.dit/*
Informatics & Telematics

Development and Evaluation of Reinforcement Learning models for the FOSSBot Open-Source educational robot

George Kazazis, Christos Chronis,
Christos Diou, Iraklis Varlamis

HAROKOPIO UNIVERSITY
GFOSS
OPEN TECHNOLOGIES ALLIANCE

Introduction

- Machine learning for real-world challenges.
- Robotics + simulations = refined algorithms & risk minimization.
- Tasks: Obstacle avoidance and navigation.
- Agent: FOSSBot
- RL Algorithms:
  - Proximal Policy Optimization (PPO)
  - Deep Q Network (DQN)

DIKW pyramid (source)

Related Work (1)
Educational Robotics

- Great potential for tertiary education.
- RL’s adaptability + educational robots = innovative teaching methods.
- Robots’ domain & roles classification – Mubin et al. 2013 [1]
- Technical creativity, Applied knowledge, Interest boost – Ospennikova et al. 2015 [2]
- Open-source education robot – FOSSBot – Chronis and Varlamis 2022 [3]

Related Work (2)
RL, Path Planning and Obstacle Avoidance

- Traditional Path Planning methods: BFS, DFS, Dijkstra’s algorithm.
  - Need for a model of the world-map (Obstacles’ positions)
- RL methods: Through trial-and-error, maximizing cumulative rewards.
  - Dynamic-Complex environments, no model needed, only experience – Sutton and Burto 2018 [4]
- DRL for obstacle avoidance [Kinect RGBD cam] – Tai and Liu 2016 [5]
- Path planner training [Demonstration learning] – Pfeiffer et al. 2017 [6]
- UAV navigation using A2C algorithm – Chronis et al. 2023 [7]

Our Approach: FOSSBot’s terrestrial self-navigation (sensors) + RL power

.dit/* Informatics & Telematics HAROKOPIO UNIVERSITY GFOSS OPEN TECHNOLOGIES ALLIANCE

Technologies used

- Environments:
  - OpenAI Gym
  - CoppeliaSim

CoppeliaSim

- Algorithms:
  - stable-baselines3

- Data Logging:
  - Weights and Biases

Weights & Biases

- Data Visualization:
  - Python’s matplotlib

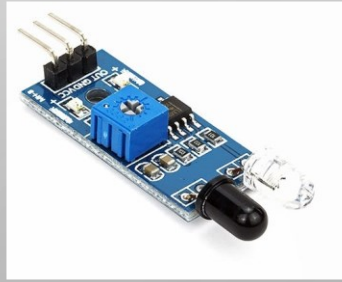
FOSSBot

- Open-source education robot
- 3D printed
- Flexible software stack
- Block-based (Blockly) or Text-based (Coding) programming

- Ultrasonic Distance Sensor (0.02 - 4m)
  - Distance, not bearing
- 3 x Infrared Obstacle Sensors (2-30cm)
  - 1: obstacle - 0: clear
- Inertial Measurement Unit (IMU)
  - Robot’s orientation & position

Software stack (source)

Ultrasonic Distance Sensor (source)



Infrared Obstacle Sensor (source)

RL Algorithms
Preliminaries

DQN – Roderick et al. 2017 [8]

Input: Observation space

Q-Value Estimation

Action Selection (ε-greedy)

Interaction & Reward

Experience Storage (replay buffer)

Weight Updates

Until Convergence

PPO – Schulman et al. 2017 [9]

Policy Initialization

Input: Observation space

Advantage Calculation

Objective Clipping

Gradient Computation

Policy Update

Epoch Iteration

Until Convergence

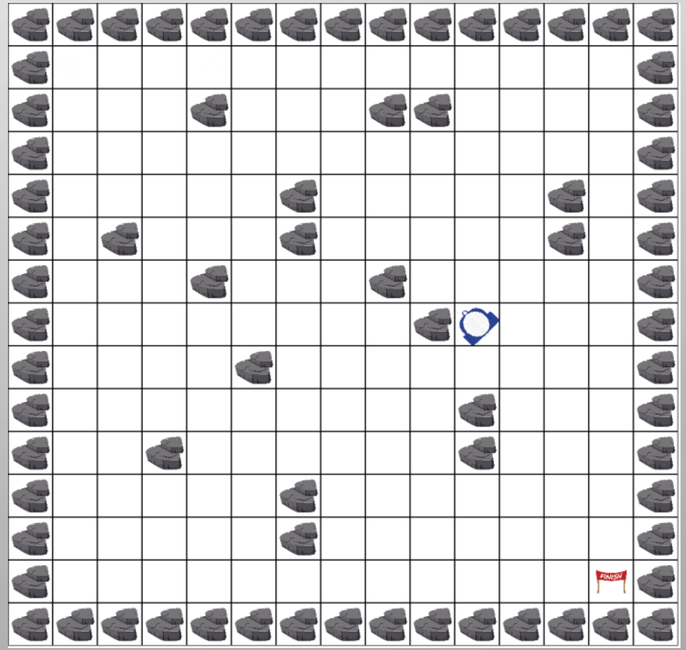
Epoch

Convergence

Experimental Setup (1)
Grid Environment

- Custom environment (OpenAI Gym).
- Action space: 3 discrete actions [Move forward, 45-degree left turn, 45-degree right turn]
- Observation space: i) Agent's angle diff from the target $\Delta\theta$ (in degrees), ii) Euclidean distance d_{eucl}, iii) Total steps (max: 200), iv) 3 IR sensors (implemented) values as a List (size: 3 / 0 or 1)
- Reward Function: $\boxed{\boldsymbol{reward = -d_{eucl}}}$
- Map: List of lists – 0: open path, 1: obstacles
- Default rewards: obstacle collision: -10,

  max steps: -10, target reached: $+1000$
- Visualization: PyGame



Grid Environment – Gym/PyGame

Experimental Setup (2)

Simulation Environment (1)

- Custom environment (CoppeliaSim).
- Action space: 3 discrete actions [Move forward, Forward-left, Forward-right]
- Observation space: i) Agent's angle diff from the target $\Delta\theta$, ii) Euclidean distance d_{eucl},

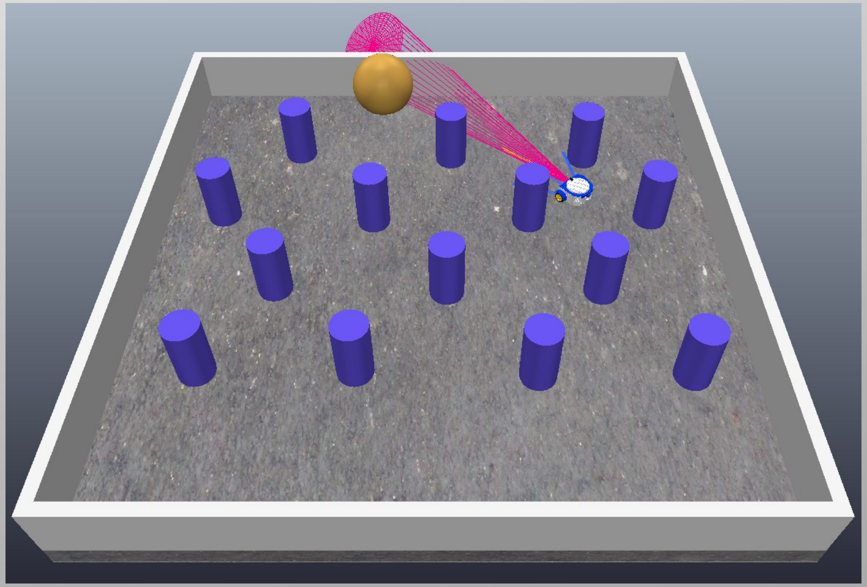
iii) Obstacle distance d_{obs} (by ultrasonic sensor), iv) 3 IR sensor binary values s_l, s_c, s_r

- Reward Function:

$$\boldsymbol{reward} = \boldsymbol{w_{obs}} \cdot \left[\boldsymbol{0.5} \cdot \left(\boldsymbol{1} - \frac{\boldsymbol{l_{arc}}}{\boldsymbol{180} \cdot \boldsymbol{d_{target}}} \right) + \boldsymbol{0.5} \cdot \left(\boldsymbol{1} - \frac{\boldsymbol{d_{target}}}{\boldsymbol{d_{max}}} \right) \right]$$

- Default rewards: obstacle collision (-100),

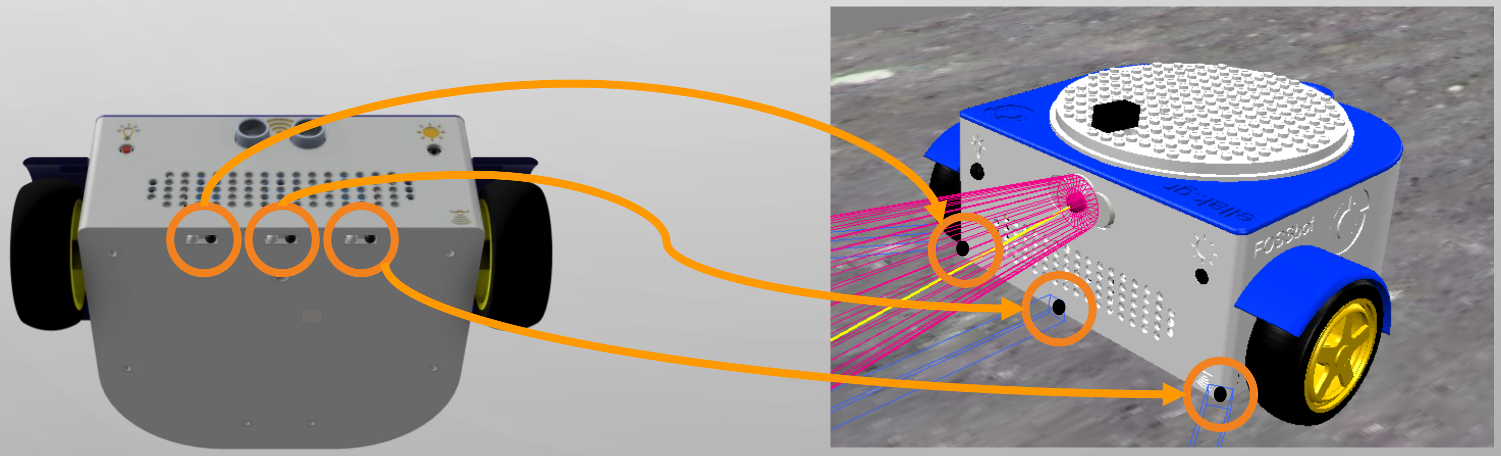
target reached $(+1000)$



Simulation Environment - CoppeliaSim

Experimental Setup (2)
Simulation Environment (2)

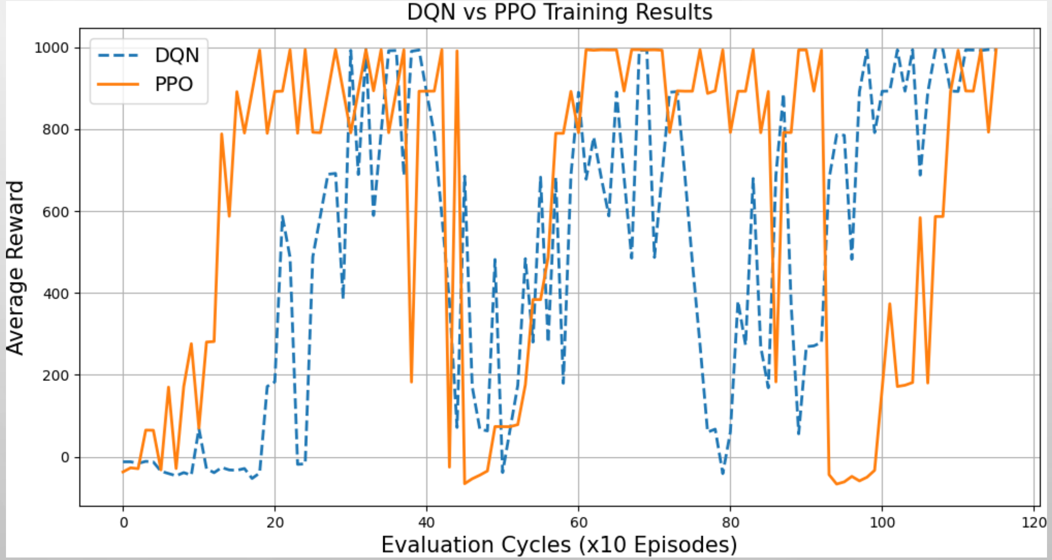
- Current FOSSBot structure – Difficulties
- Ultrasonic sensor gives distance, but not bearing.
- IR obstacle sensors come to rescue(!)
- If they detect something, we get some info about the target's bearing.



Suggested IR sensors position modification

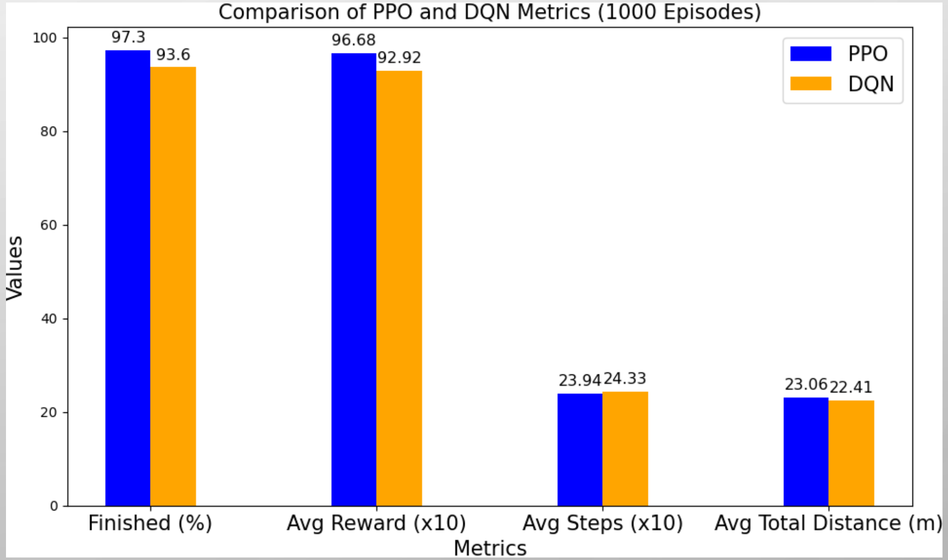
Results
Grid

Training



Evaluation Cycle: 10 evaluation episodes

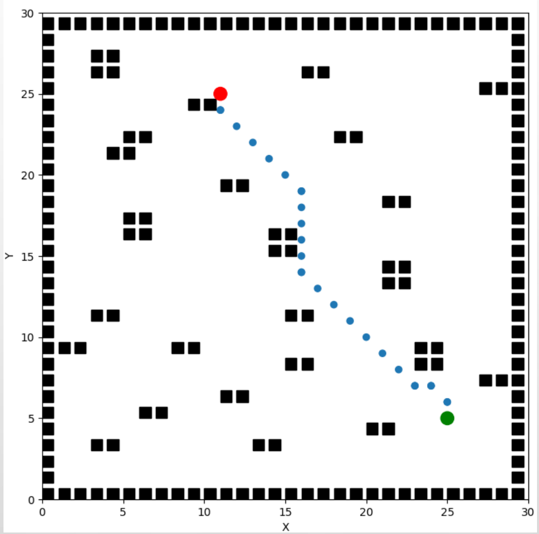
Evaluation

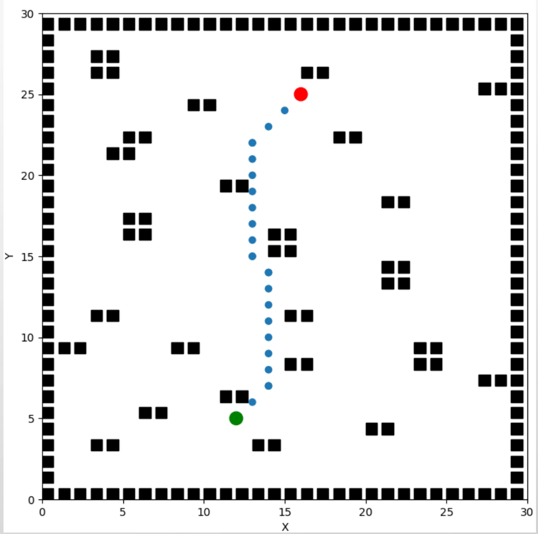


Final Evaluation: 1,000 evaluation episodes

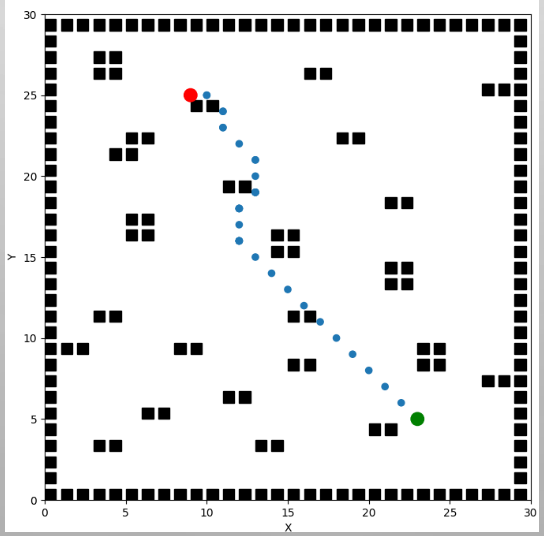
Trajectories
Grid

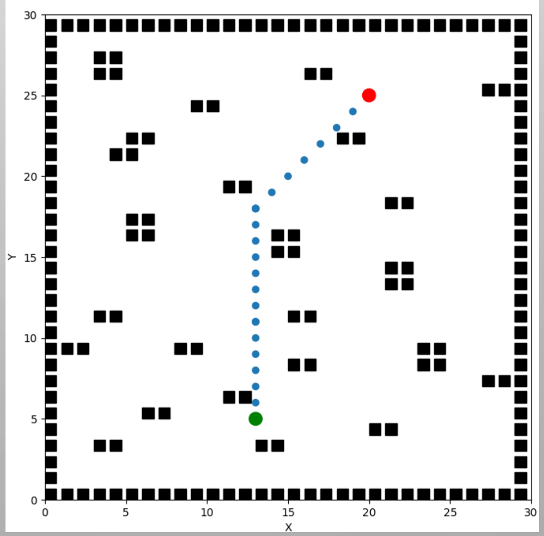
PPO-





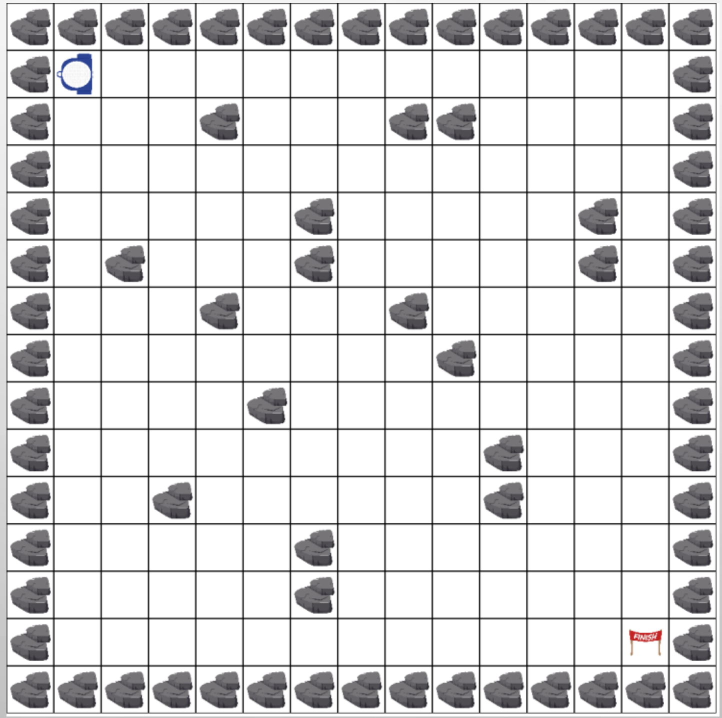
DQN-



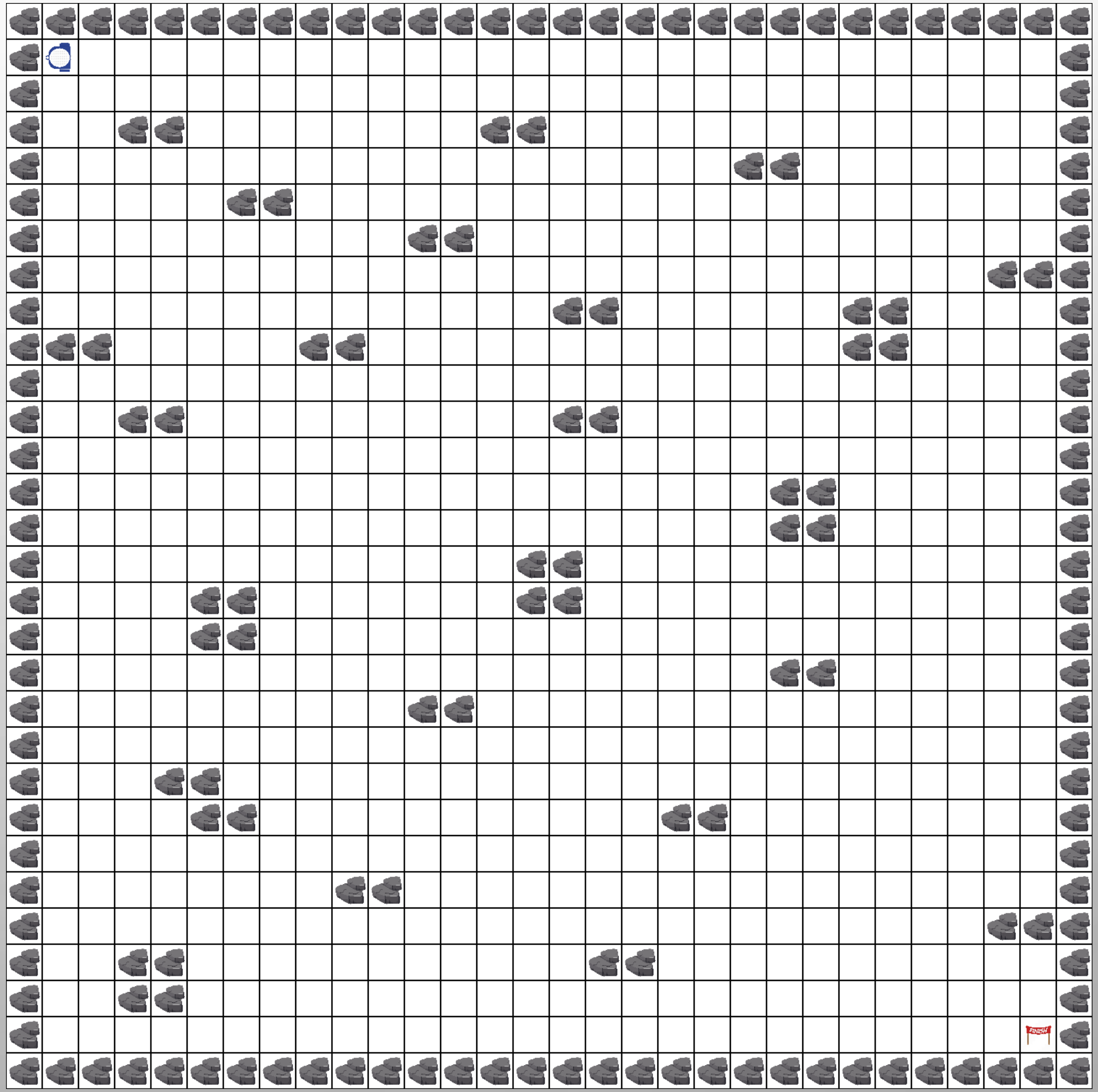


Solutions

Grid



15x15 - DQN

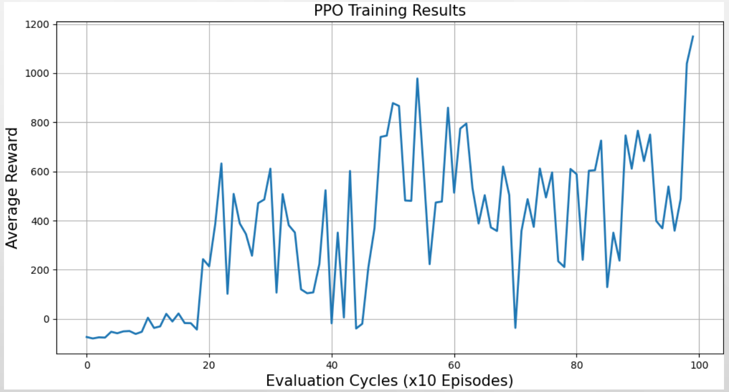


30x30 - PPO

Results

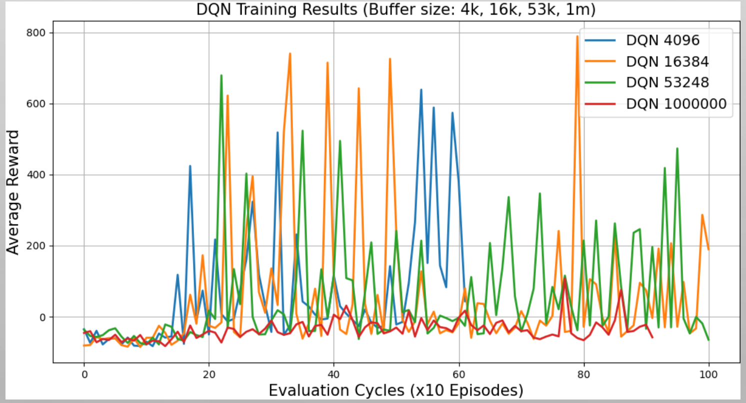
Simulation

Training



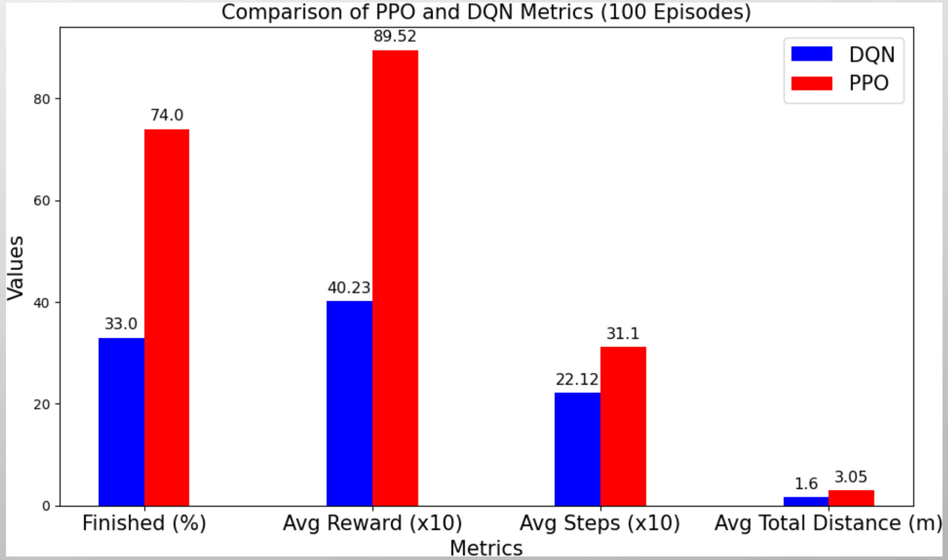
-PPO

DQN-



Evaluation Cycle: 10 evaluation episodes

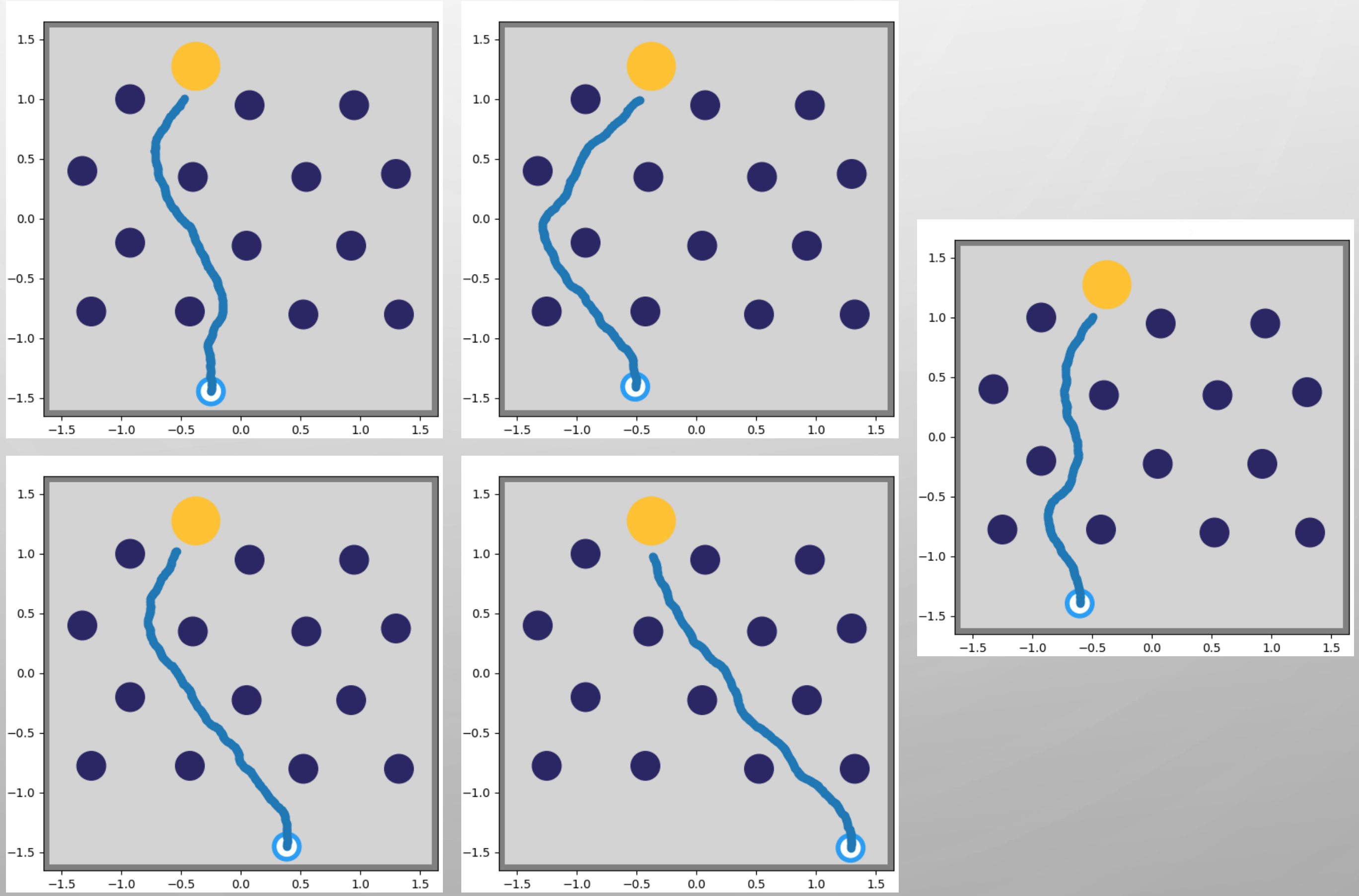
Evaluation



Final Evaluation: 100 evaluation episodes

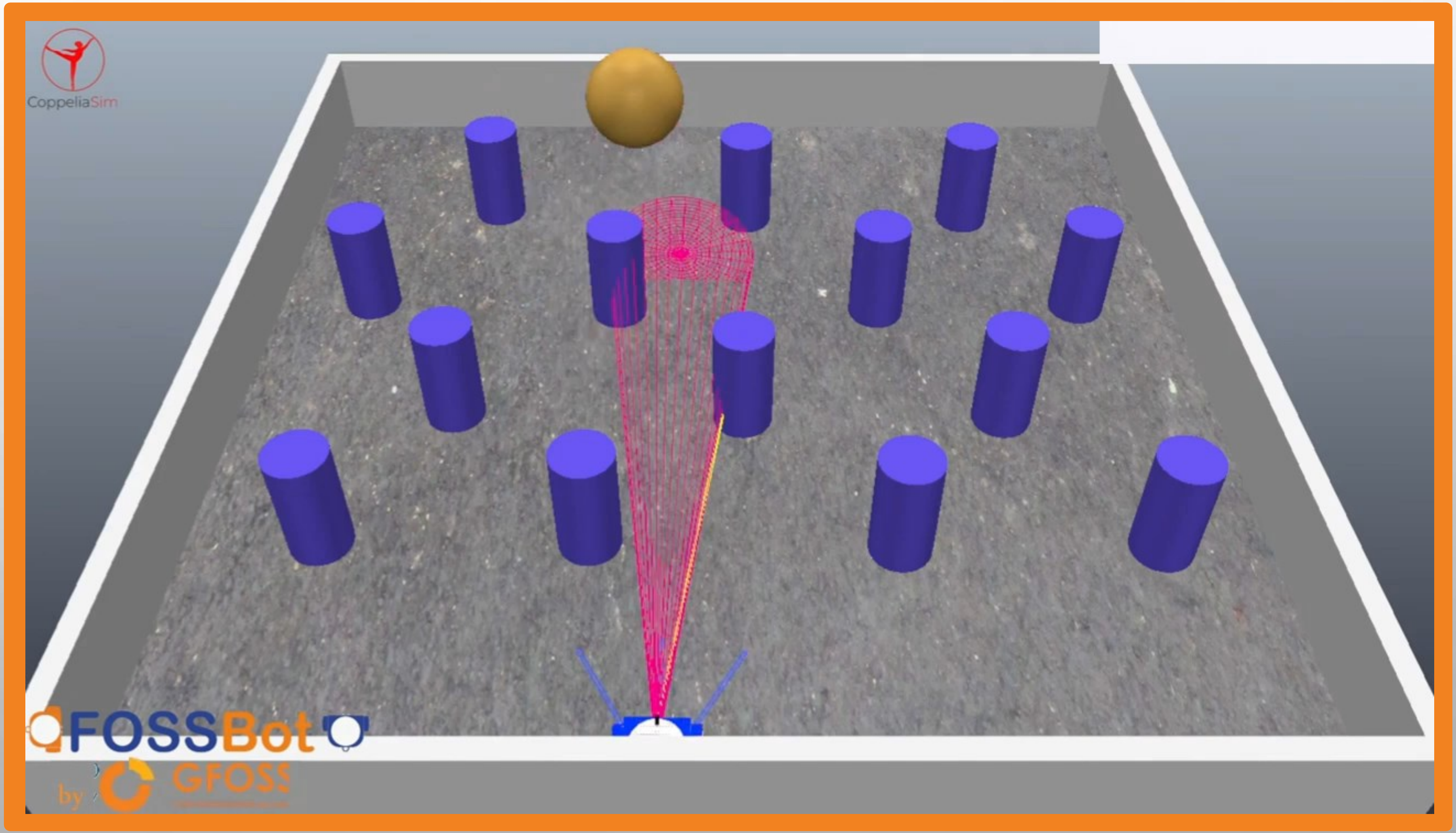
Trajectories
Simulation

PPO-



Solutions

Simulation



Conclusions

- PPO & DQN excel in simple grid environments.
- PPO outperforms DQN in complex simulations, and learns rapidly and effectively.
- FOSSBot is now autonomous in path planning.
- Future work:
  - Develop path-planning library.
  - Apply successful strategies to real FOSSBot.
  - Optimize RL algorithms.

Acknowledgments

- The authors would like to thank the Greek Open Technologies Alliance GFOSS OPEN TECHNOLOGIES ALLIANCE for supporting and funding this article.

- FOSSBot evolution through the GSoC contest
- Buying & Assembling the first 100 robots + sending them to Greek schools
- FOSSBot in academic assignments ➡ Python programming in practice

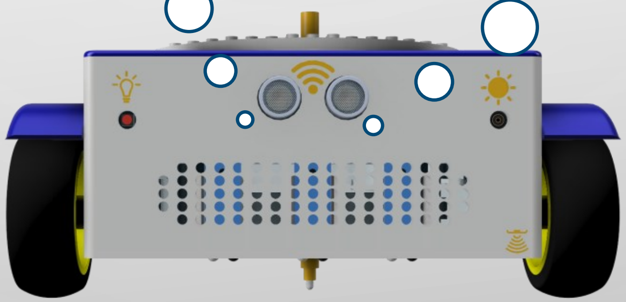
References

1. Omar Mubin, Catherine Stevens, Suleman Shahid, Abdullah Mahmud, and Jian-Jie Dong. 2013. A review of the applicability of robots in education. Technology for Education and Learning 1 (06 2013).
2. Elena Ospennikova, Michael Ershov, and Ivan Iljin. 2015. Educational Robotics as an Inovative Educational Technology. Procedia - Social and Behavioral Sciences 214 (12 2015), 18–26.
3. Christos Chronis and Iraklis Varlamis. 2022. FOSSBot: An Open Source and Open Design Educational Robot. Electronics 11, 16 (2022), 2606.
4. Richard S Sutton and Andrew G Barto. 2018. Reinforcement learning: An introduction. MIT press.
5. Lei Tai and Ming Liu. 2016. Towards cognitive exploration through deep reinforcement learning for mobile robots. arXiv preprint arXiv:1610.01733 (2016).
6. Mark Pfeiffer, Michael Schaeuble, Juan Nieto, Roland Siegwart, and Cesar Cadena. 2017. From perception to decision: A data-driven approach to end-to-end motion planning for autonomous ground robots. In 2017 IEEE International Conference on Robotics and Automation (ICRA). IEEE.
7. Christos Chronis, Georgios Anagnostopoulos, Elena Politi, Antonios Garyfallou, Iraklis Varlamis, and George Dimitrakopoulos. 2023. Path planning of autonomous UAVs using reinforcement learning. In Journal of Physics: Conference Series, Vol. 2526. IOP Publishing, 012088.
8. Melrose Roderick, James MacGlashan, and Stefanie Tellex. 2017. Implementing the Deep Q-Network. arXiv:1711.07478 [cs.LG]
9. John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal Policy Optimization Algorithms. arXiv:1707.06347 [cs.LG]

Thank you!

Finally! I know how to avoid obstacles!

Dear Lord T-800! Whatever!
Any questions?



Don't rest on your laurels buddy!
It's only the beginning!

.dit/* Informatics & Telematics

HAROKOPIO UNIVERSITY

GFOSS OPEN TECHNOLOGIES ALLIANCE