Angry Birds - 2 Player: A Pygame Challenge

Gurnoor Sohal

CS104-Project

April 2025

Abstract

This report outlines the development process, various features, and overall design of a unique two-player version of Angry Birds 2.0 using pygame-ce. It also describes the challenges faced and how they were resolved. This report aims to provide an overview of how the game works.

Contents

1 Introduction

This is a two-player game in which the aim is to get the maximum score for a player to beat the other one. Players can either try to achieve the maximum score in a limited time or destroy their opponent's fortress first to become the winner. The game can be ended prematurely too.

2 Modules

2.1 External Libraries

- **pygame**: Used for rendering graphics, managing user input events, handling sound, collision detection, and controlling frame rate. `pygame` provides the fundamental tools required for game development.

2.2 Standard Library Modules

- **random**: Used to introduce randomness, such as selecting a random bird type during gameplay and randomly assigning the initial player turn to ensure fairness. Also used to randomly assign the various block types to each fortress.
- **math**: Used for mathematical operations to implement bird physics, such as calculating projectile trajectories, angles, and distances.
- **sys**: Used for system-level operations like cleanly exiting the game when needed (e.g., calling `sys.exit()`).

3 Directory Structure

The organisation of the project files is structured as follows:

```
CS104-PROJECT
├── data
│   └── constants.py
├── media
│   ├── fonts
│   ├── images
│   └── sounds
└── modules
    ├── game_objects
    │   ├── bird.py
    │   ├── block.py
    │   ├── catapult.py
    │   └── fortress.py
    ├── front_menu.py
    └── game_over.py
```

- **main.py** - The application's entry point, responsible for initialising the front menu and launching the game.
- **modules** - Contains all supporting Python scripts for gameplay logic, menus, and game states.
- **modules/game_objects** - Subdirectory with classes for dynamic in-game objects such as birds, blocks, catapults, and fortresses.
- **media** - Stores static assets like images, sounds, and fonts used during gameplay.
- **data** - Holds constant parameters used throughout the game logic.

4 Running Instructions

4.1 Prerequisites

Ensure that **python** is installed on your system. Ensure that pygame-ce also be installed. In case pygame is already installed uninstall it first.

```
pip uninstall pygame
pip install pygame-ce
```

The file can be run as :

```
python main.py > /dev/null
```

In case Python 3 is the default on your system, then run the file as:

```
python3 main.py > /dev/null
```

4.2 Game Navigation & Gameplay

4.2.1 Introduction screen

Upon launching, the game opens with a screen with the Angry Birds logo with background music. Click on the play button to proceed. To turn on/off the background music, use the sound icon.

4.2.2 Mode Selection

You will be prompted to select a game mode:

- **Timed Mode**: Try to cause maximum destruction to your opponent's fortress to gain a higher score in a given amount of time (60 seconds).
- **Fortress Destruction**: The player who destroys their opponent's fortress first, wins the game!

4.2.3 Level Selection

There are three different levels to the game

- **Easy**: It consists of 2 stationary fortresses with the projectiles facing no air drag.
- **Medium**: It consists of 2 stationary fortresses with projectiles experiencing air drag during motion.
- **Hard**: It consists of 2 moving fortresses with projectiles experiencing air drag during motion.

4.2.4 Choosing Fortress Dimension

Before the game starts, you can choose the dimensions of your fortress: the width and height.

4.2.5 Choosing bird generation method

The player can decide whether to generate birds as :

- **Random Draw**: Every launch bird is chosen entirely at random.
  Each turn, both players get one random bird chosen using `random.choice`
  This selection is independent and occurs every turn, ensuring a new bird is provided without relying on a predefined set or replenishment system. The birds assigned to one player have no impact on those received by the other, as the generation process is entirely separate for each player.
  It minimizes decision fatigue and keeps gameplay dynamic by introducing constant variety. Because there's no waiting for options to refresh, it ensures quick turn cycles and maintains momentum throughout the match.
- **Deck Mode**: You draft one of three available birds each turn. (like Clash Royale)
  Each player starts off with three random birds to pick from. Whenever a player uses one, a new random bird takes its place. What one player gets doesn't affect what the other gets — both players have separate bird options.
  One key advantage is that it introduces a layer of strategic decision-making — players must adapt their choices based on the options available, leading to more thoughtful gameplay. It also allows for planning ahead, as players can see their next few possibilities and react accordingly.

4.2.6 The Game

- **Aim**: Each player is allowed to use one Angry Bird per turn. The player who either destroys the entire fortress of their opponent or causes maximum damage to their opponent's fortress in 60 seconds wins the game.

- **Birds** : Birds can be dragged to release from the sling. To activate their special effects use the *SPACEBAR*. Both of the methods for bird generation ensure randomness during generation, enforcing fairness during gameplay.
- **Block Destruction** : Each block has a designated health value that decreases when hit by Angry Birds, with visual feedback reflecting the damage. Once a block's health is fully depleted, it is destroyed and removed from the scene. Any blocks above it will fall due to gravity, creating a dynamic and responsive environment.

4.2.7 Scoring System

A player's score increases when their projectile hits any block of their opponent's fortress. *Bonus points* for fully damaging the block are also given.

4.2.8 Pause, Restart and Rankings

While playing the game, it can be paused/played and reloaded. The game can be ended prematurely to check the ranking by clicking on the rankings button.

4.2.9 Game Over

The game over screen displays scores for both players and declares who the winner is for the game. An option to resume the game is also provided in case of a mistake.

4.2.10 Music Settings

Background music can be toggled on/off by clicking on the sound icon

4.2.11 Exit

You can press *ESC* to exit the game session.

5 Basic & Advanced Features

5.1 Basic Features

- **Game Interface**: Developed UI with main menu and customisation of the dimensions of fortresses, wind feature, moving block feature and entering players' names before starting the main game.
- **Two Player Gameplay** : Both players launch projectiles turn-by-turn to destroy their opponent's fortress
- **Projectile Mechanics**: Implemented gravity based launching system where the angle and force is controlled by the amount of pull by the sling.

- **Game Over Conditions**: Score is assigned based on the amount of destruction to blocks of the opponent's fortress and the number of fully damaged blocks. Player with the higher score wins the game.
- **Projectile Effects**: Each projectile has its own unique effects. Chuck's special effect is an increase in its speed; Bomb's is causing more damage to blocks; Blues' splits into three, and Red travels farther horizontally, leaving a trail of its red feathers along its trajectory.

5.2 Advanced Features

- **Game Level** : 3 different game levels exist based on the presence/absence of wind effects and moving blocks.
- **Custom Theme of the Game** : Fortress dimensions are user-controlled. The player can choose the method of bird generation and the mode of the game as well.
- **Dynamic Terrain**: Elements like wind have been introduced to hamper the speed of the projectiles
- **Destructible Structures**: The various extents of damage to the block are shown using different amount of cracks in the block. The blocks are made to experience gravity too.
- **Music & Sound**: Background music is implemented
- **Extra projectile effects**: The projectile's trajectory is described by small white dots left in the trail. Red leaves a trail of red feathers in its trajectory as a special effect.
- **Recoil velocity**: Projectiles lose some momentum and bounce from the ground when they hit it.
- **Customized Fonts**
- **Responsive Buttons**

6 Project Journey

I began this project using one slingshot and drew rectangles using the pygame module for blocks. Over some weeks, I implemented projectile physics and changed the block to images. I also created multiple levels and modes for it. Subsequently, I added recoil to projectiles as they hit the ground. Later, I added various buttons and music. In the last couple of days, I customised the font and fixed some bugs in resizing the game screen.

6.1 Challenges Faced

- **Window Resizing** : I had initially hard-coded positions of buttons and blocks. Also, changing dimensions globally across the introduction, game and game over window was an issue. Stumbling upon a discussion on StackExchange[8], I found where my issue was. I added all variables shared globally in `data/constants.py` and updated variables from there instead of their copies(like I had been doing earlier using `from data.constants import *`) whenever there was an event of resizing.
- **Implementation of Block Gravity Mechanics** : Initially, the falling of blocks when a supporting block was destroyed didn't appear smooth. To improve this, I introduced a gradual downward shift using a dy value, which resulted in a much smoother and more natural falling motion.
- **Implementation of Blue Bird's Special Effect** : The original code was designed to handle the trajectory of a single bird. Extending this to support three birds introduced several challenges — including rendering the trailing circular path for all three and correctly updating the score based on the individual damage each one caused. After resolving these issues, the special effect functioned as intended.
- **Collision Detection Accuracy** : Ensuring accurate collision detection between birds and blocks, especially when multiple birds (like the blue bird split) are involved.

References

[1] Pygame CE Official Documentation https://www.pygame.org/docs/.

[2] Pygame Official Documentation https://www.pygame.org/docs/.

[3] Angry Birds Game (by GFG) https://www.geeksforgeeks.org/angry-bird-game-using-pygame/.

[4] Angry Birds Game (by estevaofon) https://github.com/estevaofon/angry-birds-python.

[5] Angry Birds Game (by marblexu) https://github.com/marblexu/PythonAngryBirds.

[6] Random Module https://docs.python.org/3/library/random.html. https://www.w3schools.com/python/ref_random_choices.asp.

[7] Math Module https://docs.python.org/3/library/math.html. https://www.w3schools.com/python/module_math.asp

[8] Window Resizing https://www.pygame.org/wiki/WindowResizing. https://stackoverflow.com/questions/20002242/how-to-scale-images-to-screen-size-in-pygame

[9] Inserting music https://www.geeksforgeeks.org/python-playing-audio-file-in-pygame/.