Initial Project Report of

Computer vision (COMP9517)

**Xiao Wang, z3480870**

**Hang Jiang, z5034230**

**Introduction**

We are pool ball fans, so we decided to apply the computer vision tech to this field, which is an important motivation of our project.

In this project, we would do:

* Recognition and Tracking: identify and track different pool balls in the pool game video
* Projection and Animation: project the real pool game onto 2-D plain board , simulate the process of the ball game. This could be used for different purposes, such as pool game analysis, visualization, training, automatic scoring, and so on.

**Analysis**

The pool games seems a pretty good context for applying computer vision, since the objects’ features are quite unique.

There are quite a few sorts of pool game, using different balls and rules. For our project, we mainly focused on 9-ball game. Also, we assume camera is static, and light is consistent.

To achieve the final goal, we can divide our project into different sub-tasks:

* Detect the pool table in the video. We need to identify the pool table area, especially some key points such as 4 corner and maybe pockets.
* Match pool table in the video to the 2D animation board. This will help us to find the homography – it will be used in the projection and animation.
* Detect balls. Recognize the ball objects.
* Identify balls. Our application need to tell exactly which number of it is for each detected(or ball candidation) ball.
* Tracking the ball in the real-time video.
* Project balls to animation board

Also, we need to consider some performing requirements below:

* the accuracy of the ball recognition.
* The efficiency of real-time tracking.

Actually, we need to find the suitable balance between these two aspects.

**Design and Implementation**

### System Overall

The programming language for this project is mainly python. We will design and implement a simple and easy interface to guide user, and output the important data. (console data, image, video, etc)

* Pool Table Detection

We provide two methods to detect the pool table, one is through a manual setting up step, the other is automatic detection function. For manual setting, we provide a snapshot of pool game to make user to determine the 4 corners via mouse click. For automatic detection part, we choose to use **Color Contours**, since the table color and light are quite unique to background. The main process is: apply mask (color range) to original image → find the **convex hull** of the table (p.s: convex hull can avoid the effect of pockets/holes) → use **Hough Transform** to find the best rectangle to cover the table. (cv2.approxPolyDP) → record 4 corner points for further usage. (homography for projection).

* Pool Ball Detection and Identification

It’s the core and difficult part, as we tried different kinds of method.

First we need to choose the color space standard. We adopt HSV instead of RGB, because it’s much more convenient to determine the range of color. (for one ball’s possible colors, the distribution is too dispersed in RGB)

For the most part we use **Color Range Mask** + **Hough Circle Detection.** However, Even after attempting adjust the parameters for many times, the result is still bad, either miss the target or generate too many false positive matches.

Now we are using **Blob Detection + Color Range Mask**, and the result is quite promising. The rough process is: apply blob to find small segments that contains one or more possible ball-objects. → apply color range mask, to do a further analysis to detect and identify the ball.

* Pool Ball Dynamic Tracking

Though the above step, we can recognize the balls in one frame (static image), but for real-time tracking we need to consider the efficiency. Further more, we can analyze the movement status to adjust our detection and improve the performance, both on efficiency and accuracy.

### Ball Trajectory Projection and 2D Animation

Use Homography data acquired in table detection to do the transformation of video position to animation board position. Draw basic ball shape and trajectory

**Milestones**

1. Have successfully identified pool table.

2. Have detected pool balls and partly identified pool balls. But sometimes it fails when the HSV color histogram of two pool balls are very similar in certain case of light situation, the balls are laying close to each other. And currently we haven’t done this part for striped balls.

3. Have basically tracked pool ball trajectory in real time. But the efficiency need to be improved by slightly changing the ball identification logic in real time.

4. Can do basic projection and display. But the interface need to be more pretty.

**Evaluation**

We will evaluate our working mainly by two difficult parts.

1. The accuracy of identifying pool balls. Basically, we would take some sample pool game videos, check sequences of frames, calculate the number of correct identification and the number of incorrect identification for specific pool balls and check its correct/incorrect distribution to get the identification accuracy. Furthermore, if it is feasible to get the true ball center, we would calculate the distance between true ball center and identified ball center using Euclidean distance to get the pool ball center identification accuracy. Could use **confusion matrix** to evaluate our ball identification performance.

2. The efficiency of real-time pool ball tracking. We would try to reduce the computation to identify pool balls for each frame as much as possible to improve the efficiency by modifying the algorithm. Basically, if the accuracy of identifying pool balls is higher, then the efficiency of ball tracking may be affected more, since it need to more computation. Thus, we would try to trade-off the pool ball identification accuracy and the real-time poll ball tracking efficiency.

**Further Work**

1. Ball Detection & identification accuracy
2. Real-time tracking efficiency
3. Projection and animation interface
4. Testing and evaluation

**Reference**

[1] A Computer Vision System for 9-Ball Pool<https://pranjalv.com/poolvision.pdf>

[2] Blob Detection Using OpenCV ( Python, C++ )<https://www.learnopencv.com/blob-detection-using-opencv-python-c/>