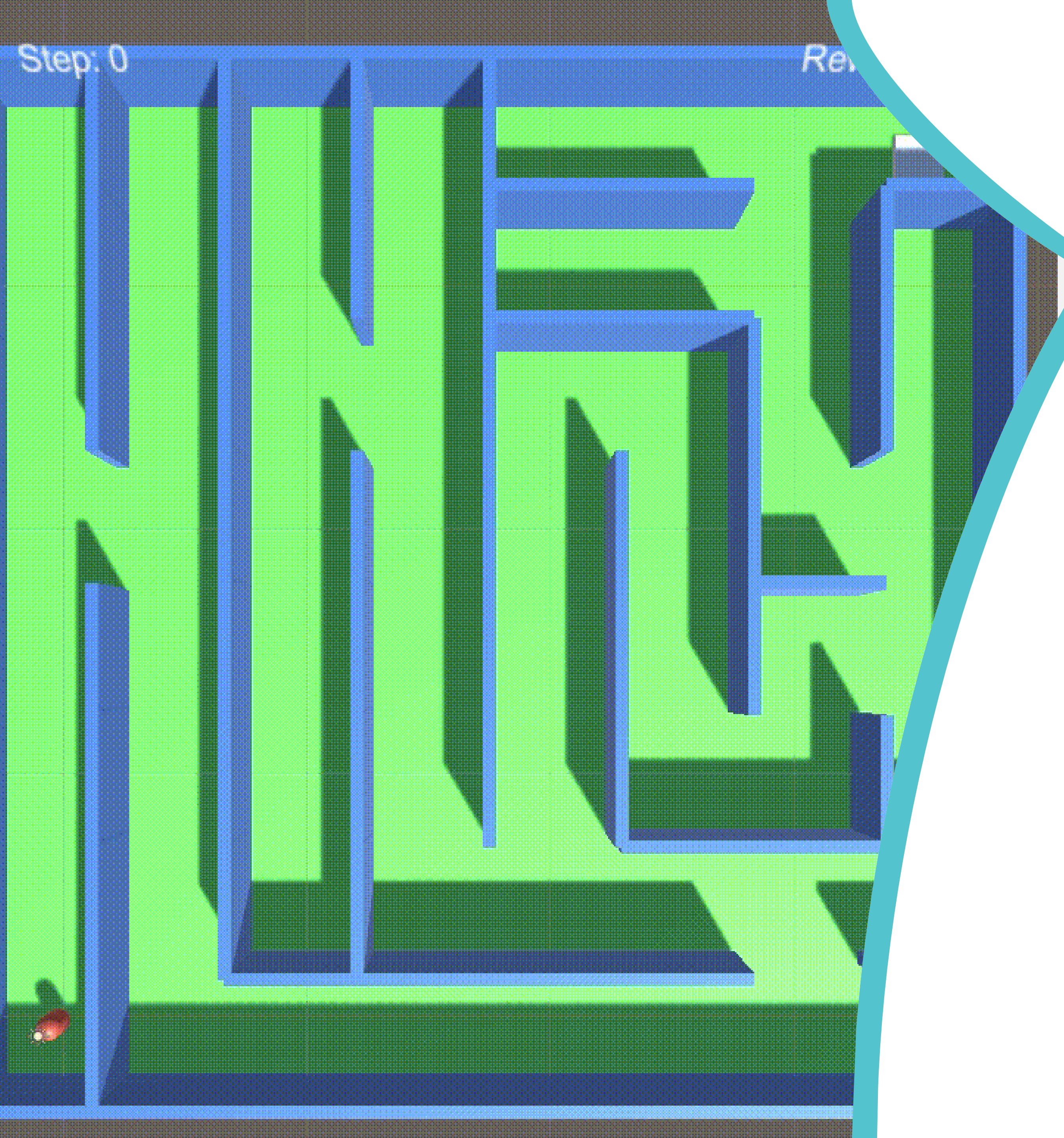


QS STARS RATED FOR EXCELLENCE 2012

FPT Education

FPT UNIVERSITY

CST491_G4

Tuning Proximal Policy Optimization Algorithm in Maze Solving with ML-Agents

CST491_G4

QS STARS RATED FOR EXCELLENCE 2012

FPT Education

FPT UNIVERSITY

Phan Duy Hùng

LECTURER AT FPT UNIVERSITY

Mac Duy Dan Truong

STUDENT OF FPT UNIVERSITY

Phan Thanh Hùng

STUDENT OF FPT UNIVERSITY

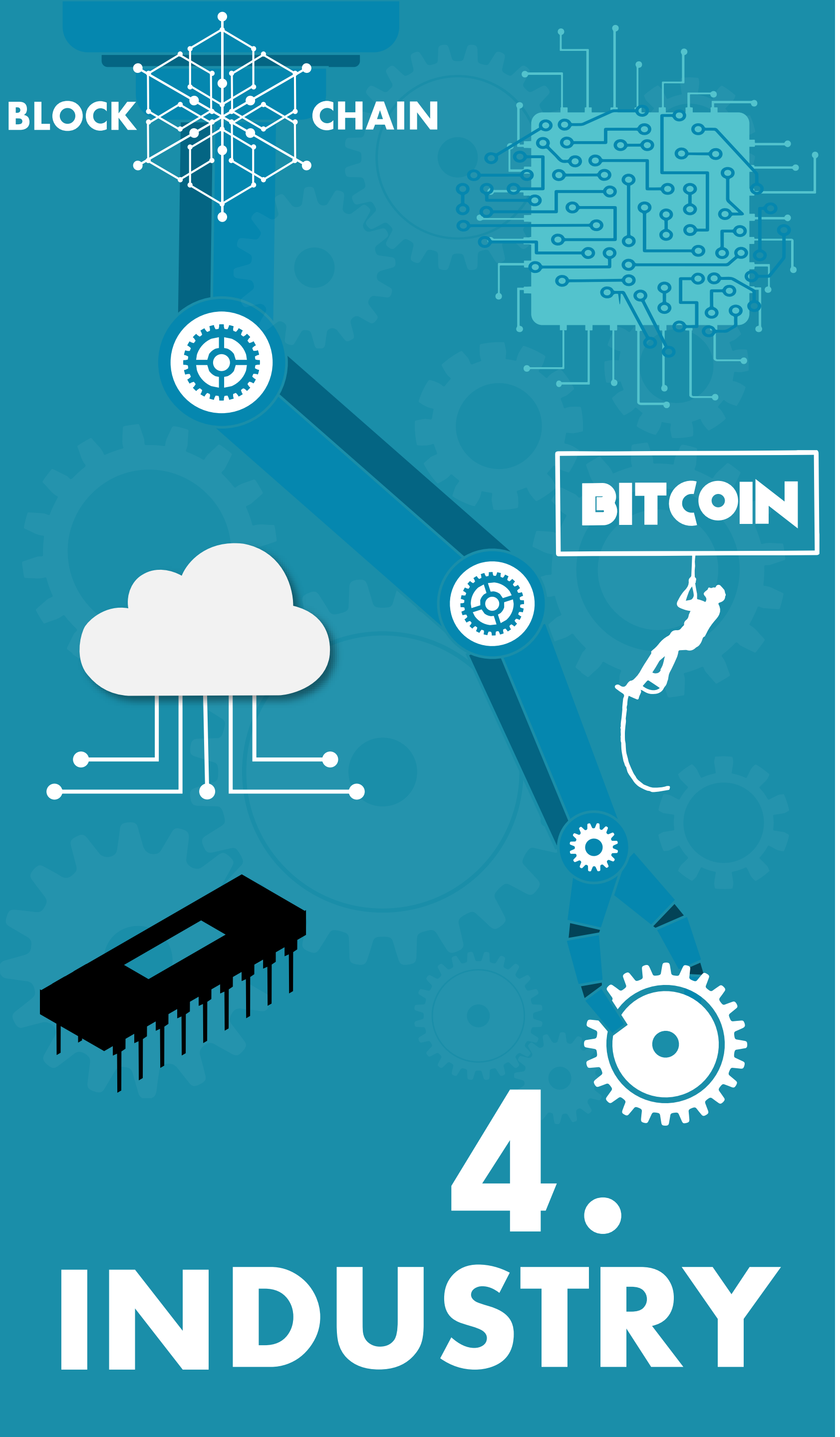
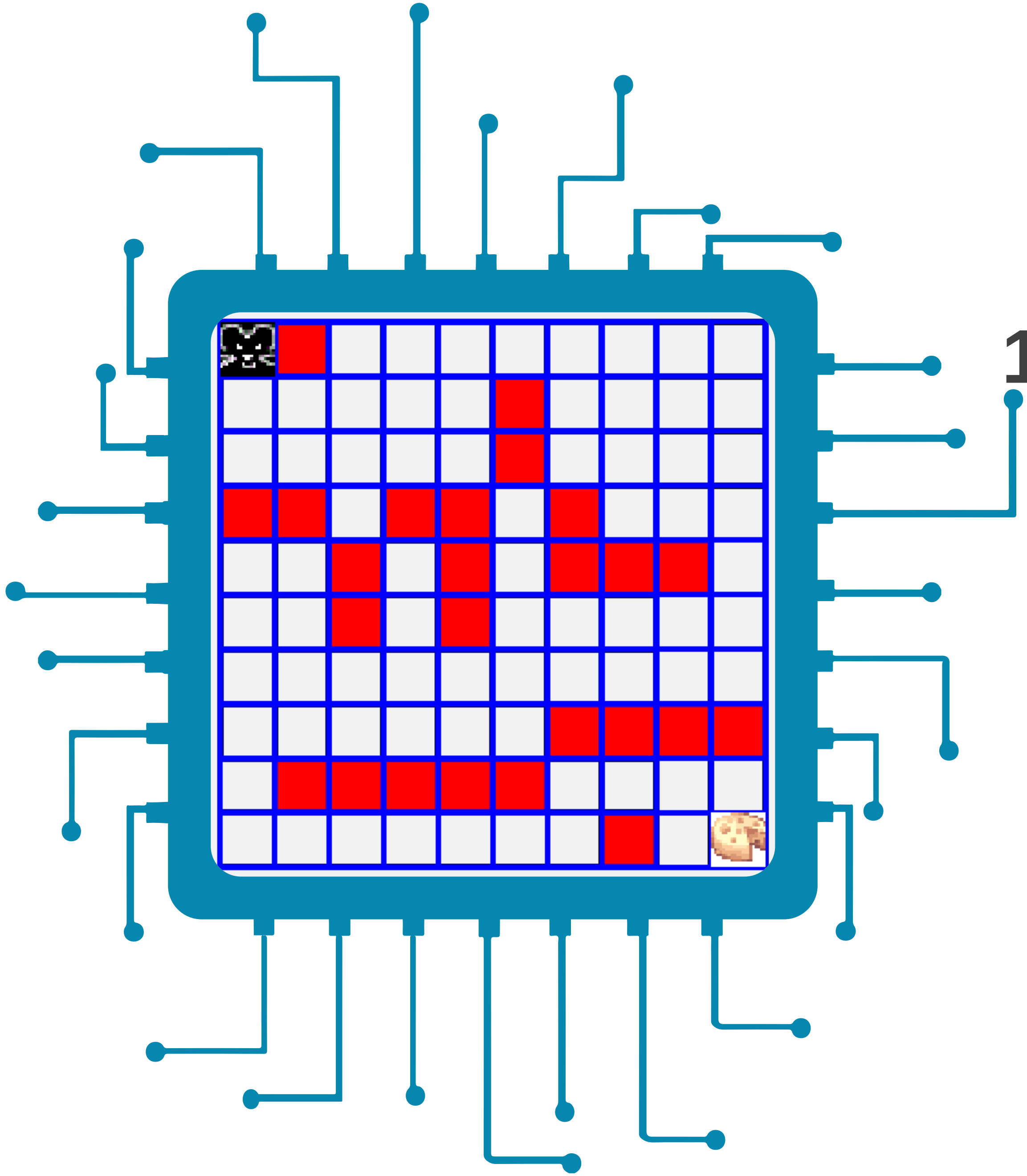


Table of Contents



1.A. Objective of the project

Today, more projects use automated software as a substitution for humans.

One of them is a puzzling maze that consists of a different branch of passages where the solver aims to reach the destination by finding the most efficient route within the shortest possible time.

1.A. Objective of the project

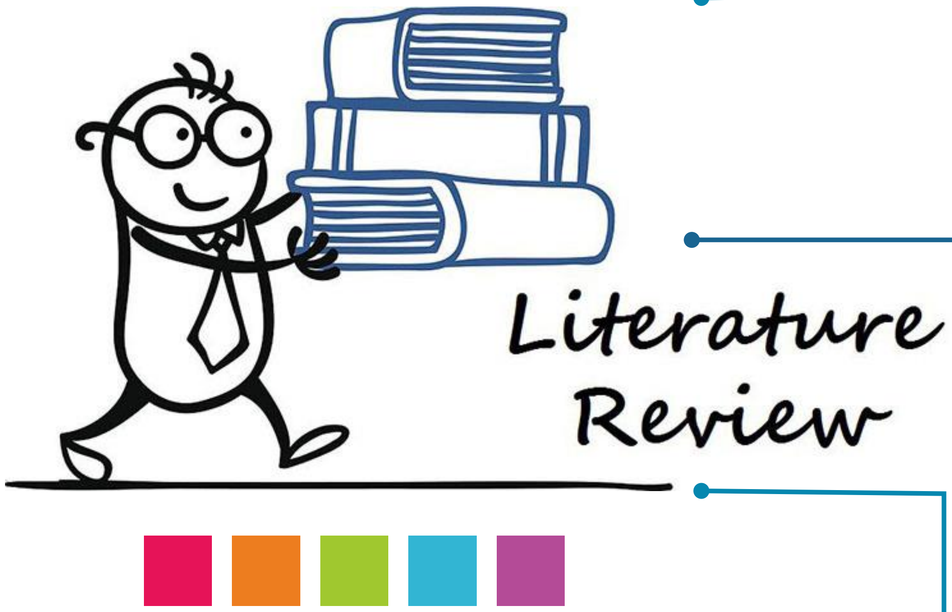
A vital issue in the usability of an RL method is sensitivity to hyperparameters.

Learning complex tasks can take hours or days, fine-tuning hyperparameters is tedious.

Thus, this research focuses on changing the hyperparameters (Beta, Epsilon, Lambd, Num_epcho)

Batch size	64
Buffer size	10240
# of Hidden Units	256
Time Horizon	1024
Learning Rate	10^{-3}
Gamma	0.99
η	0.1
β	0.01
μ	0.01

1.B. Literal reviews



J. Schulman
Jul. 2017

“Proximal Policy Optimization Algorithms,”
The algorithm was successful on various problems without tuning hyperparameter values, meaning that the results still did not achieve the best possible outcome.

J. T. Kristensen
Jul. 2020

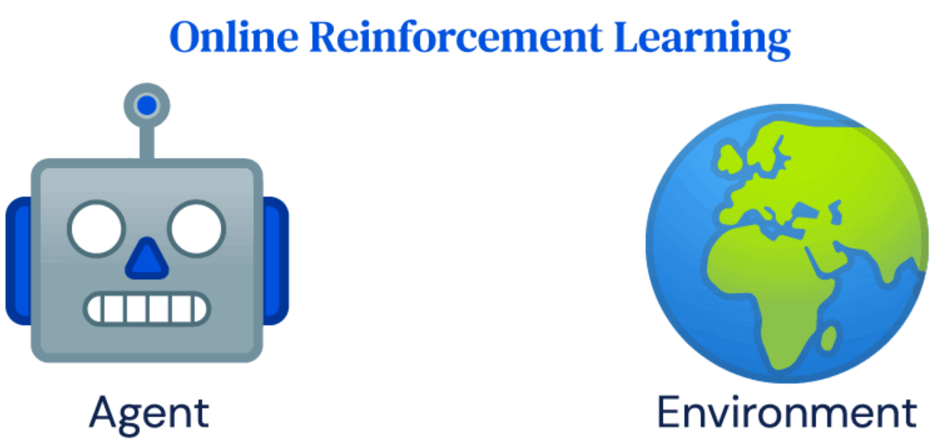
“Strategies for Using Proximal Policy Optimization in Mobile Puzzle Games,”
Successfully adapted the popular RL method PPO to a production-grade puzzle game where the environment is reset after a fixed number of steps, but not considering hyperparameter tuning

A. Valdivia
Aug. 2020

“Estimating Player Completion Rate in Mobile Puzzle Games Using Reinforcement Learning,”

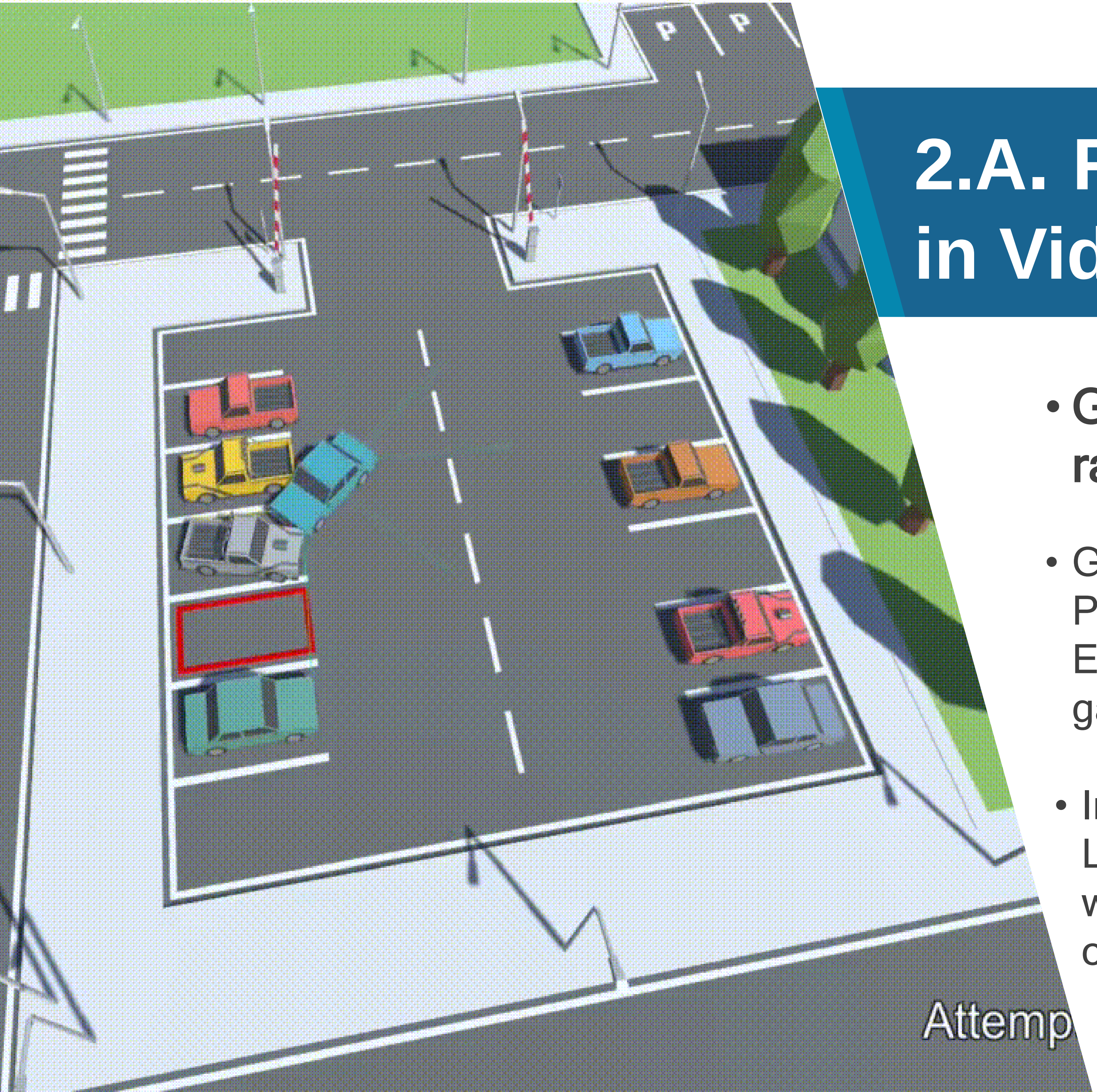
The work is only for a limited subset of sample of ~900,000 players with default values hyperparameter.

2.A. Introduction to Reinforcement Learning



Agent learns by interacting

- Reinforcement Learning is the training of Machine Learning models to make a sequence of decisions.
- Computers use trial and error to come up with solutions to problems
- The agent receives feedback in terms of a reward (or punishment) from the environment



2.A. Reinforcement Learning in Video Games

- **Gaming is a booming industry and is rapidly advancing with technology.**

- Game developers with environments like PSXLE or PlayStation Reinforcement Learning Environment focus on providing a better gaming environment by modifying the emulator.

- In addition, Reinforcement Learning has Deep Learning algorithms like AlphaGo, Alpha Zero which are gaming algorithms for games like chess, shogi, and Go.

2.B. Proximal Policy Optimization (PPO)

- Proximal Policy Optimization (PPO) is an optimization approach that uses solely first-order optimization to improve the data efficiency and reliability of Trust Region Policy Optimization (TRPO).

- The primary objective function of PPO is:

$$L^{CLIP}(\theta) = \hat{E}_t\left[\min(\theta)\,\hat{A}_t, clip(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t\right]$$

(θ) is Policy Parameter

$\hat{E}_t$ Denotes the empirical expection over timesteps

r_t is the ratio of the probability under the new and old policies, respectively

$\hat{A}_t$ is the estimated advantage at time t

ϵ is a hyperparameter, usually 0.1 and 0.2

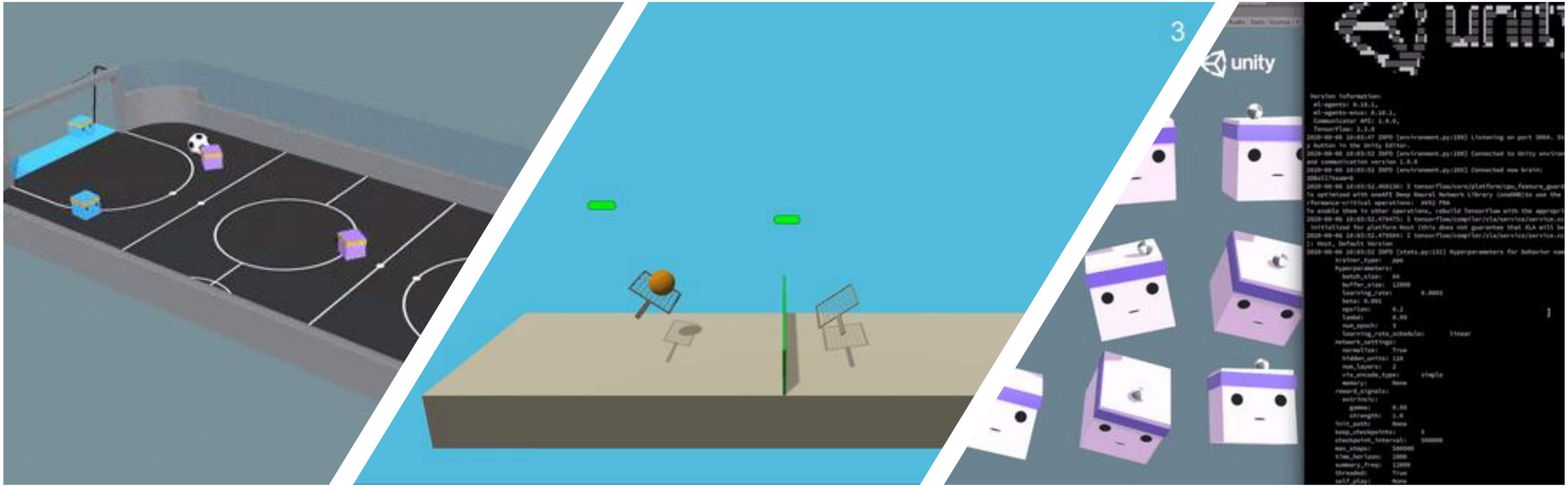
2.B. Proximal Policy Optimization (PPO)

The steps of the PPO algorithm are:

Algorithm 1 PPO, Actor-Critic Style

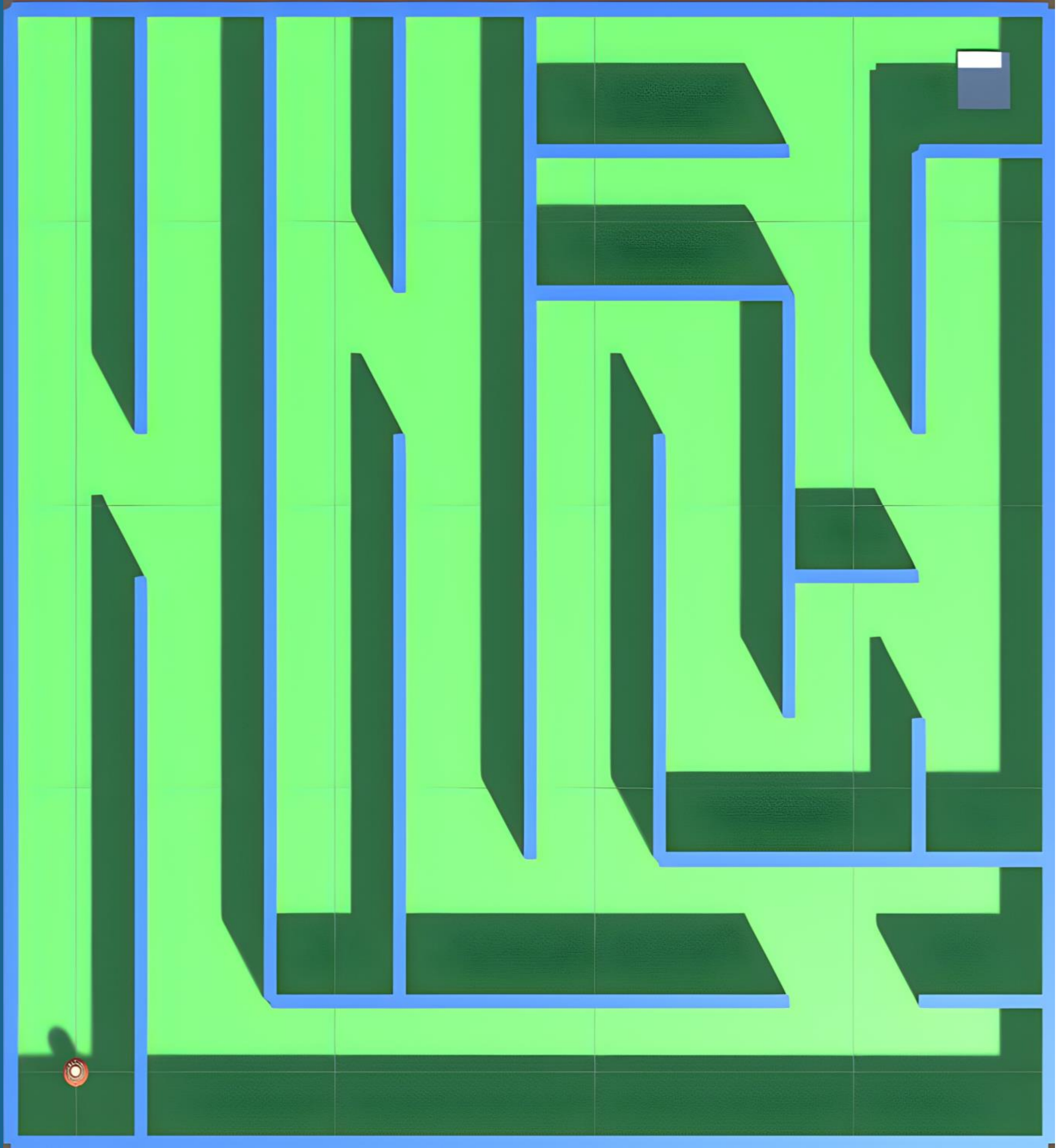
for iteration=$1, 2, \ldots$ **do**
    for actor=$1, 2, \ldots, N$ **do**
        Run policy $\pi_{\theta_{\text{old}}}$ in environment for T timesteps
        Compute advantage estimates $\hat{A}_1, \ldots, \hat{A}_T$
    end for
    Optimize surrogate L wrt θ, with K epochs and minibatch size $M \leq NT$
    $\theta_{\text{old}} \leftarrow \theta$
end for

2.C. ML-Agents Tool kit



- ML-Agents Toolkit is an open-source project that enables games and simulations to serve as environments for training intelligent agents.
- Using a simple-to-use Python API, Agents are trained using reinforcement learning, imitation learning, neuroevolution, or other machine learning methods.

Fixed Maze 8x8

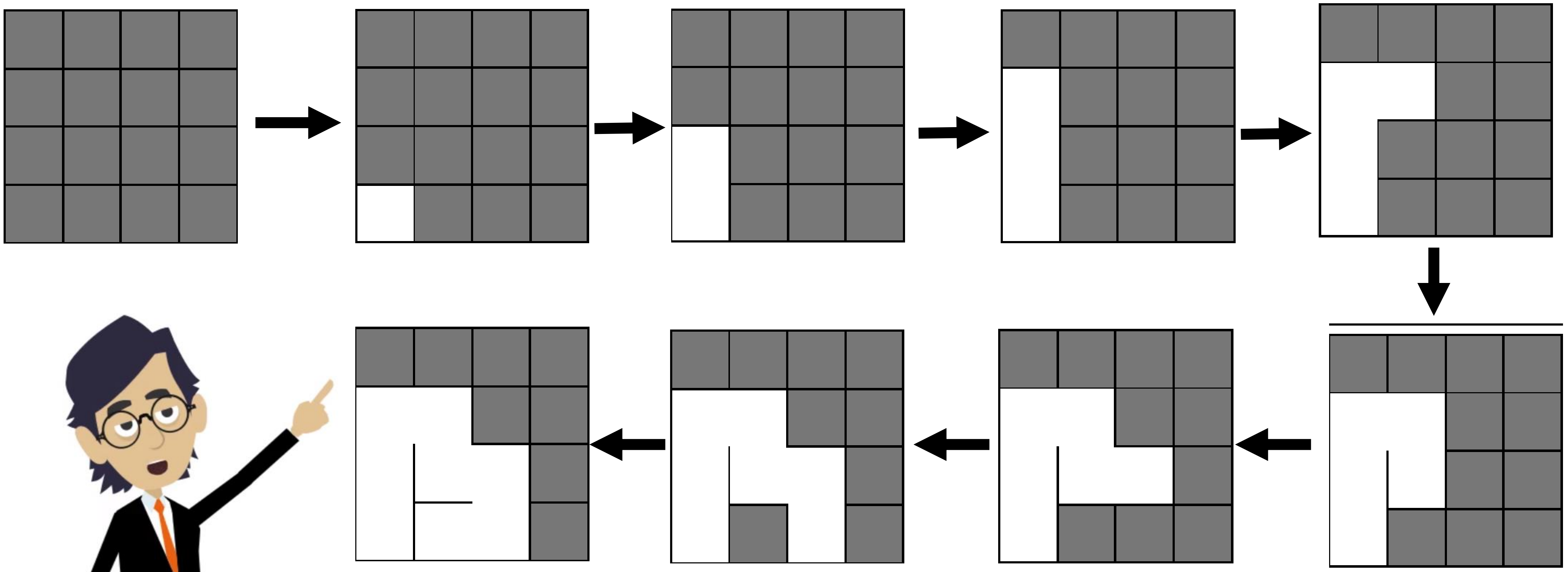


3.A. Maze design

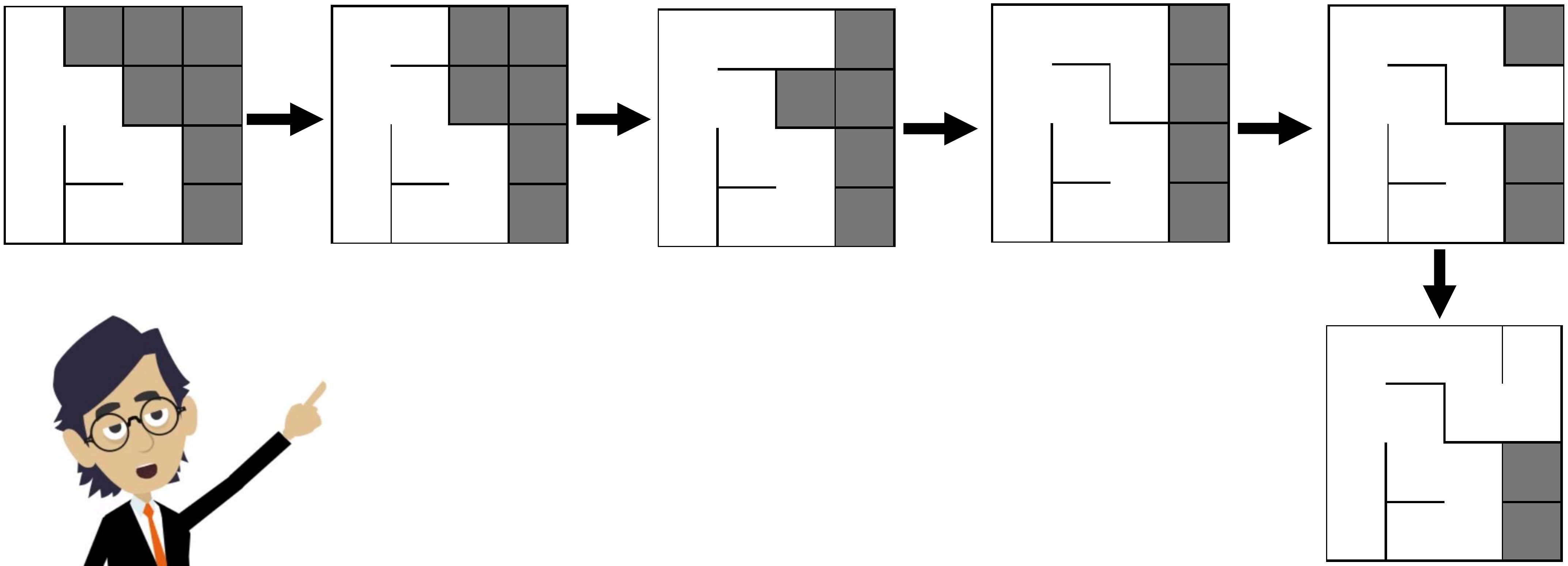
Maze design for training Agent:

- A fixed maze has 8x8 cells.
- A random maze has 3 different variants: 4x4, 6x6, 8x8 cells
- A cell includes four walls and one floor.
- There is a destination for the Agent to complete the maze. Collision with it will end an episode.

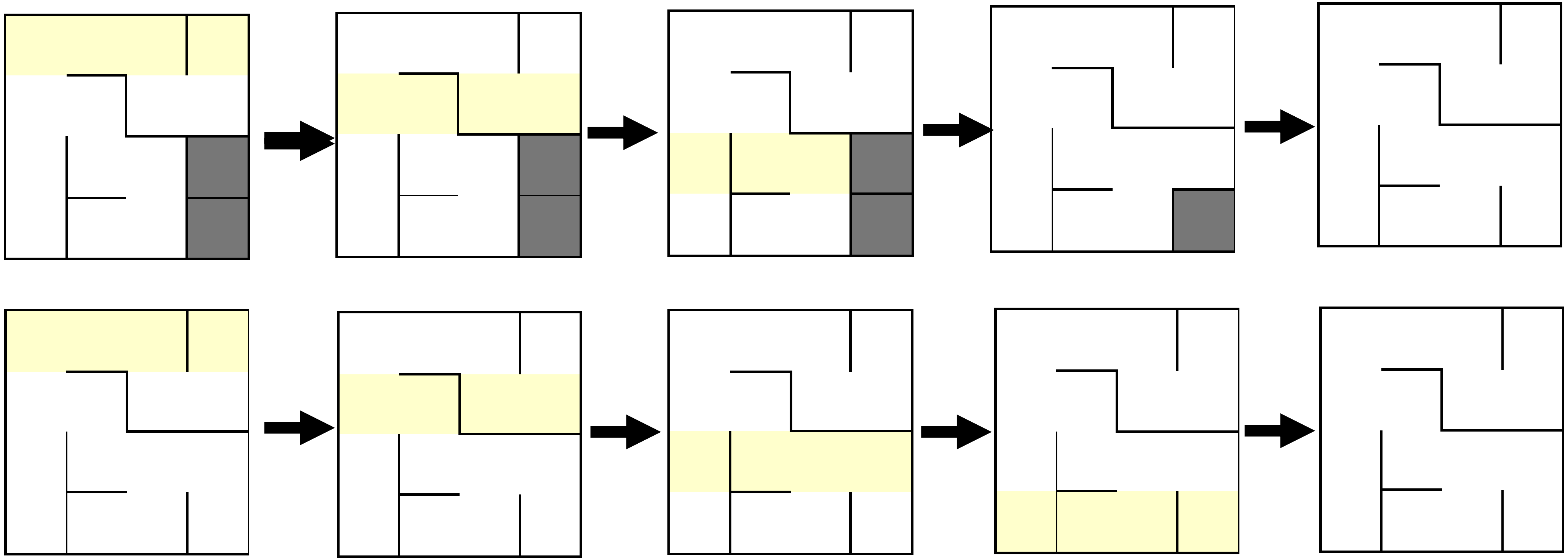
3.B. Hunt & Kill Algorithm



3.B. Hunt & Kill Algorithm



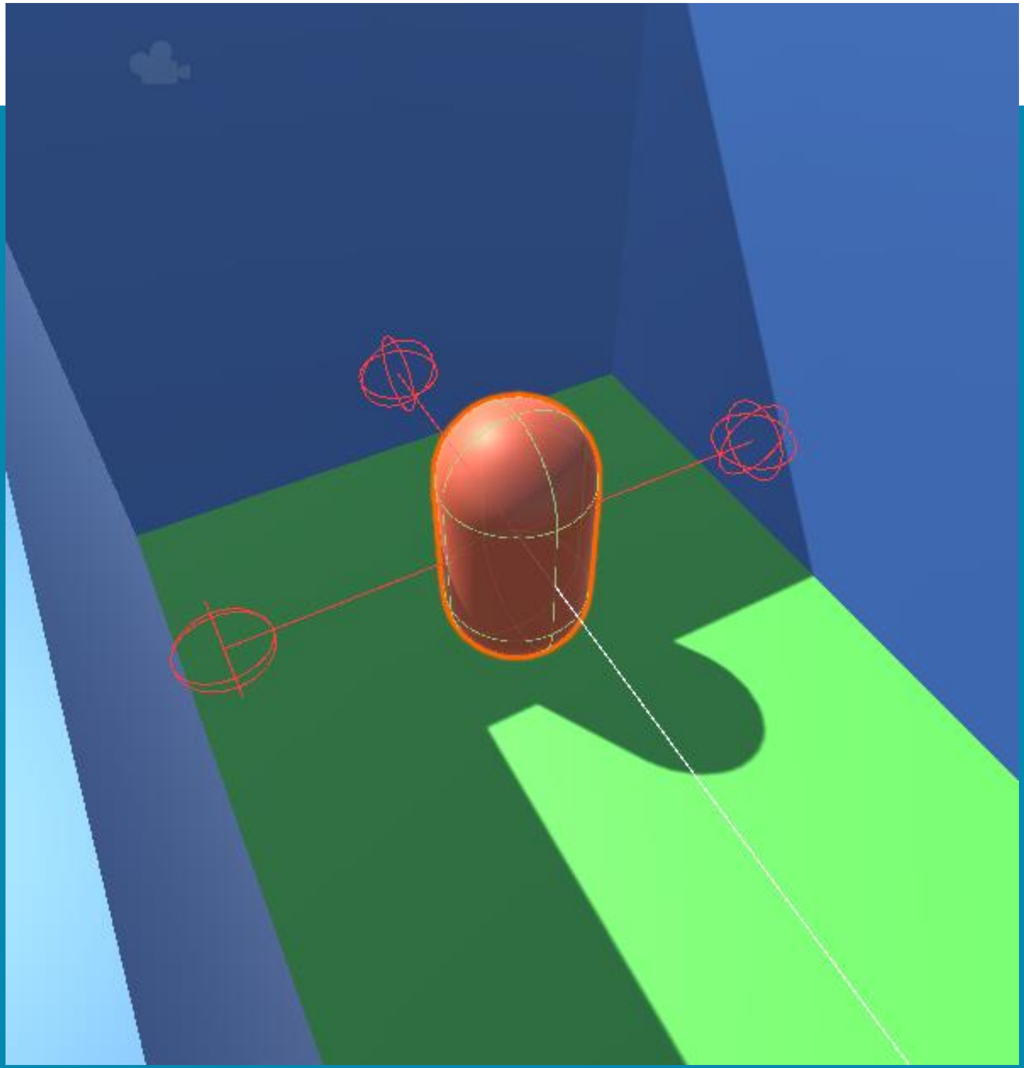
3.B. Hunt & Kill Algorithm



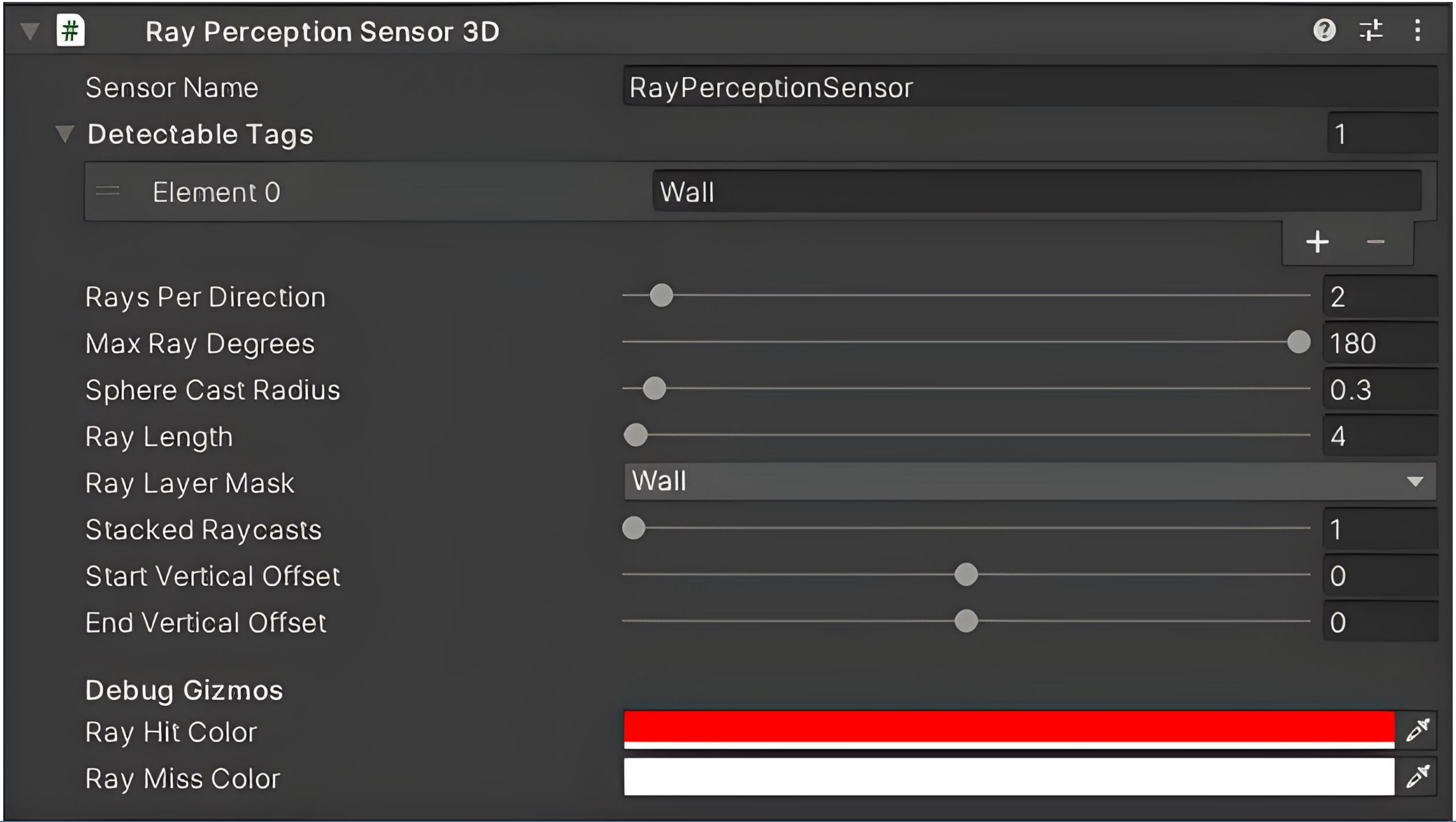
3.C. Agent Behavior

Agent’s Observation

The Agent has four raycasts on four sides around the Agent. The length of the raycast is one cell. The Agent has four 3D Ray Perception Sensors - the Agent's observations.



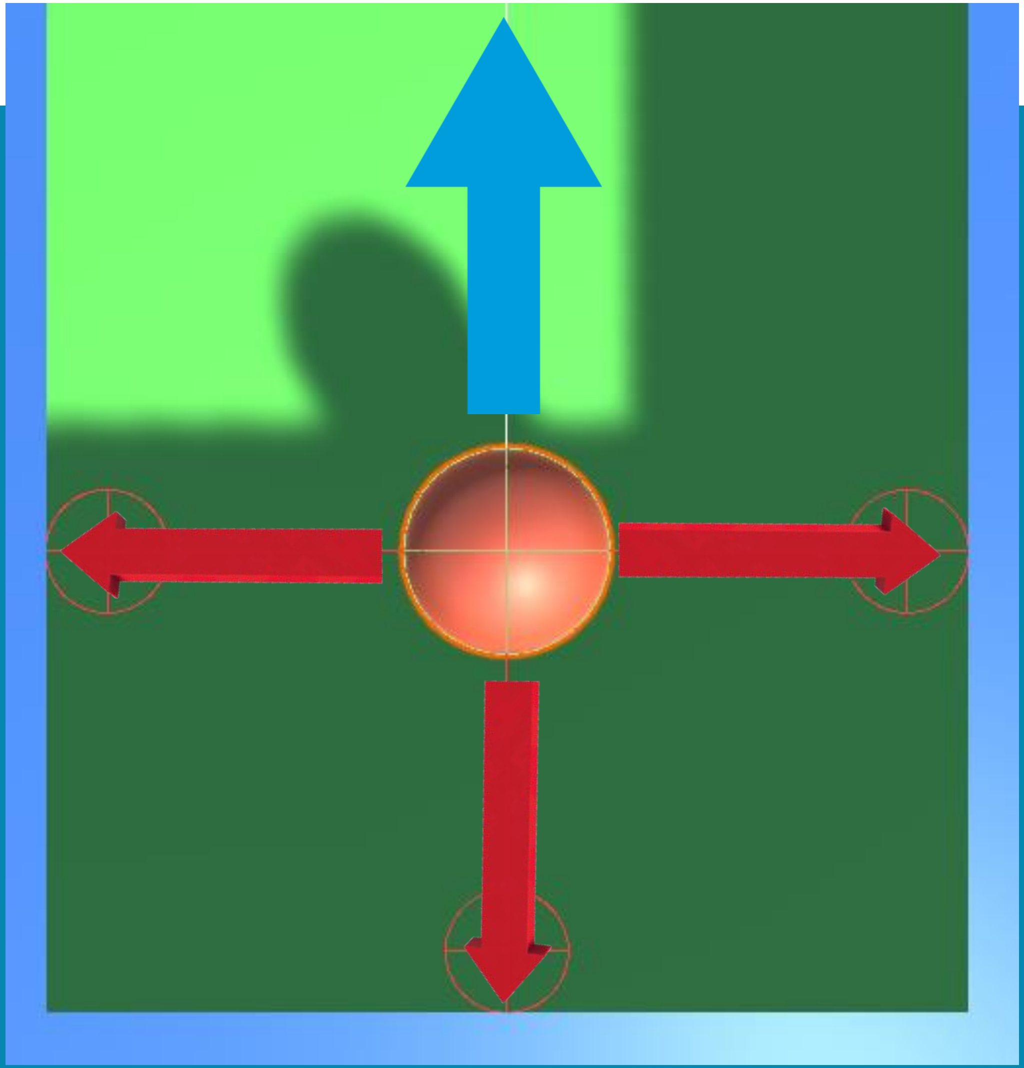
3.C. Ray Perception Sensor



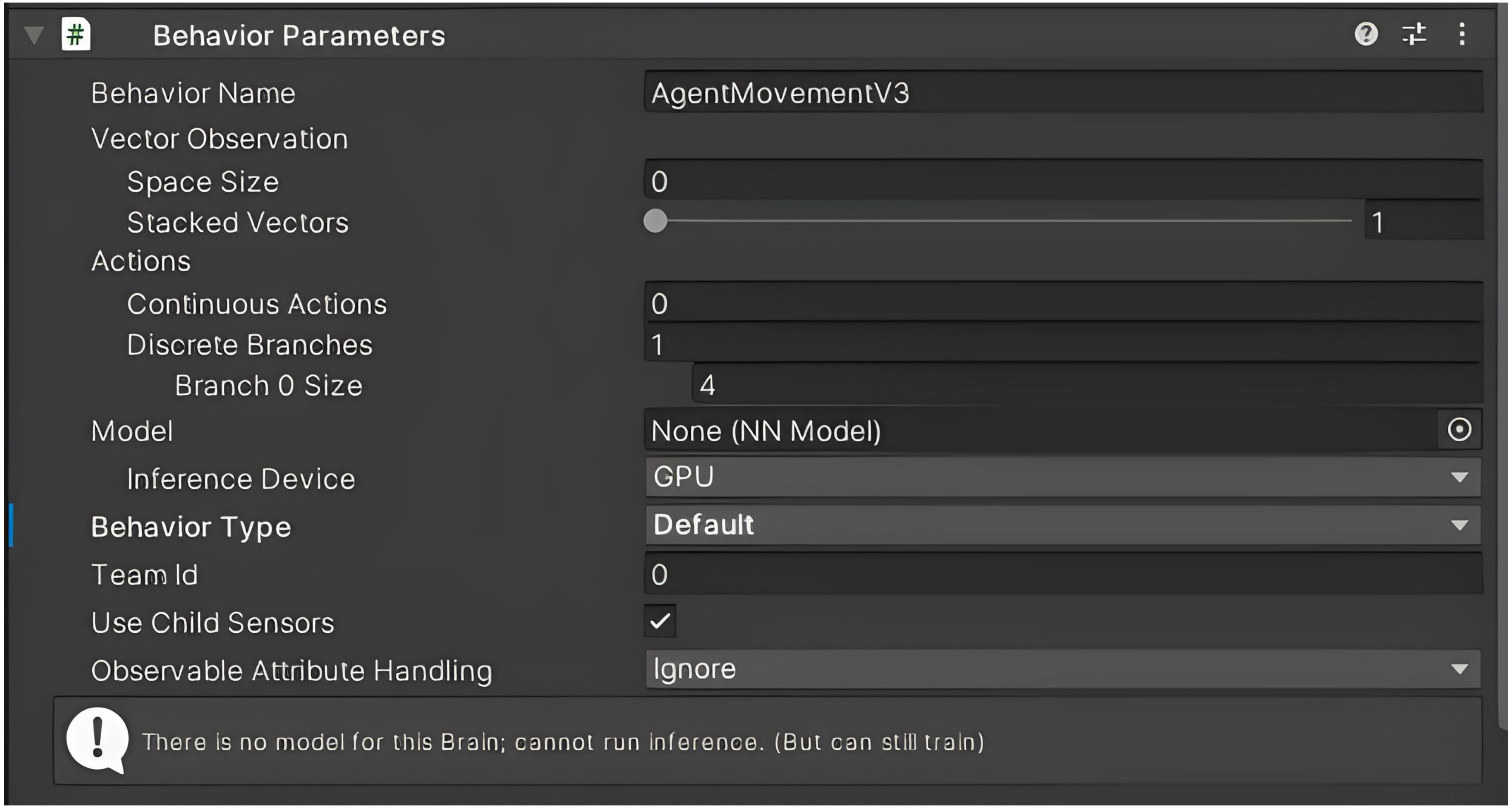
3.D. Agent behavior

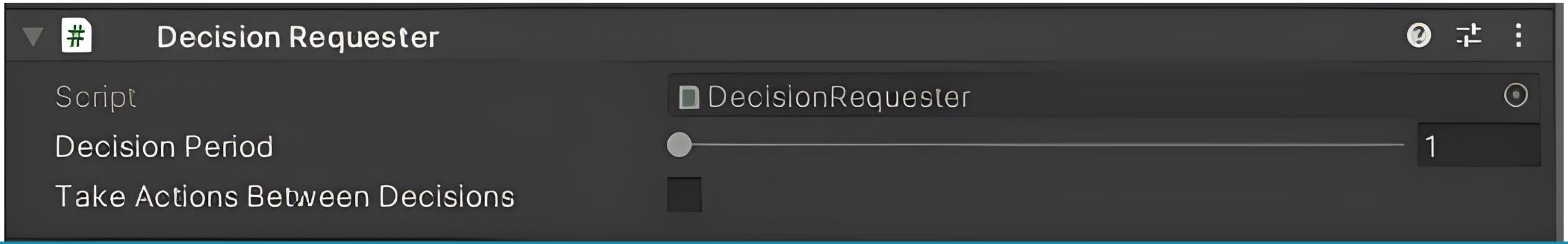
Agent’s Raycast

- Agent’s Raycast keeps Agent from being moved out of the maze.
- Raycast works as detecting colliders on the front and on 4 sides.
- When Raycast detects the walls, it will not allow Agent to go through that wall and keep Agent in the maze.



3.D. Agent's Behavior





3.D. Agent Detail Behavior

01

02

03

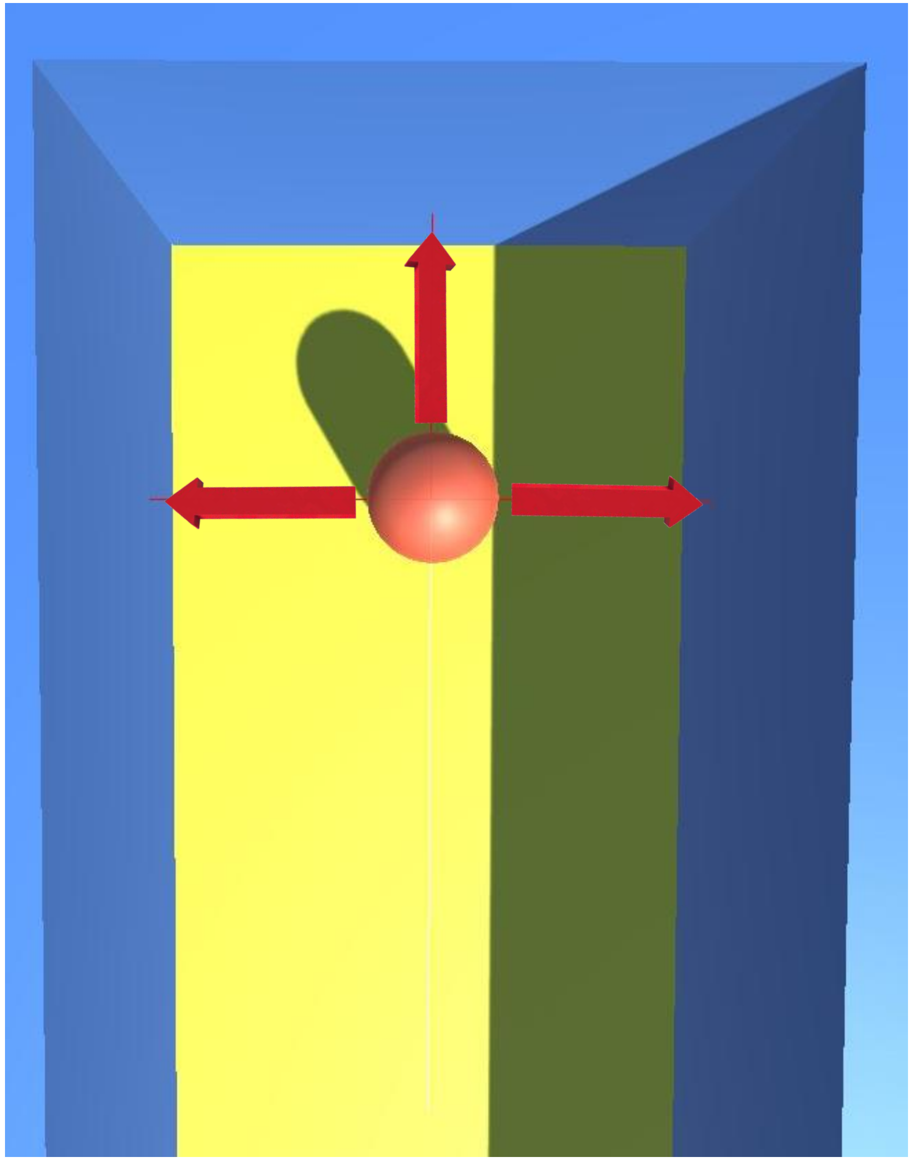
04

05

06

Agent when entering a cell will award or punished:

When the Agent moves in a specific direction and that side's raycast detects the wall, but the Agent still decides to go in that direction, -1 point.



3.D. Agent Detail Behavior

01

02

03

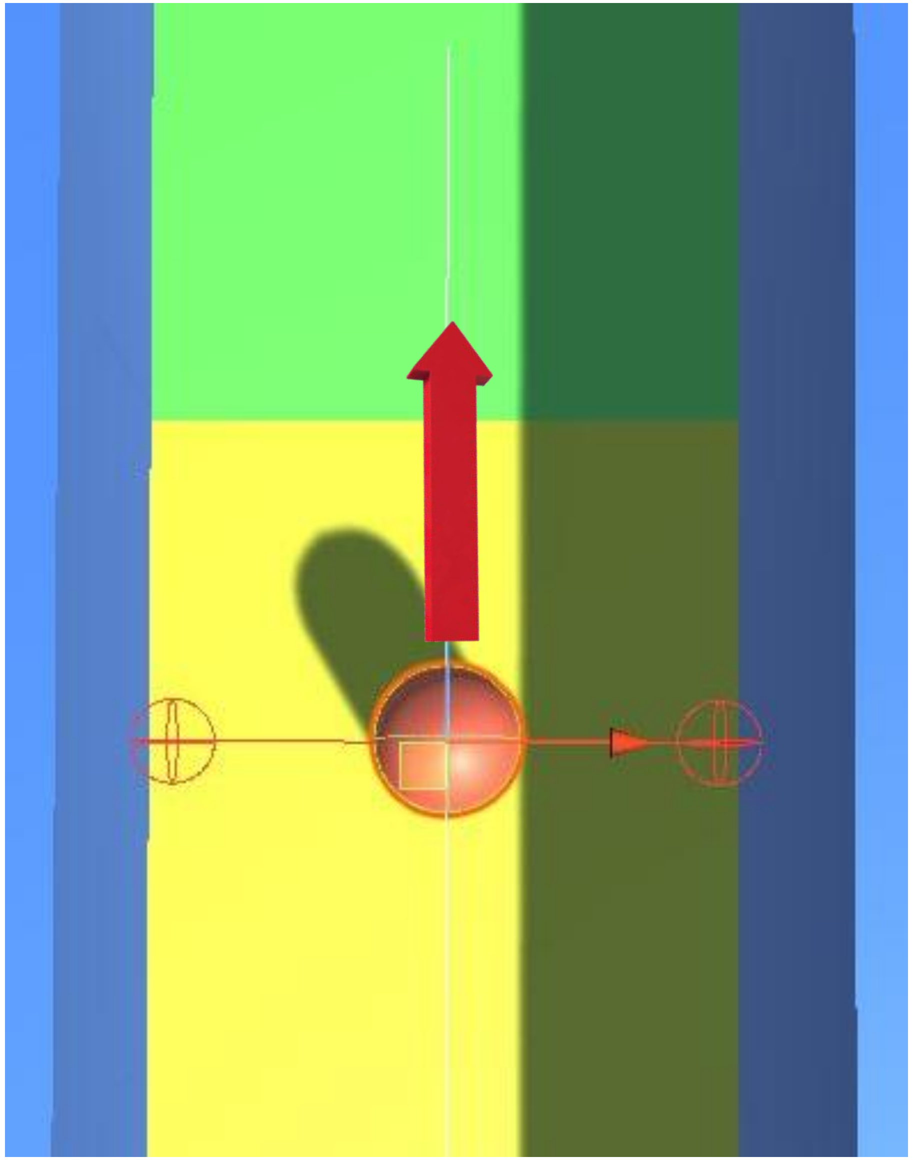
04

05

06

Agent when entering a cell will award or punished:

- When the Agent moves in a specific direction and that side's raycast detects the wall, but the Agent still decides to go in that direction, -1 point.
- Entering for the first time, Agent +3 points, and that background box turns yellow.



3.D. Agent Detail Behavior

01

02

03

04

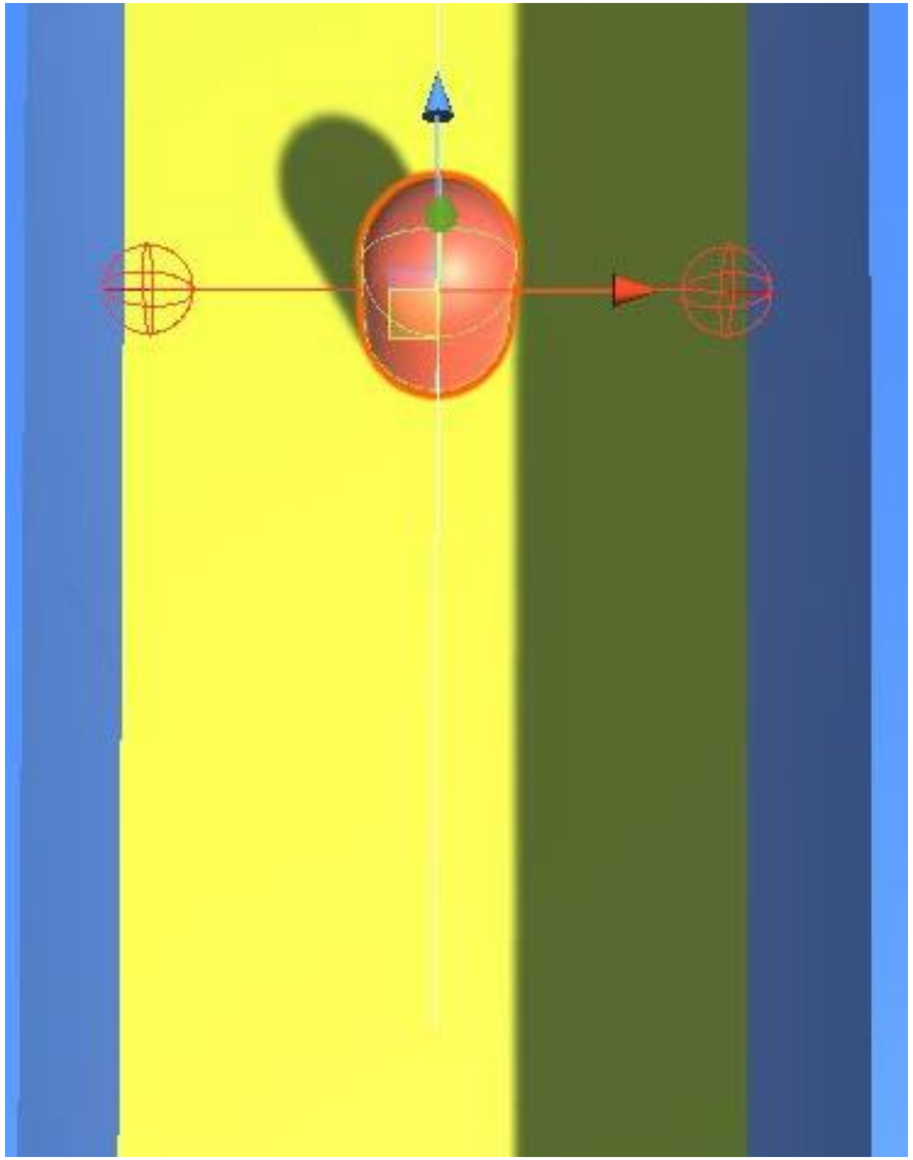
05

06

Agent when entering a cell will award or punished:

When the Agent moves in a specific direction and that side's raycast detects the wall, but the Agent still decides to go in that direction, -1 point.

Entering for the first time, Agent +3 points, and that background box turns yellow.



3.D. Agent Detail Behavior

Agent when entering a cell will award or punished:

01 When the Agent moves in a specific direction and that side's raycast detects the wall, but the Agent still decides to go in that direction, -1 point.

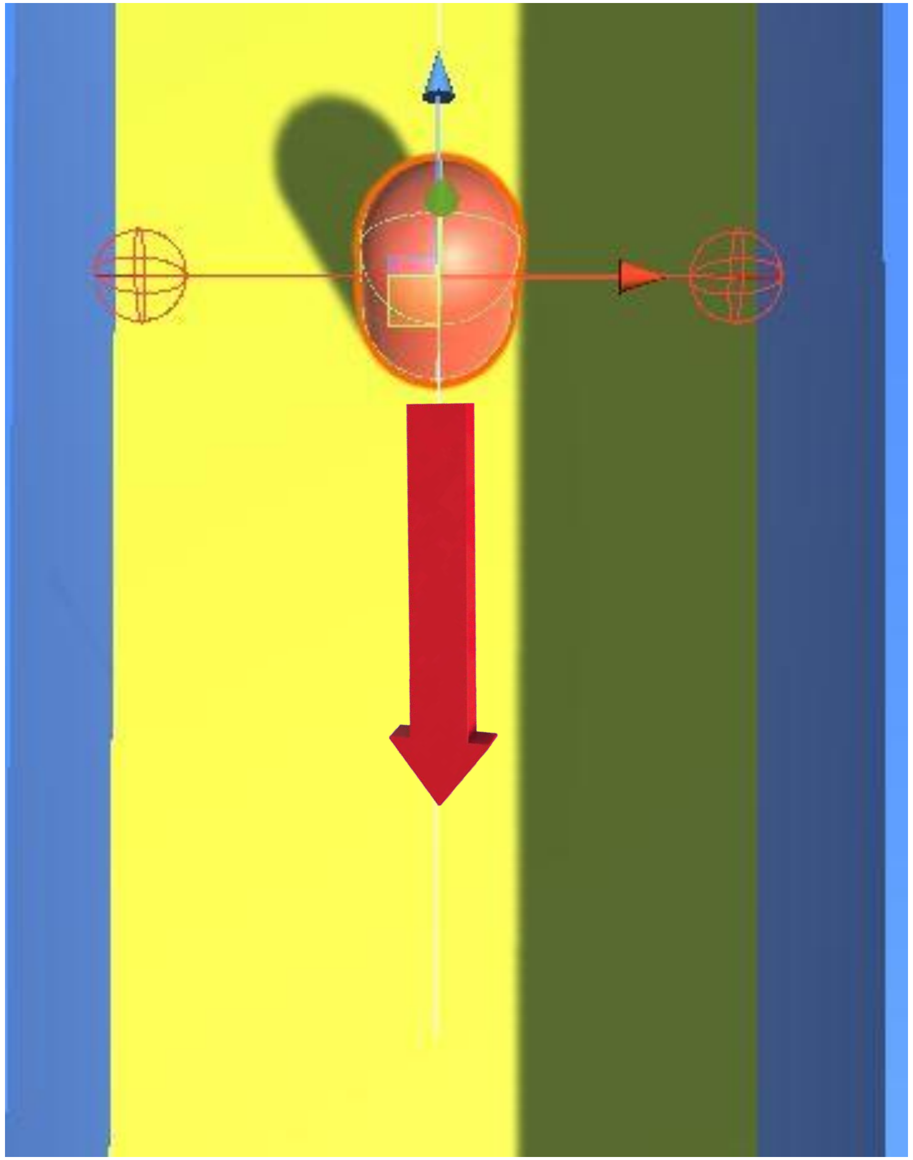
02 Entering for the first time, Agent +3 points, and that background box turns yellow.

03 Entering the second time, the Agent -0.5, and the background cell turns orange.

04

05

06



3.D. Agent Detail Behavior

01

02

03

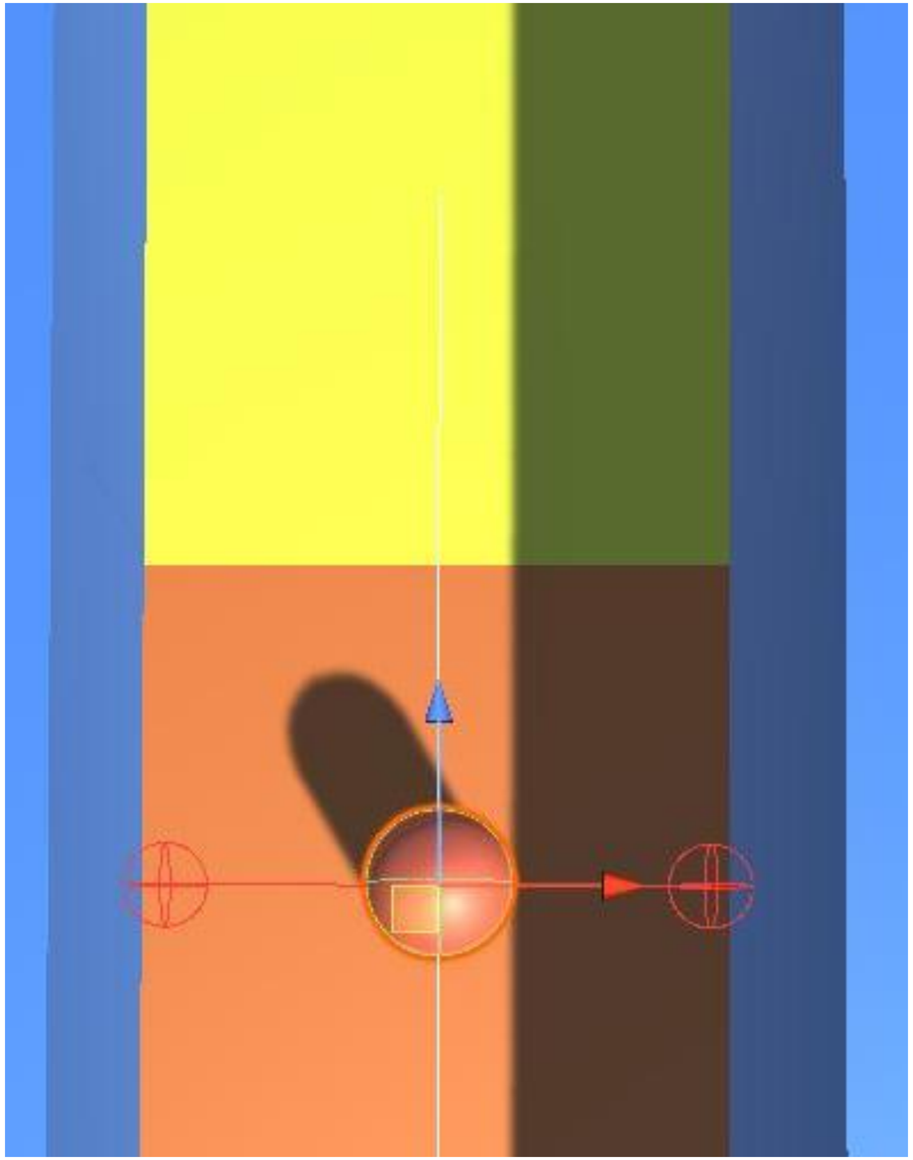
04

05

06

Agent when entering a cell will award or punished:

- When the Agent moves in a specific direction and that side's raycast detects the wall, but the Agent still decides to go in that direction, -1 point.
- Entering for the first time, Agent +3 points, and that background box turns yellow.
- Entering the second time, the Agent -0.5, and the background cell turns orange.



3.D. Agent Detail Behavior

Agent when entering a cell will award or punished:

01

When the Agent moves in a specific direction and that side's raycast detects the wall, but the Agent still decides to go in that direction, -1 point.

02

Entering for the first time, Agent +3 points, and that background box turns yellow.

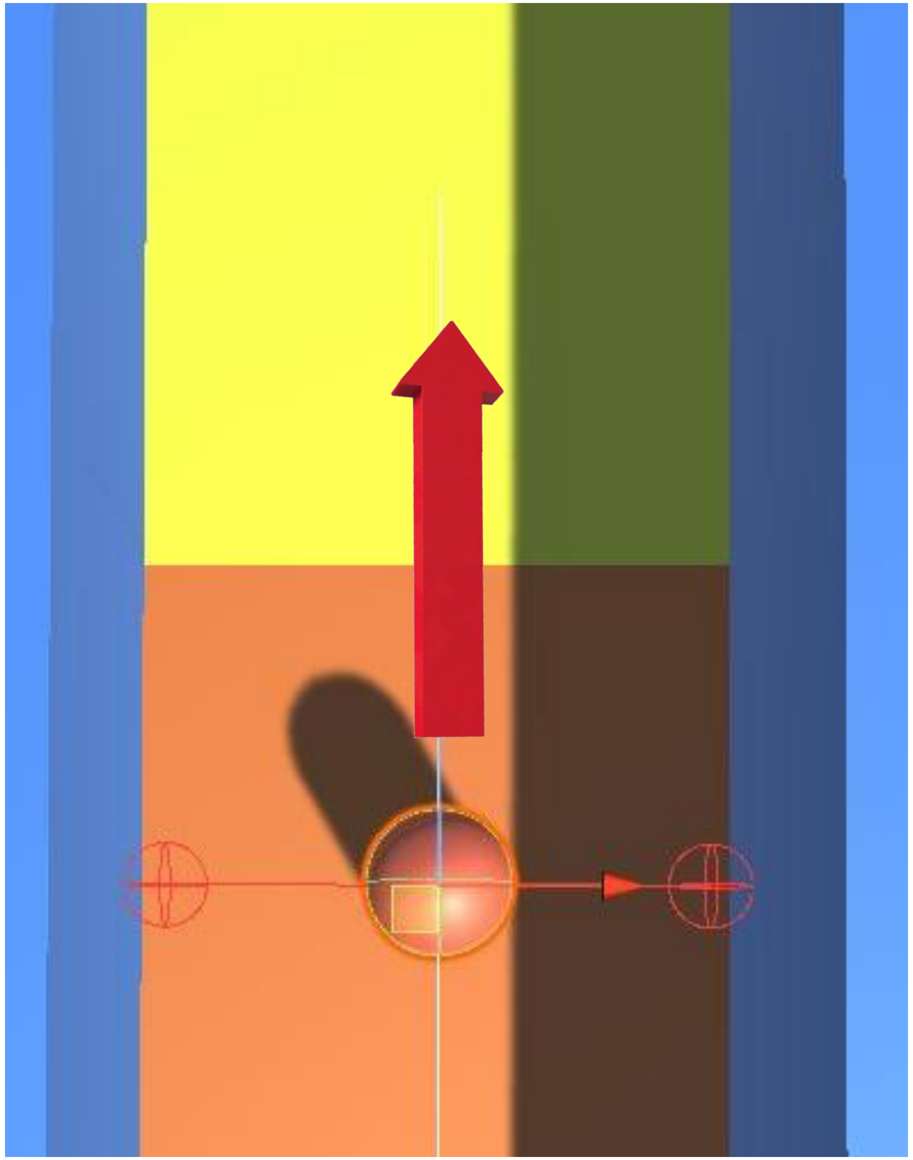
03

Entering the second time, the Agent -0.5, and the background cell turns orange.

04

05

06



3.D. Agent Detail Behavior

Agent when entering a cell will award or punished:

01

When the Agent moves in a specific direction and that side's raycast detects the wall, but the Agent still decides to go in that direction, -1 point.

02

Entering for the first time, Agent +3 points, and that background box turns yellow.

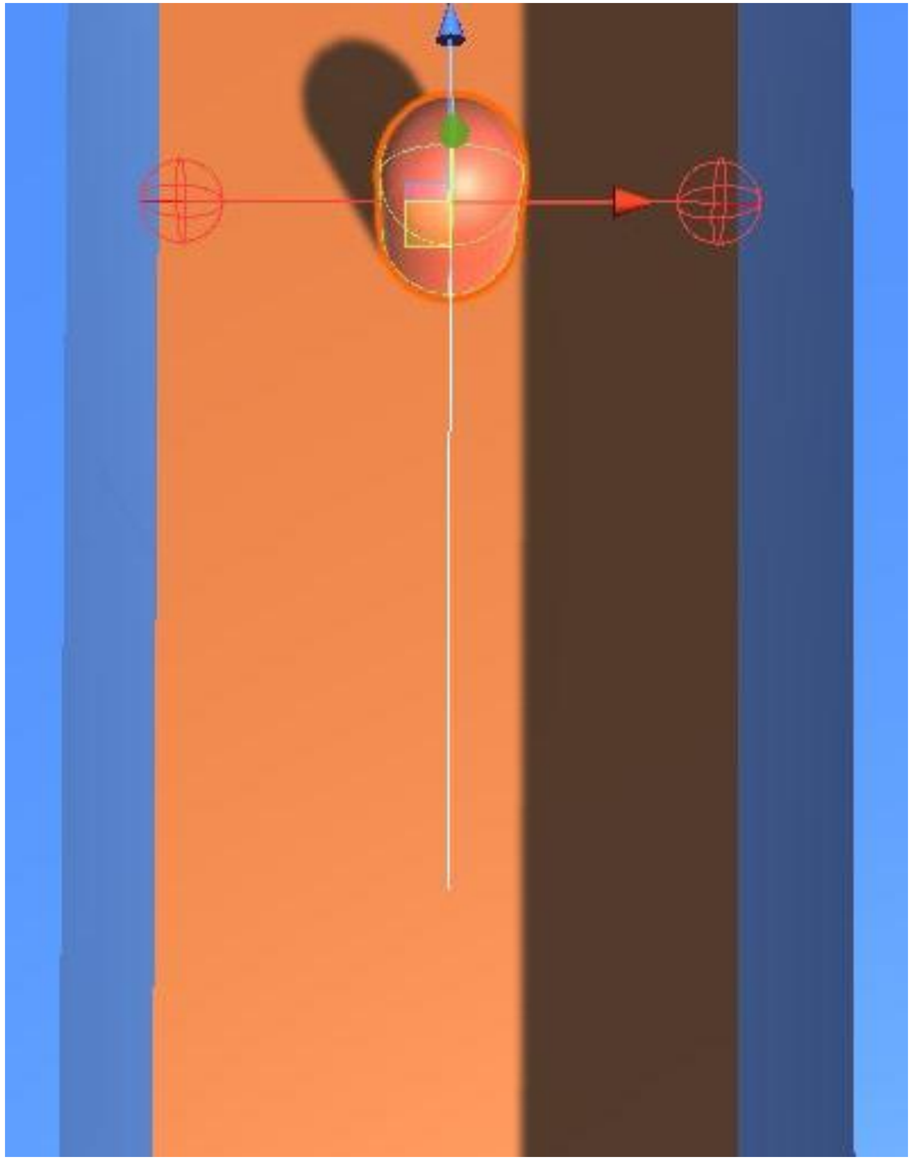
03

Entering the second time, the Agent -0.5, and the background cell turns orange.

04

05

06



3.D. Agent Detail Behavior

Agent when entering a cell will award or punished:

01 When the Agent moves in a specific direction and that side's raycast detects the wall, but the Agent still decides to go in that direction, -1 point.

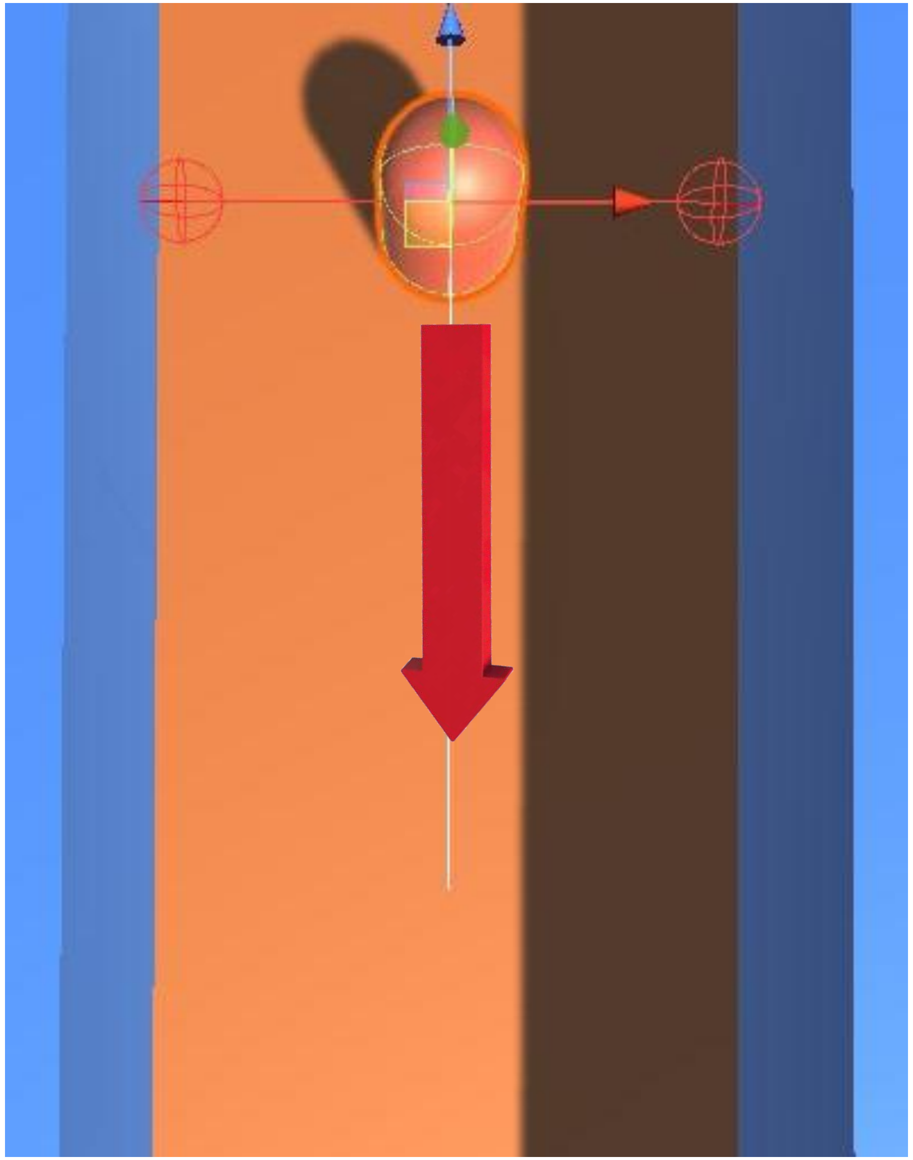
02 Entering for the first time, Agent +3 points, and that background box turns yellow.

03 Entering the second time, the Agent -0.5, and the background cell turns orange.

04 The Agent -1 point the third time, and the background box turns purple.

05

06



3.D. Agent Detail Behavior

Agent when entering a cell will award or punished:

01

When the Agent moves in a specific direction and that side's raycast detects the wall, but the Agent still decides to go in that direction, -1 point.

02

Entering for the first time, Agent +3 points, and that background box turns yellow.

03

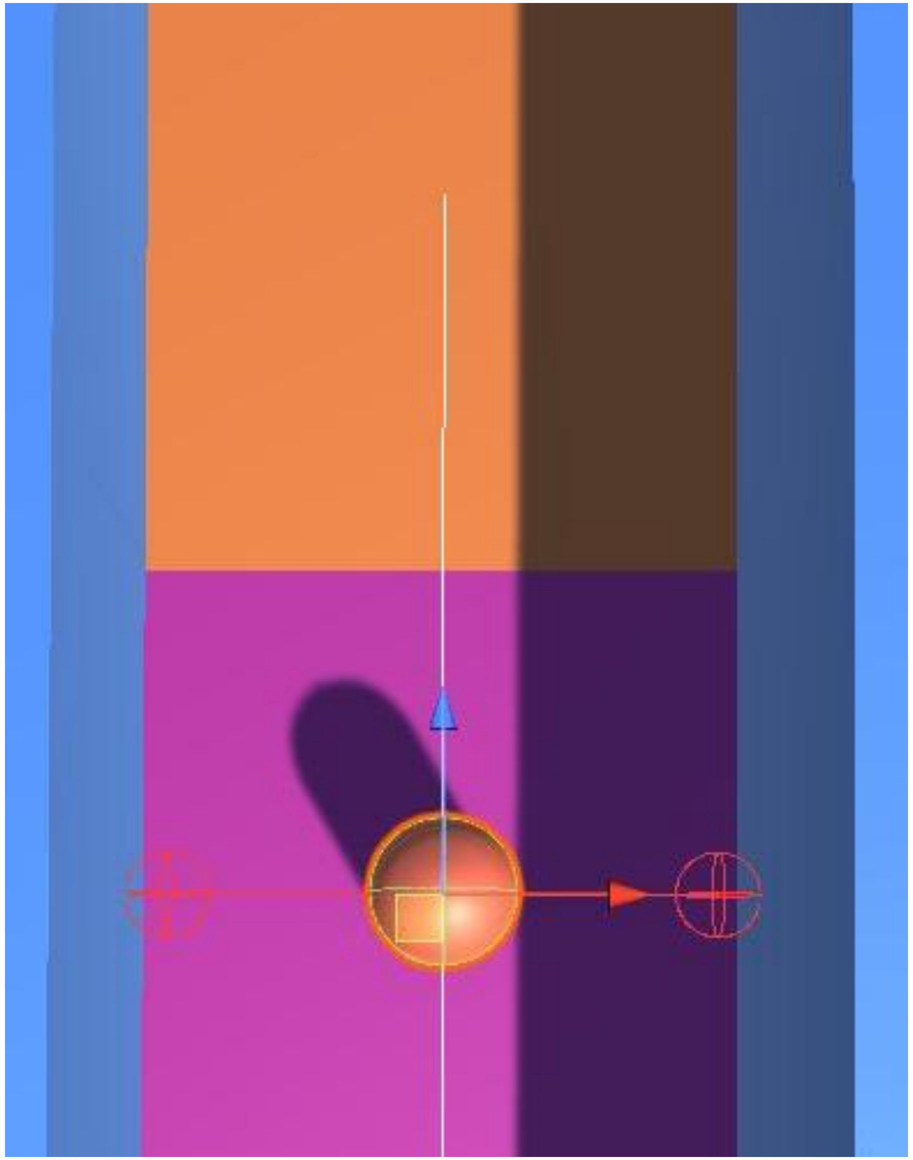
Entering the second time, the Agent -0.5, and the background cell turns orange.

04

The Agent -1 point the third time, and the background box turns purple.

05

06



3.D. Agent Detail Behavior

Agent when entering a cell will award or punished:

01 When the Agent moves in a specific direction and that side's raycast detects the wall, but the Agent still decides to go in that direction, -1 point.

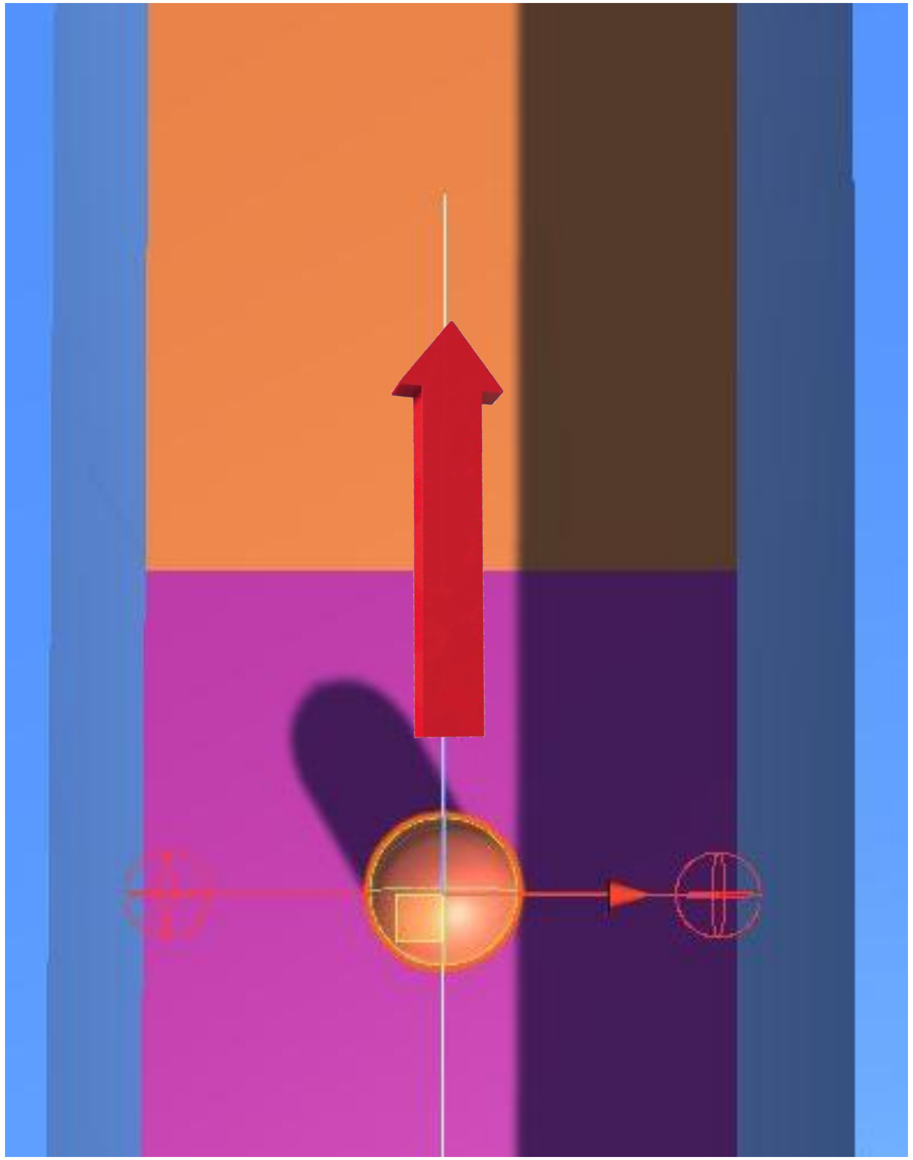
02 Entering for the first time, Agent +3 points, and that background box turns yellow.

03 Entering the second time, the Agent -0.5, and the background cell turns orange.

04 The Agent -1 point the third time, and the background box turns purple.

05

06



3.D. Agent Detail Behavior

Agent when entering a cell will award or punished:

01 When the Agent moves in a specific direction and that side's raycast detects the wall, but the Agent still decides to go in that direction, -1 point.

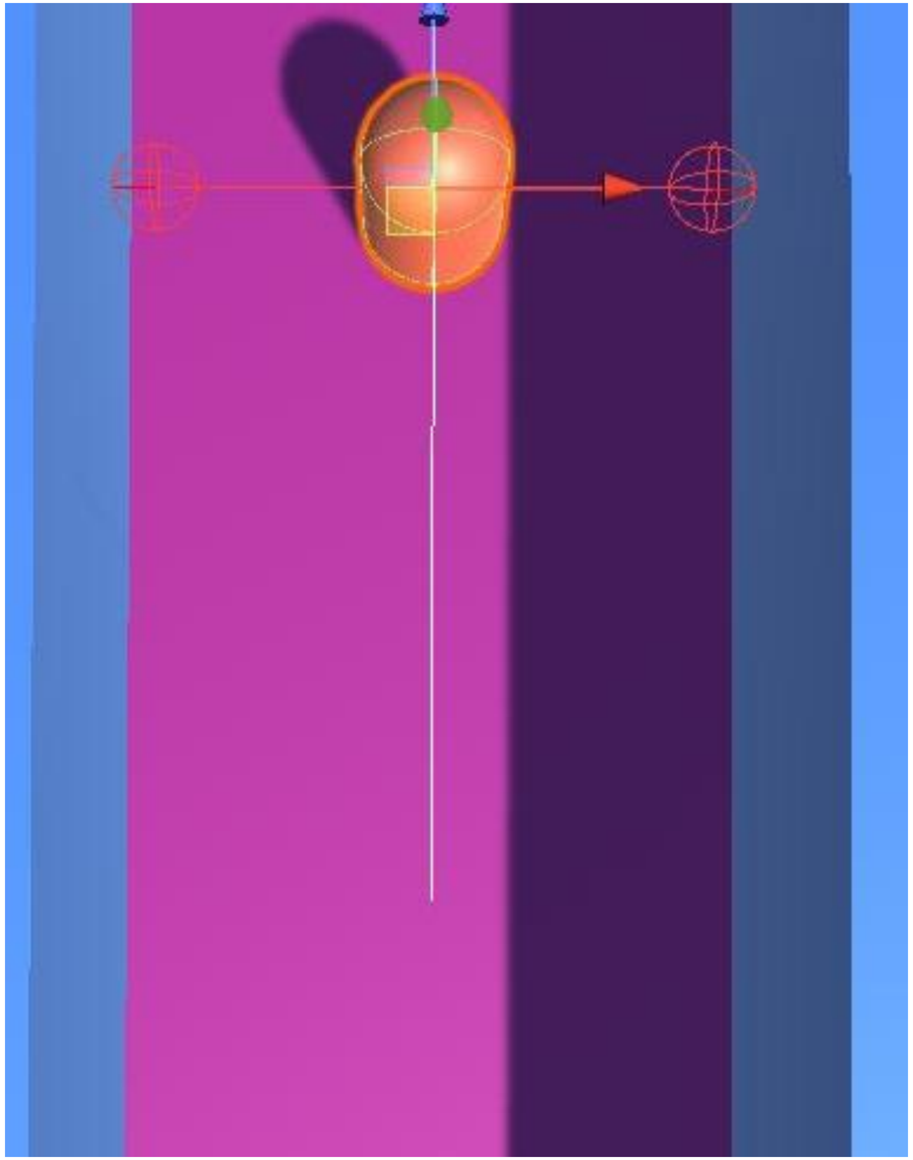
02 Entering for the first time, Agent +3 points, and that background box turns yellow.

03 Entering the second time, the Agent -0.5, and the background cell turns orange.

04 The Agent -1 point the third time, and the background box turns purple.

05 Entering from the fourth time onwards, the Agent has -2 points, and the background is still purple. Purple is the final penalty level when entering.

06



3.D. Agent Detail Behavior

Agent when entering a cell will award or punished:

01

When the Agent moves in a specific direction and that side's raycast detects the wall, but the Agent still decides to go in that direction, -1 point.

02

Entering for the first time, Agent +3 points, and that background box turns yellow.

03

Entering the second time, the Agent -0.5, and the background cell turns orange.

04

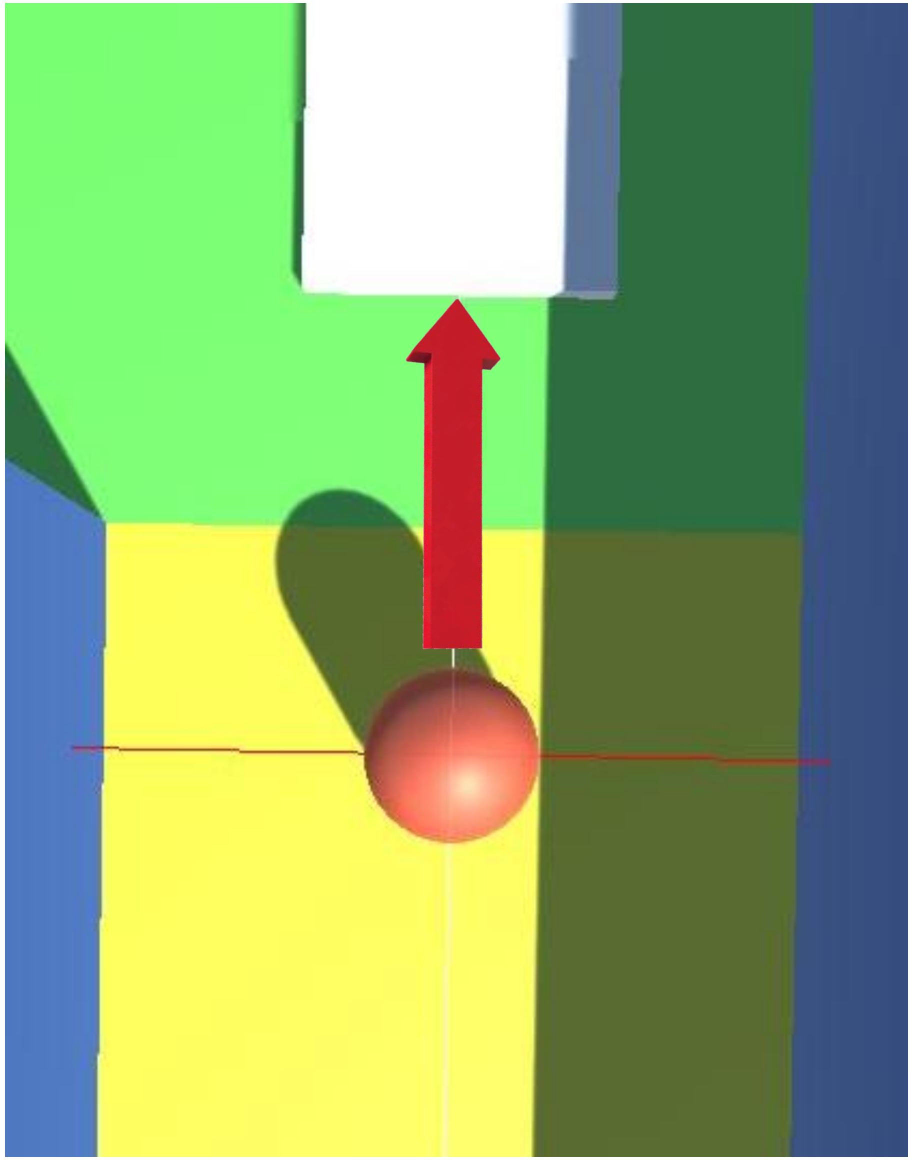
The Agent -1 point the third time, and the background box turns purple.

05

Entering from the fourth time onwards, the Agent has -2 points, and the background is still purple. Purple is the final penalty level when entering.

06

When colliding with the end of the maze, Agent will be +100 points and finish solving the maze.



3.E. Hyperparameters Configuration

Beta

This controls the strength of the entropy regularization so that the agent can explore spaces during training. Beta typically has a value between 1e-4 and 1e-2.

Epsilon

This controls how swiftly the policy can diverge from an older policy. A smaller value has stable updates on the policy. Epsilon typically has a value between 0.1 and 0.3.

Hyperparameters:

- **Batch_size: 128**
- **Buffer_size: 2048**
- **Learning_rate: 0.0003**
- **Beta: 0.005**
- **Epsilon: 0.2**
- **Lambd: 0.95**
- **Num_epoch: 3**

Learning_rate_schedule: linear

Lambd

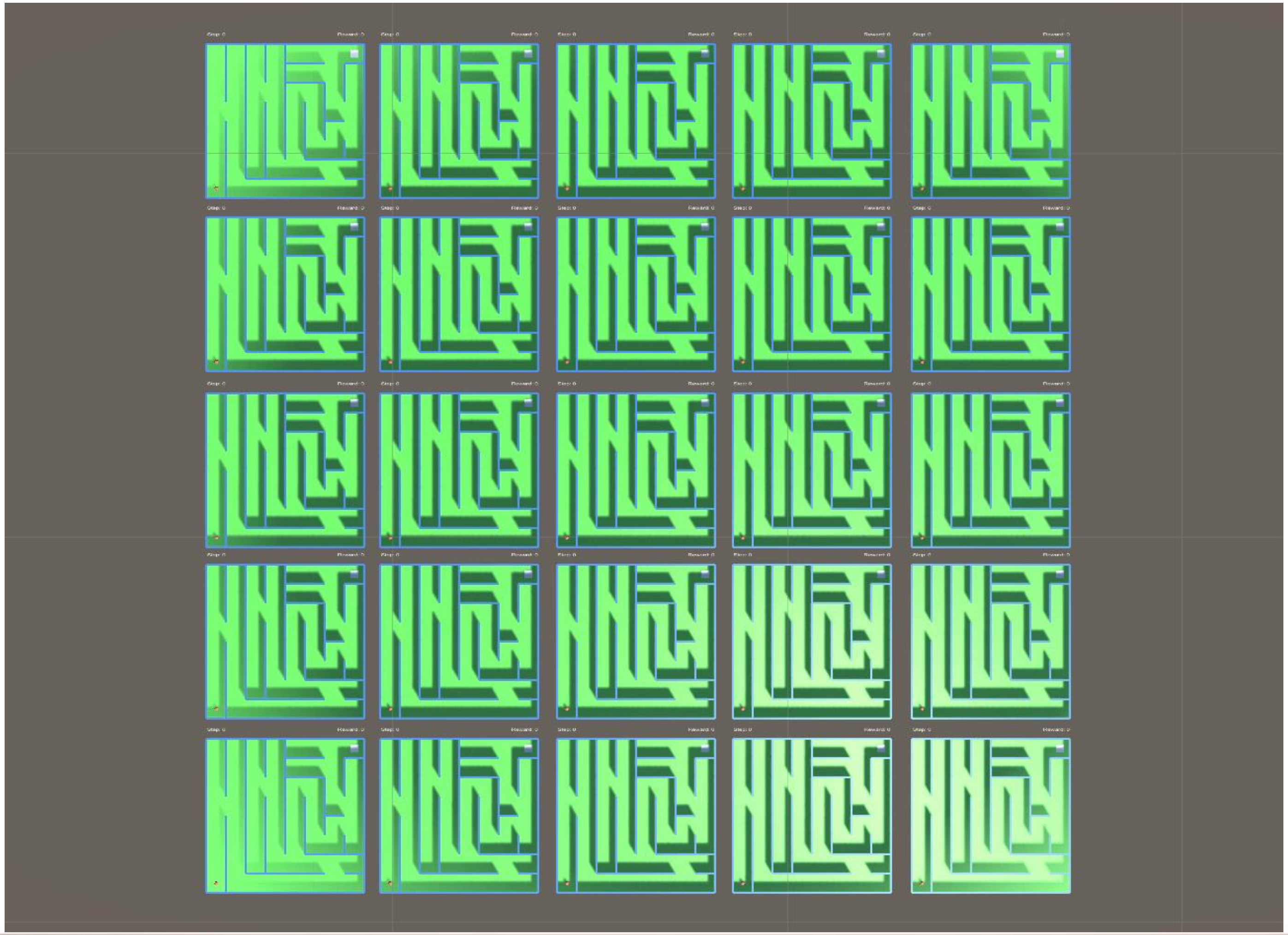
The regularization factor used in calculating GAE. Lambd typically has a value between 0.9 and 0.95.

Num_epoch

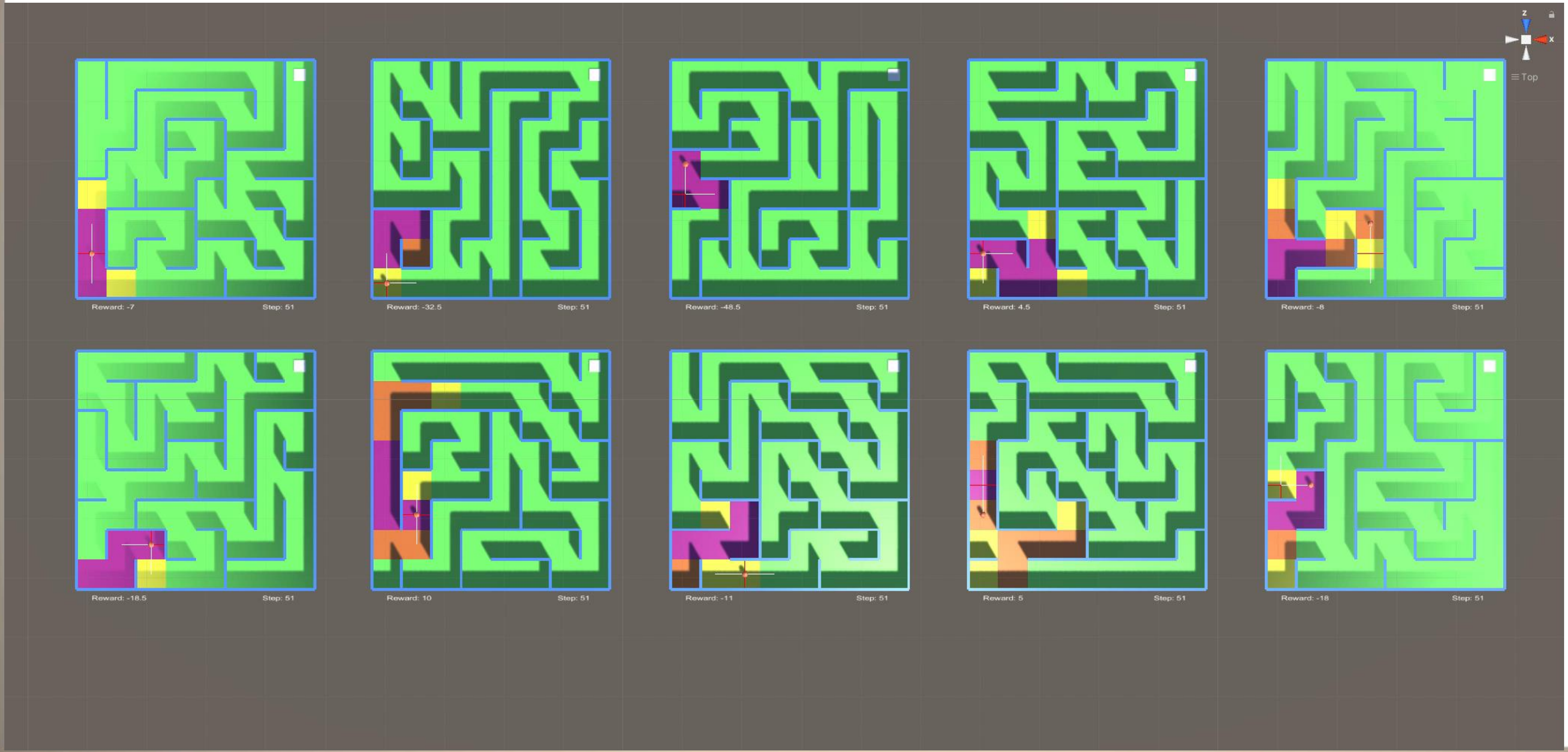
The number of passes made through the buffer before the gradient descent step is applied Num_epcho typically has a value between 3 and 10.

4.A. Agent Training Process

4.A. Agent Training Process

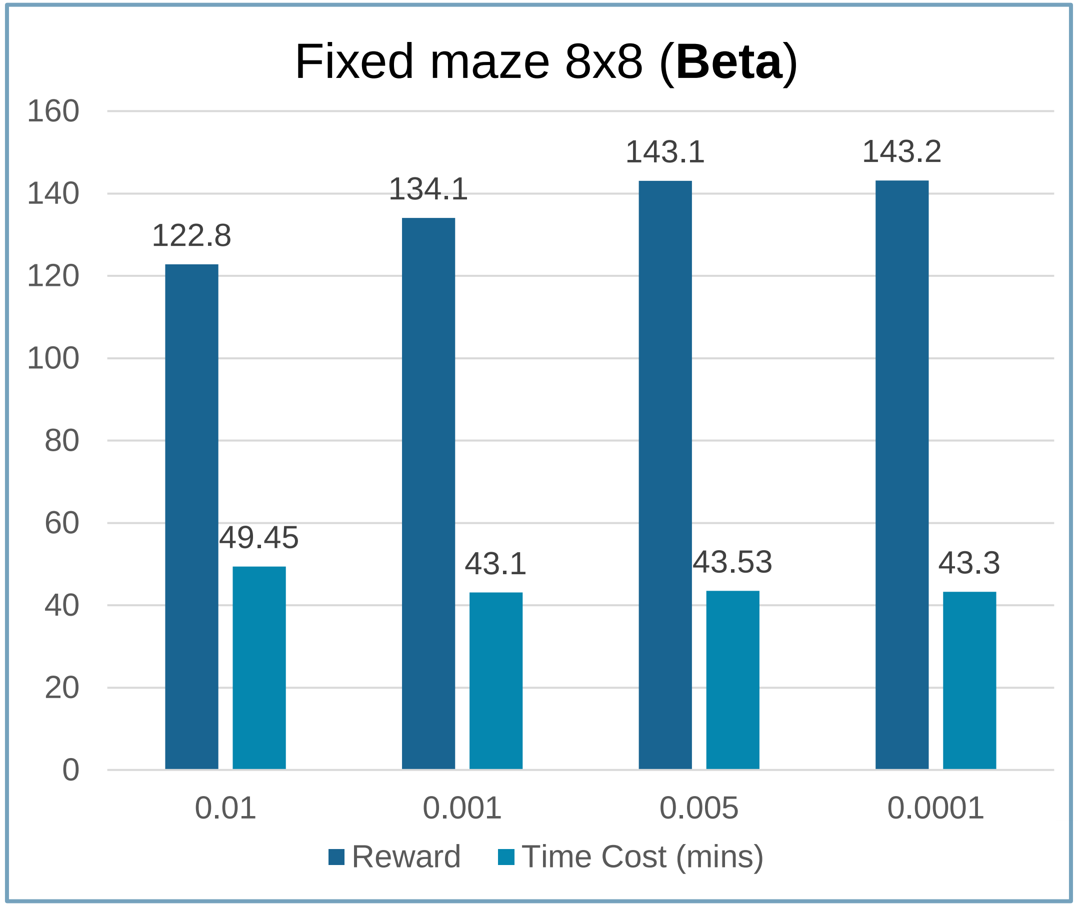


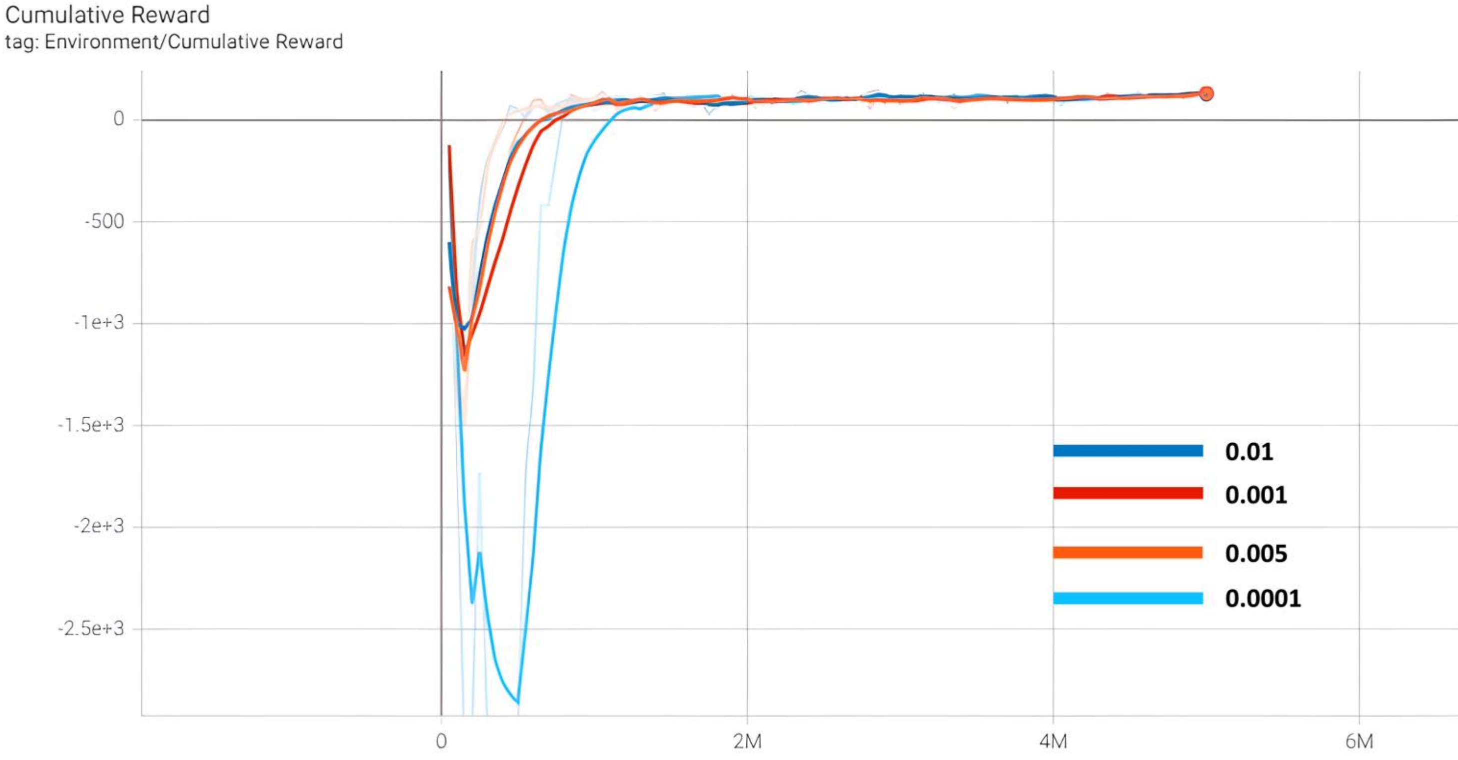
4.A. Agent Training Process



4.B. Results and Analysis

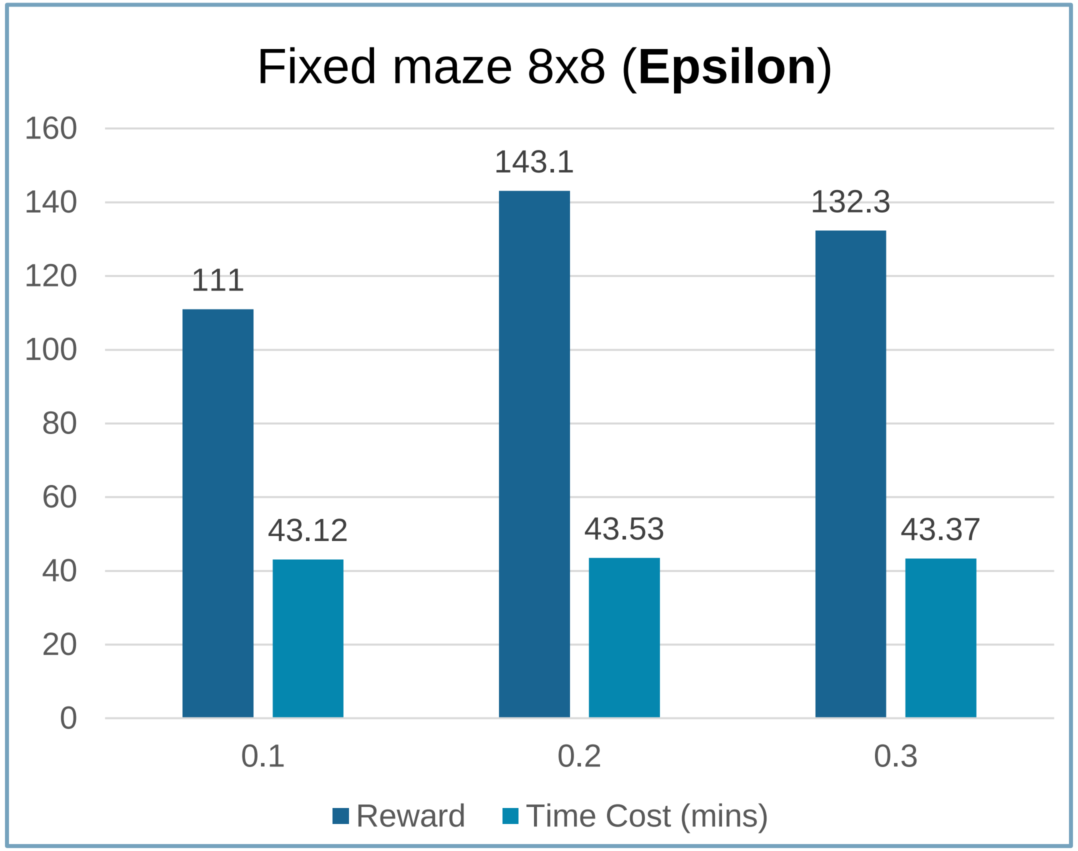
- Compare the results when changing the hyperparameter **Beta** **(Fixed 8x8 Maze)**

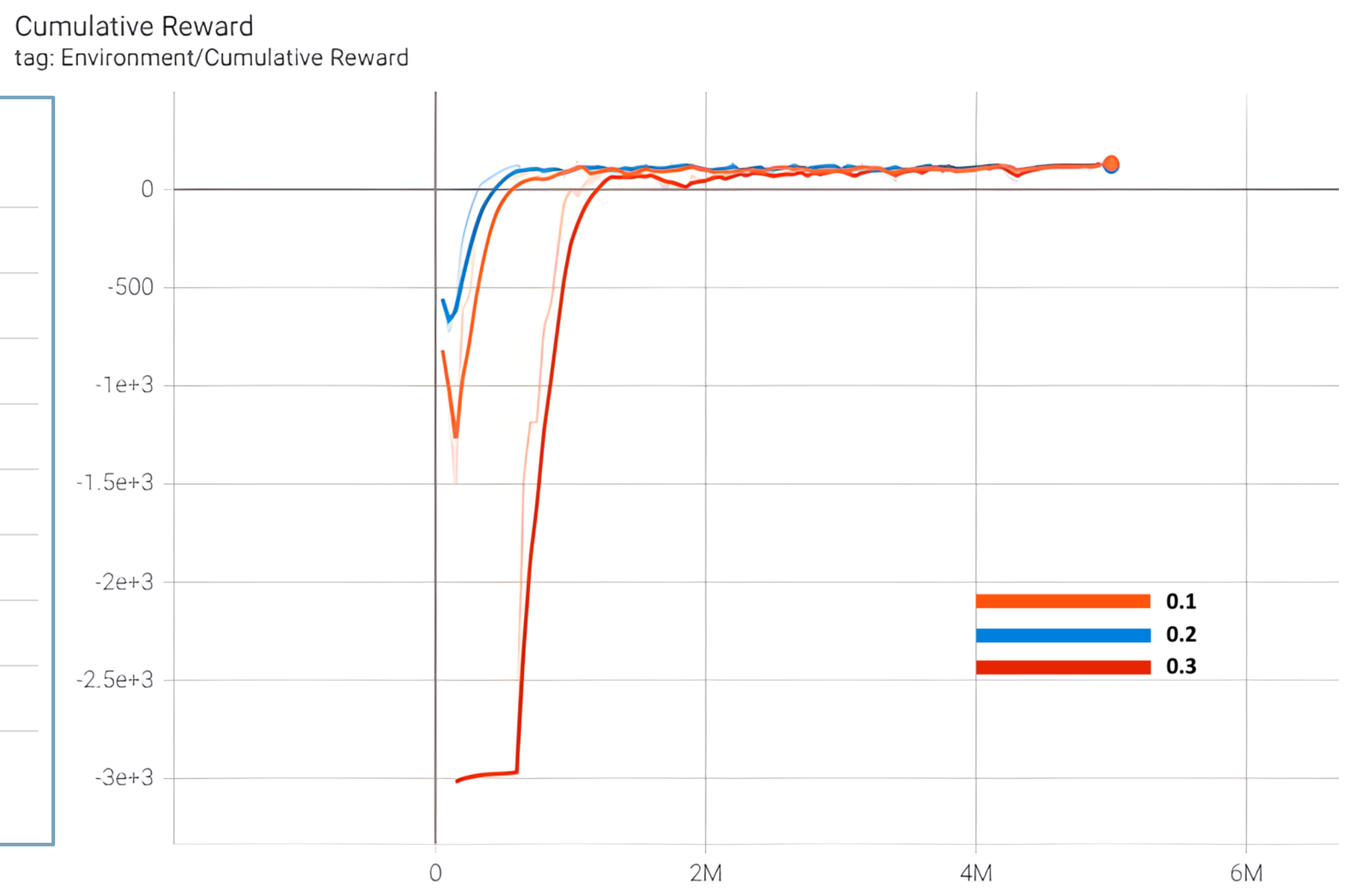




4.B. Results and Analysis

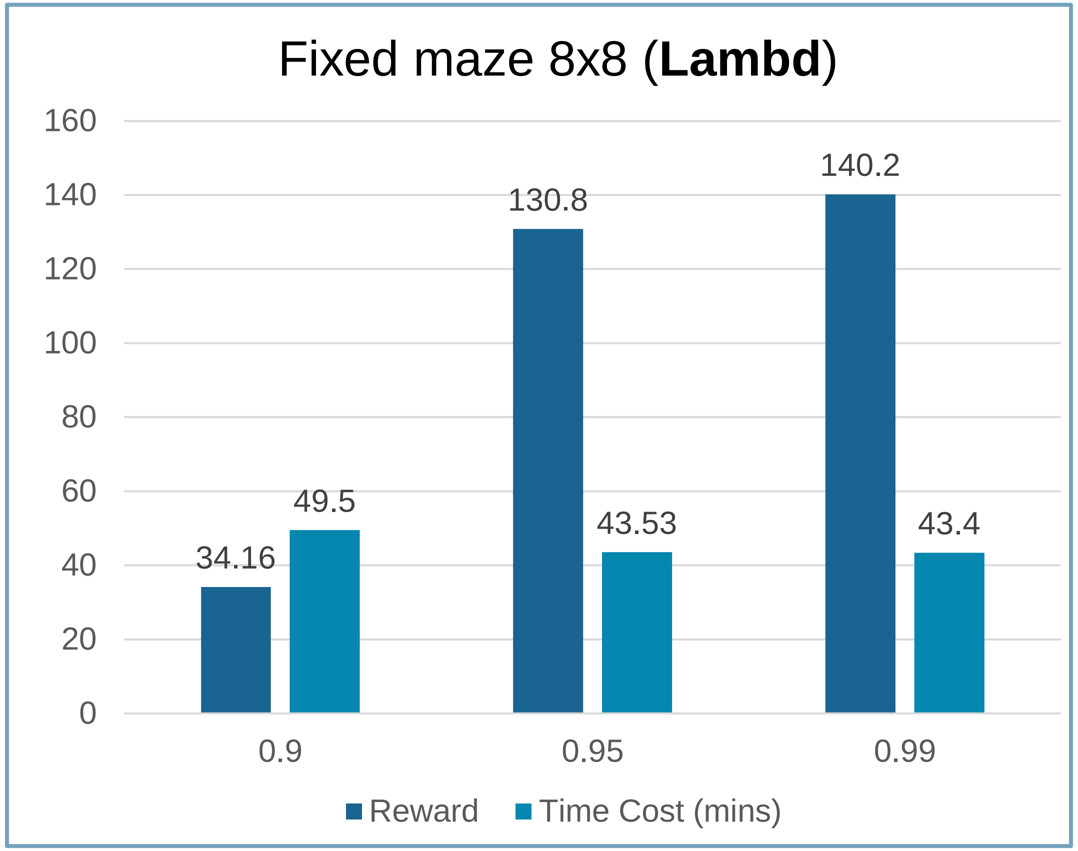
- Compare the results when changing the hyperparameter **Epsilon** **(Fixed 8x8 Maze)**

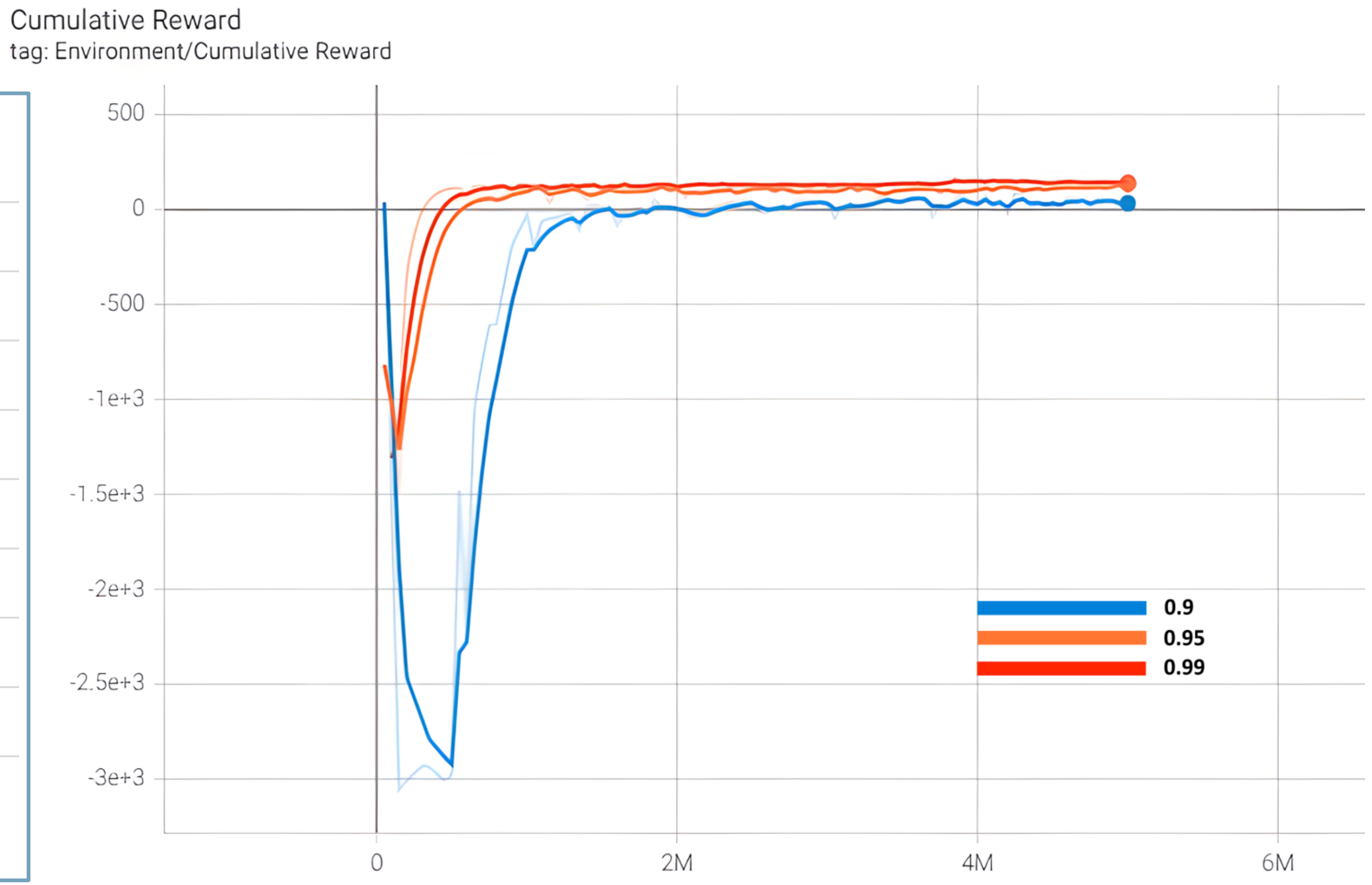




4.B. Results and Analysis

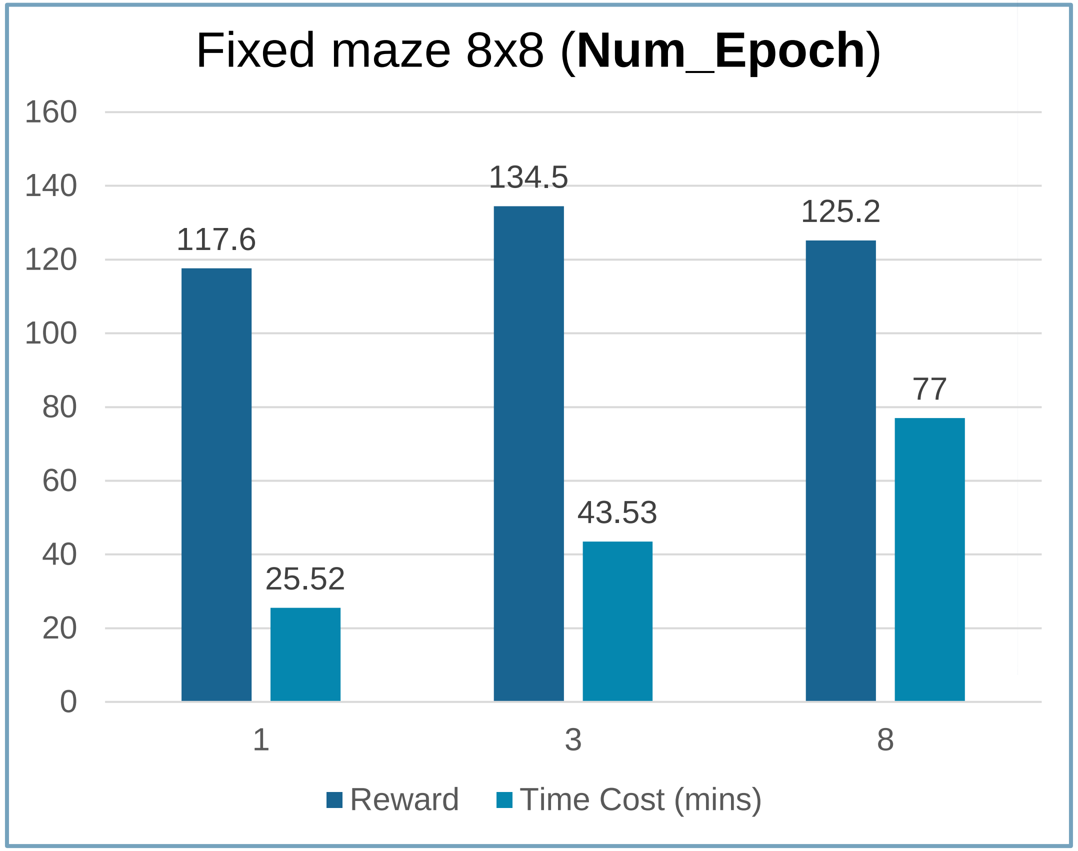
- Compare the results when changing the hyperparameter **Lambd** **(Fixed 8x8 Maze)**

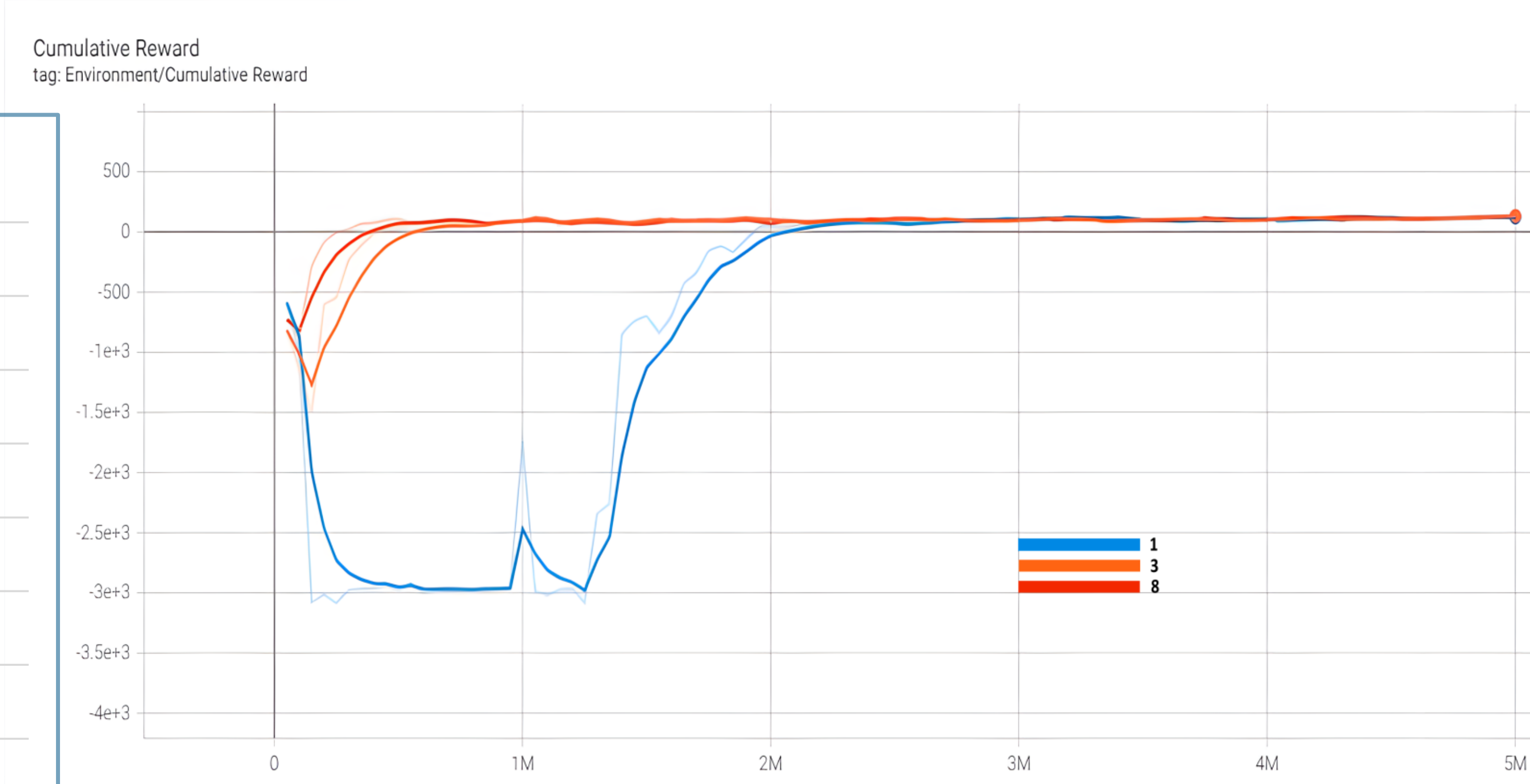




4.B. Results and Analysis

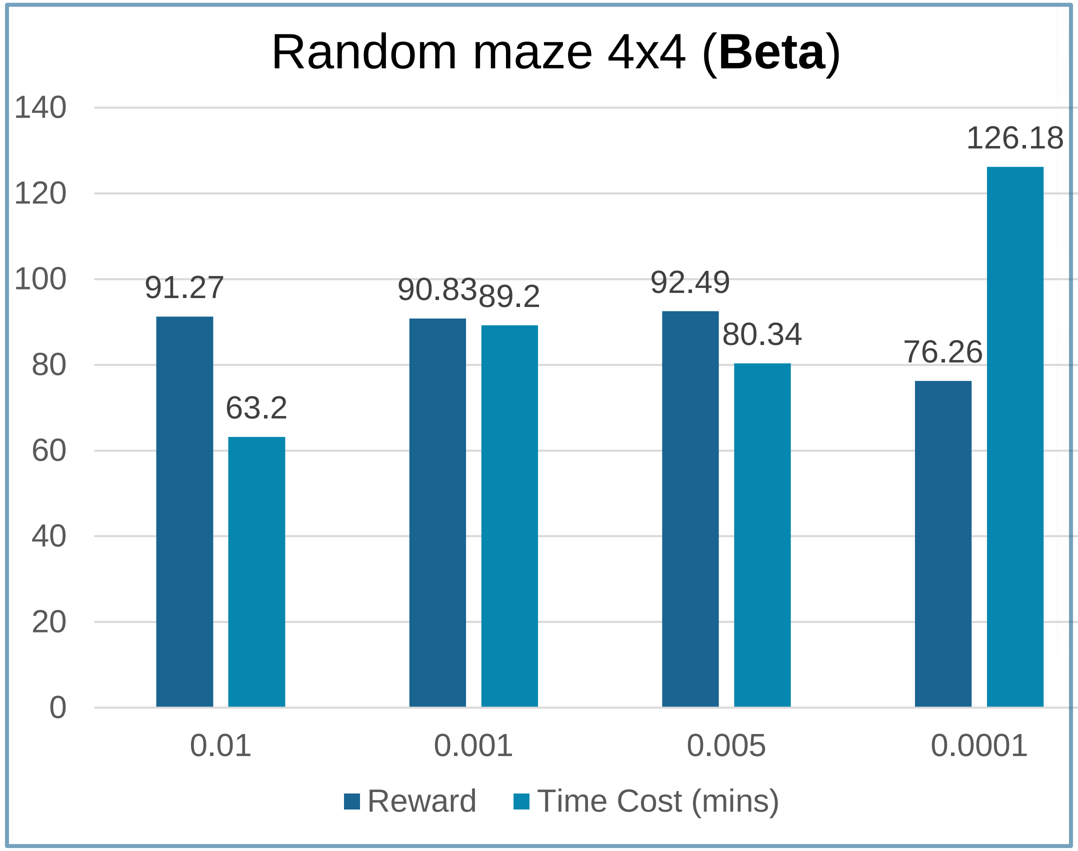
- Compare the results when changing the hyperparameter **Num_epoch** **(Fixed 8x8 Maze)**

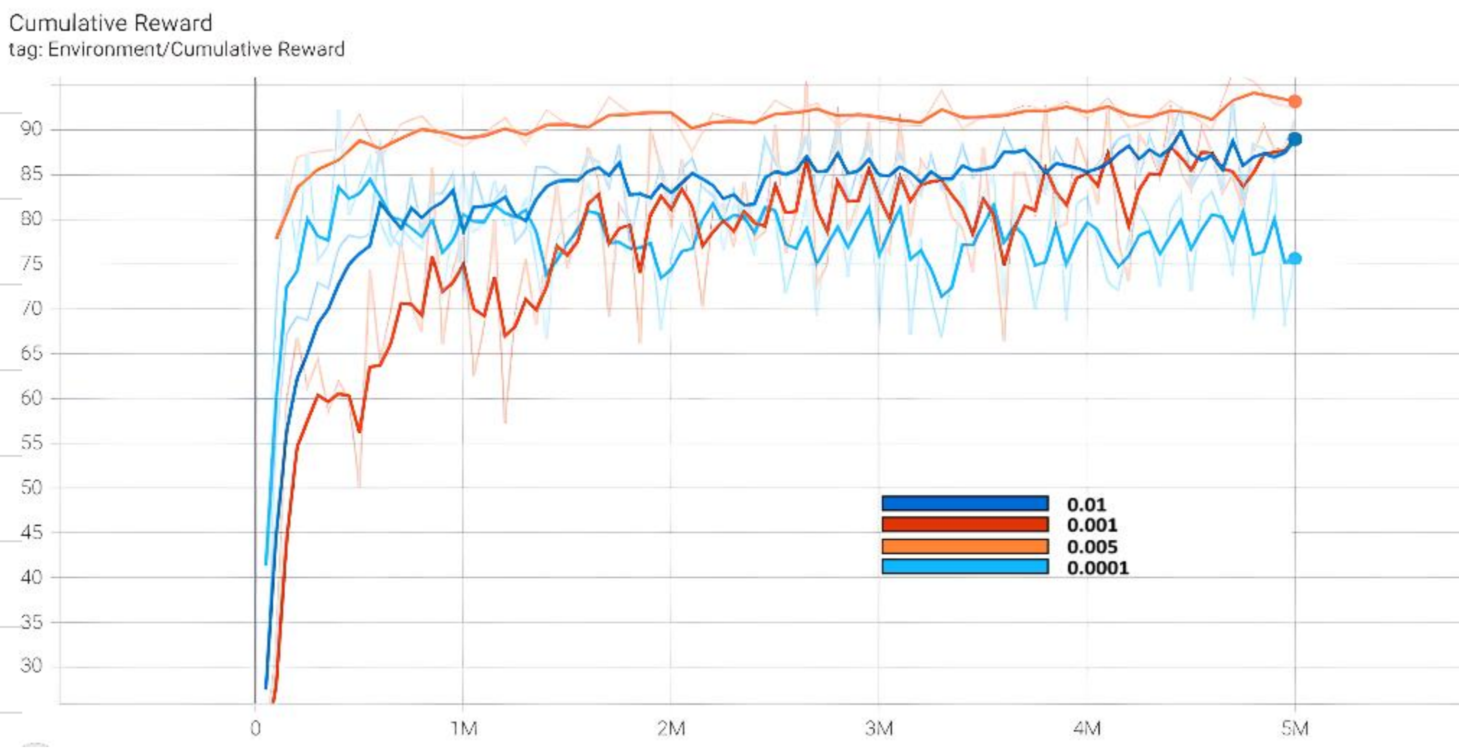




4.B. Results and Analysis

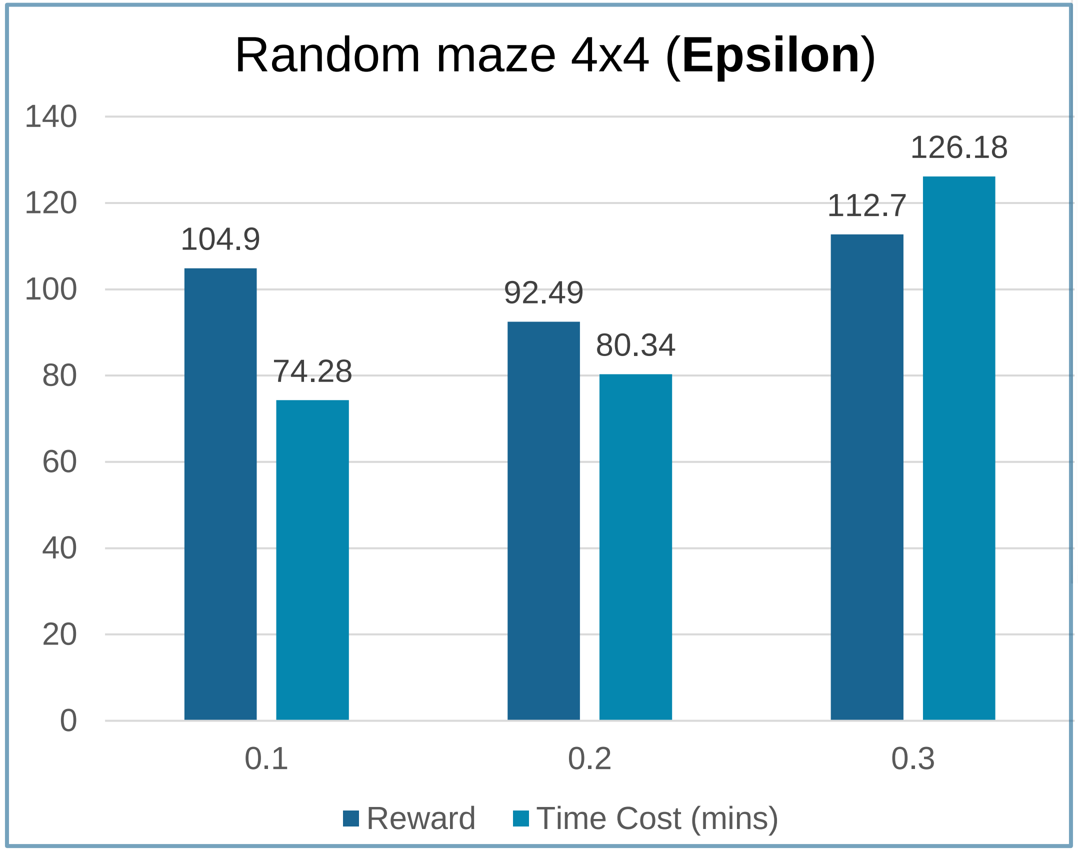
- Compare the results when changing the hyperparameter **Beta** **(Random 4x4 Maze)**

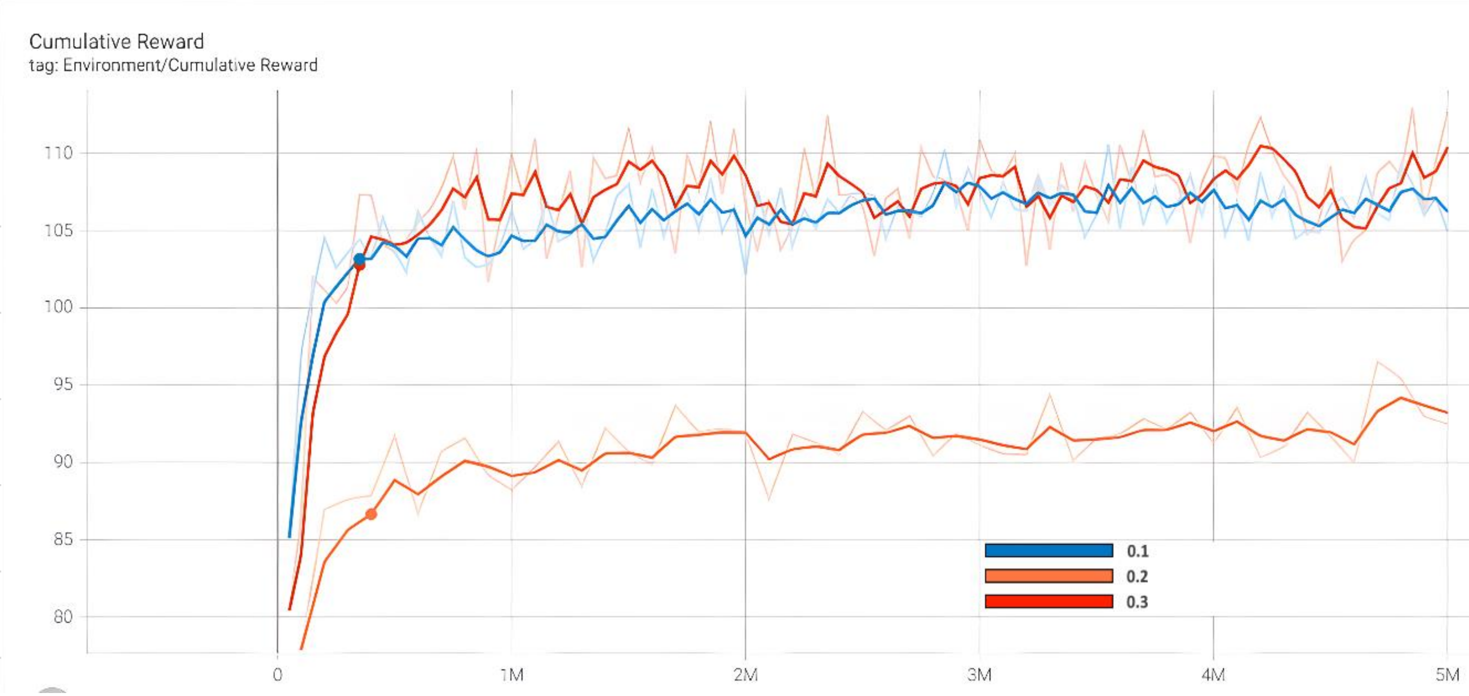




4.B. Results and Analysis

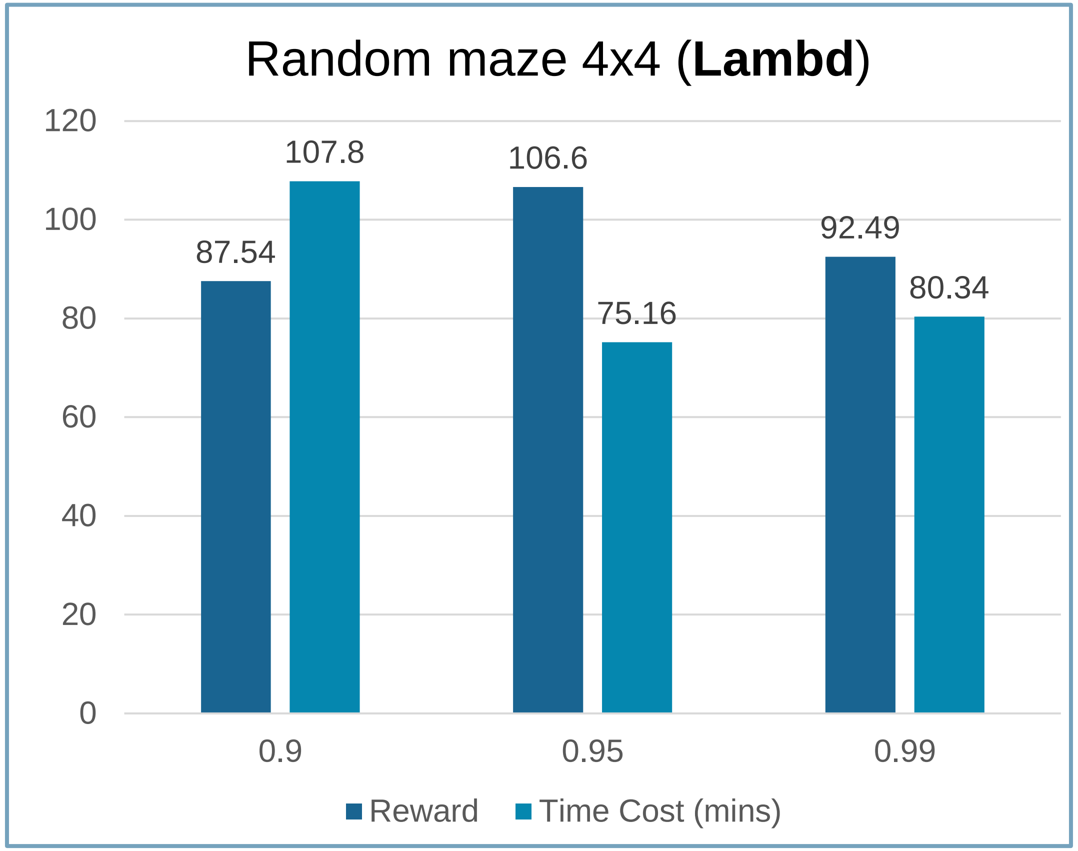
- Compare the results when changing the hyperparameter **Epsilon** **(Random 4x4 Maze)**

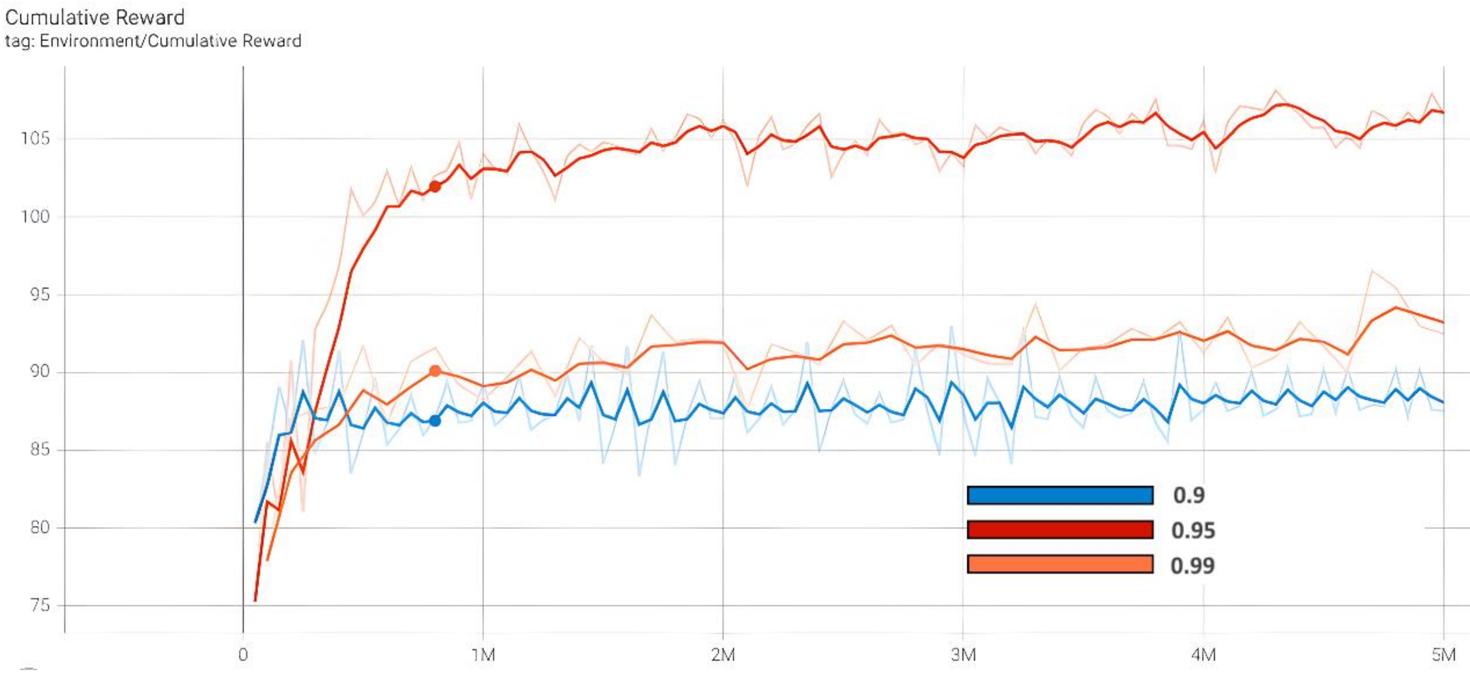




4.B. Results and Analysis

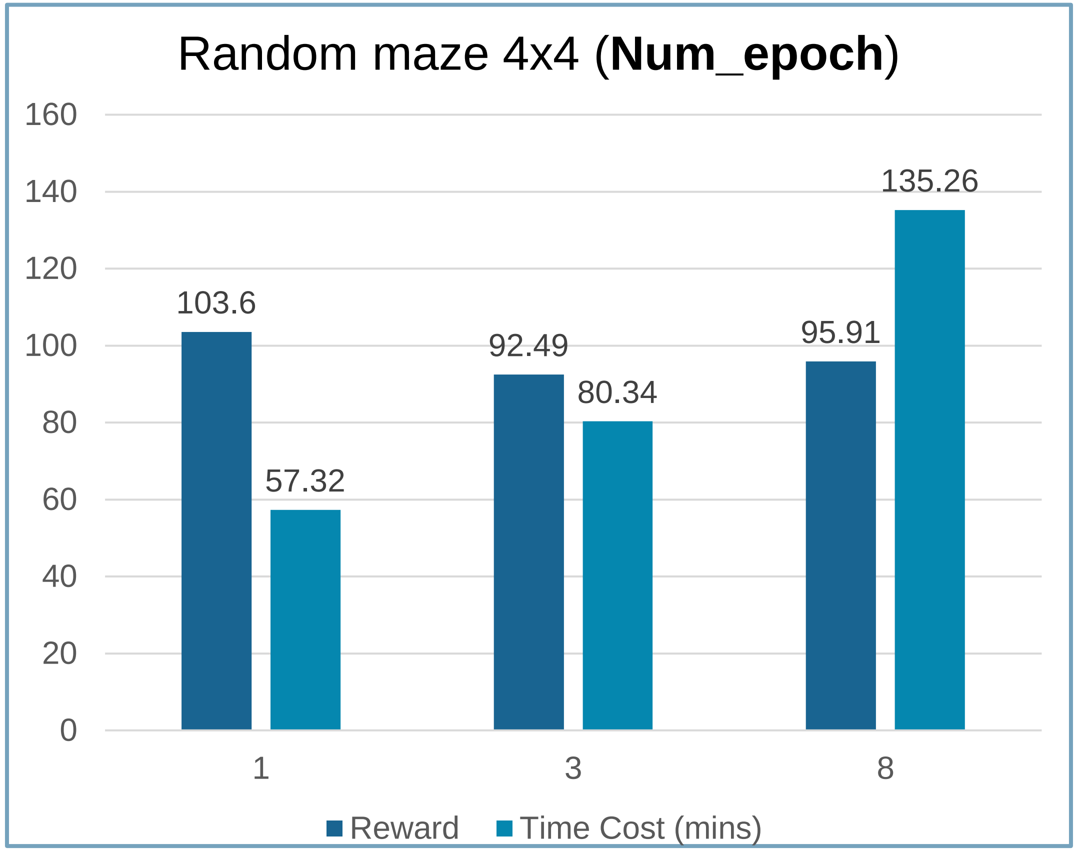
- Compare the results when changing the hyperparameter **Lambd**
  (Random 4x4 Maze)

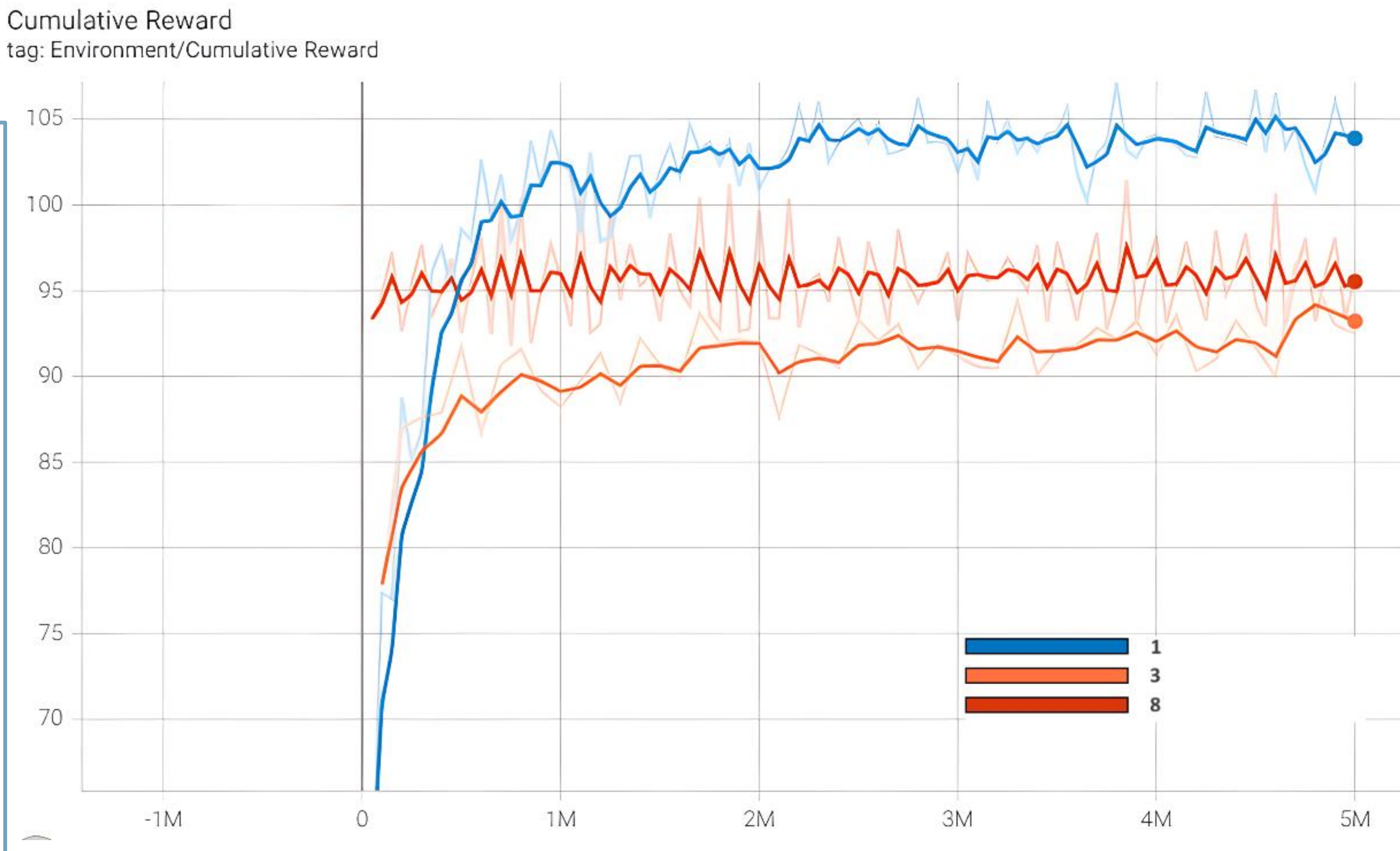




4.B. Results and Analysis

- Compare the results when changing the hyperparameter **Num_epoch** **(Random 4x4 Maze)**





5.A. Conclusion

01. Learning results

This paper gives the tuning for PPO algorithm through hyperparameters Beta, Epsilon, Lambd, and Num_epoch. The results show a clear difference between the training process and the hyperparameters. The change is based on different cases according to the complexity of the maze. Therefore, it is necessary to choose reasonable hyperparameters to set the best training results.

02. Future works

This research also provides a helpful reference for tuning hyperparameters when redeployment PPO algorithm on novel environments in the future.

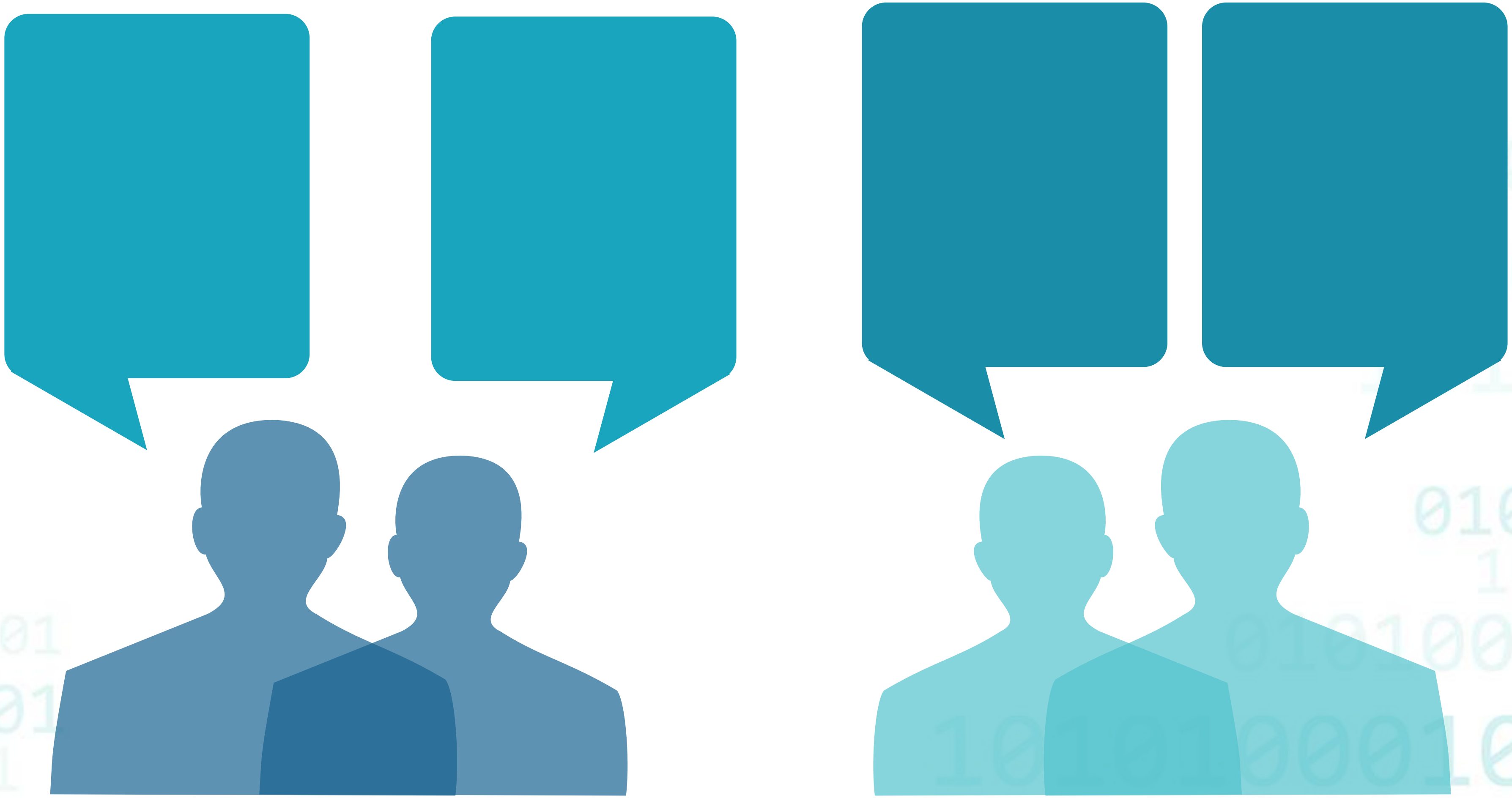
QS STARS RATED FOR EXCELLENCE 2012

FPT Education

FPT UNIVERSITY

5.B. Q&A

Tuning Proximal Policy Optimization Algorithm in Maze Solving with ML-Agents



QS STARS RATED FOR EXCELLENCE 2012

FPT Education

FPT UNIVERSITY

STEAM EDUCATION

This is the end of our project presentation

Thank you for listening