Constructive Procedural Content Generation: Evaluation of the Complexity of Procedurally Generated Maze Algorithm

Name: LIM JIA QI (17134267/1)

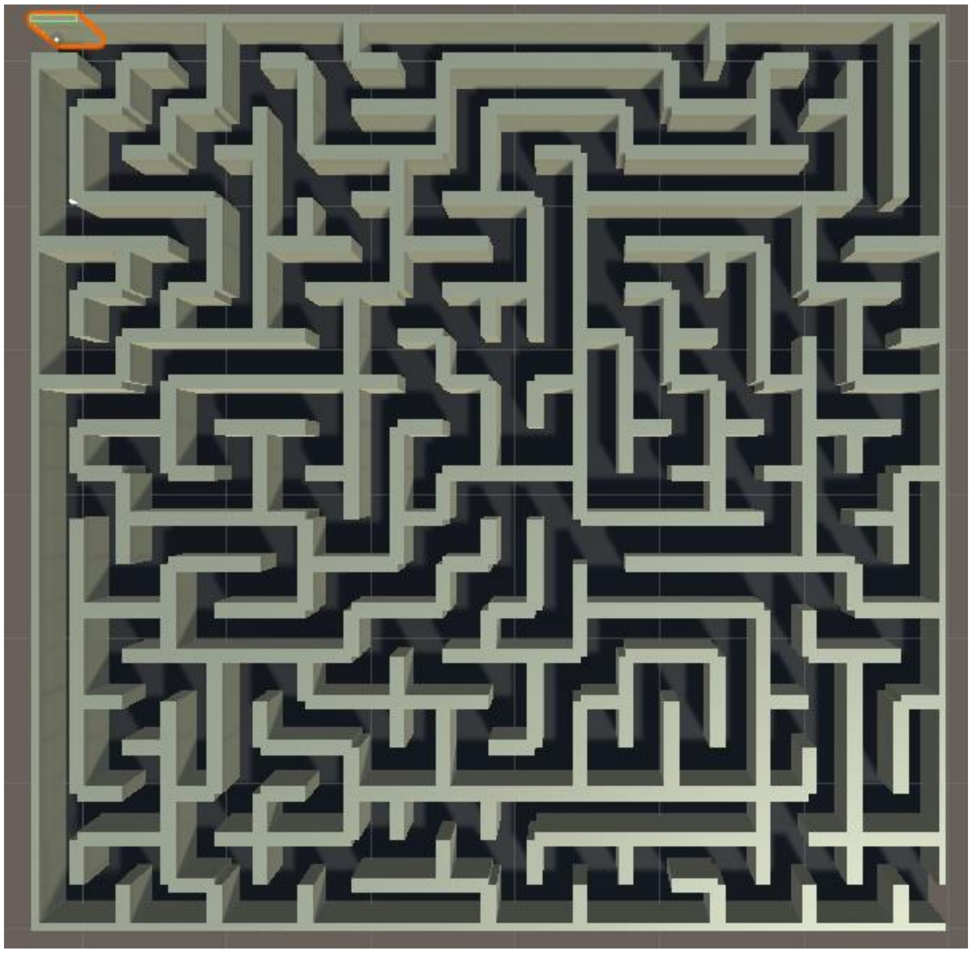
Lecturer: PROF. LOO CHU KIONG

LIM JIA QI (17134267/1)

Hunt and Kill algorithm

This algorithm is chosen as the PCG method used to generate maze in Unity. Hunt-and-Kill algorithm is the algorithm that can generate “perfect maze” where the mazes generated only have one and only solution. This algorithm works very similar to recursive backtracking algorithm since both algorithms are very likely to generate long winding passages, but it works differently when it handles dead ends (hunt mode). As this algorithm does not required any extra storage or stack so it is suitable to very large mazes. Recursion is not required in this algorithm, so potential stack overflow that will happen in recursive backtracker can be avoided. Time complexity of this algorithm is O(|V| + |E|), this algorithm works very fast but will slow down a little bit when it enters hunt mode.

Example of maze generated using Hunt and Kill algorithm from top view, the size of the maze is 20*20 and it is represented using tiles, the orange highlighted wall is the entrance of the maze.



Pseudocode of this algorithm (from https://weblog.jamisbuck.org/2011/1/24/maze-generation-hunt-and-kill-algorithm):

```
1. Choose a starting location.
2. Perform a random walk, carving passages to unvisited neighbors, until the current cell has no unvisited neighbors.
```

3. Enter “hunt” mode, where you scan the grid looking for an unvisited cell that is adjacent to a visited cell. If found, carve a passage between the two and let the formerly unvisited cell be the new starting location.
4. Repeat steps 2 and 3 until the hunt mode scans the entire grid and finds no unvisited cells.

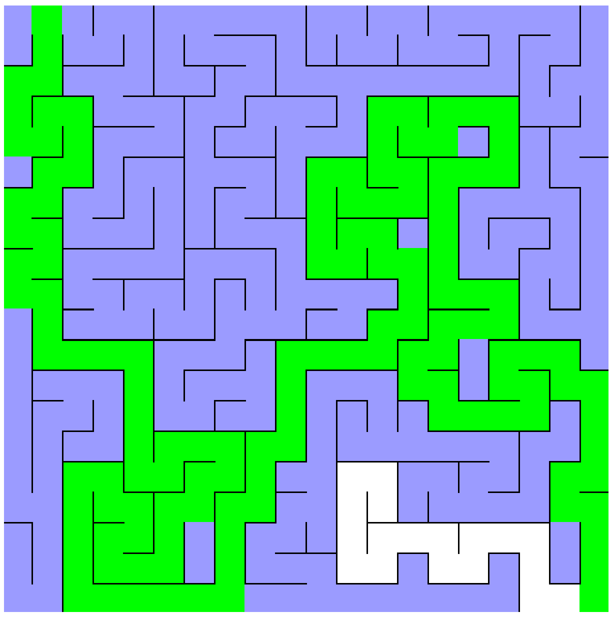
Evaluation of algorithm

To evaluate this algorithm, two AI agents are used – Breadth-first search and Depth-first search. These two AI agents will playtest the game. The tests are carried out 3 times with different mazes generated by hunt-and-kill algorithm for both agents, time taken used for the agents to reach the exit of the maze, number of tiles of the path that lead to the exit will be recorded as metrics to evaluate the difficulties of the mazes generated by calculating speed of the agents to complete the maze.

AI agents	Trials	Time of completion (s)	Numbers of tiles from entrance to exit	Speed for maze completion (tiles / time taken)
Depth-first search (DFS)	1	11.70	114	9.74
	2	14.22	102	7.17
	3	31.03	104	3.35
Breadth-first search (BFS)	1	26.13	132	5.05
	2	16.66	106	6.36
	3	21.50	146	6.79

The faster the speed for the agents to complete the maze, the less complexity and difficulty of the maze.

Examples of evaluation of algorithm by converting the maze generated into 2D using BFS, tiles with green color form the solution path, tiles in purple are the attempts by the BFS agent to find the exit:



LIM JIA QI (17134267/1)

Examples of evaluation of algorithm by converting the maze generated into 2D using BFS, tiles with green color form the solution path, tiles in red are the attempts (backtracking when meet dead-end) by the DFS agent to find the exit: