**Project Proposal**

**TP1**

Project Description

1. Name: Shuffleboard
2. Descriptions

Similar to the popular table shuffleboard game, two players alternately push pucks down on a wooden table. They could try to score points, knock out opposing pucks, or protect their own pucks. Player who received highest score will win.

Each player has 4 pucks. Depending on the final position, each puck could contribute 0~3 points. The score is not calculated until all pucks are shuffled.

Competitive Analysis

1. [HTML Table Shuffleboard](https://www.htmlgames.com/game/Table+Shuffleboard)

My game would use similar rules as this web-based game. However, instead of building different levels, my game would be a real-time game. PyGame would enable better event handling and graphic rendering in my game.

1. [10 Pin Shuffle Bowling](https://apps.apple.com/au/app/10-pin-shuffle-bowling/id341894188)

This is an excellent 3D mobile game and uses slightly different rules. My shuffleboard game would instead use 2D and pseudo 3D. However, through my game shuffleboard players could play game on PC desktop as well.

Structural Plan

game\_folder 🡪

readme.txt: basic introductions

\_\_shuffleboard\_\_.py: run this file to start the game

pg\_framework.py: includes PyGame’s main loop

shuffleboard.py: subclassing pg\_framework.py, integrates MVC

pucks.py: Puck and Puck3D classes and methods

tables.py: Table and Table3D classes and methods

splash\_screen.py: subclassing pg\_framework.py, creates splash screen

score\_screen.py: subclassing pg\_framework.py, shows up after game ends

images\_foler 🡪

puck

puck3d

table

table3d

splash\_screen

(audios\_folder 🡪)

collide

shuffle

win

lose

Algorithmic Plan

1. Convert 2D coordinates into 2.5D coordinates.

Since the background is static, the 2.5D table will just use a preprocessed image of the 2D table or load another image instead.

However, each time timerFired() is called, the coordinate of every puck3D will be calculated by a method in Puck3D class. This function takes in the coordinate of Puck in 2D table in respect to its two sides. The distance from the puck to the front end of 2D table indicates how far the puck is from the camera. Thereby, the function could proportionally rescale the Puck3D.image and proportionally adjust its coordinates on Table3D.

Notice, though, that the speed of Puck3D does not need to be calculated, since this conversion is done. I only need to make sure the coordinate conversion is correct.

1. Collision simulation

The acceleration and speed of each puck is decomposed into x (horizontal) and y (vertical) directions.

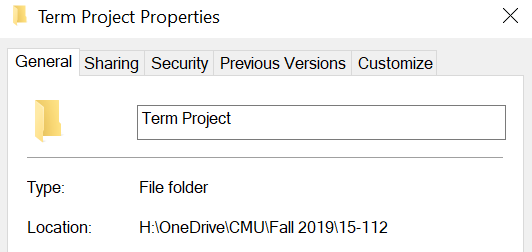
When one puck (puck1) collides with another puck (puck2), pass their coordinates, speed and puck2 object into a method in puck1. This method will use apply conservation of momentum and conservation of energy theory in 2-dimensional collision to calculate speed of each pucks after collision and change their attributes. At the next time timerFired() is called, these two pucks will move accordingly.

Timeline Plan

* Nov 17 ~ Nov 20: Collision simulation
* Nov 20 ~ Nov 22: Toggle between 2D and 2.5D graphics
* Nov 23 ~ Nov 26: 2.5D graphics algorithm
* Nov 26 ~ Nov 29: Adjust coordinate calculation; Improve image quality
* Nov 30 ~ Dec 3: Audio effects
* Dec 4: Final adjustments

Version Control Plan

I save all files about term project in my OneDrive folder. All code and images will be automatically uploaded.



Module List

* PyGame

**TP2 Update**

Changes

* I combined 2.5D and 2D graphics into the same screen
* I updated 2.5D and 2D images for pucks
* I added a new method for Table3D class, convertTo2D(), so users now interact with the game on the 2.5D graphics

**TP3 Update**

Changes

* I implemented the file saving capability, and users could save multiple game states and choose which one to load from the “Record” screen
* I created the “Settings” screen where users could drag to select puck colors and enable/disable sound
* I added the AI and the PVE (player vs environment) mode; the original two-player game is now called PVP (player vs player) mode
* I added audio effects for shuffling and collision
* I added a new button to enable users to restart the game at once
* Based on the feedback from the “User-study-a-thon” on Tuesday, I updated the image for tables so that user could easily identify the region to place the puck
* Based on the feedback, I also improved algorithm so that user could select and shuffle a puck more easily