DEPARTMENT OF INFORMATICS

TECHNISCHE UNIVERSITÄT MÜNCHEN

Bachelor's Thesis in Informatics

Reinforcement Learning for Path Planing of Robotic Arms

Anton Mai

DEPARTMENT OF INFORMATICS

TECHNISCHE UNIVERSITÄT MÜNCHEN

Bachelor's Thesis in Informatics

Reinforcement Learning for Path Planing of Robotic Arms

Bestärkendes Lernen zur Pfadplanung von robotischen Armen

Author:	Anton Mai
Supervisor:	Prof. Dr.-Ing. habil. Alois Knoll
Advisor:	Dr Zhenshan Bing
Submission Date:	February 17, 2020

I confirm that this bachelor's thesis in informatics is my own work and I have documented all sources and material used.

Munich, February 17, 2020 Anton Mai

Acknowledgments

Abstract

Contents

1 Introduction

Lee Sedol, one of the best Go players in the world, was beaten by the Go engine AlphaGo in a match. The engine was clearly stronger. AlphaGo only knew the rules at the beginning and got stronger only by playing with itself. Artificial Intelligence is quite popular nowadays because of its many use cases: Self-Driving cars, playing atari games, robotics and more. But how do these engines learn how to get so good at their areas ? The answer is reinforcement learning, an area of machine learning.

The idea of reinforcement learning is to have a state and actions that an agent can choose from. Each action results in different rewards and states. Rewards are used by agent to measure how good an action was. This process is repeated which results in the agent learning which actions in each state are better. Imagine you are a soccer player. You are standing in front of the goal (which is the state you are in). You can either shoot or pass the ball (which are your available actions). You choose to shoot, but the ball is blocked by the goalkeeper (you got a low reward). So the next time you are in front of the goal again, you will more probably try to pass the ball. This time your teammate scored a goal (you got a high reward). From this experience you learn that it is probably better to pass the ball if you are standing in front of the goal. The concept of reinforcement learning can be used in a variety of environments, for example robotic arms.

Already in the 14th century, Leonardo da Vinci made blueprints of robotic arms.

A robotic arm resembles a human arm. It consists of segments which are connected by joints. The number of joints correspond to what is called Degrees of Freedom. A robotic arm with 5 joints would have 5 Degrees of Freedom because it can pivot in 5 ways. Each joint is connected to a step motor. Step motors make the robot move very precisely. The equivalent to a human hand is the end effector. The end effector can vary depending on the tasks.

Robotic arms have many advantages. They are very accurate and consistent which is why they are mostly used for repetitive tasks or tasks that require high accuracy which

are hard for humans. This is the main reason why they are used in laboratories and hospitals for surgeries. They can also be used automatically without any human which is why they are used for manufacturing and assembly lines.

Humans still have to teach the robotic arms how to move when setting them up. For path planning of the robotic arm, a sequence of actions has to be found that solves the task. This sequence is saved and repetitively executed by the robotic arm. Finding the path still requires human labor. Either by testing or by using linear algebra a path can be found. A robotic arm needs 6 Degrees of Freedom to be able to move its end effector in every direction and orientation. This also means that robotic arms with more degrees of freedom do not have a unique path to solve the tasks. There are different paths which can vary in length and energy consumption. To improve the quality of the path and to do path planning without a human, using reinforcement learning for robotic arms is a logical approach.

...

There is an issue that prevents robotic arms to learn with reinforcement learning. It is hard to construct a suitable reward function for tasks where robotic arms are used. For example ... So either a suitable reward function has to be constructed by hand, or the simplest reward function, a binary sparse reward function has to be used. Both approaches have some issues. Constructing a reward function can be quite complicated. Also, for each task an individual reward function has to be made. So someone has to do this work which defeats the purpose of using reinforcement learning for robotic arms over path planning by hand. Depending on the case it might be easier to just plan the path without reinforcement learning. Using only a sparse reward for robotic arms is as follows. a reward is given, when the goal is reached, no reward is given when the goal is not reached. Robotic arms have usually many Degrees of Freedom, so there are many actions that can be taken by the robotic arm. It is quite unlikely for robotic arms to fulfill the task by doing random movements. Tasks like moving an object are near impossible to solve with random actions. So it is very unlikely for the robotic arm to earn a reward and learn. It takes a very long time to train a robotic arm with sparse rewards. But recently hindsight experience replay has been introduced. Hindsight experience replay allows a high learning rate even with sparse rewards.

Hindsight experience replay works as follows.

This thesis is structured as follows: Chapter 2 describes the theoretical background on robotic arms, reinforcement learning and algorithms like deep deterministic policy gradients and hindsight experience replay. Chapter 3 explains the methodology used for this thesis. Chapter 4 gives an overview of the simulation environment. In chapter 5, the experiments are presented and the results are discussed. In the last chapter, the results are summarized and suggestions for further word is provided.

2 Theoretical Background

2.1 Robotic Arms

2.2 Reinforcement Learning

2.3 Deep Deterministic Policy Gradients

2.4 Hindsight Experience Replay

3 Methodology

To provide a sound answer to how hindsight experience replay performs on harder tasks in contrast to easier tasks, the obvious approach is a quantitative approach. In order to collect data, a simulation environment is built for tasks of different difficulties. The robotic arm is trained for those tasks with hindsight experience replay. The performance for the training period is measured. The data will show the performance on the tasks over time and characteristics like learning rate and consistency is shown through the data. Data between easier and harder tasks is evaluated and compared to show how hindsight experience replay performs on different tasks. In some cases, it is clear that hindsight experience replay fails for the harder tasks. Possible reasons for the lack of performance are presented. An alternative extension for hindsight experience replay, hindsight goal generation will be used in addition to show possible approaches on solving the tasks with her for harder tasks. Hindsight goal generation will also be used on the easier tasks to make them comparable. Validity is obviously given, because the data measured is exactly the performance and learning rate of the robotic arm which is the measurement needed to evaluate the performance of hindsight experience replay. Similar results can be reproduced when repeating the data collection. The results are not necessarily exactly the same when reproduced. This is due to the nature of reinforcement learning as there is some randomness in the Markovian decision process when choosing an action. The law of large numbers states that with rising amount of samples the results will converge towards the expected probability. In context of reinforcement learning, the training time to learn needed might vary slightly, but the end performance should converge towards the same value with rising training time.

4 Simulation Environment

4.1 MuJoCo

4.2 OpenAI

4.2.1 OpenAI Gym

4.2.2 OpenAI Baselines

4.3 Model

4.3.1 FetchGolf

4.3.2 FetchToss

5 Experiments

5.1 FetchGolf

5.1.1 Task Description

5.1.2 Environment

Action Space

Observation Space

5.1.3 Results

5.1.4 Discussion

5.2 FetchToss

5.2.1 Task Description

5.2.2 Environment

Action Space

Observation Space

5.2.3 Results

5.2.4 Discussion

6 Conclusion

List of Figures

List of Tables