CSCI 3010U Course Project: Simulating Plinko

Imran Mustafa

April 5, 2024

Abstract

This is a report on how to create a Plinko game simulation using techniques that where taught in class. This includes simulating balls as they are in free fall, as well as the many types of collisions that can occur between the various objects in the game.

1 Introduction

The game system has two basic components, balls and pegs. Pegs, as their name suggest, stick out of the board and exists to give the balls something to collide with. They have only two components, a position on the board, as well as a radius which all pegs share.

The second and more sophisticated of the basic components is the ball, this physics objects is what the bulk of the simulation will be around. the ball has the same components as the peg (position and radius) but also has other factors, such as mass, drag, velocity. which are needed to model the balls motion through the game board. when in free fall the ball follows this simple system of equations.

$$mx'' = -\gamma x' \tag{1}$$

$$my'' = -\gamma y' - mg \tag{2}$$

using ODE solver from scipy and this system is trivial to model and simulate. leading to very accurate results in this part of the project.

this leads into the main challenge of the lab, collision detection and response. As all previous labs and examples that have featured collisions, the point at where they occur has been constant (i.e height ¿ 0) which has lead to simple detection algorithms. this is not the case with Plinko though, as pegs can be placed anywhere on the board and balls are constantly in motions. resulting in more complex detection to this constantly changing system.

2 Collisions

For the detection system pygame's sprites systems was used to detect collisions. sprites in pygame are a class that allow for convenient bundling of common game activities, such as updating, drawing, and most importantly for the course project, collision detection. as sprites can use various collision detectors against groups of sprites, which allows for fast and simple collision detection.

In this program `pygame.sprite.spritecollide()` which checks for collisions between a single sprite and a sprite group. along with `pygame.sprite.collide_circle()` so that the collision was treated as between two circles. This returns the a list of colliding sprites which are then processed by the sprites internal collision functions. This detection function works for both types of collisions (ball-ball and ball-peg).

2.1 ball-peg Collisions

When a ball hits a peg the response is a elastic collision, where the ball is simply reflected against the peg that it collided with the velocity along the normal vector between the two surfaces while losing 5% of its kinetic energy. This is to better simulate the slight energy loss that the ball experiences with each collision.

2.2 ball-ball Collisions

when two balls collide a similar process is employed but due to the fact that the balls a moving objects each one needs to react to the collision while accounting for the speed and position of the other ball. So it makes use of the bellow equation

$$v' = v - \frac{2m_2}{m_1 + m_2} \frac{\langle v_1 - v_2, x_1 - x_2 \rangle}{\|x_1 - x_2\|} (x_1 - x_2) \tag{3}$$

which can be simplified due to the balls sharing a mass

$$v' = v - \frac{\langle v_1 - v_2, x_1 - x_2 \rangle}{\|x_1 - x_2\|} (x_1 - x_2) \tag{4}$$

This results in the speed of the ball after the collision, witch this

which results into the ball being able to have a elastic collision. then the new direction is calculated through the same method as the ball-peg collision. along with the same 5% velocity loss.

3 Conclusions

With the inbuilt pygame collision detection methods along with the scipy's ODE solvers, then combining methods learned in class the resulting simulation can model a Plinko game experience well.

A Source Code

To view the full project and run the code for yourself please visit the GitHub repository at `https://github.com/Seg-fau1t/csci3010u-project`. This repository also includes a video demonstration of the project.