

# Course Project Report

Mohammadreza Rahbar and Michael DeMelo

100781952 and 100779096

[mohammadreza.rahbar@ontariotechu.net](mailto:mohammadreza.rahbar@ontariotechu.net) and  
michael.demelo1@ontariotechu.net

Faculty of Science  
Ontario Tech University

April 11, 2023

## Abstract

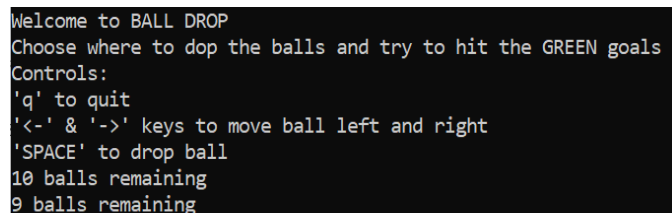
For this course (CSCI 3010) we were tasked with creating a program of some sort which is able to apply and demonstrate key course content. We achieved this by creating a program that simulates a pachinko-like game that uses falling balls that bounce off of static pegs on a two dimensional plane. This focuses on the concept of collision detection and response.

## Introduction

We created our program using the Python programming language. Our program uses libraries such as pygame, sys, random, os, numpy, scipy.integrate and time. The program is started by running the single python file 3010\_project.py. The python file uses external resources available to it in its folder, such as file images. The program is a single window that can be exited by pressing 'q', or by completing the goals within the game, where it automatically exists.

## Game Mechanics

Our game is based off of a gambling machine game Pachinko (similar to Peggle) where steel balls are dropped down a pattern of pegs and with the aim of getting them into specific areas to gain points. Using the left and right arrows you can move the ball to your desired position while dropping it using the spacebar. The ball will then drop down and bounce off of the

A screenshot of a terminal window showing the following text: Welcome to BALL DROP, Choose where to drop the balls and try to hit the GREEN goals, Controls: 'q' to quit, '<-' & '->' keys to move ball left and right, 'SPACE' to drop ball, 10 balls remaining, 9 balls remaining.

```
Welcome to BALL DROP
Choose where to drop the balls and try to hit the GREEN goals
Controls:
'q' to quit
'<-' & '->' keys to move ball left and right
'SPACE' to drop ball
10 balls remaining
9 balls remaining
```

Figure 1: Game messages printed into the console

predetermined pegs. Five random goal locations are spawned at the bottom of the pegs and when a ball hits it you get a point. The user has 10 balls in total to try to get up to five points. Messages for the game which include the amount of balls left as well as instructions, are printed into the console.

## Mathematical Concepts

We used collision detection and collision response as our main mathematical concepts to demonstrate in our project. We also used basic two dimensional physics. For our physics, we coded an updating system where there is a 2D universe (`World` class) which holds objects (`Disk2D` class) that have their own states which include their positions and velocities in the x and y planes. There is a constant negative y force applied to all objects which ensures that there is gravity within the system. Collision detection and response is computed by constantly checking the boundaries of each object. Pegs and Balls are both the same object class, but are differentiated using the `set_type()` function, so that collision response is negated for the pegs and that they're positions are kept static. The upper, left, and right boundaries of the screen are made solid, while collision response is negated for the lower boundary to allow the balls to fall down. A coefficient of restitution is used (the variable `e` set to 0.3) to ensure that there is a loss of energy from the collision responses between objects, so as to make sure the balls are realistic in their physical interactions and movement by abiding by the law of conservation of energy.



Figure 2: Steel balls colliding with the pegs under the force of gravity