[논문작성시 유의사항]

- 최종 논문은 출판사에 의해 재 편집되므로 아래의 기준에 맞춰 1단으로 편집하여 제출한다.
- 심사논문제출 시에는 저자이름 및 저자소개는 생략하고 심사 후 최종논문 제출시에 추가한다.
- 논문은 반드시 MS-WORD로 제출되어야 한다.
- 수식은 Microsoft Equation Editor 또는 MathType 을 사용하여 작성한다.
- 줄 간격은 모두 1.5줄 로 하며 폰트는 모두 휴먼명조로 작성, 영문논문일 경우 Times New Roman 으로 작성한다.
- 참고문헌 목록은 모두 영어로 작성한다.

AR을 이용한 당구 학습 시스템

Augmented Reality-based Billiards Training System

아래의 성명 및 소속은 심사논문 제출시 생략하고 최종논문 제출시 추가합니다.

성 춘 향*, 김 길 동**

*한국대학교 교육공학과, **한국대학교 공학교육과

Chun-hyang Sung*, Gil-dong Kim**

*Department of Education, Hankuk University, Seoul, 321-987, Korea
**Department of Engineering, America University, Atlanta, GA11222, USA

요 약 (10포인트 진하게)

당구는 재미있는 스포츠이지만, 처음 입문한 초심자가 득점 가능한 경로를 계산하고 올바르게 공을 쳐서 보낼 정도로 숙련되기까지의 진입 장벽이 높은 편이다. 당구 초심자가 어느 정도 수준에 도달하기 위해선 지속적인 집중과 훈련을 필요로 하는데, 적절한 동기 부여 요소가 없다면 흥미를 잃어버리기 쉽다. 본 연구는 스테레오 카메라와 VR 헤드셋을 결합한 몰입도 높은 증강 현실 플랫폼 상에서 당구 경로 안내 및 시각 효과를 통해 초심자의 흥미를 유도하고 당구 학습을 가속하는 것을 목표로 두었다. 이를 위해 영상처리를 활용하여 당구공 배치를 인식하고 Unity Engine의 물리 시뮬레이션을 통해 경로 탐색과 시각화를 수행해 실제와 유사한 경로 예측을 구현했다. 이는 당구에 처음 입문하는 초심자가 경로 설계에 대한 부담 없이 공을 올바르게 보내는 훈련에만 집중할 수 있게 만들며, 나아가 오랜 시간 알고리즘이 제안하는 경로를 익힘으로써 점진적으로 당구 숙련도를 높일 수 있다는 점에서 AR 당구의 학습 보조

도구로서의 가능성을 확인할 수 있었다.

ABSTRACT (10포인트 진하게)

Billiards is a fun and popular sport, but both route planning and cueing prevent beginners from becoming skillful. A beginner in billiards requires constant concentration and training to reach the right level, but without the right motivating factor, it is easy to lose interests. This study aims to induce interest in billiards and accelerate learning by utilizing billiard path prediction and visualization on a highly immersive augmented reality platform that combines a stereo camera and a VR headset. For implementation, the placement of billiard balls is recognized through the OpenCV image processing program, and physics simulation, path search, and visualization are performed in Unity Engine. As a result, accurate path prediction can be achieved. This made it possible for beginners to reduce the psychological burden of planning the path, focus only on accurate cueing, and gradually increase their billiard proficiency by getting used to the path suggested by the algorithm for a long time. We confirm that the proposed AR billiards is remarkably effective as a learning assistant tool.

Key Words: augmented reality; image processing; physics simulation; visualization

I. 서론

당구는 익숙해지기까지 많은 시간의 훈련과 적응을 필요로 하는 스포츠이다. 정보 매체의 발달로 학습을 필요로 하지 않는 직관적인 오락 컨텐츠가 폭발적으로 양산되는 시대에 이러한 진입 장벽은 초보자들이 발을 들이는 데 어려움으로 작용한다.

당구는 경로 설계와 큐잉(당구공을 큐로 치는 것) 두 가지가 모두 성공적으로 수행되었을 때 득점을 할 수 있다. 그러나 초보자가 나름대로 경로를 설계해 큐잉을 해도, 경로대로 공을 보내는 것 자체가 어려운 만큼 운이 아닌 실력만으로 득점을 내기는 쉽지 않다.

보통 당구는 체계적인 훈련이 아닌 놀이로서 시작하는 경우가 많으며, 놀이로서 학습을 지속하기 위해선 심리적인 보상이 필요하다. 당구에서는 득점이 이에 해당하는데, 상술한 진입 장벽에 빗대어 보면 당구 입문자는 실력이 갖춰질 때까지의 오랜 기간을 확실한 심리적인 보상 없이 훈련을 견뎌야 한다는 이야기가 된다.

당구공의 경로 안내는 바로 이러한 심리적인 보상 측면에서 출발한다. 아무런 보조 없이 당구를 연습하는 경우 초심자는 경로 설계와 큐잉 양단 모두를 병렬적으로 훈련해 나가야 하는 데다 적절한 피드백을 받기도 어려우므로 학습을 지속하기가 어려웠다.

그러나 득점할 수 있는 경로를 알 수 있다면 초심자는 큐잉에만 온전하게 집중할 수 있으며, 어느 정도 큐잉에 익숙해진 후에는 시스템이 제공하는 경로 정보를 바탕으로 경로 설계에 대한 암시적인 경험을 쌓을 수 있다.

위의 아이디어를 바탕으로 이미 시중에는 당구대 상단에 카메라와 프로젝터를 설치하여 당구공의 배치를 분석해 경로를 안내하거나, 혹은 큐의 방향을 분석해 공의 진행 경로를 예측하는 프로

젝트가 소개되었다 [1].

본 연구는 위 프로젝트와 맥락을 같이 하되, 주제의 구현에 점차 보편화되어가고 있는 증강 현실 장비를 활용한다. 이는 기존의 시스템에 비해 장소의 제약이 없는 데다, 기존의 증강 현실 장비를 활용할 수 있으므로 적은 비용과 높은 범용성을 갖게 된다.

II. 제안 방식

(그림 1)

가상 현실 기술은 말 그대로 가상으로 세계를 만들고, 머리 위치 추적 기술을 통해 가상 세계의 카메라 위치를 사용자 눈의 위치와 동기화하여 사실적인 시청각 경험을 부여하는 기술이다. 가상 현실 어플리케이션은 기존의 전통적인 3D 게임에서 카메라의 제어만을 사용자가 장착한 HMD의 헤드 트래킹 신호에 위임한 것으로, 가상 현실 세계를 만드는 것은 3D 게임을 만드는 것과 다르지 않다. 이는 증강 현실 어플리케이션에서도 마찬가지이며, 따라서 실제 이미지 위에 증강 현실을 통해 이미지를 표시하기 위해선 먼저 영상 처리를 통해 실제 물체의 위치가 가상 현실 상에서 어느 좌표에 맵핑(mapping)되는지 파악할 필요가 있다.

본 연구에서는 카메라를 통해 입력된 이미지를 분석하여 당구대와 당구공의 카메라에 대한 상대 자세를 계산하고, 증강 현실 장비의 위치 추적 기능을 통해 얻은 가상 세계에서의 사용자의 머리 자세(카메라 위치 및 회전)를 활용해 이를 가상 세계의 절대 자세로 변환한다. 당구대와 당구공의 위치가 파악되면 이를 바탕으로 물리 시뮬레이션을 수행할 수 있다. 이는 플레이어가 칠 공을 360도 전 방향에 대해, 다양한 속도로 타격 시뮬레이션을 수행한 뒤 득점하는 경로를 모두 파악하는 식으로 이루어진다. 시각화 기능은 사용자의 머리 위치 및 경로의 가중치에 따라 가장 합리적인 경로를 유니티 엔진의 다양한 파티클 시스템을 활용하여 시각화한다. 위의 세 과정은 시스템이 구동되는 동안 실시간으로 수행되는데, 자세한 설계 방식은 아래에서 다룬다.

A. 영상 처리

(그림 2)

본 연구에 사용된 당구대의 펠트는 이미지 상에서 다른 물체와 뚜렷하게 구별되는 색상이다(그림 3). 이를 HSV 색공간; 색상(Hue), 포화(Saturation, 색의 진한 정도), 명도(Value)로 변환하여 평가하면 균일한 색상을 갖는 당구대의 펠트 표면은 안정적인 색상과 포화 값을 갖게 된다.

그림 4는 HSV 색공간으로 변환한 이미지에서 전체 픽셀의 명도 값을 255로 설정, 명도 차이를 없앤 것이다. 명도의 영향을 없애자 당구대의 펠트 영역이 훨씬 더 선명하게 부각되고, 더불어 공의 색상도 뚜렷하게 구별되는 것을 볼 수 있다. 테이블 영역의 색상과 포화 값은 뚜렷하게 구별되는 범위 내에 존재하므로, 단순히 H, S 값이 일정 범위 내에 있는 픽셀을 모두 골라내는 것만으

로도 꽤 정확하게 당구대의 영역을 찾아낼 수 있다. 여기서는 OpenCV 라이브러리에서 제공하는, 각 채널의 값이 일정 범위 내에 존재하는 모든 픽셀을 찾아내는 함수인 inRange()를 사용하여 색상 값 165~15°이고 포화 값이 150~255 사이인 모든 픽셀을 True로 지정한다.

필터링 결과에 침식 연산을 적용하고 원본에서 제함으로 계산한 경계선 이미지(그림 5)로부터, 이미지 내에 존재하는 모든 도형을 찾아낼 수 있다. 각 도형은 해당 도형을 이루는 꼭지점의 좌표 목록으로 구성되어 있다. OpenCV 라이브러리에서는 findContours()라는 함수로 제공된다 [2].

그러나 OpenCV 함수의 알고리즘은 정점 사이의 직선 여부를 매우 엄격하게 평가하기 때문에 그림 5의 사각형 도형은 네 개보다 훨씬 많은 수의 정점으로 평가된다. 따라서 도형을 이루는 정점의 개수를, 가장 뚜렷한 특성을 갖는 네 개의 정점으로 근사해줄 필요가 있다. 여기서는 *Ramer-Douglas-Peucker* 직선 근사 알고리즘[3]을 구현하는 OpenCV의 approxPolyDP() 함수를, $\epsilon = 5$를 인자로 적용하여 무수한 정점들을 그림 6과 같이 네 개의 꼭지점으로 근사한다. 이후 당구 큐대 등 다른 오브젝트에 의해 당구대 영역이 침범되는 경우에도 당구대의 꼭지점 네 개가 화면 안에 있는 한 당구대 영역이 검출될 수 있게끔 OpenCV의 convexHull() 함수를 적용해 모든 당구대 영역을 볼록 다각형으로 만들어준다.

위의 과정을 거쳐 이미지 상에서 당구대와 같은 색상을 갖는 모든 영역에 대한 도형 정점 집합을 획득하게 된다. 이 때 당구대의 꼭지점 개수가 4개인 것은 자명하고, 사용자가 당구대를 바라보고 있다면 당구대 영역의 크기 또한 화면 상에서 일정 이상을 차지할 것이다. 따라서 각각의 도형을 순회하며 조건에 부합하는 도형을 찾고, 이를 당구대의 네 개 꼭지점이라 가정한다. 여기까지의 과정이 그림 2의 당구대 영역 검출에 해당한다(그림 6).

당구대는 변형되지 않는 강체(rigid body)이며, 당구대의 치수는 이미 알고 있으므로, 그림 7과 같이 당구대와 같은 치수의 물체가 원점(0, 0, 0)에 있다고 가정한다. 이 때 원점은 카메라의 위치를 의미한다. 그림 7의 당구대를 옮김에 따라 원점에 위치한 카메라의 시야에 투영된 당구대의 모양이 바뀔 것임은 자명하고, 투영된 물체가 화면 상에 나타나는 당구대의 영역과 정확하게 겹치게 되는 위치와 회전 값이 존재한다. 이를 당구대의 자세라 한다.

그림 7과 같이 모델 공간에 위치한 3D 모델의 각 정점을 화면상에 투영된 좌표와 대조하여 물체의 자세(위치와 회전)을 구하는 알고리즘을 *Perspective N-Point Algorithm; PNP* 알고리즘이라 하는데, OpenCV에서는 solvePnP()라는 함수에서 여러 논문에 근거한 알고리즘을 구현한다 [4][5]. 단, OpenCV 구현은 화면에서 검출된 2D 정점의 순서가 모델 공간의 3D 정점의 순서와 정합되어 있다고 가정하는데, 실제로는 화면 상에 검출된 당구대의 긴 변-짧은 변 정점의 순서가 모델 공간에서의 순서와 다를 수 있다. 따라서 먼저 PNP 알고리즘을 적용하고, 인덱스 목록을 방향에 관계없이 한 번 회전시켜 다시 PNP 알고리즘을 적용, 더 오차가 적은 후보를 당구대의 최종 자세로 선정한다.

이를 통해 획득한 당구대의 자세는 카메라에 대한 상대 자세이므로, 여기에 AR 헤드셋의 위치 추적 기능이 반환한 카메라의 절대 자세를 적용, 당구대의 절대 자세를 계산한다.

만약 영상 내에서 당구대의 자세를 추정해내지 못하더라도, 여전히 AR 헤드셋의 위치 추적 기능을 통해 카메라의 현재 자세가 갱신되므로 당구대의 저장된 절대 자세를 바탕으로 꾸준히 추적을 유지할 수 있다. (이로 인해, 최초 인식 이후 추가적인 당구대 인식은 AR 헤드셋 위치 추적기의 오차 보정 차원에서 이루어진다)

한 번 당구대의 자세를 획득하고 나면, 화면 상에서 검출된 당구공의 2D 중점 방향으로, 카메라에서 출발하는 광선을 투사하여 당구대 평면과 충돌시킴으로써 당구공의 3D 중점을 계산할 수 있다. 이는 그림 8에서와 같이 당구공이 모두 한 평면상에 위치하기 때문으로, 당구공의 이미지 상 중점의 UV 위치만 정확하게 획득할 수 있다면 그림 9의 방법을 통해 당구공의 중점 위치를 획득할 수 있다. 실제로 문제가 되는 것은 당구공의 중점 위치를 찾는 것으로, 그림 4에서 육안으로 당구공을 손쉽게 찾을 수 있는 것과 달리 조명의 위치, 주변광 등의 영향으로 인해 기계적으로는 이를 식별하기가 어려워진다.

본 연구에서는 변형된 템플릿 매칭을 통해 중점 탐색의 정확도와 성능 최적화 사이에서 균형을 찾고자 하였다. 공의 추정 중점 2D 좌표에서, 해당 위치에서의 공의 이미지 상 픽셀 반경 크기의 커널로 색상에 대한 템플릿 매칭을 수행하며, 이 때의 템플릿은 그림 4와 같이 명도를 제외한 색상, 휘도의 비교만을 수행한다.

그러나 일반적인 템플릿 매칭과 같이 이미지 전체에 동일한 커널로 연산을 수행할 수는 없는데, 공과의 실제 거리에 따라 커널의 반경이 동적으로 바뀌어야 하기 때문이다. 커널 크기가 동적으로 바뀌게 되면서 당구공의 중심이 될 수 있는 모든 후보에 대해 각각의 커널 연산을 적용해야 하고, 이는 OpenCV에서 제공하는 GPU 가속의 힘을 빌리더라도 성능 상의 손실을 야기한다.

따라서 희소 템플릿 커널(sparse template kernel) 개념을 도입, 전체 픽셀의 약 5~10%정도만을 샘플링하여 템플릿 매칭을 수행함으로써 성능 최적화를 달성하였다. 이 때 커널에서 샘플링하는 픽셀은 그림 10과 같이 나타난다. 그림 10에서, 내부의 초록색 샘플은 색상이 일치할 때 해당 중점 후보에 대한 가중치를 높이고, 외곽선의 빨간색 샘플은 색상이 일치할 때 후보의 가중치를 낮춤으로써 일종의 경계선 매칭까지도 수행한다.

그림 10에서 각 커널의 양의 가중치 점(초록색)은 중점에 대해 0.3~1 사이의 거리를 갖고, 음의 가중치 점(빨간색)은 1.03~1.33 사이의 거리를 갖게끔 정규화된다. 이들을 정규화 시점에 반지름의 길이만큼 곱해 가변 크기의 희소 커널 연산을 구현하게 된다. 이 때 그림 10의 커널에서 양의 가중치 샘플이 위쪽 반원 형태를 띠는 것을 볼 수 있는데, 이는 대부분의 시야각에서 공의 하단이 음영에 의해 색상과 휘도 값이 불분명한 값을 나타내기 때문에 측정에서 원천적으로 제하기 위함이다.

희소 템플릿 커널은 단순한 가중치 합 연산을 수행하기 때문에, 해당 커널이 적용될 가중치 필드를 생성해야 한다. 이는 그림 4와 같은 색상-휘도 색공간 이미지에서 각 공의 색상-휘도 대표값을 제하여 각 채널의 오차를 구하고, 각 채널간 오차의 유클리드 거리의 음수를 지수함수의 인자로 사용하여 0~1 사이에 값이 분포하는 적합도를 획득한다. 각 픽셀에 대한 위의 적합도 연산을 식으로 나타내면 다음과 같다.

$$\vec{e}_n = \begin{pmatrix} e_{h_n} \\ e_{s_n} \end{pmatrix} = \begin{pmatrix} h_n \\ s_n \end{pmatrix} - \begin{pmatrix} h_r \\ s_r \end{pmatrix}, \tag{1}$$

$$d_n = \left| \begin{pmatrix} w_h \\ w_s \end{pmatrix} \odot \vec{e}_n \right|, \begin{pmatrix} a \\ b \\ c \end{pmatrix} \odot \begin{pmatrix} d \\ e \\ f \end{pmatrix} = \begin{pmatrix} ad \\ be \\ cf \end{pmatrix}, \tag{2}$$

$$\gamma_n = \beta^{-d_n} = e^{-d_n ln\beta}. \tag{3}$$

이 때 식 (1)의 h_n, s_n은 각 픽셀의 색상 및 휘도 값, h_r, s_r은 검출하고자 하는 공의 대표 색상 및 휘도 값을 나타내며, 식 (2)의 w_h, w_s는 오차의 유클리드 거리를 계산할 때 색상과 휘도 성분 오차 각각에 적용할 가중치이다. 식 (3)의 β는 오차의 영향력을 결정하는 계수로, 높을수록 오차에 민감하게 되어 정확도가 올라가나, 주변광이나 조도 등 환경 영향에 취약해진다.

위 식에서 각 채널은 모두 32비트 부동 소수점 형태로 계산되며, 이를 위해 0~1 사이의 값으로 정규화되므로 거리 d는 실질적으로 매우 작은 값을 갖게 된다. 이를 극복하기 위해 β에는 30000의 큰 값을 지정한다. 공 인식에 활용되는 나머지 파라미터는 표 1의 값을 따른다.

위의 방법을 활용하여 그림 11의 샘플로부터 각 색상에 대한 적합도 필드를 그림 12과 같이 획득할 수 있으며, 위의 희소 템플릿 커널을 위 필드에 곱하여 공의 중점을 추정하게 된다. 그러나 공이 존재할 수 있는 영역은 테이블의 ROI에서도 매우 적은 영역에 해당하므로, 비싼 템플릿 연산을 이미지 전체에 적용하는 것은 비효율적이다. 따라서 먼저 고정된 적합도 값을 문턱값으로 사용하여 이미지를 이진화, 인덱스를 추출하여 해당 인덱스에 대해서만 템플릿 연산을 적용하게 된다.

단, 그림 12의 빨간 색과 주황색 공에 대한 적합도 필드에서도 확인할 수 있듯이 공의 적합도를 단순히 이진화할 경우 광원 상황에 따라 정작 공의 실제 중점이 아닌 경계선에만 후보 필드가 형성될 가능성이 높다. 따라서 일차적으로 이진화를 수행한 뒤 약 6~8회의 팽창-침식 연산을 반복 적용하여 당구공의 중점이 매칭 필드에 포함되도록 한다. 매칭 필드의 각 점에 대해 그림 10의 커널을, 그림 12의 적합도 필드에 적용하여 각 점의 적합도 합을 계산한다. 개념적으로 반지름 내에 목표 색상의 픽셀이 많을수록, 반지름 밖에 목표 색상의 픽셀이 적을수록 높은 적합도를 갖는다. 이후 때 적합도 합 필드에서 최대값을 갖는 점을 선택하면, 이것이 각 공의 2D 추정 중점 위치가 된다.

빨간색 공은 두 개가 존재하기 때문에 위의 과정을 한 번 더 수행해주는데, 이 때 기존에 이미

검출된 공의 적합도 합 필드는 의미가 없으므로 공의 반경으로 지워준다. 이렇게 계산된 공의 중점 위치로 원점에서 광선을 투사함으로써 카메라에 대한 공의 3D 중점 좌표를 획득할 수 있으며, 여기에 카메라 자세 행렬을 곱해 절대 위치로 바꾸어 준다. 그림 11의 프레임에서 위의 당구공 탐색 연산을 적용하여, 위에서 내려다본 시점으로 투영한 결과는 그림 13과 같다.

B. 경로 시뮬레이션

위의 과정을 통해 검출된 당구대와 각 당구공의 위치는 Unity Engine으로 구현된 AR 당구 어플리케이션으로 전달된다. 당구대와 네 공의 절대 위치를 모두 아는 경우, 물리 시뮬레이션을 통해 플레이어가 친 공의 경로를 시뮬레이션하고 득점이 나는 경우를 계산해낼 수 있다.

일반적으로 물리 시뮬레이션은 시분할, 또는 타임 스텝이라 불리는 이산적인 방법에 의해 제어된다. 예를 들어 시점 n에서 물체의 가속도가 $\vec{a}_n$이고, 시뮬레이션 간격이 Δt일 때 각각의 시뮬레이션 단계에서 다음 속도 $\vec{v}_{n+1}$와 위치 $\vec{s}_{n+1}$를 다음과 같이 계산한다.

$$\vec{v}_{n+1} = \vec{v}_n + \vec{a}_n \Delta \mathrm{t}, \tag{4}$$

$$\vec{s}_{n+1} = \vec{s}_n + \vec{v} \Delta \mathrm{t}. \tag{5}$$

이는 적분 계산을 단순화하므로 복잡한 물리 수식과 점화식의 도입이 자유롭다는 장점이 있다. 또한 시뮬레이션 간격 Δt를 조절함으로써 시뮬레이션의 정확도와 최적화 수준을 제어할 수 있다. 그러나 시분할 방식의 시뮬레이션은 전체 시뮬레이션의 길이, 그리고 정확도 수준에 따라 연산량이 선형적으로 증가한다. 이는 경로 탐색을 위해 공의 타격을 360도 방향으로, 다양한 속도에 걸쳐 수천 회 이상 시뮬레이션 해야 하는 본 연구의 목표에는 다소 적합하지 않다.

따라서 본 연구에선 시분할이 아닌 이벤트 기반의 물리 시뮬레이션 기능을 새로 설계하였다. 이벤트 기반 모델은 시분할 기반 시뮬레이션에서 연산의 대부분을 차지하는, 충돌이 없는 기간에 대한 시뮬레이션 전체를 생략하는 모델이다. 일반적인 시분할 시뮬레이션이 매 시뮬레이션 스텝마다 Δt의 시간 동안 모든 물체가 움직인 거리를 연산하고, 이후 모든 물체의 충돌을 검사하는 것과 달리, 이벤트 기반 모델은 구체 궤적 교차[8]를 바탕으로 매우 긴 시간 간격에 대해 모든 물체의 충돌을 검사하고, 가장 먼저 충돌이 발생하는 시간에 대해 모든 물체의 이동을 시뮬레이션한 뒤, 해당 지점에서 충돌한 물체의 충돌 물리를 계산한다. 즉, 이벤트 기반 충돌 연산은 복잡도가 $O(n^2)$인 충돌 검사를 시뮬레이션 기간 동안 충돌이 발생하는 횟수만큼만 시행하게 되며, 이는 시분할 방식에 비해 최적화 측면에서 많은 이득을 갖는다.

물론 속도가 상수나 자연 감쇠($v(t) = V_0 e^{-\alpha t}$)가 아닌 경우 궤적 검사의 복잡도가 매우 크게 증가하게 되므로, 가속도 함수가 존재하는 경우 시뮬레이션 자체가 불가능하다는 큰 단점이 있지만, 당구의 경우 초기 속도를 제외하면 외력이 작용하지 않으므로 이벤트 기반 시뮬레이션의 도입이

적절하다 판단되었다.

일차적으로 구체 궤적 검사를 통해 충돌 여부를 검사하고 나면, 실질적인 충돌 물리 반응을 계산해야 한다. 현재 모델은 당구공과 쿠션에 작용하는 물리[6]를 정교하게 구현하는 대신, 공과 공, 공과 쿠션 사이의 탄성 계수 ϵ을 직접 지정하고[7], 공과 쿠션 충돌 시 공에 적용되는 회전을 경험에 기반한 휴리스틱을 통해 구현하였다.

공과 공, 공과 쿠션 사이에 적용되는 충돌을 계산하기 위해 다음의 식을 적용한다.

$$V_{np} = V_n \text{ proj } P_{ct}, \tag{6}$$

$$V'_{1p} = \frac{(m_1 - \epsilon m_2)V_{1p} + (1+e)m_2 V_{2p}}{m_1 + m_2}, \tag{7}$$

$$V'_1 = V_1 + \left(V'_{1p} - V_{1p}\right). \tag{8}$$

식 (6)의 P_{ct}는 두 객체 사이의 충돌 지점이고, 식 (8)의 V'_1는 충돌 이후 객체 1의 속도이다. 충돌한 다른 물체인 V_2에도 위와 같은 식이 적용된다. 만약 공과 쿠션이 충돌하는 경우 m_2를 무한대로 두어, $V'_{1p} = -\epsilon V_{1p}$로 계산한다. ϵ은 공과 공 사이의 충돌에서 0.77, 공과 쿠션의 충돌에서 0.73이다.

회전의 영향 또한 어느 정도 고려하는데, 일반적으로 당구공이 충돌 이후 일정 시간 후에 진행 속도와 구르는 방향이 일치하게 된다는 점을 모사하여 최소 구름 시작 시간 t_{roll}을 설정, 마지막 충돌부터 흐른 시간 Δt에 따라 회전 속도가 진행 방향과 일치하게끔 현재 회전 속도 $\vec{\omega}_n$과 최대 회전 속도 $\vec{\omega}_{n(max)}$ 사이를 다음의 식과 같이 선형 보간한다.

$$\alpha = \frac{\Delta t}{t_{roll}}, \tag{9}$$

$$\vec{\omega}_{n(max)} = \frac{\vec{v}_n \times (-\hat{u}_{up})}{r}, \tag{10}$$

$$\Delta \vec{\omega}_{n(max)} = \vec{\omega}_{n(max)} - \vec{\omega}_n, \tag{11}$$

$$\vec{\omega}_{n+1} = \vec{\omega}_n + \Delta \vec{\omega}_{\mathrm{n(max)}} \cdot min(\alpha, 1). \tag{12}$$

식 (11)의 $\hat{u}_{up}$은 위 방향 벡터, r은 공의 반지름이다. 식 (12)의 $\vec{\omega}_{\mathrm{n+1}}$은 충돌 시점 t_{n+1}에서의 회전 속도가 된다. 실험에서 공이 진행 방향의 최대 구름 속도에 도달하는 시간 $t_{roll} = 0.7$로 설정하였다. 이는 실제와는 현저히 다른 수치이지만, 반복적인 실험을 통해 실제 경로를 비교적 잘 반영하는 값으로 설정하였다.

공과 공 사이에 충돌이 발생할 시 일차적으로 위의 탄성 계수 ϵ에 기반한 충돌식으로 계산된 속도 v_0를 회전을 고려하여 조작한다. 이 때 회전의 반영 비율을 마찰 상수 $f_{s(ball)} = 0.23$로 직접 지정한다. 이 과정은 다음과 같으며, 이를 충돌한 공 두 개에 각각 적용한다.

$$\vec{v}' = \vec{v}_0 + rf_{s(ball)}\vec{\omega}_{n+1} \times \hat{u}_{up}. \tag{13}$$

공과 쿠션 사이의 충돌은 좀 더 복잡하게 계산된다. 위와 마찬가지로 v_0을 계산하고, 여기에 쿠션 평면과 공의 접점에서 공의 회전에 의해 발생하는 마찰력을 모사한다. 이는 공의 회전 속도 벡터 $\vec{\omega}$와 접촉한 쿠션 평면의 법선 $\hat{n}$의 외적 방향 속도와 같다.

단, 중력을 고려하지 않으므로 수직 방향 속도를 제거하며, 여러 실험과 관찰에 근거하여 법선 방향의 속도도 제거하였다. 이는 상향 벡터와 법선 벡터의 외적 결과로 접점의 속도 벡터를 스케일링 하는 것과 같다. 이 때 회전의 어느 정도를 속도에 반영할지를 결정하기 위해 마찰 상수 $f_{s(table)} = 0.67$을 적용한다. 다음은 이를 식으로 표현한 것이다.

$$\vec{v}_{contact} = \left(rf_{s(cushion)}\vec{\omega}_{n+1} \times \hat{n}\right) \odot \left(\hat{\mathrm{u}}_{\mathrm{up}} \times \hat{\mathrm{n}}\right), \tag{14}$$

$$\vec{v}' = \vec{v}_0 + \vec{v}_{contact}. \tag{15}$$

충돌이 공의 회전에도 영향을 주게끔, 공의 쿠션 방향의 속도 벡터 $\vec{v}_p$에서 속도 벡터 $\vec{v}_0$을 제하여 마찰 방향의 속도 벡터 $\vec{v}_f$를 계산하고, 이를 쿠션의 법선과 외적하여 각속도 벡터를 계산한 뒤 상수 $c = 0.2$의 비율만큼 각속도에 누적한다. 다음은 이를 식으로 표현한 것이다.

$$\vec{\omega}' = \vec{\omega}_{n+1} + \frac{c}{r}\left(\vec{v}_p - \vec{v}_0\right) \times \hat{n}. \tag{16}$$

위의 휴리스틱을 바탕으로 시뮬레이션 엔진을 구현하고, 당구공의 배치가 바뀔 때마다 공의 360도 각도를 0.5° 단위로 나눈 뒤, 시작 속도 0.15, 0.3, 0.6, 0.9[m/s]에 대해 각각 총 2880번 시뮬레이션을 수행한다. 모든 경로에 대한 시뮬레이션 결과 중 득점하는 경로를 일차적으로 획득하고, 득점 시 공과의 충돌 각도, 충돌 전 외부 요인에 의한 이동 여부 등 여러 가지 요소를 고려하여 최적의 경로 목록을 필터링한다. 이 목록에서 사용자의 머리 방향과 가장 가까운 경로를 선택해 시각화하게 된다.

III. 실험 결과

최종적으로 완성된 프로그램의 실기 화면은 그림 14와 같다. 기본적으로 시뮬레이션은 당구공의 정중앙을 타점으로 두고 타격하는 조건만을 시뮬레이션 하는데, 공의 속도가 0.6m/s 이내인 경우 방향과 속도를 정확하게 맞추었을 때 납득할 만한 정확도를 보였다. 여기서 정확도는 경로에서 3° 이상 벗어나지 않고 득점하는 비율로 정의된다.

그러나 실험에 당구공이 다소 작고 가벼워 회전의 영향을 매우 크게 받으므로, 조금이라도 타점이 어긋나면 쿠션과의 충돌 이후 예측과 현저히 다른 경로를 향하는 경향을 보인다. 그러나 이

러한 부분은 사용자로 하여금 정확한 타격을 강제하므로, 학습 유도 측면에서는 바람직하다고 볼 수 있다.

공의 배치를 일치시키고 여러 가지 속도에 대해 실험한 결과, 공을 밀어 치는 경우, 즉 공의 분리각이 15° 이하일 때는 경로 예측을 신뢰하기 어려웠으며, 분리각 30~75°로 일반적인 충돌이 일어날 때, 공의 시작 속도 0.3~0.6m/s 구간에서 90% 이상의 정확도를 보였다. 쿠션에 대한 충돌은 한 번 충돌할 때는 거의 모든 시작 속도 구간에서 90% 이상의 정확도를 보였다. 그러나 두 번째 충돌은 공의 시작 속도에 증가함에 따라 반사각이 실제보다 좁게 평가되는 경향을 보이며, 시작 속도 0.9m/s에서는 두 번 이상의 쿠션 충돌을 신뢰하기 어려웠다. 대부분의 경우에서 시작 속도 0.3m/s, 0.6m/s는 시작 속도 및 방향 조건을 정확하게 맞추는 한 60% 이상의 정확도를 보이며, 이는 당구 입문자의 득점률을 고려했을 때 높은 정확도를 얻을 수 있음을 보여준다.

영상 처리 프로그램의 경우 의존할 수 있는 시각 정보가 사용자의 시야에 위치한 카메라 하나뿐임에도 불구하고, 테이블과 각 공의 3D 좌표를 안정적으로 추출하고, VR 헤드셋 및IMU 내장 스테레오 카메라의 위치 추적 기능에서 발생하는 드리프트(drift) 현상을 효과적으로 보정할 수 있었다. 또한 영상 처리 알고리즘은 OpenCV에서 제공하는 GPU 가속 연산 클래스인 UMat의 적극적인 활용, 또 II.A에서 언급한 각종 최적화 기법 등을 통해 한 프레임당 처리 속도를 평균 33ms 수준까지 낮출 수 있었다(AMD Rygen 3900x 및 RTX 2060S 기준).

또한, 연구 기간 동안 AR 당구를 체험해 본 체험자들의 사용 평가에 따르면, 체험자는 일차적으로 당구공 자체에 적용된 시각 효과에 흥미를 느끼고, 당구를 전혀 접해본 적이 없는 경우에도 어플리케이션이 제시하는 경로를 바탕으로 득점을 내며 큰 심리적인 보상과 동기를 얻을 수 있었다.

IV. 결론

AR 당구의 기술적인 구현을 통해 알고리즘이 제시한 속도 및 방향 조건을 정확하게 갖추었을 때, 50% 이상의 확률로 성공 가능한 경로를 제시할 수 있었다. 충돌 계산이 휴리스틱에 기반을 둔 만큼 모든 상황에서 이상적인 정확도를 보이지는 못하나, 이는 많은 경우 알고리즘이 제시하는 조건을 갖추기 어려운 당구 초심자가 인지하기 어려운 수준의 오차로서, 당구 학습 보조라는 본 연구의 본질적인 목적을 고려하면 납득 가능한 수준의 구현으로 생각된다.

무엇보다도 AR 당구는 기존의 시스템[1]과 달리 시각화 표현의 범위가 입체로 확장되므로 몰입도 측면에서 상당한 이점을 갖고, 장기적으로 AR 장치의 소형화/보급화에 따라 새로운 학습 플랫폼으로 활용될 수 있음을 보여줬다.

감사의 글

이곳에 감사의 글을 입력한다.

참 고 문 헌

참고문헌 들은 반드시 본문 내에서 인용되어야 하며 [1, 2], [3-7], [1, 2, 5-8] 과 같은 형식으로 인용한다.

참고문헌은 반드시 영문으로 작성하며 번호와 본문을 아래의 보기와 같이 왼쪽 정렬한다. 단행본 제목은 Italic 이용한다. 참고문헌번호는 [1], [2], [3]의 형태 사용한다. 참고문헌 작성 순서는 논문인용번호, 논문저자명, 논문제목, 발행기관명(이탤릭체), 게재지명(논문지명), 집권호, 수록면, 발행년월 순으로 작성한다. 아래의 참고문헌 작성예를 참조하여 작성한다.

[1] Interesting Engineering, "This AR projector system acts like a billiards coach," *Youtube*, 2018 [Online]. Available: https://youtu.be/zHVW2_lH9vs (Accessed 7 Oct. 2020)

[2] OpenCV, "Structural Analysis and Shape Descriptors." Jul. 18, 2020 [Online]. Available: https://docs.opencv.org/4.4.0/d3/dc0/group__imgproc__shape.html (Accessed 7 Oct. 2020)

[3] Wikipedia, "Ramer-Douglas-Peucker algorithm", Aug. 22, 2020 [Online]. Available: https://en.wikipedia.org/wiki/Ramer%E2%80%93Douglas%E2%80%93Peucker_algorithm (Accessed 7 Oct. 2020)

[4] Xiao-Shan Gao, Xiao-Rong Hou, Jianliang Tang, and Hang-Fei Cheng. "Complete solution classification for the perspective-three-point problem." *Pattern Analysis and Machine Intelligence*, IEEE Transactions on, 25(8):930–943, 2003.

[5] Pengyu Cong, Zhiwei Xiong, Yueyi Zhang, Shenghui Zhao, and Feng Wu. "Accurate dynamic 3d sensing with fourier-assisted phase shifting." *IEEE Journal of Selected Topics in Signal Processing,* 9(3):396–408, 2015.

[6] Mathavan, Senthan & Jackson, M & Parkin, Robert. "A theoretical analysis of billiard ball dynamics under cushion impacts." *Proceedings of The Institution of Mechanical Engineers Part C-journal of Mechanical Engineering Science* - PROC INST MECH ENG C-J MECH E. 1. 1-10. 10.1243/09544062JMES1964, 2010.

[7] 3DGameProgram, "#3, 질점의 충돌," [Online]. Available: https://sites.google.com/site/3dgameprogram/home/physics-modeling-for-game-programming/-gibongaenyeomdajigi---mulli/-jiljeom-ui-chungdol (Accessed Oct 17. 2020)

[8] M. Gomez, "A Swept-Sphere Sweep Test", *Simple Intersection Tests For Games*, GAMASUTRA, Oct. 18, 1999 [Online]. Available: https://www.gamasutra.com/view/feature/131790/simple_intersection_tests_for_games.php?page=2 (Accessed Oct. 17, 2020)

그림과 표

모든 그림이나 표는 본문 내에서 반드시 인용되어야 한다. 모든 그림과 표는 참고문헌 다음에 위치시키도록 한다. 그림과 표의 제목은 국문과 영문으로 동시에 설명하도록 하고 국문이 위에 영문이 아래에 위치한다. 단, 영문논문일 경우 국문제목은 생략한다. 그림은 가로 사이즈 기준 17cm이상 해상도 300dpi 이상 되어야 한다. 그림의 영문제목에는 마지막에 마침표를 입력한다.

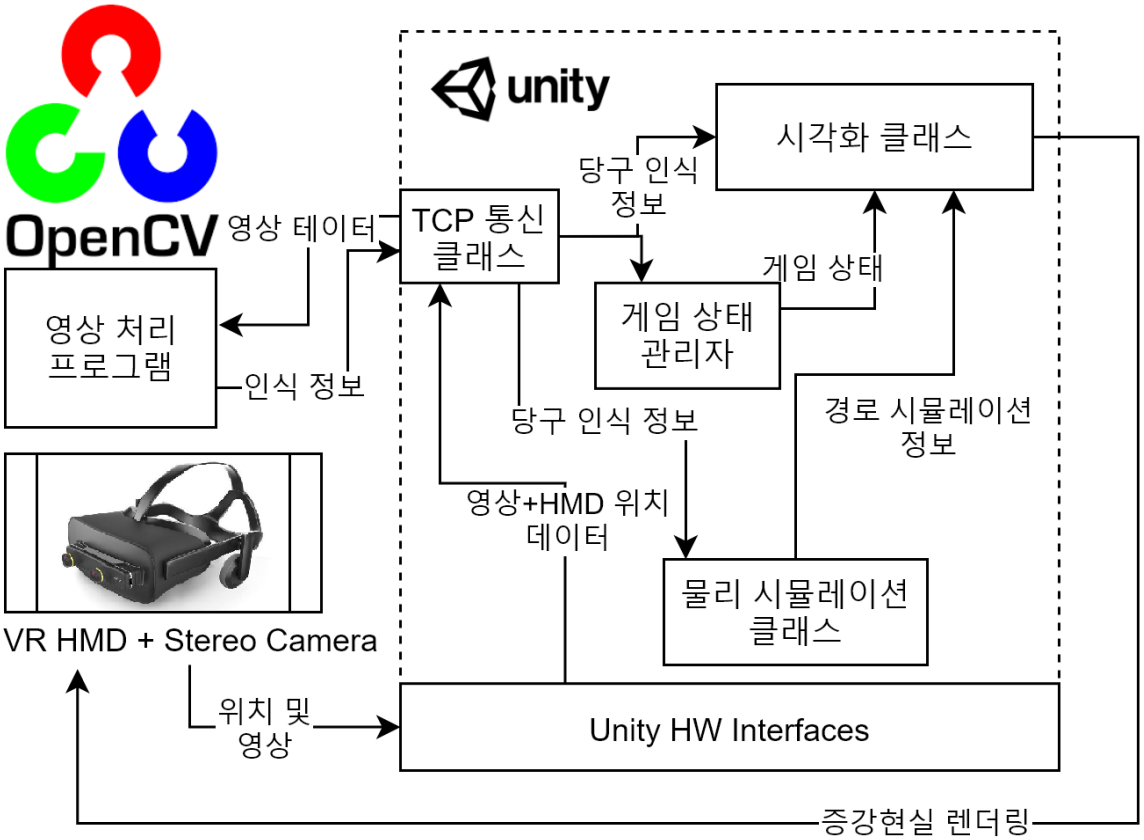


그림 1. 시스템 구성도

Figure 1. System block diagram.

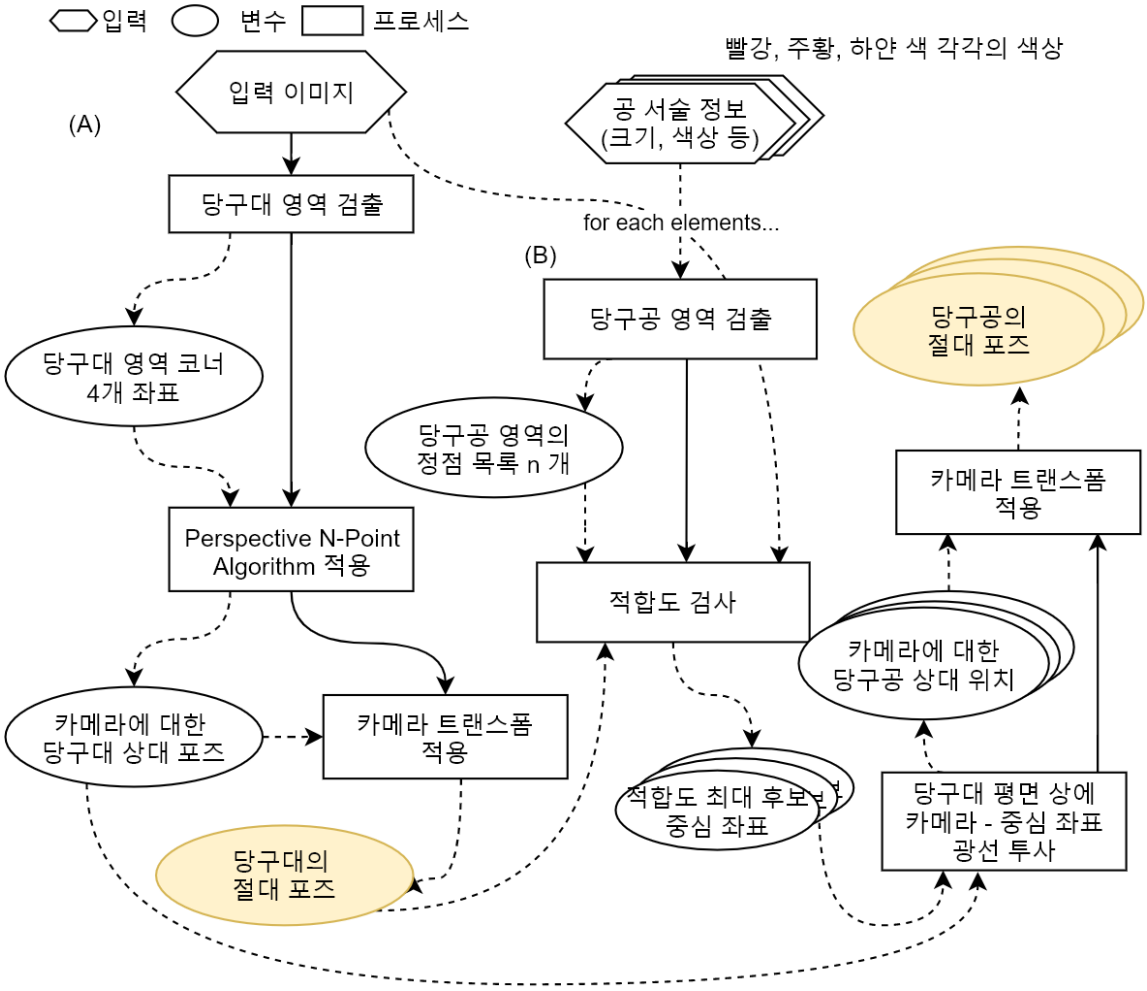


그림 2. 영상 인식 수행 절차

Figure 2. Process of image recognition.

그림 3. 과제에 사용된 미니 당구 세트 실물 이미지

Figure 3. Actual image of mini billiards set, which is used for this project.

그림 4. 명도(Value) 성분을 제거한 이미지

Figure 4. Image without value channel.

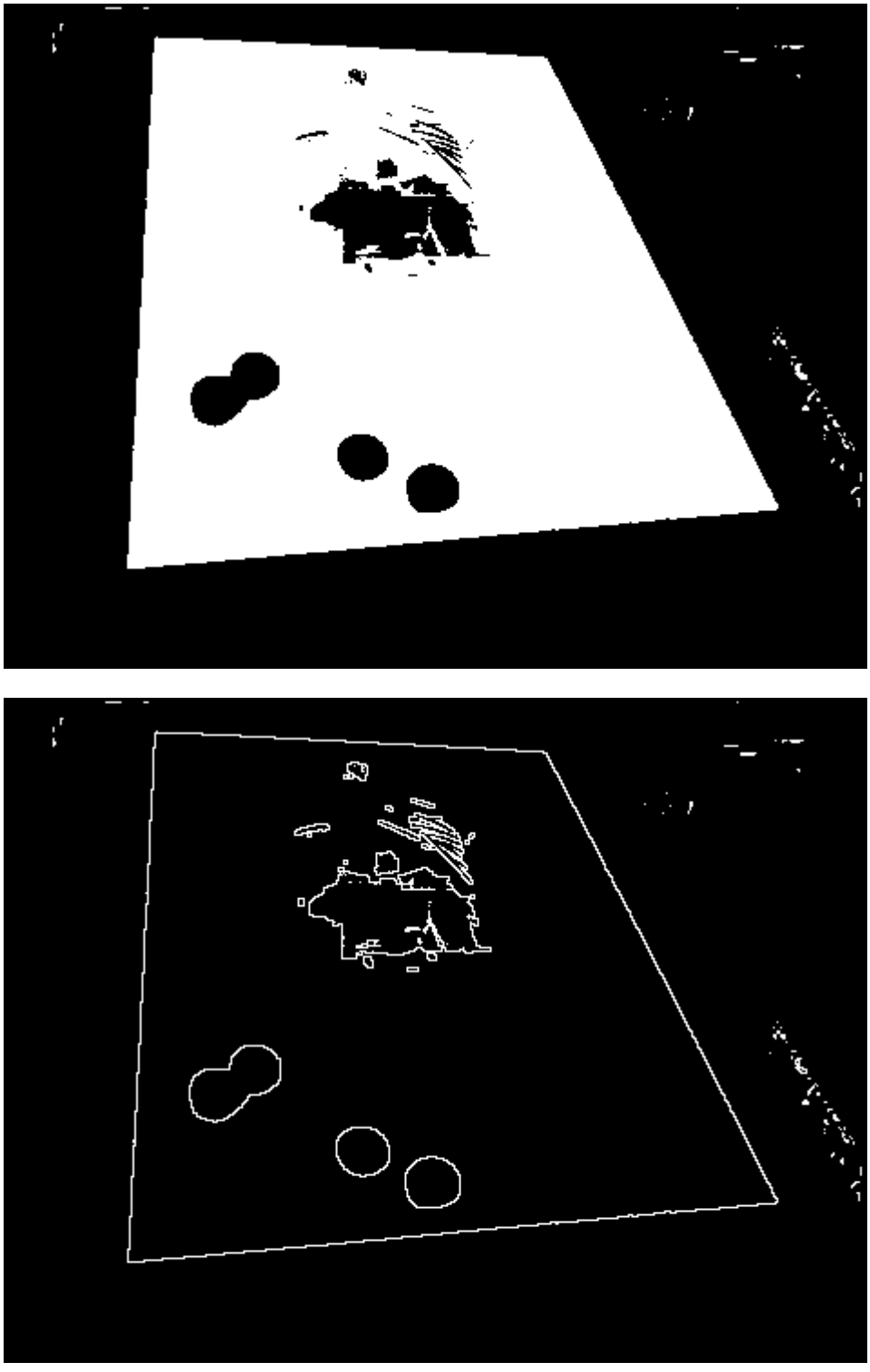


그림 5. HSV 색공간에서 단순 범위 필터 적용 결과 및 침식을 통한 경계 검출
Figure 5 Result of simple range filter and edge detection using erosion.

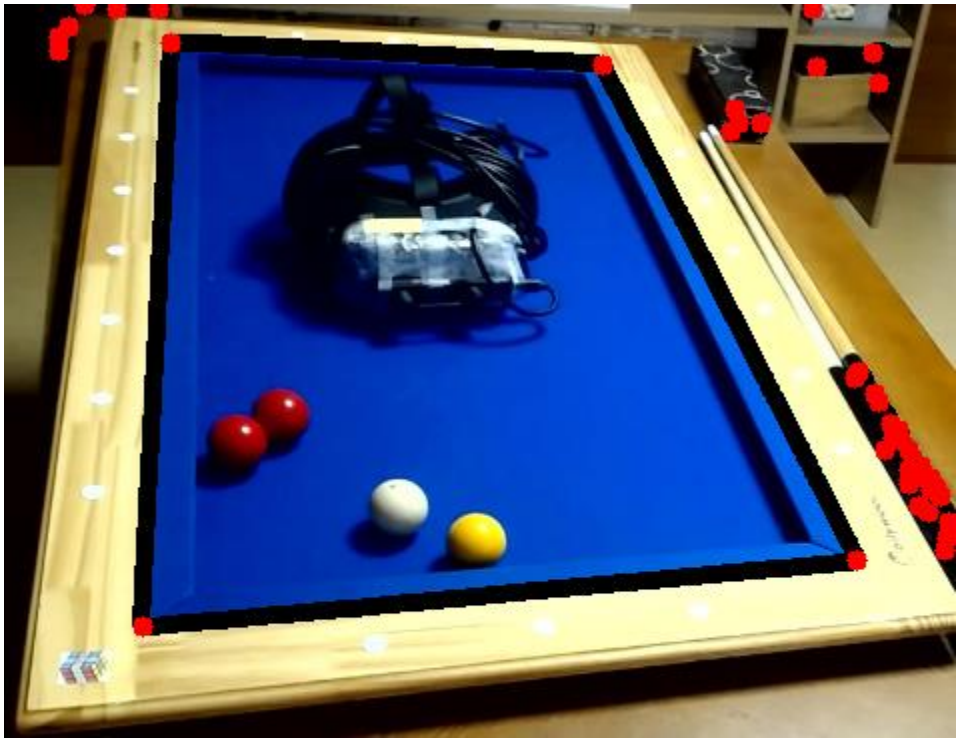


그림 6. 그림 5로부터 추출하고, *Ramer-Douglas-Peucker* 알고리즘을 적용하여 근사된 꼭지점 목록
Figure 6 List of points which is extracted from Figure 5, and is approximated using *Ramer-Douglas-Peucker* algorithm.

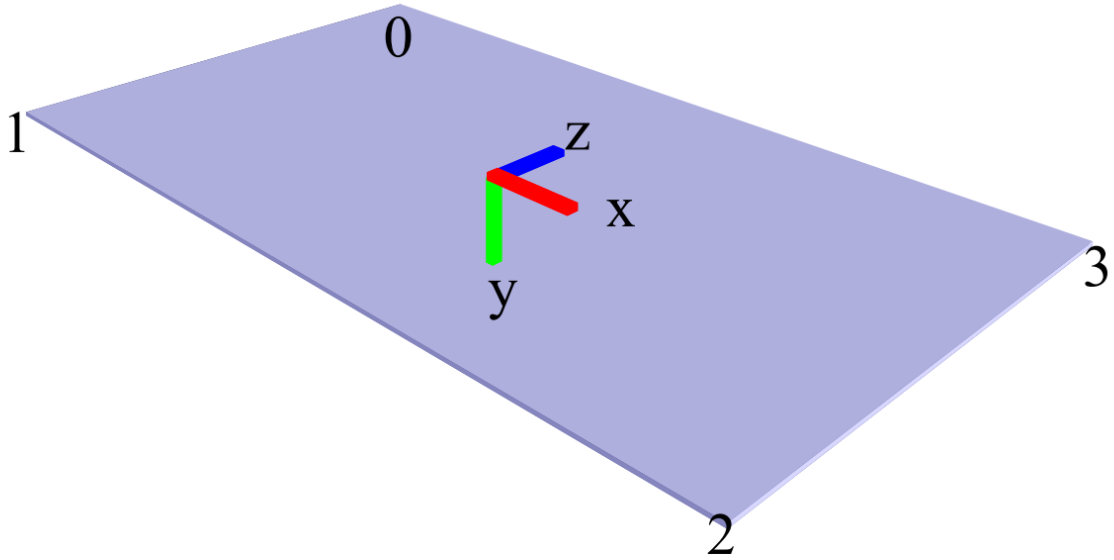


그림 7. 원점에 위치한 당구대 모델과 각 정점의 순서
Figure 7 Table at origin point and the order of table's vertexes.

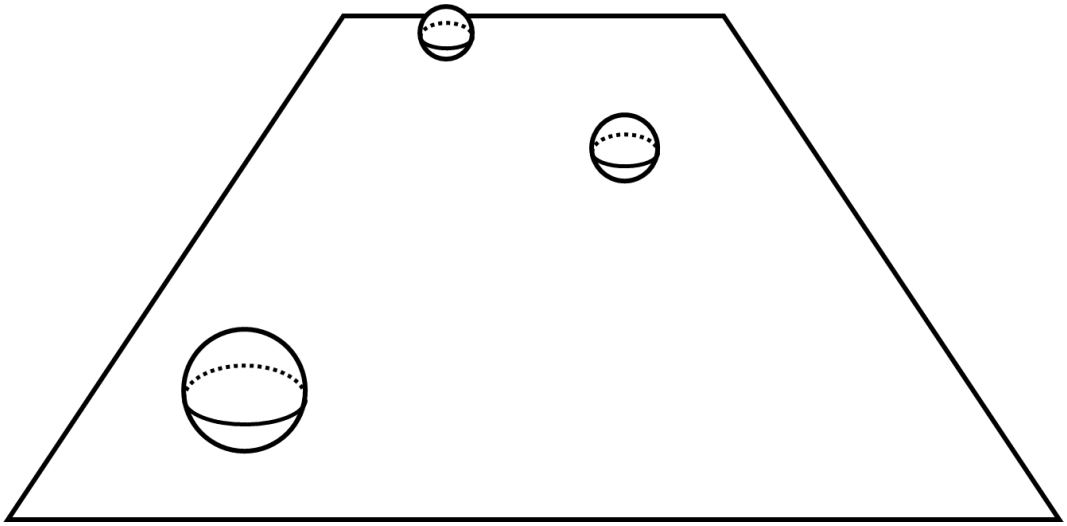


그림 8. 당구대 평면 상에 위치하는 당구공의 중점
Figure 8. Center of the billiard balls located on the table plane.

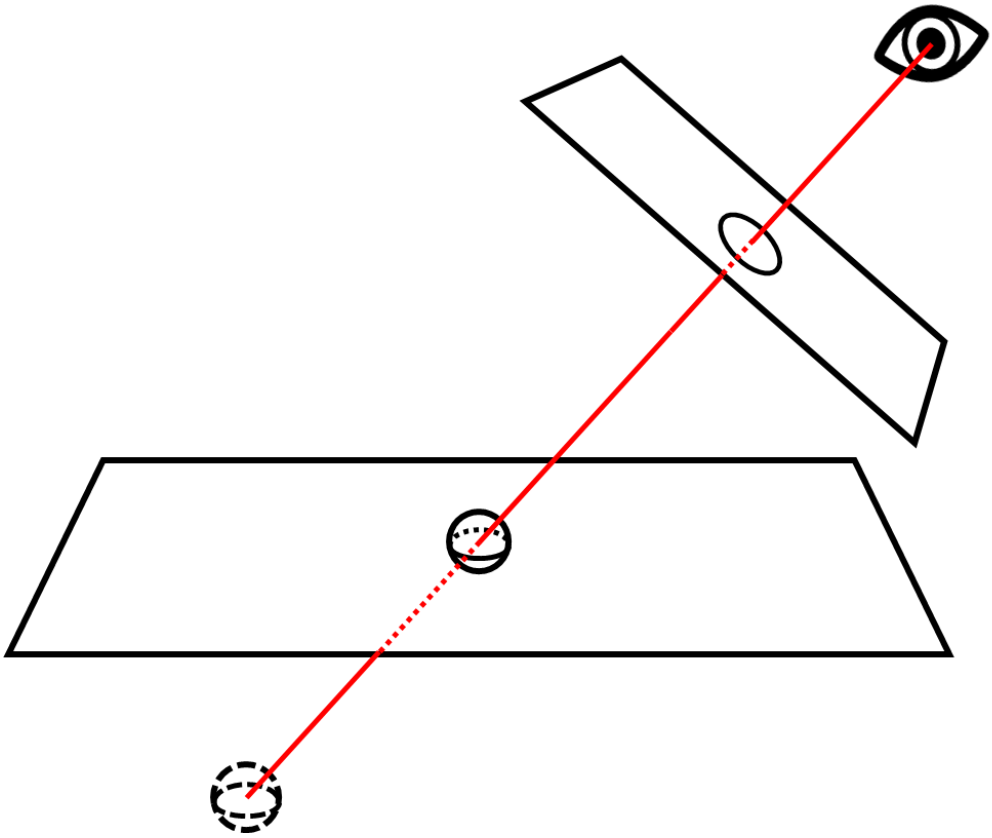


그림 9. 광선 투사를 통한 당구공의 중점 추정
Figure 9. Estimation of billiards ball center coordinate via ray casting.

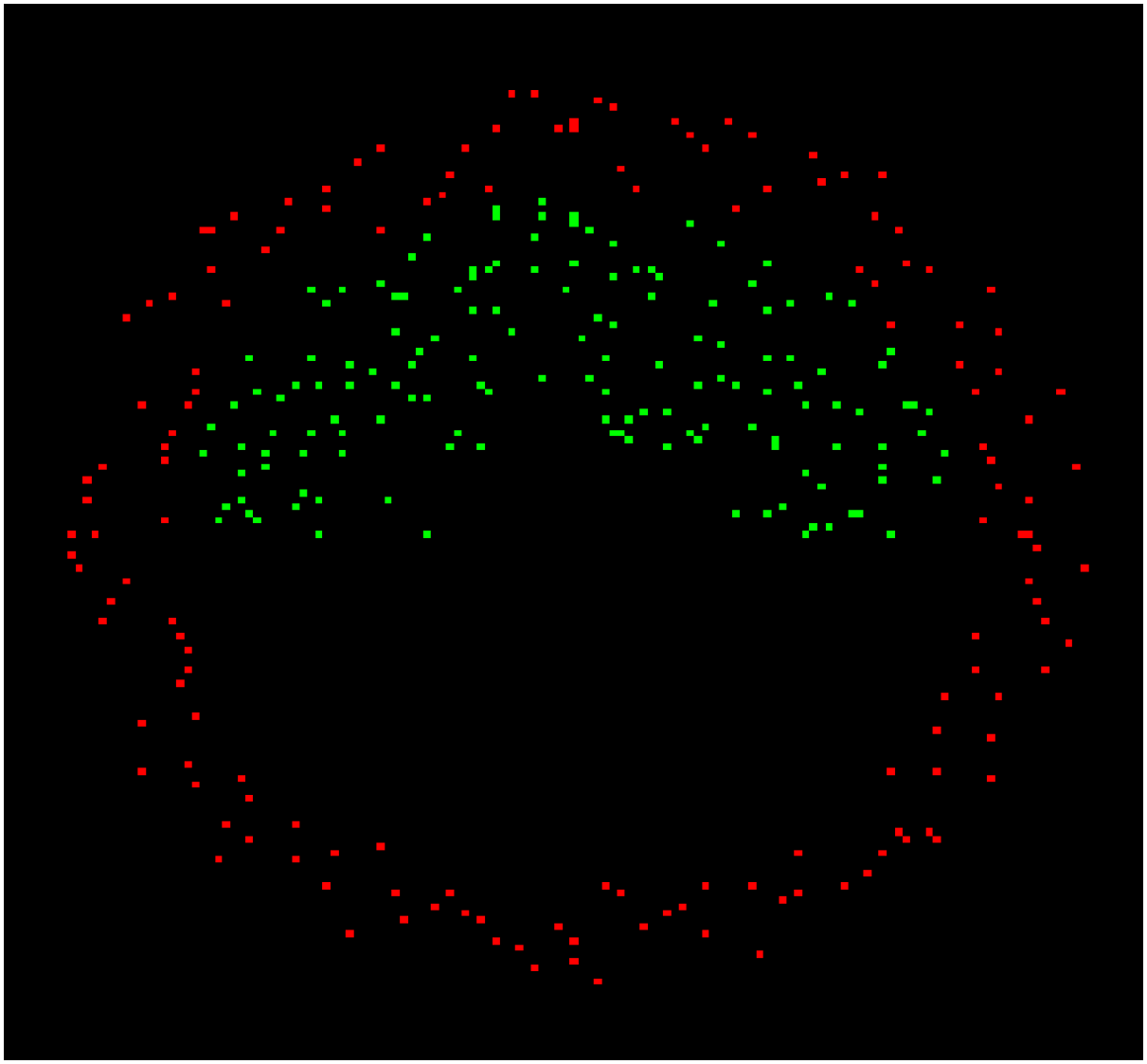
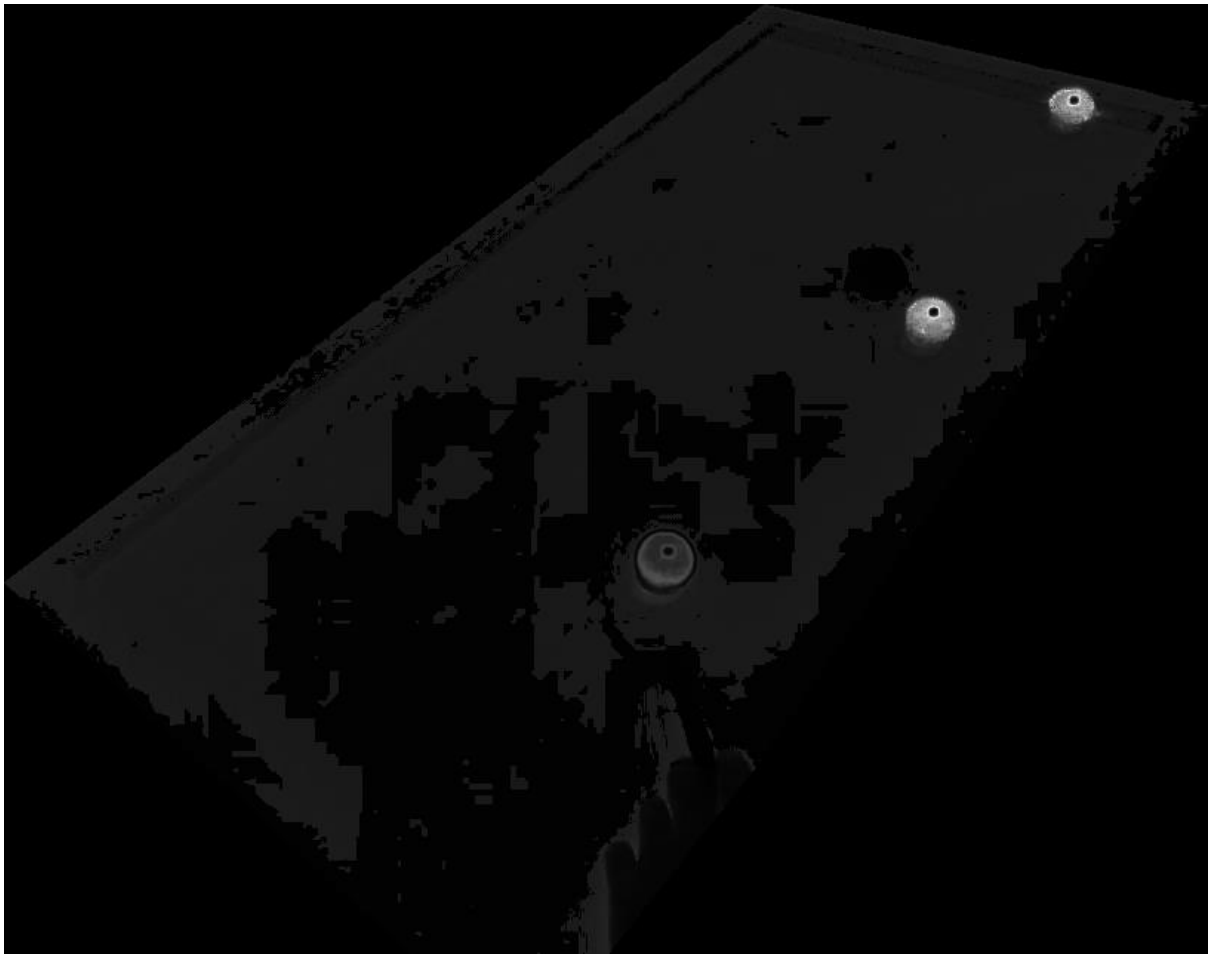


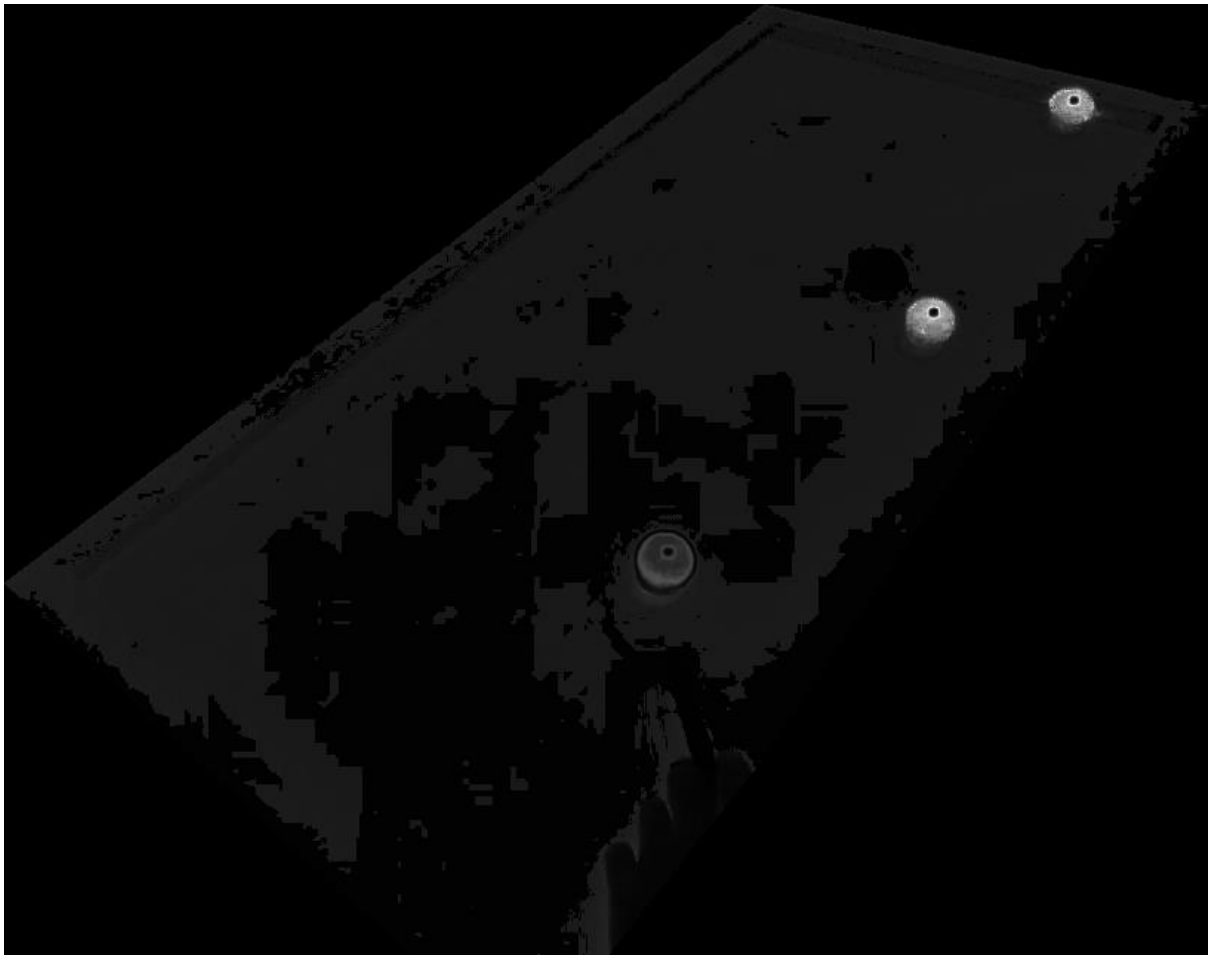
그림 10. 희소 템플릿 커널

Figure 10. Representation of Sparse Template Kernel.

그림 11. 공 인식 입력 이미지

Figure 11. Input image for ball detection.





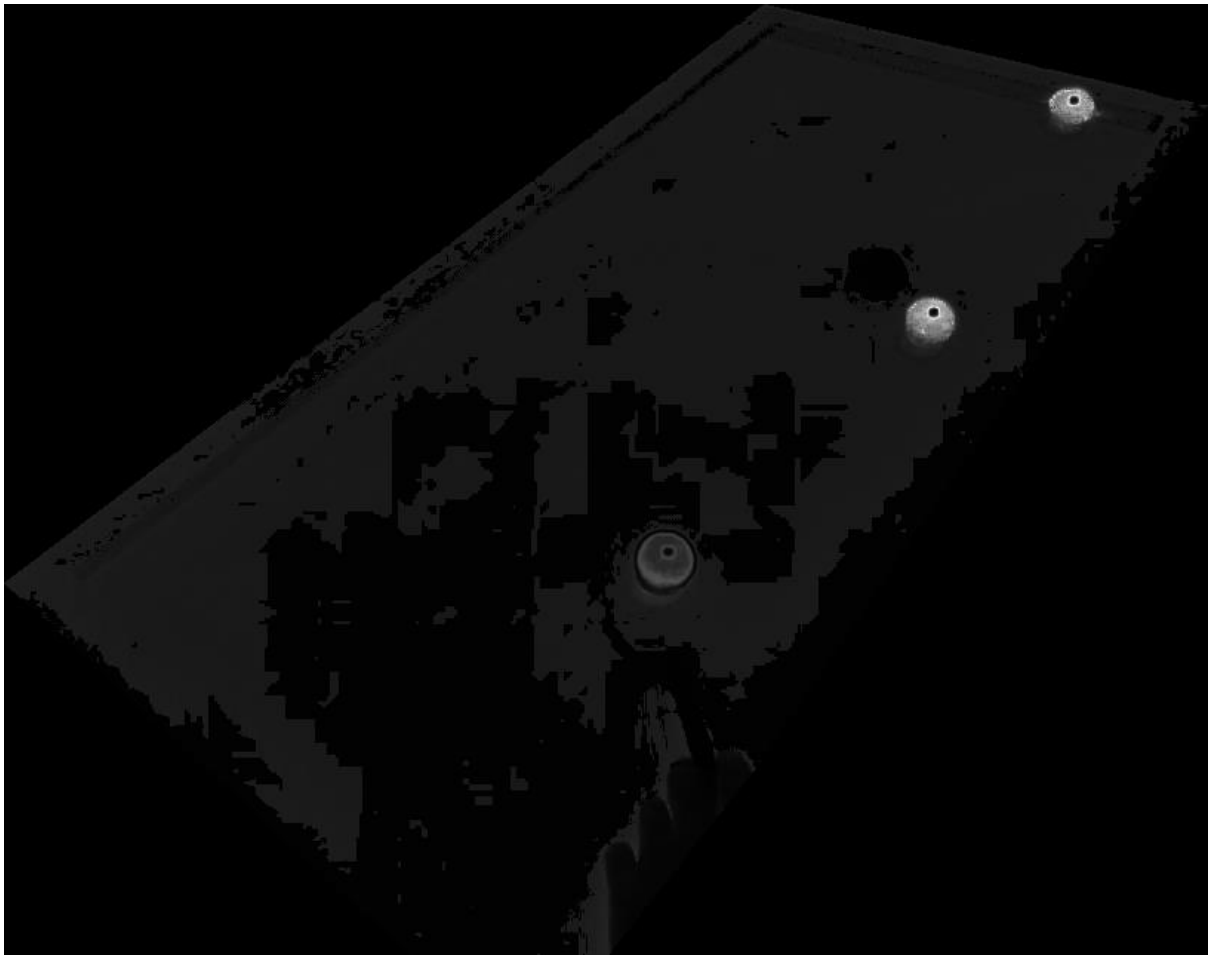


그림 12. 각각 빨강, 주황, 하양 공의 적합도 필드
Figure 12. Red, Orange and White ball's suitability fields, respectively.

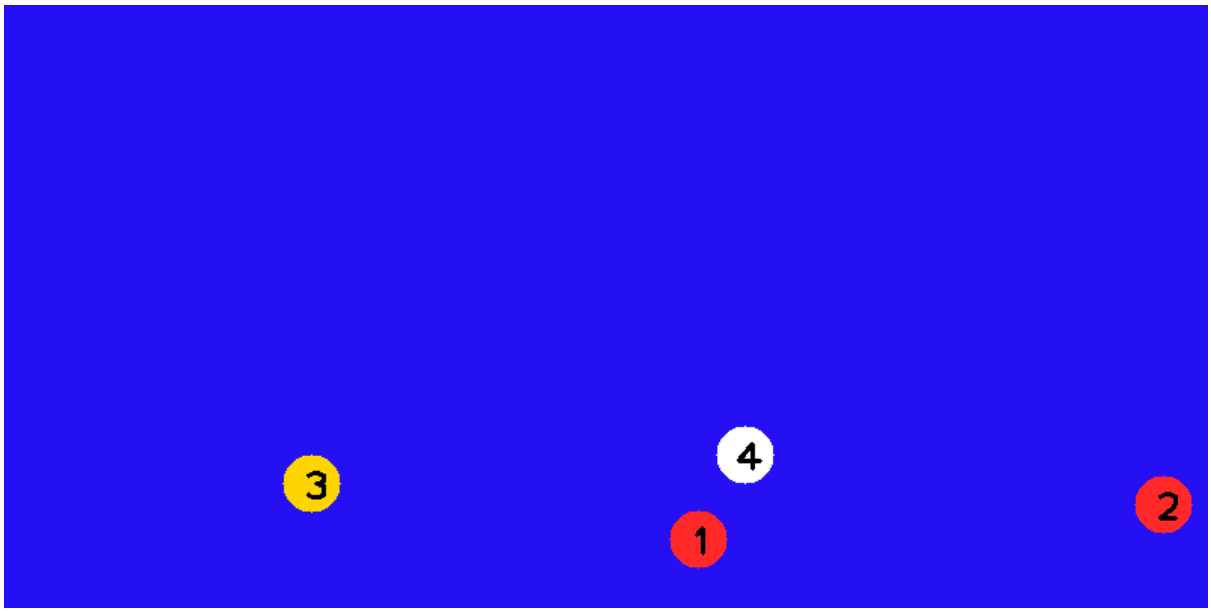
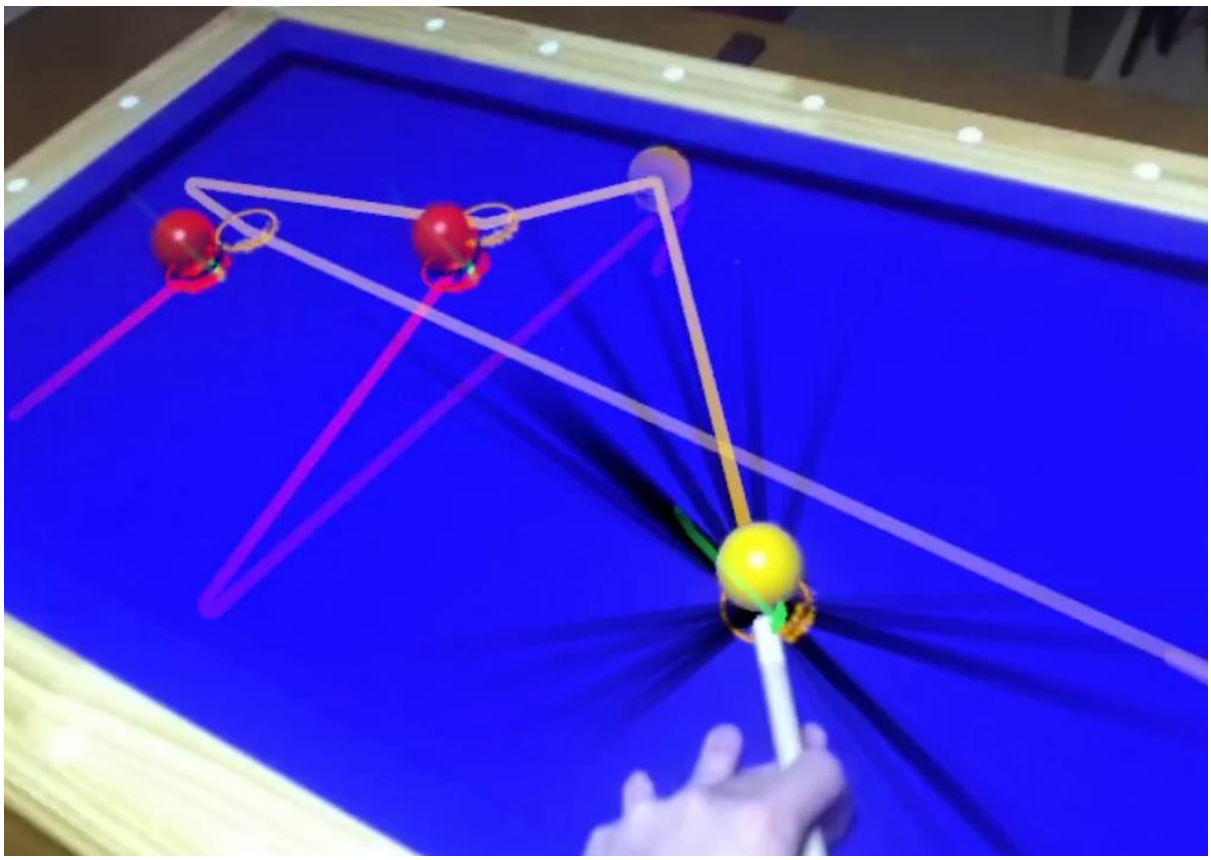


그림 13. 탐색된 공의 위치를 평면에 투영한 결과

Figure 13. Ball positions projected on billiard table plane.



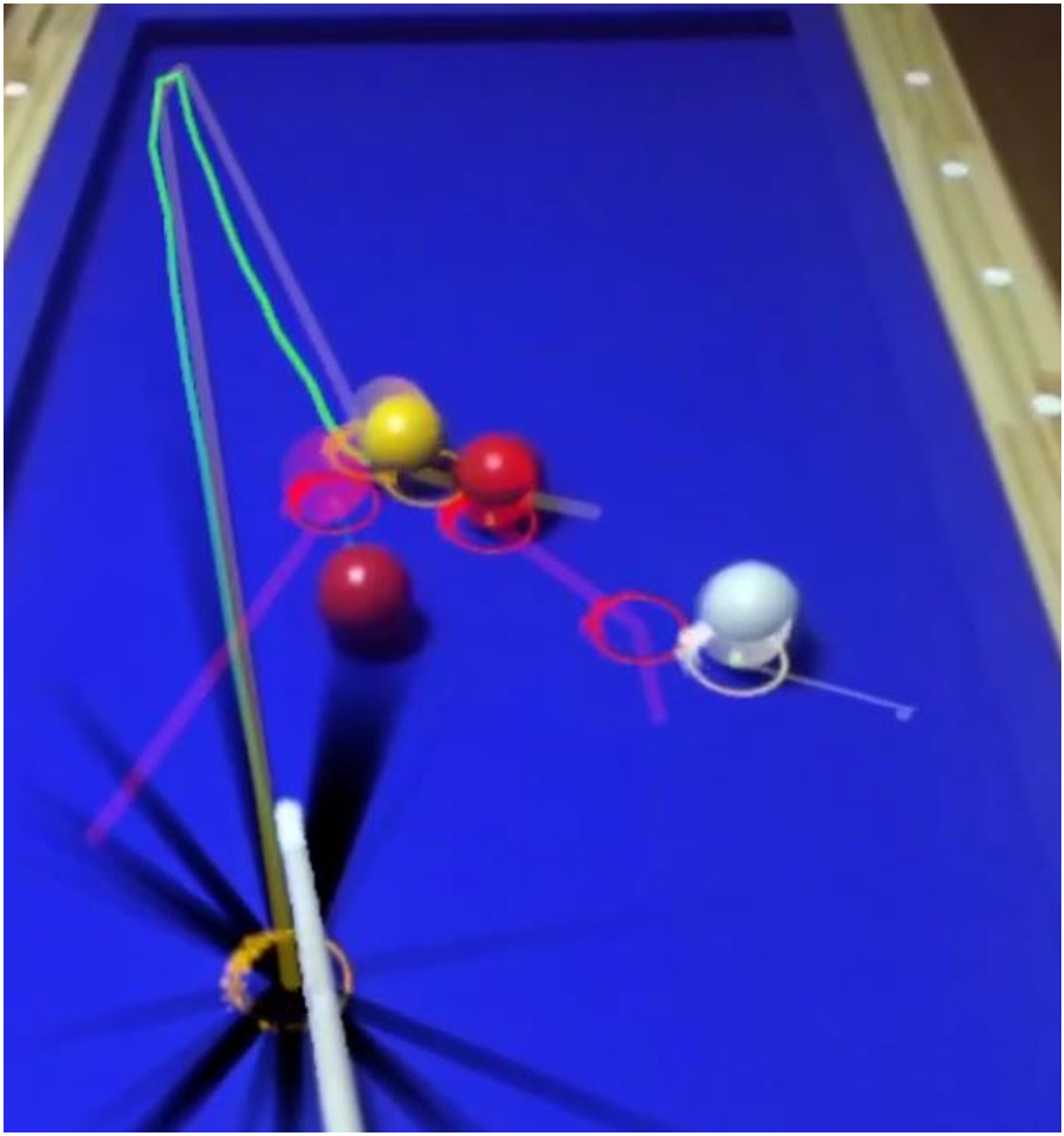


그림 14. 실제 구동 화면. 검은색 선은 득점이 가능한 방향을 모두 나타내며, 주황색 선은 사용자의 시야 각도에 따라 활성화된 예상 득점 경로이다. 초록 색의 궤적은 공이 지나온 자리로, 아래 이미지에서는 비교적 정확하게 궤적을 따라 득점하는 것을 볼 수 있다.

Figure 14. Actual footage. The black lines represents all directions in which scoring is possible, and the orange line is the predicted scoring path activated according to the user's viewing angle. The green trajectory is where the ball has passed, and in the image below, you can see that the scoring is relatively accurate.

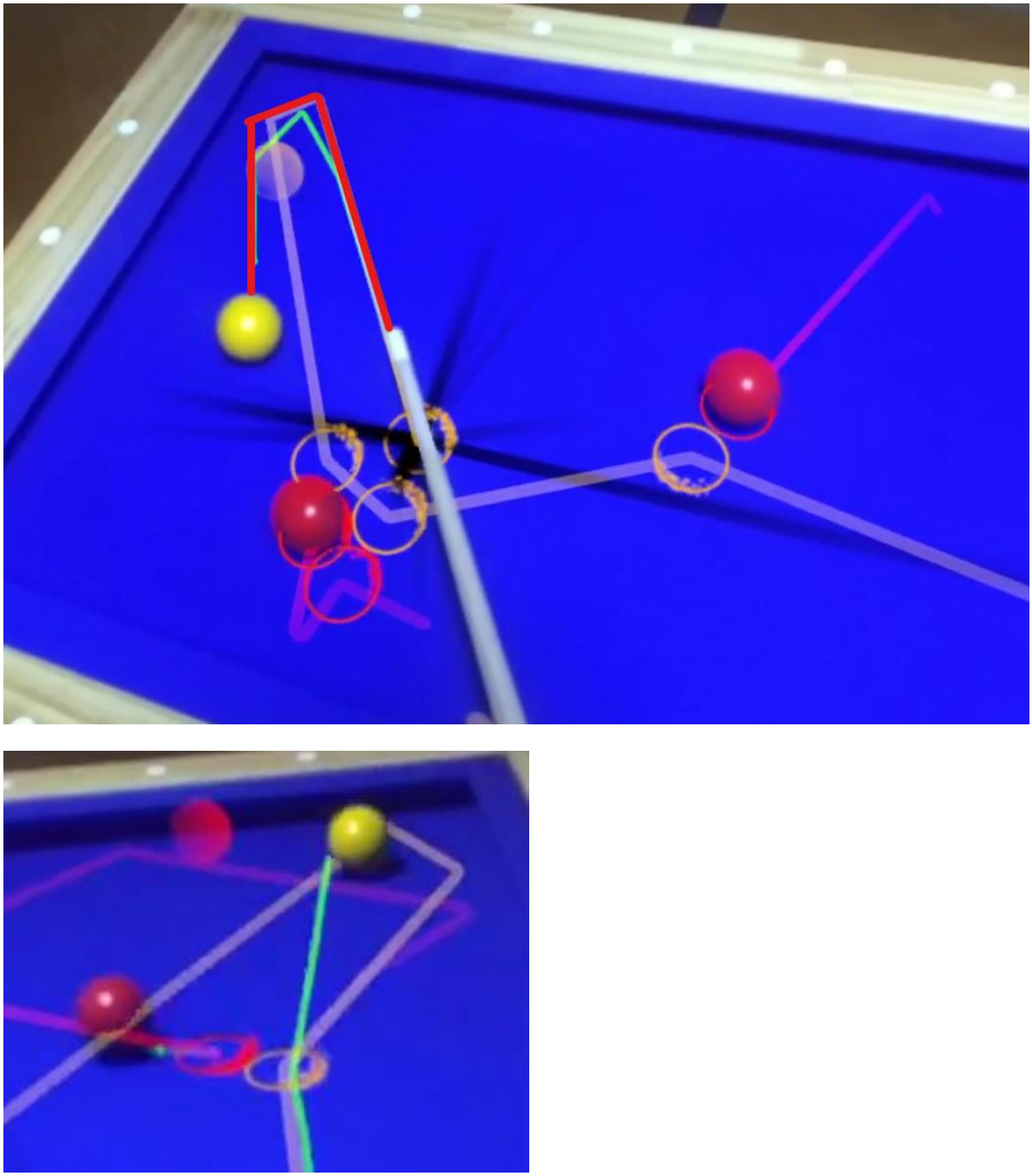


그림 15. 예측이 부정확하게 이루어지는 사례. 위의 경우 예측보다 실제 경로(빨간색 선)가 더 넓게 벌어졌다. 아래의 경우 주황색의 예측 경로보다 초록색의 실제 경로가 분리각이 더 작았다.

Figure 15 Examples of inaccurate predictions. In the above case, the actual path (red line) was wider than the prediction. In the case below, the actual path in green color had a smaller separation angle than the predicted path in orange color.

표의 사용도 그림과 동일한 형식을 이용하며 표의 캡션은 표의 상단의 위치시킨다. 표는 MS-워드에서 작성한 것이여야 하고 이미지로 가능한 대처하지 않도록 한다.

표 1. 당구공 인식 파라미터

Table 1. Parameters of billiard ball recognition

Ball color	$h_{r[0\sim180]}$	$s_{r[0\sim255]}$	$w_{h[\text{ratio}]}$	$w_{s[\text{ratio}]}$	w_n
Red	125	195	3	1	1
Orange	90	200	2	1	1.5
White	35	33	2	1	3

저 자 소 개

다음의 저자소개는 심사논문단계에서는 생략하며 최종논문 제출 시에 추가합니다.

사 진

성 춘 향 (Chun-hyang Sung) 정회원

1992년 2월 : 한국대학교 전자공학과 졸업

1994년 2월 : 한국대학교 전자공학과 석사

1996년 3월 ~ 현재 : 한국대학교

전자공학과 박사과정

<관심분야> 전자공학, 통신공학, 광통신 공학

사 진

김 길 동(Gil-dong Kim) 정회원

1992년 2월 : 한국대학교 전자공학과 졸업

1994년 2월 : 한국대학교 전자공학과 석사

1996년 3월 ~ 현재 : 한국대학교

전자공학과 박사과정

<관심분야> 전자공학, 통신공학, 광통신 공학