The HIP-PATTIE HOP-PATTIE Frog Game

Kazuko Hass
CSSIE 450
Program description
12/17/01
# Table of Contents

I. Project Description ............................................................................................................. 3  
II. Structures/Classes............................................................................................................. 4  
III. State Transition Diagram ............................................................................................... 5  
III. State Transition Diagram ............................................................................................... 5  
IV. Program limitations ........................................................................................................ 7  
V. Features For Next Version ............................................................................................... 8  
Some web sites that discuss/explain Maze generate Algorithm ........................................ 9
I. Project Description

My game requires the player to travel to a goal from a starting position and calculates the amount of time that it takes to reach the goal. The player can control the motion of the main character, which is a frog, by changing the direction and speed of frog’s motion (Increase/Decrease). The starting position is the frog’s house and the goal is his girlfriend’s house.

In the game, there are two types of obstacles. One of them is a brick wall, these are located in the various locations around the game board, much like a maze. The frog cannot walk on and/or pass through the brick, so the player must maneuver the frog around each wall in order to avoid them. Another obstacle that the player will come up against is a monster. There is the number of the monsters hiding throughout the game board. During his travels the player may encounter the monster, if the frog collides with the monster he will become bigger; thus, it is hard to maneuver the obstacles. Therefore, avoiding the monsters allows the player to reach a goal faster. Also if the frog becomes too big, he cannot move in the maze (stuck in the maze), in this case, this game ends and the player loses.

There are two windows: the world and viewpoint windows. The World Window allows the player an overhead view of the entire game, and the Viewpoint Window allows the player to follow the frog’s motion. However, the player can zoom in or out of the WW, so the player can see more or less of the world whenever the player likes.
II. Structures/Classes

I have modified my mp4 for this final project. I created Tmonster, BrickMaze, Tgirlfrog, and Text classes, and added TtextureRectangle class to this program. Tmonster and Tgirlfrog are derived from TgraphicsObject class, and Text class is from TdrawableObject class. BrickMaze has list of Trectangles (25 * 25 array) and store whether the position has wall or not and also store the position of each monsters. (so they don’t collide each other).
### III. State Transition Diagram

<table>
<thead>
<tr>
<th>Event</th>
<th>Action</th>
</tr>
</thead>
</table>
| **When program start**        | Set fGameMode = eIntro  
Set up everything that is needed to  
Display the Intro screen         |
| **Right Mouse button down**   | If(fGameMode == eNothingActive)  
FGameMode = eDragNormalView; |
| **Right Mouse button drag**   | If(fGameMode == eDragNormalView & &  
The current mouse location is in the window)  
Update the game window location |
| **Right Mouse button up**     | If(fGameMode == eDragNormalView)  
fGameMode = eNothingActive; |
| **Left Mouse button down**    | If(fGameMode == eNothingActive)  
fGameMode = eZoomNormalView; |
| **Left Mouse button drag**    | If(fGameMode == eZoomNormalView & &  
The current mouse location is in the window)  
Update the game window location and size |
| **Left Mouse button up**      | If(fGameMode == eDragNormalView)  
fGameMode = eNothingActive |
| **‘e’ key or End button is pressed** | fGameMode = eIntro;  
Reset speed bar value to be the initial value  
Clear Timer  
Delete all objects from the object list.  
Set up everything that is needed to display the intro screen |
| **Play buttons is pressed**   | fGameMode = eNothingActive  
initialize all values  
-- create maze, frog, girl frog, and monster objects and insert in the list  
Start timer |
| **Direction buttons/arrow key is pressed** | Change the frog’s direction of velocity |
| **‘<’/’>’/Speed slider value is changed** | Increase/decrease the magnitude of the frog’s velocity |
| **Rotate/size/position slider value is changed** | Check which parts is selected  
Then rotate/size/position the parts |
| **The player reach to the goal** | fGameMode = eEndingGame  
Stop the timer  
Delete all objects in the list  
Display winning message  
fGameMode = eIntro  
Reset speed bar value to be the initial value  
Clear Timer  
Set up everything that is needed to display the intro |
| **The player loses the game** | fGameMode = eEndGame  
|                             | Clear Timer  
|                             | Delete all objects in the list  
|                             | Display lost message  
|                             | fGameMode = eIntro  
|                             | Reset speed bar value to be the initial value  
|                             | Set up everything that is needed to display the intro |
| **Exit button/Esc key is pressed** | Terminate the program |
IV. Program limitations

The greatest limitation of this game is the location of monsters. Since the locations are random and the program does not check whether the location is valid or not in relation to the goal. Some monsters may be located where they may not be in or block the path to the goal.

Secondly, the monster’s direction to move is determined by random numbers. Depending upon the series of random numbers, the monster may move the opposite direction from the frog, instead of moving towards the frog to prevent him from reaching the goal. Even if the frog moves towards the monster.

Another limitation of this game is the shape of the maze. It is easy for the user to see the shortest path to the goal. As a result, it may not take long for the user to find the best way to the goal.

In addition, the motions of the frog and monsters are not natural. When the main character collides with obstacles such as a brick wall, they do not naturally bounce away; surprisingly they bounce in an unexpected direction that may confuse the user. Moreover, when the frog and monster collides with each other, it does not look like a real collision. This is because I stopped their motion before they actually collide (it is easier to implement the after effects this way). They do touch each other as they collide with brick walls.

These limitations may allow the user to travel to the goal without interaction with monsters. As a result, when this is the case it creates or makes the game become boring.
V. Features For Next Version

The future versions of this game should use better algorithm to draw the maze, since the shape of the maze is very important in this kind of game. Also the world size should be much larger, so it makes it more difficult for the user to see the path to the goal.

These depend on the difficulty level of the game. This version’s medium/hard maze can be used for the easiest level in the future versions. However, the future versions of this game should not allow the user to choose the level of the game. Instead, as the user passes the lower level, they go to the next level. Then, the next version of this game should allow the user to save/load their game, so the user can start from the level.

Also, in the future versions, the frog and monsters’ motions and the monster’s locations should be calculated with better formulas. So their motions look more smooth and natural, than this version. The reactions of the characters after their collision should be improved. Moreover, in the future version of this game would determine the monsters’ initial position and the direction to travel depending on the path to the goal and the frog’s current location. So the user cannot reach to the goal without interacting with the monsters.

In the future version, I would improve my graphics (a frog and monsters), so the user can be attracted to the characters.

Finally I would clean up my codes and design such as deleting unnecessary codes, modifying to the better algorithm to calculate the collision, and making more classes. So that future versions of this game can be more efficient, faster, and maintainable.
Some web sites that discuss/explain Maze generate Algorithm

http://www.campus.ne.jp/~ishigami/CREATION/GENERAL/MAZE.htm
(Japanese)

(Japanese)

http://www.astrolog.org/labyrinth.htm
(English)

http://www.easystreet.com/~jkirwan/misc004.html
(English)

http://encyclozine.com/Puzzles/Mazes/About/
(English)

http://www.mazeworks.com/mazegen/mazetut/howto.htm
(English)

http://www.chaoticarts.de/team/mgricken/gptricks/gp_lab.html
(English)

(English)