

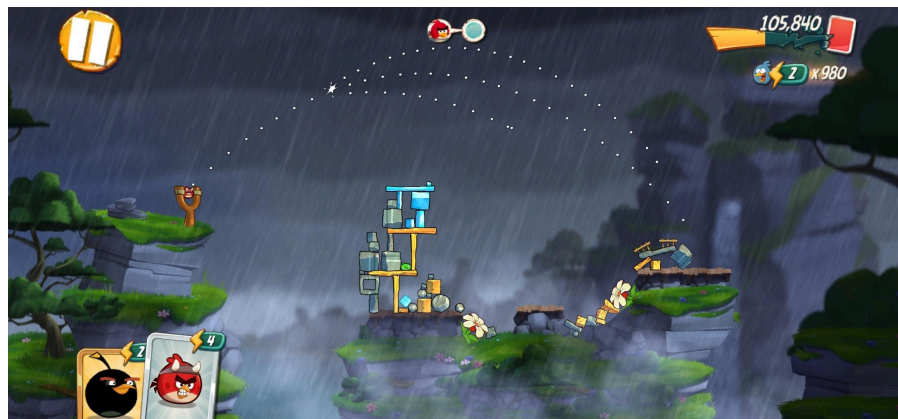
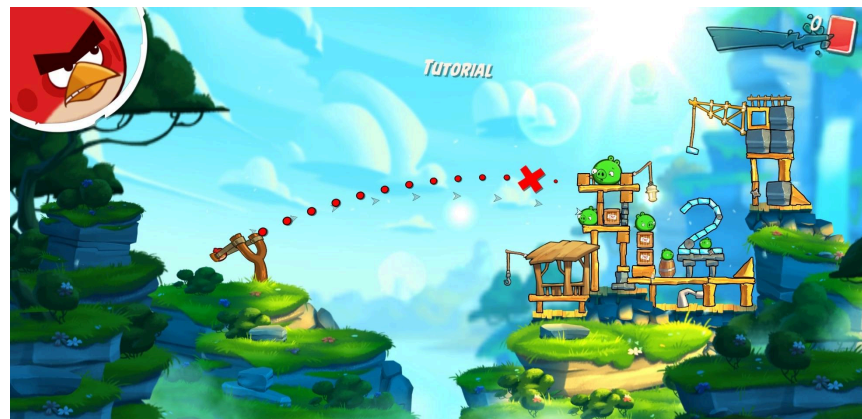
Angry Birds - 2 Player: A Pygame Challenge

Author: Evuri Mohana Sreedhara Reddy, UG TA, CS 104/108:
2024-25

General Description

You might have played **Angry Birds**, where the objective is to launch a projectile from the catapult - the bird to knock down structures and eliminate targets. The challenge is to launch the projectile at the right angle with the proper force to achieve maximum destruction.

ANGRY BIRDS

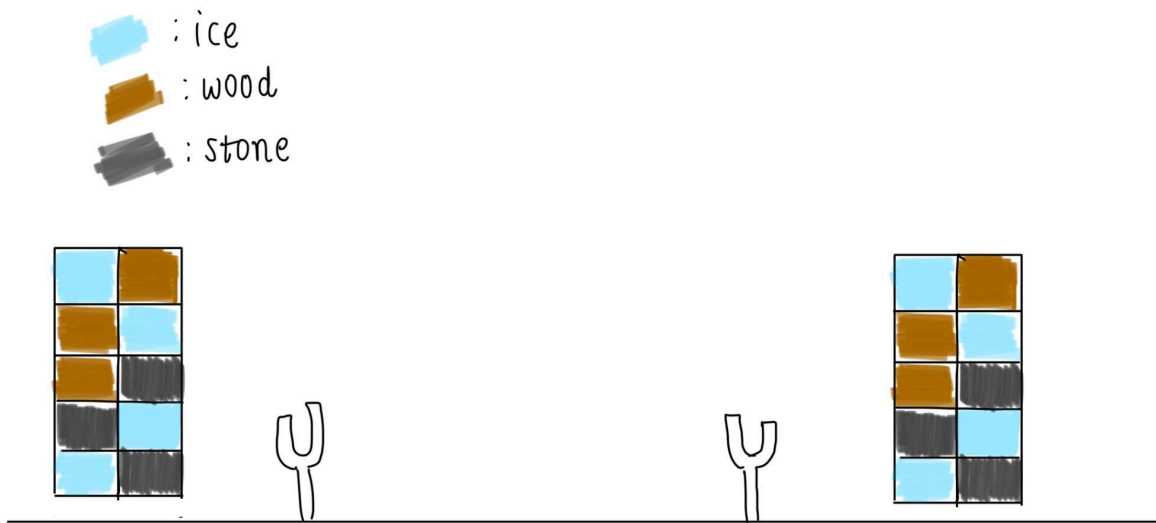


Instead of attacking some of the game's targets (the default single player), this game will be a **turn-by-turn-based attack** on your opponent's fortress.

The aim of the project is to develop a **2-player** game using **Pygame (pygame-ce)** while incorporating **realistic physics (in the projectile)** and **interactive mechanics**.

The Game Overview - Aspects of the game

Instead of only the left catapult and the structures on the right, the game has **two basic structures (pre-existing): rectangular blocks** on the left and right behind the **players' catapult to attack the opponent's structures**. The game screen's skeleton should look like this:



The Catapult

The catapult (shooter) is similar to the original game. The key difference is that we have **two players** and **one catapult for each**. The objective of the player is to destroy the other player's fortress. Note that this catapult should be unaffected when an opponent's projectile passes through it.

The Fortress (the “structures” of the player)

The **fortress' structure** consists of **simple cubes or blocks** placed in a grid (or any form you would like). The blocks can be of three types:

- ❖ Wood,
- ❖ Ice, and
- ❖ Stone.

Each **block has health**. The **block gets destroyed after its health drops to zero**. The reason for the different types of blocks is that users exploit the advantages of the other angry birds (the projectiles).

The Projectiles - Angry Birds

The projectiles are of four types:

- ❖ **Red:** These do balanced damage to all kinds of blocks.
- ❖ **Chuck:** Does more damage to wood blocks and less to others.
- ❖ **Blues:** Does more damage to ice blocks and less to others.
- ❖ **Bomb:** Does more damage to stone blocks and less to others.

The more damage can be represented by a more significant drop in the health of the blocks.

The Game

Aim of the Player

This is a turn-based 1v1 game in which the player's goal is to destroy all the blocks of his opponents. Each player is allowed to use **one Angry Bird per turn**. The player who first destroys the opponent's entire fortress wins.

Birds Generation

There can be multiple ways that the game can generate birds. A few examples are:

- ❖ **Replenishing batch-wise:** Initially, both players have one of the different projectile types. If the birds finish, another set of projectiles is replenished (for the base case, there are four birds; for each player, birds are replenished after four turns).
- ❖ **Random generation:** Initially, each player can choose from 3 birds (randomly generated). If a player uses a card, a new bird is given to the player randomly. The choice of birds by one player does not affect the other player's birds, as they are independent of each other. (This is quite similar to the card generation in Clash Royale if you have played)

It's up to you how the bird generation works. Mention the method you used in the report.

Blocks destruction

The health of the blocks reduces as the angry birds hit them. After a block is completely destroyed, you can make it disappear without changing any other blocks (they can remain floating in the air).

Basic Objectives

The objectives include incorporating **ALL** of the above features into:

1. **Game Interface:** Develop a UI with a main menu and enter the players' names before starting the main game. The start of the game should take you to the gameplay screen.
2. **2-Player Gameplay:** Players launch projectiles competing to destroy structures. Turn by turn-based.

3. **Projectile Mechanics:** Implement a gravity-based launching system where players use mouse input (drag and pull) to control the angle and force.
4. **Game Over Conditions:** Define winning and losing conditions based on destroying the opponent's structure/fort to an extent/entirely.
5. **Projectile Effects:** Add projectiles with unique effects based on the type of the projectile.

Additional Customizations

Complete all the basic objectives before implementing the additional customizations.

Some examples are mentioned below:

1. **Game Level:** You can have various levels in the game. You can implement three versions: Easy, Medium, and Difficult. Some ideas for the difficulty level:
 - a. More blocks on a more complex level.
 - b. Wind flow is added in the opposite direction of the shooting player.
2. **Custom theme for the game:** Instead of using the standard angry birds, you can try a new theme (new set of projectiles, blocks, and background) to make your game unique.
3. **Dynamic Terrain:** Introduce elements like wind or moving obstacles for the different difficulty levels.
4. **Scoring System:** Assign points based on accuracy and the precision of the destruction caused to the target.
5. **High Score Leaderboard:** Track performance across sessions.
6. **Destructible Structures:**
 - a. There should be various stages of destruction visible for the structures/blocks (You can do something like this: If the integrity of a structure is between x% to y%, show the corresponding image, and try to have at least five different variants of the structures).
 - b. In the basic objectives, we have mentioned that if a block gets destroyed, the other blocks remain without changing their positions. Here, you can implement the blocks falling under gravity, too.

Feel free to add more customizations, but remember not to remove any basic implementations.

Report and Viva

The project will be evaluated on core mechanics, creativity, gameplay, and innovation. A **LaTeX project report** should be provided, and a **viva** will be conducted.

Important Instructions

- ❖ The project needs a report written in **LaTeX**. The report should include:
 1. **Modules:** List of external libraries used and their purpose.
 2. **Directory Structure:** Breakdown of all files and their functions.



3. **Running Instructions:** How to execute and play the game, including controls.
 4. **Basic & Advanced Features:** Implemented mechanics with explanations and images.
 5. **Project Journey:** Key learnings, challenges, and how they were addressed.
 6. **Bibliography:** References and links to external sources.
- ❖ **Allowed Modules:**
 - The list of the allowed modules are:
 - `pygame-ce`
 - `numpy`
 - `random`
 - `math`
 - `sys`
 - `os`
 - **If you want to use any new modules, get permission from the author.**
 - ❖ The marks distribution for the tasks (**15 marks**) will also be updated here.
 - **NULL**
 - **Note: Customization will be extra credit** and can **compensate for poor performance** in other exams. But this is **capped at three marks** (20% of 15).
 - ❖ **PLAGIARISM = SEVERE PENALTIES & DADAC.** Don't risk it. We are **very** serious about this. **Cite all references** in the report to avoid plagiarism issues.
 - ❖ **Any corrections to the problem statement based on feedback will also be updated and highlighted in this document.**
 - ❖ **Modularize your code.** Keep different functionalities in separate scripts for better readability. Also, use functions/classes wherever required. This has weightage (refer to the marks distribution).
 - Sample project with a **modularized codebase**:
<https://github.com/rennaMAhcuS/Maze>.
 - **File naming conventions:**
 - `main.py` → The primary game script (entry point).
 - `report.pdf` → The LaTeX project report.
 - Additional files (e.g., images, sounds, custom modules) should be described in the report.
 - **Not following file naming conventions or submission guidelines will result in a penalty.**
 - ❖ **AI tools are allowed but not recommended.** You **must** be able to explain your code in the viva.

All The Best!

For any questions:

- **WhatsApp (preferred):** +91 91779 50989
- **Mail:** Evuri Mohana Sreedhara Reddy

References:

1. [Pygame CE official website](#). You can refer to the **documentation** from here.
2. **The original game:**  Angry Birds 2 - Official Gameplay Trailer
 - a. Some example single-player implementations (Note that they have used the **pymunk** physics library. This library is **not allowed** for the projectile motion. You have to simulate it on your own):
 - i. <https://github.com/estevaofon/angry-birds-python>
 - ii. <https://github.com/marblexu/PythonAngryBirds>
3. To know the difference between **pygame** and **pygame-ce**, refer to this video:
 Pygame CE - Better & Faster