Angry Birds 1 vs 1 Two Player Game with Pygame CE

CS108 Project Report

Nitheesh Kumar Vennela

April 30, 2025

Contents

1 Introduction

Angry Birds 1 vs 1 Two Player Game with Pygame CE is a competitive multiplayer game where two players take turns launching birds at each other's fortresses. Inspired by the classic Angry Birds game, this version introduces a player-vs-player mode using the Pygame CE framework.

The objective is to destroy the opponents structure using different types of birds, each with unique abilities. Players must aim, launch, and strategically use special powers to maximize damage. The game combines physics, timing, and skill, offering a fun and challenging two-player experience.

2 Project Tree

```
project/
+-- core/
|   +-- BirdManager.py
|   +-- Bird.py
|   +-- blocks.py
|   +-- catapult.py
|   +-- fortress_manager.py
|   +-- Player.py
|   +-- sound_manager.py
|   +-- sprites.py
+-- screens/
|   +-- gameplay.py
|   +-- init_screen.py
|   +-- menu_screen.py
|   +-- name_screen.py
|   +-- settings.py
|   +-- winner_screen.py
+-- resources/
|   +-- images/
|   +-- sounds/
|   +-- fonts/
+-- main.py
+-- constants.py
+-- latex.pdf
```

3 Core Directory

This folder contains the main logic and classes for game mechanics, including:

- `BirdManager.py`
- `Bird.py`
- `blocks.py`
- `catapult.py`
- `fortress_manager.py`
- `Player.py`
- `sound_manager.py`
- `sprites.py`

4 Screens Directory

This folder contains Python files that manage different game screens:

- `gameplay.py`
- `init_screen.py`
- `menu_screen.py`
- `name_screen.py`
- `settings.py`
- `winner_screen.py`

5 Resources Directory

This folder contains static assets used in the game:

- `images/` - Game images and sprites.
- `sounds/` - Sound effects and background music.
- `fonts/` - Fonts used in UI and other text displays.

6 Main Files

These are top-level project files:

- `main.py` - Entry point of the application.
- `constants.py` - Global constants.
- `latex.pdf` - This document.

7 Used Python Modules

The project uses the following Python modules:

- `pygame` - For graphics, events, and audio.
- `math` - Mathematical operations and functions.
- `sys` - System-specific parameters and functions.
- `time` - Timing-related functions.
- `random` - Random number generation.

8 Running Instructions

To run the game, ensure that Python 3 and the required libraries are installed.

Steps

1. Install Python 3: Visit `https://www.python.org/downloads/` to download and install it.
2. Install dependencies (if not already installed):

   ```
   pip install pygame
   ```

3. Run the project using:

   ```
   python3 main.py
   ```

Note: You must have the required Python modules pre-installed before running the project.

9 Game Screens Overview

This section describes the different screens in the game and their functionalities.

1. Init Screen

The initial screen shown when the game starts. It typically displays a logo or loading animation and transitions automatically to the menu screen.

Figure 1: Example of the Init Screen

2. Menu Screen

This is the main hub from which the player can navigate to other parts of the game. It contains three buttons:

- **Play** – Starts the game.

- **Settings** – Opens the settings screen.
- **Exit** – Quits the game.

Figure 2: Example of the Menu Screen

3. Settings Screen

This screen allows players to customize aspects of the game:

- **Mode Toggle** – Switch between Dark Mode and Day Mode.
- **Sound Toggles** – Six sound categories (e.g., background music, effects) that can be toggled on or off individually.
- **Back Button** – Returns to the Menu Screen.

4. Name Screen

Before gameplay begins, players must enter their names.

- Contains two input fields: `Player 1 Name` and `Player 2 Name`.
- Validation logic:
  1. Neither name can be empty.
  2. Names must be different.
- After successful input, proceeds to the Gameplay screen.

5. Gameplay Screen

This screen contains the main game logic, visuals, and interactions. It is shown after valid player names are entered and the game begins.

Figure 3: Example of the Settings Screen

Figure 4: Example of the Name Screen

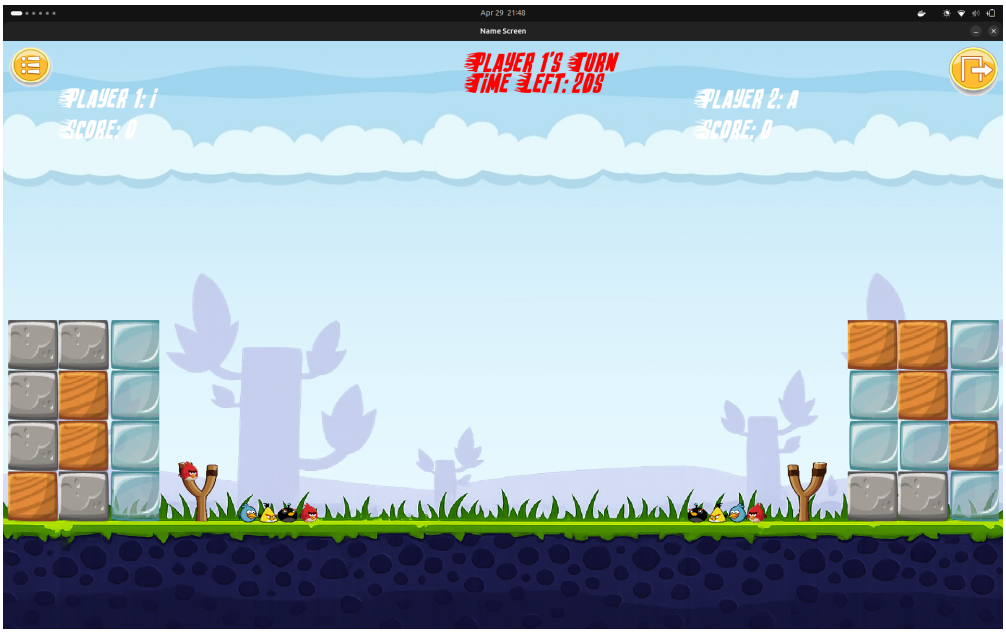

Figure 5: Example of the Gameplay Screen

10 How to Play

Game Objective

The goal of the game is to strategically launch birds from catapults to destroy the opposing player's structures using turn-based mechanics.

Controls

- Use the mouse to click and drag the bird at the catapult.
- Release the drag to launch the bird in a chosen direction.

Gameplay Overview

- The game begins with **two catapults**, one for each player.
- **Four birds** are preloaded at the start—each assigned to a catapult.
- Only the bird **nearest to its respective catapult** can be clicked and load to catapult.
- Blocks or structures are **randomly generated**.
- The game operates on a **turn-by-turn basis**, switching between players.
- Each turn allows the player to launch one bird.

Game Rules

1. Players take alternating turns to shoot birds from their catapults.
2. Only the bird closest to the catapult can be selected each turn.
3. Drag-and-release mechanics simulate slingshot-style launches.
4. The game continues until a win condition is met.

Winning Conditions

The winner is determined based on the destruction of blocks and the players' scores, ensuring fair turns for both. The rules are:

1. If a player destroys all of the opponent's blocks:
    - The game waits until the opponent has had an equal number of turns.
    - If the second player **fails** to destroy all blocks in their turn, the **first player wins**.
    - If the second player **also destroys all blocks**, the winner is decided by comparing scores.
2. If **time runs out** before all blocks are destroyed:
    - The winner is the player with the **higher score**.
    - If scores are **equal**, the match is declared a **tie**.

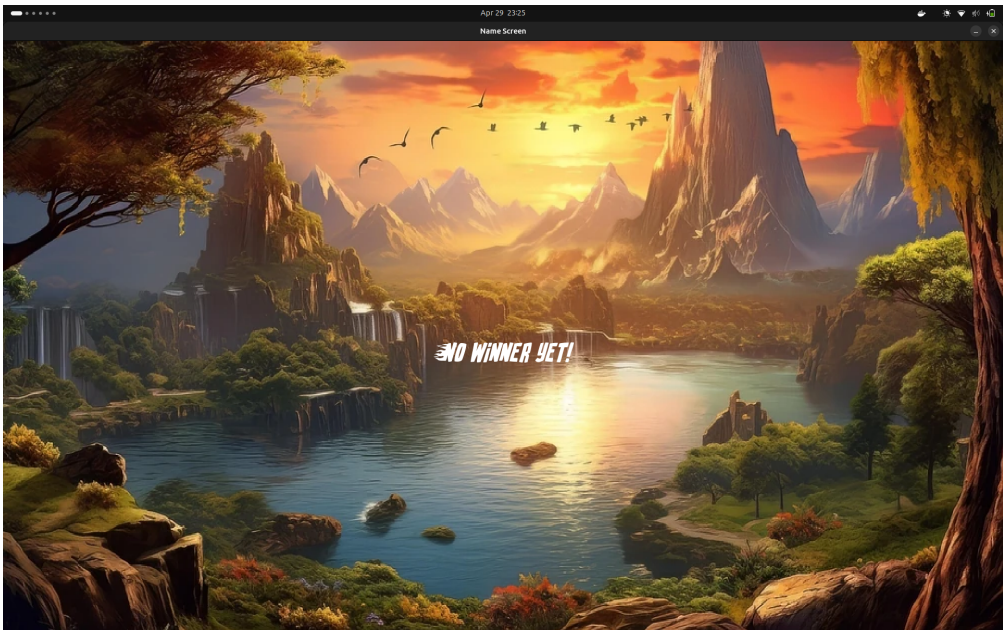

Figure 6: Example of the Winning Screen

11 Game Requirements and Customizations

This section lists the basic functionality needed for the game to run, as well as advanced features that enhance the experience. Example code snippets are provided where applicable.

Basic Features

The following are essential components for a working version of the game:

- Basic drag-and-release mechanics for launching birds.
- Two catapults for turn-based gameplay.
- Random loading of birds and block structures.
- Alternating turns between two players.

- Score tracking and input validation.
- Screen transitions and interactive UI.
- Loading of sounds and images.

Advanced Features

These features provide enhanced gameplay and customization:

- Gravity applied to blocks for realistic physics.
- Bird superpowers such as speed boost, splitting, or explosion.
- Background music integration.
- Toggle between dark mode and light mode.
- different images for blocks based on their life.

12 Project Challenges and Learning Experience

This project marked my first experience in game development, and it has been a valuable and exciting learning journey. I learned how to build a complete game from scratch using `pygame-ce`, which allowed me to create interactive graphics and implement core game features.

At the beginning, many concepts were new to me, such as:

- How to detect collisions between game objects.
- How to organize code using object-oriented programming.
- How the game loop works, and how to manage events like mouse clicks.
- How to load and use sounds, images, and fonts.

Sometimes, I found it difficult to figure out how to implement certain features or structure my code correctly. To solve these problems, I:

- Took time to think through the logic step-by-step.
- Referred to official `pygame` documentation and tutorials.
- Asked questions online and searched for code examples.
- Used ChatGPT to understand new concepts and fix errors in my code.

By working through these challenges, I gained a much better understanding of Python, how games are built, and how to make my code more organized. I also improved my problem-solving skills and became more confident in building interactive programs.

Overall, this project helped me grow as a developer. I learned how to think critically about game design and how to implement features that enhance the player experience. I am excited to continue exploring game development and applying what I've learned in future projects.

13 Super Powers

13.1 Red

Effect: No observable superpower.
Subjects exposed to red energy display no measurable enhancements or abilities. The energy is classified as inert in terms of superhuman activation.

13.2 Yellow

Effect: Speed enhancement.
Exposure to yellow-based stimuli significantly increases the subject's speed. Reaction time and movement velocity are enhanced up to 300% of baseline measurements.

13.3 Blue

Effect: Triplication.
Subjects interacting with blue energy are capable of dividing into three identical versions of themselves. Each version retains equal cognitive ability and physical strength of the original.

13.4 Black

Effect: Increased gravity.
The presence of black energy causes an intense local gravitational field. Subjects experience a gravitational pull estimated at 3–5 times Earth's gravity, resulting in limited mobility.

References

[1] Khinsider Archive. Angry birds friends original game soundtrack, 2022. Accessed: 2025-04-30.

[2] Marcus Renna. Maze - modularized basis, 2024. GitHub repository. Accessed: 2025-04-30.

[3] Pygame Community. Pygame ce documentation, 2024. Accessed: 2025-04-30.

[4] Angry Birds Wiki. Wordmark designs backgrounds, 2023. Accessed: 2025-04-30.

[3] [4] [1] [2]