COMP 521 - Assignment 1

Yi Tian Xu
260520039

Feburary 4, 2015

1 Doorway/projectile behavior and interaction

1.1 Strategy for setting up the doorways such that the maze is solvable

The three sets of rooms are randomly chosen on a 30×30 grid. One secondary room is picked to be connected with the exit.

Then the braided maze is generated using Aldous-Broder/Wilson algorithm with the rest of the cells. When choosing a random branch to the maze, the algorithm is allowed to choose a cell that it has already connected to the maze with a small probability except if the cell is part of a room. This creates a braided maze.

To make a doorway for each primary room, the algorithm randomly remove a wall from each primary room that is not part of a secondary room. Then a wall between each primary room and its associated secondary room is remove to make a door to the secondary room.

This allows the player to go to any primary room from the start of the maze.

The core functions for this part are in the file Maze.cs.

1.2 Strategy for setting up the sequence of doorways the player need to take

The order of the color is predetermined in the code. For the sake of notation, let c_0, c_1, c_2 and c_4 be the colors. The last projectile and the last doorway to the exit of the maze are always colored with c_3.

Let S_i be a secondary room with door color c_i, $i \in \{0, 1, 2\}$. Let P_j be a projectile with color c_j, $j \in \{0, 1, 2, 3\}$.

Consider a randomly generated maze with a set of colored rooms and doors, a set of colored projectiles, an entrance and and the exit accessible from a secondary room S_i. S_i must contain projectile P_3. Then our algorithm chooses:

$S_{[(i+1) \mod 3]}$ to contain projectile P_i;

$S_{[(i+2) \mod 3]}$ to contain projectile $P_{[(i+1) \mod 3]}$

Then I can set the entrance to contain projectile $P_{[(i+2) \mod 3]}$

This guarantees that the player must go through all the secondary room before exiting the maze.

This function is called `distributeBullets()` in file MazeManager.cs.

1.3 Strategy for door/projectile and anything-else/projectile interaction

Each door from the primary room to the secondary room is set to a color that is the same as its corresponding projectile. A door only open when it's hit by a projectile of the same color. This is implemented as matching the tags of the colliding objects.

In the game, the projectiles are design as large spheres. They are allowed to roll on the ground or bounce off the wall when they are projected. Once it stops moving (e.g.: when it's forward movement is blocked by walls), the projectile is no longer effective to open a door. To detect when its stops moving, the algorithm checks the difference in its position between two frame is less than a certain threshold.

To detect that a projectile did not hit its corresponding target, I made a counter that keeps tracks if a projectile has become useless and if a door is openned. That is, when a projectile is fired, if a door hit by the projectile opens, it'll increment the counter by a unit. When projectile loses its usefulness, it'll decremenet the counter by a unit and check if the the final value is negative. If no door got opened during the process, then the counter's value is negative and this will determine whether the player has lose or not.

The algorithm for this are in files MazeManager.cs, Bullet.cs and Door.cs.

2 Note for other behaviors

2.1 Player projectile interaction

Player can pick up a projectile when that projectile is at most some small distance from the player. When "." is pressed, the algorithm goes through an array storing all the projectiles and check the distance. Once a projectile is used, it's removed from the array and thus can no longer be picked up.

The algorithm for this are in files MazeManager.cs and Player.cs.

2.2 Bounded starting position and terrain

The player is confined in a rectangular space containing the entrance of the maze when the game starts. This ensures that the player doesn't move away from the maze entrance or enter the maze from the maze exit.

The player is not allowed to move off the terrain. This is done by checking if the absolute value of player's x or z coordinate is greater than a larger number (700).

3 Narrative

Figure 1 shows the petri-net. The P_i's represent the projectiles and the D_i's represent the doors.

The game is 0-pointless. When the player fires the projectiles while not aiming at the correct door, the player looses immediately. In the game, if the player does this, a text message will appear saying that the player will stay in the maze forever. (On the other hand, if the player wins, then a text message saying that the player is free appears.)

As there is a path from any state reachable from the start state in the petri-net to the winning state, $p = 0$.

Figure 1: Petri-Net