(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2025-0034192
(43) 공개일자 2025년03월10일

(51) 국제특허분류(Int. Cl.)
A63F 1/04 (2006.01) *A63F 1/06* (2006.01)
A63F 1/18 (2006.01) *A63F 13/213* (2014.01)
G06V 20/52 (2022.01) *G07F 17/32* (2006.01)
(52) CPC특허분류
A63F 1/04 (2013.01)
A63F 1/067 (2013.01)
(21) 출원번호 10-2025-7006570(분할)
(22) 출원일자(국제) 2017년05월16일
심사청구일자 없음
(62) 원출원 특허 10-2023-7042359
원출원일자(국제) 2017년05월16일
심사청구일자 2023년12월29일
(85) 번역문제출일자 2025년02월27일
(86) 국제출원번호 PCT/AU2017/050452
(87) 국제공개번호 WO 2017/197452
국제공개일자 2017년11월23일
(30) 우선권주장
2016901829 2016년05월16일 오스트레일리아(AU)

(71) 출원인
엔제루 구루푸 가부시키가이샤
일본국 시가켄 히가시오미시 아오노쵸 4600반치
(72) 발명자
보 니얏 딘 민
오스트레일리아 3067 빅토리아 애버츠포드 하퍼 스트리트 9 레벨 1
샬라 서브하시
오스트레일리아 3067 빅토리아 애버츠포드 하퍼 스트리트 9 레벨 1
리 지
오스트레일리아 3067 빅토리아 애버츠포드 하퍼 스트리트 9 레벨 1
(74) 대리인
특허법인코리아나

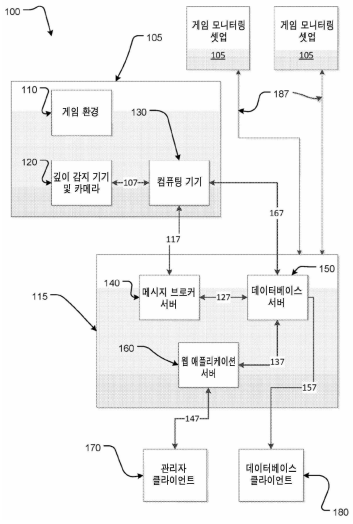
전체 청구항 수 : 총 35 항

(54) 발명의 명칭 **자동화된 테이블 게임 활동 인식을 위한 시스템 및 방법**

(57) 요 약

일부 실시예들은 자동화된 게임 인식을 위한 시스템에 관한 것으로, 상기 시스템은 : 테이블 게임을 포함하는 시야의 이미지 프레임들을 캡처하도록 구성된 적어도 하나의 이미지 센서; 상기 시야의 심도(depth of field) 이미지들을 캡처하도록 구성된 적어도 하나의 깊이 센서; 및 상기 이미지 프레임들 및 상기 심도 이미지들을 수신하도록 구성되고 그리고 상기 수신된 이미지 프레임들 및 상기 심도 이미지들을 처리하여 상기 시야에 나타나는 적어도 하나의 게임 상태의 자동화된 인식을 생성하도록 구성되는 컴퓨팅 기기를 포함한다. 실시예들은 또한 자동화된 게임 인식을 위한 방법들 및 컴퓨터 판독 가능 매체에 관한 것이다. 추가 실시예들은 게임 테이블 상에서의 게임 플레이 그리고/또는 게임 이벤트들을 모니터링하기 위한 방법들 및 시스템들에 관한 것이다.

대 표 도 - 도1



(52) CPC특허분류
A63F 1/18 (2013.01)
A63F 13/213 (2015.01)
G06V 20/52 (2023.08)
G07F 17/322 (2013.01)
G07F 17/3225 (2013.01)
G07F 17/3293 (2013.01)

명 세 서

청구범위

청구항 1

자동화된 게임 인식을 위한 시스템으로서,

상기 시스템은 :

테이블 게임을 포함하는 시야의 이미지 프레임들을 캡처하도록 구성된 적어도 하나의 이미지 센서;

상기 시야의 심도(depth of field) 이미지들을 캡처하도록 구성된 적어도 하나의 깊이 센서; 및

상기 이미지 프레임들 및 상기 심도 이미지들을 수신하도록 구성되고 그리고 상기 수신된 이미지 프레임들 및 상기 심도 이미지들을 처리하여 상기 시야에 나타나는 적어도 하나의 게임 상태의 자동화된 인식을 생성하도록 구성되는 컴퓨팅 기기를 포함하는, 시스템.

청구항 2

청구항 1에 있어서,

상기 이미지 프레임들은 가시 스펙트럼 내에 있거나 가시 스펙트럼을 구성하는 이미지들 또는 적외선 또는 자외선 이미지들을 포함하는, 시스템.

청구항 3

청구항 1 또는 청구항 2에 있어서,

상기 심도 이미지들은 상기 시야에 대한 ToF(time of flight) 데이터 포인트들 및 심도를 반영하는 위상 정보 데이터 포인트들을 포함하는, 시스템.

청구항 4

청구항 1 내지 청구항 3 중 어느 한 항에 있어서,

상기 시야에 나타나는 적어도 하나의 게임 상태는 게임 시작, 칩 검출, 칩 값 추정, 칩 스택 높이 추정 및 게임 종료 중 하나 이상 또는 모두를 포함하는, 시스템.

청구항 5

자동화된 게임 인식 방법으로서,

상기 방법은 :

테이블 게임을 포함하는 시야의 이미지 프레임들을 획득하는 단계;

상기 시야의 심도(depth of field) 이미지들을 획득하는 단계; 및

상기 수신된 이미지 프레임들 및 상기 심도 이미지들을 처리하여 상기 시야에 나타나는 적어도 하나의 게임 상태의 자동화된 인식을 생성하는 단계를 포함하는, 방법.

청구항 6

청구항 5에 있어서,

상기 이미지 프레임들은 가시 스펙트럼 내에 있거나 가시 스펙트럼을 구성하는 이미지들 또는 적외선 또는 자외선 이미지들을 포함하는, 방법.

청구항 7

청구항 5 또는 청구항 6에 있어서,

상기 심도 이미지들은 상기 시야에 대한 ToF(time of flight) 데이터 포인트들 및 심도를 반영하는 위상 정보 데이터 포인터들을 포함하는, 방법.

청구항 8

청구항 5 내지 청구항 7 중 어느 한 항에 있어서,

상기 시야에 나타나는 적어도 하나의 게임 상태는 게임 시작, 칩 검출, 칩 값 추정, 칩 스택 높이 추정 및 게임 종료 중 하나 이상 또는 모두를 포함하는, 방법.

청구항 9

자동화된 게임 인식을 위한 명령들을 포함하는 비-일시적 컴퓨터 판독 가능 매체로서,

상기 명령들은 하나 이상의 프로세서들에 의해 실행될 때 :

테이블 게임을 포함하는 시야의 이미지 프레임들을 획득하는 작업;

상기 시야의 심도(depth of field) 이미지들을 획득하는 작업; 및

상기 수신된 이미지 프레임들 및 상기 심도 이미지들을 처리하여 상기 시야에 나타나는 적어도 하나의 게임 상태의 자동화된 인식을 생성하는 작업의 수행을 유발하는, 비-일시적 컴퓨터 판독 가능 매체.

청구항 10

청구항 9에 있어서,

상기 이미지 프레임들은 가시 스펙트럼 내에 있거나 가시 스펙트럼을 구성하는 이미지들 또는 적외선 또는 자외선 이미지들을 포함하는, 비-일시적 컴퓨터 판독 가능 매체.

청구항 11

청구항 9에 있어서,

상기 심도 이미지들은 상기 시야에 대한 ToF(time of flight) 데이터 포인트들 및 심도를 반영하는 위상 정보 데이터 포인터들을 포함하는, 비-일시적 컴퓨터 판독 가능 매체.

청구항 12

청구항 9에 있어서,

상기 시야에 나타나는 적어도 하나의 게임 상태는 게임 시작, 칩 검출, 칩 값 추정, 칩 스택 높이 추정 및 게임 종료 중 하나 이상 또는 모두를 포함하는, 비-일시적 컴퓨터 판독 가능 매체.

청구항 13

게임 테이블의 테이블 표면상의 게임 플레이를 모니터링하는 방법으로서,

상기 방법은 :

상기 테이블 표면상의 다수의 제1 미리 정의된 관심 영역들 중 임의의 하나에서 게임 물체의 존재를 식별하기 위해 상기 테이블 표면의 캡처된 이미지들을 실시간으로 분석하는 단계;

상기 다수의 제1 미리 정의된 관심 영역들 중 임의의 하나에 존재하는 것으로 식별되는 게임 물체에 응답하여, 게임 이벤트에 대한 타임스탬프를 기록하는 단계; 및

서버에 게임 이벤트 데이터를 전송하는 단계를 포함하며,

상기 게임 이벤트 데이터는 상기 타임스탬프, 상기 게임 이벤트의 표시, 그리고 상기 다수의 제1 미리 정의된 관심 영역들 중 임의의 하나의 식별자를 포함하는, 방법.

청구항 14

청구항 13에 있어서,

상기 게임 물체는 게임 카드인, 방법.

청구항 15

청구항 13에 있어서,

상기 게임 물체는 위치 마커인, 방법.

청구항 16

청구항 13 내지 청구항 15 중 어느 한 항에 있어서,

상기 분석하는 단계는 상기 테이블 표면상의 적어도 하나의 내기 물체(wager object)의 존재를 식별하는 단계를 포함하는, 방법.

청구항 17

청구항 16에 있어서,

상기 적어도 하나의 내기 물체는 상기 게임 물체와 상이한, 방법.

청구항 18

청구항 17에 있어서,

상기 적어도 하나의 내기 물체의 존재는 다수의 제2 미리 정의된 관심 영역들 중 하나 이상에서 식별되는, 방법.

청구항 19

청구항 18에 있어서,

상기 분석하는 단계는 상기 하나 이상의 제2 미리 정의된 관심 영역들에서 내기 물체들의 하나 이상의 그룹을 식별하는 단계를 포함하는, 방법.

청구항 20

청구항 19에 있어서,

상기 분석하는 단계는 :

상기 테이블 표면에 대하여 상기 내기 물체의 하나 이상의 그룹들 각각의 높이를 추정하는 단계; 및

상기 내기 물체들의 각 그룹에 존재하는 내기 물체들의 수를 추정하는 단계를 더 포함하는, 방법.

청구항 21

청구항 20에 있어서,

상기 분석하는 단계는 상기 내기 물체들의 각 그룹의 최상단의 내기 물체의 색상을 식별하는 단계를 더 포함하는, 방법.

청구항 22

청구항 21에 있어서,

적어도 하나의 내기 물체의 존재가 식별되는 각각의 제2 미리 정의된 관심 영역과 연관된 내기 금액을 자동으로 추정하는 단계를 더 포함하고,

상기 추정은 내기 물체들의 각 그룹의 최상단의 내기 물체의 식별된 색상, 그리고 각각의 제2 관심 영역 내의 내기 물체들의 각 그룹의 추정된 내기 물체들의 수에 기초하는, 방법.

청구항 23

청구항 13 내지 청구항 22 중 어느 한 항에 있어서,

상기 캡처된 이미지들은 멀티 스펙트럼 이미지들을 포함하며,

상기 분석하는 단계는 상기 테이블 표면상의 상기 다수의 제1 미리 정의된 관심 영역들 또는 제2 미리 정의된 관심 영역들 중 임의의 하나에서 게임 물체의 존재를 식별하기 위해 상기 멀티 스펙트럼 이미지들에 대해 멀티 프레임 처리를 수행하는 단계를 더 포함하는, 방법.

청구항 24

게임 테이블의 테이블 표면상의 게임 플레이를 모니터링하는 시스템으로서,

상기 시스템은 :

테이블 표면의 이미지들을 캡처하도록 구성되는 적어도 하나의 카메라; 및

상기 카메라와 통신하는 컴퓨팅 기기를 포함하며,

상기 컴퓨팅 기기는 상기 테이블 표면상의 다수의 제1 미리 정의된 관심 영역들 중 임의의 하나에서 게임 물체의 존재를 자동으로 식별하기 위해 상기 테이블 표면의 캡처된 이미지들을 실시간으로 분석하도록 구성되는, 시스템.

청구항 25

청구항 24에 있어서,

상기 게임 물체는 게임 카드인, 시스템.

청구항 26

청구항 25에 있어서,

상기 게임 물체는 위치 마커인, 시스템.

청구항 27

청구항 24 내지 청구항 26 중 어느 한 항에 있어서,

상기 컴퓨팅 기기는 상기 테이블 표면상의 적어도 하나의 내기 물체(wager object)의 존재를 식별하도록 더 구성되는, 시스템.

청구항 28

청구항 27에 있어서,

상기 적어도 하나의 내기 물체는 상기 게임 물체와 상이한, 시스템.

청구항 29

청구항 28에 있어서,

상기 적어도 하나의 내기 물체의 존재는 다수의 제2 미리 정의된 관심 영역들 중 하나 이상에서 식별되는, 시스템.

청구항 30

청구항 29에 있어서,

상기 컴퓨팅 기기는 상기 하나 이상의 제2 미리 정의된 관심 영역들에서 내기 물체들의 하나 이상의 그룹을 식별하도록 더 구성되는, 시스템.

청구항 31

청구항 30에 있어서,

상기 적어도 하나의 카메라는 :

상기 테이블 표면에 대한 상기 게임 물체들의 깊이 데이터를 상기 컴퓨팅 기기에 전달하는 깊이 감지 기기를 더 포함하며,

상기 컴퓨팅 기기는 상기 테이블 표면에 대하여 상기 내기 물체의 하나 이상의 그룹들 각각의 높이를 추정하도록 더 구성되며,

상기 컴퓨팅 기기는 상기 내기 물체들의 각 그룹에 존재하는 내기 물체들의 수를 추정하도록 더 구성되는, 시스템.

청구항 32

청구항 31에 있어서,

상기 컴퓨팅 기기는 상기 내기 물체들의 각 그룹의 최상단의 내기 물체의 색상을 식별하도록 더 구성되는, 시스템.

청구항 33

청구항 32에 있어서,

상기 컴퓨팅 기기는 적어도 하나의 내기 물체의 존재가 식별되는 각각의 제2 미리 정의된 관심 영역과 연관된 내기 금액을 자동으로 추정하도록 더 구성되고,

상기 추정은 내기 물체들의 각 그룹의 최상단의 내기 물체의 식별된 색상, 그리고 각각의 제2 관심 영역 내의 내기 물체들의 각 그룹의 추정된 내기 물체들의 수에 기초하는, 시스템.

청구항 34

청구항 24 내지 청구항 33 중 어느 한 항에 있어서,

상기 캡처된 이미지들은 멀티 스펙트럼 이미지들을 포함하며,

상기 컴퓨팅 기기는 상기 테이블 표면상의 상기 다수의 제1 미리 정의된 관심 영역들 또는 제2 미리 정의된 관심 영역들 중 임의의 하나에서 게임 물체의 존재를 식별하기 위해 상기 멀티 스펙트럼 이미지들에 대해 멀티 프레임 처리를 수행하도록 더 구성되는, 시스템.

청구항 35

게임 테이블상의 게임 이벤트들을 자동으로 모니터링하기 위한 시스템으로서,

상기 시스템은 :

상기 게임 테이블의 게임 영역상의 다수의 게임 물체들의 깊이를 캡처하도록 구성되는 깊이 이미징 기기;

상기 게임 영역의 시각 이미지들을 캡처하도록 구성되는 다수의 시각 이미징 카메라들;

다수의 게임들과 연관된 구성 데이터, 상기 게임 테이블의 구성, 관심 영역들의 위치, 게임 물체들을 인식하기 위한 패턴들, 그리고 상기 게임 테이블상의 게임 물체들의 상태 변화에 따른 게임 이벤트들의 정의를 포함하는 게임 구성 모듈; 및

상기 깊이 이미징 기기 및 상기 다수의 시각 이미징 카메라들로부터 데이터를 수신하고 그리고 상기 게임 영역상의 물체들 그리고 게임 도중 발생하는 게임 이벤트들을 자동으로 인식하기 위해 상기 게임 구성 모듈의 상기 구성 데이터에 액세스하도록 구성되는 컴퓨터 시스템을 포함하는, 시스템.

발명의 설명

기 술 분 야

[0001] 설명되는 실시예들은 일반적으로 테이블 게임을 모니터링하는 것에 관한 것이다. 특히, 실시예들은 게임 장소들에서 테이블 게임들의 이벤트들을 모니터링하기 위한 시스템들 및 방법들에 관한 것이다.

배 경 기 술

[0002] 카지노 및 기타 장소에서는 이제 플레이어들을 모니터링하고 그들의 비즈니스 전략을 수립하기 위해 감시 기술 및 기타 관리 소프트웨어를 사용하고 있다. 그들은 실시간 행동 분석, 알고리즘(또는 프로세스), 그리고 플레이어 추적 기술을 이용하여, 플레이어 수익을 극대화하고 인력을 최적화하며 그리고 장소 수익을 극대화하는 게임 유형에 대한 장소 바닥 공간 할당을 최적화하고자 한다. 대부분의 카지노 고객은 동전, 지폐 또는 티켓 대신 플레이어 카드를 사용하게 하는 로열티 프로그램에 참여한다. 이로 인해 카지노는 개별 도박 행위를 기록 및 분석하고, 플레이어 프로파일을 작성하고, 그리고 도박꾼이 베팅하는 금액, 그들의 승패, 그리고 그들이 슬롯머신 버튼을 누르는 속도 등을 기록할 수 있는 기회를 얻는다. 그러나, 테이블 게임은 슬롯머신이나 버튼식 게임 머신들보다 쉽게 모니터링되지 않는다.

[0003] 테이블 게임을 모니터링하고 관리하는 시스템은 일반적으로 설치 및 유지보수 비용이 많이 드는 것으로 나타났으며, 그리고 진정으로 유용하기 위해 필요한 정확도 수준을 달성하지 못했다. 다른 옵션들로는 카지노 칩 및 기타 오프라인 수율 관리 솔루션들에 센서들을 포함시키는 것이 포함되지만, 이는 비효율적인 것으로 입증되었다. 게임 장소의 운영 환경은 빠른 속도로 진행되며, 시각 및 청각 소음 및 산만함이 많으며, 카드들과 베팅 칩들이 테이블에서 불규칙한 위치들에 있을 수 있으며, 그리고 조명이 상당히 다를 수 있다.

[0004] 카지노 또는 기타 다른 게임 장소는 특정 이벤트의 발생 또는 비-발생에 베팅하는 플레이어들을 포함하는 바카라(Baccarat), 블랙잭(Balckjack), 룰렛(Roulette)과 같은 테이블 기반 게임을 진행한다. 개별 게임에는 게임을 시작하거나 게임 중에 배치된 베팅의 결과를 결정하거나 게임을 종료하는 고유한 세트의 정의된 이벤트들이 있다. 대부분의 게임들은 지정 딜러에 의해 진행되며, 이러한 지정 딜러는 각 게임마다 특정한 행동들을 취해 게임을 시작하거나, 배치된 베팅 결과를 결정하는 이벤트들을 트리거하거나 또는 게임을 종료할 수 있다.

[0005] 카지노 및 기타 게임 장소들은 게임 테이블이나 경기 표면(playing surface)에서 발생하는 이벤트들과 관련된 거래 데이터를 확인하는 데 관심이 있다. 이 정보는 카지노의 비즈니스 전략을 계획하고 플레이어들의 행동을 모니터링하는데 도움이 될 수 있다. 테이블 게임의 이벤트 및 결과에 관한 정보는 카지노가 최적의 인력 배치, 특정 게임에 대한 바닥 공간 할당, 그리고 기타 수익 향상 또는 고객 경험 향상 결정을 확인하는 기반이 될 수 있다. 카지노에서 사용되는 게임 테이블에서 발생하는 이벤트들과 관련된 거래 데이터를 확인하는 한 가지 방법은 테이블들의 서브세트에서 발생하는 이벤트들을 시각적으로 검사하고 관찰된 정보를 보고하는 개인들에 의한 무작위 샘플링이다. 보고된 정보는 카지노 테이블의 전반적인 활동 수준을 추정하기 위해 추론될 수 있다. 그러나, 이러한 육안 검사는 한 시간 또는 그 이상의 간격으로 발생하며, 인간의 판단에 의존하기 때문에, 그러한 방법을 이용하는 것은 비효율적일 수 있다.

[0006] 테이블 게임들을 모니터링하고 관리하는 시스템들은 일반적으로 설치 및 유지보수 비용이 많이 드는 것으로 나타났으며, 그리고 진정으로 유용하기 위해 필요한 정확도 수준을 달성하지 못했다. 다른 옵션들로는 카지노 칩 및 기타 오프라인 수율 관리 솔루션들에 센서들을 포함시키는 것이 포함되지만, 이는 비효율적인 것으로 입증되었다. 게임 장소의 운영 환경은 빠른 속도로 진행되며, 시각 및 청각 소음 및 산만함이 많으며, 카드들과 베팅 칩들이 테이블에서 불규칙한 위치들에 있을 수 있으며, 그리고 조명이 상당히 다를 수 있다.

[0007] 게임 장소들에서 테이블 게임들의 이벤트들을 모니터링하기 위한 종래 기술과 관련된 하나 이상의 단점들 또는 약점들을 해소하거나 개선하는 것 또는 적어도 유용한 대안을 제공하는 것이 바람직하다.

[0008] 본 명세서 전체에 걸쳐, "포함하다(comprise)"라는 단어, 또는 "포함하다(comprises)" 또는 "포함하는(comprising)"과 같은 변형들은 명시된 요소, 정수 또는 단계, 또는 요소들, 정수들 또는 단계들의 그룹의 포함하지만, 다른 요소, 정수 또는 단계, 또는 요소들, 정수들 또는 단계들의 그룹을 배제하지 않음을 의미하는 것으로 이해될 것이다.

[0009] 본 명세서에서, 요소가 옵션들의 리스트 중"적어도 하나"일 수 있다는 설명은 요소가 리스트된 옵션들 중 하나일 수 있거나, 또는 리스트된 옵션들 중 두 개 이상의 조합일 수 있다는 것이 이해되어야 한다.

[0010] 본 명세서에 포함된 문서들, 동작들, 재료들, 기기들, 물품들 등에 대한 모든 논의는 이러한 사안들 중 일부 또는 전부가 선행 기술 자료의 일부를 구성하거나 본 출원의 각 청구항의 우선 날짜 이전에 존재했던 본 개시서와 관련된 분야에서 일반적인 지식으로 인정되는 것으로 받아들여서는 안된다.

발명의 내용

[0011] 제1 양상에 따르면, 일부 실시예들은 자동화된 게임 인식을 위한 시스템을 제공하며, 상기 시스템은 : 테이블 게임을 포함하는 시야의 이미지 프레임들을 캡처하도록 구성된 적어도 하나의 이지 센서; 상기 시야의 심도

(depth of field) 이미지들을 캡처하도록 구성된 적어도 하나의 깊이 센서; 및 상기 이미지 프레임들 및 상기 심도 이미지들을 수신하도록 구성되고 그리고 상기 수신된 이미지 프레임들 및 상기 심도 이미지들을 처리하여 상기 시야에 나타나는 적어도 하나의 게임 상태의 자동화된 인식을 생성하도록 구성되는 컴퓨팅 기기를 포함한다.

[0012] 제2 양상에 따르면, 일부 실시예들은 자동화된 게임 인식 방법을 제공하며, 상기 방법은 : 테이블 게임을 포함하는 시야의 이미지 프레임들을 획득하는 단계; 상기 시야의 심도(depth of field) 이미지들을 획득하는 단계; 및 상기 수신된 이미지 프레임들 및 상기 심도 이미지들을 처리하여 상기 시야에 나타나는 적어도 하나의 게임 상태의 자동화된 인식을 생성하는 단계를 포함한다.

[0013] 추가 양상에 따르면, 일부 실시예들은 자동화된 게임 인식을 위한 비-일시적 컴퓨터 판독 가능 매체를 제공하며, 상기 비-일시적 컴퓨터 판독 가능 매체는 하나 이상의 프로세서들에 의해 실행될 때 다음의 수행을 유발하는 명령들을 포함한다 : 테이블 게임을 포함하는 시야의 이미지 프레임들을 획득하는 작업; 상기 시야의 심도(depth of field) 이미지들을 획득하는 작업; 및 상기 수신된 이미지 프레임들 및 상기 심도 이미지들을 처리하여 상기 시야에 나타나는 적어도 하나의 게임 상태의 자동화된 인식을 생성하는 작업.

[0014] 상기 이미지 프레임들은 가시 스펙트럼 내에 있거나 가시 스펙트럼을 구성하는 이미지들 또는 적외선 또는 자외선 이미지들을 포함할 수 있다. 상기 심도 이미지들은 상기 시야에 대한 ToF(time of flight) 데이터 포인트들 그리고/또는 심도를 반영하는 위상 정보 데이터 포인터들을 포함할 수 있다. 상기 시야에 나타나는 적어도 하나의 게임 상태는 게임 시작, 칩 검출, 칩 값 추정, 칩 스택 높이 추정 및 게임 종료 중 하나 이상 또는 모두를 포함할 수 있다. 게임 시작 및/또는 게임 종료는 카드 검출 또는 돌리 검출에 의해 수행될 수 있다. 테이블 게임은 포커, 블랙잭 또는 바카라와 같은 카드 게임이거나 또는 룰렛과 같이 카드 기반이 아닌 게임일 수 있다.

[0015] 일부 실시예들은 게임 테이블의 테이블 표면상의 게임 플레이를 모니터링하는 방법에 관한 것으로, 상기 방법은 : 상기 테이블 표면상의 다수의 제1 미리 정의된 관심 영역들 중 임의의 하나에서 게임 물체의 존재를 식별하기 위해 상기 테이블 표면의 캡처된 이미지들을 실시간으로 분석하는 단계; 상기 다수의 제1 미리 정의된 관심 영역들 중 임의의 하나에 존재하는 것으로 식별되는 게임 물체에 응답하여, 게임 이벤트에 대한 타임스탬프를 기록하는 단계; 및 서버에 게임 이벤트 데이터를 전송하는 단계를 포함하며, 상기 게임 이벤트 데이터는 상기 타임스탬프, 상기 게임 이벤트의 표시, 그리고 상기 다수의 제1 미리 정의된 관심 영역들 중 임의의 하나의 식별자를 포함한다.

[0016] 상기 게임 물체는 게임 카드 또는 위치 마커일 수 있다. 상기 분석하는 단계는 상기 테이블 표면상의 적어도 하나의 내기 물체(wager object)의 존재를 식별하는 단계를 포함할 수 있다. 상기 적어도 하나의 내기 물체는 상기 게임 물체와 상이할 수 있다. 상기 적어도 하나의 내기 물체의 존재는 다수의 제2 미리 정의된 관심 영역들 중 하나 이상에서 식별될 수 있다. 상기 분석하는 단계는 상기 하나 이상의 제2 미리 정의된 관심 영역들에서 내기 물체들의 하나 이상의 그룹을 식별하는 단계를 포함할 수 있다.

[0017] 상기 분석하는 단계는 : 상기 테이블 표면에 대하여 상기 내기 물체의 하나 이상의 그룹들 각각의 높이를 추정하는 단계; 및 상기 내기 물체들의 각 그룹에 존재하는 내기 물체들의 수를 추정하는 단계를 더 포함할 수 있다. 상기 분석하는 단계는 상기 내기 물체들의 각 그룹의 최상단의 내기 물체의 색상을 식별하는 단계를 더 포함할 수 있다.

[0018] 상기 방법은 적어도 하나의 내기 물체의 존재가 식별되는 각각의 제2 미리 정의된 관심 영역과 연관된 내기 금액을 자동으로 추정하는 단계를 더 포함할 수 있으며, 이 경우 상기 추정은 내기 물체들의 각 그룹의 최상단의 내기 물체의 식별된 색상, 그리고 각각의 제2 관심 영역 내의 내기 물체들의 각 그룹의 추정된 내기 물체 수에 기초한다.

[0019] 상기 캡처된 이미지들은 멀티 스펙트럼 이미지들을 포함할 수 있으며, 상기 분석하는 단계는 상기 테이블 표면상의 상기 다수의 제1 미리 정의된 관심 영역들 또는 제2 미리 정의된 관심 영역들 중 임의의 하나에서 게임 물체의 존재를 식별하기 위해 상기 멀티 스펙트럼 이미지들에 대해 멀티 프레임 처리를 수행하는 것을 포함할 수 있다.

[0020] 일부 실시예들은 게임 테이블의 테이블 표면상의 게임 플레이를 모니터링하는 시스템에 관한 것으로, 상기 시스템은 : 테이블 표면의 이미지들을 캡처하도록 구성되는 적어도 하나의 카메라; 및 상기 카메라와 통신하는 컴퓨팅 기기를 포함하며, 상기 컴퓨팅 기기는 상기 테이블 표면상의 다수의 제1 미리 정의된 관심 영역들 중 임의의 하나에서 게임 물체의 존재를 자동으로 식별하기 위해 상기 테이블 표면의 캡처된 이미지들을 실시간으로 분석

하도록 구성된다.

[0021] 상기 게임 물체는 예를 들어 게임 카드 또는 위치 마커일 수 있다. 상기 컴퓨팅 기기는 상기 테이블 표면상의 적어도 하나의 내기 물체(wager object)의 존재를 식별하도록 구성될 수 있다. 상기 적어도 하나의 내기 물체는 상기 게임 물체와 상이할 수 있다. 상기 적어도 하나의 내기 물체의 존재는 다수의 제2 미리 정의된 관심 영역들 중 하나 이상에서 상기 컴퓨팅 기기에 의해 식별된다. 상기 컴퓨팅 기기는 상기 하나 이상의 제2 미리 정의된 관심 영역들에서 내기 물체들의 하나 이상의 그룹을 식별하도록 구성될 수 있다.

[0022] 상기 적어도 하나의 카메라는 상기 테이블 표면에 대한 상기 게임 물체들의 깊이 데이터를 상기 컴퓨팅 기기에 전달하는 깊이 감지 기기를 더 포함할 수 있다. 상기 컴퓨팅 기기는 상기 테이블 표면에 대하여 상기 내기 물체의 하나 이상의 그룹들 각각의 높이를 추정하도록 더 구성될 수 있다. 상기 컴퓨팅 기기는 상기 내기 물체들의 각 그룹에 존재하는 내기 물체들의 수를 추정하도록 더 구성될 수 있다. 상기 컴퓨팅 기기는 상기 내기 물체들의 각 그룹의 최상단의 내기 물체의 색상을 식별하도록 더 구성될 수 있다.

[0023] 상기 컴퓨팅 기기는 적어도 하나의 내기 물체의 존재가 식별되는 각각의 제2 미리 정의된 관심 영역과 연관된 내기 금액을 자동으로 추정하도록 구성될 수 있으며, 이 경우 상기 추정은 내기 물체들의 각 그룹의 최상단의 내기 물체의 식별된 색상, 그리고 각각의 제2 관심 영역 내의 내기 물체들의 각 그룹의 내기 물체들의 추정된 수에 기초한다.

[0024] 상기 캡처된 이미지들은 멀티 스펙트럼 이미지들을 포함할 수 있으며, 상기 컴퓨팅 기기는 상기 테이블 표면상의 상기 다수의 제1 미리 정의된 관심 영역들 또는 제2 미리 정의된 관심 영역들 중 임의의 하나에서 게임 물체의 존재를 식별하기 위해 상기 멀티 스펙트럼 이미지들에 대해 멀티 프레임 처리를 수행하도록 구성될 수 있다.

[0025] 일부 실시예들은 게임 테이블상의 게임 이벤트들을 자동으로 모니터링하기 위한 시스템에 관한 것으로, 상기 시스템은 : 상기 게임 테이블의 게임 영역상의 다수의 게임 물체들의 깊이를 캡처하도록 구성되는 깊이 이미징 기기; 상기 게임 영역의 시각 이미지들을 캡처하도록 구성되는 다수의 시각 이미징 카메라들; 다수의 게임들과 연관된 구성 데이터, 상기 게임 테이블의 구성, 관심 영역들의 위치, 게임 물체들을 인식하기 위한 패턴들, 그리고 상기 게임 테이블상의 게임 물체들의 상태 변화에 따른 게임 이벤트들의 정의를 포함하는 게임 구성 모듈; 및 상기 깊이 이미징 기기 및 상기 다수의 시각 이미징 카메라들로부터 데이터를 수신하고 그리고 상기 게임 영역상의 물체들 그리고 게임 도중 발생하는 게임 이벤트들을 자동으로 인식하기 위해 상기 게임 구성 모듈의 상기 구성 데이터에 액세스하도록 구성되는 컴퓨터 시스템을 포함한다.

[0026] 본원에 설명된 방법들은 완전히 자동화될 수 있으며, 이에 따라 게임 활동 모니터링은 사람의 판단이나 개입 없이 발생할 수 있다. 그러나, 일부 인간의 상호 작용은 베팅에 대한 관심 영역들을 설정하고, 그리고 카드들 또는 돌리(dolly)들과 같은 게임 물체들의 위치를 찾는 것과 같이 시스템 구성 단계들에서 발생할 수 있다.

도면의 간단한 설명

[0027] 도 1은 게임 모니터링 시스템의 블록도이다.

도 2는 도 1의 게임 모니터링 시스템의 일부를 형성하는, 자동화된 테이블 게임 인식을 위한 시스템의 개략도이다.

도 3은 도 1의 시스템의 게임 환경의 일부를 형성할 수 있는 게임 테이블의 표면 이미지이다.

도 4는 도 1의 시스템의 게임 환경의 일부를 형성할 수 있는 다른 게임 테이블의 표면 이미지이다.

도 5는 도 1의 시스템의 게임 환경의 일부를 형성할 수 있는 또 다른 게임 테이블의 표면 이미지이다.

도 6은 도 1의 시스템의 게임 환경의 일부를 형성할 수 있는 또 다른 게임 테이블의 표면 이미지이다.

도 7은 도 1의 시스템에서 사용하기 위한 깊이 감지 기기 및 카메라의 블록도이다.

도 8은 일부 실시예들에 따른 깊이 감지 기기 및 카메라의 하우징 내부의 정면도이다.

도 9는 도 1의 시스템의 컴퓨팅 기기의 블록도이다.

도 10은 도 1의 시스템의 메시지 브로커 서버(Message Broker Server)의 블록도이다.

도 11은 도 1의 시스템의 데이터베이스 서버의 블록도이다.

도 12는 도 1의 시스템의 웹 애플리케이션 서버의 블록도이다.

도 13은 도 1의 시스템의 게임 환경의 일부를 형성할 수 있는 다른 게임 테이블의 구성을 관리하기 위한 인터페이스를 도시하는 웹 애플리케이션의 스크린샷이다.

도 14는 도 1의 시스템의 게임 환경의 일부를 형성할 수 있는 다른 게임 테이블의 구성을 관리하기 위한 인터페이스를 도시하는 웹 애플리케이션의 다른 스크린샷이다.

도 15는 도 1의 시스템의 게임 환경의 일부를 형성할 수 있는 다른 게임 테이블의 구성을 관리하기 위한 인터페이스를 도시하는 웹 애플리케이션의 다른 스크린샷이다.

도 16은 샘플 이미지에 대한 여러 유형의 스레스홀딩 연산(thresholding operation)들의 결과들을 보여주는 이미지 세트이다.

도 17은 샘플 이미지에 대한 추가 유형의 스레스홀딩 연산(thresholding operation)들의 결과들을 보여주는 이미지 세트이다.

도 18은 샘플 이미지에 대한 침식(erosion) 및 팽창(dilation) 작업들의 결과를 나타내는 이미지 세트이다.

도 19는 에지 검출 프로세스의 흐름도이다.

도 20은 도 19의 프로세스의 일부에서의 에지 검출을 설명하기 위한 다이어그램이다.

도 21은 도 19의 에지 검출 프로세스에 적용된 예시적 기준을 설명하기 위한 예시적인 그래프이다.

도 22는 상이한 파라미터들을 갖는 샘플 이미지에 대한 윤곽 검출 프로세스의 적용을 설명하기 위한 이미지 세트이다.

도 23a는 평면 추정 프로세스가 적용되는 포인트들의 세트의 플롯이다.

도 23b는 평면 추정 프로세스의 적용 결과 그리고 도 23a의 플롯에 도시된 포인트들까지의 상기 추정된 평면의 직교 거리의 플롯이다.

도 24는 일부 실시예들에 따른 게임 모니터링 시스템의 흐름도이다.

도 25는 추가 실시예들에 따른 게임 모니터링 시스템의 흐름도이다.

도 26a 및 도 26b는 다른 게임 테이블에서의 일부 카드 검출 프로세스들의 적용을 나타내는 이미지 프레임들이다.

도 27a는 카드 및 칩 검출 프로세스들이 적용될 수 있는 다른 게임 테이블의 이미지 프레임이다.

도 27b는 도 27a의 이미지 프레임에 스레스홀딩 기술(thresholding technique)을 적용함으로써 획득되는 도 27a의 게임 테이블의 이미지 프레임이다.

도 28a 및 도 28b는 적외선 카메라에 의해 획득된 이미지 프레임들이며, 다른 게임 테이블에서의 일부 카드 및 칩 검출 프로세스들의 적용을 도시한다.

도 29a는 칩 검출 프로세스에 대한 입력일 수 있는 이미지 프레임이다.

도 29b는 도 29a의 이미지 프레임 상에 이진 스레스홀딩 연산(binary thresholding operation)을 적용함으로써 획득된 이미지 프레임이다.

도 29c는 도 29b의 이미지 프레임에 침식 작업을 적용함으로써 획득된 이미지 프레임이다.

도 29d는 도 29c의 이미지 프레임에 확장 작업을 적용함으로써 획득된 이미지 프레임이다.

도 29e는 도 29a의 입력 이미지 프레임 상에 칩 검출 프로세스를 적용한 결과를 나타내는 이미지 프레임이다.

도 30은 게임 테이블의 뷰의 일부분이 방해받는 경우 게임 테이블에 칩 검출 프로세스를 적용한 결과를 나타내는 이미지 프레임이다.

도 31은 도 30a의 장애물이 이미지 프레임에 없는 경우에 획득된 도 30a의 게임 테이블에 칩 검출 프로세스를 적용한 결과를 나타내는 이미지 프레임이다.

도 32는 칩 검출 프로세스를 게임 테이블에 적용한 결과를 나타내는 이미지 프레임을 도시한 것으로, 입력 이미

지 프레임은 시각적 이미지 카메라에 의해 캡처된 이미지에 기초한다.

도 33은 도 32의 게임 테이블에 칩 검출 프로세스를 적용한 결과를 나타내는 이미지 프레임을 도시하며, 입력 이미지 프레임은 적외선 카메라에 의해 캡처된 이미지에 기초한다.

도 34는 일부 실시예에 따른 다중 프레임 처리 기술의 흐름도이다.

발명을 실시하기 위한 구체적인 내용

[0028] 설명된 실시예들은 일반적으로 테이블 게임들을 모니터링하는 것에 관한 것이다. 특히, 실시예들은 게임 장소들에서 테이블 게임들의 이벤트들을 모니터링하기 위한 시스템들 및 방법들에 관한 것이다.

[0029] **게임 모니터링 시스템** : 도 1은 일부 실시예들에 따른 게임 모니터링 시스템(100)의 블록도이다. 시스템(100)은 다수의 게임 모니터링 셋업들(105), 게임 모니터링 인프라 구조(115), 관리자 클라이언트(Administrator Client)(170), 그리고 데이터베이스 클라이언트(180)를 포함할 수 있다. 게임 모니터링 셋업(105)은 게임 환경(110), 깊이 감지 기기 및 카메라(120) 및 컴퓨팅 기기(130)를 포함한다. 시스템(100)은 카지노와 같은 게임 장소의 하나 이상의 게임 룸들에서의 설치 및 작업에 적합하다. 게임 룸들은 각각 그곳에 배치된 하나 또는 다수의 게임 테이블들을 가지며, 이러한 테이블들 중 일부 또는 각각은 개별 게임 모니터링 셋업(105)의 일부를 형성할 수 있다. 예를 들어 Microsoft™ Kinect 또는 Asus™ Xtion 또는 Infineon™ 3D Image Sensor REAL3™ 같은 상업적으로 사용 가능한 기기들 또는 카메라 기능들을 가진 다른 유사한 깊이 감지 기기들은 깊이 감지 기기 및 카메라로서 사용될 수 있다. 깊이 감지 기기 및 카메라(120)는 컴퓨팅 기기(130)와 결합되거나 컴퓨팅 기기(130)에 연결되어, 컴퓨팅 기기(130)로부터 명령들을 수신하고, 그리고 링크(107)를 사용하여 기록된 데이터를 컴퓨팅 기기(130)에 전송한다. 예를 들어, Microsoft™ Kinect는 컴퓨팅 기기 상의 USB 포트를 사용하여 컴퓨팅 기기에 연결될 수 있다.

[0030] 게임 장소는 여러 게임 환경(예를 들어, 테이블 게임들이 플레이되는 영역 또는 방)을 가질 수 있으며, 그리고 이러한 게임 환경들 각각을 모니터링하기 위해, 게임 모니터링 셋업(105)이 여러개 있을 수 있다. 다수의 게임 모니터링 셋업들(105)은 네트워크 링크(187)를 사용하여 공통의 게임 모니터링 인프라 구조(115)와 결합되거나 링크될 수 있다. 네트워크 링크(187)는 컴퓨팅 기기(130)와 메시지 브로커 서버(140) 간의 네트워크 링크(117); 및 데이터베이스 서버(150)와 컴퓨팅 기기(130) 간의 네트워크 링크(167)를 포함한다. 게임 모니터링 인프라 구조(115)는 또한 둘 이상의 다른 게임 장소들에서의 게임 모니터링 셋업들(105)과 결합되거나 링크될 수 있다. 게임 장소가 많은 수의 게임 환경들(110)을 가질 수 있는 일부 실시예들에서, 게임 모니터링 인프라 구조(115) 중 다수는 동일한 장소에서 게임 모니터링 셋업들(105)의 서로 다른 서브세트들과 결합될 수 있다.

[0031] 게임 모니터링 인프라 구조(115)는 메시지 브로커 서버(140), 데이터베이스 서버(150) 및 웹 애플리케이션 서버(160)를 포함한다. 메시지 브로커 서버(140)는 양방향 네트워크 링크(117)를 통해 다수의 컴퓨팅 기기들(130)에 연결될 수 있다. 네트워크 링크(127)는 데이터 또는 명령의 전달을 가능하게 하기 위해 메시지 브로커 서버(140)와 데이터베이스 서버(150) 사이에 존재할 수 있다. 네트워크 링크(137)는 데이터 또는 명령의 전달을 가능하게 하기 위해 웹 애플리케이션 서버(160)와 데이터베이스 서버(150) 사이에 존재할 수 있다. 서버들(140, 150 및 160) 각각은 독립형 서버들로서 구현될 수 있거나, 또는 하나 이상의 물리적 서버들 상의 별개의 가상 서버들로서 구현될 수 있거나, 또는 클라우드 컴퓨팅 서비스로 구현될 수 있다. 서버들(140, 150 및 160) 각각은 또한 증가된 성능 또는 높은 가용성 요건들(high availability requirements)을 처리하도록 구성된 둘 이상의 서버들의 네트워크를 통해 구현될 수도 있다.

[0032] 관리자 클라이언트(170)는 예를 들어 컴퓨터 또는 태블릿과 같은 최종 사용자 컴퓨팅 기기일 수 있고, 그리고 네트워크 링크(147)를 통해 웹 애플리케이션 서버(160)에 접속될 수 있다. 데이터베이스 클라이언트(180)는 최종 사용자 컴퓨팅 기기일 수 있으며, 또는 다른 최종 사용자 컴퓨팅 기기들 또는 다른 데이터베이스들에 데이터를 중계하기 위한 인터페이스일 수 있으며, 그리고 네트워크 링크(157)를 통해 데이터베이스 서버(150)에 연결될 수 있다.

[0033] **게임 환경** : 게임 환경(110)의 구성은 수행되는 특정 게임에 따라 다를 수 있지만, 실시예들 중 어느 하나에 의해 모니터링되는 대부분의 게임들은 공통 요소들을 갖는다. 도 2는 일부 실시예들에 따른 자동화된 테이블 게임 인식(200)을 위한 시스템을 도시한다. 본 발명의 현재 설명되는 실시예의 시스템의 주요 기능들은 게임이 시작하고 종료할 때를 검출하고, 배치된 칩들의 위치를 검출하고, 그리고 칩 스택의 값 및 높이(얼마나 많은 칩들이 있는지)를 추정하는 것이다. 이러한 시스템은 이미지 처리 기술과 감지 기기 특징들의 조합을 기반으로 한다.

[0034] 게임 환경(110)은 게임이 진행되는 경기 표면 또는 게임 테이블(210)을 포함한다. 경기 표면(210)은 일반적으로 실질적으로 수평인 평면이고, 그리고 게임 모니터링 시스템(100)에 의해 검출될 수 있는 카드들(211) 또는 칩들(213) 또는 다른 물체들과 같은 다양한 게임 물체들을 그 위에 배치할 수 있다. 깊이 감지 기기 및 카메라(120)는 깊이 감지 기기 및 카메라(120)를 깊이 감지 기기의 시야 내의 임의의 방해물 위에 위치시키는 높이에서 기둥 또는 포스트(220) 상에 배치될 수 있으며, 그리고 깊이 감지 기기 및 카메라(120)의 시야를 게임 테이블(210)을 향하여 약간 아래쪽으로 향하게 하도록 기울어질 수 있다. 방해물들은 예를 들어 테이블에서 게임을 진행하는 딜러, 또는 게임 참가자, 또는 통행인과 같은 일시적인 방해물들일 수 있다. 일부 실시예들에서, 깊이 감지 기기 또는 카메라(120)는 게임 테이블(210)의 경기 표면의 방해받지 않는 뷰를 얻기 위해 딜러의 어깨의 뒤쪽 및 위에 배치될 수 있다. 깊이 감지 기기 또는 카메라(120) 그리고 컴퓨팅 기기(130)의 위치는 게임 테이블(210)에 위치한 기둥 또는 포스트 상의 다른 디스플레이 스크린들의 위에 있거나 또는 그곳에 인접할 수 있다.

[0035] 도 3은 블랙잭 게임을 위해 구성된 게임 테이블의 경기 표면의 일부를 나타내는 이미지(300)이다. 경기 표면 또는 게임 테이블은 다수의 미리 정의된 관심 영역들을 포함하며, 그리고 게임의 특성에 따라, 게임의 작업에 관한 특정 배향 및 기능을 가질 수 있다. 미리 정의된 관심 영역은 게임 카드 또는 내기 물체들(wager objects) 같은 특정 게임 물체들의 검출을 위해 지정될 수 있다. 예를 들어, 도 3에서, 제1 미리 정의된 관심 영역들(305)은 플레이어에게 할당된 카드들(211)을 위치시키도록 지정되며, 그리고 제2 미리 정의된 관심 영역들(308)은 플레이어가 게임에서 베팅할 수 칩들 또는 내기 물체들(213)을 위치시키도록 기정된다. 일부 실시예들에서, 하나 이상의 미리 정의된 관심 영역은 하나 이상의 다른 미리 정의된 관심 영역과 중첩될 수 있다. 일부 실시예들에서, 하나의 미리 정의된 관심 영역은 다른 미리 정의된 관심 영역의 일부를 형성할 수 있다.

[0036] 게임의 참가자들은 베팅을 할 수 있는 플레이어들과 게임을 진행하는 딜러들을 포함한다. 베팅을 하거나 게임을 진행하기 위해, 게임 물체들로 설명되는 물체들은 플레이어들 또는 딜러들에 의해 사용된다. 게임 물체들은 그들을 식별하기 위한 특정 마킹들을 갖는 특정 모양의 카드들(211)을 포함할 수 있으며, 칩들 또는 내기 물체들(213) 또는 다른 이러한 물체들은 플레이어가 게임에서 베팅할 수 있는 금액을 지정할 수 있거나, 또는 위치 마커 또는 룰렛 게임에서 사용되는 돌리(dolly)와 같은 게임의 결과를 지정할 수 있는 별개의 모양을 갖는 다른 물체들을 포함할 수 있다. 게임은 게임 시작, 게임 중 플레이어들의 베팅 배치, 게임 중 중간 결과, 그리고 게임의 최종 결과를 결정하는 게임 종료를 포함하는 일련의 게임 이벤트들을 통해 진행된다. 게임 중에, 플레이어는 베팅 배치를 위해 지정된 관심 영역에 자신의 베팅 물체들(즉, 베팅 토큰들 또는 칩들)을 배치하여 베팅을 할 수 있다. 예를 들어, 도 3에 도시된 블랙잭 게임에서, 플레이어는, 베팅을 위해 자신의 지정된 영역(308)에, 칩들(213) 같은 하나 이상의 내기 물체들을 배치시킴으로써 게임 중에 베팅을 할 수 있다. 칩들 또는 내기 물체들은 관심 영역 내에서 그룹들 또는 스택들로 배열될 수 있다. 종종, 내기 물체들의 그룹 또는 스택은 공통 색상의 내기 물체들을 포함할 것이다.

[0037] 특정 게임들에서, 플레이어는 성공 가능성과 연관된 관심 영역, 그리고 베팅과 연관된 지불금(payoff)을 선택할 수 있다. 예를 들어, 도 6의 경기 표면(210)은 룰렛 게임에 대한 베팅 영역을 마킹하는 경기 표면이다. 경기 표면(600) 상의 서로 다른 관심 영역들(308)은 플레이어의 베팅에 대한 서로 다른 성공 전망 및 플레이어의 베팅에 대한 서로 다른 지불금을 가질 수 있다. 경기 표면들은, 게임과 관련된 다양한 규칙들 및/또는 베팅 약속들(conventions)에 따라, 다양한 관심 영역들의 위치 및 구조와 관련하여 여러 가지 상이한 구성들을 가질 수 있다. 도 4는 바카라 게임을 위해 설계된 게임 테이블의 이미지(400)이다. 도 5는 일부 실시예들에 따라 다른 게임을 위해 설계된 게임 테이블의 이미지(500)이다.

[0038] **깊이 감지 기기 및 카메라** : 깊이 감지 기기 및 카메라(120)는 기기 앞의 시야의 시각 이미지들 및 깊이 정보를 캡처하는 기능들을 수행한다. 기기(120)는 게임 테이블 상에 식별된 모든 지정된 관심 영역들을 캡처하기 위해 경기 표면 또는 게임 테이블 앞에 놓인다. 기기(120)는 적외선 프로젝터(710), 적외선 센서(720), 카메라(730), 프로세서(740), 통신 포트(750), 그리고 적외선 센서(720) 및 카메라(730)를 프로세서(740)와 연결하는 내부 데이터 링크(705)를 포함한다. 내부 데이터 링크(706)는 프로세서(740)를 통신 포트(750)에 연결한다. 깊이 감지 기기 및 카메라(120)는 사람이 볼 수 있는 그리고/또는 사람이 볼 수 없는 광의 여러 스펙트럼으로부터 이미지들을 캡처할 수 있다. 예를 들어, 깊이 감지 기기 및 카메라(120)는 카메라(730)를 통해 가시광선 스펙트럼으로부터 그리고 적외선 센서(720)를 통해 적외선 스펙트럼으로부터 이미지를 캡처할 수 있으며, 그리고 결과적으로 멀티-스펙트럼 카메라로서 동작할 수 있다.

[0039] 깊이 감지 기기 및 카메라(120)는 비행 시간(Time of Flight) 기술에 의존하여 자신 앞의 시야의 깊이 또는 장

면의 깊이를 감지할 수 있다. 적외선 프로젝터(710)는 펄스 또는 변조된 광을 사인파 또는 구형파일 수 있는 연속파로 투사할 수 있다. 깊이 정보의 정확도를 높이기 위해 여러 위상의 투영 광이 투영되고 감지될 수 있다. 적외선 프로젝터(710)에 의해 방출된 광 펄스와 적외선 센서(720)에 의해 감지된 반사 펄스 간의 측정된 위상 시프트는 기기 앞의 시야의 깊이를 계산하기 위해 프로세서에 의존된다. 일부 실시예들에서, 깊이 감지 기기 및 카메라(120)는 투사된 광 패턴을 사용하는 원리에 기초하는 구조화된 광 3D 스캐너, 그리고 깊이 감지를 위한 카메라를 포함할 수 있다. 일부 실시예들에서, 깊이 감지 기기 및 카메라(120)는 2 개 이상의 렌즈들 및 대응 이미지 센서들로부터의 이미지들을 사용하여 깊이 감지 기능을 수행하는 스테레오 카메라를 포함할 수 있다. 깊이 감지 기기 및 카메라(120)는 특히 게임 환경의 그리고 테이블 경기 표면(210)의 깊이 정보를 획득하기 위해 깊이 또는 범위 감지의 다른 대안적인 수단들 또는 둘 이상의 기술들의 조합에 의존할 수 있다.

[0040] 감지된 깊이 정보는 프로세서에 의해 결정된 픽셀 그리드 정보와 결합되어 통신 포트(750)에 출력으로서 제공된다. 또한, 픽셀 그리드 정보는 카메라(730)에 의해 캡처된 시각 이미지들과 결합되고 깊이 정보와 결합되어 포트(750)를 통해 출력으로서 제공된다. 깊이 정보와 별도로, 적외선 센서(720)는 시야에 의해 반사된 적외선의 강도를 감지하고, 이 정보는 프로세서(740)에 의해 픽셀 그리드 정보와 결합되어 포트(750)로 전달된다. 포트(750)는 예를 들어 USB 포트와 같은 물리적 포트 또는 무선 네트워크 어댑터와 같은 무선 송신기 형태일 수 있다. 깊이 감지 기기 및 카메라(120)는 서로 맵핑될 수 있는 서로 다른 좌표 공간의 감지된 깊이, 색상, 그리고 적외선 이미지 데이터를 반환하여, 기기의 시야 내의 특정 영역 또는 포인트와 관련된 통합된 깊이, 색상 그리고 적외선 데이터를 얻는다. 도 8은 일 실시예에 따른 깊이 감지 기기 및 카메라(800)의 하우징 내부의 정면도이며, 그리고 적외선 프로젝터(710), 적외선 센서(720) 및 카메라(730)를 포함하는 그 컴포넌트들의 일부를 도시한다.

[0041] **컴퓨팅 기기** : 깊이 감지 기기 및 카메라(120)에 의해 생성된 데이터는 통신 포트(990)를 통해 컴퓨팅 기기(130)에 의해 수신된다. 포트(990)는 센서 데이터를 수신하거나 깊이 감지 기기 및 카메라(120)에 명령들을 전송하기 위해 통신 포트(750)와 연결되는 USB 포트 또는 무선 어댑터의 형태일 수 있다. 컴퓨팅 기기(130)의 하드웨어 컴포넌트들(910)은 메모리(914), 프로세서(912), 그리고 컴퓨팅 기기의 동작에 필요한 다른 컴포넌트들을 포함한다. 메모리(914)는 필요한 소프트웨어 모듈(920)을 저장하며, 상기 소프트웨어 모듈(920)은 : 이미지 처리 라이브러리(922); 깊이 감지 기기 및 카메라 API(924); 런타임 환경 드라이버(926); 게임 모니터링 모듈(928); 배치 스크립트(930), 스케줄링된 작업(932); 및 메시지 생성자 모듈(934)을 포함한다.

[0042] 이미지 처리 라이브러리(922)는 게임 모니터링 모듈(928)에 의해 착수된 이미지 처리 단계들에 필요한 이미지들 및 다른 프로그램들에 대한 스레스홀딩 연산들(thresholding operations), 모폴로지 연산을 수행하는 것과 같이 기본적인 이미지 처리 작업들을 수행하기 위한 프로그램들의 세트이다. OpenCV는 사용될 수 있는 이미지 처리 라이브러리의 일례이다. 깊이 감지 기기 및 카메라 API(924)는 컴퓨팅 기기(930)가 하나 이상의 깊이 감지 기기 및 카메라(920)와의 통신 채널을 설정하게 할 수 있는 프로그램들의 세트이다. 예를 들어, Microsoft™ Kinect™ 기기가 깊이 감지 기기 및 카메라로서 사용된다면, Windows™ SDK용 Kinect는 깊이 감지 기기 및 카메라 API(924)로 사용될 것이다. 이 API(924)는 컴퓨터 기기(130)가 적절한 프로토콜로 깊이 감지 기기 및 카메라(120)에 질의를 하고 반환된 결과의 형식을 이해할 수 있게 한다. 이 API(924)는 깊이 감지 기기 및 카메라(120)에 의해 생성된 데이터가 게임 모니터링 모듈(928)에 의해 수신 및 처리될 수 있게 한다. 또한 컴퓨팅 기기(130)는 게임 모니터링 모듈(928)의 실행에 필요한 의존성을 제공하기 위해 필요한 런타임 환경 드라이버들(926)을 갖는다. 게임 모니터링 모듈(928)은 게임 진행 중에 발생하는 게임 이벤트들을 모니터링하는 프로그램들을 포함한다.

[0043] 소프트웨어 모듈들(920)은 또한 게임 모니터링 모듈(928)에 필요한 하우스키핑(housekeeping) 및 유지보수 작업들을 수행하기 위해 윈도우 파워 셸 스크립트들 또는 다른 스크립트 언어의 스크립트 형태일 수 있는 배치 스크립트들을 포함한다. 배치 스크립트들(930)은 윈도우 스케줄러 작업들 또는 다른 유사 작업 스케줄링 서비스들의 형태일 수 있는 스케줄링된 작업들(932)을 통해 스케줄에 기초하여 실행될 수 있다. 게임 모니터링 모듈(928)로부터의 명령들에 기초한 메시지 생성자 모듈(934)은 메시지 프로커 서버(140)에 전달되는 메시지들을 생성한다. 메시지 생성자 모듈은 예를 들어 RabbitMQ 또는 Kafka와 같은 표준 메시징 시스템을 기반으로 할 수 있다. 구성 모듈(940)에 저장된 메시지 브로커 구성(942)에 기초하여, 메시지 생성자 모듈(934)은 통신 포트(990) 및 네트워크 링크(117)를 통해 메시지 브로커 서버(140)에 메시지들을 전달할 수 있다. 또한, 구성 모듈(940)은 테이블 구성(942) 및 게임 시작 및 종료 트리거 구성(944)을 포함한다. 구성 모듈(940)의 컴포넌트들은 메모리(914)에 하나 이상의 구성 파일들의 형태로 저장된다. 구성 파일들은 예를 들어 XML 형식으로 저장될 수 있다.

[0044] **메시지 브로커 서버** : 메시지 브로커 서버(140)는 메시지 브로커 서비스를 구현하고, 그리고 네트워크 링크

(117)를 통해 다수의 컴퓨팅 기기들(130)로부터의 메시지들을 청취한다. 메시지 브로커 서버(140)는 공통 로컬 네트워크 내의 컴퓨팅 기기(130)와 동일한 구내(premise)에 배치될 수 있거나, 또는 옥외에(원격으로) 배치될 수 있지만 메시지 및 데이터의 전달을 가능하게 하기 위해 2 개의 구내들 사이에 설정된 네트워크 링크(117)를 통해 여전히 통신상태에 있을 수 있다. 메시지 브로커 서버(140)는 중앙 집중화되고 다수의 게임 장소들 내의 컴퓨팅 기기들(130)에 연결되어, 중앙 집중화된 메시지 브로커 서비스를 제공할 수 있다. 메시지 브로커 서버(140)는 메모리(1014), 프로세서(1012) 및 서버의 동작에 필요한 다른 하드웨어 컴포넌트들을 포함하는 하드웨어 컴포넌트들(1010)을 갖는다. 메시지 큐 모듈(1020)은 다수의 구성 기기들(130)로부터의 메시지들을 수신, 해석 및 처리하기 위한 큐를 구현한다. 메시지들은 메시지 브로커 서버와의 데이터 및 명령의 양방향 전송을 가능하게 할 수 있는 네트워크 어댑터 형태 또는 다른 유사 포트들의 형태일 수 있는 통신 포트(1090)를 통해 수신된다. 메시지 큐 모듈(1020)은 RabiitMQ 또는 Kafka와 같은 메시지 브로커 패키지를 통해 구현될 수 있다. 메시지 큐 모듈(1020)은, 게임 테이블에서 발생하는 게임 이벤트들에 관한 트랜잭션 정보를 포함하는 메시지를 수신하면, 데이터베이스 파싱 모듈(1030)을 개시한다. 데이터베이스 파싱 모듈(1030)은 메시지 큐 모듈(1020)에 의해 수신된 메시지를 네트워크 링크(127)를 통해 데이터베이스 서버(150) 상에서 이후에 실행되는 데이터베이스 쿼리로 파싱한다.

[0045] **데이터베이스 서버** : 데이터베이스 서버(150)는 메시지 브로커 서버(140)로부터 게임 이벤트 데이터를 수신하고, 웹 애플리케이션 서버(160)를 통해 관리되는 테이블 구성 데이터를 저장하고, 그리고 게임 모니터링 시스템(100)에 의해 캡처된 게임 이벤트 데이터에 대한 액세스를 제공하기 위해 데이터베이스 클라이언트(180)에 대한 저장소로서 기능하는 목적으로 사용된다. 데이터베이스 서버(150)는 메모리(1114), 프로세서(1112) 및 서버의 동작에 필요한 다른 하드웨어 컴포넌트를 포함하는 하드웨어 컴포넌트들(1110)을 갖는다. 통신 포트(1190)는 하나 이상의 네트워크 링크들을 통해 데이터베이스 서버(150)와의 데이터 및 명령의 양방향 전송을 가능하게 할 수 있는 네트워크 어댑터 또는 다른 유사한 포트들의 형태일 수 있다. 데이터베이스 모듈(1120)은 MySQL™, Postgres 또는 Microsoft™ SQL 서버와 같은 데이터베이스 관리 시스템을 통해 구현될 수 있다.

[0046] 데이터베이스 모듈(1120)은 테이블 구성 데이터(1122) 및 게임 이벤트 데이터(1124)를 포함하는 데이터를 보유한다. 게임 이벤트 데이터(1124)는 게임 테이블 또는 경기 표면에서 발생하는 게임 이벤트들을 나타내는 트랜잭션 데이터를 포함한다. 게임 이벤트 데이터를 형성하는 기록들은 게임 이벤트가 인식된 시간에 대한 타임스탬프; 상기 게임 이벤트가 발생한 게임 테이블에 대한 고유 식별자; 베팅 배치, 게임의 중간 결과, 게임의 최종 결과와 같은 게임 이벤트들의 성질에 대한 식별자; 게임 이벤트와 연관된 관심 영역의 식별자; 관심 영역과 관련된 베팅 값의 추정치; 및 게임 이벤트를 나타내는 다른 관련 속성들을 포함할 수 있다.

[0047] 테이블 구성 데이터(1122)는 : 게임 테이블들 및 관련 컴퓨팅 기기(130)에 대한 고유 식별자들; 다각형 형태의 게임 테이블 상의 관심 영역들(308)의 위치 및 상기 다각형들의 종점들을 형성하는 깊이 감지 기기 및 카메라(120)와 연관된 픽셀들의 좌표들; 관심 영역(308)의 특성, 즉 그 영역이 카드들을 두기 위한 영역인지, 또는 칩들을 놓기 위한 영역인지, 또는 검출될 특정 게임 물체를 놓기 위한 영역인지 여부; 게임 시작 및 종료 트리거링 이벤트들의 특성, 즉 게임의 시작이 관심 영역에 카드를 놓음으로써 검출되는지 또는 특정 관심 영역에 특정 게임 물체의 배치에 의해 검출되는지 여부; 예를 들어 게임 모니터링 모듈(928)에 의한 검출을 가능하게 하기 위해 카드들 또는 칩들과 같은 게임 물체들에 대한 모델 윤곽들; 그리고 게임 모니터링 시스템(100)에 의해 의존되는 파라미터들을 나타내기 위해 필요한 다른 관련 데이터를 포함한다. 일부 실시예들에서, 테이블 구성 데이터(1122) 및 게임 이벤트 데이터(1124)는 별도의 데이터베이스 서버들에 보관되어 게임 모니터링 시스템(100)의 더 큰 확장성과 관리성을 가능하게 할 수 있다.

[0048] 데이터베이스 서버(150)는 또한 테이블 구성 데이터(1122)를 개별 컴퓨팅 기기(130)에 전파하는 기능을 수행하는 테이블 구성 전파 모듈(1140)을 포함한다. 테이블 구성 전파 모듈은 먼저 예를 들어 XML 파일과 같은 구성 파일의 형태로 테이블 구성(942), 게임 시작 및 종료 트리거 구성(944) 및 메시지 브로커 구성(946)을 생성하는 데이터베이스 스크립트 및 명령 라인 스크립트의 조합을 통해 구현될 수 있다. 생성된 구성 파일들은 네트워크 링크(167)에 응답하는 통신 포트(1190)를 통해 개별 컴퓨팅 기기(130)에 전달될 수 있다. 네트워크 링크(167)는 데이터베이스 서버(150) 및 컴퓨팅 기기(130)가 동일한 로컬 네트워크에 있는 경우 로컬 네트워크일 수 있으며, 또는 데이터베이스 서버(150) 및 컴퓨팅 기기(130)가 별개의 네트워크들에 위치하는 경우 다수의 컴퓨터 네트워크들에 걸쳐있는 네트워크 링크일 수 있다. 구성 파일들의 전송은 예를 들어 파일 전송 프로토콜 또는 SSH 파일 전송 프로토콜과 같은 적절한 네트워크 프로토콜을 통해 이루어질 수 있다.

[0049] **웹 애플리케이션 서버** : 웹 애플리케이션 서버(160)는 데이터베이스 서버(150) 상의 테이블 구성 데이터(1122)의 구성 및 관리를 용이하게 하는 웹 애플리케이션을 호스팅한다. 웹 애플리케이션 서버(160)는 메모리(1214),

프로세서(1212) 및 서버의 동작에 필요한 다른 필요한 하드웨어 컴포넌트들을 포함하는 하드웨어 컴포넌트들(1210)을 갖는다. 통신 포트(1290)는 하나 이상의 네트워크 링크들을 통해 웹 애플리케이션 서버(160)와의 데이터 및 명령의 양방향 전송을 가능하게 할 수 있는 네트워크 어댑터 또는 다른 유사한 포트들의 형태일 수 있다. 웹 애플리케이션 서버는 사용자가 데이터베이스 서버(150) 상의 테이블 구성 데이터(1122)를 생성 및 업데이트할 수 있게 하는 웹 인터페이스들을 포함하는 웹 애플리케이션 모듈(1220)을 포함한다. 웹 애플리케이션은 예를 들어 python의 Django 또는 ASP.NET 또는 다른 유사한 웹 프레임워크들과 같은 웹 애플리케이션 프레임워크를 통해 구현될 수 있다. 또한, 웹 애플리케이션 서버(160)는 웹 인터페이스를 통해 웹 애플리케이션 모듈(1220)에 의해 수신된 명령들을, 관리자 클라이언트(170)에 의해 착수된 동작들을 반영하는 테이블 구성 데이터(1122)를 생성하거나 업데이트할 특정 데이터베이스 질의들 또는 명령들로 변환하는 데이터베이스 파싱 모듈(1230)을 포함한다. 데이터베이스 질의들 또는 명령들은 네트워크 링크(137)를 통해 데이터베이스 서버(150) 상에서 실행된다. 네트워크 링크(137)는 데이터베이스 서버(150) 및 웹 애플리케이션 서버(150)가 공통 네트워크에 있는 경우 로컬 영역 네트워크 링크일 수 있으며, 또는 데이터베이스 서버(150) 및 웹 애플리케이션 서버(150)가 별개의 네트워크들에 위치하는 경우 다수의 컴퓨터 네트워크들에 걸쳐있을 수 있다.

[0050] **웹 인터페이스** : 도 13은 게임 환경(110)의 일부를 형성할 수 있는 게임 테이블의 일 실시예의 구성을 관리하기 위한 인터페이스를 도시하는 웹 애플리케이션의 스크린샷(1300)이다. 게임 테이블을 설정하는데 필요할 수 있는 파라미터들로서, 테이블에 대한 고유 식별자를 포함할 수 있는 파라미터들, 그리고 관련 컴퓨팅 기기의 IP 어드레스는 예를 들어 스크린 영역(1310)에 위치된다. 테이블 구성(942)을 코드화하기 위해 컴퓨팅 기기(130)에 전파될 수 있는 XML 구성 파일의 일부가 스크린 영역(1320)에 도시된다. 버튼(1330)은 추가의 게임 테이블에 대한 기록들을 생성하는데 사용될 수 있고, 제출 버튼(1340)은 사용자가 새로운 구성을 제출할 수 있게 한다.

[0051] 도 14는 게임 환경(110)의 일부를 형성할 수 있는 게임 테이블의 일 실시예의 구성을 관리하기 위한 다른 인터페이스를 나타내는 웹 애플리케이션의 다른 스크린샷(1400)이다. 배치 버튼(1410)은 네트워크 링크(167)를 통해 컴퓨팅 기기(130)에 저장된 구성들의 세트를 배치하기 위해 클릭될 수 있다. 삭제 버튼(1424)은 임의의 저장된 구성들을 삭제하기 위해 클릭될 수 있다. 1420은 컴퓨팅 기기(130)에 구성 정보를 저장하고 전파하는데 사용될 수 있는 또 다른 XML 파일의 일부의 샘플이다. 스크린 영역들(1412, 1414 및 1416)은 깊이 감지 기기 및 카메라(130)로부터의 깊이, 색상 및 적외선 이미지 스트림들을 나타낸다. 개별 스트림들과 관련된 구성들의 세부사항들은 버튼(1422)을 클릭함으로써 볼 수 있다. 구성 세부사항들은 버튼(1418)을 클릭함으로써 삭제될 수 있다. 스크린 영역(1440)은 사용자가 저장 버튼(1430)을 클릭함으로써 저장될 수 있는 모든 테이블들에 대한 디폴트 구성을 설정할 수 있게 한다.

[0052] 버튼(1430)은 배치 전에 게임 테이블 구성들에 대한 변경들을 저장하는데 사용될 수 있다.

[0053] 도 15는 게임 환경(110)의 일부를 형성할 수 있는 게임 테이블의 일 실시예의 구성을 관리하기 위한 다른 인터페이스를 나타내는 웹 애플리케이션의 다른 스크린 샷(1500)이다. 스크린샷(1500)에 도시된 인터페이스는 관심 영역의 유형들, 위치들 및 경계들이 사용자 인터페이스 툴을 사용하여 정의될 수 있게 한다. 이와 같이 정의된 관심 영역들은 일단 그것들이 게임 구성 데이터에 저장되면 본원에 언급된 바와 같이 "미리 정의된 관심 영역"이 된다. 깊이 감지 기기 및 카메라(120)에 대한 게임 테이블의 위치가 변경되면 버튼(1510)이 클릭되어 게임 테이블의 리프레시된 이미지를 획득할 수 있다. 스크린샷(1500)에 도시된 이미지 프레임(1515)은 하나 이상의 경계가 있는 관심 영역들(1520)을 포함하며, 각각의 경계가 있는 관심 영역은 다각형(1525)에 의해 정의된다. 커스텀 다각형들(custom polygons)은 선택 가능한 핸들들(1560)을 사용하여 드로잉될 수 있으며, 그리고 예를 들어 다각형들의 리스트(1530)에 추가될 수 있다. 저장 버튼(1540)은 사용자가 다각형에 대한 변경을 저장할 수 있게 하며, 재-매핑버튼(1550)은 사용자가 기존의 다각형들을 다른 위치들로 재매핑할 수 있게 한다.

[0054] 게임의 자동 인식을 위한 시스템을 개발하려면, 게임의 동작을 이해해야한다. 우리는 두 종류의 게임 테이블을 다룬다. 하나는 카드 기반 게임이나 카드 게임으로, 게임이 플레이되는 기본 기기로서 플레이 카드들을 사용하는 모든 게임이며, 전통적이거나 게임 특정적이다. 이 유형의 예들로는 블랙잭, 바카라 및 포커가 있다. 다른 유형의 테이블 게임은 카드를 기반으로 하지 않는다(예를 들어, 룰렛). 이 게임에서, 플레이어들은 하나의 숫자 또는 숫자의 크기, 빨간색 또는 검은색 같은 색상들, 또는 숫자가 홀수인지 짝수인지에 베팅할 수 있다. 우승한 수와 색을 결정하기 위해, 딜러는 휠을 한 방향으로 돌린 다음, 휠 주변을 돌고 있는 기울어진 원형 트랙 주위에서 반대 방향으로 공을 회전시킨다.

[0055] 카드 기반 게임의 동작은 다음과 같다.

[0056] - 플레이어들은 얼마동안 베팅을 할 수 있다.

[0057] - 딜러는 '더 이상 베팅을 하지 않습니다'고 말하고 플레이어들에게 카드들을 제공하기 시작한다. 이것은 게임이 시작될 때이다.

[0058] - 게임 결과가 확정된 후, 딜러는 패배한 선수의 칩들을 모으고 이긴 칩을 나눠준다.

[0059] - 그 다음, 딜러는 모든 카드들을 없앤다. 이거은 게임이 끝나는 때이다.

[0060] 비-카드 기반 게임들(특히, 룰렛)의 동작은 다음과 같다.

[0061] - 플레이어들은 얼마동안 베팅을 해야 한다.

[0062] - 딜러는 휠을 돌리고 공이 멈추기 전까지 잠시 기다린 다음, '더 이상 베팅을 하지 않습니다'라고 알린다. 이것은 게임이 시작될 때이다.

[0063] - 공이 멈추면, 딜러는 테이블 위 위닝 넘버(winning number) 위에 돌리(dolly)를 둔다. 승리(winning)/패배(losing) 칩들은 딜러에 의해 할당/수집된다.

[0064] - 그 후, 게임이 종료된다.

[0065] **전체 모니터링 프로세스** : 게임 모니터링 시스템(100)은 그것의 동작에 있어서 두 가지 기본적인 측면들을 보강한다 : 윤곽 검출; 및 평면 추정. 윤곽 검출은 깊이 감지 기기 및 카메라(120)에 의해 캡처된 이미지들 내 형상들을 인식하기 위해 실시간으로 또는 거의 실시간으로 수행될 수 있는 한 세트의 프로세스들 또는 기술들을 포함한다. 거의 실시간에 가까운 처리는 이벤트 발생 후 수 초, 예를 들어 2-3 초 이하의 레이턴시로 처리하는 것을 포함할 수 있다. 평면 추정은 깊이 감지 기기 및 카메라(120)에 의해 캡처된 깊이 데이터 및 이미지들에 기초하여 게임 테이블 또는 경기 표면을 나타내는 평면의 위치를 추정하기 위해 실시간으로 또는 거의 실시간으로 수행될 수 있는 한 세트의 프로세스들 또는 기술들을 포함한다. 평면 추정의 단계가 수행되면, 획득된 평면 위치 정보는 깊이 감지 기기 및 카메라(120)에 의해 캡처된 추가 깊이 데이터와 결합되어, 게임 테이블 상의 게임 물체들의 스택의 높이를 추정할 수 있고 그리고 예를 들어 칩들의 스택과 같은 게임 물체들의 스택과 관련된 값에 대한 추론을 할 수 있다.

[0066] **이미지 사전-처리 단계들** : 윤곽 검출 또는 평면 추정 기술을 적용하기 전에, 깊이 감지 기기 및 카메라(120)에 의해 캡처된 이미지들에 다수의 이미지 전-처리 단계들이 적용된다. 이러한 이미지 전-처리 단계들은 윤곽 검출 및 평면 추정 기술들을 구현하는 프로세스들의 성능과 정확성을 향상시킨다.

[0067] **스레스홀딩(Thresholding)** : 이용될 수 있는 하나의 이미지 전-처리 기법은 스레스홀딩이다. 깊이 감지 기기 및 카메라(120)는 게임 환경(110)의 컬러 및 적외선 이미지 프레임들의 데이터를 반환한다. 스레스홀딩 기술들은 색상 및 적외선 데이터 스트림 모두에 적용될 수 있다. 색상 또는 적외선 스트림의 특정 프레임의 각 픽셀은 픽셀의 색상이나 강도를 나타내는 숫자값으로 표시된다. 색상 이미지 프레임은 스레스홀딩 연산을 수행하기 전에 그레이스케일 이미지 프레임으로 변환될 수 있다. 글로벌 스레스홀딩(global thresholding)은 스레스홀딩을 구현하는 한 가지 방법이다. 각 픽셀 값이 임의의 임계값과 비교되는 글로벌 스레스홀딩에서, 픽셀 값이 상기 임계값보다 크면, 흰색에 해당하는 값(예를 들어, 8 비트 스케일에서 255)을 할당받고, 그렇지 않다면 검정색에 해당하는 값(예를 들어 8 비트 스케일에서 0)을 할당받는다. 일련의 이미지들(1600)을 통해, 도 16은 샘플 이미지(1601)에 대한 스레스홀딩의 결과의 예를 도시한다. 이미지(1602)는 값 127을 사용하는 이미지(1601)에 대한 글로벌 스레스홀딩의 적용의 결과이며, 이미지(1601)는 8 비트 포맷으로 표현된다. 글로벌 스레스홀딩은 다양한 실제 응용들에 충분하지 않을 수 있다. 이미지는 이미지의 여러 부분들에서 서로 다른 조명 상태들을 가질 수 있으며, 그리고 글로벌 스레스홀딩 기술을 적용하면 약한 조명 상태들을 가진 이미지의 부분들을 줄일 수 있다.

[0068] 적응형 스레스홀딩(adaptive thresholding)은 글로벌 스레스홀딩의 한계들을 해결할 수 있는 대안이다. 적응형 스레스홀딩에서는, 이미지의 서로 다른 작은 영역들에 대한 임계값들이 발견되어 스레스홀딩을 위해 상기 영역들에 적용된다. 적응형 스레스홀딩에 대한 임계값들은 하나의 픽셀의 이웃에 있는 픽셀들의 평균값, 또는 가중치들이 가우시안 분포로부터 취해질 수 있는 이웃 픽셀들의 가중치 합을 취함으로써 계산될 수 있다. 도 16에서, 이미지(1603)는 적응형 평균 스레스홀딩 기술의 적용 결과의 예이며, 이미지(1604)는 원본 이미지(1601)에 적응형 가우시안 스레스홀딩 기술의 적용 결과의 예이다. 스레스홀딩의 또 다른 대안 방법은 Otsu의 2원화(binarization)이다. 이 방법에서, 스레스홀딩은 이미지 히스토그램에 기초하여 수행된다. 일련의 이미지들(1700)을 통해, 도 17은 대표적인 히스토그램(1702)을 갖는 샘플 이미지 세트(1701)에 대한 Otsu의 이진화 기술의 적용 예를 도시한다. 대안적인 스레스홀딩 기술들 중 하나 이상은 전처리 단계에서 그리고 색상 또는 적외선 이미지 프레임들에 적용될 수 있다. 이미지 처리 라이브러리(922)는 이미지 전-처리 단계에서 게임 모니터링 모

듈(928)에 의해 호출될 수 있는 제안된 스레스홀딩 기술들을 구현하는 재사용 가능한 라이브러리들을 제공할 수 있다.

[0069] **모폴로지 변환(morphological transformation)** : 모폴로지 변환들은 이미지 형상을 기반으로 하는 작업들이다. 일반적으로 이는 이진 이미지들에 대해 수행된다. 이는 두 개의 입력들을 필요로 한다. 하나는 우리의 원래 이미지며, 두 번째는 작업의 특성을 결정하는 구조화 요소(structuring element) 또는 커널이라고 한다. 두 가지 기본 모폴로지 연산자들은 침식(Erosion)과 팽창(Dilation)이다.

[0070] 이미지들에서 검출될 피처들을 향상시키고 윤곽 검출 프로세서들의 성능 및 정확성을 향상시키기 위해, 깊이 감지 기기 및 카메라(120)에 의해 캡처된 이미지들에 대해 모폴로지 변환들이 수행된다. 침식 및 팽창은 이미지 전처리 단계에서 적용될 수 있는 모폴로지 변환들의 예들이다. 침식 및 팽창 프로세스들은 모두 2 개의 입력들, 깊이 감지 기기 및 카메라(120)에 의해 캡처된 매트릭스 형태의 이미지 데이터 및 입력 이미지에 대해 수행된 모폴로지 연산의 특성을 결정하는 구조화 요소 또는 커널을 필요로 한다. 커널은 정사각형 또는 원 모양일 수 있으며 그리고 정의된 중심을 가지며 그리고 입력 이미지를 통해 다각측량(traversing)함으로써 연산으로서 적용된다.

[0071] **침식** : 침식의 모폴로지 변환은 이미지를 통해 다각측량하는 커널을 사용하여 이미지 내 전경 물체들(foreground objects)을 샤프닝하는 것을 포함하며, 픽셀 값은 1의 값 또는 커널에 해당하는 모든 값들이 1일 때에만 흰색에 해당하는 값 또는 흰색에 해당하는 값으로 유지된다. 크기가 3 x 3 또는 5 x 5 이거나 다른 크기들의 커널들은 침식 작업을 위해 사용될 수 있다. 침식 작업은 전경 물체들의 경계를 침식한다. 일련의 이미지들(1800)을 통해, 도 18은 침식 및 팽창 연산자들의 적용예를 도시한다. 입력 이미지(1801)에 대한 침식 작업의 효과의 예는 침식 출력 이미지(1802)에서 볼 수 있다. 침식의 작업은 이미지 처리 라이브러리(922)에서 미리 정의된 라이브러리에 의해 수행될 수 있다. 예를 들어, OpenCV 라이브러리가 사용되는 경우, "침식" 기능은 깊이 감지 기기 및 카메라(120)에 의해 캡처되는 이미지 상에서 동작하도록 게임 모니터링 모듈(928)에 의해 호출될 수 있다.

[0072] 침식을 달성하기 위해, 커널은 (2D 컨볼루션에서처럼) 이미지를 통해 슬라이드한다. 원래 이미지 내 픽셀(1 또는 0)은 커널 아래의 모든 픽셀들이 1인 경우에만 1로 간주되고, 그렇지 않으면 침식된다(0으로 설정된다).

[0073] **팽창** : 팽창 작업은 침식의 반대이다. 예를 들어, 3 x 3 정방 행렬 커널을 사용하는 팽창 연산시, 커널 중앙의 픽셀은 1의 값, 또는 대응 커널의 값들 중 임의의 하나가 1일 때 흰색에 해당하는 값, 또는 흰색에 해당하는 값으로 남겨질 수 있다. 입력 이미지(1802)에 대한 이러한 동작의 효과의 예는 침식 출력 이미지(1803)에서 볼 수 있다. 팽창의 결과로, 이미지 내 피처들은 보다 연속적이고 커진다. 팽창 동작은 이미지 처리 라이브러리(922)에서 미리 정의된 라이브러리에 의해 수행될 수 있다. 예를 들어, OpenCV 라이브러리가 사용되면, "확장" 기능은 깊이 감지 기기 및 카메라(120)에 의해 캡처되는 이미지 상에서 동작하도록 게임 모니터링 모듈(928)에 의해 호출될 수 있다.

[0074] 팽창은 침식의 정반대이다. 여기서, 커널 아래의 적어도 하나의 픽셀이 1이면 픽셀 요소는 1이다. 따라서 이미지 내 흰색 영역이 증가하거나 전경 물체의 크기가 커진다. 일반적으로 소음 제거와 같은 경우에는 침식 후에 팽창이 발생한다. 왜냐하면 침식은 화이트 노이즈를 제거하지만 또한 우리의 물체를 축소시키기 때문이다. 그래서 우리는 그것을 팽창시킨다. 노이즈 요소들은 침식에 의해 제거되기 때문에 팽창에 의해 다시 도입되지 않지만, 물체 영역은 증가한다. 또한 이는 물체의 분할된 부분들을 결합하는데 유용하다.

[0075] 스레스홀딩 기술을 이미지에 적용하면 이진 이미지가 생성된다. 이미지에 존재하는 피처들을 더욱 강화하기 위해, 침식과 팽창의 모폴로지 변환들이 적용된다. 바람직하게는, 모폴로지 변환들은 이미지들로부터의 노이즈 감소, 개별 요소들의 분리 및 이미지 내의 서로 다른 요소들의 결합을 보조한다.

[0076] 이미지 윤곽은 이미지에 표현된 물체의 경계를 따라 모든 연속점들을 연결하는 곡선을 포함한다. 윤곽들은 형상 분석 및 물체 검출 및 인식에 유용한 도구이다. 윤곽 근사법(contour approximation)은 특정 모양과 응용에서 원하는 모양의 유사성을 근사화하는데 사용된다. 원하는 형상은 예를 들어 다각형 또는 원 또는 타원의 형태일 수 있다. 정확도와 성능을 향상시키기 위해, 에지 검출 작업이 수행된 후에 이진 이미지들에서 윤곽 감지 작업들이 수행될 수 있다.

[0077] **에지 검출** : 에지 검출은 이미지 내의 물체들의 경계들을 찾는 이미지 처리 기술이다. 이는 밝기의 불연속성을 감지하여 작동한다. 그 중, Canny는 다음과 같은 단계들로 설명될 수 있는 널리 알려진 다단계 에지 검출 알고리즘(또는 프로세스)이다.

[0078] 에지 검출은 인접한 픽셀들 및 픽셀 클러스터들 간의 밝기 불연속점들을 검출함으로써 수행될 수 있다. 이 작업을 수행하기 위해 여러 이미지 처리 기술들이 사용될 수 있다. 일부 실시예들은 깊이 감지 기기 및 카메라(120)에 의해 캡처된 이미지들에서 에지들을 검출하기 위해 Canny 에지 검출 오퍼레이터 또는 프로세스를 구현한다. 도 19는 Canny Edge 검출 오퍼레이터의 구현에 관련된 일련의 단계들을 나타내는 흐름도(1900)이다. 단계(1910)는 오퍼레이터에 대한 입력으로서 사용하기 위한 이미지의 준비를 포함한다. 이 단계는 적절한 스레스홀딩 기법을 이미지에 적용하고, 에지 검출 프로세스의 나머지의 성능을 향상시키기 위해 침식 또는 팽창을 적용하는 것을 포함할 수 있다. 이 단계(1920)는 이미지로부터 원하지 않는 노이즈를 감소시키는 단계를 포함한다. 이것은 예를 들어 5 x 5 가우시안 필터링 커널의 적용으로 달성될 수 있다. 이 단계는 이미지의 피처들을 스무스하게하고 프로세스의 나머지의 성능을 향상시킨다. 사용될 수 있는 가우시안 필터링 커널의 일례는 다음과 같다 :

[0079]
$$\begin{bmatrix} 2 & 4 & 5 & 4 & 2 \\ 4 & 9 & 12 & 9 & 4 \\ 5 & 12 & 15 & 12 & 5 \\ 4 & 9 & 12 & 9 & 4 \\ 2 & 4 & 5 & 4 & 2 \end{bmatrix}$$

[0080] 단계(1930)는 이미지의 강도 변화도(intensity gradient)의 추정을 포함한다. 이 작업을 수행하기 위해, 입력 이미지는 두 개의 Sobel 커널들에 의해 필터링된다. 커널 G_x의 연산은 이미지의 수평 방향의 1차 미분을 반환하며, 커널 G_y의 연산은 입력 이미지의 수직 방향의 1차 미분을 반환한다. 사용될 수 있는 커널 G_x 및 커널 G_y은 다음과 같다 :

[0081]
$$Gx = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad Gy = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

[0082] 입력 이미지의 1차 수평 방향 미분 및 1차 수직 방향 미분에 기초하여, 에지 변화도(edge gradient) G와 각 픽셀의 방향 Θ이 다음과 같이 계산될 수 있다 :

[0083]
$$G = \sqrt{G_x^2 + G_y^2} \qquad \theta = \tan^{-1}\left(\frac{G_x}{G_y}\right)$$

[0084] 변화도 방향은 일반적으로 에지들에 수직하며, 그리고 수직, 수평 및 두 개의 대각선 방향들을 나타내는 4 개의 각도들 중 하나로 반올림(rounded)될 수 있다.

[0085] 단계 1940에서, 에지 강도 이미지의 완전한 스캔은 에지 또는 원하는 에지를 구성하지 않을 수 있는 임의의 원하지 않는 픽셀들을 제거하기 위해 수행될 수 있다. 이는 각 픽셀이 그 변화도 방향으로 그 이웃에서 극대값인지를 검사함으로써 달성된다. 도 20의 그래프(2000)에 도시된 바와 같이, 점 A는 수직 방향의 에지 상에 있고, 변화도 방향은 에지에 수직하다. 점 B와 점 C는 변화도 방향 내에 위치하므로, 점 A는 점 B와 점 C와 비교되어 그것이 극대점을 형성하는지 관찰할 수 있다. 만약 그렇다면, 그것은 프로세스의 그 다음 단계(1950)에서 고려되며, 그렇지 않으면 점 A에 픽셀 값 0을 할당함으로써 억제될 수 있다. 결과는 이진 이미지로, 값 1의 픽셀은 얇은 에지에 해당하며, 값 0의 픽셀은 에지가 아닌 것에 해당한다.

[0086] 단계 1950에서, 이전 단계에서 검출된 에지들 중 어느 것이 오탐(false positive)이 아닌 정탐(true positives)인지 추정되며, 이는 그것들이 입력 이미지에 의해 표현되는 실제 세계의 에지를 나타낼 가능성이 높다는 것을 의미한다. 이 작업을 수행하기 위해, 두 개의 임계값들이 정의될 수 있다 : minVal 및 maxVal. maxVal 보다 큰 강도 변화도를 갖는 에지들은 확실한 에지로 간주되며, minVal 아래의 에지들은 에지들이 아닌 것으로 간주되어 무시된다. 2 개의 임계값들 내에 있는 에지들은 그것들의 연결 특성에 의해 에지들 또는 비-에지들로서 더 분류될 수 있다. 그것들이 확실한 에지 픽셀들에 연결되면, 그것들은 에지의 일부를 형성하는 것으로 간주된다. 그렇지 않으면, 그것들은 오탐들로 버려질 수 있다. 예를 들어 도 21의 그래프(2100)에서, 에지 A는 maxVal보다 높기 때문에, 그것은 정탐으로 간주될 수 있다. 에지 C가 maxVal 미만이지만, 에지 A에 연결되어 있으므로, 정탐 에지로 처리될 수 있으며, 전체 곡선이 유효한 것으로 간주될 수 있다. 에지 B가 minVal보다 크고 에지 C와 동일한 영역에 있지만, 어떠한 정탐 에지들에도 연결되어 있지 않으므로, 오탐으로 처리될 수 있다. minVal 값 및 maxVal 값은 최적의 결과를 얻도록 선택된다. 예를 들어, minVal은 20 내지 60의 값으로 설정될 수 있으며, maxVal은 60 내지 180의 값으로 설정될 수 있다. 이 단계는 또한 작은 픽셀 클러스터들의 형태의 노이즈를 제거할 수 있다.

[0087] 도 19에서 식별된 단계들 중 일부 또는 전부는 이미지 처리 라이브러리(922)에서 이용 가능한 프로그램들을 통해 수행될 수 있다. 예를 들어, OpenCV 라이브러리가 사용된다면, "canny" 에지 검출 함수 호출이 사용될 수 있다. 에지 검출의 다른 대안적인 방법들은 입력 이미지 내의 에지들의 동일한 식별 결과를 얻기 위해 canny 에지 검출에 대한 대안으로서 이용될 수 있다.

[0088] **윤곽 검출** : 에지 검출 연산자가 입력 이미지에 적용되어 에지들을 식별한 후에, 윤곽 검출 프로세스들은 에지 검출 작업의 결과에 적용되어, 예를 들어 다각형 또는 원과 같은 특정 모델 형상들과 이미지 내의 형상들의 유사도를 근사화할 수 있다.

[0089] 윤곽들은 같은 색상이나 강도를 가진 모든 연속점들을 (경계를 따라) 연결하는 곡선으로 설명될 수 있다. 윤곽들은 형상 분석 및 객체 감지 및 인식에 유용한 도구이다. 정확성을 높이기 위해, 이진 이미지들이 윤곽 검출 알고리즘들(또는 프로세스들)에 대한 입력으로서 사용되는 것이 제안된다. 따라서 윤곽들을 찾기 전에, 스레스홀딩 또는 canny 에지 검출을 적용하는 것이 좋다. 우리의 출원에서, 디지털화된 이진 이미지들의 토폴로지 분석을 위해 경계 추적 알고리즘들(또는 프로세스들)을 사용한다. 이러한 알고리즘들(또는 프로세스들)은 이진 이미지의 경계들 사이의 주변 관계들(surroundness relations)을 결정한다. 바깥 경계들과 홀 경계들이 각각 픽셀들의 연결된 컴포넌트들과 홀들에 일대일 대응하기 때문에, 알고리즘(또는 프로세스)은 이진 이미지의 표현을 산출하며, 상기 이진 이미지의 표현으로부터, 이미지를 재구성하지 않고 일부 피처들을 추출할 수 있다. 첫 번째의 수정된 버전인 두 번째 경계 추적 알고리즘(또는 프로세스)은 최외곽 경계들(즉, 홀들로 둘러싸여 있지 않은 외측 경계들)만을 구할 수 있다.

[0090] 윤곽 근사법(contour approximation) 또한 수행된다. 이는 우리가 지정한 정밀도에 따라, 윤곽 모양을 더 적은 수의 정점(verticle)들을 갖는 다른 모양(다각형)으로 근사한다. 이는 다음과 같이 Douglas-Peucker 알고리즘을 통해 구현될 수 있다.

[0091] function DouglasPeucker (PointList [], epsilon)

[0092] // Find the point with the maximum distance

[0093] dmax = 0

[0094] index = 0

[0095] end = length (PointList)

[0096] for i = 2 to (end - 1) {

[0097] d = perpendicularDistance (PointList [i],

[0098] Line (PointList [1] , PointList [end]))

[0099] if (d > dmax) {

[0100] index = i

[0101] dmax = d

[0102] }

[0103] }

[0104] // If max distance is greater than epsilon,

[0105] // recursively simplify

[0106] if (dmax > epsilon) {

[0107] // Recursive call

[0108] recResults1 [] = DouglasPeucker (PointList [1... index],

[0109] epsilon)

[0110] recResults2 [] = DouglasPeucker (PointList [index ... end],

[0111] epsilon)

[0112] // Build the result list

[0113] ResultList [] = { recResults1 [1... length (recResults1) -1],

[0114] recResults2 [1... length (recResults2)]}

[0115] } else {

[0116] ResultList [] = { PointList [1] , PointList [end]}

[0117] }

[0118] // Return the result

[0119] return ResultList []

[0120] end

[0121] 도 22에서, 일련의 이미지들(2200)은 Douglas-Peucker 알고리즘의 다양한 적용 단계들을 나타낸다. 이미지(2210)는 입력 이미지로서 사용될 수 있는 원본 이미지이다. 이미지(2220) 내의 라인(2205)은 아크 길이의 10 %에 해당하는 엡실론(epsilon)의 값에 대한 근사 곡선을 나타낸다. 이미지(2230) 내의 라인(2215)은 아크 길이의 1 %에 해당하는 엡실론의 값에 대한 근사 곡선을 나타낸다.

[0122] 윤곽 추정 작업들은 게임 모니터링 모듈(928)에서 그들을 호출함으로써 이미지 처리 라이브러리(922)에서 사전 패키지된 함수들을 사용함으로써 수행될 수 있다. 예를 들어, OpenCV가 윤곽 추정 프로세스를 구현하는데 사용된다면, 예를 들어 함수들 "findContours" 또는 "drawContours" 또는 "approxPolyDP"가 프로세스 구현을 위해 호출될 수 있다.

[0123] **평면 검출 프로세스** : 게임 테이블 또는 경기 표면상의 칩들의 스택의 높이를 추정하기 위한 사전 단계로서, 게임 모니터링 시스템(100)은 게임 테이블 또는 경기 표면을 포함하는 평면의 위치를 추정한다. 평면의 방정식을 추정하는 한 가지 방법은 주성분 분석(Principal Component Analysis; PCA)을 이용하는 것이다. PCA는 일련의 데이터에서 피팅된 모델까지의 수직 거리를 최소화한다. 이것은 직교 회귀(Orthogonal Regression) 또는 TLS(Total Least Square)라고 알려진 선형 사례(linear case)이다. 예를 들어, 두 개의 데이터 벡터들 x와 y가 주어지면, 각 점(x_i, y_i)에서 라인까지의 수직 거리를 최소화하는 두 변수들이 있는 선형 방정식 형태의 라인이 추정될 수 있다. 보다 일반적으로, r-차원 초평면은 p-차원 공간에 적합할 수 있으며, 여기서 r은 p보다 작다. r의 선택은 PCA 동안 유지할 컴포넌트들의 수를 선택하는 것과 같다.

[0124] 평면의 기본 수학적 모델은 다음과 같이 공식화될 수 있다 :

[0125] $$ax + by + cz + d = 0$$

[0126] 값 a, b, c 및 d는 게임 테이블 또는 경기 표면에서 선택된 점들로부터의 거리를 최소화하기 위해 추정될 필요가 있다. 칩 검출 단계 동안 획득된 이진 버전의 이미지에 기초하여, 칩들로서 검출되지 않은 100 개 이상의 점들이 PCA에 대한 입력 점들로서 선택될 수 있다. 이러한 선택된 점들은 2차원 공간의 점들이기 때문에, 점들과 연관된 깊이 정보는 2차원 공간에서 좌표들을 얻기 위해 이용된다. Pinhole 카메라 모델의 원리는 2차원 공간의 점들을 3차원 공간의 점들로 변환하기 위해 사용된다. 깊이 감지 기기 및 카메라(120)의 광학 중심의 x, y 좌표들로서 (C_x, C_y), 깊이 값 Z, 그리고 좌표들 (x, y)을 갖는 2차원 공간 내의 점이 주어질 때, 3차원 공간 내의 동일한 점의 좌표(X, Y, Z)는 다음 방정식들을 사용하여 결정될 수 있다 :

[0127] $$\frac{Y}{Z} = \frac{x - C_x}{f} \qquad \frac{Y}{Z} = \frac{y - C_y}{f}$$

[0128] 도 23a 및 도 23b는 3차원 좌표 그래프(2300)에서 한 세트의 포인트들(2310)에 PCA를 적용하는 것을 보여준다. 도 23b의 좌표 그래프(2301)에서, PCA의 적용은 점들(2310)과 평면(2350) 간의 직교 거리들(2320)을 최소화하는 장소(2350)의 식별을 가능하게 한다. 평면 추정 작업들은 게임 모니터링 모듈(928)에서 그들을 호출함으로써 이미지 처리 라이브러리(922)에서 사전 패키지된 함수들을 사용하여 수행될 수 있다. 예를 들어, OpenCV가 윤곽 추정 프로세스를 구현하는데 사용되면, 예를 들어 클래스 "cv::PCA"에 의해 구현되는 함수들이 평면 추정 프로

세스 구현을 위해 호출될 수 있다.

[0129] 칩들의 값을 추정하기 위해, 이미지들의 유클리드 거리가 사용되어 칩들을 분류할 수 있다. 비교 목적으로 칩 템플릿 이미지들이 미리 수집될 것이다. 그런 다음, k-최근접 이웃 알고리즘(k-nearest neighbours algorithm)이 사용되어 칩들의 값을 할당한다. 그러나, 유사한 색상을 가진 몇몇의 칩 유형들이 있다. 이러한 유형들의 칩들을 구분하기 위해, 적외선 이미지로부터의 칩 유형 반사율(reflectivity)을 더 사용하여 동일한 k-최근접 이웃 알고리즘으로 그것들을 분류한다.

[0130] 테이블 평면을 가진 후, 각 칩의 중심에서 그 평면까지의 거리는 칩들의 스택의 높이를 얻기 위해 추정될 수 있다. 또한, 칩들의 수는 선형 또는 비-선형 매핑에 의해 이 높이를 기반으로 추정될 수 있다. 점에서 평면까지의 거리를 계산하는 것은 다음과 같이 도출될 수 있다. 3차원 공간에서 평면이 다음과 같이 주어지고

[0131] $ax + by + cz + d = 0$,

[0132] 그리고 점 $x_0 = (x_0; y_0; z_0)$이 주어지면, 평면에 대한 법선 벡터는 다음과 같이 주어지며 :

[0133] $$v = \begin{bmatrix} a \\ b \\ c \end{bmatrix},$$

[0134] 그 다음 점으로부터 평면까지의 거리는 다음과 같이 계산된다 :

[0135] $$D = \frac{|ax_0 + by_0 + cz_0 + d|}{\sqrt{a^2 + b^2 + c^2}}$$

[0136] 카드 검출을 위한 알고리즘(또는 프로세스)이 수행될 수 있다. 예를 들어 카드 기반 게임 및 비-카드 기반 게임의 두 가지 유형의 게임들이 존재할 수 있다. 카드 기반 게임의 경우, 1의 알고리즘(또는 프로세스)이 사용되어 카드들을 검출하고 게임 시작 이벤트들을 트리거할 수 있다. 룰렛의 경우, 돌리 검출이 사용되어 게임 시작 및 종료 이벤트들을 트리거할 수 있다. 게임 모니터링 시스템이 돌리(또는 위치 마커)를 검출하면, 돌리가 검출되기 몇 초 전, 예를 들어 20 ~ 30 초전에, 게임이 시작되었다고 추측할 수 있다. 돌리는 딜러에 의해 제거될 때까지 그 위치를 유지할 수 있다. 초기 검출 후 돌리의 제거는 게임 종료 이벤트를 트리거할 수 있다. 돌리의 반사 특징(reflective feature)으로 인해, 컬러 이미지 대신에 적외선 이미지가 사용되어 이를 검출할 수 있다.

[0137] 알고리즘 1 카드 검출 알고리즘(프로세스) :

[0138] 1. 절차 CardDection(img). 그레이 스케일 이미지 img 입력

[0139] 2. img에 Canny 에지 검출 적용

[0140] 3. 에지 세그먼트들 사이의 잠재적인 홀들을 제거하기 위해 canny 출력 팽창

[0141] 4. img에 Canny 에지 검출을 적용

[0142] 5. 윤곽선 찾고 근사하기

[0143] 6. 다음 기준을 만족하는 윤곽들 수용 :

[0144] 윤곽의 면적은 카드들의 면적 크기 이내여야 한다(예를 들어 40 cm^2 내지 70 cm^2 또는 50 cm^2 내지 60 cm^2)

[0145] 윤곽은 근사 후에 4 개의 정점들을 가져야 한다

[0146] 연결 에지들(joint edges) 간의 각도의 코사인은 작아야 한다(예를 들어, - 0.2 내지 0.2 또는 - 0.1 내지 0.1)

[0147] 알고리즘 2 돌리 검출 알고리즘(프로세스) :

[0148] 1. 절차 DollyDection(img). 적외선 이미지 img 입력

[0149] 2. img에 글로벌 스레스홀딩(global thresholding) 적용

[0150] 3. 작은 물체들을 제거하기 위해 출력을 침식

[0151] 4. 크기 기준을 충족하는 남은 물체가 존재한다면, 그것은 검출된 돌리가 될 것이다.

[0152] 5. 칩 검출. 칩들을 검출하기 위해, 백그라운드에서 칩들을 세분화하기 위해 적응형 스레스홀딩을 사용한다. 출력 바이너리는 침식되고 팽창되어 작은 물체들을 제거할 것이다. 이후, 크기 기준을 충족하는 형태가 뚜렷하지 않은 것들(blob)은 모두 칩들로 검출될 것이다. 이 알고리즘(또는 프로세스)은 컬러 이미지 또는 적외선 이미지 모두에 적용될 수 있다. 아래의 서브-섹션들은 사용된 기법들의 재검토이며, 알고리즘(또는 프로세스)의 세부사항은 알고리즘 3에 설명되어 있다.

[0153] 알고리즘 3 칩 검출 알고리즘(또는 프로세스)(프로세스) :

[0154] 1. 절차 CardDection(img). 그레이스케일 이미지 img 입력

[0155] 2. img에 적응형 스레스홀딩을 적용

[0156] 3. 작은 물체들을 제거하기 위해 출력을 침식 및 팽창

[0157] 4. 크기 기준을 충족하는 남은 물체가 존재한다면, 그것은 검출된 돌리가 될 것이다

[0158] 크기 기준 또는 각도의 코사인 기준과 같은 특정 기준은 테이블 구성(942)의 일부로서 구성 모듈(940)에 저장될 수 있다.

[0159] 칩 높이 추정 알고리즘은 알고리즘 4에서 볼 수 있다 :

[0160] 알고리즘 4 칩 스택 높이 추정 알고리즘(프로세스) :

[0161] 1. 절차 ChipHeightEstimate(img). 깊이 이미지 img 입력

[0162] 2. 칩들이 아닌 선택된 테이블 점들을 3D 좌표들로 변환

[0163] 3. 이 점들을 테이블 평면으로 불리는 평면에 맞추기(fitting)

[0164] 4. 칩 스택의 각 중심으로부터 이 평면까지의 거리를 찾아 칩 스택 높이를 구하기

[0165] 5. 스택의 칩들의 수를 추정하기 위해 상기 높이를 단일 칩의 높이로 나누기

[0166] 도 26a 및 도 26b는 각각 카드 검출 및 칩 검출이 수행되는 중인 게임 테이블의 일 실시예의 이미지 프레임(2600, 2650)이다. 도 26a에서, 경계(2620)는 카드가 검출된 이미지 프레임의 영역을 나타낸다. 경계(2630)는 칩이 검출된 이미지 프레임의 영역을 나타낸다. 카드(2610)는 깊이 감지 기기 및 카메라(120)에 완전히 제공되지 않았고 게임 테이블(210) 상에서 돌려지고 있는 과정에 있기 때문에 카드 검출 알고리즘(또는 프로세스)에 의해 검출되지 않았다. 도 26b에서, 도 26a에 제시된 카드와 상이한 카드(211)가 검출되고 경계(2620)에 의해 둘러싸여진다. 도 26b의 칩들(213)은 도 26a의 칩들과 다른 위치들에 있고 경계(2630)로 검출되었다.

[0167] 도 27a 및 도 27b는 각각 카드 검출 및 칩 검출이 수행되는 중인 게임 테이블의 일 실시예의 이미지 프레임(2700, 2710)이다. 도 27b는 이진 스레스홀딩 기법들 중 임의의 하나를 도 27a에 적용한 결과이다. 도 27b의 형상(2710)은 카드들(211)에 대한 잠재적 오탐지(false positive detection)들일 수 있다. 이러한 오탐들은 형상의 면적을 카드의 예상 면적과 비교하는 알고리즘 1의 단계6에 의해 제거될 수 있다. 유사하게, 형상들(2720)은 형상들(2720)의 면적과 칩들(213)의 예상 면적의 비교에 의해 침들(213)에 대한 오탐지들로서 제거될 수 있다. 도 28a는 카드들(211)과 칩들(213) 같은 게임 물체들을 갖는 게임 테이블(2800)의 일실시예의 적외선 이미지를 나타낸다. 도 28b에서, 칩들(211)은 칩 검출 알고리즘 3에 의해 식별되고, 검출을 나타내기 위해 회색으로 표시된다. 컬러 이미지 프레임들에 게임 물체 검출 프로세스들을 적용한 결과들은 칩 스택 높이의 검출 및 추정의 정확성을 향상시키기 위해 적외선 이미지 프레임들과 결합될 수 있다.

[0168] 도 29a 내지 도 29e는 일부 실시예들에 따른 이미지 프레임(2900)에 대한 칩 검출 프로세스의 적용을 도시한다. 이미지 프레임(2900)은 깊이 감지 기기 및 카메라(120)에 의해 캡처될 수 있다. 도 29b는 도 29a의 입력 이미지 프레임(2900)에 이진 스레스홀딩 연산을 적용함으로써 획득된 이미지 프레임(2901)이다. 도 29c의 이미지 프레임(2902)은 도 29b의 이미지 프레임(2901)에 침식 작업을 적용함으로써 획득된다. 도 29d는 도 29c의 입력 이미지 프레임(2902)에 팽창 작업을 적용함으로써 획득된 이미지 프레임(2903)이다. 침식 및 팽창 작업들의 조합은 입력 이미지 프레임의 노이즈를 감소시키고 칩과 같은 관찰 가능한 피처들을 더 두드러지게 만드는 것을 돕는다. 도 29e는 칩 검출 프로세스의 결과들을 도시하는 이미지 프레임(2904)이다. 검출된 칩들(213)은 윤곽선(2920)에 의해 식별된다. 도 29d의 형상들(2910)은 칩 검출 프로세스의 일부로서 기준을 만족시킬 수 없기 때문에 칩으로 식별되지 않는다. 기준은 칩으로서 검출될 형상에 대한 크기 범위, 또는 다른 유사한 구별 기준일 수 있다.

[0169] 일부 실시예들은 게임 테이블들의 게임 이벤트들을 모니터링하기 위해 도 24의 흐름도(2400)의 단계들을 따른다. 프레임 그래버(2410)는 컬러 비디오 프레임들, 깊이 프레임들 및 적외선 프레임들을 포함하는 3 개의 프레임 스트림을 수집한다. 미가공 컬러 이미지들은 1920 x 1080 해상도를 갖는다. 깊이 스트림은 가시 영역의 모든 점의 깊이 값을 제공한다. 깊이 값은 상기 관측된 점의 센서로부터의 거리이다. 적외선 스트림을 사용하면 시야의 더 어두운 부분을 보다 선명하게 볼 수 있다.

[0170] 게임 시작 감지는 2420에서 발생한다. 게임이 시작될 때를 검출하기 위해, 일부 실시예는 카드들을 트리거로서 사용한다. 예를 들어, 룰렛과 같이 카드들이 없으면, 게임이 끝날 때 돌리가 검출될 수 있으며, 게임 시작 이벤트가 n번째 프레임에서 다시 트리거링된다.

[0171] 칩 검출 및 추정은 2430에서 발생한다. 이는 칩들의 위치를 검출하고, 칩 캐시 값을 추정하고, 그리고 얼마나 많은 칩들이 해당 칩 스택에 있는지를 추정하는 프로세스이다. 상기 알고리즘(또는 프로세스)는 적응형 스레스홀딩 및 모폴로지 연산을 기반으로 한다. 이는 컬러 및 적외선 이미지들 모두에 적용될 수 있다.

[0172] 칩 값 추정은 2460에서 적외선 이미지의 각 칩 유형들의 반사 피처들 그리고 컬러 이미지들의 템플릿 매칭을 사용함으로써 수행된다. 그 다음, 칩 높이 추정은 칩의 상단 표면과 테이블 평면 사이의 거리를 기반으로 계산된다.

[0173] 게임 종료 검출(2460)은 게임 시작 검출과 반대되는 프로세스이다. 딜러가 카드를 치우면, 게임 종료 이벤트가 트리거링된다. 룰렛의 경우, 돌리 제거 검출이 이 이벤트를 트리거할 것이다.

[0174] **게임 시작 및 종료 검출** : 도 25는 게임 테이블들상의 이벤트들을 모니터링하기 위해 일부 실시예들이 구현할 수 있는 게임 모니터링 프로세스(2500)의 흐름도를 도시한다. 이 프로세스에서 두 개의 중요한 단계들은 게임 시작 검출(2520) 및 게임 종료 검출(2560)이다. 게임 시작 및 종료 트리거들을 정의하는 이벤트들의 특정 속성은 게임 시작 및 종료 트리거 구성(944)에 저장될 수 있고, 그리고 게임 모니터링 모듈(928)에 의해 참조되어 게임이 테이블상에서 시작되었거나 끝났는지를 추정할 수 있다. 예를 들어, 블랙잭의 게임을 위해 지정된 테이블에 대해, 단계(2510)에서 추출된 이미지 프레임 내의 하나 이상의 카드들의 존재는 게임의 시작으로서 취급될 수 있다. 마찬가지로, 게임이 시작된 후, 이미지 프레임 내에 어떠한 카드들도 존재하지 않는다는 것은 게임 종료 검출(2560)의 목적을 위한 게임의 종료로서 취급될 수 있다. 룰렛과 같이 카드를 기반으로 하지 않는 게임의 경우, 돌리와 같은 다른 게임 물체들의 존재가 게임의 시작 및 종료 트리거들로 사용될 수 있다. 게임 시작 또는 종료 트리거의 특정 형상 및 특성은 컴퓨팅 기기(130)의 구성 모듈(940)의 게임 시작 및 종료 트리거 구성(944)에 저장될 수 있다. 이러한 구성들은 데이터베이스 서버(150)상의 웹 애플리케이션 서버(160)를 통해 관리될 수 있으며, 그리고 이어서 테이블 구성 전파 모듈(1140)을 통해 컴퓨팅 기기(130)에 전송될 수 있다.

[0175] **전체 모니터링 프로세스** : 게임 모니터링 프로세스(2500)는 이미지 프레임(2510)의 추출 단계로 시작한다. 이 단계는, 깊이 감지 기기 및 카메라(120)에 의한 게임 환경(110)의 이미지 캡처, 그리고 게임 모니터링 모듈(928)에 의한 분석을 위해 이용 가능한 컴퓨팅 장치(130)로의 링크(107)를 통한 상기 캡처된 이미지의 전송을 포함한다. 상기 이미지들은 컬러 포맷으로 또는 적외선 포맷으로, 또는 두 포맷 모두로, 또는 깊이 감지 기기 및 카메라가 이미지를 캡처할 수 있는 임의의 다른 포맷으로 캡처될 수 있다. 다음 단계(2520)는 예를 들어 알고리즘 1 : 카드 검출 알고리즘, 또는 알고리즘 2 : 돌리 검출 알고리즘의 적용에 의해 게임의 시작을 검출하는 것을 포함한다. 게임의 시작이 검출되면, 게임 테이블 또는 경기 표면의 평면을 자동으로 추정하는 단계(2530)가 수행된다. 단계(2530)는 상술한 PCA를 사용하여 수행될 수 있고, 그 결과들은 컴퓨팅 기기(130)의 메모리에 저장될 수 있다.

[0176] 다음 단계(2540)는 칩들의 검출을 포함한다. 이 단계는 알고리즘 3 : 칩 검출 알고리즘을 통해 수행될 수 있다. 칩이 검출된다면, 다음 단계(2550)는 이미지 내의 칩들의 스택의 값을 추정하는 것일 수 있다. 이 단계는 실시간으로 또는 거의 실시간으로 구현될 수 있는 알고리즘 4 : 칩 스택 높이 추정 알고리즘 또는 프로세스를 구현함으로써 수행된다. 또한, 프로세스의 일부는 스택의 최상부에 있는 칩 또는 내기 물체의 색상을 자동으로 추정하는 것, 그리고 상기 테이블 구성 모듈(942)에서 상기 색상과 관련된 값을 검색하는 것을 포함할 수 있다. 검색된 값은 전체 스택의 값을 자동으로 추정하기 위해 상기 스택 내의 내기 물체들 또는 칩들의 추정된 개수와 곱해질 수 있다. 이 알고리즘 또는 프로세스의 적용 결과들은 컴퓨팅 기기(130)의 메모리에 저장될 수 있으며, 그리고 이벤트 또는 트랜잭션으로서 기록될 수 있다. 또한, 검출된 칩 스택은 그것을 동일한 게임 테이블상의 다른 칩 스택과 구별하기 위해 테이블상의 관심 영역과 관련된다.

[0177] 단일 관심 영역상의 칩들 또는 내기 물체들의 다수의 스택들은 개별적으로 검출될 수 있고, 그리고 각 스택의

추정된 값은 관심 영역의 고유 식별자와 함께 개별적으로 기록될 수 있다. 대안적으로, 단일 관심 영역상의 다수의 칩 스택들의 값은 수집되어 관심 영역의 고유 식별자와 함께 기록될 수 있다. 칩들의 다수의 스택들의 높이는 한 번의 반복으로, 또는 알고리즘 4의 구현의 여러 반복으로 추정될 수 있다. 칩들의 스택의 값을 추정한 후에, 다른 이미지 프레임은 단계(2514)를 통해 그래빙(grabbing)되며, 그리고 게임 종료 검출 단계(2560)가 수행된다. 게임 종료 검출은 예를 들어 단계(2514)에서 그래빙된 이미지 프레임 내의 게임 테이블에 카드가 없는 것 또는 돌리가 없는 것을 찾음으로써 수행될 수 있다. 게임 종료가 검출되지 않으면, 단계(2512)에서 다른 이미지 프레임이 그래빙되며, 그리고 단계(2540)에서 칩들의 검출의 다른 반복이 뒤따른다.

[0178] 게임의 종료가 검출되면, 그 다음 단계(2570)에서, 모니터링 프로세스 동안 캡처된 이벤트 데이터는 메시지 브로커 서버(140)를 통해 게임 이벤트 데이터(1124)로서 저장되도록 데이터베이스 서버(150)에 전송된다. 다른 실시예들에서, 데이터베이스 서버(150)에 트랜잭션을 보고하는 단계는 이벤트들이 실시간으로 또는 거의 실시간으로 검출됨에 따라 발생할 수 있다. 이 이벤트 데이터는 이벤트가 발생한 타임스탬프, 칩들이 검출된 관심 영역들, 칩들의 스택들의 추정된 값, 게임 시작 및 종료 시간, 그리고 게임 모니터링 모듈(928)에 의해 캡처된 다른 관련 파라미터들을 포함할 수 있다. 데이터베이스 서버에 트랜잭션들을 보고한 후, 게임 모니터링 시스템은 게임 테이블을 계속 모니터링하고 단계(2510) 및 단계(2520)에서 다음 게임의 시작을 기다릴 수 있다. 게임 중에 캡처된 이벤트 데이터의 일부 기록들의 예들은 다음과 같다 :

[0179] 게임 기록 1 <Table ID: R.2745; Game ID: 17.0.20170421150202; Game Start Time: 2017-04-21 15:02:02; Game End Time: 2017-04-21 15:04:02>

[0180] 게임 물체 기록 1 <Table ID: R.2745; Game ID: 17.0.20170421150202; Region of Interest ID: Player-1, Wager Object Value: 5; Wager Object Count: 5>

[0181] 게임 물체 기록 2 <Table ID: R.2745; Game ID: 17.0.20170421150202; Region of Interest ID: Player-2, Wager Object Value: 10; Wager Object Count: 2>

[0182] 상기의 게임 이벤트 데이터 예들에서, 고유 식별자 "Game ID: 17.0.20170421150202"를 갖는 게임 기록 1은 고유 식별자 "Table ID: R.2745"에 의해 식별되는 고유한 게임 테이블 또는 경기 표면과 연관된다. 또한, 게임 기록은 게임 시작 및 종료 시간 타임스탬프 값들을 포함한다. 두 개의 게임 물체 기록들, 게임 물체 기록 1 및 게임 물체 기록 2가 게임 기록 1과 연관된다. 게임 물체 기록 1은 고유한 식별자 "Player-1"를 갖는 관심 영역에서 검출된 5의 값을 갖는 5 개의 내기 물체들의 추정을 나타낸다. 게임 물체 기록 2는 고유 식별자 "Player-2"를 갖는 관심 영역에서 검출된 10의 값을 갖는 2 개의 내기 물체들의 추정을 나타낸다.

[0183] **멀티 프레임 처리** : 전술한 게임 물체 검출에 대한 접근법들은 임의의 처리를 수행하기 위해 단일 프레임들 또는 스냅샷들을 고려한다. 이 접근법에 의해 생성된 결과는 오탐을 방지하고 멀티 프레임 처리 기술들을 통해 더 높은 정확성을 달성하도록 더욱 향상될 수 있다. 이 기술은 게임 물체 검출 및 칩 높이 추정을 위해 모든 프레임을 처리하는 것과 모든 처리 반복 후에 획득된 결과들을 로깅(logging)하는 것을 포함한다. 로깅된 결과들에 특정 처리 규칙들을 적용하여 게임 물체 검출 및 칩 스택 높이 추정 단계들을 개선할 수 있다. 예를 들어, 여러 프레임들에 걸쳐 추정된 칩 스택 값들의 평균을 사용하여 잘못된 관측을 한 몇 가지 프레임들을 보상할 수 있다. 또한, 제1 카드 검출이 지연될 수 있는 게임들에서, 게임 모니터링 모듈(928)은 캡처된 프레임으로 제 시간에 거슬러 올라가서 카드 검출을 다시 수행하여, 새로운 게임의 시작을 소급하여 식별할 수 있다. 다수의 관심 영역들에서 카드들을 검출하고 그리고 관심 영역들의 특정 비율(예를 들어, 8 개의 영역들 중 3 개의 영역들, 또는 12 개의 영역들 중 2 개의 영역들)이 카드들을 갖는 것으로 검출되는 경우에만 게임이 개시되는 것으로 처리함으로써, 게임을 트리거링하는 움직이는 물체들이 감소될 수 있다.

[0184] 또한, 깊이 감지 기기 및 카메라(120)의 뷰로부터 일시적으로 방해받을 수 있는 게임 물체들을 검출하기 위해 다중 프레임 처리 기술들이 이용될 수 있다. 도 31은 딜러의 머리(3010) 형태의 장애물을 갖는 게임 테이블의 뷰를 나타내는 이미지 프레임(3000)이다. 도 31은 이미지 프레임(3000)의 몇 초 전 또는 몇 초 후에 취해진 도 30의 게임 테이블의 뷰를 나타내는 이미지 프레임(3100)이다. 이미지 프레임(3100)은 장애물(3010)을 가지지 않으며, 그리고 칩 검출 알고리즘의 적용은 이미지 프레임(3000)에서 이전에 검출되지 않은 칩(3113)을 식별하게 한다. 2-4초 이상 또는 1-2초 이상 또는 1 초 이내와 같이 적은 시간 동안 획득될 수 있는 이미지 프레임들에 대한 게임 물체 검출 프로세스들의 적용에 의해 획득된 결과들의 조합이 사용되어 게임 물체 검출 프로세스의 전체적인 정확성을 향상시킬 수 있다.

[0185] 또한, 상술된 멀티 프레임 처리 기법은, 예를 들어 시각적 이미지 카메라 및 적외선 카메라로부터 획득된 이미

지들, 또는 하나 이상의 깊이 감지 기기들 및 카메라(120)로부터 획득된 이미지들과 같이, 상이한 소스들로부터 획득된 이미지 프레임들로 확장될 수 있다. 도 32는 깊이 감지 기기 및 카메라(120)의 일부일 수 있는 시각적 이미지 카메라로부터 획득된 이미지 프레임(3200)이다. 이 이미지(3200)에서, 칩 검출 프로세스의 적용은 칩(3210)을 식별함으로써 오탐 결과를 생성할 수 있다. 도 33은 이미지 전처리 단계들 및 에지 및 윤곽 검출 기술들이 미가공 이미지 프레임에 적용된 후에 적외선 카메라로부터 획득된 이미지 프레임(3300)이다. 이미지 프레임(3300)에서의 칩 검출 프로세서들의 적용은 이미지 프레임(3200)에서 생성된 오탐 결과를 생성하지 않는다. 따라서, 다수의 상이한(유형의) 이미지 프레임들에 게임 물체 검출 프로세서들을 적용함으로써 획득된 결과들의 조합은 몇몇 실시예들에서는 게임 물체 검출 결과들의 정확도를 향상시키는데 효과적일 수 있다.

[0186] 도 34의 흐름도는 일부 실시예들이 구현할 수 있는 멀티 프레임 처리 기법(3400)의 일 구현예이다. 단계(3410)는 깊이 감지 기기 및 카메라(120)를 통해 두 개의 이미지 프레임들(A 및 B)을 획득하는 단계를 포함한다. 이미지 프레임들(A 및 B)은 시각적 이미지 카메라 또는 적외선 카메라로부터의 것일 수 있고, 그리고 예를 들어 캡처 시간에 의해 수밀리초만큼 분리될 수 있다. 대안적으로, 이미지 프레임들(A 및 B)은 예를 들어 동일한 시점에서 각각 시각 이미지 카메라 및 적외선 카메라로부터 획득될 수 있다. 두 이미지 프레임들(A 및 B)은 실질적으로 동일한 시각에서 취해지며, 그리고 동일한 게임 테이블 또는 경기 표면을 나타낸다.

[0187] 단계(3420)는 이미지 프레임들(A 및 B)의 각각의 관심 영역에 게임 물체 검출 프로세스를 적용하는 것을 포함한다. 게임 물체 검출 프로세스는 예를 들어 카드 검출 프로세스, 또는 칩 또는 내기 물체 검출 프로세스일 수 있다. 주어진 관심 영역에 대한 게임 물체 검출 프로세스로부터 획득된 결과들은 예를 들어 다음 중 하나 이상을 포함할 수 있다 : 관심 영역 내의 게임 물체의 존재의 식별; 관심 영역 내의 게임 물체들의 수; 관심 영역 내의 게임 물체들의 개별 그룹들의 수; 각각의 식별된 그룹 내의 게임 물체들의 수; 및/또는 게임 물체의 색상. 게임 물체 검출 프로세스가 특정 관심 영역 내의 게임 물체를 식별하지 않는다면, 그 프로세스의 결과는 null 결과일 수 있으며(보고된 결과가 없음) 또는 0 개의 게임 물체들이 검출되었음을 나타내는 결과일 수 있다. 특정한 미리 정의된 관심 영역들은 특정 게임 물체의 존재를 식별하기 위해 지정될 수 있다. 예를 들어, 미리 정의된 관심 영역은 게임 물체 검출 프로세스의 일부로서 게임 카드들의 식별을 위해 지정될 수 있다. 다른 미리 정의된 관심 영역들은 게임 물체 검출 프로세스의 일부로서 내기 물체들의 식별을 위해 지정될 수 있다.

[0188] 단계(3430)는 단계(3420)에서 이미지 프레임들(A 및 B)에 대해 게임 물체 검출 프로세스를 적용하여 획득된 결과들을 비교하는 것을 포함한다. 이미지 프레임들(A 및 B)로부터 획득된 결과들이 매칭된다면, 예를 들어 이미지 프레임들(A 및 B)에서 검출된 게임 물체들의 위치, 개수 및 색상이 동일하다면, 단계(3440)에서, 상기 획득된 결과들은 컴퓨팅 기기(130)에 의해 게임 이벤트로서 수용되며 그리고 보고된다.

[0189] 이미지 프레임들(A 및 B)로부터 획득된 결과들이 검출된 게임 물체들의 위치, 또는 개수, 또는 색상의 관점에서 매칭되지 않는다면, 그 다음 단계(3450)에서, 깊이 감지 기기 및 카메라(120)로부터 이미지 프레임(C)이 획득된다. 이미지 프레임(C)은 다른 소스, 예를 들어 시각 이미지 카메라 또는 적외선 카메라로부터 캡처될 수 있으며; 또는 이미지 프레임들(A 및 B)이 획득된 시간으로부터 (예를 들어 수 밀리초 만큼) 분리된 시각에서 취해질 수 있다. 모든 이미지들(A, B 및 C)은 실질적으로 동일한 시각에서 취해지며, 그리고 동일한 게임 테이블 또는 경기 표면을 나타낸다.

[0190] 단계(3460)에서, 단계(3420)에서 적용된 게임 물체 검출 프로세스는 또한 이미지 프레임(C)에 적용된다. 단계(3470)에서, 단계(3420) 및 단계(3460)에서 이미지 프레임들(A, B 및 C)에 대해 게임 물체 검출 프로세스를 적용하여 획득된 결과들이 비교된다. 이미지 프레임(A)으로부터 획득된 결과들이 이미지(C)로부터 획득된 결과들과 매칭된다면, 이미지 프레임(B)으로부터 획득된 결과들은 이상(anomaly)으로서 버려지며, 단계(3440)에서 이미지 프레임(A 또는 C)으로부터 획득된 결과들이 받아들여진다. 이미지 프레임(B)으로부터 획득된 결과들이 이미지(C)로부터 획득된 결과들과 매칭된다면, 이미지 프레임(A)으로부터 획득된 결과들은 이상(anomaly)으로 버려지며, 단계(3440)에서 이미지 프레임(B 또는 C)으로부터 획득된 결과들이 받아들여진다. 만약, 이미지 프레임들(A, 또는 B, 또는 C)로부터 얻어진 결과들 중 어느 것도 매칭되지 않는다면, 더 나은 게임 물체 검출 결과들을 획득하기 위해, 단계(3410)에서 새로운 이미지 세트가 획득될 수 있다.

[0191] 일부 실시예들에 대한 설명에서, 실시예들을 형성하기 위해 결합되는 소프트웨어 및 하드웨어 시스템의 특정 추상적 개념들인 도면들이 참조된다. 이러한 추상적 개념들은 그 자체로 이해되어야 하며, 실시예를 이해하는 것을 돕고 그것들의 복제 또는 구현을 가능하게 하기 위해 본원에 제시된다. 시스템을 블록들로 나누는 것은 시스템의 기능들에 따르며, 그리고 구현시 그러한 방식으로 논리적으로 또는 물리적으로 분리되는 소프트웨어 또는 하드웨어 컴포넌트를 가질 필요가 없다는 것을 이해하면서 제시된다. 마찬가지로, 시스템의 기능들이 본 개시서

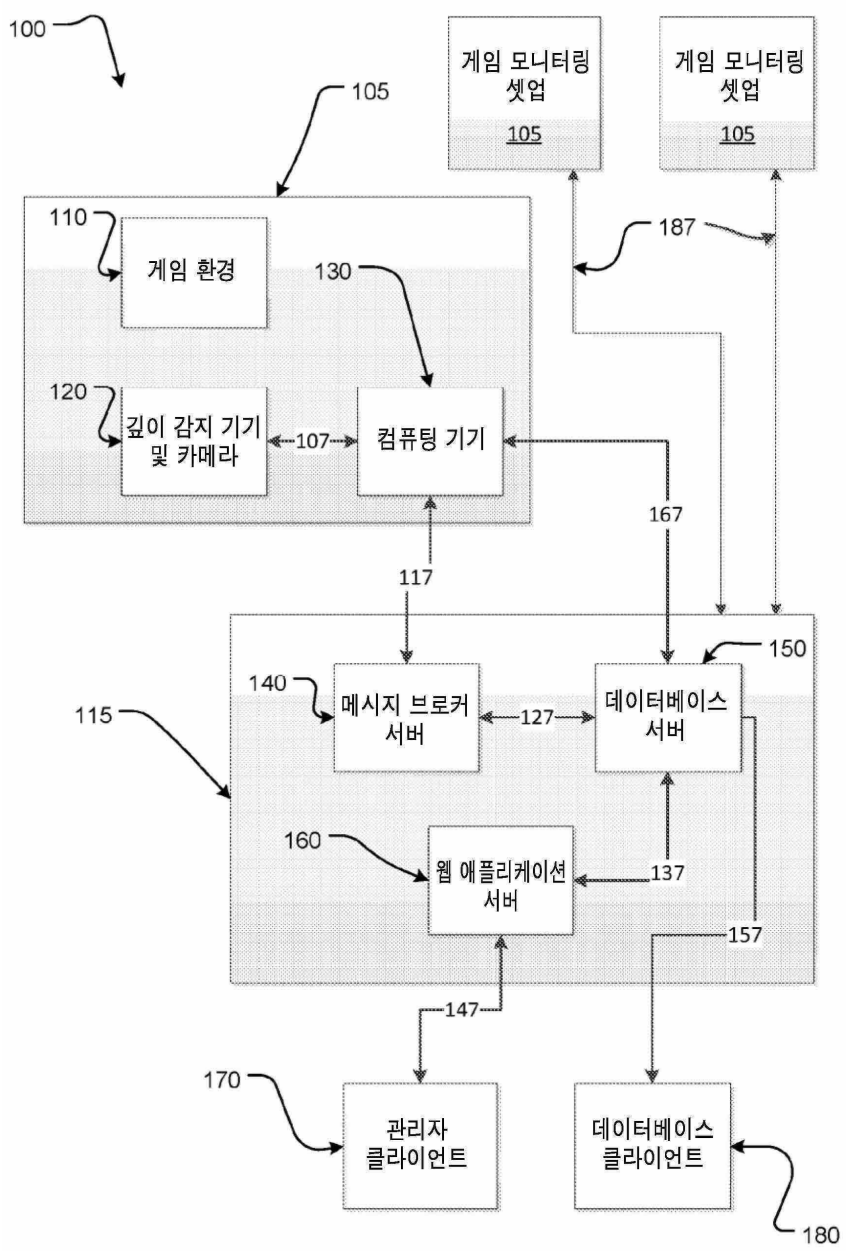
에서 벗어나지 않으면서 상이하게 분배될 수 있으므로, 물리적 장비 설치는 하나의 가능성에 불과하다. 편의상, 시스템 및 특정 예들이 블랙잭, 바카라 및 룰렛 게임들의 맥락에서 설명된다. 시스템의 일부 또는 모든 구성요소들은 다른 테이블 게임에도 적용될 수 있음이 이해되어야 한다.

[0192] 일부 실시예들은 카메라 뷰에 대한 초기 배경 이미지를 더 취할 수 있다. 이 배경 이미지는 빈 테이블일 수 있다. 이 이미지는 이미지 내의 핵심 피처들(테이블 레이아웃)을 기반으로 한 조감도로 와핑(warping)될 수 있다. 이 새로운 이미지는 정규화된 배경 이미지(normalized background image)라고 지칭될 수 있으며 그리고 그 후에 배경 추상화(background abstraction)에 사용될 수 있다.

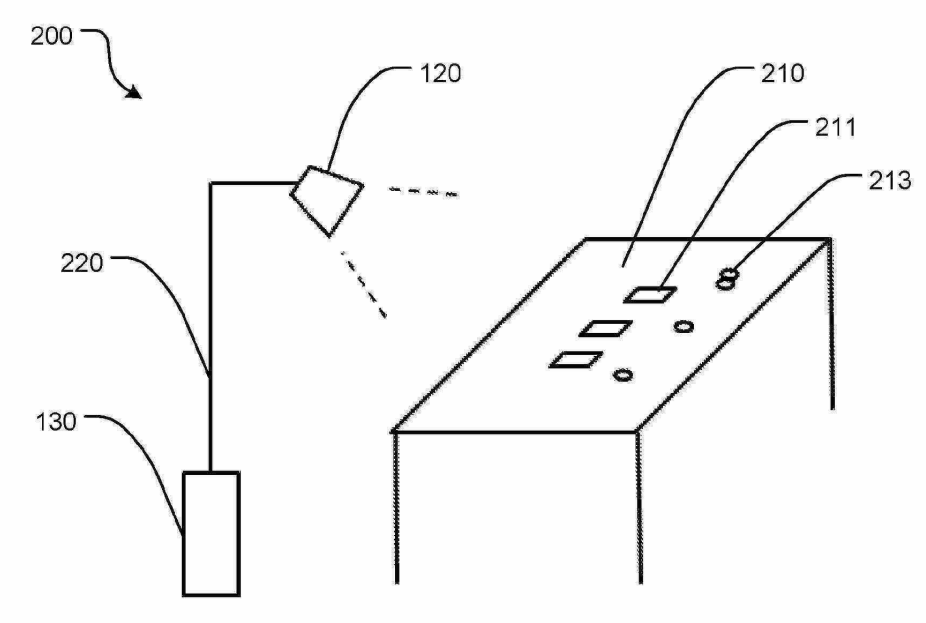
[0193] 당업자라면, 본 개시물의 넓은 범용 범위를 벗어나지 않고, 상술한 실시예들에 대해 다양한 변형들 및/또는 수정들이 이루어질 수 있음을 이해할 것이다. 따라서, 본 실시예들은 모든 면에서 예시적인 것으로 그리고 제한적이지 않은 것으로 간주되어야 한다.

도면

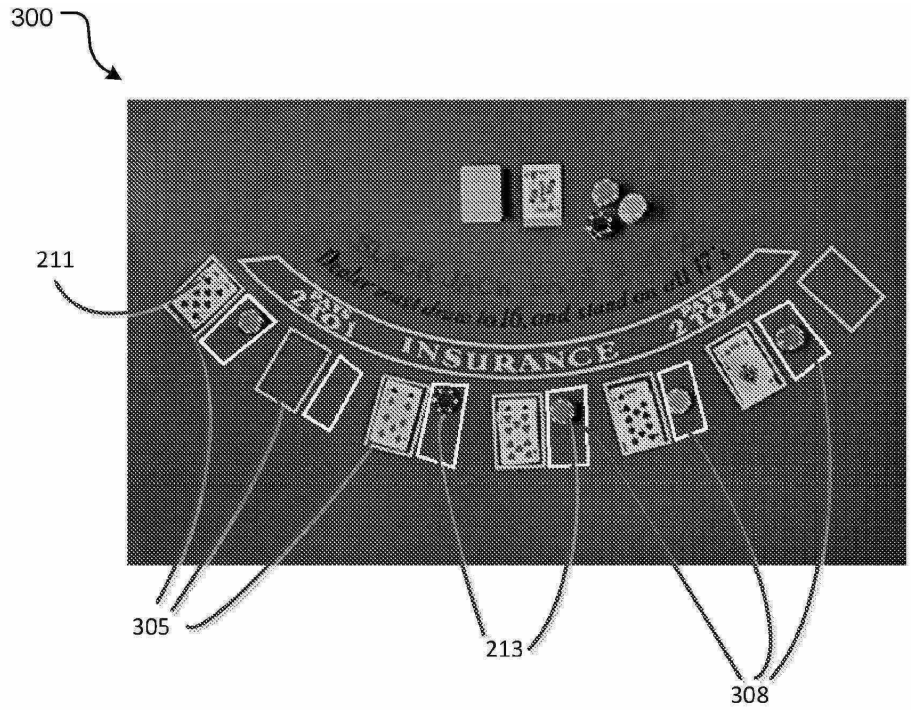
도면1



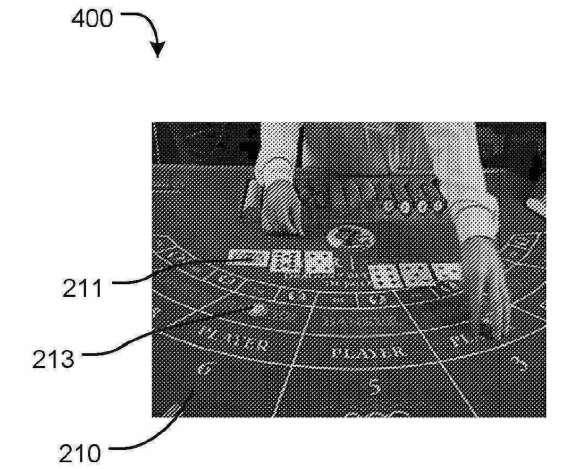
도면2



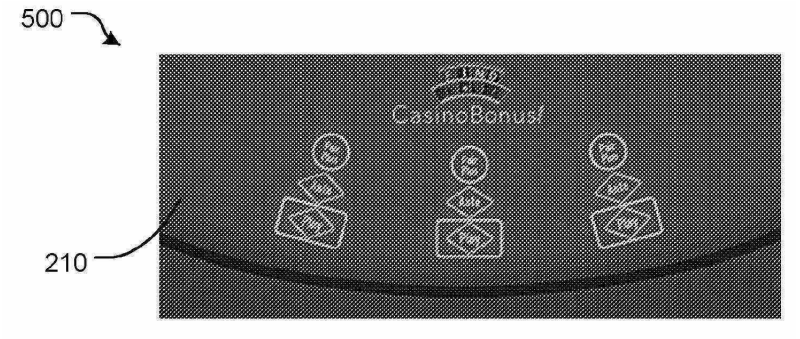
도면3



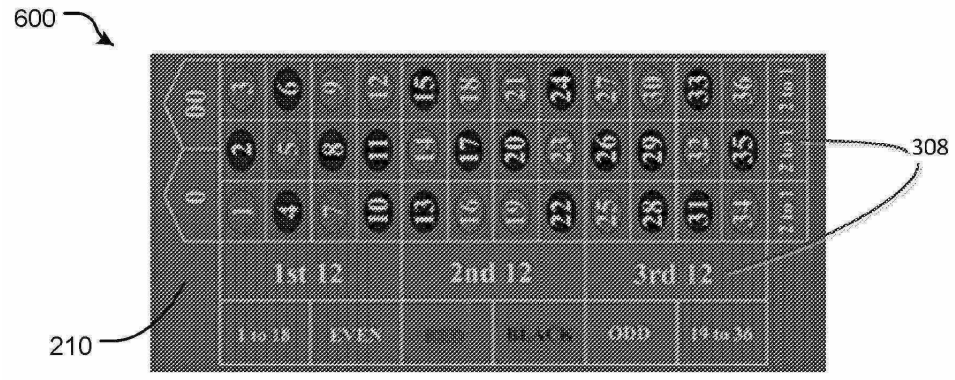
도면4



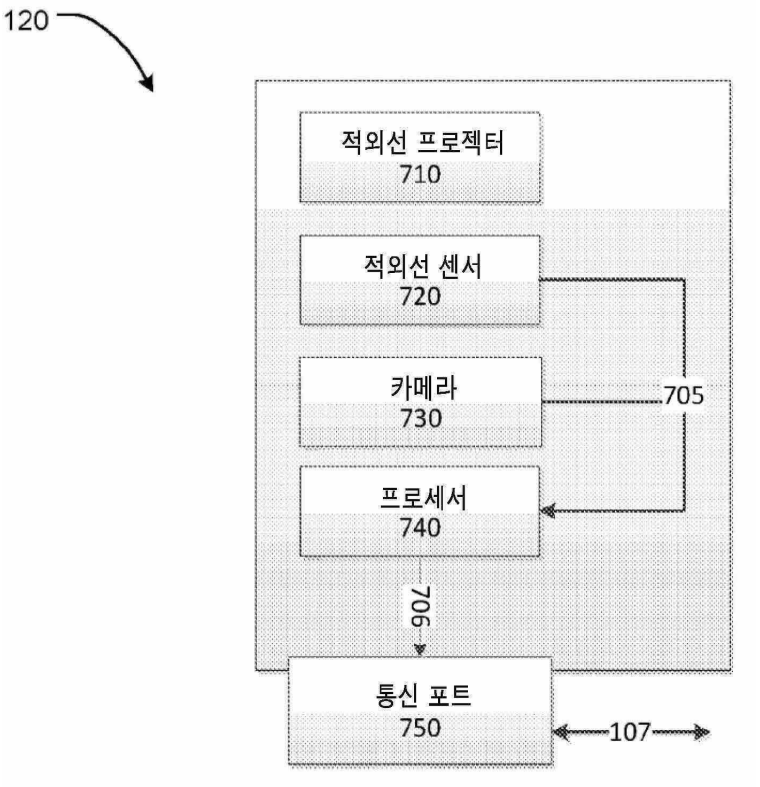
도면5



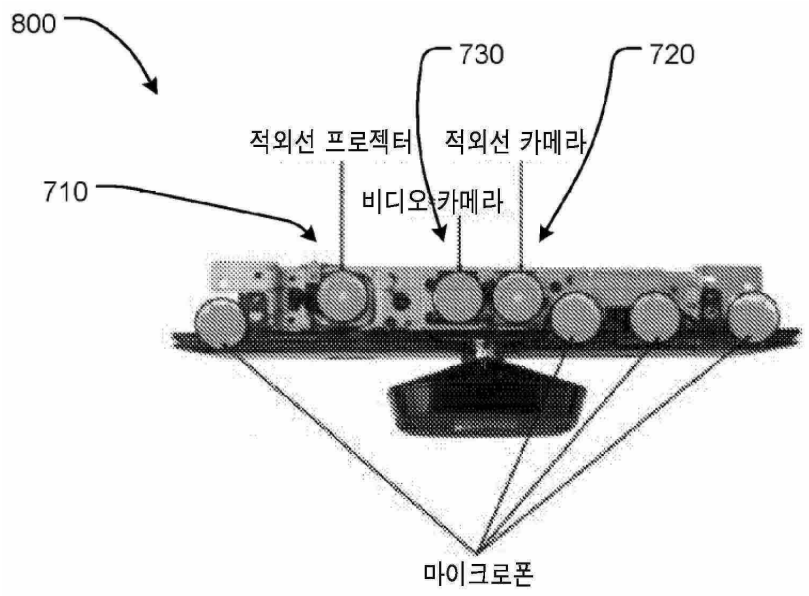
도면6



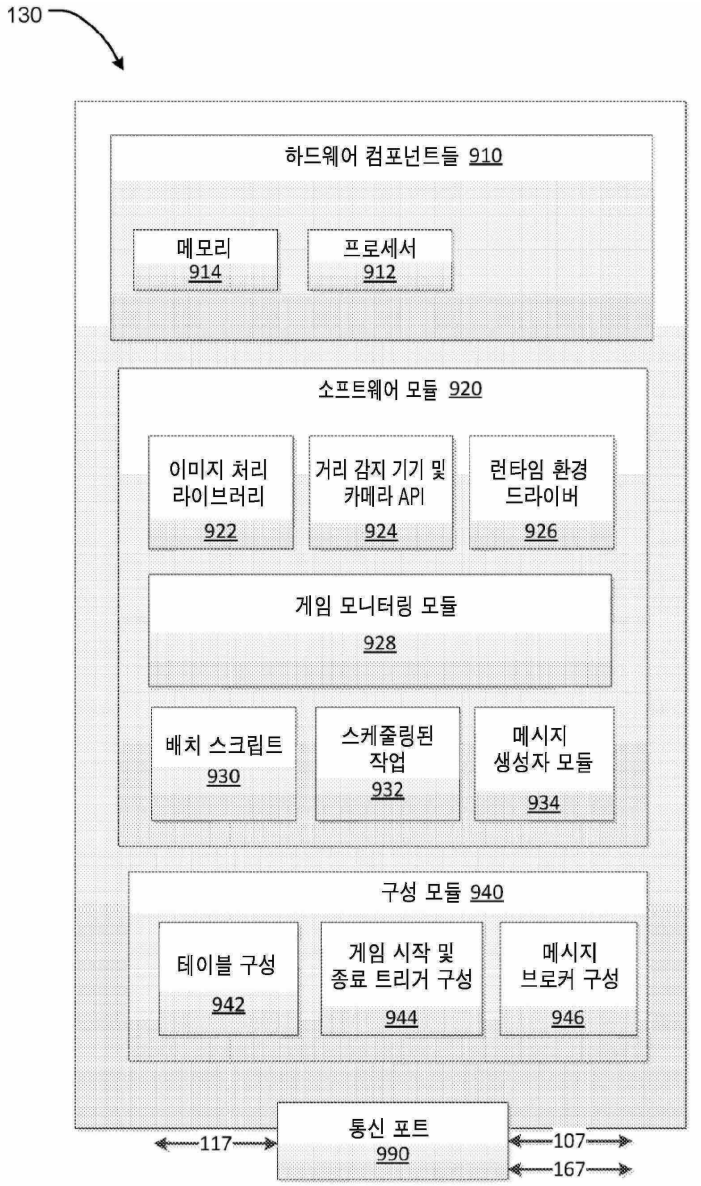
도면7


120
적외선 프로젝터
710
적외선 센서
720
카메라
730
705
프로세서
740
706
통신 포트
750
107

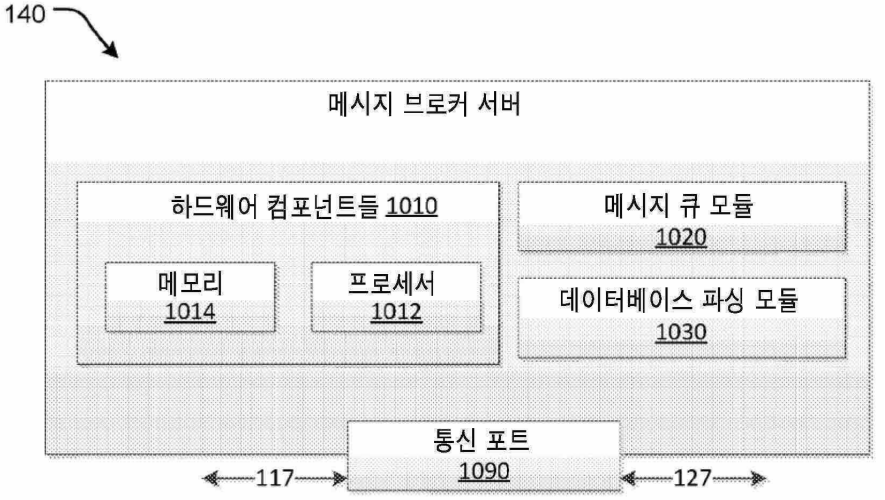
도면8


800
730
720
적외선 프로젝터
적외선 카메라
710
비디오 카메라
마이크로폰

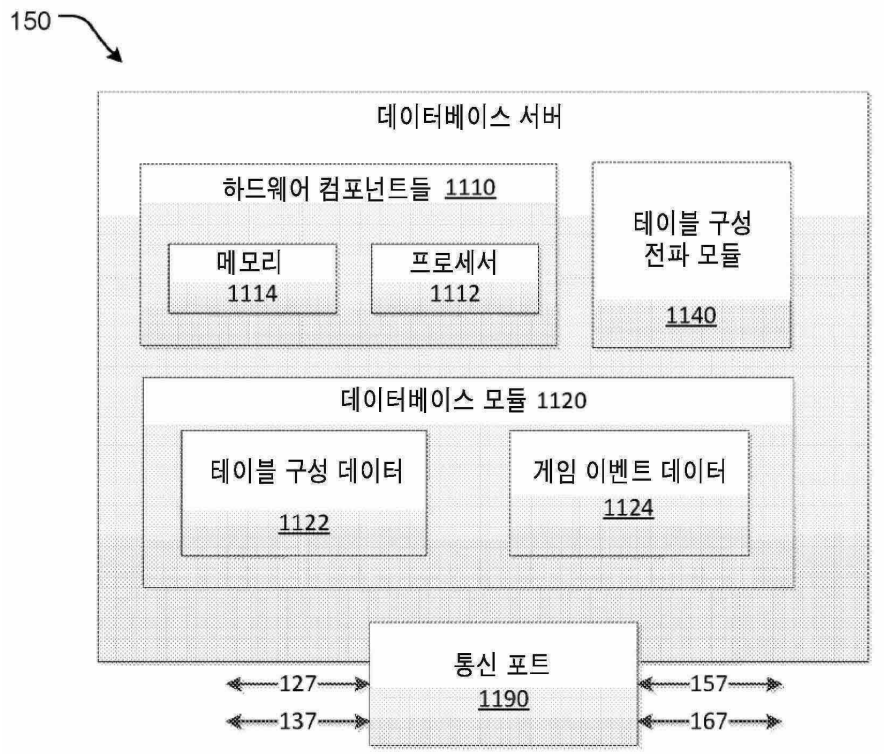
도면9


130
하드웨어 컴포넌트들 910
메모리
914
프로세서
912
소프트웨어 모듈 920
이미지 처리
라이브러리
922
거리 감지 기기 및
카메라 API
924
런타임 환경
드라이버
926
게임 모니터링 모듈
928
배치 스크립트
930
스케줄링된
작업
932
메시지
생성자 모듈
934
구성 모듈 940
테이블 구성
942
게임 시작 및
종료 트리거 구성
944
메시지
브로커 구성
946
통신 포트
990
117
107
167

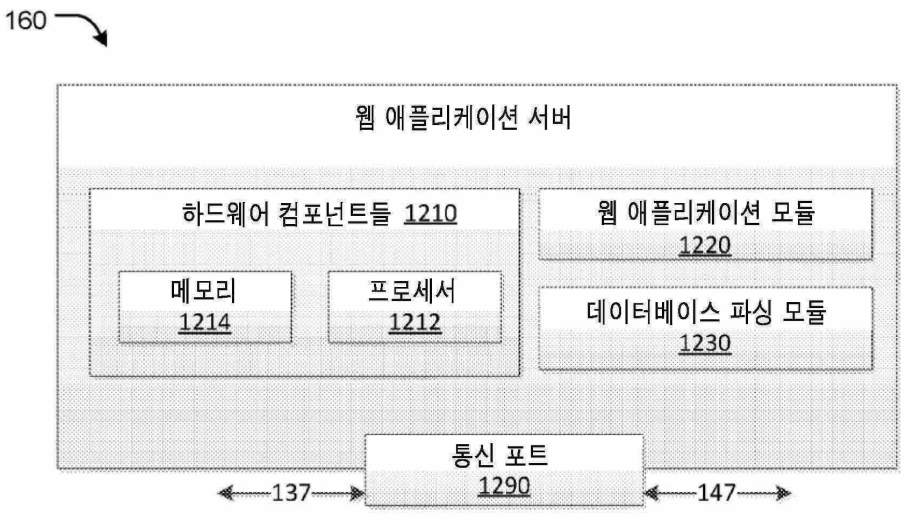
도면10


140
메시지 브로커 서버
하드웨어 컴포넌트들 1010
메모리
1014
프로세서
1012
메시지 큐 모듈
1020
데이터베이스 파싱 모듈
1030
통신 포트
1090
117
127

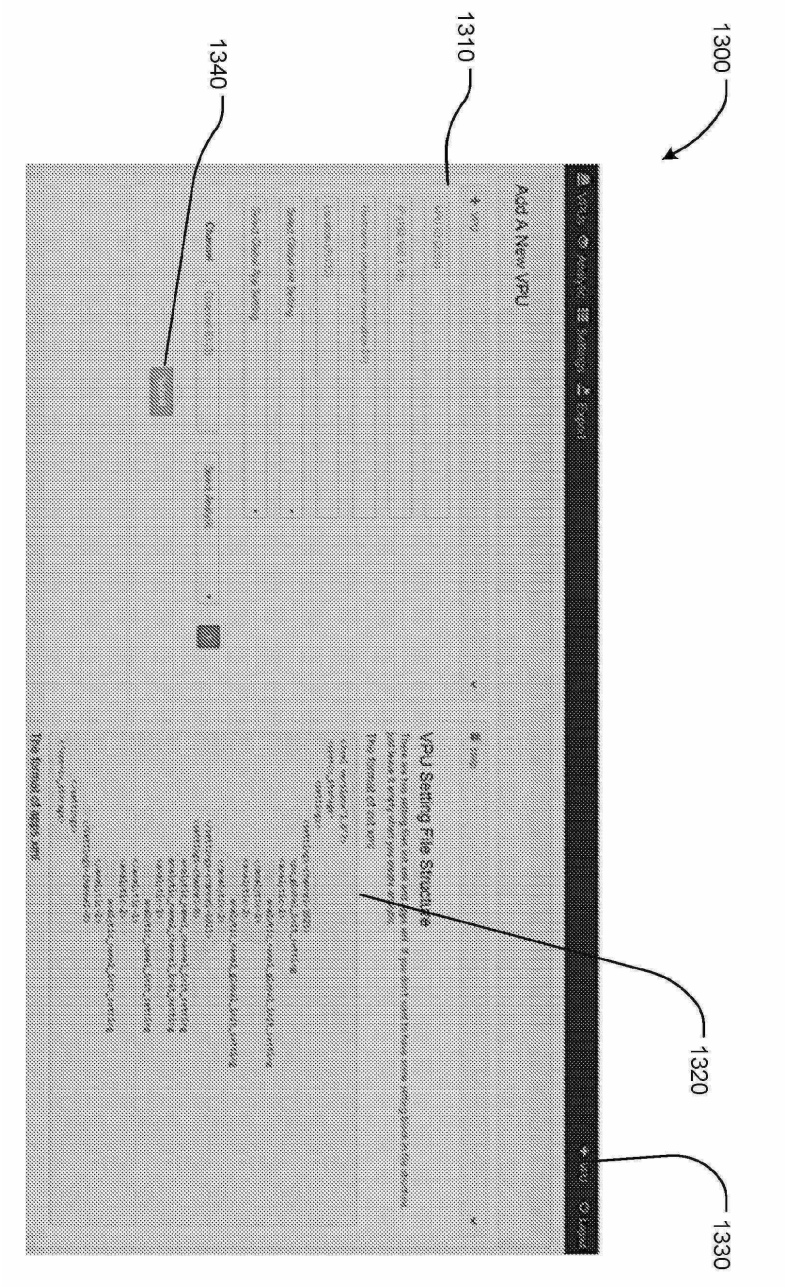
도면11


150
데이터베이스 서버
하드웨어 컴포넌트들 1110
메모리
1114
프로세서
1112
테이블 구성
전파 모듈
1140
데이터베이스 모듈 1120
테이블 구성 데이터
1122
게임 이벤트 데이터
1124
통신 포트
1190
127
137
157
167

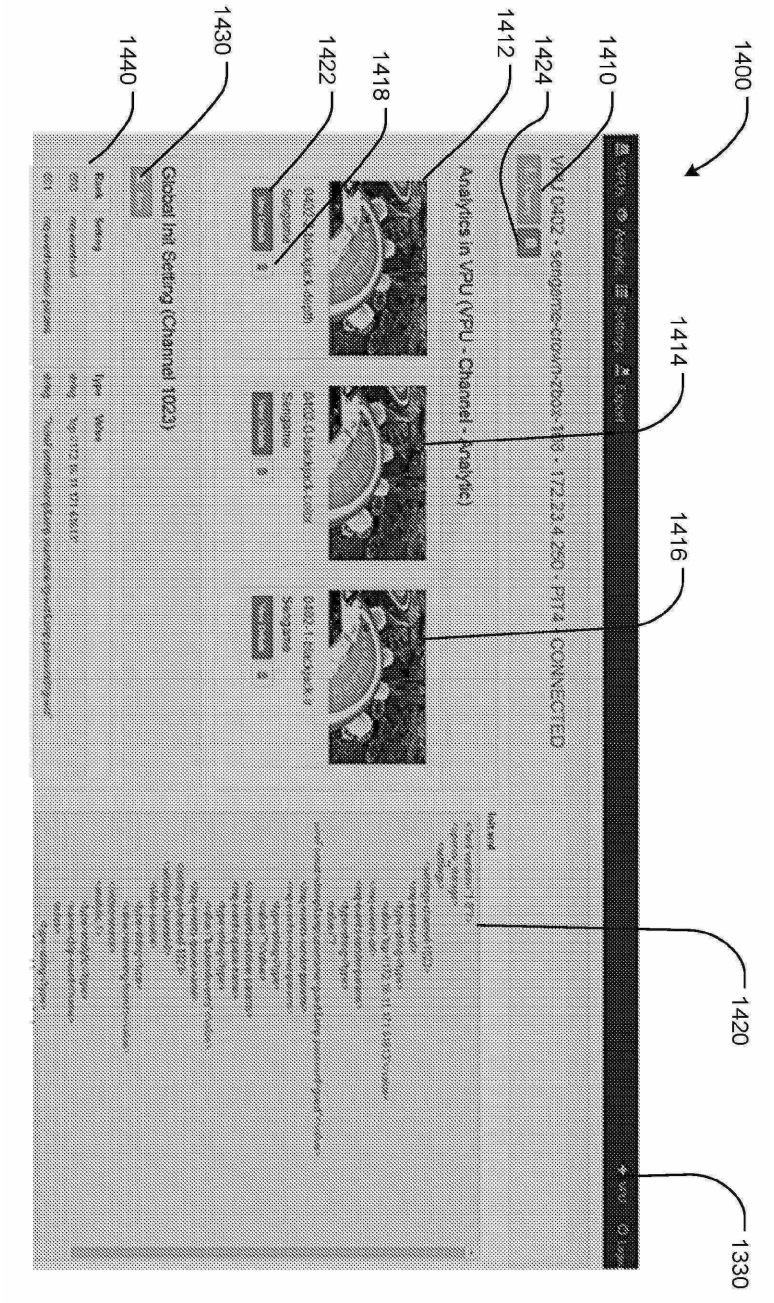
도면12


160
웹 애플리케이션 서버
하드웨어 컴포넌트들 1210
메모리
1214
프로세서
1212
웹 애플리케이션 모듈
1220
데이터베이스 파싱 모듈
1230
통신 포트
1290
137
147

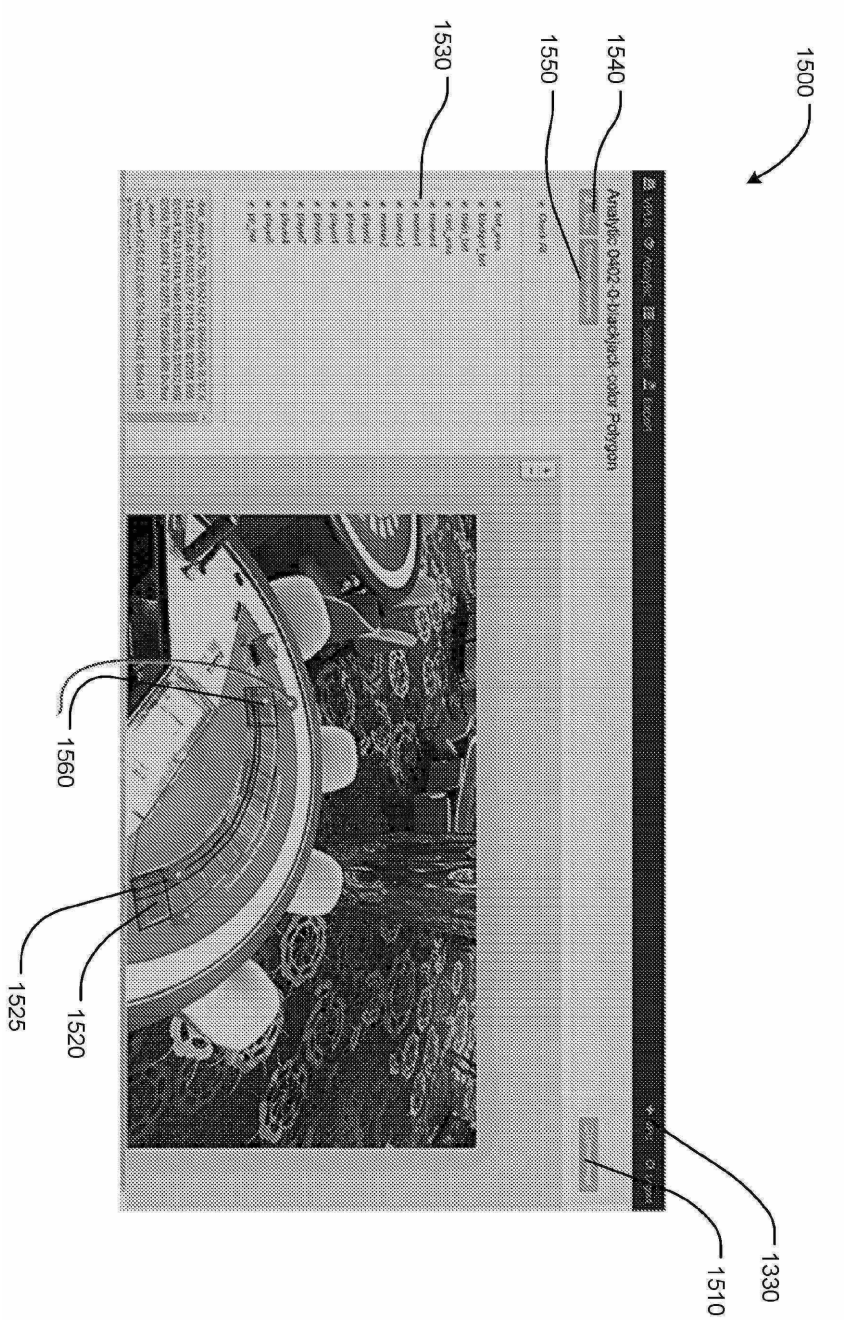
도면13


1300
1310
1340
1320
1330
Add A New VPU
VPU Setting File Structure

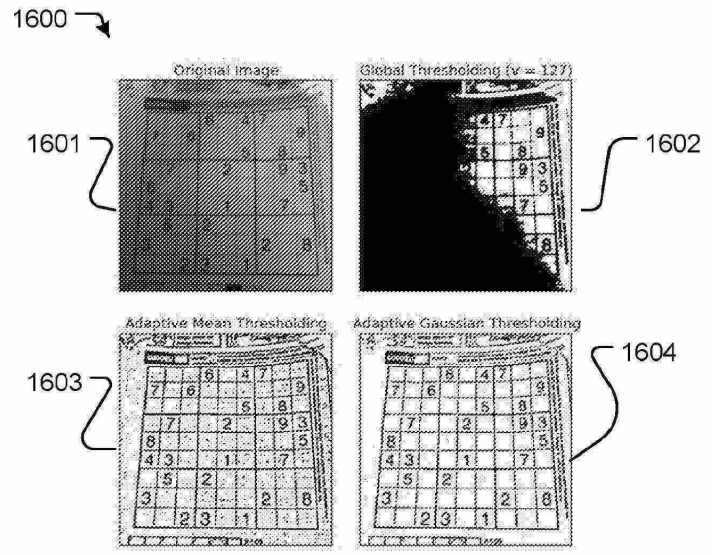
도면14


1400
1410
1412
1414
1416
1418
1420
1422
1424
1430
1440
1330
Analytics in VPU (VPU - Channel - Analyse)
Global Init Setting (Channel 1023)

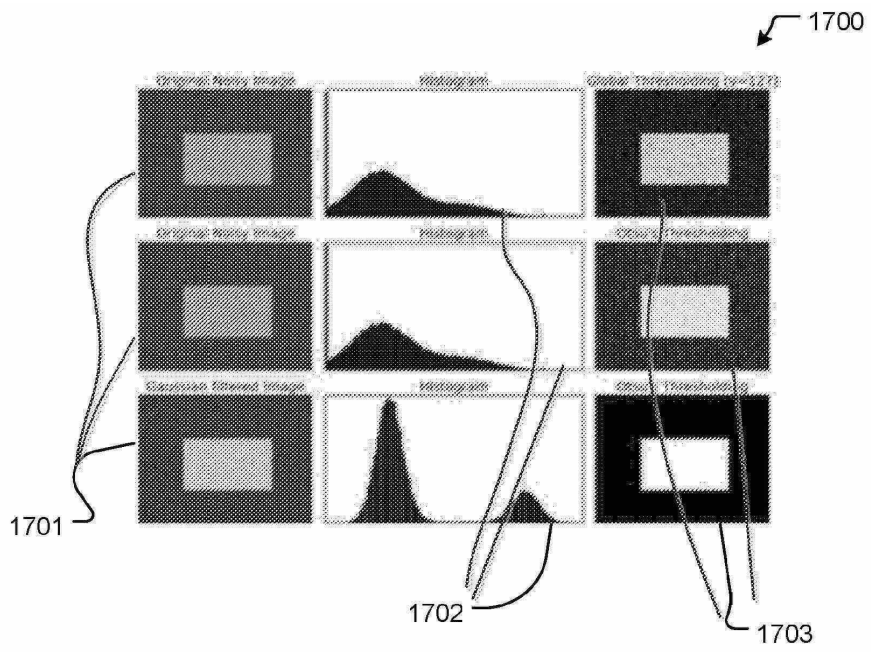
도면15



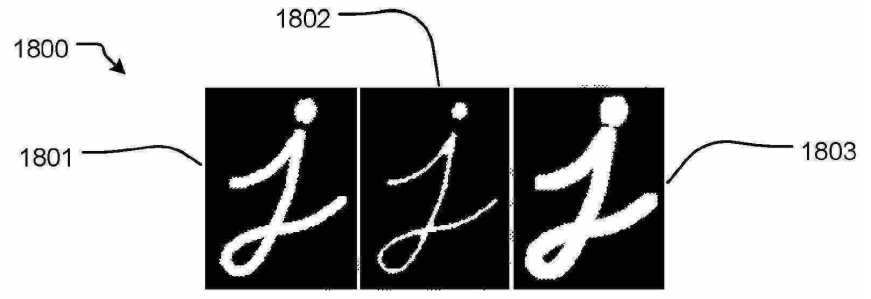
도면16



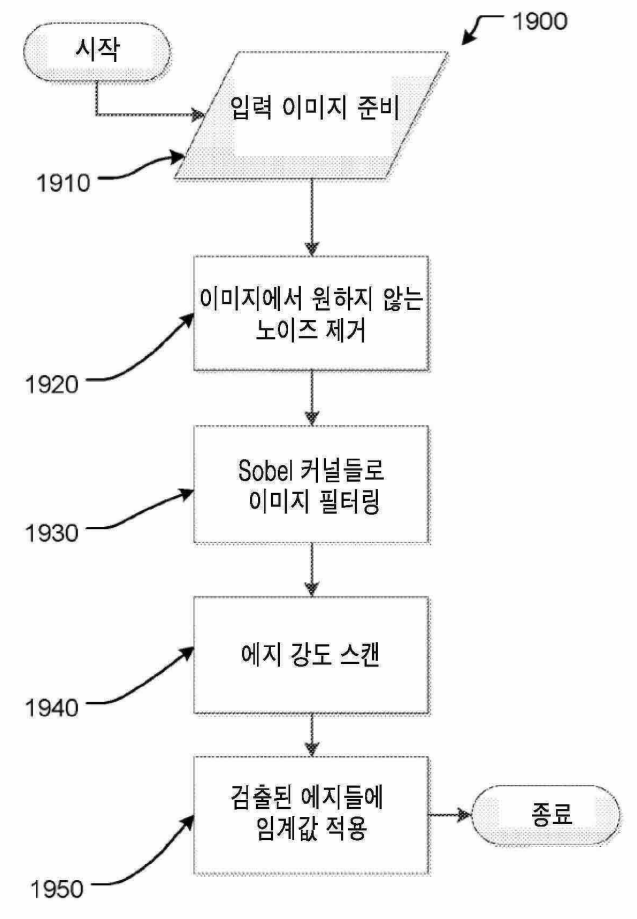
도면17



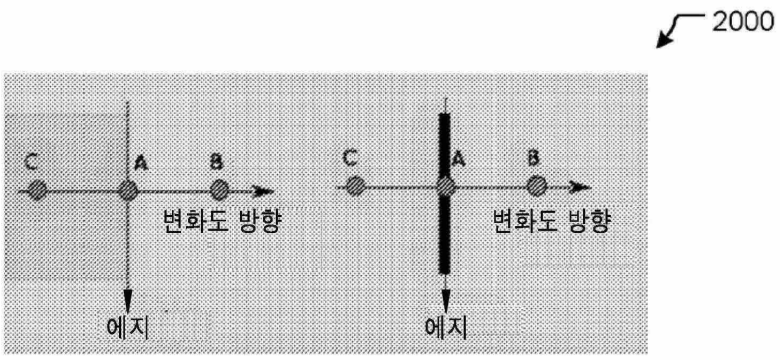
도면18



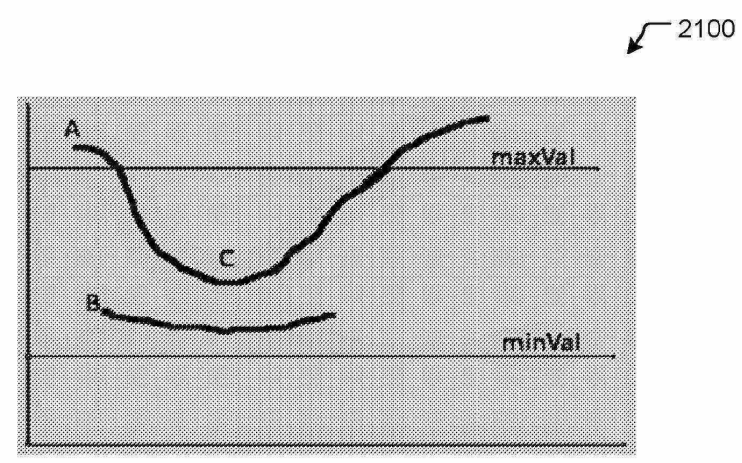
도면19


1900
시작
입력 이미지 준비
1910
이미지에서 원하지 않는
노이즈 제거
1920
Sobel 커널들로
이미지 필터링
1930
에지 강도 스캔
1940
검출된 에지들에
임계값 적용
1950
종료

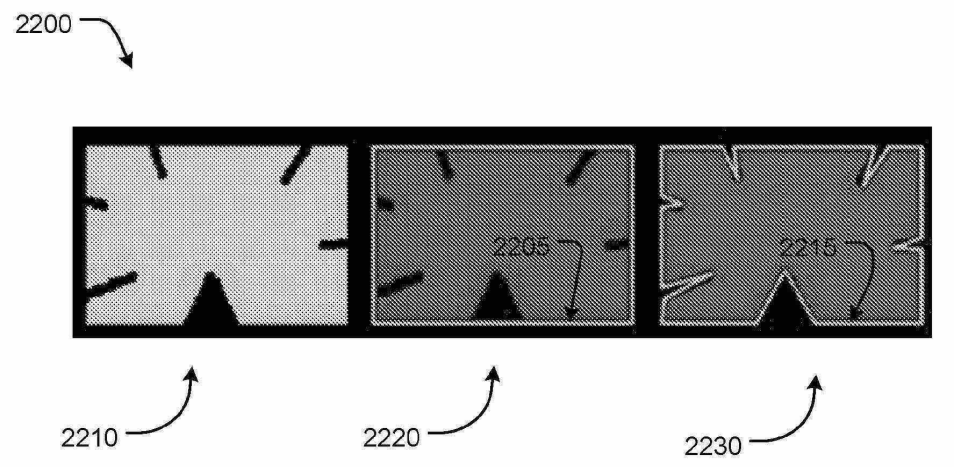
도면20


2000
C
A
B
변화도 방향
에지
C
A
B
변화도 방향
에지

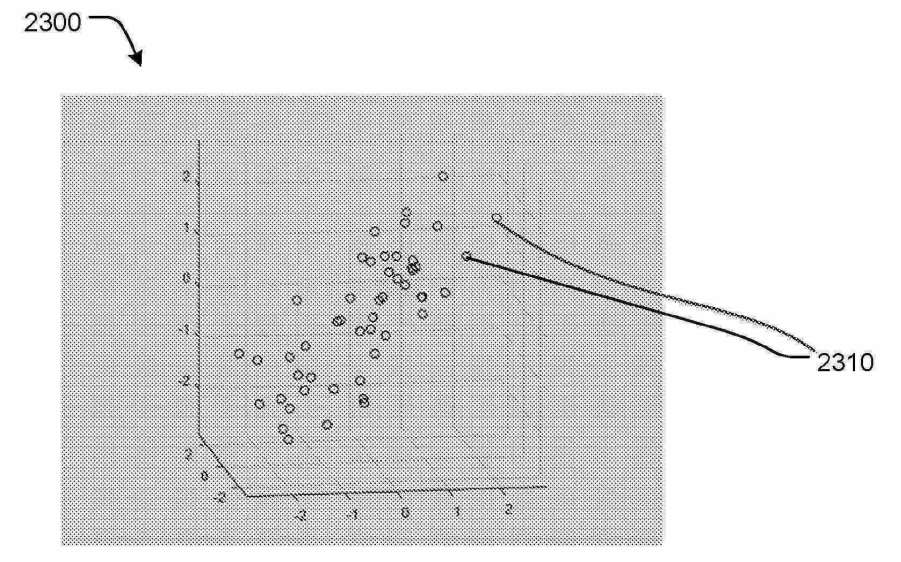
도면21



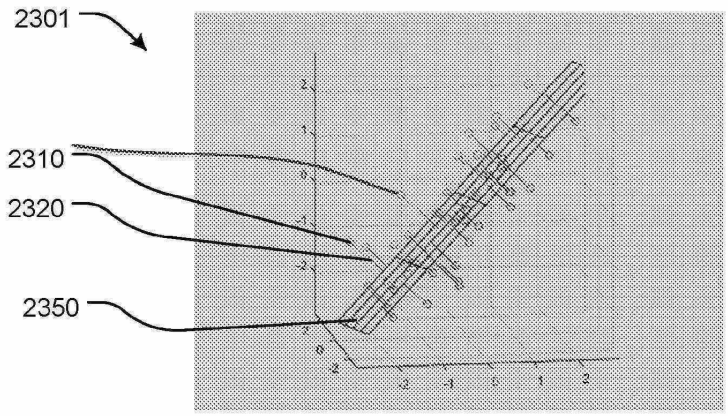
도면22



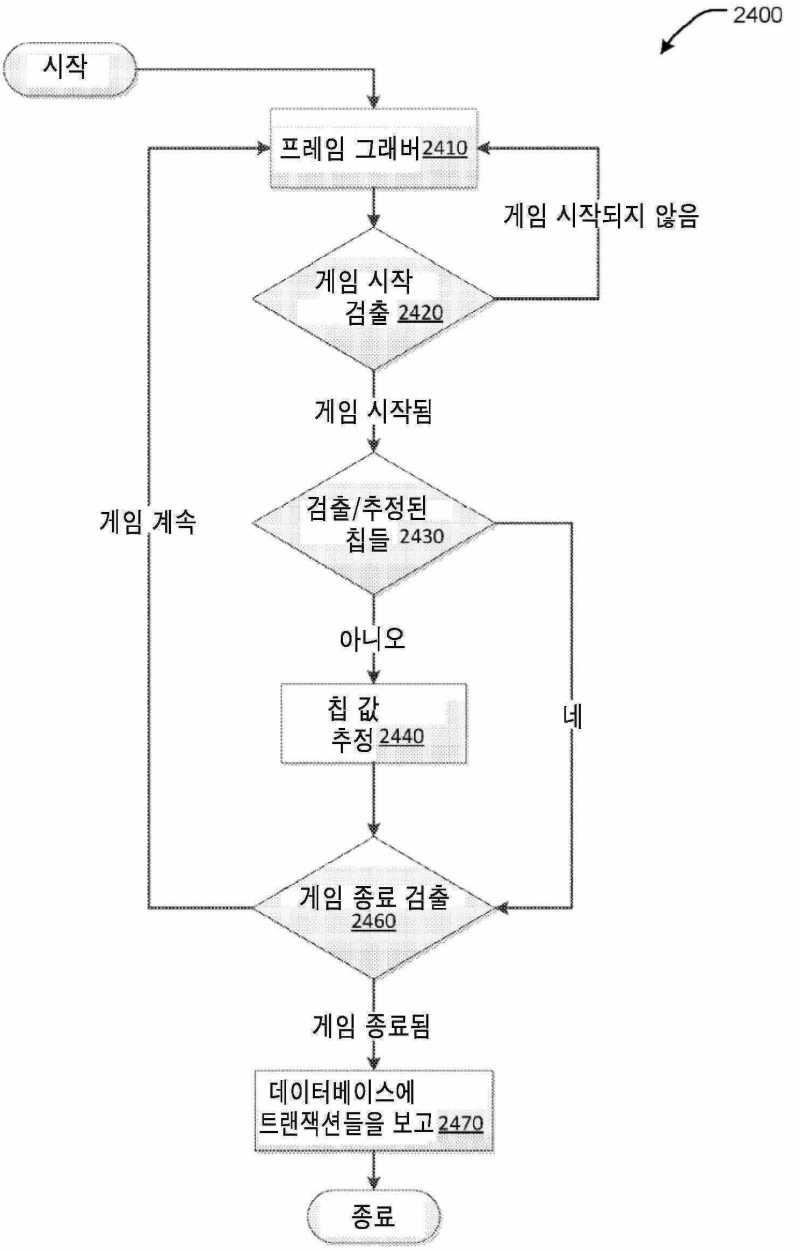
도면23a



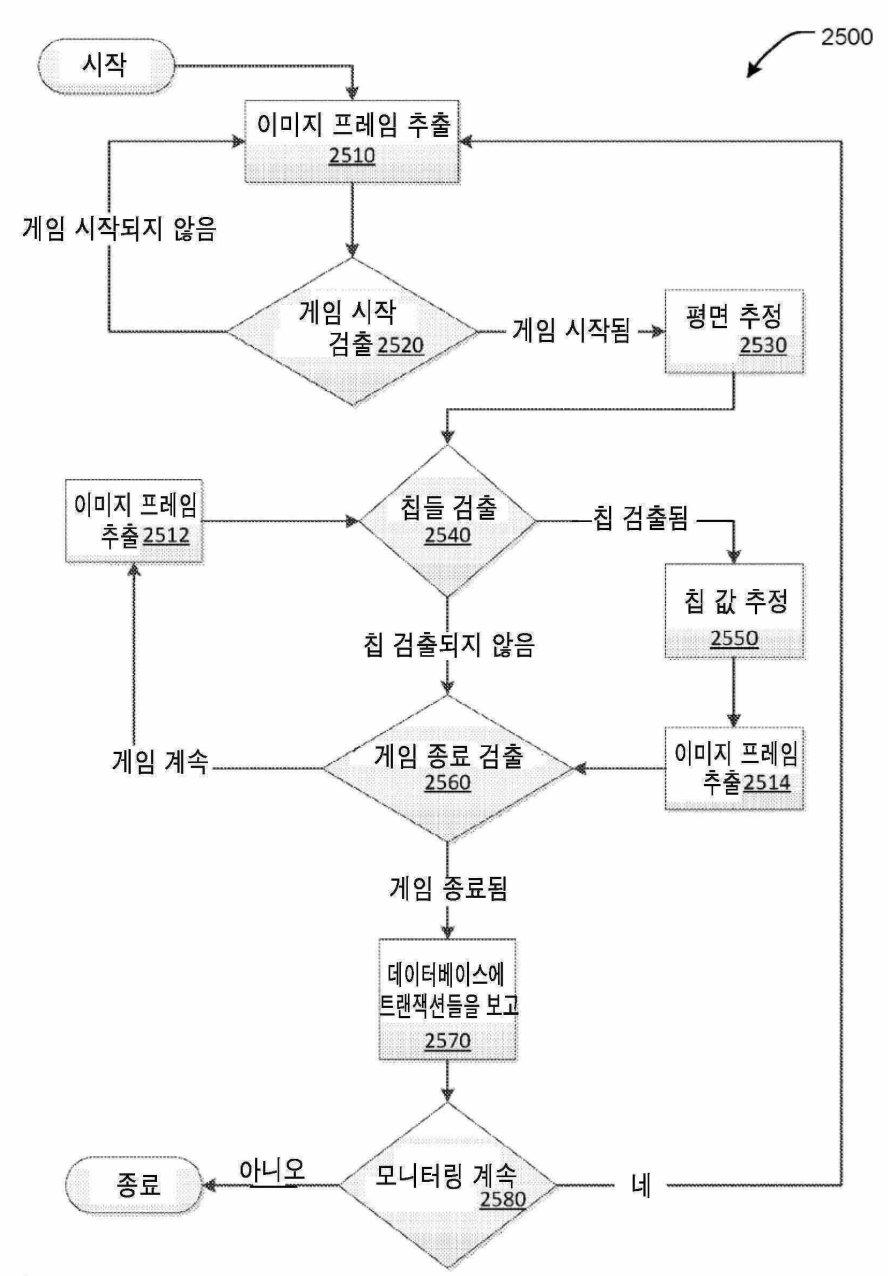
도면23b


2301
2310
2320
2350

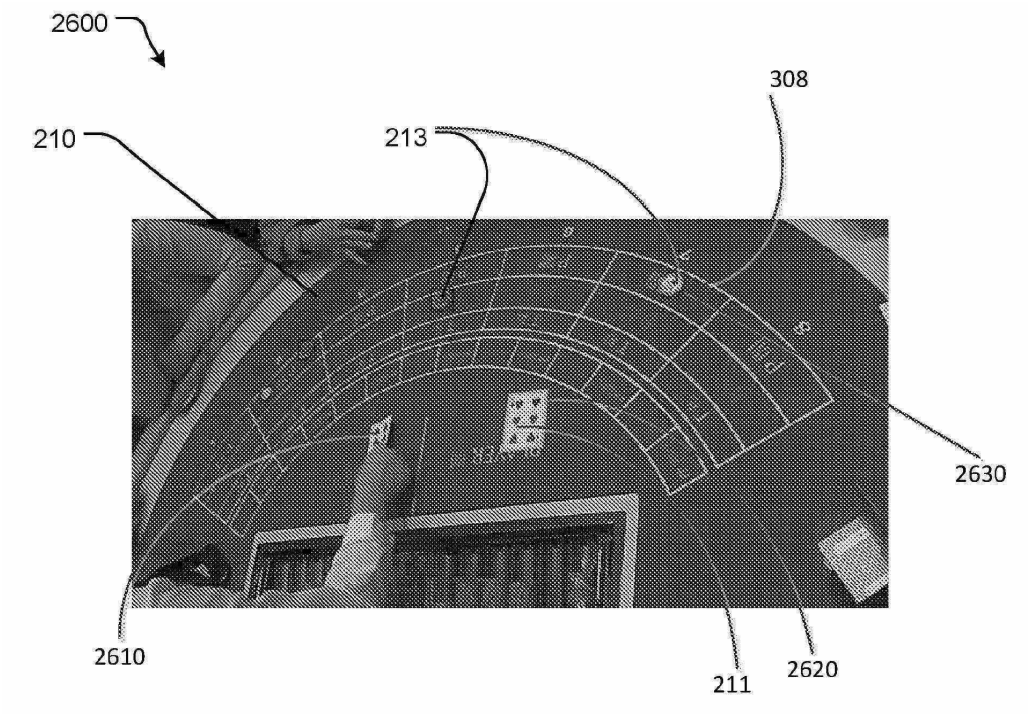
도면24


2400
시작
프레임 그래버 2410
게임 시작되지 않음
게임 시작 검출 2420
게임 시작됨
검출/추정된 칩들 2430
게임 계속
아니오
네
칩 값 추정 2440
게임 종료 검출 2460
게임 종료됨
데이터베이스에 트랜잭션들을 보고 2470
종료

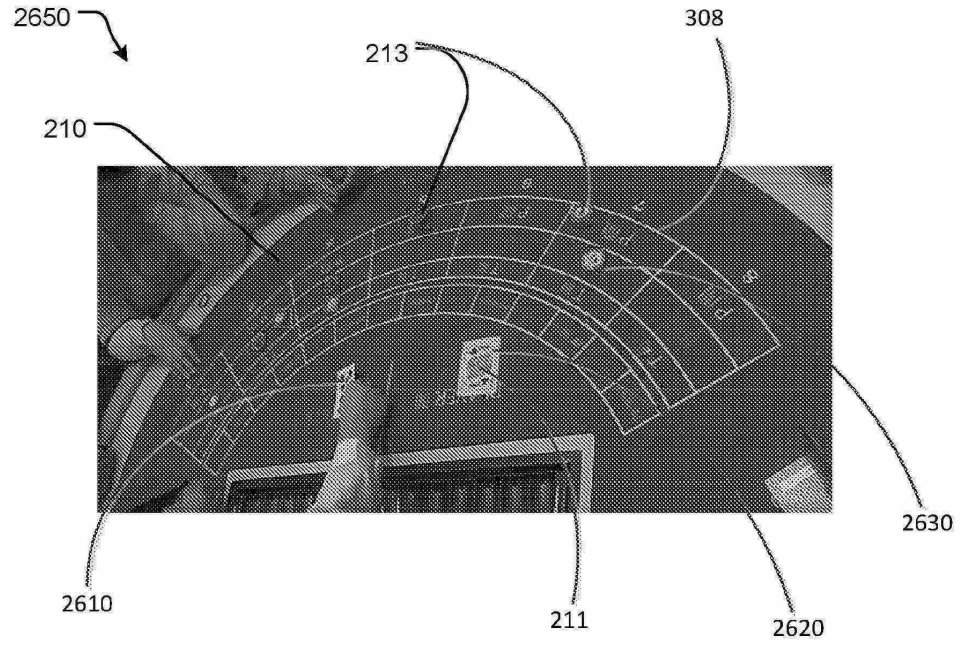
도면25


2500
시작
이미지 프레임 추출
2510
게임 시작되지 않음
게임 시작
검출 2520
게임 시작됨
평면 추정
2530
이미지 프레임
추출 2512
칩들 검출
2540
칩 검출됨
칩 값 추정
2550
칩 검출되지 않음
게임 계속
게임 종료 검출
2560
이미지 프레임
추출 2514
게임 종료됨
데이터베이스에
트랜잭션들을 보고
2570
종료
아니오
모니터링 계속
2580
네

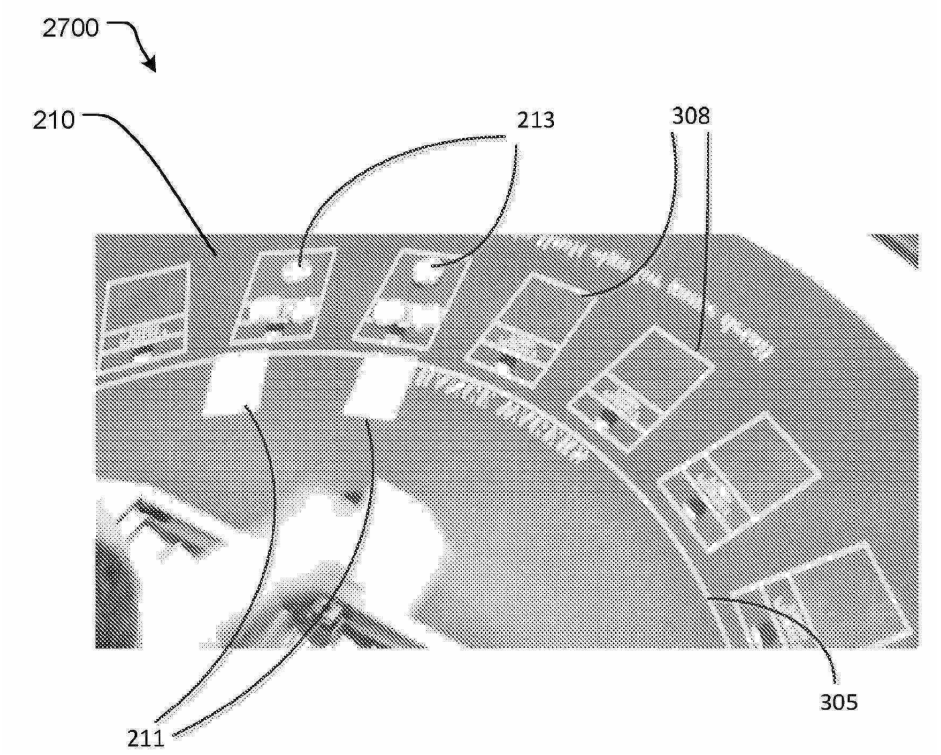
도면26a


2600
308
210
213
2630
2610
211
2620

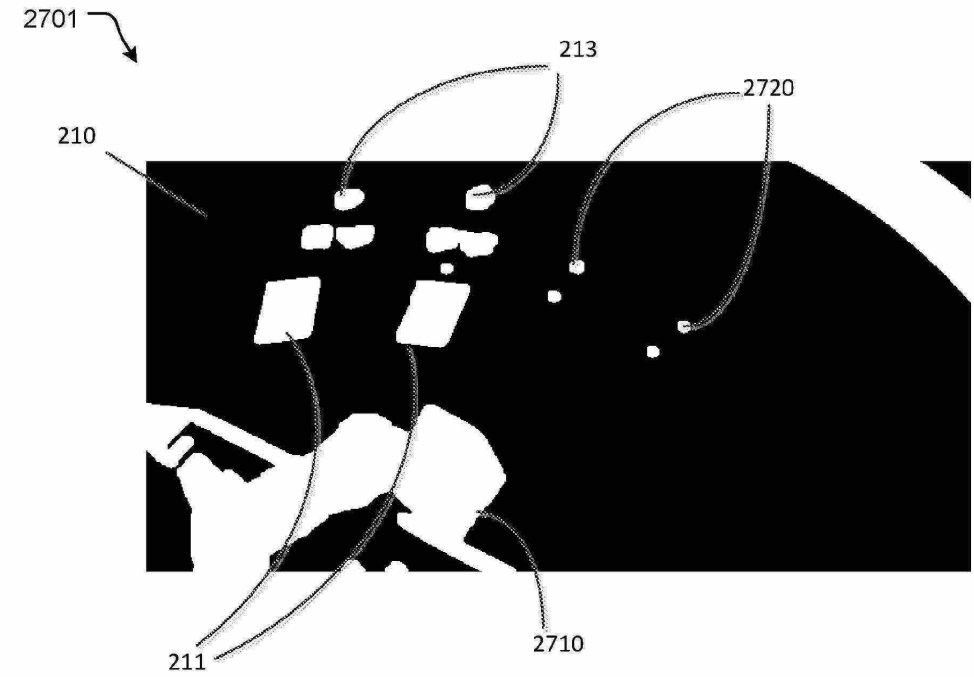
도면26b


2650
213
308
210
2630
2610
211
2620

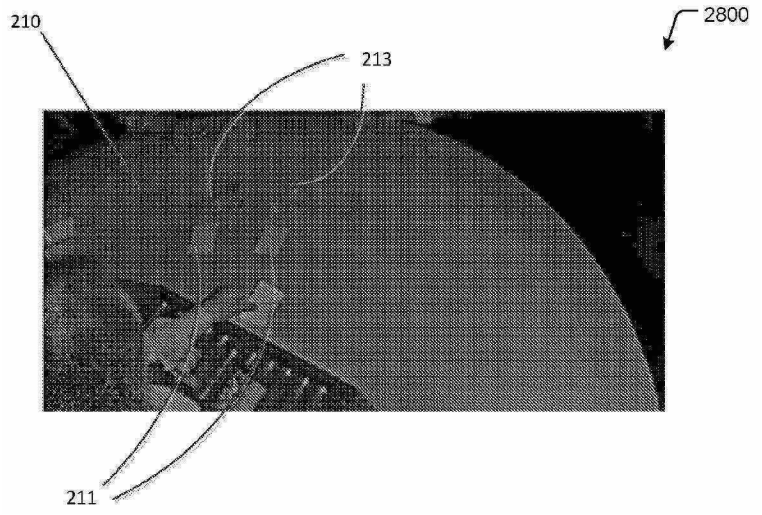
도면27a



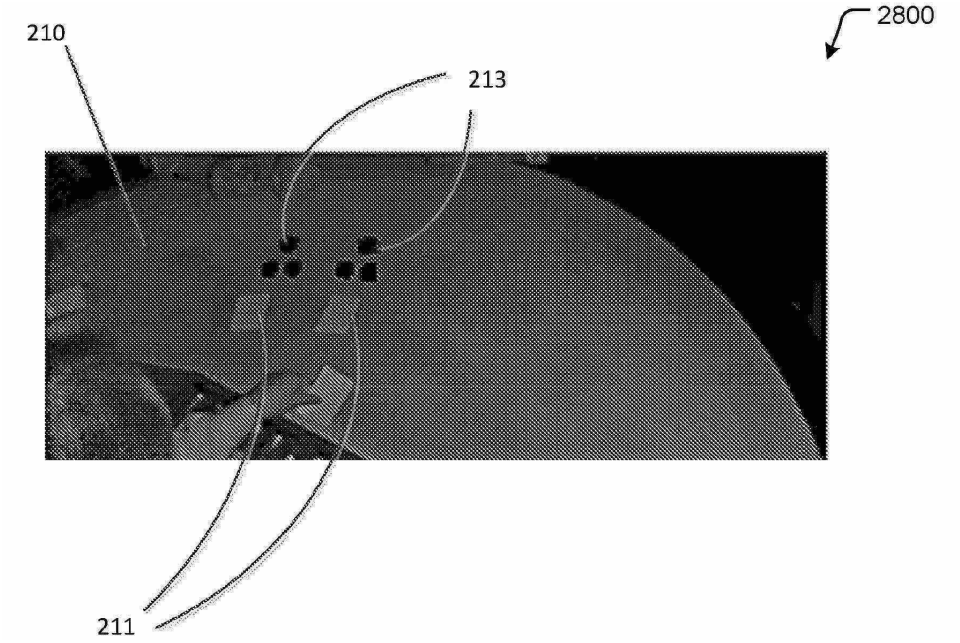
도면27b



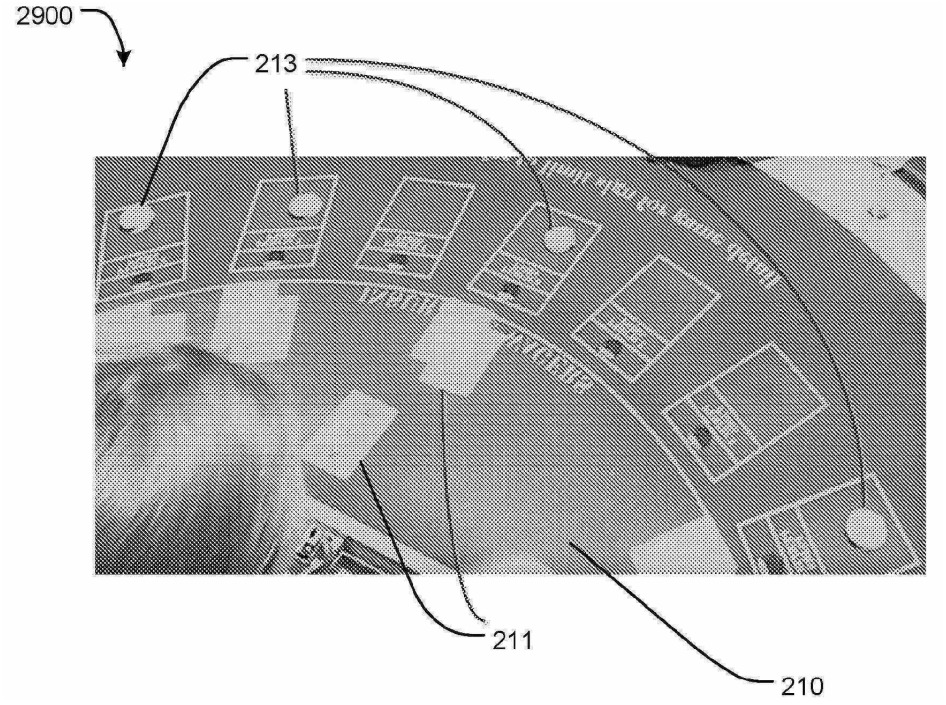
도면28a



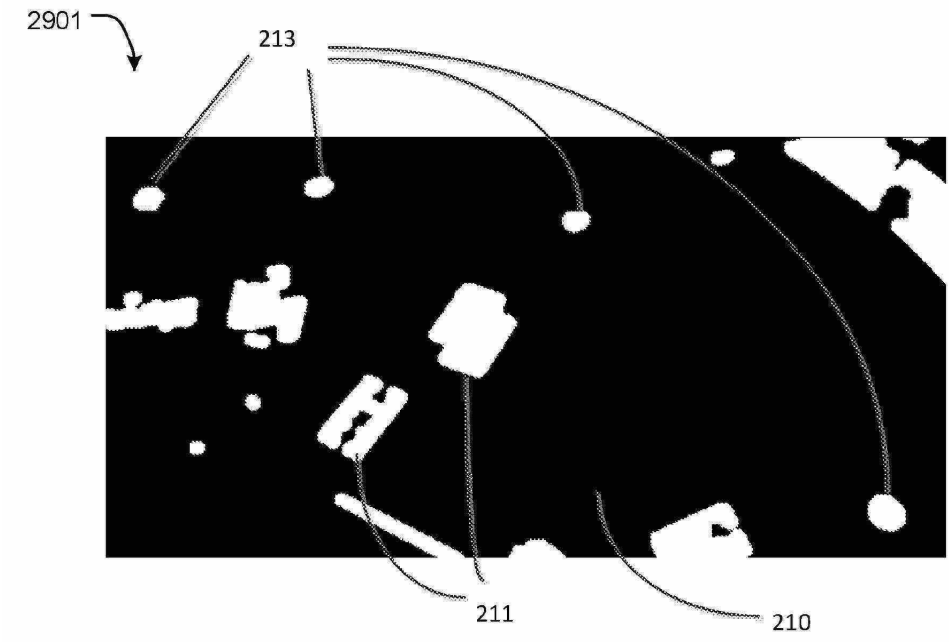
도면28b



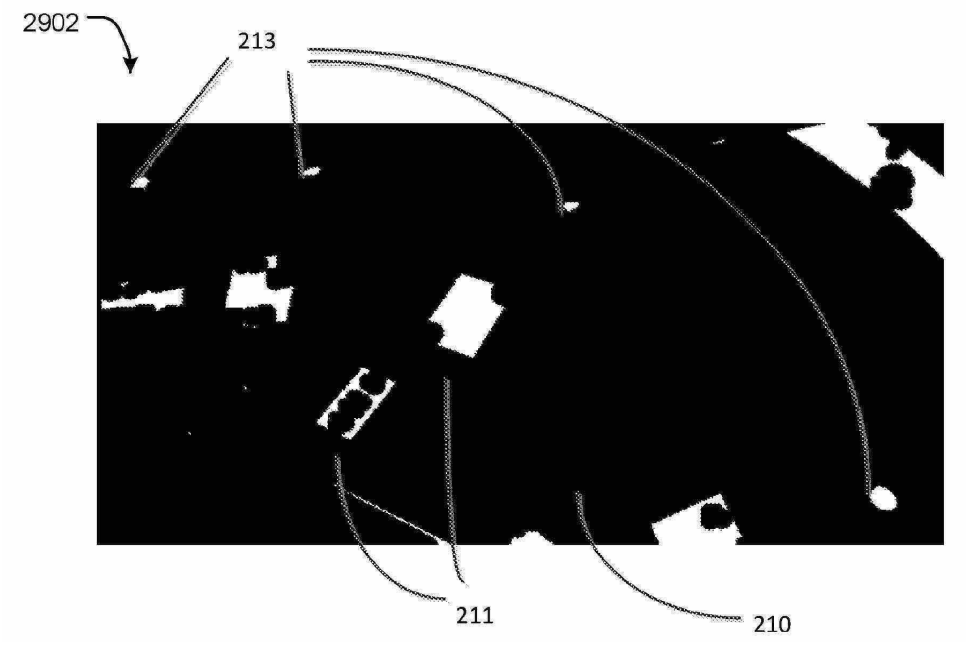
도면29a



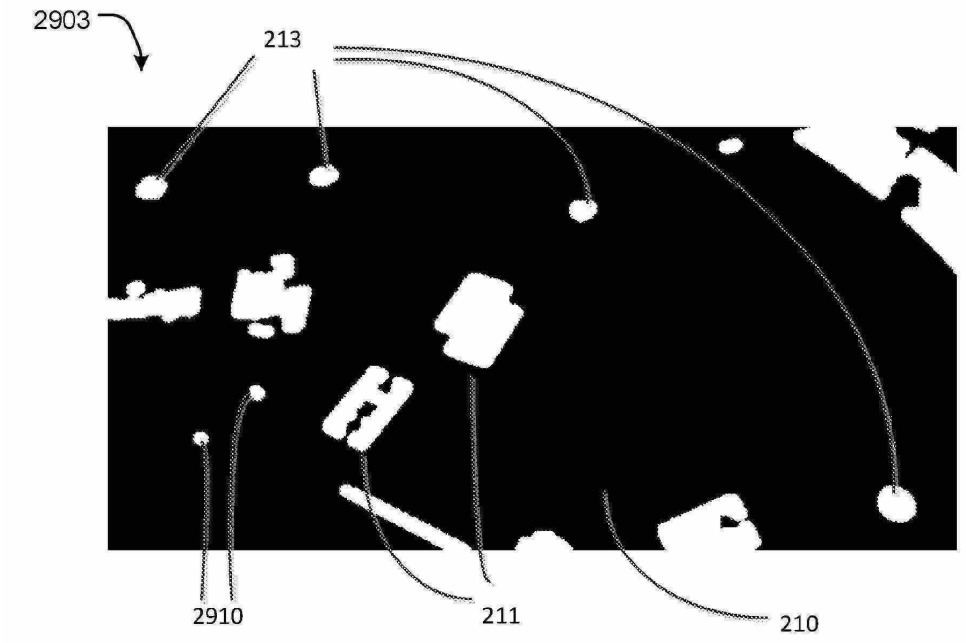
도면29b



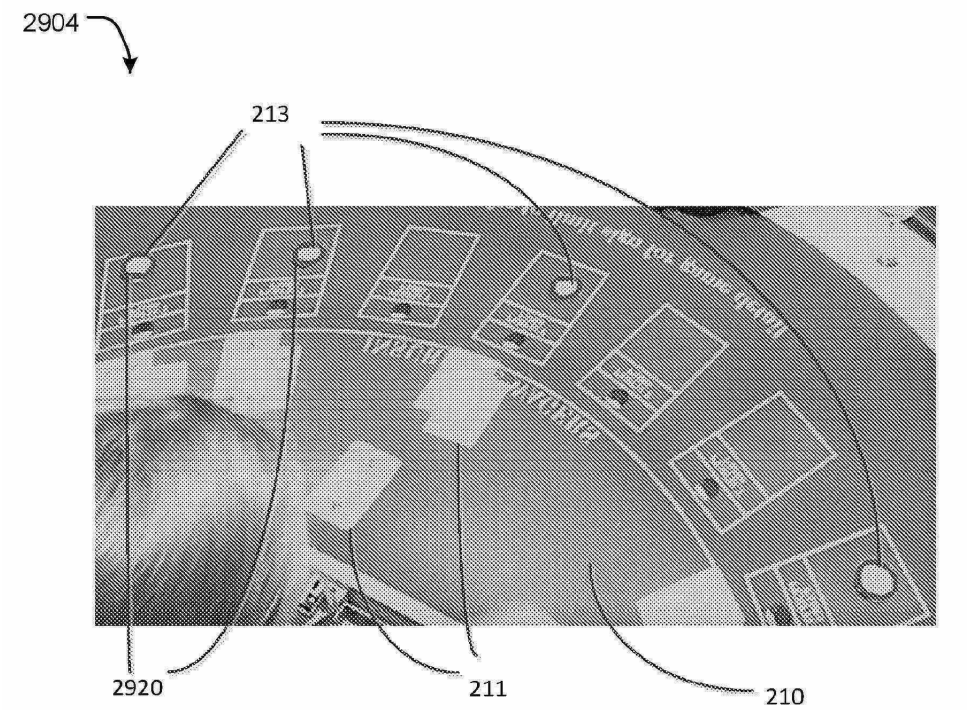
도면29c



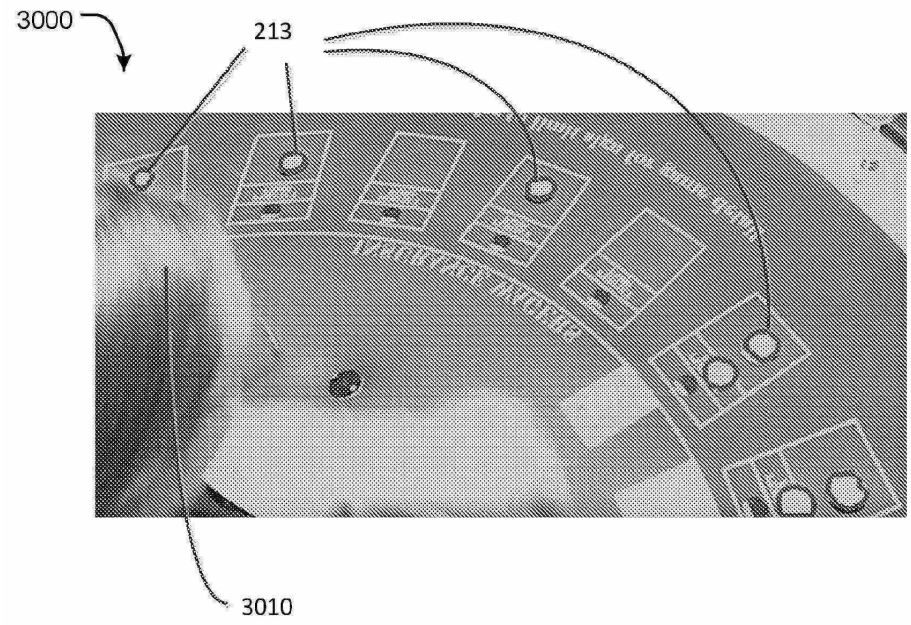
도면29d



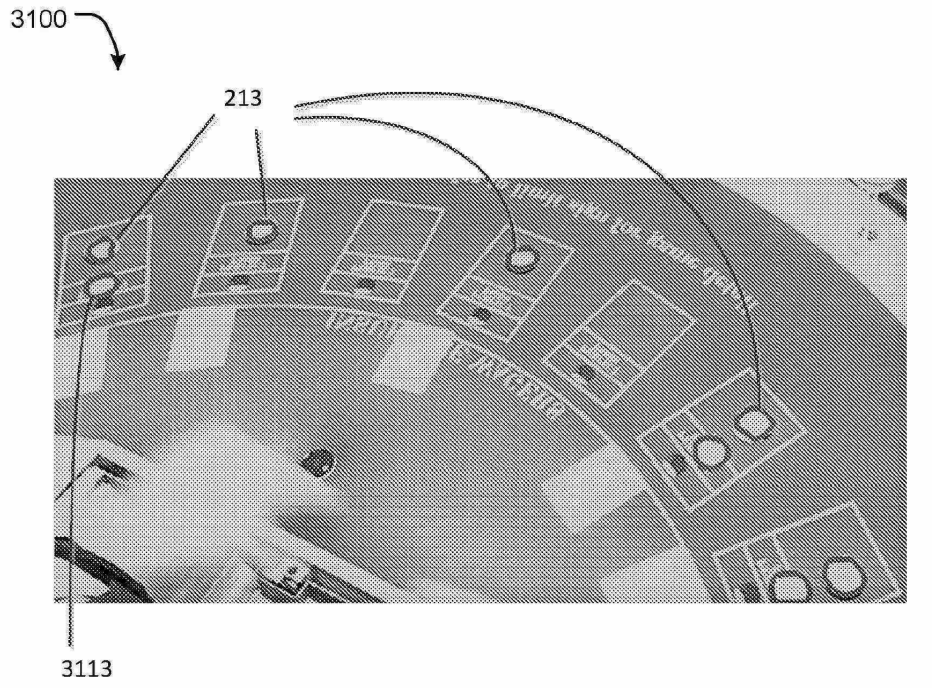
도면29e



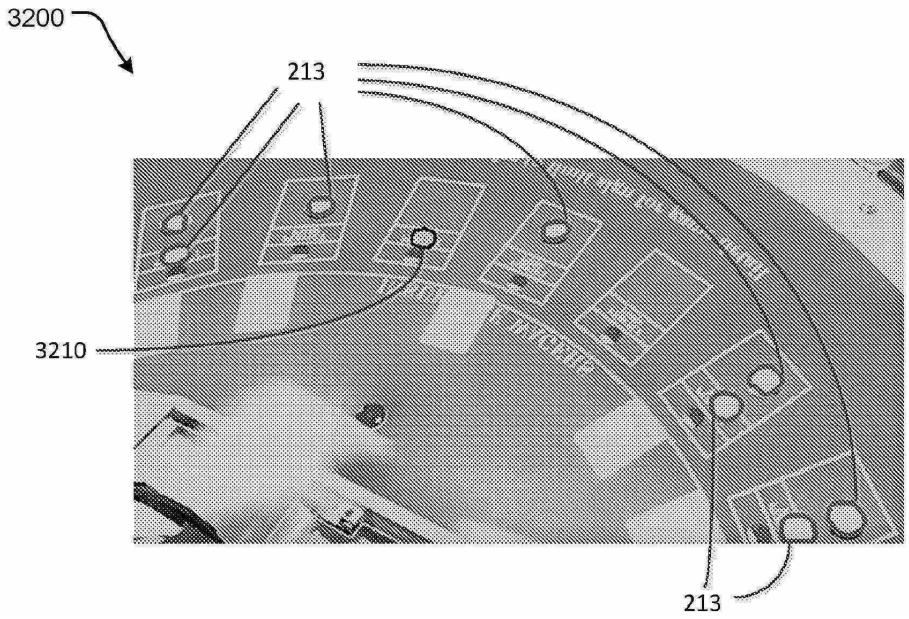
도면30



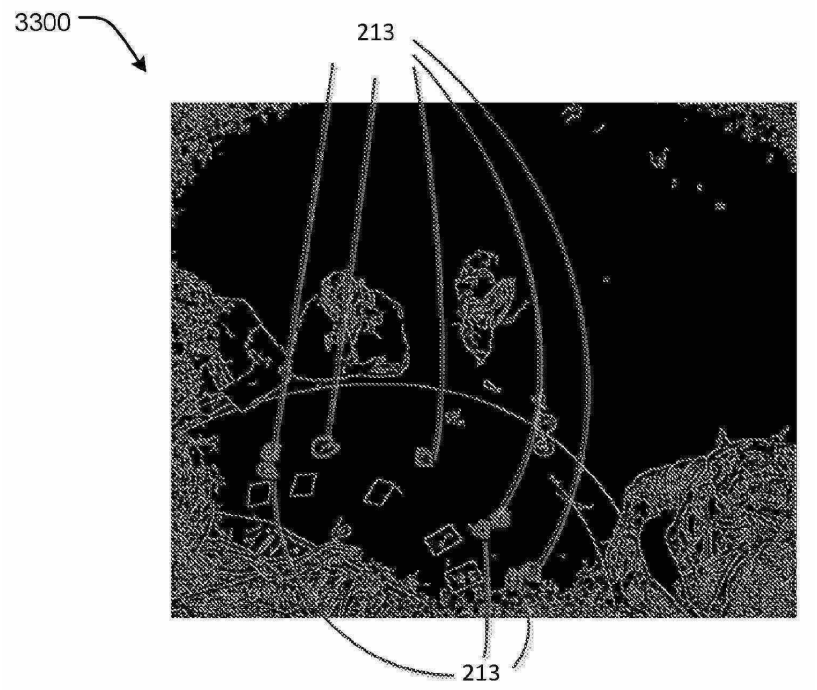
도면31



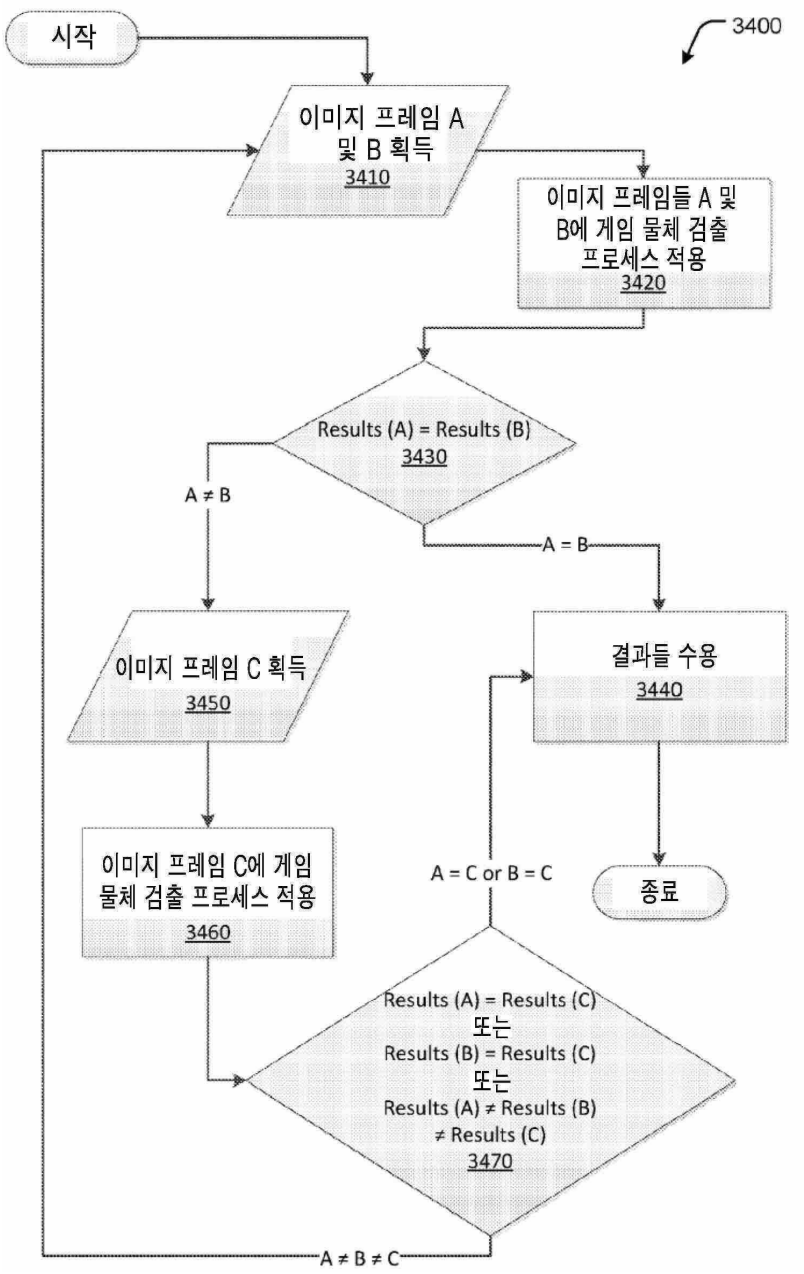
도면32



도면33



도면34


시작
3400
이미지 프레임 A 및 B 획득
3410
이미지 프레임들 A 및 B에 게임 물체 검출 프로세스 적용
3420
Results (A) = Results (B)
3430
A ≠ B
A = B
이미지 프레임 C 획득
3450
결과들 수용
3440
이미지 프레임 C에 게임 물체 검출 프로세스 적용
3460
A = C or B = C
종료
Results (A) = Results (C)
또는
Results (B) = Results (C)
또는
Results (A) ≠ Results (B)
≠ Results (C)
3470
A ≠ B ≠ C