Autonomous Agent-Based system for navigation and obstacle avoidance using machine learning

M Shmurygin*

Kazakh-British Technical University, Tole Bi 59, Almaty, Kazakhstan

**Corresponding author's e-mail: shmurygin.mikhail@gmail.com*

Abstract

The goal of the project is the development of autonomous robotic platform that can navigate and explore unknown environment by itself. This paper describes the usage of Reinforcement Learning algorithms like Queue Learning (Q-learning) and its modifications and optimizations on the real physical model. The algorithm for finding the optimal strategy is implemented in the C++ code using the "Q-Learning" method. Training with reinforcement is a method of machine learning, in which a model is trained with no information about the system, but has the ability to perform any actions in it. Actions translate the system into a new state and the model receives some reward from the system.

Keywords: Reinforcement Learning, Microcontroller, Robot, Q-Learning, Navigation

1 Introduction

The Q-Learning algorithm was proposed by Watkins in 1989. It is a part of learning algorithms that uses reinforcement. The usage of this algorithm can be applied to the system where agent should find an optimal strategy for interaction. For the training process "rewards" that have numerical values are used. The effective training and navigation of the platform depends on the construction of the state-action pair estimation function.

2 Navigation planning

The goal of the work is to teach robotic platform to learn the unknown environment. The agent is placed in a rectangular arena with a fixed surroundings in it and requested to calculate the shortest path. The goal is to calculate the navigation plan from starting point to the exit. This task is complexed because arena contains multiple obstacles and thus the finding of optimal path is impossible without initial exploration. Ultrasonic sensor that is placed on the rotation platform helps to avoid collisions and create a mapping of the space. The usage of such technique allow us to teach robotic agent to find optimal path without human intervention.

3 Q-Learning

The basis of the Q-Learning method is the matrix of the weights of the system state. The matrix Q is a collection of all possible states of the system and the weights of the system's response to various actions. In this problem, possible combinations of the parameters of the system are 16 = 4 ^ 2. In each of the 16 states of the system, it is possible to perform 4 different actions (Action = 0,1,2,3). Actions are set as following: North – 0, South – 1, East – 2, West – 3. The initial values of the Q matrix are set to zero and then will be updated during the learning process. Table 1 shows the initial state of the R matrix which represents rewarding values of the Q learning update function. The zero column contains the row index, the four columns equal the weights for the actions 0, 1, 2 and 3. Each line represents the unique state of the system. When initializing the table, we assign all the weights to the corresponding values to block the transitions we do not need e.g. north to south not allowed and north to north is not allowed.

TABLE 1 Initialization of the matrix R

	N	**S**	**E**	**W**
N	0	-1	0	0
S	-1	-1	0	0
E	10	-10	-1	-1
W	10	-10	-1	-1

In the simplest form the one-step Q-Learning formula is defined:

$$Q(S_t, A_t) \leftarrow$$

$$Q(S_t, A_t) + \alpha \left[R_{t+1} + \gamma \max_a Q(S_{t+1}, a) - Q(S_t, A_t) \right]$$

Where γ or Gamma represents the learning parameter. The range of this value is set from 0 to 1. When Gamma is close to zero the agent will tend to consider only immediate rewards. If Gamma is closer to one, the agent will consider future rewards with greater weight, willing to delay the reward. We set gamma parameter to 0.9 which promises to be the optimal value. The calculation of the suitable gamma parameter is one of the goals of the future research.

4 Q-Learning Software

The Learning software and corresponding algorithm can be described as follows:

Initialize $Q(s, a), \forall s \in A(s)$, arbitrarily, $Q(terminal - state, \cdot) = 0$

Repeat (for each episode):

Initialize S:

Repeat (for each step episode):

Choose A from S using policy derived from Q

Take action A, observe R, S'

$$Q(S_t, A_t) + \alpha \left[R_{t+1} + \gamma \max_a Q(S_{t+1}, a) - Q(S_t, A_t)\right]$$

$$S \leftarrow S'$$

The algorithm described above will repeat until the goal state is achieved. Agent will calculate the actions with highest rewards and update the Q matrix for each corresponding pair state-action. Each episode of this algorithm is considered as a training session. Which means that the more session the platform is accomplishing the more optimized Q matrix is becoming.

5 Hardware platform

For this project a specialized hardware platform was developed that uses Arduino microcontroller. Important factor for the autonomous robotics is the limitations that are set on the computational power. In this case all calculations and processing is done on the platform itself without human interaction. However, Bluetooth module for the monitoring of the current state of the learning parameters and distance sensors is used. Robotic platform is represented by a 4-wheel rover that can easily move in four different directions.

6 Conclusions

During the research process the usage of Reinforcement learning algorithms was applied to the real physical agent. Specialized hardware platform has been developed.

In future work it is planned to use different optimization methods for Q-Learning algorithm in order to speed up the learning process and make system more adaptive to the surrounding conditions. Another direction of research is to learn what other factors can be applied for the reward computation including the speed of the motors and power consumption. The other important goal is to learn robotic platform adapt to dynamically changing environment.

Acknowledgments

I want to thank Yakunin Kirill my science supervisor for great support in preparing material and working with theoretical aspects of machine learning algorithms applied to the hardware systems and also the Faculty of Information Technologies of Kazakh-British Technical University for providing the financial support in constructing of the hardware platform of this project.

References

[1] Sutton R S, Barto A G 2012 *Reinforcement Learning: An Introduction*
[2] *Q-Learning tutorial* http://mnemstudio.org/path-finding-q-learning-tutorial.htm
[3] *Learning Reinforcement algorithm on real physical model* https://habrahabr.ru/post/308094/ -
[4] Nazmul Siddique, Hojjat Adeli *Computational Intelligence: Synergies of Fuzzy Logic* Neural Networks and Evolutionary Computing
[5] Monk S 2011 *Evil Genius* USA: McGraw-Hill/TAB Electronics
[6] Margolis M 2011 *Arduino Cookbook* USA: O'Reilly Media, Inc