Mapless Navigation with Deep Reinforcement Learning

Final Year Project Interim Presentation

Minghong Xu (201601082)

Department of Electrical Engineering and Electronics

University of Liverpool

13 December 2022

Navigation in Mobile Robots

Several applications of robotic system, such as rescue operation and patrolling, require robots be capable of navigating themselves to reach a goal position.

The objective of mobile robot navigation is to produce plans/policy which move a robot from its current position to a navigation goal, also assuring it won't crash against obstacles, or get lost in the process.

Map-based Navigation

Map-based navigation is the conventional approach of achieving the aforementioned objective. This approach consists of

- Self-localisation
- Path planning
- Mapping

Deep Reinforcement Learning (DRL)

Reinforcement learning method is defined as any effective way of solving **stochastic optimal control** problems such as those formulated as Markov decision processes (MDPs).

Dynamic programming (DP) is a class of solution methods of optimal control problems.

The combination of DP and artificial neural networks (ANNs) is termed as **neuro-dynamic programming** in control theory. Deep reinforcement learning (DRL) is the term used in the artificial intelligence literature.

ANN is used as an optimisable function approximator to overcome the DP's limitation called *curse of dimensionality* which means computational requirements grow exponentially with the number of state variables.

Objectives

The main objective of this project is to train a collision avoidance policy by a DRL algorithm without map information to navigate from the starting position to desired goal without running into obstacles in unknown cluttered indoor environments.

The policy takes as input the transform from robot frame to goal frame, the odometry data from wheel encoders, and data from LiDAR. In exchange, the policy outputs velocity commands which sent to the base controller of the robot.

- Specific: I think yes.
- Measurable: The robot can reach the goal at least once.
- Achievable: I have resources and skills needed to complete.
- Relevant: It's a robotics project.
- Time-bound: By the end of the FYP.

Non-Goals

The policy is safe.

The policy is robust.

The policy has well generalisation properties.

The policy is applicable to outdoor environments.

Context

Intuitively, humans and pets often move around indoors without reference to maps or floor plans.

Mapping consists of

- Map building
- Map interpretation
- Map maintaining

which imposes intensive computational demand.

Map-based approaches consist of at least six modules, each involves a number of parameters that need to be tuned manually rather than being able to learn from past experience automatically. Moreover, it is difficult for these approaches to generalise well to unanticipated scenarios.

Motivation

DRL approach is simpler than map-based approaches.

- Less modules involved
- Less parameters
- Better generalisation (expected, not guaranteed)

After training, only forward propagation, which is simply matrix multiplications for multi-layer perceptron (MLP) architecture neural network, is involved in the use of policy. With the development of tensor processing hardware, computation cost will be small.

It may not be as effective as conventional approaches, but because of the generality of DRL, it is worth trying. Even if it doesn't work well, I can still learn knowledge about machine learning from this project.

Outline Requirements

- Implementation of a DRL algorithm
- Training program
- Training environment in a simulator
- Defining reward function

Initial Design

DDPG & TD3 components:

- Neural network architecture: MLP
- Experience replay buffer: uniformly sampled or prioritised
- Exploration strategy: Gaussian noise injection in action space or parameter space

Training program adheres to Gym API which is a de facto DRL programming standard.

Robot selection : Turtlebot3, since it has a great ecosystem on Robot Operating System (ROS).

Program of the training environment partly adhere to Gym API and is wrote for Classic Gazebo Simulator. Classic Gazebo Simulator was selected because it has the best integration with ROS1.

Initial Design (continue)

The reward r at time step t is a weighted sum of three terms r^g, r^c, and r^s:

$$r_t = r_t^g + r_t^c + r_t^s$$

The robot is awarded by r_g for reaching the navigation goal:

$$r_t^g = \begin{cases} r_{big}, & \text{if } \|p_t - g\| < \text{tolerance} \\ \epsilon(\|p_{t-1} - g\| - \|p_t - g\|), & \text{otherwise} \end{cases}$$

When the robot collides with obstacles, it is penalised by r_t^c:

$$r_t^c = \begin{cases} r_{collision}, & \text{if collision} \\ 0, & \text{otherwise} \end{cases}$$

A small fixed penalty r^s is given at each time step.

Gantt Chart

https://github.com/MinghongAlexXu/final-year-proj/blob/main/README.md

Next Steps

- Complete the program for the simulated training environment.
- Train a policy and evaluate it in an unanticipated environment.
- Draft the final report and prepare a demo for bench inspection.

Summary

Summarising, this presentation covers the background, objectives, motivation, requirements, and planning of the project. The initial design and what has been done so far are reported, and the next steps are outlined.

Table of Contents