Catventure

1st LIEN,CHE-KUAN
NYCU EE
Hsinchu, Taiwan
lian.ee11@nycu.edu.tw

2nd LIEN,WEI-CHIH
NYCU EE
Hsinchu, Taiwan
wayne.ee11@nycu.edu.tw

***Abstract*—Catventure is a small game created using Pygame. Players take on the role of a little black cat navigating through various traps and collecting cat food cans. In this document, we will introduce the gameplay and program architecture of this game, and finally gather player gameplay data for analysis.**

I. Introduction

This game reference from ADVENTURE ISLANDS release of Poor bunny, we have modified many parts, such as the new shooting function, increase the type of traps, etc.

II. Gameplay

A. Game Objectives

In the game, players take on the role of a small black cat with the objective of collecting as many cat food cans as possible. The quantity of cat food cans collected serves as the game score. Throughout the game, there will be numerous traps that hinder your scoring progress, and the number of traps will increase over time. However, there will also be power-up items that can enhance your abilities and make you stronger.

B. Controls

Press the ”A” key to move the cat left, the ”D” key to move the cat right, the ”S” key to make the cat drop down from a platform, and the space bar to make the cat jump. Additionally, you can use the left mouse button to shoot projectiles.

C. Traps

Arrows will randomly fall from the sky, saws will randomly generate on the ground and move horizontally, and darts will randomly generate and bounce around the screen. Ghosts will appear when the cat collects cat food cans, with yellow ghosts bouncing around the screen and purple ghosts actively chasing the cat. All of these traps can be eliminated using projectiles.

D. Items

The shield power-up grants the cat invincibility for three seconds, protecting it from any harm caused by traps. The health potion allows the cat to recover 1 HP, but it cannot exceed the maximum health limit. The flame power-up transforms the cat’s projectiles into a spread of shotgun-like pellets, allowing the cat to fire 10 shots at once within a cone-shaped area.

Fig. 1. Game Screenshot

III. Program Architecture

A. Libraries

The entire game is primarily designed using Pygame due to its convenient features. The Sprite attribute’s Group functionality in Pygame facilitates collision detection, allowing for efficient handling of game elements. Additionally, Pygame provides excellent support for handling graphics, text, and sound effects.

Other libraries utilized in the game include random and math. The random library is used for managing random objects, while the math library enables mathematical calculations required for various aspects of the game.

B. Class

- The GameObject class serves as the base class for all other classes. It inherits from the pygame.sprite.Sprite class and defines common attributes such as x and y coordinates, image loading, and collision position updates. It represents various objects in the game. The pygame.sprite.Sprite class is a built-in class in Pygame that allows multiple objects to be grouped together for convenient collision detection.
- The Player class inherits from the GameObject class and represents the player object in the game. It includes methods for player movement, shooting, and collision handling.
- The Bullet class inherits from the GameObject class and represents the bullet object in the game. It defines methods for bullet movement and boundary detection.

- The Enemy1 class inherits from the GameObject class and represents the enemy 1 object in the game. It defines methods for the enemy to track the player.
- The Enemy2 class inherits from the GameObject class and represents the enemy 2 object in the game. It defines methods for enemy movement and boundary detection.
- The Arrow class inherits from the GameObject class and represents the arrow object in the game. It defines methods for the arrow to descend and boundary detection.
- The Saw class inherits from the GameObject class and represents the saw object in the game. It defines methods for horizontal movement, boundary detection, and animation updates.
- The Dart class inherits from the GameObject class and represents the dart object in the game. It defines methods for the dart to track the player.
- The Platform class inherits from the GameObject class and represents the platform object in the game. It defines methods for horizontal movement and boundary detection.

C. Art Design

- Character Design: The protagonist of the game is a cat, as cats are widely adored by many people. Players can enjoy the fun of controlling a nimble cat in the game. Other objects such as ghosts, arrows, and darts are represented using simple and clear icons, ensuring that players can easily understand their roles and interactions within the game.
- Animations: To create animations, you can organize the image files into a dictionary, where each key represents a specific character state, such as jumping, walking, or idle. The characters and objects have dynamic animations that enhance the gameplay experience and provide visual feedback to the player's actions.
- Sound Effects: Sound effects play a crucial role in enhancing the gaming experience and creating an immersive atmosphere in Catventure. We have incorporated various sound effects into the game, such as injury, shooting, collecting cat food, etc.
- User Interface (UI): The game's UI is designed to be intuitive. Presenting scores, HP, etc using simple buttons and text.

IV. Player Test

A. Packaging

Use Pyinstaller to package the whole game into an exe file, it should be noted that the picture files and sound files need to be placed into the same folder, otherwise the game will run with errors.

B. Testing

We added a function inside the program to automatically write the score and game time into the file at the end of the game, so that the player only needs to return the recorded data to the developer after playing to complete the whole testing process.

C. Data Analysis

The following is the statistical chart of play time and score, there are about 100 data, the average play time is 40 seconds, the highest play time is 105 seconds, the average score is 14.6, the highest score is 43 points. We hope to control the time within two minutes, otherwise players will feel tired if they play too long.

D. Player Feedback

Some players respond that the game difficulty is too high, perhaps we can set up a menu to adjust the difficulty in the future, so that each player can experience the most suitable level of difficulty

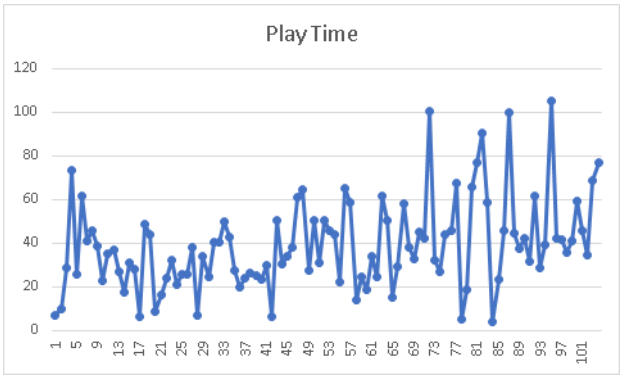

Fig. 2. Play Time Statistics Chart

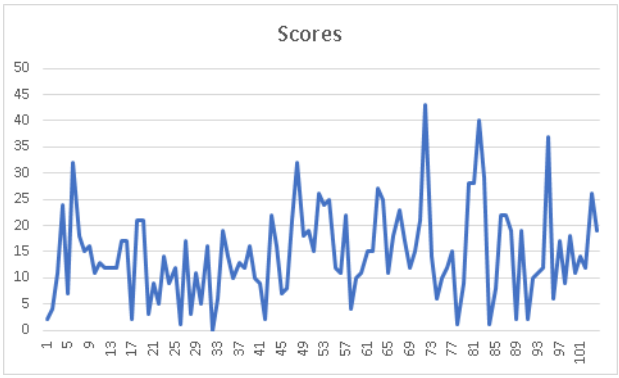

Fig. 3. Score Statistics Chart

V. Conclusion

We encountered various challenges and difficulties in the process of developing games, usually not mentioned in the course, such as animation production and UI interface design, but also gained valuable experience and knowledge as a result, by the way, ChatGPT provided the most help. At the same time, I also learned about various aspects of game development, including visual design, sound and game balance. Finally, I think the game can add two-player mode, two cats competing for cat food cans, or add different stages, each stage can change different types of terrain or traps.

REFERENCES

[1] OpenAI. "ChatGPT: Large Language Models." OpenAI. Accessed June 1, 2023. https://www.openai.com/research/chatgpt

[2] Pygame Documentation". [Online]. Available: https://www.pygame.org/docs/. Accessed on: June 1, 2023.

[3] "Poor Bunny". [Online]. Available: https://poki.com/en/g/poor-bunny. Accessed on: June 1, 2023.