REINFORCEMENT LEARNING BASED AUTOMATED PATH PLANNING IN GARDEN ENVIRONMENT USING DEPTH - 'RAPiG-D'

- S SATHIYA MURTHI - 2018504604
- PRANAV BALAKRISHNAN - 2018504581
- C ROSHAN ABRAHAM - 2018504591

BE ELECTRONICS AND COMMUNICATION ENGINEERING

GUIDED BY:
DR. V. SATHIESH KUMAR,
ASSISTANT PROFESSOR,
DEPARTMENT OF ELECTRONICS ENGINEERING
MIT CAMPUS, ANNA UNIVERSITY

TABLE OF CONTENTS

- Introduction
- Motivation
- Objectives
- Literature Survey
- Methodology and Flowchart
- Results and Discussions
- Conclusion
- References

INTRODUCTION

- Over the years, robots have been implemented in a wide range of applications and environments.
- In turn, this has led to dealing with a host of environments that are increasingly dynamic and unknown.
- Path planning with reinforcement learning would aid the automation of multiple tasks with minimal human intervention.

MOTIVATION

- A learning based approach, can be used to navigate and map an unknown environment and trace the best possible path between any two points.

- Stereo camera based depth mapping would be better suited to identify any type of obstacle.

- Current methods are limited by their inability to adapt to new environments.

OBJECTIVES

- To perform hardware implementation of reinforcement learning based path planning to generate a map of an unknown environment by performing depth estimation based obstacle detection and analyze its performance.

- To estimate the best possible path to reach a given destination.

LITERATURE SURVEY

AUTHOR NAME	TITLE	YEAR OF PUBLICATION	JOURNAL	DESCRIPTION
Valentyn N. Sichkar	Reinforcement Learning Algorithms in Global Path Planning for Mobile Robot	2019	International Conference on Industrial Engineering, Applications and Manufacturing (ICIEAM)	• This paper implemented two Reinforcement learning algorithms – Q-learning and SARSA in a software simulated virtual environment. • It analyzed their performance. • The algorithm learnt the optimal path to get maximum payoff and reached goal while avoiding obstacles.

LITERATURE SURVEY

AUTHOR NAME	TITLE	YEAR OF PUBLICATION	JOURNAL	DESCRIPTION
Martin Gromniak, Jonas Stenzel	Deep Reinforcement Learning for Mobile Robot Navigation	2019	Asia-Pacific Conference on Intelligent Robot Systems (ACIRS)	• Developed training procedure, set of actions available, suitable state representation, and a reward function. • Evaluated using a simulated real-time environment • The experimental evaluation showed that DRL can be applied successfully to robot navigation.

LITERATURE SURVEY

AUTHOR NAME	TITLE	YEAR OF PUBLICATION	JOURNAL	DESCRIPTION
Jing Xin, Huan Zhao, Ding Liu	Application of Deep Reinforcement Learning in Mobile Robot Path Planning	2019	IEEE robotics and automation letters	• A Deep Q-network (DQN) is designed and trained to approximate the mobile robot state-action value function and Q value corresponding to each possible mobile robot action is determined by the well trained DQN. • The current optimal mobile robot action is selected by the action selection strategy to the goal point while avoiding obstacles ultimately

METHODOLOGY - SARSA

- **Reinforcement learning** is a machine learning training method based on rewarding desired behaviors and/or punishing undesired ones. In general, a reinforcement learning agent is able to perceive and interpret its environment, take actions and learn through trial and error.

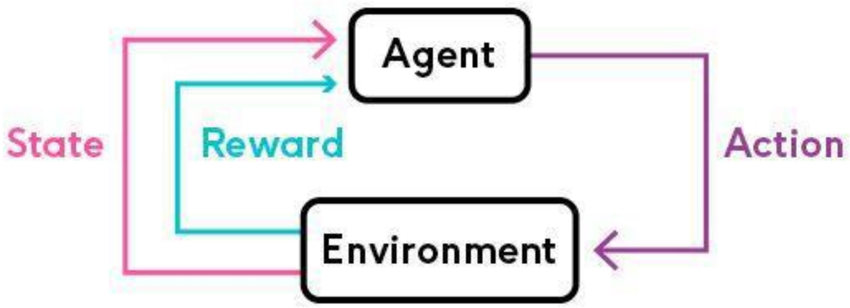


Figure 1: SARSA Illustration
Reference: https://medium.com/@vishnuvijayanpv/what-is-reinforcement-learning-e5dc827c8564

- The SARSA algorithm involves learning the environment by choosing actions at each state using a policy function.
- At each state, an action is chosen and then the environment provides the reward for that action along with the next state. Based on the reward, the Q-values are updated in the Q-table.

METHODOLOGY - OBSTACLE DETECTION

- **Stereo vision** is used for the computation of depth based on the binocular disparity between the images of an object in left and right eyes.
- Stereo Camera based depth estimation is used for detecting obstacles in the testing region
- Generated depth map is processed by thresholding to identify objects within a certain distance.
- This is then used to detect the presence of obstacles that are large enough in the immediate next cell of the testing space.

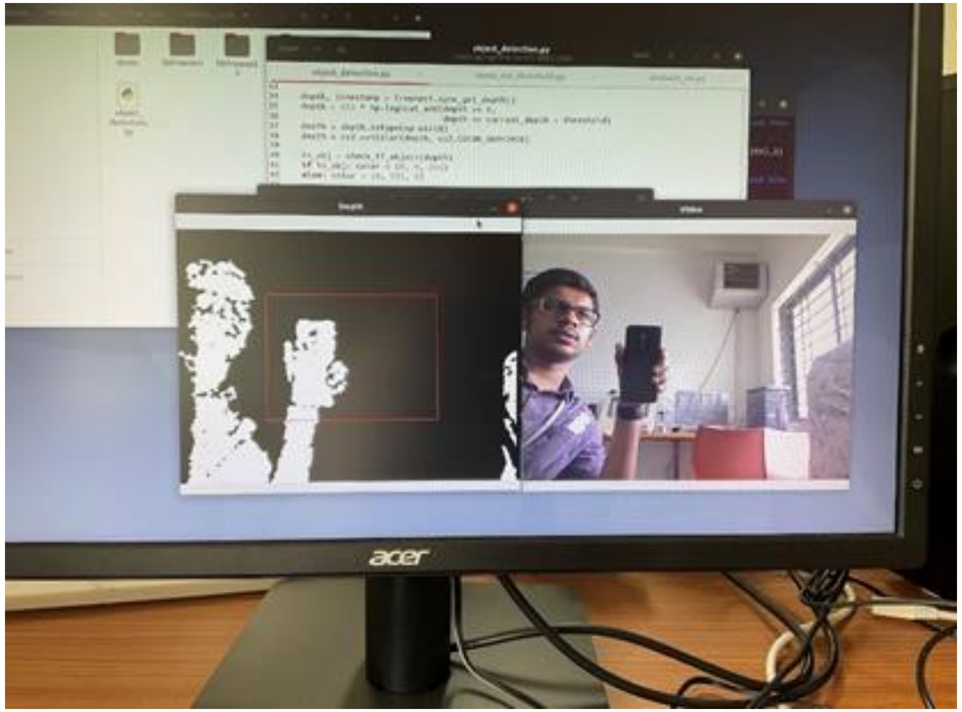


Figure 2: Obstacle detection output using stereo camera depth estimation

METHODOLOGY – HARDWARE IMPLEMENTATION

- To perform hardware implementation, a robot is designed using NVIDIA Jetson Nano Developer Kit as the Micro-controller and XBOX 360 Kinect camera.
- The Kinect camera provides the required depth map by utilizing a stereo camera setup along with an IR camera.
- An MPU6050 sensor is used to accurately turn by integrating the angular velocity output of the sensor to obtain angular displacement.
- A Li-Po battery is used to power four 12V 100 RPM DC motors which are controlled by the L298N motor driver.
- The Jetson Nano and the Kinect camera are currently externally powered using a 5V4A AC.

Figure 3: Testing space with robot and obstacles

ROBOT DESIGNED

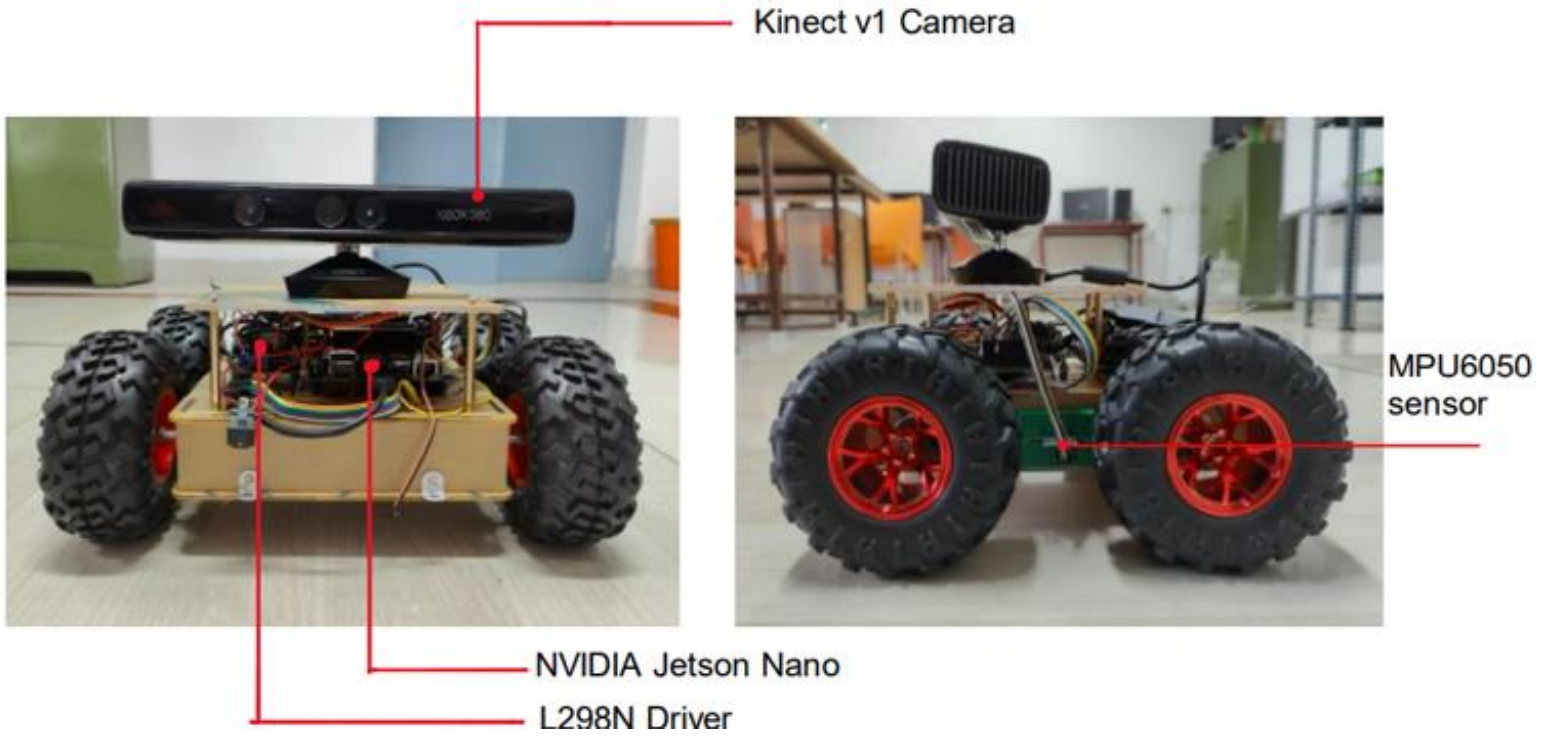
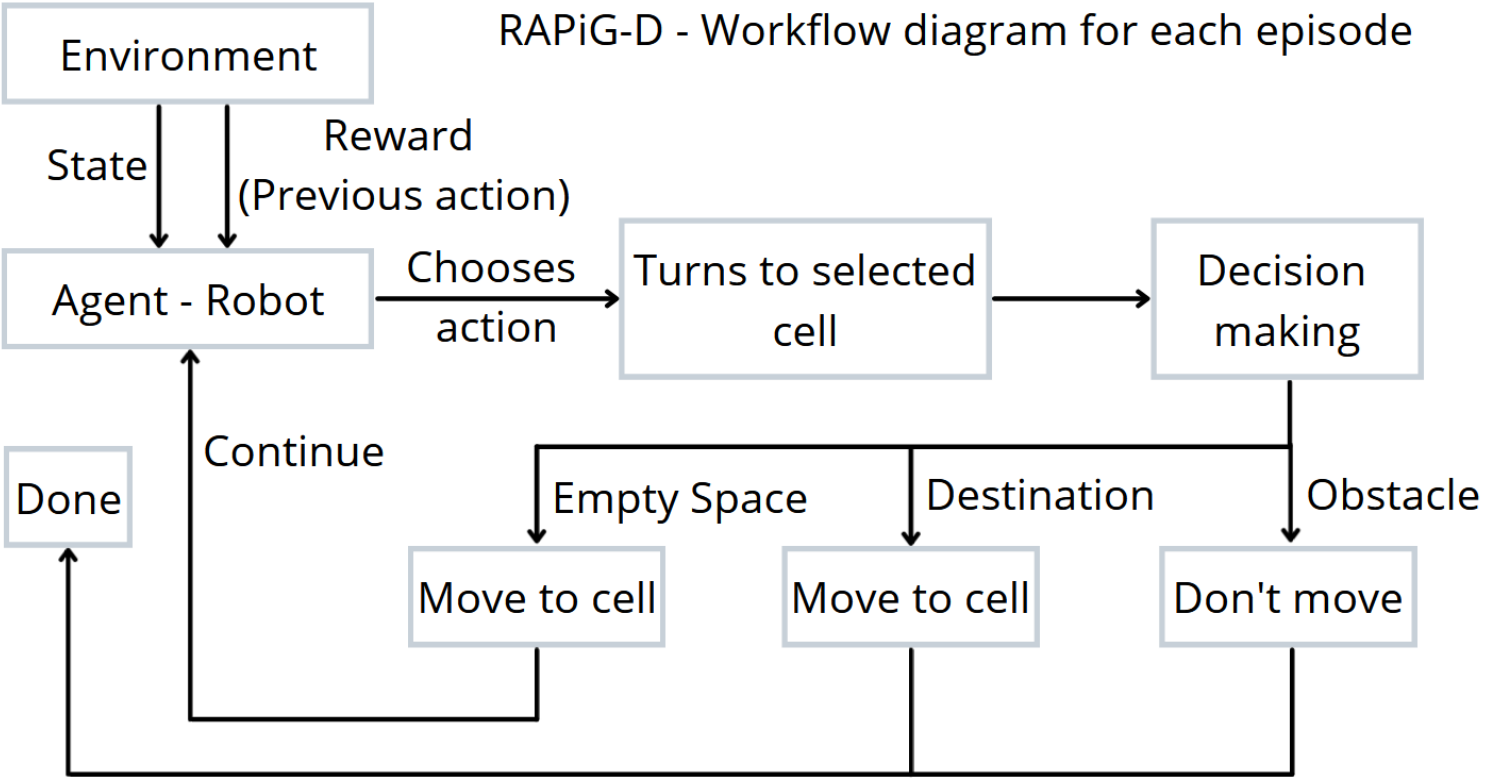


Figure 4: Robot designed for hardware implementation

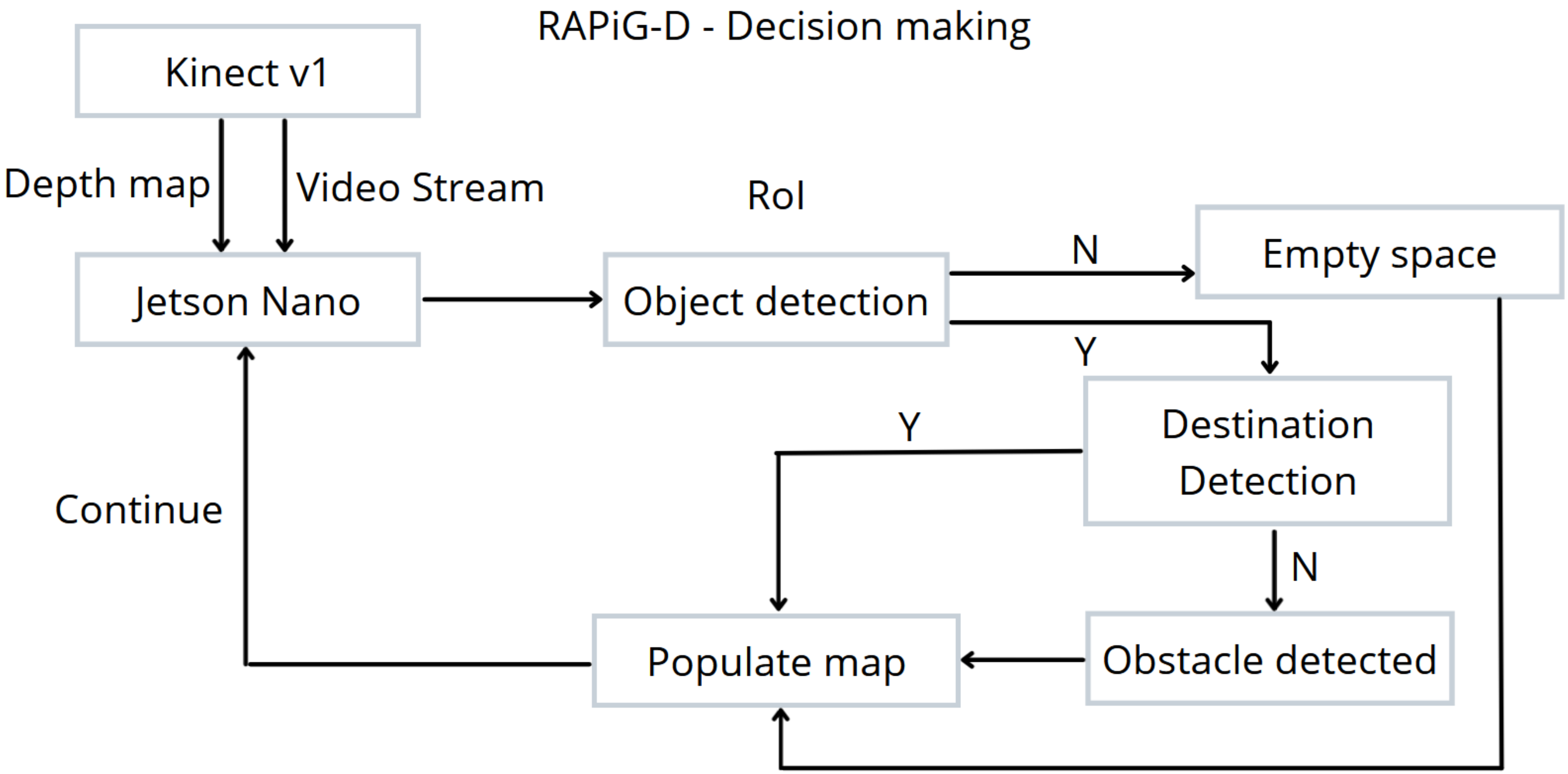
COMPONENTS USED

COMPONENTS	SPECIFICATION	NUMBER USED
Jetson Nano	Developer Kit - 4GB	1
Robot Chassis	Dimensions: 250 x 200 x 46 mm	1
DC motor	100 rpm 12V	2
Wheels	Diameter-130mm Width-60mm	4
Kinect v1 Camera	Stereo camera, IR camera	1
Motor Driver IC	L298N	1
MPU-6050	6 Axis motion tracking	1
Battery	11.1 V 4200mah Li-Po	1

FLOWCHART


RAPiG-D - Workflow diagram for each episode
Environment
State
Reward
(Previous action)
Agent - Robot
Chooses
action
Turns to selected
cell
Decision
making
Continue
Done
Empty Space
Destination
Obstacle
Move to cell
Move to cell
Don't move

FLOWCHART


RAPiG-D - Decision making
Kinect v1
Depth map
Video Stream
RoI
Jetson Nano
Object detection
N
Empty space
Y
Destination Detection
Y
N
Continue
Populate map
Obstacle detected

ALGORITHM

Step 1: Reset environment at start of each episode
Step 2: Choose action at each state based on policy function
Step 3: Check validity of next state
Step 4: Turn to required direction based on action chosen
Step 5: Check next state for presence of obstacle
Step 6: If no obstacle, move to next state and continue
Step 7: Check if new state is the destination given
Step 8: If destination, current route is checked for shortest path from start to destination
Step 9: Each episode ends when either an obstacle is detected or destination is reached and the obstacles are populated in the map
Step 10: A specified number of episodes are run and the agent learns the optimal path and location of obstacles

RESULTS AND DISCUSSIONS

- To check the validity of the path planning algorithm, software simulation was performed on a larger 10 x 10 environment with 10 obstacles and the agent ran 1000 episodes.
- Software implementation takes less time per episode. The start point is (0,0) and the destination is (9,9).
- The obstacle coordinates are manually set and fed to the algorithm.
- Based on the policy, the RL agent goes around the environment, checking each coordinate to see if it is an obstacle, an empty space, or the destination.
- As the episodes progress, the number of steps taken in each episode reduces. This is indicative of the learning process of the agent, allowing it to adapt to the unknown environment after exploring and mapping obstacles and identifying the shortest path to the given destination.

RESULTS AND DISCUSSIONS

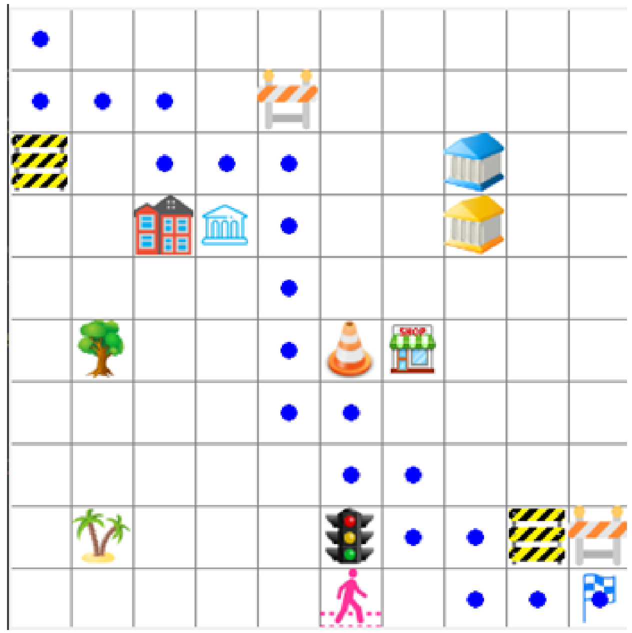


Figure 5: Map generated in software implementation

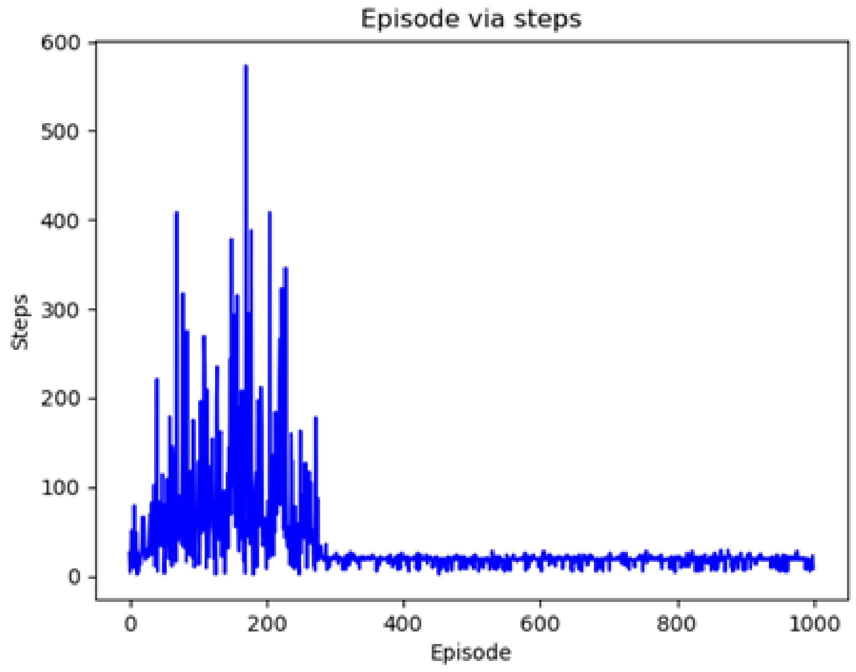


Figure 6: Plot of number of steps in each episode – Software implementation

RESULTS AND DISCUSSIONS

- The robot agent was allowed to explore a 3 x 3 testing space for 10 episodes.
- The starting point is at (0,0) and the destination is at (2,2).
- The agent successfully detected obstacles in adjacent cells and populated the detected obstacles as expected.
- The movement is also accurate, controlled by the MPU6050 gyro sensor.
- After 10 episodes, a map is generated along with the optimal path (Figure 7)

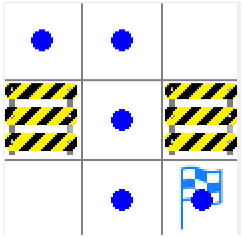


Figure 7: Map generated in hardware implementation

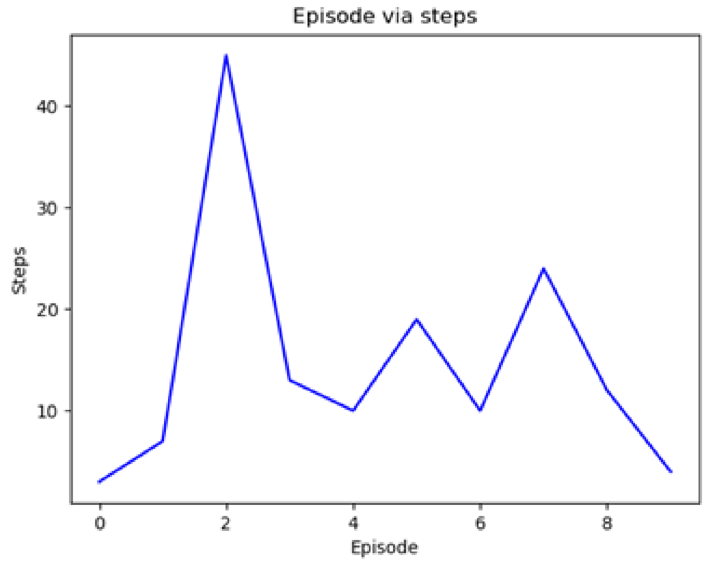
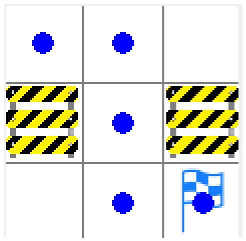


Figure 8: Plot of number of steps in each episode – Hardware implementation

RESULTS AND DISCUSSIONS

	Up	Down	Right	Left
[0.0,0.0]	0.00	-0.010000	3.615840e-08	0.00
Obstacle	0.00	0.000000	0.000000e+00	0.00
[40.0,0.0]	0.00	0.000008	0.000000e+00	0.00
[80.0,0.0]	0.00	-0.010000	0.000000e+00	0.00
[40.0,40.0]	0.00	0.000882	-1.000000e-02	-0.01
[40.0,80.0]	0.00	0.000000	4.900995e-02	0.00
[0.0,80.0]	-0.01	0.000000	2.673090e-04	0.00
Goal	0.00	0.000000	0.000000e+00	0.00



Final Q-Table generated by the RL SARSA algorithm

RESULTS AND DISCUSSIONS

- Figure 9 shows the shortest path determined by the algorithm.
- It is observed that at any given state the Q-value corresponding to an action that would lead to an obstacle in the next state is negative and the action that would allow the agent to move closer to the destination or to the destination itself is positive.
- The agent then chooses which action to take at each state and is able to identify the optimal path to reach the destination while avoiding obstacles.

```
The shortest route: 4
The longest route: 14
[40.0, 0.0]
[40.0, 40.0]
[40.0, 80.0]
[80.0, 80.0]
```

Figure 9: Shorted route and longest route

CONCLUSION

- The robot as an RL agent, successfully performs RL based path planning to accurately map the environment and its obstacles and determine the shortest path to reach a given destination.
- In the 3x3 environment, the robot agent performs accurate obstacle detection implementing depth estimation and by using the SARSA algorithm, identifies the optimal path to the destination within 10 episodes.
- As the number of obstacles and the size of the environment increases the time taken to identify the shortest path would also increase.
- Thus, a simple RL algorithm such as SARSA is used to identify an optimal path and is able to allow the robot agent to map an unknown environment.

FUTURE WORKS

- Can be further extended to larger environments and even dynamic environments.
- Future implementations may involve increasing the degrees of freedom of the agent and the environment states and even going beyond cell-based environments.
- It can also be implemented in diversified applications ranging from garden or warehouse management, Air crash investigations, Search and Rescue operations, serving food at restaurants, and even Space exploration.

REFERENCES

- Valentyn N. Sichkar, “Reinforcement Learning Algorithms in Global Path Planning for Mobile Robot,” In: International Conference on Industrial Engineering, Applications and Manufacturing, 2019

- Martin Gromniak and Jonas Stenzel ,"Deep Reinforcement Learning for Mobile Robot Navigation", Asia-Pacific Conference on Intelligent Robot Systems (ACIRS),July 2019

- Guillaume Sartoretti , Justin Kerr , Yunfei Shi, Glenn Wagner, T. K. Satish Kumar, Sven Koenig, and Howie Choset,“PRIMAL: Pathfinding via Reinforcement and Imitation Multi-Agent Learning”, Ieee robotics and automation letters, , Vol. 4, no. 3, July 2019.

- Hyansu Bae , Gidong Kim , Jonguk Kim , Dianwei Qian 4 and Sukgyu Lee , “Multi-Robot Path Planning Method Using Reinforcement Learning”, Multidisciplinary Digital Publishing Institute, Vol 3,No 4, May 2019.

- Jing Xin,Huan Zhao,Ding Liu,”Application of Deep Reinforcement Learning in Mobile Robot Path Planning”, Ieee robotics and automation letters, Vol 2, No 3 , January 2019.

END OF
PRESENTATION