지션, 및 배향 데이터를 생성하기 위해 사용되는 센서들과 같은 하나 또는 둘 이상의 센서들(1924)를 더 포함할 수 있다. 이러한 센서들(1924)은 자이로스코프, 가속도계, 자력계, 비디오 카메라, 컬러 센서, 또는 다른 모션, 포지션, 및 배향 센서일 수 있거나, 이를 포함할 수 있다. 센서들(1924)은 또한, 모션, 포지션, 및 배향 데이터를 생성하기 위해 카메라 또는 컬러 센서에 의해 외부에 보여질 수 있는 일련의 능동 또는 수동 마커들과 같은 센서들의 서브부분들을 포함할 수 있다. 예를 들어, VR 헤드셋은, 외부 카메라로 보거나 광(예를 들어, 적외선 또는 가시광선)에 의해 조명될 때, 모션, 위치 및 방향 데이터를 생성하기 위해 소프트웨어에 의한 해석을 위한 하나 또는 둘 이상의 참조 지점들을 제공할 수 있는 반사기 또는 광(예를 들어, 적외선 또는 가시광선)과 같은 여러 마커를 외부에 포함할 수 있다. HMD(102)는 HMD(102)의 환경에서 기지국에 의해 투영되거나 브로드캐스트되는 광(예를 들어, 적외선 또는 가시광선)에 민감한 광 센서를 포함할 수 있다.

[0165] 일 실시예에서, 센서(들)(1924)는 관성 측정 유닛(IMU)(1926)을 포함할 수 있다. IMU(1926)는, 가속도계, 자이로스코프, 자력계, 및/또는 모션 검출, IMU(1926)와 관련된 에러 수정에 적합한 다른 센서, 또는 이들의 일부 조합으로부터 수신된 측정 신호를 기반으로 캘리브레이션 데이터를 생성하는 전자 디바이스일 수 있다. 측정 신호들에 기반하여, IMU(1926)와 같은 모션 기반 센서는 HMD(102)의 초기 위치에 대한 HMD(102)의 추정 위치를 나타내는 캘리브레이션 데이터를 생성할 수 있다. 예를 들어, 다수의 가속도계는 병진이동 모션(전방/후방, 위/아래, 좌측/우측)을 측정할 수 있고, 다수의 자이로스코프는 회전 모션(예를 들어, 피칭, 요잉, 및 롤링)을 측정할 수 있다. 예를 들어, IMU(1926)는 측정 신호들을 빠르게 샘플링하고, 샘플링된 데이터로부터 HMD(102)의 추정 위치를 계산할 수 있다. 예를 들어, IMU(1926)는 속도 벡터를 추정하기 위해 시간에 따라 가속도계들로부터 수신된 측정 신호들을 적분할 수 있고, HMD(102) 상의 기준 지점의 추정 위치를 결정하기 위해 시간에 따라 속도 벡터를 적분할 수 있다. 기준 지점은 HMD(1900)의 위치를 설명하기 위해서 사용될 수 있는 지점이다. 기준 지점은 일반적으로 공간 내의 지점으로서 정의될 수 있지만, 다양한 실시형태에서, 기준 지점은 HMD(102) 내의 지점(예를 들어, IMU(1926)의 중심)으로서 정의된다. 대안적으로, IMU(1926)는 샘플링된 측정 신호들을, 캘리브레이션 데이터를 결정하는 외부 콘솔(또는 다른 컴퓨팅 장치)에 제공한다.

[0166] 센서들(1924)는 센서 데이터를 고속으로 제공하기 위해 비교적 고주파수에서 작동할 수 있다. 예를 들어, 센서 데이터는 1000 Hz(또는 1 밀리초마다 1개의 센서 판독 값)의 속도로 생성될 수 있다. 이러한 방식으로, 초당 1000개의 판독 값들이 취해진다. 센서가 이러한 속도(또는 더 빠른 속도)로 이렇게 많은 데이터를 생성할 때, 모션을 예측하기 위해서 사용되는 데이터 세트는 수십 밀리초 정도의 비교적 짧은 시간 기간 동안에도 상당히 크다.

[0167] 언급된 바와 같이, 일부 실시형태에서, 센서들(1924)은 3D 공간에서 HMD(102)의 위치 및/또는 배향, 포즈 등을 추적하기 위해 HMD(102)의 환경에서 기지국에 의해 방출되는 광에 민감한 광 센서들을 포함할 수 있다. 위치 및/또는 배향의 계산은 광 펄스들의 타이밍 특성 및 센서들(1924)에 의해 검출된 광의 존재 또는 부재에 기초할 수 있다.

[0168] HMD(102)는 시선 추적(eye tracking) 모듈(1928)을 더 포함할 수 있다. HMD(102) 내부의 카메라 또는 다른 광학 센서는 사용자의 눈들의 이미지 정보를 캡처할 수 있고, 눈 추적 모듈(1928)은 (예를 들어, 왜곡 조정 목적들을 위해) HMD(102)에 대한 각각의 눈의 동공간 거리, 안구간 거리, 3차원(3D) 포지션(비틀림 및 회전(즉, 롤, 피치, 및 요)의 크기 및 각각의 눈에 대한 시선 방향들을 포함함)을 결정하기 위해 캡처된 정보를 사용할 수 있다. 일 예에서, 적외선 광은 HMD(102) 내에서 방출되고 각각의 눈으로부터 반사된다. 반사된 광은 HMD(102)의 카메라에 의해 수신 또는 검출되고, 각각의 눈에 의해 반사된 적외선 광의 변화들로부터 눈 회전을 추출하기 위해 분석된다. 사용자(104)의 눈들을 추적하기 위한 많은 방법들이 시선 추적 모듈(1928)에 의해 사용될 수 있다. 따라서, 눈 추적 모듈(1928)은 각각의 눈의 최대 6개의 자유도들(즉, 3D 포지션, 롤, 피치, 및 요)을 추적할 수 있고, 추적된 양들의 적어도 서브세트는 시선 지점(즉, 사용자가 보고 있는 가상 장면 내의 3D 위치 또는 포지션)을 추정하기 위해 사용자(104)의 2개의 눈들로부터 결합될 수 있다. 예를 들어, 아이 트래킹 모듈(1928)은 이전 측정으로부터의 정보, 사용자(104) 머리의 위치를 식별하는 측정, 및 전자 디스플레이(들)(1916)에 의해 제시된 장면을 설명하는 3D 정보를 통합할 수 있다. 따라서, 사용자(104)가 보고 있는 HMD(102)에 의

해 제시된 가상 장면에서 시선 지점을 결정하기 위해, 사용자(104)의 눈들의 포지션 및 배향에 대한 정보가 사용된다.

[0169] HMD(102)는 머리 추적 모듈(1930)을 더 포함할 수 있다. 머리 추적 모듈(1930)은 위에서 설명된 바와 같이, 사용자(104)의 머리 모션(머리 회전을 포함함)을 추적하기 위해 센서들(1924) 중 하나 이상을 레버리지(leverage)할 수 있다. 예를 들어, 머리 추적 모듈(1930)은 HMD(102)의 최대 6개의 자유도들(즉, 3D 포지션, 롤, 피치, 및 요)을 추적할 수 있다. 이러한 계산들은 일련의 프레임들(300)의 모든 프레임(300)에서 행해져, 애플리케이션(예: 비디오 게임)이 다음 프레임(300)에서 장면을 렌더링하는 방법을 결정할 수 있고/있거나, 합성기(1910)가 머리 위치 및 배향에 따라 재투영된 프레임(300)을 렌더링하는 방법을 결정할 수 있다. 일부 실시형태에서, 머리 추적 모듈(1930)은 현재 및/또는 과거 데이터에 기초하여 HMD(102)의 미래 위치 및/또는 방향을 예측하도록 구성된다. 이것은, 사용자(104)가 디스플레이(들)(1916) 상에서 광(및 그에 따라, 이미지)을 실제로 보기 전에 애플리케이션이 프레임(300)을 렌더링하도록 요청받기 때문이다. 따라서, 다음 프레임(300)은 프레임(300)을 렌더링하기 전 대략 25 내지 30 밀리세컨드(ms)와 같은 초기 시점에서 만들어진 머리 위치 및/또는 배향의 이러한 미래 예측에 기초하여 렌더링될 수 있다. 호스트 컴퓨터(106)가 HMD(102)에 통신 가능하게(예를 들어, 무선으로) 연결된 분산 시스템에서, 머리 포즈의 미래 예측은 네트워크 레이턴시, 압축 작동 등을 처리하기 위해 프레임(300)에 대한 조명 시간보다 30 ms 이상 앞서 만들어질 수 있다. 머리 추적 시스템(1930)에 의해 제공되는 회전 데이터는 HMD(102) 회전의 방향 및 HMD(102) 회전의 양 두 가지 모두를 측정의 임의의 적합한 단위로 결정하기 위해서 사용될 수 있다. 예를 들어, 회전 방향은, 좌측, 우측, 위, 및 아래에 대응되는 양의 또는 음의 수평 방향 및 양의 또는 음의 수직 방향의 관점에서 단순화되고 출력될 수 있다. 회전의 양은 디그리, 라디안 등으로 될 수 있다. 각속도는 HMD(102)의 회전의 속도를 결정하기 위해 계산될 수 있다.

[0170] 도 20은 본원에 개시된 기술들이 구현될 수 있는 호스트 컴퓨터(106)의 예시적인 컴포넌트들을 예시한다. 예시된 구현예에서, 호스트 컴퓨터(106)는 하나 또는 둘 이상의 프로세서들(2000) 및 메모리(2002)(예를 들어, 컴퓨터 판독 가능 매체(2002))를 포함한다. 일부 구현예에서, 프로세서(들)(2000)는 CPU(들), GPU(들)(304), CPU(들) 및 GPU(들)(304) 모두, 마이크로프로세서, 디지털 신호 프로세서 또는 다른 프로세싱 유닛 또는 당업계에 공지된 컴포넌트를 포함할 수 있다. 대안적으로 또는 추가로, 본원에서 기능적으로 설명된 것은 적어도 부분적으로 하나 또는 둘 이상의 하드웨어 로직 컴포넌트들에 의해 수행될 수 있다. 예를 들어, 한정됨 없이, 사용될 수 있는 예시적인 유형의 하드웨어 로직 컴포넌트는 FPGA, ASIC, ASSP, SOC, CPLD 등을 포함한다. 추가적으로, 프로세서(들)(2000) 각각은, 프로그램 모듈, 프로그램 데이터, 및/또는 하나 또는 둘 이상의 운영 체제들을 또한 저장할 수 있는 자체 로컬 메모리를 구비할 수 있다.

[0171] 메모리(2002)는, 컴퓨터 판독 가능 명령어들, 데이터 구조들, 프로그램 모듈들 또는 다른 데이터와 같은 정보의 저장을 위한 임의의 방법 또는 기술로 구현된 휘발성 및 비휘발성 메모리, 착탈식 및 비착탈식 매체를 포함할 수 있다. 이러한 메모리는, 원하는 정보를 저장하는 데 사용될 수 있으며 컴퓨팅 디바이스에 의해서 액세스될 수 있는 RAM, ROM, EEPROM, 플래시 메모리 또는 다른 메모리 기술, CD-ROM, DVD 또는 기타 광 저장 디바이스, 자기 카세트, 자기 테이프, 자기 디스크 저장 디바이스, 또는 기타 자기 저장 디바이스, RAID 저장 시스템, 또는 임의의 다른 매체를 포함하나, 이에 한정되지 않는다. 메모리(2002)는 CRSM로서 구현될 수 있으며, 이는 메모리(2002)에 저장된 명령어들을 실행하기 위해 프로세서(들)(2000)에 의해 액세스 가능한 임의의 이용 가능한 물리적 매체일 수 있다. 하나의 기본 구현예에서, CRSM은 RAM 및 플래시 메모리를 포함할 수 있다. 다른 구현예에서, CRSM은, ROM, EEPROM, 또는 원하는 정보를 저장하기 위해서 사용될 수 있고, 프로세서(들)(2000)에 의해서 액세스될 수 있는 임의의 다른 유형(tangible)의 매체를 포함할 수 있으나, 이에 한정되지 않는다.

[0172] 일반적으로, 호스트 컴퓨터(106)는 본원에 설명된 기법들, 기능, 및/또는 동작들을 구현하도록 구성된 로직(예를 들어, 소프트웨어, 하드웨어, 및/또는 펌웨어 등)을 포함할 수 있다. 컴퓨터 판독 가능 매체(2002)는, 본원에 설명된 기법, 기능, 및/또는 동작을 수행하기 위해 프로세서(들)(2000) 상에서 실행되도록 구성될 수 있는 다양한 모듈, 예컨대 명령어, 데이터스토어 등을 포함할 수 있다. 운영 체제(2004), 비디오 게임(게이밍 애플리케이션(2010))과 같은 애플리케이션들(2008)을 포함하는 비디오 게임 클라이언트(2006), 및 렌더 컴포넌트(2012)의 형태의 예시적인 기능 모듈들은 컴퓨터 판독 가능 매체(2002)에 저장되고, 프로세서(들)(2000) 상에서 실행 가능한 것으로 도시된다. 일부 실시형태에서, 추가적인 또는 상이한 기능 모듈들이 컴퓨터 판독 가능 매체(2002)에 저장될 수 있고, 프로세서(들)(2000) 상에서 실행 가능할 수 있다.

[0173] 운영 체제(2004)는 다른 모듈들을 위하여 호스트 컴퓨터(106) 내의 그리고 이에 결합된 하드웨어를 관리하도록 구성될 수 있다. 비디오 게임 클라이언트(2006)는 비디오 게임들(또는 비디오 게임 프로그램들)과 같은 프로그램들을 론치(launch)하고 실행하도록 구성되는 실행 가능 클라이언트 애플리케이션을 나타낼 수 있다. 다시 말

해서, 비디오 게임 클라이언트(2006)는, HMD(102) 및 호스트 컴퓨터(106)를 포함하는 시스템 상에서 비디오 게임들을 플레이하기 위해서 사용 가능한 게이밍 소프트웨어를 포함할 수 있다. 비디오 게임 클라이언트(2006)가 설치된 상태에서, 호스트 컴퓨터(106)는 컴퓨터 네트워크(예: 인터넷)를 통해 원격 시스템으로부터 비디오 게임을 수신(예를 들어, 다운로드, 스트리밍 등)하는, 그리고 비디오 게임 클라이언트(2006)를 통해 비디오 게임을 실행하는 능력을 가질 수 있다. 비디오 게임이 호스트 컴퓨터(106) 상의 다운로드 및 실행을 위해 개별적으로 구매될 수 있는 직접 구매 모델, 구독 기반 모델, 비디오 게임이 일정 기간 동안 임대 또는 리스되는 콘텐츠 배포 모델과 같은 임의의 유형의 콘텐츠 배포 모델이 이러한 목적을 위해 활용될 수 있다. 따라서, 호스트 컴퓨터(106)는 비디오 게임 라이브러리(2014) 내에 하나 또는 둘 이상의 비디오 게임들을 포함할 수 있다. 이러한 비디오 게임들은 비디오 게임 클라이언트(2006)를 로딩함으로써 검색 및 실행될 수 있다. 일 예에서, 사용자(104)는, 게임의 실행을 시작하기 위해 비디오 게임 클라이언트(2006)를 로딩하고 비디오 게임을 선택함으로써 구매한 그리고 비디오 게임 라이브러리(2014)에 다운로드한 다수의 비디오 게임들 중 하나를 플레이하도록 선택할 수 있다. 비디오 게임 클라이언트(2006)는 사용자들이 크리덴셜(credential)(예를 들어, 사용자 계정, 비밀번호 등)을 사용하여 비디오 게임 서비스에 로그인하는 것을 허용할 수 있다.

[0174] 호스트 컴퓨터(106) 상에서 실행되는 애플리케이션(들)(2008)은 그래픽 기반 애플리케이션(들)(2008)(예를 들어, 비디오 게임(2010))일 수 있다. 애플리케이션(2008)은 일련의 프레임에 대한 픽셀 데이터를 생성하도록 구성되고, 픽셀 데이터는 궁극적으로 HMD(102)의 디스플레이 패널(들)(1916) 상에 대응되는 이미지를 제시하기 위해서 사용된다. 런타임 동안, 주어진 프레임에 대해, 렌더 컴포넌트(2012)는 이 프레임에 대한 예측된 "조명 시간"을 결정할 수 있다. 프레임에 대한 이 예측된 "조명 시간"은 HMD(102)의 디스플레이 패널(들)(1916)의 발광 소자들이 프레임에 대해 조명할 시간을 나타낸다. 이 예측은, 무엇보다도, 호스트 컴퓨터(106)와 HMD(102) 사이의 무선 통신 링크의 고유 레이턴시뿐만 아니라, 예측된 렌더링 시간 및/또는 프레임 버퍼(들)로부터 픽셀의 알려진 스캔 아웃 시간을 처리할 수 있다. 다시 말해서, 무선 통신 링크에 대한 예측은 유선 통신 링크에 대한 예측과 다를 수 있다. 예를 들어, 렌더 컴포넌트(2012)는, 유선 통신 링크에 대해, 미래의 제1 양의 시간(예를 들어, 미래에 약 22 밀리초)인 조명 시간을 예측할 수 있는 반면, 렌더 컴포넌트(2012)는, 무선 통신 링크에 대해, 데이터를 유선 연결 대 무선 연결을 통해 전송할 때 레이턴시의 고유한 차이들로 인해 미래에 더 많은 제2 양의 시간(예를 들어, 미래에 약 44 밀리초)인 조명 시간을 예측할 수 있다.

[0175] 본원에 설명된 바와 같이, 호스트 컴퓨터(106)는 또한, HMD(102)의 머리 추적 모듈(1930)에 의해 생성된 머리 추적 데이터(208)를 HMD(102)로부터 수신할 수 있다. 이 머리 추적 데이터(208)는, HMD(102)의 타겟 프레임 레이트 및/또는 재생율에 대응되는 주파수, 또는 다른(예를 들어, 더 높은) 주파수, 예를 들어, 1000 Hz(또는 1 밀리초마다 1개의 센서 판독 값)와 같은 임의의 적절한 주파수에서 생성되고/되거나 전송될 수 있다. 렌더 컴포넌트(2012)는, 머리 추적 데이터(208)에 적어도 부분적으로 기반하여 HMD(102)가 예측 조명 시간에 있을 예측 포즈를 결정하도록 구성된다. 다음으로, 렌더 컴포넌트(2012)는 예측된 포즈를 나타내는 포즈 데이터를 예측된 포즈에 기초하여 프레임을 렌더링(예를 들어, 프레임에 대한 픽셀 데이터 생성)하기 위해 실행 애플리케이션(2008)에 제공할 수 있고, 렌더 컴포넌트(2012)는 프레임(300)과 연관된 픽셀 데이터(110(1))를 애플리케이션(2008)으로부터 획득할 수 있다.

[0176] 호스트 컴퓨터(106)는, 네트워크 및/또는 HMD(102)와 같은 제2 디바이스에 대한 무선 연결을 용이하게 하기 위해 안테나(2020)에 결합된 무선 유닛(2018)을, 한정됨 없이, 포함하는 통신 인터페이스(들)(2016)를 더 포함할 수 있다. 무선 유닛(2018)은 Wi-Fi, 블루투스, 무선 주파수(RF) 등과 같은 다양한 무선 테크놀로지들 중 하나 이상을 구현할 수 있다. 호스트 컴퓨터(106)는 네트워크 및/또는 HMD(102)와 같은 제2 디바이스에 대한 유선 연결을 용이하게 하는 물리적 포트들을 더 포함할 수 있다는 것이 이해될 것이다.

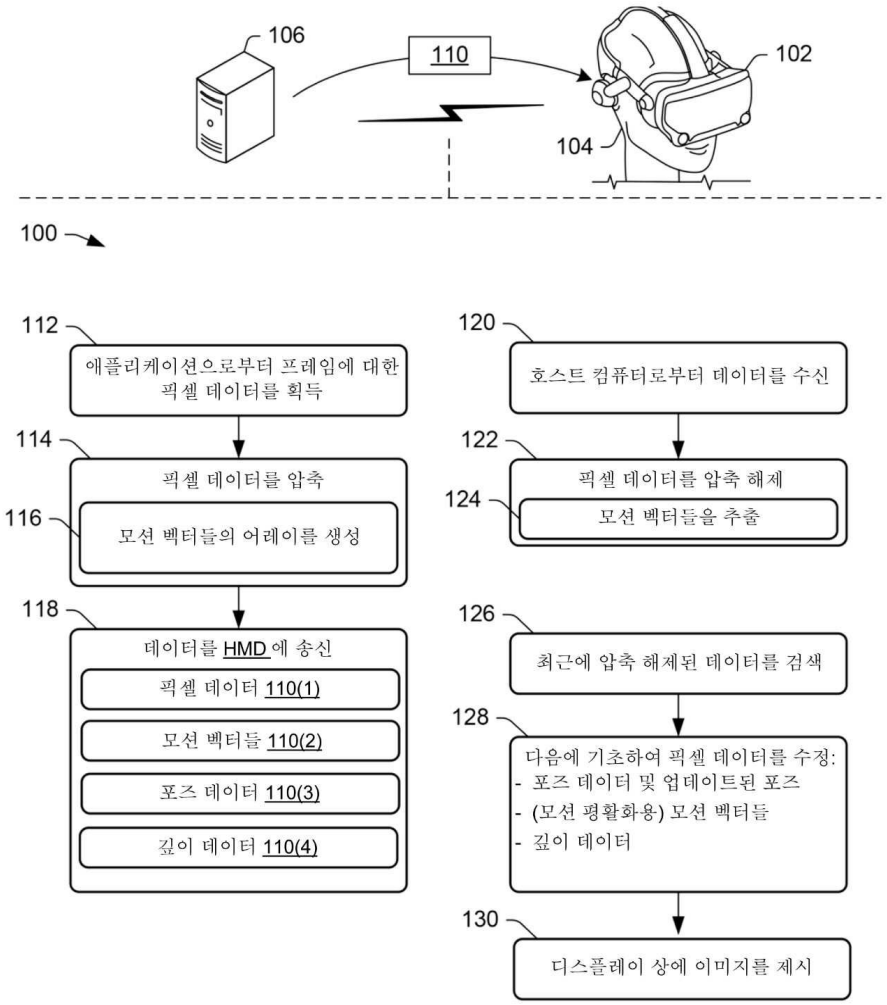
[0177] 일반적으로, 호스트 컴퓨터(106)는 입력 디바이스들(2022) 및 출력 디바이스들(2024)을 갖는다. 입력 디바이스들(2022)은 키보드, 키패드, 마우스, 터치 스크린, 조이스틱, 제어 버튼, 마이크, 카메라 등일 수 있다. 출력 디바이스들(2024)은, 디스플레이(들), 광 요소(예를 들어, LED), 햅틱 감각들을 생성하기 위한 진동기, 및/또는 스피커(들)(예를 들어, 헤드폰들) 등을, 이에 한정됨 없이, 포함할 수 있다. 일부 실시형태에서, HMD(102) 상에서 구현되는 것으로 도시된 컴포넌트들의 서브세트가 호스트 컴퓨터(106) 또는 HMD(102)와 별개인 다른 컴퓨팅 디바이스 상에서 구현될 수 있고/있거나, 호스트 컴퓨터(106) 상에서 구현되는 것으로 도시된 컴포넌트들의 서브세트가 호스트 컴퓨터(106)와 별개인 HMD(102) 또는 다른 컴퓨팅 디바이스 상에서 구현될 수 있다는 것이 이해될 것이다.

[0178] 첨부된 청구범위에 정의된 특허 대상은 구조적 특징들에 특정되는 술어로 설명되었지만, 설명된 이 특정 특징들에 반드시 한정되지 않는다는 것이 이해될 것이다. 오히려, 특정 특징들은 청구범위를 구현하는 예시적인 형태

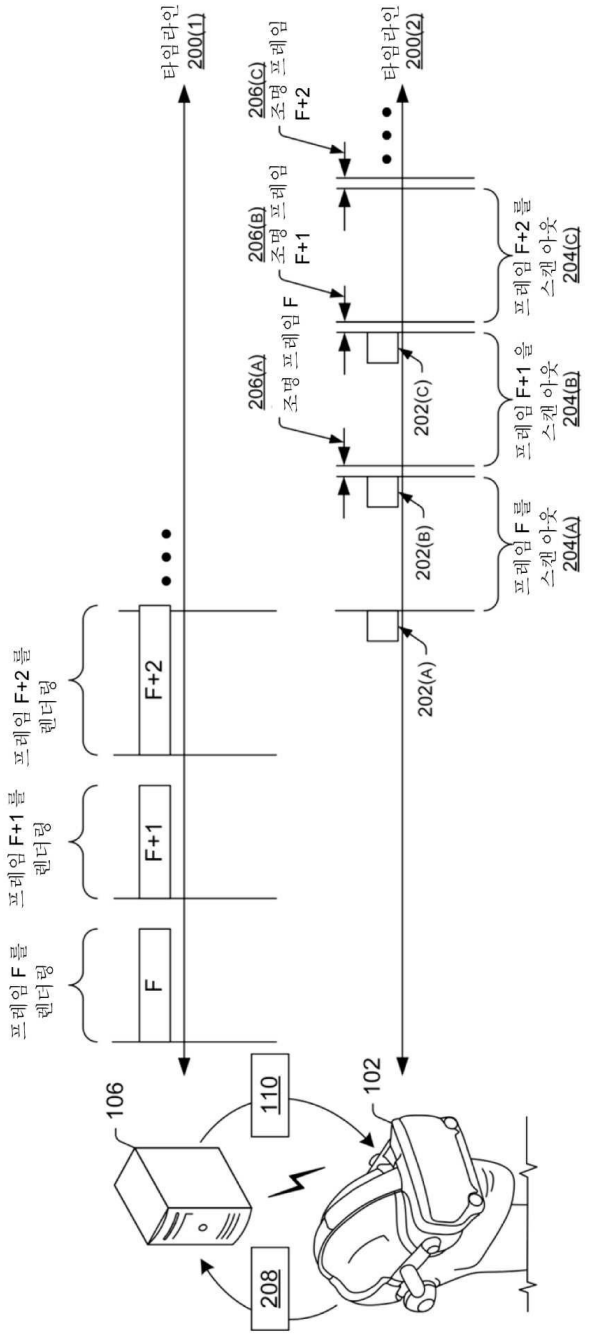
들로서 개시된다.

도면

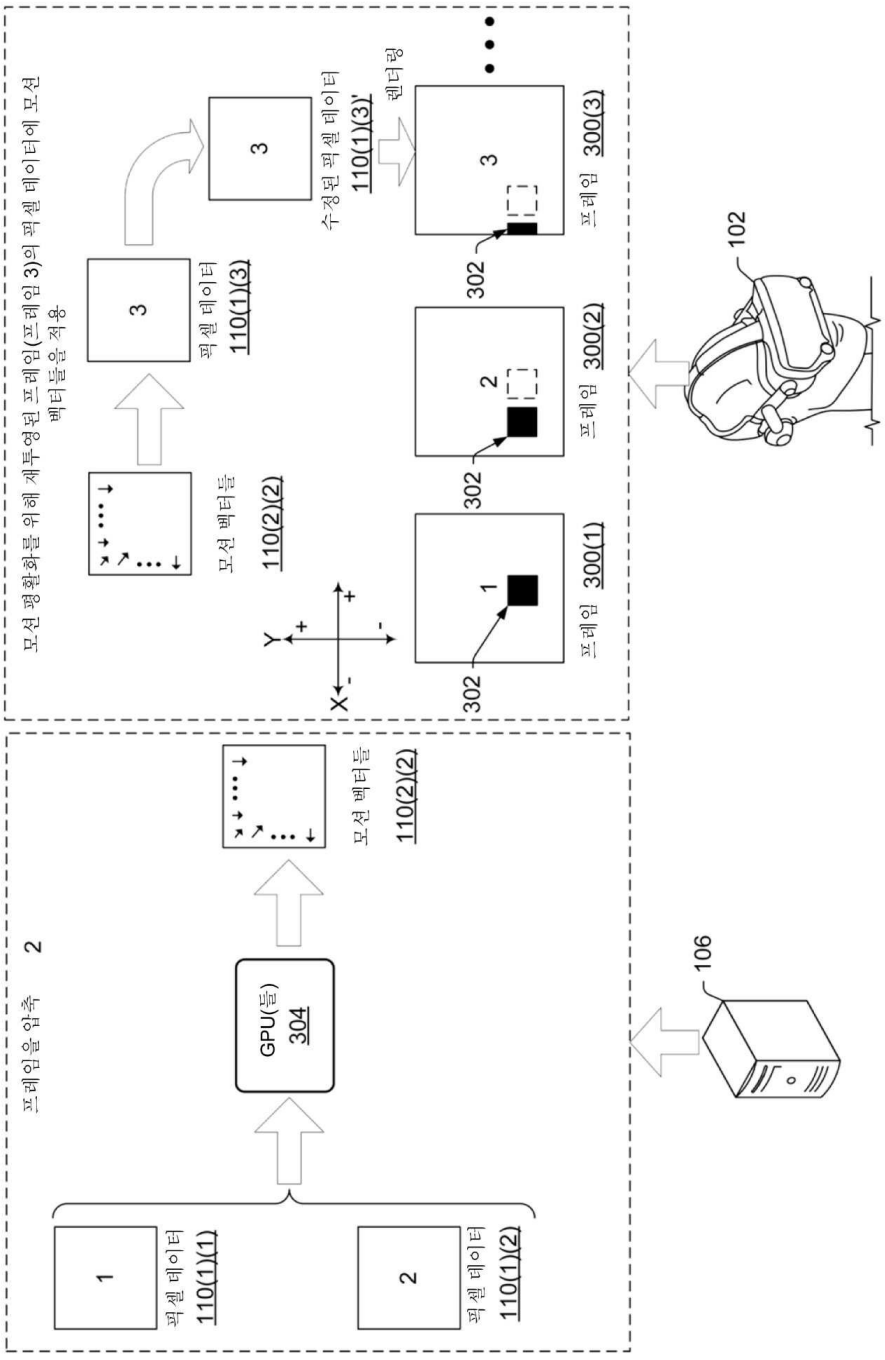
도면1



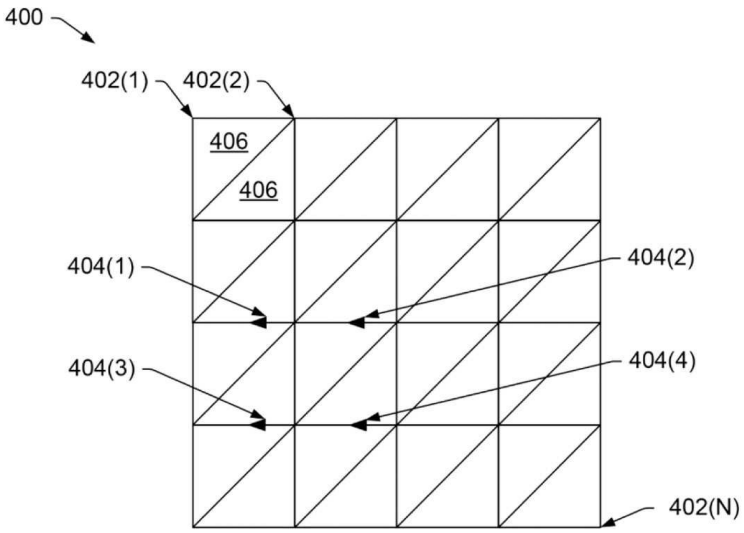
도면2


타임라인
200(1)
타임라인
200(2)
206(C)
조명 프레임
F+2
206(B)
조명 프레임
F+1
206(A)
조명 프레임 F
프레임 F 를
렌더링
프레임 F+1 를
렌더링
프레임 F+2 를
렌더링
F
F+1
F+2
202(A)
202(B)
202(C)
프레임 F 를
스캔 아웃
204(A)
프레임 F+1 을
스캔 아웃
204(B)
프레임 F+2 를
스캔 아웃
204(C)
106
110
102
208

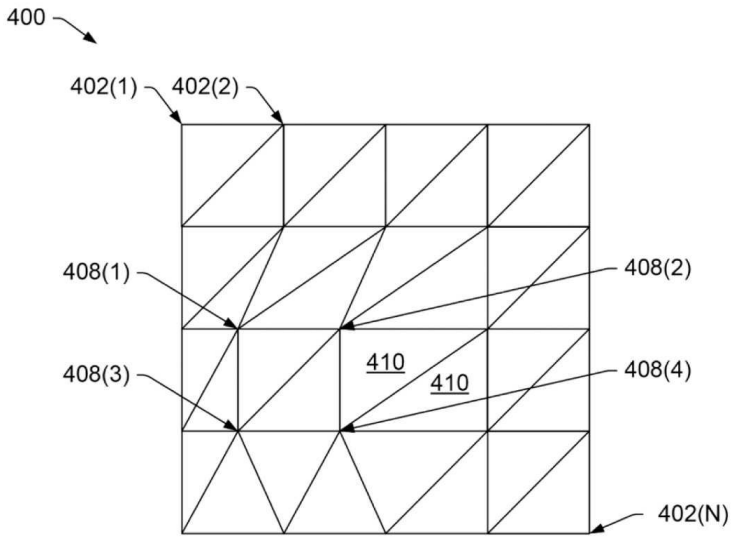
도면3


프레임을 압축 2
1
픽셀 데이터
110(1)(1)
2
픽셀 데이터
110(1)(2)
GPU(들)
304
모션 벡터들
110(2)(2)
106
모션 평활화를 위해 재투영된 프레임(프레임 3)의 픽셀 데이터에 모션 벡터들을 적용
3
픽셀 데이터
110(1)(3)
모션 벡터들
110(2)(2)
Y
+
-
X
-
+
302
1
프레임 300(1)
302
2
프레임 300(2)
302
3
프레임 300(3)
수정된 픽셀 데이터
110(1)(3)'
렌더링
102

도면4

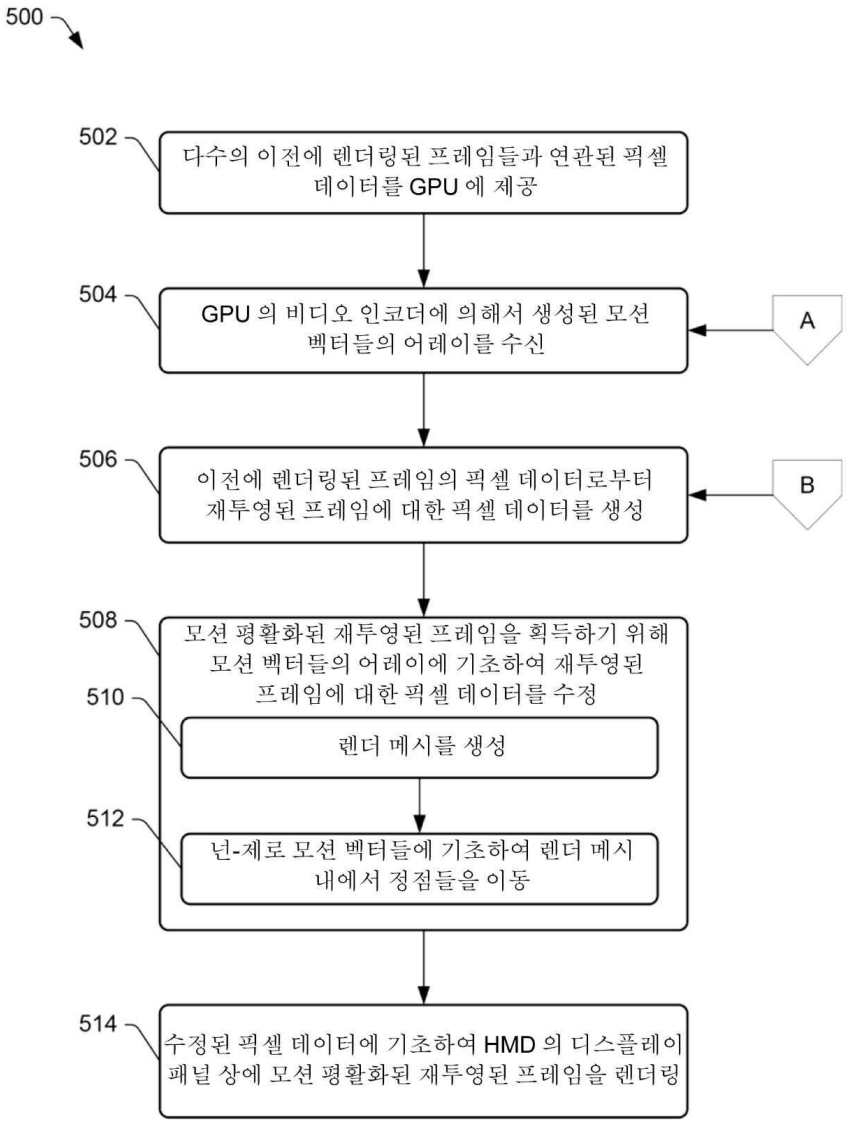


모션 평활화 전

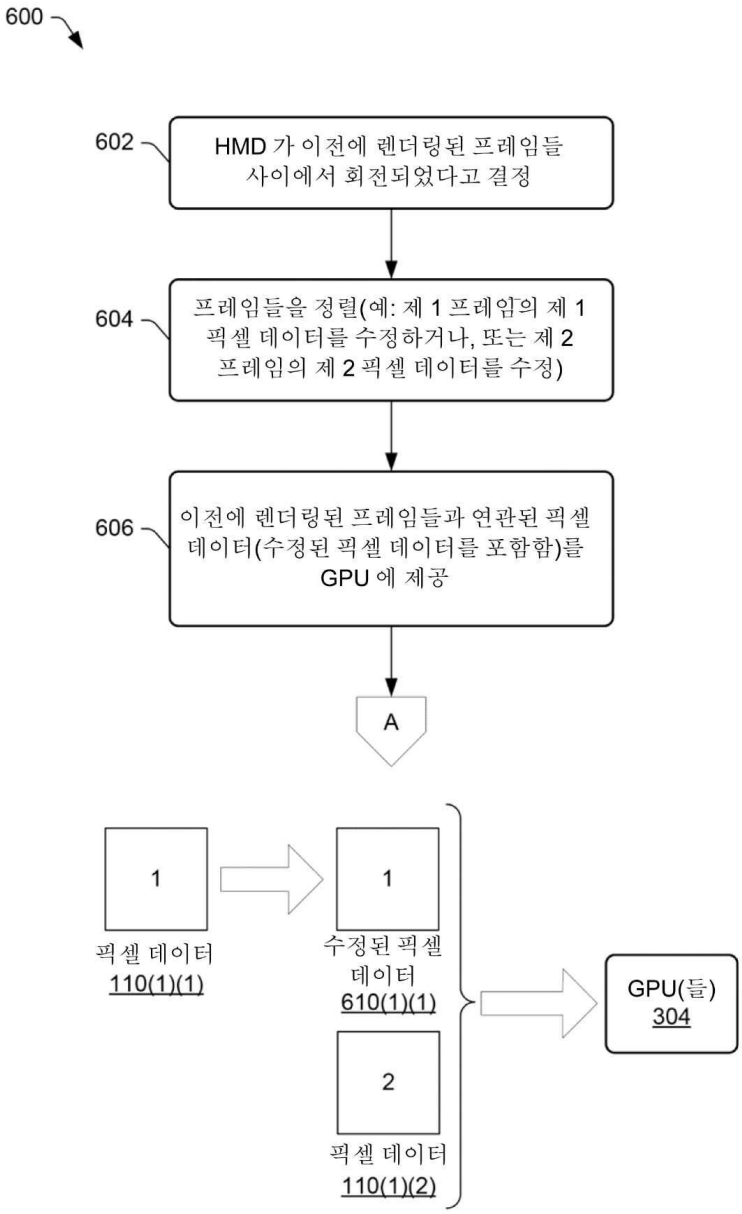


모션 평활화 후

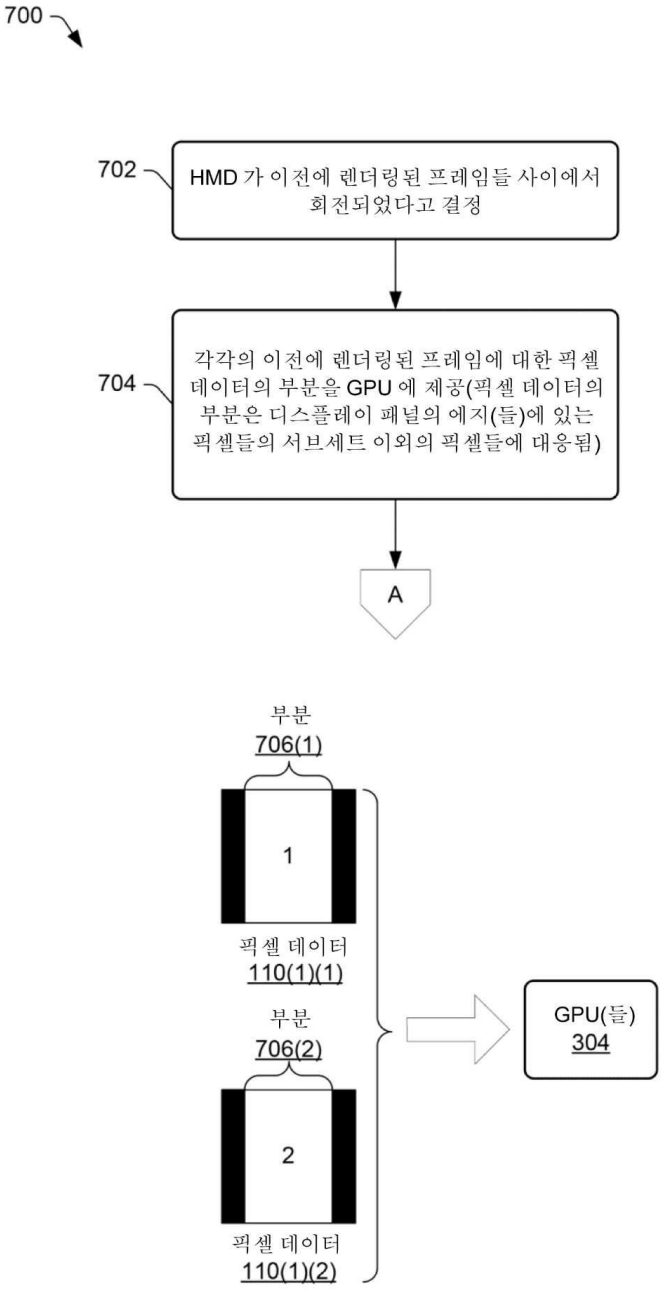
도면5


500
502
다수의 이전에 렌더링된 프레임들과 연관된 픽셀 데이터를 GPU 에 제공
504
GPU 의 비디오 인코더에 의해서 생성된 모션 벡터들의 어레이를 수신
A
506
이전에 렌더링된 프레임의 픽셀 데이터로부터 재투영된 프레임에 대한 픽셀 데이터를 생성
B
508
모션 평활화된 재투영된 프레임을 획득하기 위해 모션 벡터들의 어레이에 기초하여 재투영된 프레임에 대한 픽셀 데이터를 수정
510
렌더 메시를 생성
512
넌-제로 모션 벡터들에 기초하여 렌더 메시 내에서 정점들을 이동
514
수정된 픽셀 데이터에 기초하여 HMD 의 디스플레이 패널 상에 모션 평활화된 재투영된 프레임을 렌더링

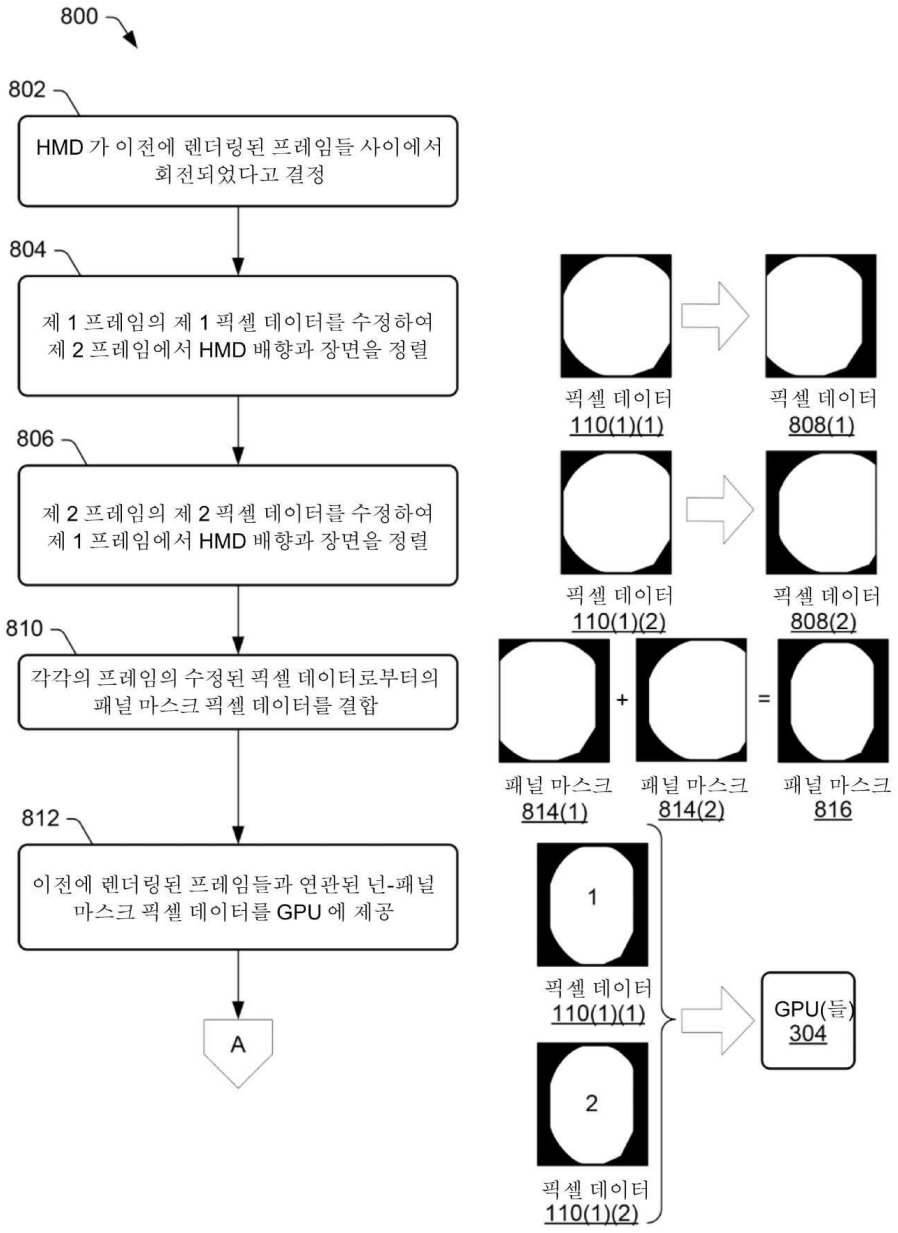
도면6


600
602
HMD 가 이전에 렌더링된 프레임들 사이에서 회전되었다고 결정
604
프레임들을 정렬(예: 제 1 프레임의 제 1 픽셀 데이터를 수정하거나, 또는 제 2 프레임의 제 2 픽셀 데이터를 수정)
606
이전에 렌더링된 프레임들과 연관된 픽셀 데이터(수정된 픽셀 데이터를 포함함)를 GPU 에 제공
A
1
픽셀 데이터
110(1)(1)
1
수정된 픽셀 데이터
610(1)(1)
2
픽셀 데이터
110(1)(2)
GPU(들)
304

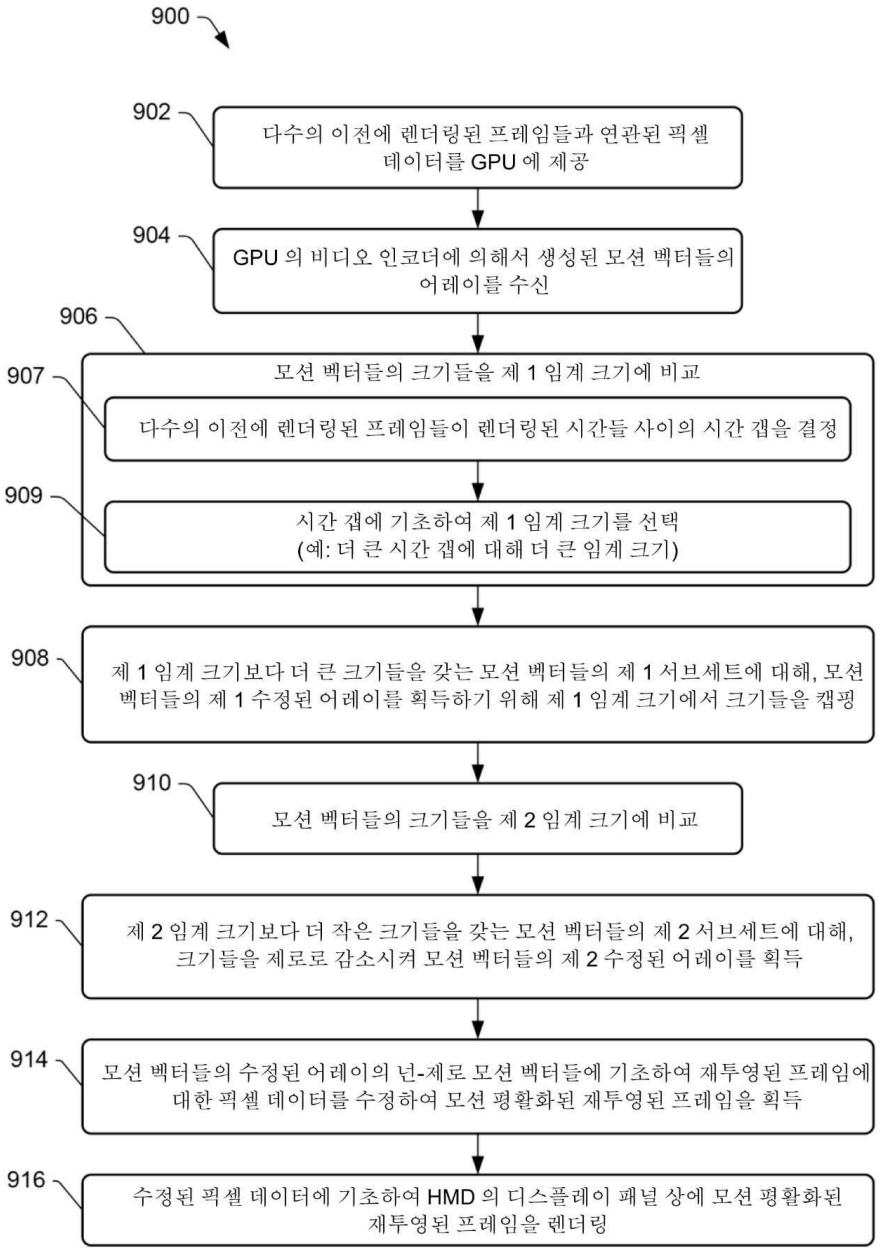
도면7


700
702
HMD 가 이전에 렌더링된 프레임들 사이에서 회전되었다고 결정
704
각각의 이전에 렌더링된 프레임에 대한 픽셀 데이터의 부분을 GPU 에 제공(픽셀 데이터의 부분은 디스플레이 패널의 에지(들)에 있는 픽셀들의 서브세트 이외의 픽셀들에 대응됨)
A
부분
706(1)
1
픽셀 데이터
110(1)(1)
부분
706(2)
2
픽셀 데이터
110(1)(2)
GPU(들)
304

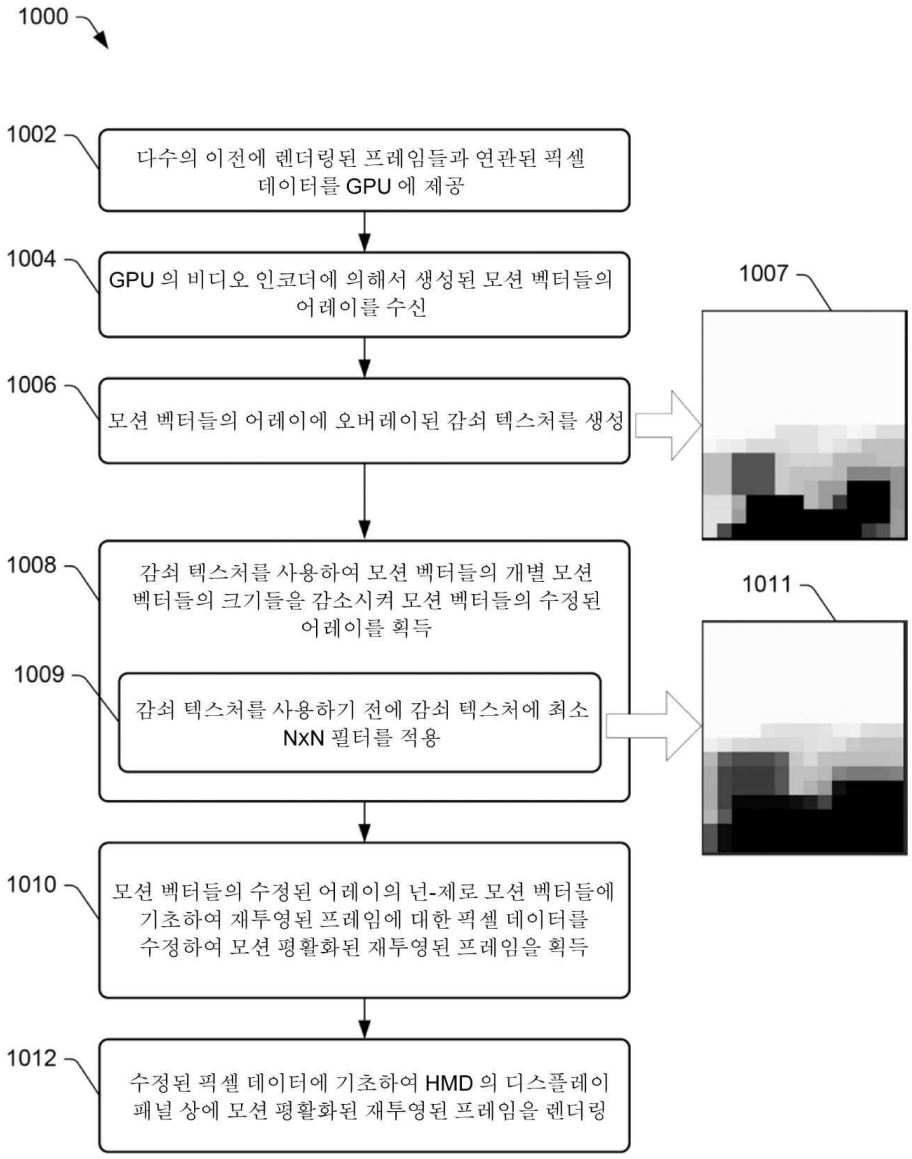
도면8


800
802
HMD 가 이전에 렌더링된 프레임들 사이에서 회전되었다고 결정
804
제 1 프레임의 제 1 픽셀 데이터를 수정하여 제 2 프레임에서 HMD 배향과 장면을 정렬
806
제 2 프레임의 제 2 픽셀 데이터를 수정하여 제 1 프레임에서 HMD 배향과 장면을 정렬
810
각각의 프레임의 수정된 픽셀 데이터로부터의 패널 마스크 픽셀 데이터를 결합
812
이전에 렌더링된 프레임들과 연관된 넌-패널 마스크 픽셀 데이터를 GPU 에 제공
A
픽셀 데이터 110(1)(1)
픽셀 데이터 808(1)
픽셀 데이터 110(1)(2)
픽셀 데이터 808(2)
+
=
패널 마스크 814(1)
패널 마스크 814(2)
패널 마스크 816
1
픽셀 데이터 110(1)(1)
2
픽셀 데이터 110(1)(2)
GPU(들) 304

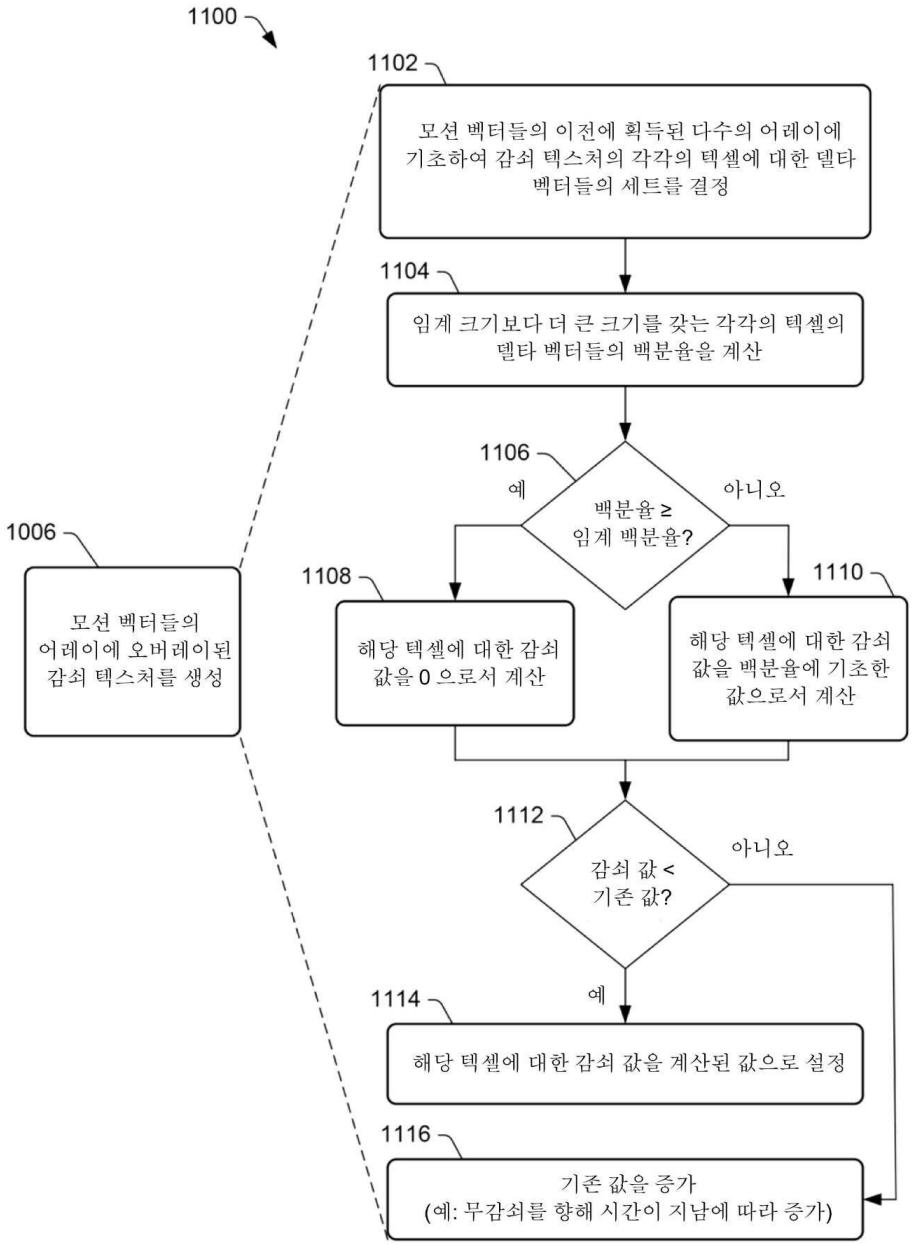
도면9


900
902
다수의 이전에 렌더링된 프레임들과 연관된 픽셀 데이터를 GPU 에 제공
904
GPU 의 비디오 인코더에 의해서 생성된 모션 벡터들의 어레이를 수신
906
모션 벡터들의 크기들을 제 1 임계 크기에 비교
907
다수의 이전에 렌더링된 프레임들이 렌더링된 시간들 사이의 시간 갭을 결정
909
시간 갭에 기초하여 제 1 임계 크기를 선택 (예: 더 큰 시간 갭에 대해 더 큰 임계 크기)
908
제 1 임계 크기보다 더 큰 크기들을 갖는 모션 벡터들의 제 1 서브세트에 대해, 모션 벡터들의 제 1 수정된 어레이를 획득하기 위해 제 1 임계 크기에서 크기들을 캡핑
910
모션 벡터들의 크기들을 제 2 임계 크기에 비교
912
제 2 임계 크기보다 더 작은 크기들을 갖는 모션 벡터들의 제 2 서브세트에 대해, 크기들을 제로로 감소시켜 모션 벡터들의 제 2 수정된 어레이를 획득
914
모션 벡터들의 수정된 어레이의 넌-제로 모션 벡터들에 기초하여 재투영된 프레임에 대한 픽셀 데이터를 수정하여 모션 평활화된 재투영된 프레임을 획득
916
수정된 픽셀 데이터에 기초하여 HMD 의 디스플레이 패널 상에 모션 평활화된 재투영된 프레임을 렌더링

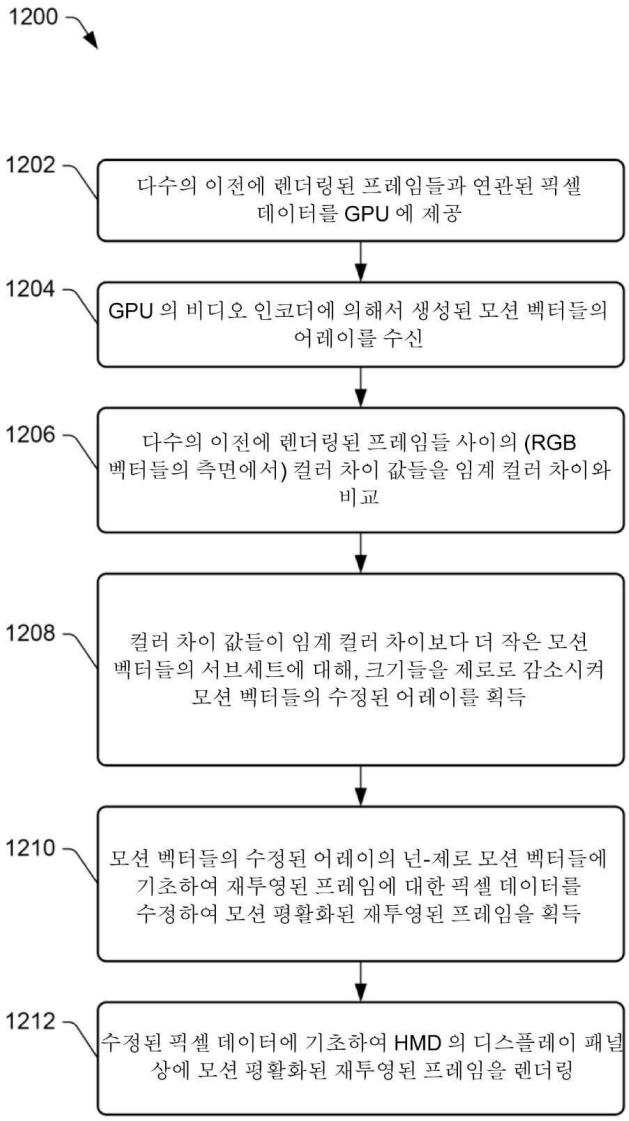
도면10


1000
1002
다수의 이전에 렌더링된 프레임들과 연관된 픽셀 데이터를 GPU 에 제공
1004
GPU 의 비디오 인코더에 의해서 생성된 모션 벡터들의 어레이를 수신
1006
모션 벡터들의 어레이에 오버레이된 감쇠 텍스처를 생성
1007
1008
감쇠 텍스처를 사용하여 모션 벡터들의 개별 모션 벡터들의 크기들을 감소시켜 모션 벡터들의 수정된 어레이를 획득
1009
감쇠 텍스처를 사용하기 전에 감쇠 텍스처에 최소 NxN 필터를 적용
1011
1010
모션 벡터들의 수정된 어레이의 넌-제로 모션 벡터들에 기초하여 재투영된 프레임에 대한 픽셀 데이터를 수정하여 모션 평활화된 재투영된 프레임을 획득
1012
수정된 픽셀 데이터에 기초하여 HMD 의 디스플레이 패널 상에 모션 평활화된 재투영된 프레임을 렌더링

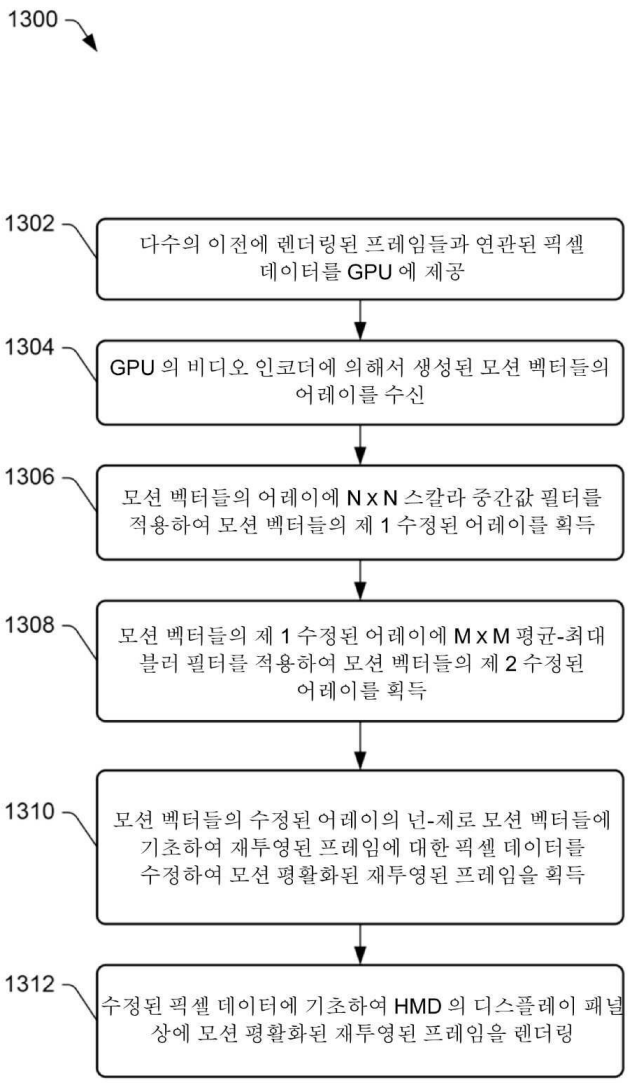
도면11


1100
1102
모션 벡터들의 이전에 획득된 다수의 어레이에 기초하여 감쇠 텍스처의 각각의 텍셀에 대한 델타 벡터들의 세트를 결정
1104
임계 크기보다 더 큰 크기를 갖는 각각의 텍셀의 델타 벡터들의 백분율을 계산
1106
백분율 ≥ 임계 백분율?
예
아니오
1006
모션 벡터들의 어레이에 오버레이된 감쇠 텍스처를 생성
1108
해당 텍셀에 대한 감쇠 값을 0 으로서 계산
1110
해당 텍셀에 대한 감쇠 값을 백분율에 기초한 값으로서 계산
1112
감쇠 값 < 기존 값?
아니오
예
1114
해당 텍셀에 대한 감쇠 값을 계산된 값으로 설정
1116
기존 값을 증가
(예: 무감쇠를 향해 시간이 지남에 따라 증가)

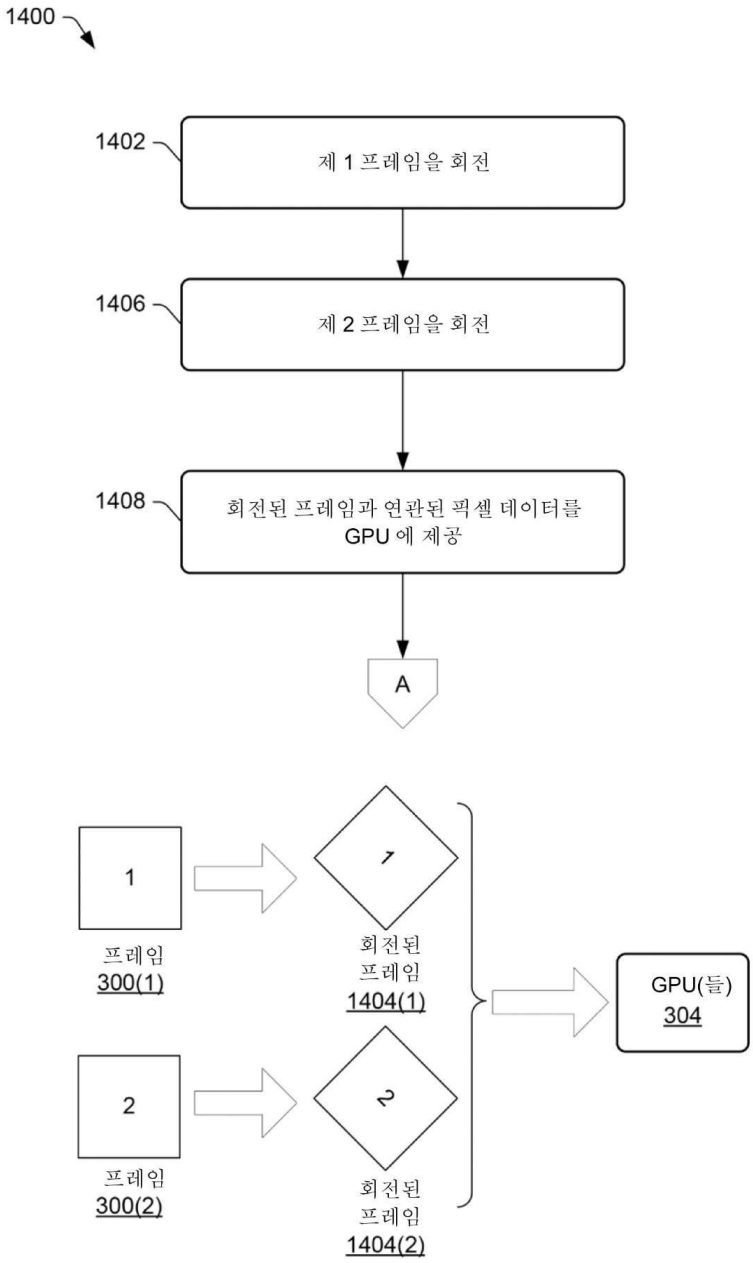
도면12


1200
1202
다수의 이전에 렌더링된 프레임들과 연관된 픽셀 데이터를 GPU 에 제공
1204
GPU 의 비디오 인코더에 의해서 생성된 모션 벡터들의 어레이를 수신
1206
다수의 이전에 렌더링된 프레임들 사이의 (RGB 벡터들의 측면에서) 컬러 차이 값들을 임계 컬러 차이와 비교
1208
컬러 차이 값들이 임계 컬러 차이보다 더 작은 모션 벡터들의 서브세트에 대해, 크기들을 제로로 감소시켜 모션 벡터들의 수정된 어레이를 획득
1210
모션 벡터들의 수정된 어레이의 넌-제로 모션 벡터들에 기초하여 재투영된 프레임에 대한 픽셀 데이터를 수정하여 모션 평활화된 재투영된 프레임을 획득
1212
수정된 픽셀 데이터에 기초하여 HMD 의 디스플레이 패널 상에 모션 평활화된 재투영된 프레임을 렌더링

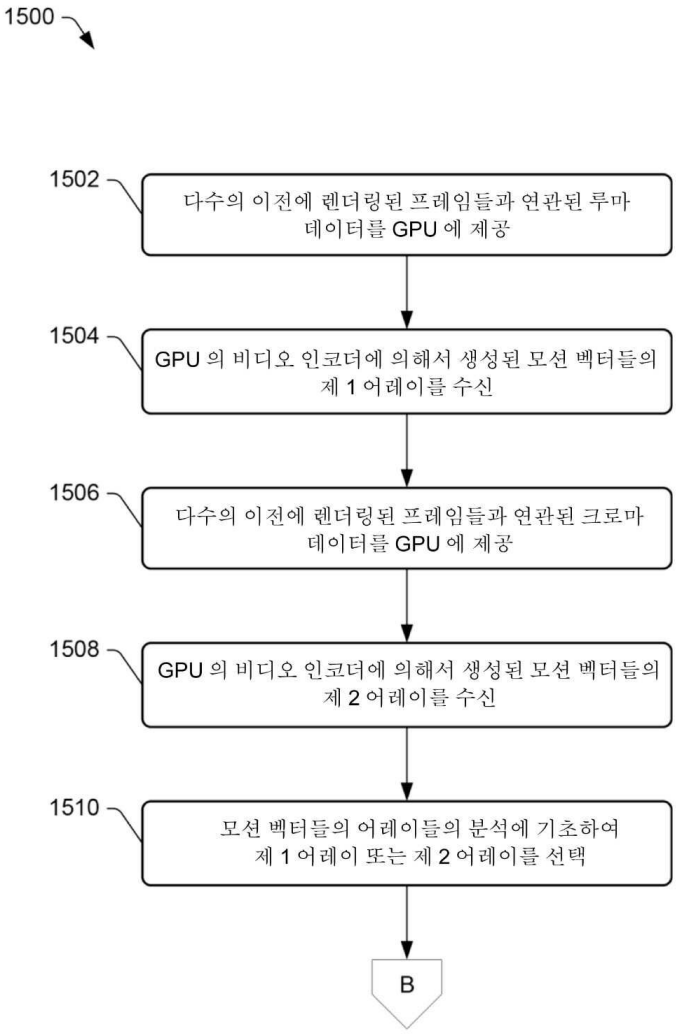
도면13


1300
1302
다수의 이전에 렌더링된 프레임들과 연관된 픽셀 데이터를 GPU 에 제공
1304
GPU 의 비디오 인코더에 의해서 생성된 모션 벡터들의 어레이를 수신
1306
모션 벡터들의 어레이에 N x N 스칼라 중간값 필터를 적용하여 모션 벡터들의 제 1 수정된 어레이를 획득
1308
모션 벡터들의 제 1 수정된 어레이에 M x M 평균-최대 블러 필터를 적용하여 모션 벡터들의 제 2 수정된 어레이를 획득
1310
모션 벡터들의 수정된 어레이의 넌-제로 모션 벡터들에 기초하여 재투영된 프레임에 대한 픽셀 데이터를 수정하여 모션 평활화된 재투영된 프레임을 획득
1312
수정된 픽셀 데이터에 기초하여 HMD 의 디스플레이 패널 상에 모션 평활화된 재투영된 프레임을 렌더링

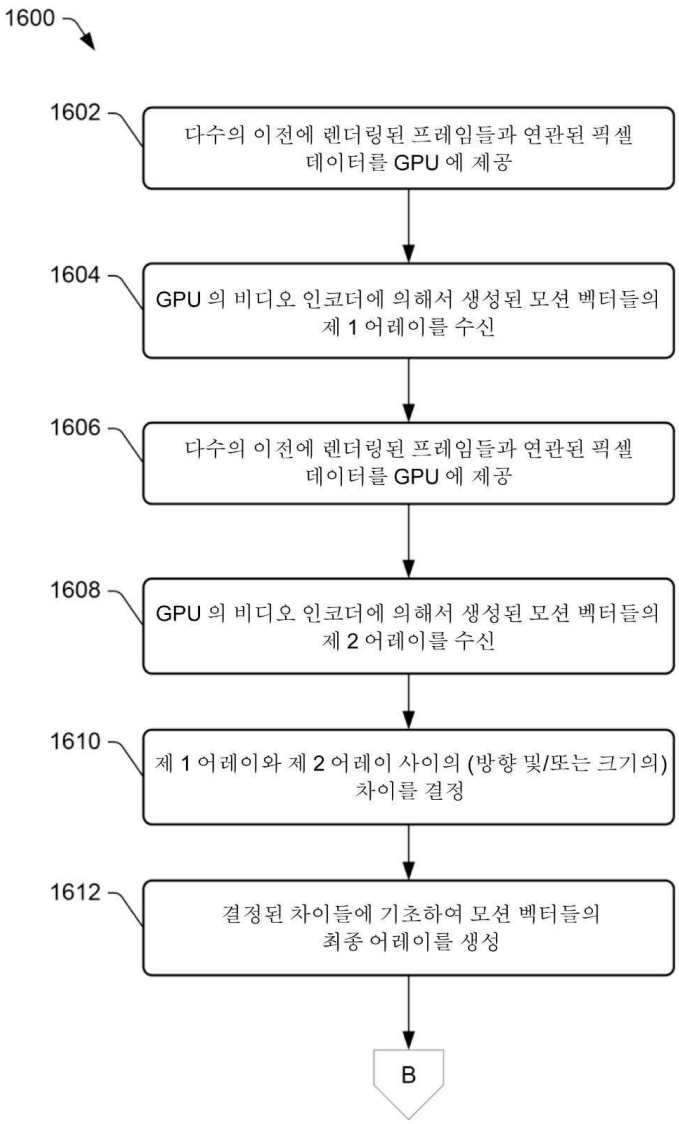
도면14


1400
1402
제 1 프레임을 회전
1406
제 2 프레임을 회전
1408
회전된 프레임과 연관된 픽셀 데이터를 GPU 에 제공
A
1
프레임
300(1)
1
회전된
프레임
1404(1)
2
프레임
300(2)
2
회전된
프레임
1404(2)
GPU(들)
304

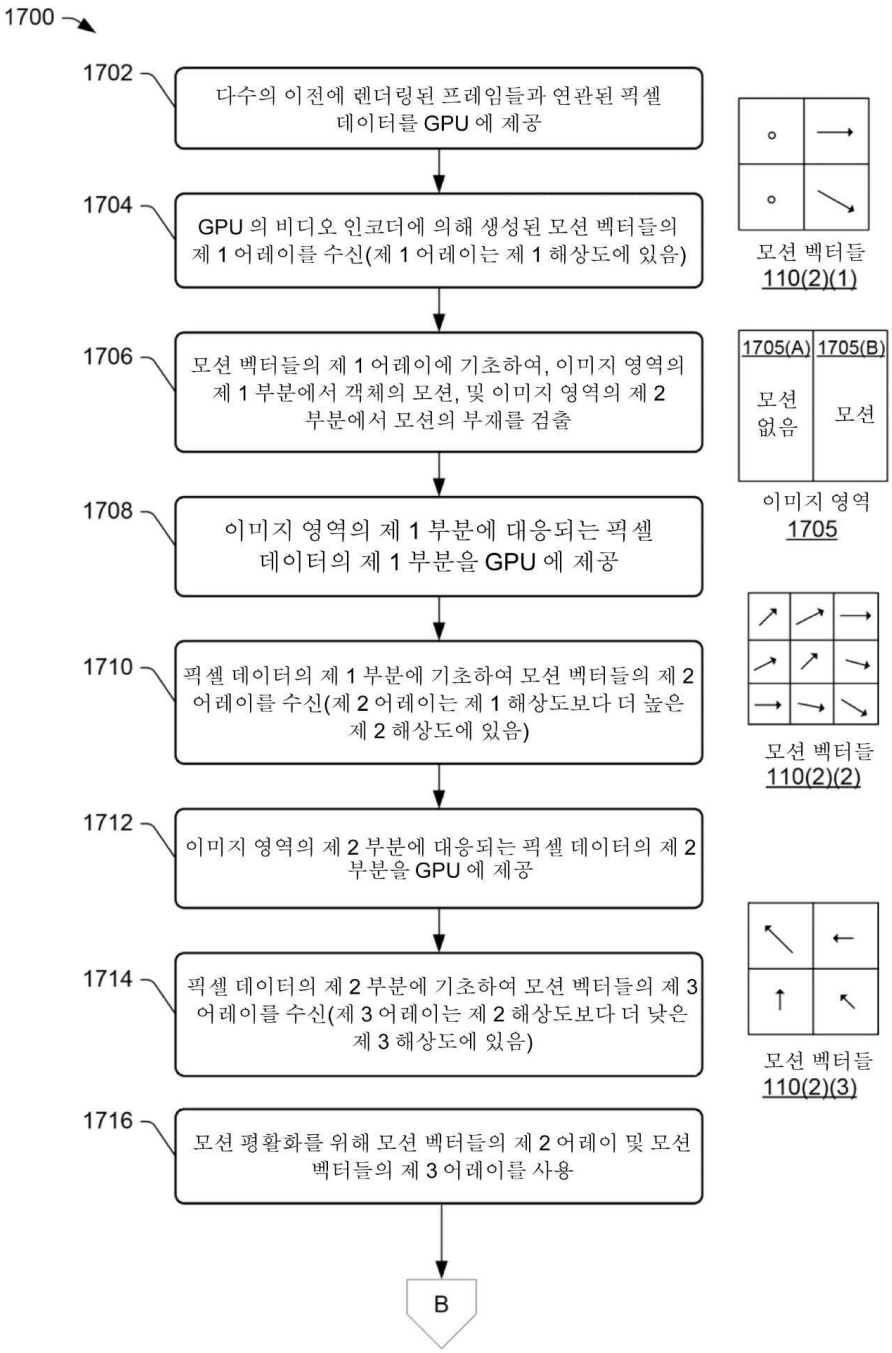
도면15


1500
1502
다수의 이전에 렌더링된 프레임들과 연관된 루마 데이터를 GPU 에 제공
1504
GPU 의 비디오 인코더에 의해서 생성된 모션 벡터들의 제 1 어레이를 수신
1506
다수의 이전에 렌더링된 프레임들과 연관된 크로마 데이터를 GPU 에 제공
1508
GPU 의 비디오 인코더에 의해서 생성된 모션 벡터들의 제 2 어레이를 수신
1510
모션 벡터들의 어레이들의 분석에 기초하여 제 1 어레이 또는 제 2 어레이를 선택
B

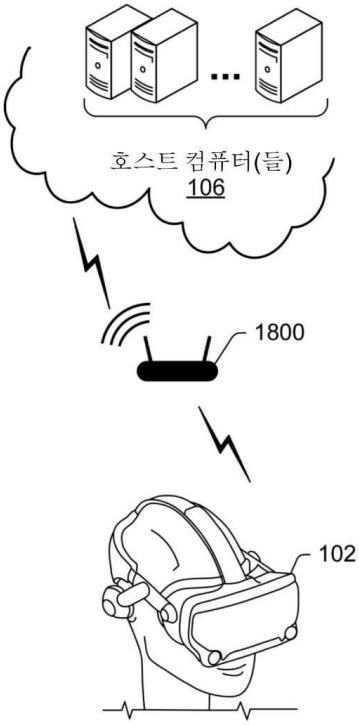
도면16


1600
1602
다수의 이전에 렌더링된 프레임들과 연관된 픽셀 데이터를 GPU 에 제공
1604
GPU 의 비디오 인코더에 의해서 생성된 모션 벡터들의 제 1 어레이를 수신
1606
다수의 이전에 렌더링된 프레임들과 연관된 픽셀 데이터를 GPU 에 제공
1608
GPU 의 비디오 인코더에 의해서 생성된 모션 벡터들의 제 2 어레이를 수신
1610
제 1 어레이와 제 2 어레이 사이의 (방향 및/또는 크기의) 차이를 결정
1612
결정된 차이들에 기초하여 모션 벡터들의 최종 어레이를 생성
B

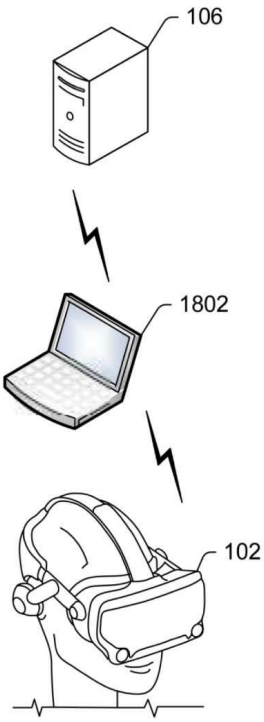
도면17


1700
1702
다수의 이전에 렌더링된 프레임들과 연관된 픽셀 데이터를 GPU 에 제공
1704
GPU 의 비디오 인코더에 의해 생성된 모션 벡터들의 제 1 어레이를 수신(제 1 어레이는 제 1 해상도에 있음)
1706
모션 벡터들의 제 1 어레이에 기초하여, 이미지 영역의 제 1 부분에서 객체의 모션, 및 이미지 영역의 제 2 부분에서 모션의 부재를 검출
1708
이미지 영역의 제 1 부분에 대응되는 픽셀 데이터의 제 1 부분을 GPU 에 제공
1710
픽셀 데이터의 제 1 부분에 기초하여 모션 벡터들의 제 2 어레이를 수신(제 2 어레이는 제 1 해상도보다 더 높은 제 2 해상도에 있음)
1712
이미지 영역의 제 2 부분에 대응되는 픽셀 데이터의 제 2 부분을 GPU 에 제공
1714
픽셀 데이터의 제 2 부분에 기초하여 모션 벡터들의 제 3 어레이를 수신(제 3 어레이는 제 2 해상도보다 더 낮은 제 3 해상도에 있음)
1716
모션 평활화를 위해 모션 벡터들의 제 2 어레이 및 모션 벡터들의 제 3 어레이를 사용
B
모션 벡터들
110(2)(1)
1705(A)
1705(B)
모션 없음
모션
이미지 영역
1705
모션 벡터들
110(2)(2)
모션 벡터들
110(2)(3)

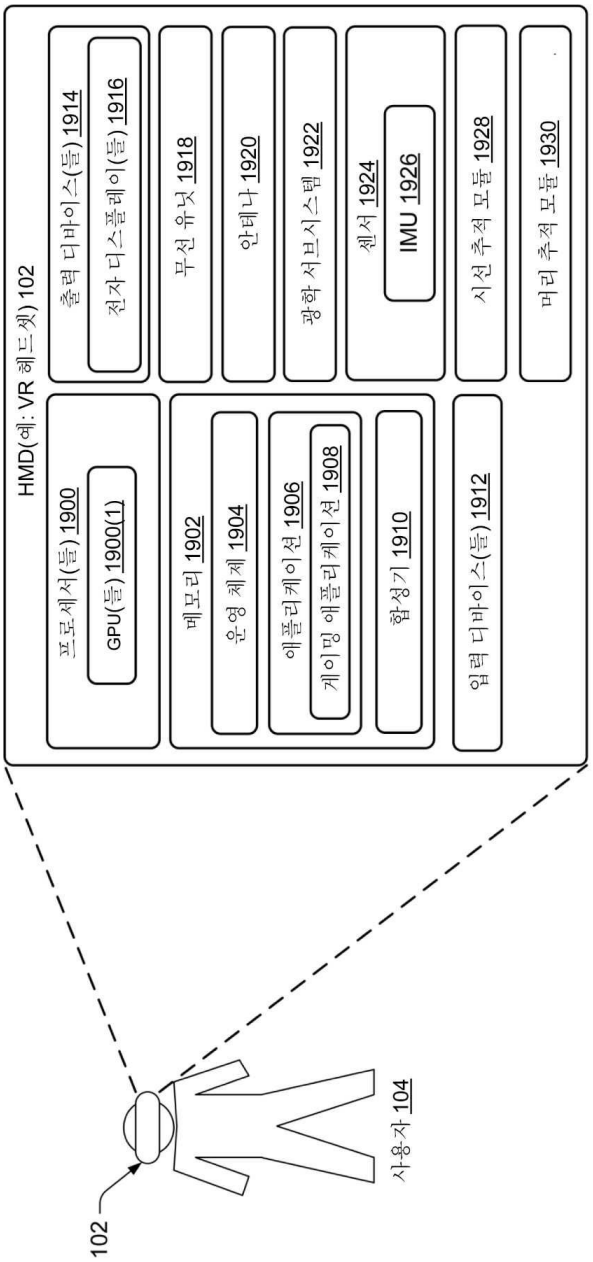
도면18a



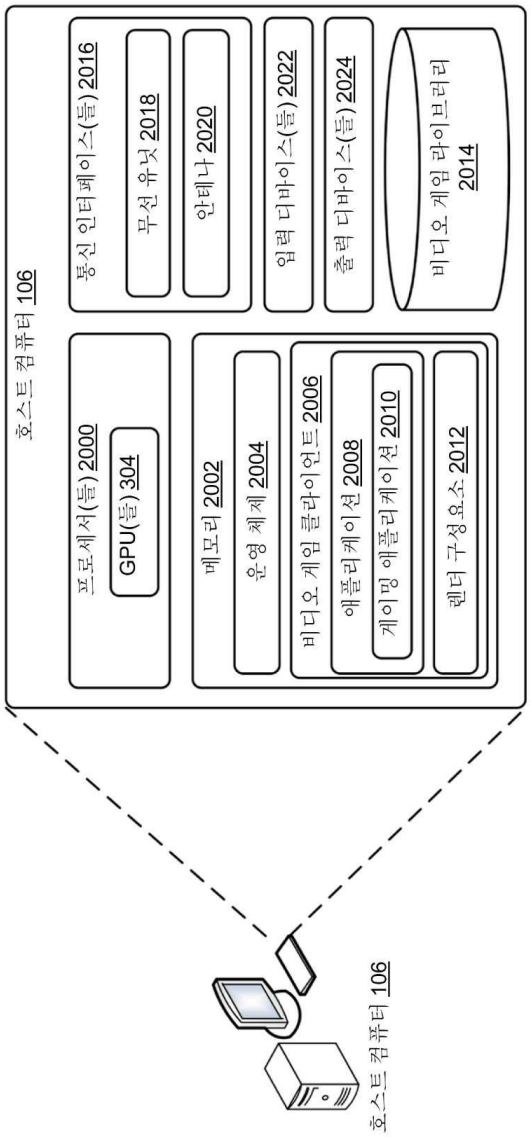
도면18b



도면19


HMD(예: VR 헤드셋) 102
프로세서(들) 1900
GPU(들) 1900(1)
메모리 1902
운영 체제 1904
애플리케이션 1906
게이밍 애플리케이션 1908
합성기 1910
입력 디바이스(들) 1912
출력 디바이스(들) 1914
전자 디스플레이(들) 1916
무선 유닛 1918
안테나 1920
광학 서브시스템 1922
센서 1924
IMU 1926
시선 추적 모듈 1928
머리 추적 모듈 1930
사용자 104
102

도면20


호스트 컴퓨터 106
프로세서(들) 2000
GPU(들) 304
메모리 2002
운영 체제 2004
비디오 게임 클라이언트 2006
애플리케이션 2008
게이밍 애플리케이션 2010
렌더 구성요소 2012
통신 인터페이스(들) 2016
무선 유닛 2018
안테나 2020
입력 디바이스(들) 2022
출력 디바이스(들) 2024
비디오 게임 라이브러리 2014
호스트 컴퓨터 106