Reinforcement learning for robust mobile robot navigation control

Project Plan

Zhicun Tan(s202711)
Master of Science in Engineering

2023

DTU ELECTRO
Department of Electrical and Photonics Engineering

Reinforcement learning for robust mobile robot navigation control , Project Plan

Report written by:
Zhicun Tan(s202711)

Advisor(s):
Jens Christian Andersen, Associate Professor at the Department of Electrical and Photonics Engineering of DTU
Emmanuel Dean, Electrical Engineering Department of Chalmers University of Technology

DTU Electrical Engineering
Technical University of Denmark
2800 Kgs. Lyngby
Denmark

elektro@elektro.dtu.dk

Project period:	1 February 2023- 1 July 2023
ECTS:	30
Education:	M.Sc.
Field:	Electrical Engineering
Class:	Public
Edition:	1. edition
Remarks:	This report is submitted as partial fulfilment of the requirements for graduation in the above education at the Technical University of Denmark.
Copyrights:	©Zhicun Tan, 2023

Contents

CHAPTER 1

Introduction

Modern manufacturing industry needs fewer humans in the factories to cut costs and improve efficiency. Autonomous Transport Robots(ATR) is thus designed to deliver items to the production line. However, classical navigation and control methods might fail in a highly dynamic environment. To this end, a reinforcement learning(RL) agent might handle unsafe conditions in a better manner.

1.1 Project background

The ATRs work on a dynamic and stochastic factory floor, which requires the agent to be able to react to potential collisions e.g. with a walking passenger or a toolbox on its way. Moreover, the end time of delivery should also be strictly controlled otherwise the whole production line might pause due to a lack of spares at some stages.

Another big challenge comes from the sensors equipped by ATRs. Robots rely heavily on sensors e.g. IMU, odometer or scanners. The guidance and control system might easily fail when one or more sensors break down or simply provide low-accurate measurements.

1.2 Project description

This thesis will propose a robust navigation controller using reinforcement learning methods. The controller should be able to control the ATR to follow the desired trajectory, avoid collisions and keep the delivery deadline. The obstacles could be static or dynamic. The learning step will be performed online.

The experiment will be implemented in a simulation environment that mimics different scenarios in factories and warehouses. Besides, the performance difference between the RL-based controller and the MPC-based controller will be evaluated.

The project is in collaboration with Volvo GTO.

CHAPTER 2

Methodology

2.1 Literature study

The main reference will be *Reinforcement Learning: An Introduction* by Richard S. Sutton and Andrew G. Barto. Other papers related to the implementation of an RL-based controller applied on mobile robots, vessels, etc. will be included as well.

An earlier master thesis was trying to use model predictive control (MPC) [JF22]to solve the task. So it is worth reading to understand the advantages and drawbacks of the MPC method as a reference with RL controller.

2.2 Understanding the current MPC system, ROS2 simulation, and navigation environment

As can be seen from fig2.1, the current project is quite enormous. It includes modules e.g. path generator, trajectory generator controller, etc. Reading through each of these modules will provide a good understanding of the whole system, which benefits later when implementing the new controller.

2.3 Develop RL algorithms

Basic reinforcement learning algorithms are quite straightforward. A classic problem is ”grid-based path-planning” problem. However, it’s still not very clear to me how to make RL algorithm a robust controller.

A possible procedure could be, implementing a basic RL controller demo used in another scenario. Then investigating how to modify or migrate it to make it applicable to the ATR project.

Apart from inventing wheels myself, open-source RL libraries like Gym might be helpful when designing RL framework. But how to make the library collaborates with ROS2 is still unknown.

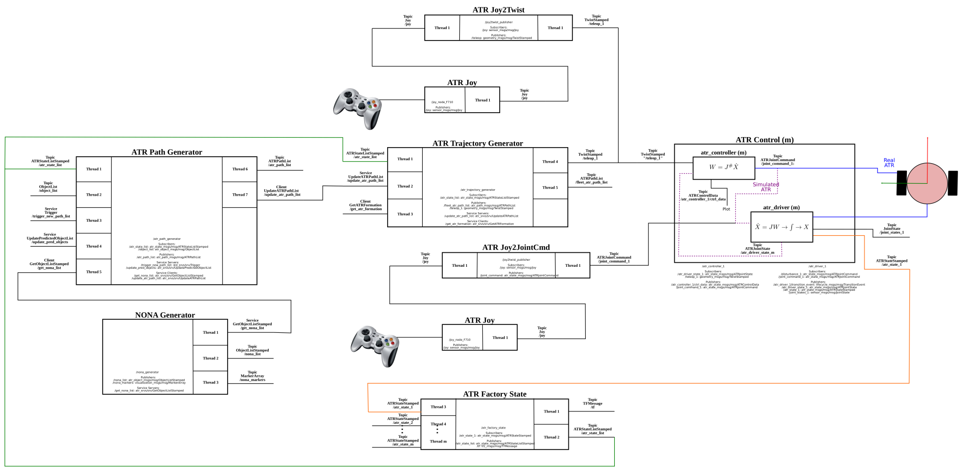

Figure 2.1: System Overview.

2.4 Make a system implementation

This is the critical part of this thesis. Once the independent RL controller is implemented, it needs to be integrated with the current simulation environment. The action output, measurement input and reward mechanism might need to be redesigned so that the controller fits the factory environment.

A visualization module to show the learning status, e.g. plotting the value table, might as well be implemented.

One final thing that may be critical is the switching schema which enables the agent to shut itself down and re-enable the MPC controller once it founds the RL learning might already fail.

2.5 Run demonstration to validate the result

A demonstration should be provided as part of the outcome of the thesis work. The simulation environment could be randomized and dynamic to evaluate the performance of RL controller as much as possible. Failure cases are of importance as well.

The comparison between two controllers is another part of the evaluation. The testing results should be evaluated in several terms e.g. task completeness, delivery latency, risk of collisions, etc.

CHAPTER 3
Tasks

3.1 Tasks

The general tasks are listed as follows.

1) Implement a RL controller that can drive the ATR following the path without obstacles.

2) The controller can generate a new trajectory to avoid the static obstacle.

3) The controller can handle the occurrence of dynamic obstacles.

4) The controller can handle other kinds of disturbances e.g. motor error or missing path information.

The above general tasks are divided into several sub-tasks.

1) Design the structure of RL controller. Define the inputs and outputs of the controller.

2) Construct training environments with no obstacles, one/multiple static obstacles, and one/more dynamic obstacles. Define the shape and action policy of the dynamic obstacles. Design the model of disturbances.

3) Define the feasible number of training episodes. Define the action and reward function of RL.

4) Train the value table or network

5) Evaluate the RL model for different scenarios.

3.2 Timeline

T1. Get the simulator working: Understand the current system and figure out where the RL algorithms should work at. Solve the problems that might appear when running the current workspace in Ubuntu 22.04, ROS2 Humble.

T2. One ATR follows a fixed trajectory: Implement a basic RL controller that can drive the robot towards the target following a given path. This path will not be blocked by any obstacles.

T3. One ATR follows its trajectory with static obstacles: There will be one static obstacle in the environment. The RL controller should now be able to avoid the collision and bring the robot back on track safely and quickly.

T4. One ATR follows its trajectory with dynamic obstacles: The obstacle is now moving in the factory. Adapt the RL controller so that it is capable of handling such circumstances.

T5. Verification: Design random generated cases to test the performance of the RL controller.

Bibliography

[JF22] Max Edin Jakobsson and Muhamed Faraj. *Trajectory planning for a fleet of autonomous transport robots*. 2022.

DTU Electro
Department of Electrical and Photonics Engineering
Technical University of Denmark

Ørsteds Plads
Building 343
DK-2800 Kgs. Lyngby
Denmark

www.electro.dtu.dk