Name: Dhruva Reddy (andrewID: ppreddy)

TP Mentor: Grace Cui

Recitation Section: M

15-112 Term Project Proposal:

Jetpack Mazeride

Project Description:

Jetpack Mazeride is an adaptation of the popular mobile game Jetpack Joyride developed by Halfbrick Studios. In this modified version of the side scrolling game, the core mechanics of floating, dodging obstacles and attempting to collect coins will remain but the game will replace the powerup token with a neat maze challenge where the player will attempt to escape a maze (generated by arranging coins in a random pattern) before the timer runs out. If he/she successfully does so, they will obtain a high score multiplier. If not the game returns to its original state and will end when the player hits an obstacle

Competitive Analysis:

After scouring the web, I found numerous projects that attempt to recreate Jetpack Joyride but most either utilize the unity engine or implement the pygame module to simplify game design instead of working directly with tkinter.

In the term project gallery, there was a version of this game created by Aashav Mehta in Fall 2019 titled “Barry; Jetpacks; and Friends”. In his interpretation of the game, he implements some form of pathfinding with a rocket tracking the player’s movement. He also utilizes an axis bounded square hit box for collision detection and implements a coin-based score tracking.

My version of the game differs from his by

1. Generating angled obstacles on top of the flat vertical and horizontal tazers (with more complex angular collision detection)
2. Implements powerups as per the original game concept but converts the functionality of the powerup to unlock a special game mode (maze)
3. Score tracking combines both distance and coin collection with a multiplier added by successfully completing the powerup maze

Structural Plan:

The finalized project will be segmented into numerous working files for core objects or algorithms in the game. For instance, my current working project has files marked out for coins, obstacles, and the player. An additional folder will be created to organize images, sprites etc. I then call on each of the functions or files in the main Jetpack\_Joyride.py game file.

This main file will be where the Model-View-Controller (MVC) framework will be and will consist of the core tkinter and cmu graphics functions such as keyPressed and redrawAll. It will also be where the game modes will be defined and toggled through (splash screen, main game screen, maze screen, game over screen)

Functions specific to object/terrain generation or collision detection will be defined in the subfiles for those objects

Algorithmic Plan:

The part of the project that might be the most algorithmically complex might be generating the maze and I would approach this by using the depth first search algorithm.

This would implement recursion with backtracking to generate a path from a starting point to an ending point. 4 possible movement directions will be defined and walls or paths will be defined on the grid with either a 1 or a 0. If a wall is hit or the path ahead is out of bounds the code will backtrack and cycle through the other possible directions to eventually generate a path to the end.

Timeline Plan:

|  |  |
| --- | --- |
| Date | Deliverable |
| 16th April 2022 (TP1) | * Get score * Get Coins to generate in a path * Collision detection for both coins and obstacles * Get game over screen * Modify game speed over time |
| 19th April 2022 | * Add the powerup object * Get game start splash screen * Get instructions splash screen * Get maze mode working (implement DFS algorithm) * Replace objects with images and sprites |
| 22nd April 2022 (TP2) | * Implement leaderboard with username input |
| 24th April 2022 | * Clean up UI elements for falling and dying * Potentially explore other maze generation algorithms * Explore additional powerups to reduce the speed of the game or give the main player more lives |
| 27th April 2022 (TP3) | * Finishing touches and code compilation * README file and video |

Version Control Plan:

I plan on storing my code on GitHub where I currently already have a repository. As I work towards each checkpoint, I plan on pushing 3 equally spaced versions of the code into the repository to make sure that I have a relatively recent working file to revert to should my code break at any point in the development process.

A screenshot of a computer

Description automatically generated with medium confidence

Module List:

Nil

TP2 Update:

Encountered numerous problems with the implementation of the DFS algorithm and it’s still slightly buggy. Leaderboard has been implemented and am working towards adding more obstacles before TP3

Storyboard:

Diagram

Description automatically generated

Diagram

Description automatically generated