Rouglike Game

Andrei Aiurov
ČVUT–FIT
aiuroand@fit.cvut.cz

January 7, 2024

1 Introduction

This report was created as an attachment to semester work on CTU FIT in Prague for BI-PYT.21 in 2024. The task was to create a rouglike game using python. Rouglike is usually a game for 1 player, whose purpose is to find an exit from current location. During this process, player should solve different puzzles and fight monsters that are haunting him.
This game can be implemented in many different ways. I decided to create classic 2D game for PC.

2 Game description

My game consists of several parts: main menu, level selecting menu, rules menu and game itself. Their descriptions are summarized below:

1. Main menu.

It contains 3 buttons

- Start Game - selecting takes you to the map selecting menu.
- Rules - selecting takes you to the game rules.
- Quit - selecting closes the window.

Figure 1: Main menu screenshot

2. Rules menu.

It contains visualzation of game rules

Figure 2: Rules menu screenshot

3. Level selection menu.

It contains names of all available maps. Level can be selected by pressing 1-9 button on keyboard.

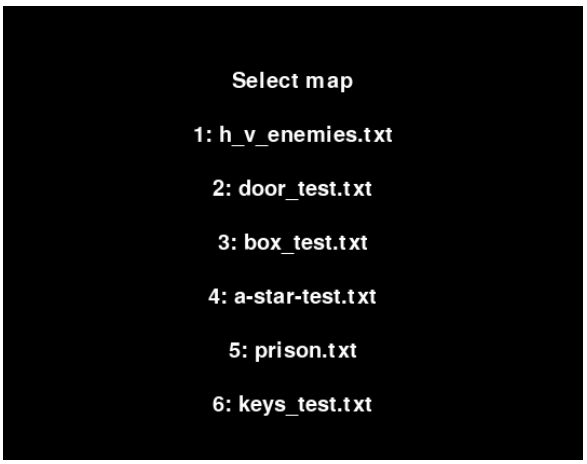

Figure 3: Level selecting menu screenshot

4. Game.

All game rules were actually described in second item. I want to describe all object my game has:

- Player - green circle that can be controlled using 'W' 'A' 'S' 'D' buttons.
- Enemy - red circle, that has 3 beharvior patterns (All 3 will be described later).

Figure 4: Player

Figure 5: Enemy

- Wall - wall of the white color is the main object that creates maze. Can not be destructed.
- Exit - door of brown color is exit door. If player reaches this object he wins.

Figure 6: Wall

Figure 7: Exit door

- Key - key have one of 3 colors and can be collected by player.
- Door - object that can not be traversed without a key of required color.

Figure 8: Keys of all 3 colors

Figure 9: Doors of all 3 colors

3 Enemy behavior

There are 3 behavior types for enemies in my game: 1 and 2 are trivial, 3 is smart.

- Horizontal walking enemy - repeatedly goes up and down, almost not dangerous and easy to avoid.
- Vertical walking enemy - repeatedly goes left to right, almost not dangerous and easy to avoid.
- Following enemy - constantly follows player using the shortest path if there is a way between him and player. Stays otherwise. He is 2 times slower than player, but the most dangerous object on the whole map.

2 first enemies use simple approach: if there is a free cell above (on the left) - make one step above (to the left), if there is not - change the direction. Last enemy uses smarter approach, every move he calculates the best route to player using A* pathfinding algorithm modified for 2D maze [1] and makes 1 step in this path's direction.

4 Technical part and testing

The game and it's logic was created using Python 3.12.10. All visualisation was created using pygame library of 2.5.2 version.
Game structure follows main idea of game loop (**Input** (get player and enemies moves), **Update** (check if player found a key or monster caught him), **Render** (plot map and current entities positions)).
The game can also be tested using pytest library. The tests control PEP8 format. They also allow to test, if my program can correctly determine damaged files on input (More to map files format in my README.md).

5 Conclusion

As a result, I got a functioning game, that represents Rouglike gameplay basics. Already at this stage the game works as I planned, however, in the future it has a huge potential for adding new features, new maps, new monsters, new items and so on.

References

[1] Muhammad Ahsan Naeem. A* pathfinding in 2d maze. online, 2021. https://levelup.gitconnected.com/a-star-a-search-for-solving-a-maze-using-python-with-visualization-b0cae1c3ba92.