MAZE GAME

PUNIT YADAV

April 2024

Contents

1 Objective

This project is intended to create an auto-generating Maze game using pygame

2 Introduction to Maze Game

This program generates a maze game that changes its path every time we play another match. Players need to complete the game in limited no of lives and time.There score is also calculated to compare there progresses.

3 Modules

To make this program work these are some of the Modules utilized

- pygame: This is a set of Python modules designed for writing video games. It includes computer graphics and sound libraries.
- sys: This module provides access to some variables used or maintained by the Python interpreter and to functions that interact strongly with the interpreter. In this code, it is used for the sys.exit() function to quit the game.It used to exit the program when user close the game
- random: This module in Python is a standard library module that provides functions for generating random numbers. The random module is commonly used in various applications, including simulations, games, cryptography, and statistical analysis, to introduce randomness and variability into programs.

- time: Time module in Python provides various time-related functions. We can use it to track the elapsed time during gameplay and apply time limits to each level.In my program time module is used to Tracking time in the gameplay function: Inside the gameplay function, the time.time() function is called to retrieve the current time when the gameplay loop starts. This time value is stored as $\text{start}_{t}ime to track the starting point of the level. And to calculate elapsed time and remaining time.$

4 Directory Structure

.1 maze_game/. .2 game.py. .2 haki.mp3 .2 sound3.mp3 .2 brook.mp3 .2 path.txt

5 Running instructions

To run the game, follow these steps:

1. Ensure you have Python installed on your system.

2. Download the game files and save them in a directory.
3. Arrange the directory as shown in the Directory structure section
4. Run the game by executing the Python script with the command `game.py`.
5. Follow the on-screen instructions to navigate the main menu and start playing the game.
6. Use the arrow keys to move the player character through the maze.
7. Reach the endpoint to complete the level and advance to the next level.

6 Task Completed

6.1 Basic completion

Implemented Features:

- Basic maze generation using the randomized algorithm often referred to as a "randomized depth-first search" or "recursive backtracker" algorithm.
- Player movement within the maze using arrow keys.
- Collision detection to prevent the player from moving through walls.
- Score tracking and display.
- Game over screen with options to start a new game or return to the main menu.
- Added a camera that focus only on the player and see limited part of the maze
- Density of maze differ according to difficulty level selected

6.2 Advanced completion

- Added sound effect like background sound that playes throughout the game and some sounds on selecting options on main menu or clearing a level
- I also have added the leader board that notes the score of top five players

7 Project journey

Throughout the project, I gained valuable experience in game development using the Pygame library. I encountered challenges such as implementing maze generation algorithms and managing game state transitions. However, through research and experimentation, I overcame these challenges and successfully completed the game. Overall, the project taught me about problem-solving, algorithm implementation, and game design principles.

textbfHere are some of the images shared by me of my maze game

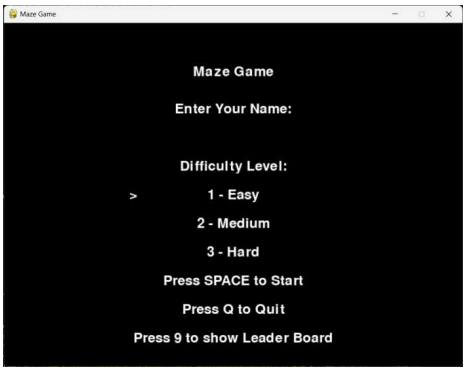

Figure 1: Example Image

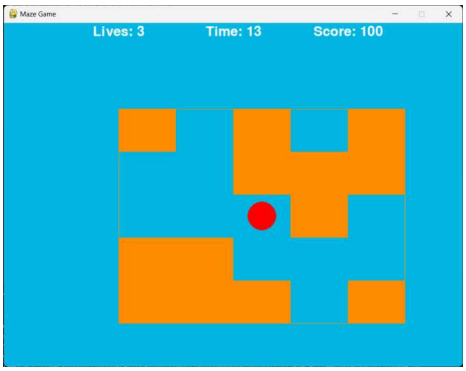

Figure 2: Example Image

Figure 3: Example Image

8 Bibliography

- "Pygame Documentation" - `pygame.org/docs`
- Kemeshwari Chebrolu. CS108 Course Slides
- Learned several concepts from AI