Connor Owens

11/10/2023

Fall MAT 267 Dr. Scott Surgent

Honors Enrichment Project Review Paper

Once the project was approved. I started developing the User-Interface through Tkinter in Python3. The first user interface I created was with 3 buttons, an entry for the number of balls, and a frame for each ball in the simulation. Each frame had 2 entries, one for the ball's velocity vector and the other for the coordinate of the ball. As well, as a canvas, Tkinter drawing widget, to display the simulation. I chose the Pack widget positioning manager in Tkinter because it allows for automated positioning to deal with any window size. However, this was at the cost of precision and control of where the widgets would exist. It took approximately 3 hours of tweaking and development to get the interface acceptable. Next, I created functions for each of the 3 buttons (to create the simulation set-up, to reset the simulation, and to start the simulation). Then I created two new Python classes, vectors and simulation objects. These allowed for consistent vector calculations and storing of object values. After creating this groundwork, I created a simulation function that would draw the balls and check for collisions between the balls and bounds. I used the physics calculations from Unacademy(Source) to develop the code for the collisions between balls. To do this I created new functions to do vector calculations such as scalar multiplication/division, vector addition/subtraction, and dot product. This took more hours than I can remember but a conservative estimate would be 6 hours. This was the rough draft of the project offering a simulation of the collision of balls using vectors. I presented this rough draft to Dr. Surgent. He provided feedback to me that the project should be gamified like 8-ball and use vector angles and magnitude rather than providing the vector in component style. As well, as some kind of explanation of the project and its reasoning. With these suggestions, I went and did a complete overhaul of how the input and vector calculation was done. The user input was updated to two buttons to reset the simulation or run the simulation and two entries to control the cue ball. One entry was for the velocity vector magnitude (speed of the cue ball) and the other the angle of the cue ball from the positive horizontal direction in degrees from 0 to 360. The initial switch took ~2 hours of effort but bug identification and fixing ~4 hours. I created pocket objects and a score counter for the balls to collide with and to correlate with the number of balls in the pockets. Once everything was in the 8-ball style, a massive bug became clear. When the balls hit each other they would hit without respect to what parts of the objects are hitting. For example, if there is a vertical rectangle object. If a ball hits the left bottom corner of the rectangle with a velocity vector A, the collision system returns the resultant velocity vector B. If another ball hits the top left corner of the ball with velocity vector A, the resultant vector would be B. This is because the collision calculation takes only the velocity vectors of the objects into account as if they are single-point particles. This results in a simulation that is not "true" or realistic. Because of this bug, when the cue ball would break the balls it would collide with 1 ball as if it was head on and the ball would go straight back rather than glancing as if in real life. Due to this bug, I made a simple change in gamification where the goal of the game is to knock the balls into holes based on levels. The bug still exists but now it is less apparent and still fun to play. I created three basic levels. The next steps for this project are to fix the bug by changing the collision calculations to take into account the object's respective positioning, create more interesting levels, create a "true" 8-ball game mode, and allow for past-level play. The project can be edited and contributed to at the Github