# Visual Aids for 8 Ball Pool Using Markerless Tracking

Samuel Bain
Dept. Mechatronics Engineering
University of Canterbury
Christchurch, New Zealand
swb54@uclive.ac.nz

Richard Green
Dept. Computer Science and Software Engineering
University of Canterbury
Christchurch, New Zealand
richard.green@canterbury.ac.nz

ABSTRACT

**This paper proposes a system to help aid beginner pool players using markerless tracking algorithms. The system primary role is to identify key features, such as the table edges, cue stick and white ball using their geometric properties as identifiers. The system then uses this information to accurately predict the path of the white ball until a collision is made. The proposed system provided positive results across a range of table felts, using a variety of Canny edge detection and Hough algorithms. Of the tested circumstances, the system was able to detect the table edges 87.5% of the time. The cue stick and white ball were identified across 86.1% of circumstances under ideal lighting conditions. The remaining balls were identified as accurately as 97.8% of times when tested on a standard green felt. This was considered sufficiently accurate to compute further supporting information.**

***Keywords* —** 8 ball pool, shot tracking, image projection, Hough lines, Hough circles, markerless tracking.

## I. INTRODUCTION

8-ball pool is a common game shared by professionals and beginners. Players aim to strike a white ball with a cue stick to impact and manoeuvre their designated balls into the six pockets.

8-ball pool is a difficult game for beginners and requires large amounts of practice in order to gain proficiency. Many beginner players struggle with lining up the cue stick in the correct orientation for the white ball to strike the receiving ball in the desired manner. This holds particularly true for shots where the receiving ball is far away from the white ball or when the player is required to rebound the white ball off the edge of the table in order to achieve the desired shot.

The game of pool has been widely adopted in the online space. Games such as 8 Ball Pool developed by Miniclip have received in excess of 500 million downloads on the Google Play Store alone [1]. The online game requires a lot less skill to become proficient, removing the aspect of hand-eye coordination and making it easier to perform more complex shots. One way in which the online version significantly simplifies the game is by projecting a line onto the table showing the path of the white ball, as shown in Fig 1.

Fig. 1: 8 Ball Pool online.

By introducing some form of visual aid for beginner players, it may be possible to entice more players to the real game. Furthermore, it may help all players in developing their skills and understanding of the game quicker.

This paper proposes a method to project visual aids on a pool table, to enable beginner players to visualise the results of their shots and increase their accuracy. The method involves non-intrusive tracking software so as to allow the software to be implemented on any pool table with minimal set up and calibration.

## II. BACKGROUND

### A. Camera Position

The position of the camera has a large effect on the system's ability to consistently provide accurate shot estimates. Two potential solutions have been used most frequently in prior research.

The first method commonly implemented by prior researchers was to capture the table from an arbitrary viewpoint, often implementing the use of wearable computers. This method has the advantage of reducing the need for a rig set up and alignment. The system developed by Hideaki Uchiyama and Hideo Saito [6] used this approach along with other researchers [3][10]. These researchers discussed the limitations of the method to only work using a tethered system and mentioned that the computation could not be completed efficiently using only onboard processing. This method also requires additional computation to perform 2D projections of the pool table, introducing additional sources of error to the system.

The second camera position commonly used is directly above the center of the table [8][9]. This position assures that the whole table is always within the bounds of the frame. This method also reduces preprocessing requirements by assuring the table is perpendicular to the direction of the camera. This significantly simplifies the geometric calculations for the ball path projection.

A further option used in prior research was a low-cost stereo vision approach of using two cameras [7], again mounted above the table. This approach allowed the researchers to define the 3D direction of the cue relative to the cue ball. This system could predict more information about the shot including spin, widening the application of the resulting projection to account for more cases.

### B. Table and Edge Detection

Detecting the edges of the table provides two benefits to the overall functionality of the program. Firstly, the software can ignore anything outside of the boundaries of the table, creating a more predictable environment. Secondly, the system can project the path of the ball after it rebounds off the cushion, meaning more complex shots can be displayed.

Most researchers used a colour segmentation approach in order to find the edges of the felt. As explained by Corey Bernard [3], erosion and dilation are then performed in order to fill in holes in the segmented area and to remove noise. As done by Jebara, Eyster, Weaver, Starner and Pentland [10], the system is often trained specifically for a pool table with green felt, limiting the application of the software to specific pool tables.

Another common option is to use a Canny edge detection algorithm followed by a Hough lines transform. Prior to the canny edge detection, a Gaussian blur is applied to the image and the resulting image is convert to grey scale. The canny edge detection algorithm works by taking the 8 adjacent pixels around a pixel in the image to get a 3x3 matrix of grey scale values [12]. A Sobel operator is used to calculate the gradient size in both the x and y direction, again expressed as a 3x3 matrix,

$$S_C = \begin{bmatrix} a_0 & a_1 & a_2 \\ a_3 & c & a_4 \\ a_5 & a_6 & a_7 \end{bmatrix} \tag{1}$$

By convolution of $S_C$ , the partial derivatives are expressed as the sums,

$$G_x = (a_2 + 2 * a_3 + a_4) - (a_0 + 2 * a_7 + a_6) \qquad (2a)$$

$$G_y = (a_0 + 2 * a_1 + a_2) - (a_6 + 2 * a_5 + a_4) \qquad (2b)$$

Using Pythagoras and trigonometry, the magnitude and direction of the gradient are extracted. Thresholds are used in order to disregard gradients below a certain intensity, and the Hough lines algorithm searches for adjacent points perpendicular to the gradient line that have the same gradient direction. The shadow caused by the slight overlap of the cushion on the playing area creates a distinct gradient, most easily identified when the lighting source is perpendicular to the playing area.

This method was adopted by Christian Spain [2] in conjunction with a Sobel filter in the x and y directions in order to solely extract the edges perpendicular to the frame.

The final method used by other researchers was to prompt the user to select the table corners themselves [11], eliminating the risk of a false reading. However, this method does require the user to initialize the program each time, requiring additional user input.

### *C. Ball and Cue Detection*

Previous researchers have used an HSV colour segmentation in order to distinguish the balls from the surrounding felt, exploiting the shiny nature of the balls [2][3]. A potential issue with this was that some felts have high saturation values that rival that of the balls. Hence, the felt was indistinguishable from the balls in an HSV colour space in some cases.

Another method adopted by other researchers to identify the balls was to use a differencing algorithm from the initial frame [4]. This method introduces major flaws if the system is implemented once gameplay has already begun, which is unideal if users begin capturing the video midway through a game. This method is also susceptible to errors if the lighting conditions change throughout the game - i.e. If the user dims the light in the room. This would cause the entire frame to be included in the final difference.

### *D. Limitations*

The greatest limitation of current research is the limited application of programs to solely a green felt table, due to the use of HSV or RGB colour segmentations [2][3][10]. This program aims to remove this limitation by no longer relying on presumed colours for identification, whilst still upholding the ability to complete markerless tracking.

## III. Method

### *A. Aims and Assumptions*

The goal of this paper was to improve on the accuracy of previous research into Visual Pool Aids using marker-less tracking. The program should be able to:

- Correctly detect table edges.
- Correctly identify all balls present on the table for contact predictions.
- Correctly identify the direction of the cue.
- Display the predicted path of the white ball up to first contact with a coloured ball.
- Be applicable to a range of pool tables.
- Outperform accuracy from method outlined by Christian Spain, University of Canterbury [2].

The program works under the following assumptions:

- The camera is mounted above the centre of the table and has a view of the whole table.
- The edges of the table are perpendicular to the edges of the frame.
- The cue stick is not largely obstructed by the player.
- The game is performed under optimal lighting conditions.
- The player strikes the white ball in the centre, generating no spin.

### *B. Camera Setup and Test Footage*

The method described in this paper assumes the camera to be mounted directly above the table. This was chosen in order to simplify the algorithms and to allow for future iterations of the system to easily project the predicted path back onto the table from above.

The test footage used was from an online pool simulator. This was chosen over a real pool table as it was the easiest way to simulate ideal lighting conditions and reduced the need to worry about camera mounting and calibration. Using an online simulator also allowed for the system to be tested on multiple different felt colours. This meant the extent of the system could be more thoroughly tested.

### *C. Edge Detection*

For the edges of the table to be detected, the image was firstly blurred and converted to grayscale. This helped to reduce the effects of shadowing on the cushions. The edges were then extracted from the converted image using a simple Hough lines algorithm, whereby a Canny filter detects sharp gradients within the image and the Hough lines transform connects adjacent points to form edges. A large maximum gap between adjacent points was used in order to ensure the edges extended beyond the centre pocket. As shown in the

Fig. 2, the Hough lines transform still detects multiple lines around the perimeter of the table as well as the cue stick.

*Fig. 2: Hough lines transform for detecting potential table edges.*

A similar approach was used to extract the table edges to that of Christian Spain [2], whereby any lines not parallel to the edges of the frame are ignored and the inner most edges are used for each side of the table.

### D. Detecting The Cue and White Ball

The direction of the cue was essential towards creating the path projection for the white ball. Again, a broad Hough line transform was applied to the image in order to extract potential edges of the cue stick. Similarly, potential white balls were identified using a Canny filter in conjunction with a Hough circles transform. A Hough circles algorithm was chosen over a colour segmentation approach to reduce the susceptibility to errors caused by shadows and changing light conditions. The potential cue edges and white balls are depicted in Fig. 3.

*Fig. 3: Potential edges and white balls using Hough transforms.*

As can be seen from Fig. 3, the system has identified two potential white balls, one being the true white ball and the other being the yellow solids ball. From this, a novel approach was used in order to confirm which circle and edges should be used to identify the cue stick and white ball. The system exploits the fact the when a player is preparing to play a shot, they place the end of the cue stick close to the white ball and aim the cue stick towards the white ball. Hence, the cue stick acts as a makeshift pointer for the system to identify which ball is the white ball. This has the additional bonus that the system only attempts path prediction when the player is lining up their shot, and therefore will not project path prediction when otherwise unintended. The angle of the two edges of the cue stick were averaged in order to find the resulting cue direction.

### E. Detecting Coloured Balls

The coloured balls were also detected using a Canny threshold and Hough circles algorithm. This was chosen over the more common methods of HSV colour segmentation and differencing algorithms due to their notorious susceptibility to changes in lighting and shadows. The colour segmentation method also had limited accuracy on some felt surfaces, which vary widely for different pool tables. The selected algorithm detected all the balls on the table including the white ball. The white ball was removed by superimposing the white ball position found in the section above. This meant the predicted path would not assume that the white ball will collide with itself.

### F. Overlaying the Predicted Path

The path of the ball was predicted up until one of two conditions were met:

- The white ball was predicted to collide with a coloured ball.
- The white ball had hit the cushion three times.

The collision specification was implemented as the most important section of the white ball path is up until the first collision, and errors in the ball path are amplified beyond the first collision. The cushion specification was implemented as shots rarely require more than two cushion rebounds, and players are limited by their supplied power. Also, the longer the path of the ball, the larger the resultant error would be for the balls end position. This meant the system would have limited application beyond this distance due to the tolerances in the system.

The path of the ball was first calculated to the table edge the cue is facing. The intersection point was found between the balls centre and a line offset from the table edge by the radius of the ball. Once this path was found, the system then attempted to detect the earliest collision with balls along this path. This was achieved by overlaying lines propagating from the centre of each colour ball by a ball diameter in either direction perpendicular to the path of the ball, as depicted in Fig. 4.

*Fig. 4: Collision detection algorithm for white ball with coloured balls.*

If the predicted ball path did not intersect with any of the other balls, the new ball angle was calculated from the interaction with the table edge.

If a collision was detected, a new algorithm was run in which determined the position of the white ball upon contact with the coloured ball using geometric principles. Finally, the direction of the coloured ball was estimated using conservation of linear momentum, as derived by rld-physics-problems.com [5] and depicted in Fig. 5.

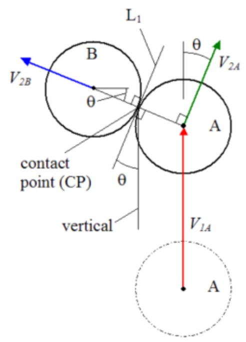


*Fig. 5: Trajectory of pool balls after a collision using conservation of momentum [5].*

The final result is shown in Fig.6.


*Fig. 6: Shot projection onto pool table.*

## IV. Results

### A. Apparatus

- Device: Lenovo™ Ideapad™ 710s Laptop
- OS: Windows 10
- Processor: Intel® Core™ i7-6500U CPU @ 2.50GHz
- IDE: Wing 101 7.1
- Language: Python 3
- Footage: Virtual Pool Game

### B. Results and Discussion

In order to directly compare against previous research, the system was tested against three main criteria: Locating the table edges, locating the cue and white ball, and locating the coloured balls.

Initially, a simple test was run in order to see if the system could correctly identify the table edges for different felt colours. Although minor offsets due to shadows were allowed for, an edge detection on the edge of the felt was counted as a false reading. The results are displayed in Table I.

Table I: Number of table edges detected for each felt colour

| Felt Colour | Table Edges Identified |
|---|---|
| Green | 4/4 |
| Blue | 4/4 |
| Red | 1/4 |
| Coffee | 4/4 |
| Dark Burgundy | 4/4 |
| Dark Cyan | 4/4 |

The program was able to successfully find the right cushion across 87.5 % of circumstances. The only felt colour in which the edge detection algorithm struggled was with the red felt. In this case, the edges of the felt were detected as opposed to the cushions. It was unclear why the algorithm was not able to detect the table edges for the red felt using the same Canny Edge Detection thresholds. However using different threshold values allowed for the edges to be found correctly.

The second test was to see what percentage of the times the proposed system was able to correctly identify the cue and white ball. The test was run on the two most common pool felt colours. Footage was taken of a player taking five different shots from various points on the table, and the percentage of frames in which the cue stick and white ball were identified were recorded. The results are shown in Table II.

Table II: Cue and White Ball Detection Results

| Felt Colour | Test 1(%) | Test 2(%) | Test 3(%) | Test 4(%) | Test 5(%) | Ave. |
|---|---|---|---|---|---|---|
| Green | 67.47 | 100 | 81.82 | 100 | 87.23 | 87.3% |
| Blue | 71.43 | 80.23 | 90.77 | 92.65 | 90.00 | 85.0% |

The program was able to correctly identify the cue stick and white ball 87.3% of times on the green felt and 85 % of the times on the blue felt. The times in which the program struggled the most was when the white ball was near the edge of the table and the cue stick was mostly over the table edge. As the table edges and cue stick were of relatively similar colour, the gradients in greyscale values were minute, hence the Hough lines algorithm failed to recognise the cue in some cases.

The final test was to see how many of the ball could be identified so as to allow the program to consistently predict ball collisions. The same five test shots were used for each felt colour as above. The percentage of balls found were averaged below in Table III, along with the percentage of frames in which the white ball was properly excluded.

Table 3: Coloured ball detection results

| Felt Colour | Percentage of balls correctly identified | Percentage of frames where white ball excluded |
|---|---|---|
| Green | 97.8% | 81% |
| Blue | 91.5% | 83.6% |

The Hough circles algorithm was able to work extremely well on the green felt, resulting in a ball detection rate of 97.8%. The stripes on the balls were expected to be of concern when placed at varying orientations, however this had little effect on the algorithms ability to detect the balls.

On the blue felt, 91.5% of balls were detected correctly. It was expected that this result would be slightly lower than that of the green felt as the Hough Circle thresholds were tuned to work optimally on the green felt.

Of the times that the program failed to detect a ball, 80% were due to the cue and white ball detection algorithm mistaking one of the yellow balls for a white ball, hence excluding it from the coloured ball detection. The other 20% of times were when the ball was right in the corner pocket.

### C. Run time

The run time of the program was calculated to be on average 0.203 s per frame for only identification, and up to 0.213 s per frame with the addition of the path prediction algorithm. This is relatively slow for the simple level of computation, however, is sufficient for the application as this would allow four frames per second to be overlayed in real time. The program speed could be significantly reduced by optimising some of the detection algorithms, however this was beyond the scope of the paper.

### D. Comparison

The main aim of this paper was to develop a system that outperformed the accuracy of previous research, namely that of Christian Spain, University of Canterbury [2].

Christian Spain used similar Hough line algorithms in order to detect the edges of the table, however with the addition of a Sobel filter. This resulted in a success rate of 79% as compared to 85% for the method proposed in this paper. The main improvement however was that using the proposed method, all four edges were detected in 5/6 of the tests, as compared to 1/6 in the opposing method. Other methods, such as that of Akira Suganuma and Fumiya Taka [11], had theoretical success rates of 100% for table edge detection, however required the user to manually select the table edges each time they played.

The proposed system was able to correctly identify the cue and white ball 86.2% of the time across two felt colours, as opposed to 67% of the time in Christian Spain's method using only one colour of felt. The novel approach of using the cue stick and white ball to cross check the identifications meant broader Canny thresholds could be used for detecting potential white balls and cue stick edges. This resulted in the system being able to detect the objects in a wider range of conditions.

A further improvement on the method is that of Tyohei Takita, Noriaki Kashimoto and Masanobu Takahashi [7], to whom developed a way of finding the cue direction using a colour segmentation to find the white tip of the cue stick. Using this method, the team of researchers were able to find the cue stick in 100 % of the test cases. Furthermore, from a stereo image, the team of researchers were able to extract the 3D cue direction, with an average error of only 0.77 degrees.

The final comparison was of the coloured ball detection algorithm. Christian Spain sourced the ball detection algorithm from Corey Barnard [3], in which an HSV filter distinguished the balls from the rest of the table. This method resulted in a success rate of 68% from an arbitrary viewpoint, and 93.8% from a fixed viewpoint above the table. The proposed method achieved a success rate of 94.3% correct ball detections using two different table felts. This would not be achievable using the system proposed by Corey Bernard as the HSV thresholds would have to be altered for the different felt colour, whereas the Hough circles algorithm works relatively similarly for both felt colours. Unlike the method proposed by Corey Bernard, the proposed method had no trouble detecting balls that were in contact with one another. The opposing method used a simple erosion and dilation of the ball segmentation in order to reduce noise, however had the unintended effect of concatenating touching balls into a single object, resulting in a success rate of only 26.7 % in these cases.

## V. Conclusion

### A. Outcomes

The proposed method was able to achieve a relatively high success rate of detecting the table edges, cue stick and balls, using simple algorithms that did not constrain the system to certain felt colours or lighting conditions.

Cushion detection was achieved across 87.5% of circumstances tested using a Hough Projection, across a total of six different felt colours. The only felt colour in which the algorithm failed to detect the table edges was red, and the origins of this failure was unclear. However, the red table could still be detected if the Canny thresholds were adjusted slightly, so was not of major concern.

The cue stick and white ball were correctly detected 86.1% of times, using recordings from two different table colours. The Canny edge detection algorithm failed when only a small segment of the cue stick was shown in the frame, and the cue stick was overlaying the table edge. This was due to the insignificance of the gradient in greyscale value between the cue stick and table edge, both of which are made of wood. This small gradient was filtered out as it was below the edge detection threshold, introduced to reduce noise in the detection algorithm.

The coloured balls were able to be detected in 97.8% of the tested frames on the green felt, and 91.5% of the time on the blue felt. This was considered to be relatively high considering the simplicity of the Hough circles algorithm. The white ball was removed from the ball detection by doing a subtraction of any balls detected within the radius of the white ball. This was achieved 82.3% of the time across all tests, with failure being mainly accounted to the white ball not being detected using the previous algorithm.

### B. Future Work

The above method was tested using ideal images with optimised lighting conditions. Future work could be implemented in order to increase the application of the system. Such improvements could involve a shadow detection and removal algorithm, such as that described in the recent study by Young-Choon Kim, Tae-Wuk Bae and Sang-Ho Ahn [13].

The method was also tested without any people in the frame, which requires the unrealistic expectation for players to not lean across the table while taking a shot. The addition of a person in the frame would most likely affect the system's ability to detect the cue stick. Therefore, a detection algorithm that detects only the tip of the cue stick, such as that described by Tyohei Takita, Noriaki Kashimoto and Masanobu Takahashi [7], could help to mitigate these new issues.

The overall usability of the program could be greatly increased. As the main goal of this paper was to achieve high detection rates of the cue stick, balls and table edges, the path prediction algorithm was not prioritised. Future work would involve removing any bugs present in the path prediction algorithm and reducing the error in the angle of the projected line. A further improvement on the system would be to have an algorithm to advise the player what ball they should pocket next, such as that described by T. Nierhoff, O. Kourakos and

S. Hirche [14]. Alternatively, the algorithm could be paired with interactive training programs to tailor the system towards professional players, such as those described by L. Larsen, R. Jensen, K. Jensen and S. Larsen [15]. However, the current system as it stands is still considered sufficient to aid beginner players through simplifying the game of pool.

# VI. References


[1] "Google Play", 13 April 2021. [Online]. Available: https://play.google.com/store/apps/details?id=com.miniclip.eightballpool&hl=en&gl=US. [Accesed 22 April 2021]

[2] C. Spain and R. Green, "Simple guidance system for novice pool players," Department of Computer Science and Software Engineering, Unviersityof Canterbury, Christchurch, 2020.

[3] C. Barnard and R. Green, "Estimating positions of pool balls," Department of Computer Science and Software Engineering, Unviersity of Canterbury, Christchurch, 2014.

[4] C. Gray and R. Green, "Statistics tracking for eight-ball players," Department of Computer Science and Software Engineering, Universityof Canterbury, Christchurch, 2013

[5] "Real World Physics Problems", 25 April 2015. [Online]. Available: https://www.real-world-physics-problems.com/physics-of-billiards.html#:~:text=The%20physics%20behind%20billiards%20(or,the%20collision%20is%20nearly%20elastic.&text=For%20collisions%20between%20balls%2C%20momentum,like%20in%20any%20other%20collision). [Accessed 15 April 2021]

[6] H. Uchiyama and H. Saito, "AR Display of Visual Aids for Supporting Pool Games by Online Markerless Tracking," 17th International Conference on Artificial Reality and Telexistence (ICAT 2007), 2007, pp. 172-179, doi: 10.1109/ICAT.2007.35.

[7] K. Takita, N. Hashimoto and M. Takahashi, "Support system for pocket billiards — Estimation of 3D cue direction," 2012 Proceedings of SICE Annual Conference (SICE), 2012, pp. 2001-2004.

[8] C. Shih and W. C. Chu, "A Vision Based Interactive Billiard Ball Entertainment System," 2007 First IEEE International Workshop on Digital Game and Intelligent Toy Enhanced Learning (DIGITEL'07), 2007, pp. 200-202, doi: 10.1109/DIGITEL.2007.7.

[9] J. Alderman, M. Kappor, Y. Nakhimovsky andC. Cope, "BankShot: A Visualization Aid for the Game of Pool," 2006

[10] T. Jebara, C. Eyster, J. Weaver, T. Starner andA. Pentland, "Stochasticks: Augmenting thebilliards experience with probabilistic visionand wearable computers," MassachusettsInstitute of Technology, Massachusetts, 1997.

[11] Suganuma and F. Takata, "A Supporting System for Beginners of Billiards Based on Predicting the Path of the Balls by Detecting the Cue Stick," *International Journal of Innovations in Engineering and Technology, 2014*

[12] L. Q. Tan and B. He, "The Research of Implementation Method of Canny Edge Detection of Video on FPGA," 2020 International Conference on Information Science, Parallel and Distributed Systems (ISPDS), 2020, pp. 1-4, doi: 10.1109/ISPDS51347.2020.00008.

[13] Y. Kim, T. Bae and S. Ahn, "Background subtraction with shadow removal using hue and texture model for moving object detection," 2020 International Conference on Electronics, Information, and Communication (ICEIC), 2020, pp. 1-2, doi: 10.1109/ICEIC49074.2020.9051164.

[14] T. Nierhoff, O. Kourakos and S. Hirche, "Playing pool with a dual-armed robot," 2011 IEEE International Conference on Robotics and Automation, 2011, pp. 3445-3446, doi: 10.1109/ICRA.2011.5980204.

[15] L. Larsen, R. Jensen, K. Jensen and S. Larsen "Development of an automatic pool trainer," ACE '05: Proceedings of the 2005 ACM SIGCHI International Conference on Advances in computer entertainment technology, June 2005, pp. 83–87