CSAI 301
Artificial Intelligence

Project Phase II

Prepared by

Ahmed Mostafa 202201114

Omar Yasser 202201589

Seif Eldin 202200973

Yusuf Tamer 202201929

Zewail City

28/12/2024

Table Of Contents

Table Of Contents **2**
Model Problem **3**
The Q-Learning agent **3**
Per Episode Calculations: 3
Monitoring **4**
Agent's behaviour, movement patterns, and Policy **4**
Comparisons **4**
20 x 20 Maze 5
30 x 30 Maze 6
40 x 40 Maze 7
50 x 50 Maze 8

Model Problem

Problem definition: Finding the shortest path from the starting point to the goal position using the Q-Learning agent

State space/Problem size: m x n, 50 x 50 = 2500
Actions: move North, South, East, and West
Initial state: agent at (1,1)
Goal state: reach (m × n)
Transition function: move from one cell to the maximum rewarding cell using one of the possible moves through an open path
Reward Function: if it is the goal state it rewards 100 otherwise -1
Assumptions:

The Q-Learning agent

Parameters: **Maze** which we will try to reach the goal of, **starting point/** initial state, **epsilon:** which will determine if we use a random action or the optimal action, **directions**/Actions, **open**: which defines whether there is a wall or not in the direction we are looking at, **Number of episodes**, **max_steps**, **alpha**, **gamma**.

Initialization: We created the Q-Table by looping through the rows and columns and adding them to the Q-Table, with the key being the row and column and the values are another dictionary where each of the keys is one of the 4 states with their corresponding values equaling 0

Per Episode Calculations:

First, we increment the step by 1, then we get a random value to identify whether we will choose a random action or the max profiting action based on whether the random number is less than or equal to the epsilon or not. After choosing the direction, we set the next_state to that direction if and only if that direction is open, which defines if it is a wall or not. After deciding on the next_state, we update the reward and the q_table's q_value of the current state's best action using the following formula:
$Q(s,a) \leftarrow Q(s,a) + \text{alpha} \times [\text{reward} + (\text{gamma} \times \max(Q(s', a'))) - Q(s, a)]$

Finally, we move on to the next state. We repeat these processes until we reach the goal state or we reach our maximum steps.

Monitoring

We observed that the agent struggled to find a path for a 50x50 maze given an episode limit of 5000, when the limit was increased to 20,000, the agent successfully solved up to a 60x60 maze. This indicates a direct correlation between the number of episodes and the solving ability with a time tradeoff for training and parameter optimization.

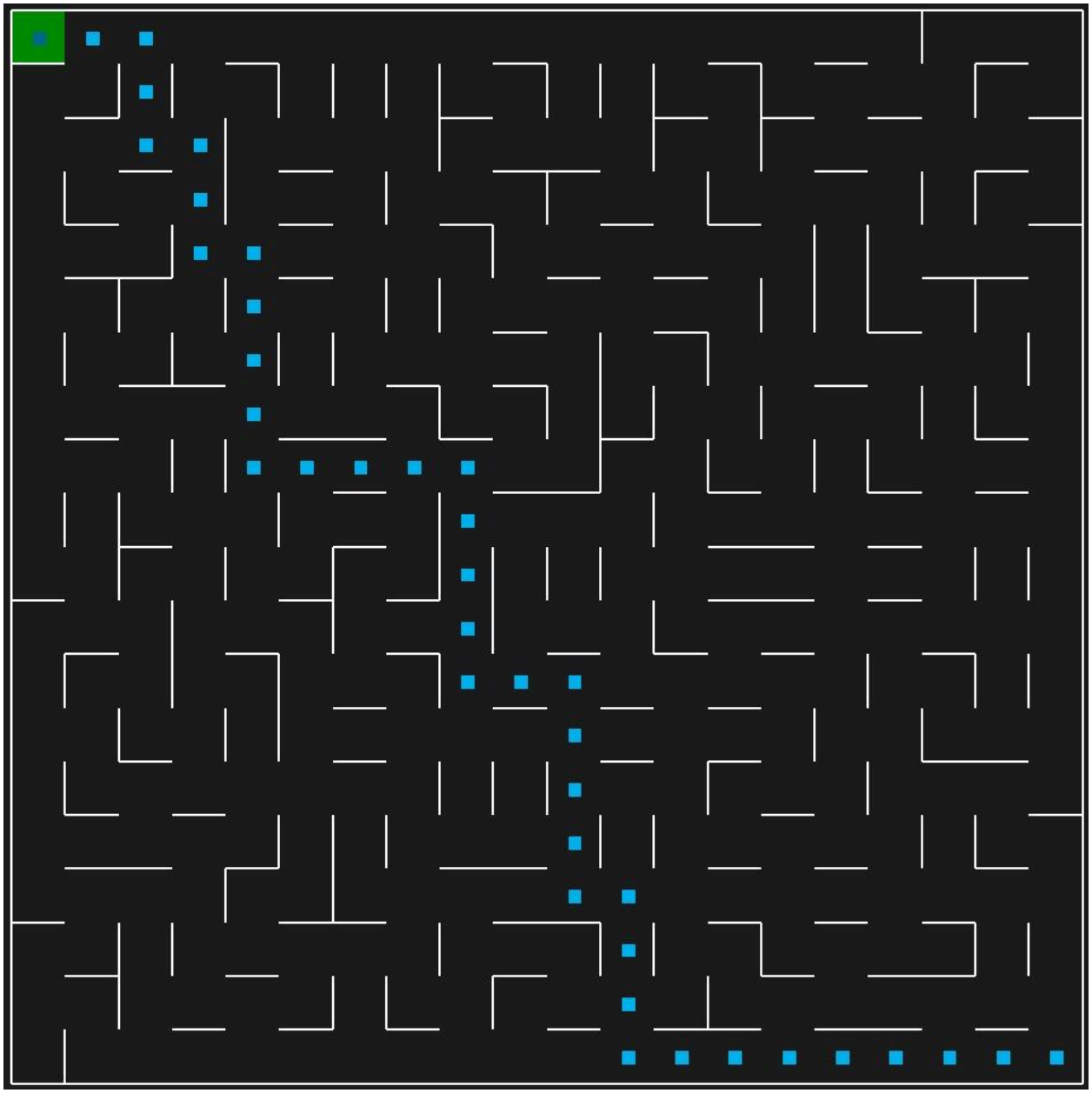
Agent's behaviour, movement patterns, and Policy

The agent initially explores the maze with mostly random actions due to the Q-values being zero. However, as training progresses, it updates its values based on the rewards received. The learning rate of 0.1 allows for smaller progressive updates, while the high discount factor of 0.9 encourages the agent to prioritize the goal state rewards. Using the positive and negative rewards in getting closer or further away from the goal helps the agent optimize its pathing to reach the shortest path. More information can be found in the Per Episode Calculation section.

Comparisons

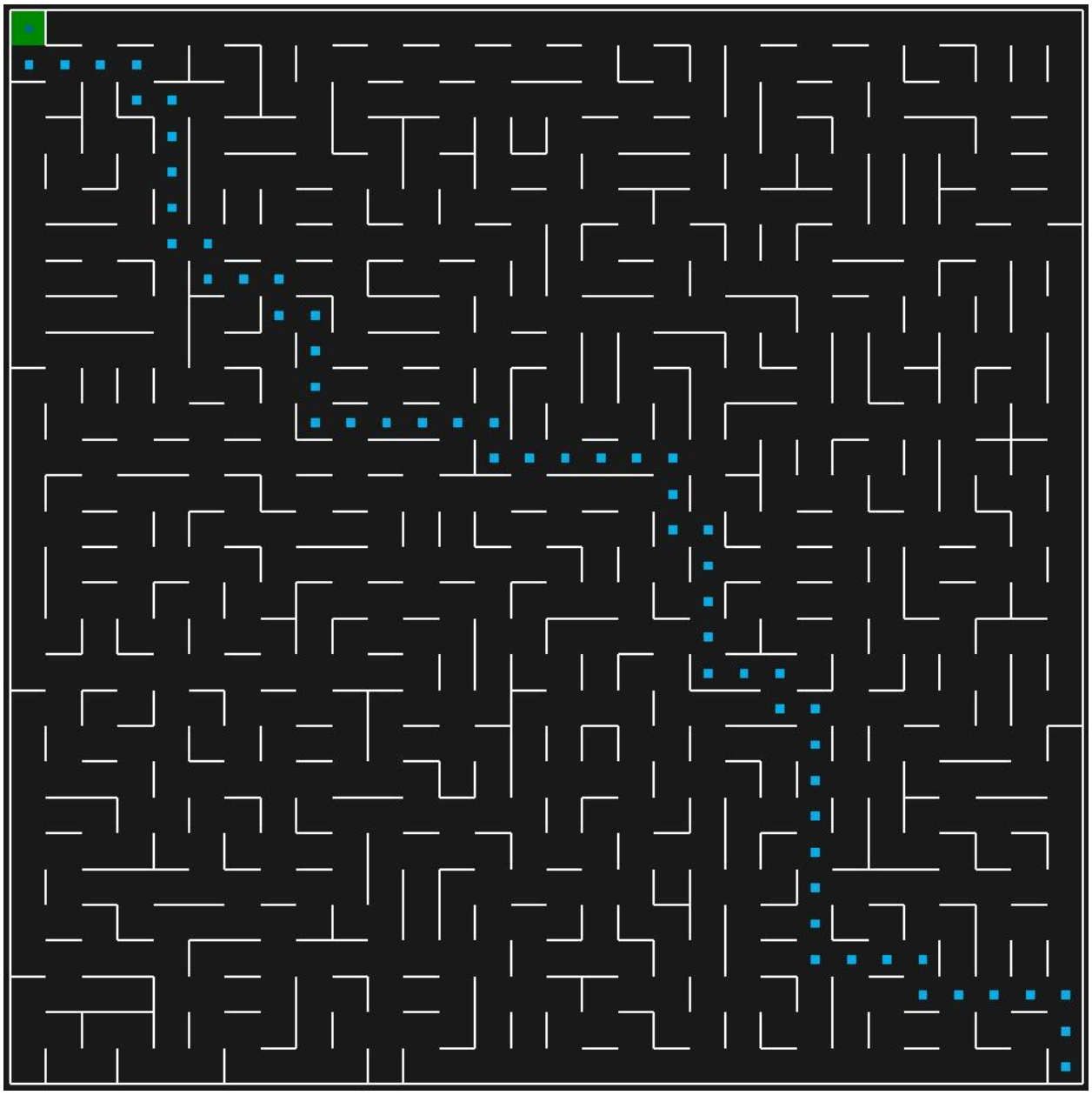
Size\\\\\\\\\\\\\\\\\\Metric	Midway	End	Time at End (sec)
20x20	100%	100%	17.94
30x30	99.89%	99.94%	30.34
40x40	99.59%	99.97%	49.67
50x50	99.02%	99.5%	83.89

20 x 20 Maze



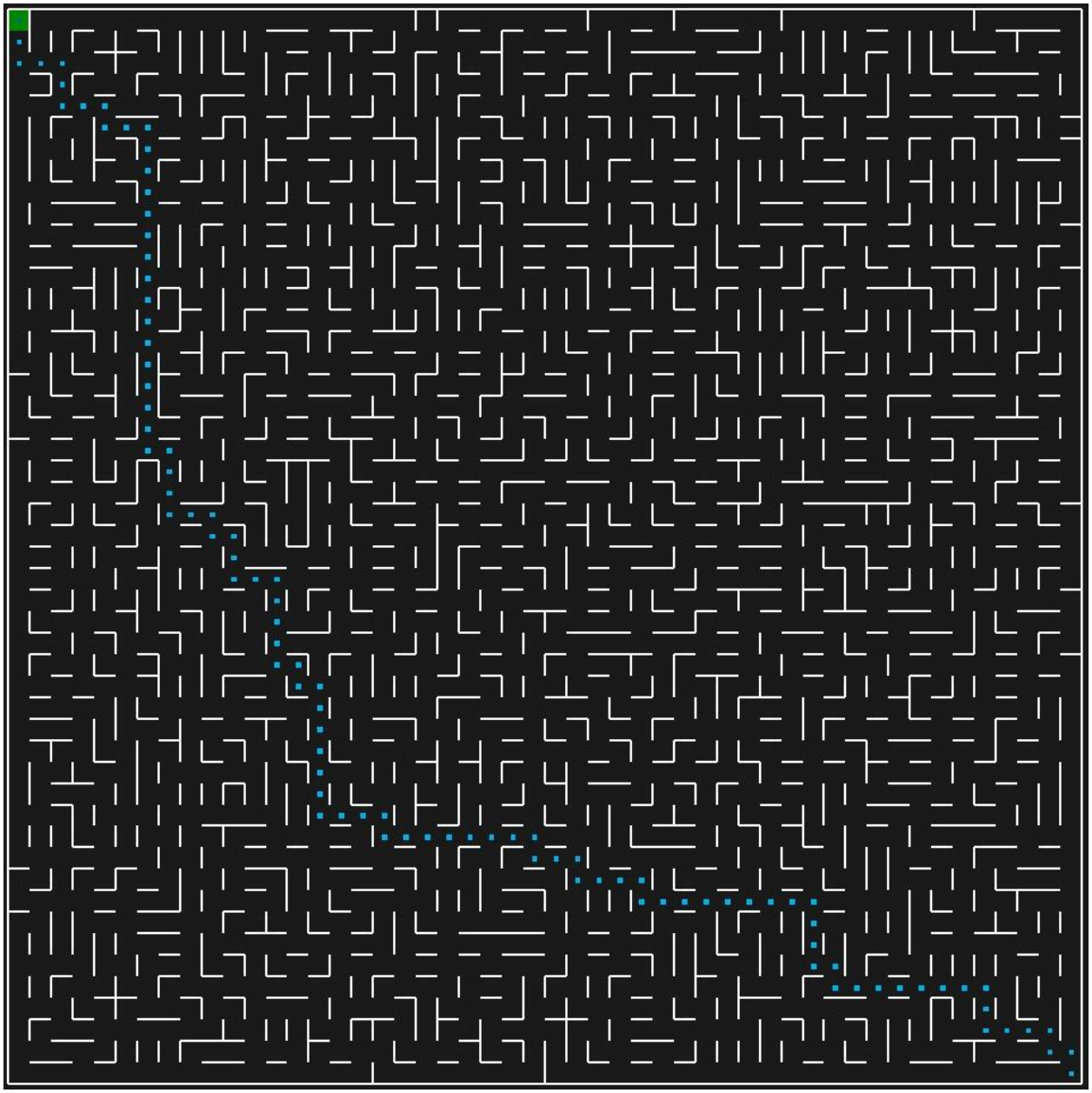
Episode Count: 5000
Step count: 100

30 x 30 Maze



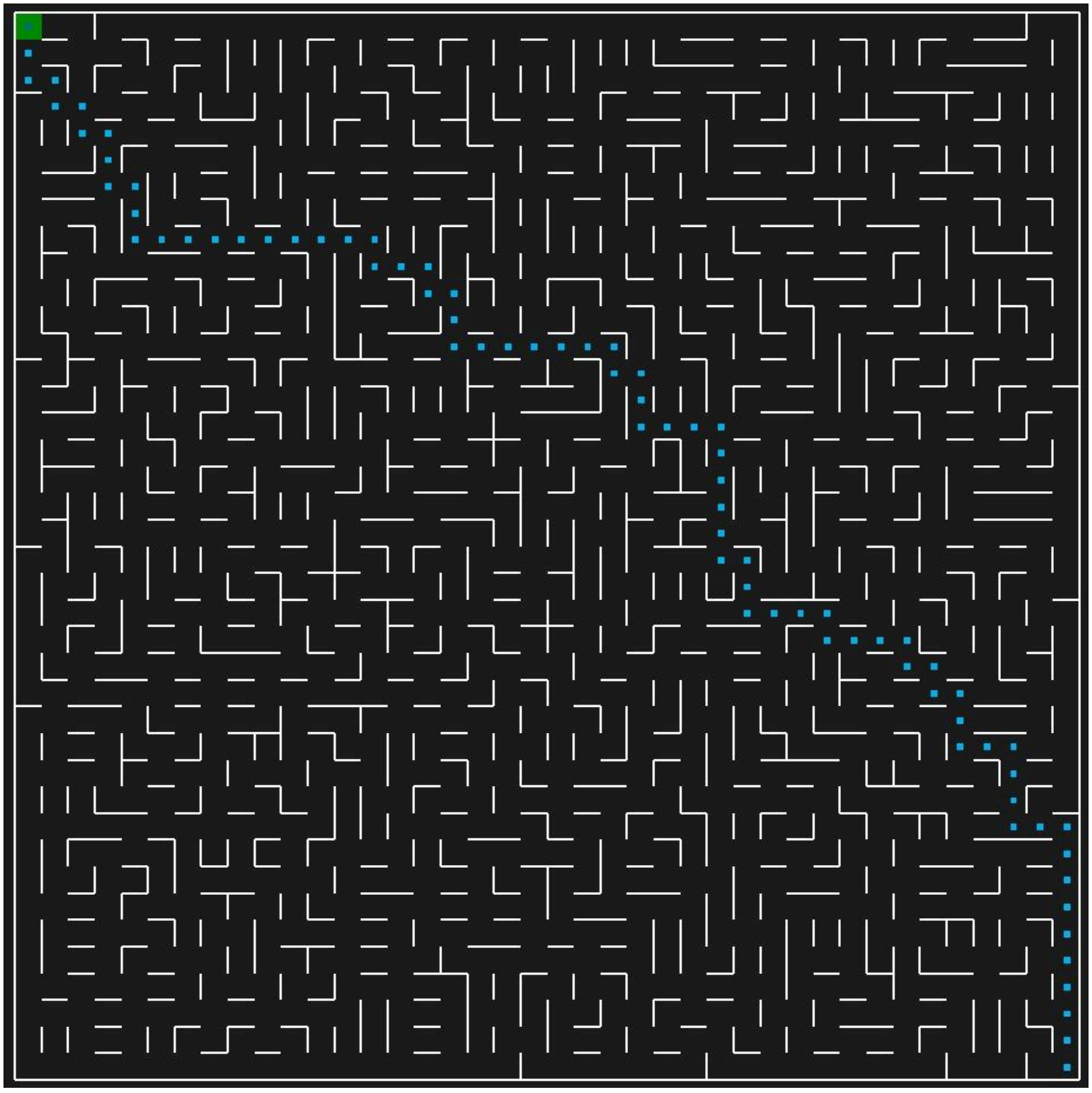
Episode Count: 5000
Step count: 200

40 x 40 Maze



Episode Count: 10,000
Step count: 1000

50 x 50 Maze



Episode Count: 20,000
Step count: 3000