Project Report for Smart Mobility Lab

Project Report: TurtleBot3 Reinforcement Learning

Student ID: 12200293

Name: Rakhmatov Shakhzodbek

Introduction:

Our project involves implementing reinforcement learning on TurtleBot3, a popular robot platform. The goal is to train TurtleBot3 to perform tasks autonomously through a process of trial and error, allowing it to learn from its interactions with the environment.

Components:

1. TurtleBot3 Robot: We are using the TurtleBot3 robot, a compact and versatile robot commonly used in research and education.

2. ROS (Robot Operating System): The project leverages ROS, a flexible framework for writing robot software. ROS helps in managing hardware abstraction, device drivers, communication between processes, and more.

3. Gazebo Simulation: Gazebo is employed for simulating the robot's environment. This allows us to train and test the reinforcement learning algorithms in a controlled virtual space before deploying them to the physical robot.

4. Reinforcement Learning: We use reinforcement learning, a type of machine learning, to train TurtleBot3. Reinforcement learning enables the robot to learn from its actions and make decisions to maximize a reward signal.

Code Overview:

1. TurtleBot Main Code: This code initializes the TurtleBot, subscribes to sensor data (such as camera images and laser scans), and controls the robot's movements. The robot moves in its environment while collecting data that will be used for training the reinforcement learning model.

2. Gazebo Interface Code: This code interacts with the Gazebo simulation environment. It provides functions to pause, unpause, and reset the simulation. The position of the TurtleBot is also configured based on different scenarios.

3. Reinforcement Learning Code: The main reinforcement learning code orchestrates the training process. It sets up the learning environment, defines the neural network model for the agent, and utilizes the stable-baselines3 library for reinforcement learning. The training process involves multiple episodes, where the TurtleBot learns to perform tasks through interactions with its environment.

Training Process:

1. Initialization: The project starts by initializing the ROS node, defining parameters for training, and creating the TurtleBot3 environment.

2. Model Architecture: We design a convolutional neural network (CNN) model for reinforcement learning. The model takes images from the robot's camera as input and outputs actions for the robot to take.

3. Training Loop: The training loop consists of multiple episodes, each comprising a sequence of steps where the TurtleBot interacts with the environment, receives rewards, and adjusts its behavior to maximize cumulative rewards.

4. Monitoring and Logging: The project monitors and logs the training progress, including rewards obtained during each episode and visualizations of the training process.

5. Saving and Loading Model: Trained models are saved periodically to resume training later or deploy the learned policy on the physical TurtleBot.

Conclusion:

In summary, our project focuses on empowering TurtleBot3 with the ability to autonomously learn and adapt to its surroundings using reinforcement learning. The combination of ROS, Gazebo simulation, and a well-designed neural network facilitates the training process. The aim is to deploy a trained model onto the physical TurtleBot3, enabling it to perform tasks in real-world environments based on its learned behaviors.

References:

Certainly! Here are some general references and resources that can be helpful for a project involving TurtleBot3, ROS, Gazebo simulation, and reinforcement learning:

1. TurtleBot3 Documentation:

- [TurtleBot3 Documentation] (https://emanual.robotis.com/docs/en/platform/turtlebot3/overview/): Comprehensive documentation for TurtleBot3, including hardware specifications, software setup, and tutorials.

2. ROS (Robot Operating System):

- [ROS Wiki] (http://wiki.ros.org/): The official ROS Wiki provides documentation, tutorials, and resources for understanding and working with the Robot Operating System.

3. Gazebo Simulation:

- [Gazebo Documentation] (http://gazebosim.org/tutorials): Tutorials and documentation for Gazebo simulation, which is commonly used for testing and simulating robots.

4. Reinforcement Learning:

- [OpenAI Gym] (https://gym.openai.com/): A toolkit for developing and comparing reinforcement learning algorithms. It provides various environments and tools for reinforcement learning experiments.

- [Stable Baselines3] (https://github.com/DLR-RM/stable-baselines3): A set of high-quality implementations of reinforcement learning algorithms in Python, built on top of OpenAI Gym.

5. Neural Networks with Keras:

- [Keras Documentation] (https://keras.io/): Documentation for Keras, a high-level neural networks API. It is often used for building and training neural network models.

6. Python Programming:

- [Official Python Documentation] (https://docs.python.org/3/): The official documentation for Python, the programming language commonly used in ROS and reinforcement learning projects.

7. Books:

- "Programming Robots with ROS" by Morgan Quigley, Brian Gerkey, and William D. Smart: This book introduces programming robots using ROS.

- "Reinforcement Learning: An Introduction" by Richard S. Sutton and Andrew G. Barto: A widely used textbook for understanding the fundamentals of reinforcement learning.