Reinforcement Learning
PID controller

Aqeel Labash

June 24, 2016

Introduction

The project idea is to use reinforcement learning to keep a robot away from wall by certain distance instead of using PID[1] controller. The problem in PID controller that it needs manual tuning for parameters for it to work fine.

PID-controller

PID stands for proportional–integral–derivative controller. Usually when we want a robot to move to a certain point moving. Moving it by static value won't be sufficient, that's because it might overshoot and the orders aren't instantly executed. Also, wheels might slip. In industry PID commonly used and it's given by following formula :

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau)d(\tau) + K_d \frac{de(t)}{dt}$$

Those three parts acts for :

1. P deal with the current error value.
2. I deal with the past errors so if P made error in previous steps it will be fixed here.
3. D expect the future error depending on the speed of error change.

The problem with the previous solution is it depend on K_p, K_i, K_d which require tuning and take long time to be tuned.

Using Sarsa as Replacement

Here I propose using Sarsa algorithm instead of the previous formula. And here is the steps I followed :

1. Build a robot simulator
2. Generate domains representing the robot distance from the wanted point.
3. Generate actions [-255,255] represent the motor speed.
4. implement Sarsa

For the results please check Robot_Emulator.ipynb

Using linear function approximation

The implementation already there and got some results but I believe I have some bugs in the code. And it take way much more time to train.

[1] proportional–integral–derivative controller