Reinforcement learning for UAV attitude control

Wu, Po Hsun

Department of Aerospace Engineering
Tamkang University

June 9, 2022

Contents

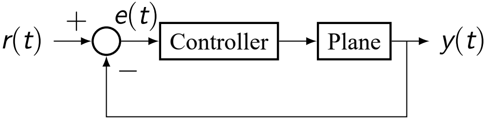
1. Introduction
2. Paper Review
3. Reinforcement Learning
4. Expectation
5. Q&A

Introduction

1. What is Attitude Control?
2. What is Reinforcement Learning?

What is Attitude Control?

- For Classical Control Theory



$r(t)$: reference
$y(t)$: state
$e(t)$: error

Figure 1: Block diagram for classical control

What is Reinforcement Learning?

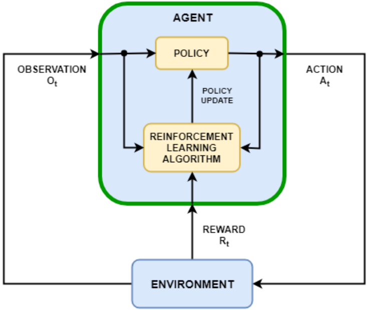
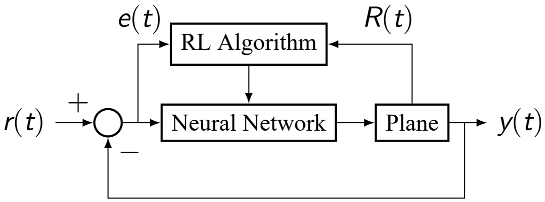


Figure 2: Reinforcement Learning architecture[1]

Mix it together

- Apply reinforcement learning to control theory.



$r(t)$: reference
$y(t)$: state
$e(t)$: error
$R(t)$: Reward

Figure 3: Block diagram for Neural Network controller

Paper Review

- W. Koch, "Flight controller synthesis via deep reinforcement learning," *CoRR*, vol. abs/1909.06493, 2019
  1. Adding noise to the plane(or environment).
  2. Using Gazebo as a physics simulator.
- W. Koch, R. Mancuso, R. West, *et al.*, "Reinforcement learning for UAV attitude control," *CoRR*, vol. abs/1804.04154, 2018
  1. Provide a training framework.
  2. Comparing some RL algorithm training results.

Reinforcement Learning

- RL is a area of Mechine Learning.
- RL will interact with the environment.
- RL aims to achieve the maximum reward by changing the neural network parameters.

$$\arg\max_{\theta} R(t) \tag{1}$$

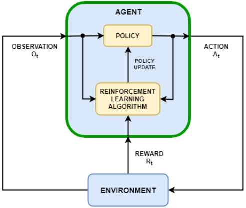


Figure 4: Reinforcement Learning architecture[1]

Artificial Neural Network

- Artificial Neural Network is a nonlinear model.
- (2) and (3) is the equations of Neural Network

$$\begin{bmatrix} a_1^{(1)} \\ a_2^{(1)} \\ \vdots \\ a_m^{(1)} \end{bmatrix} = \sigma \left(\begin{bmatrix} w_{1,0} & w_{1,1} & \cdots & w_{1,n} \\ w_{2,0} & w_{2,1} & \cdots & w_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ w_{m,0} & w_{m,1} & \cdots & w_{m,n} \end{bmatrix} \begin{bmatrix} a_1^{(0)} \\ a_2^{(0)} \\ \vdots \\ a_n^{(0)} \end{bmatrix} + \begin{bmatrix} b_1^{(0)} \\ b_2^{(0)} \\ \vdots \\ b_m^{(0)} \end{bmatrix} \right) \tag{2}$$

$$a^{(1)} = \sigma\left(\mathbf{W}^{(0)} a^{(0)} + \mathbf{b}^{(0)}\right), \quad \begin{cases} \mathbf{W} \in \mathbb{R}^{m \times n} \\ \mathbf{a} \in \mathbb{R}^{n} \\ \mathbf{b} \in \mathbb{R}^{m} \end{cases} \tag{3}$$

Artificial Neural Network

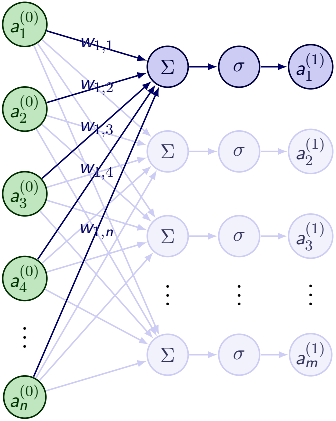


Figure 5: Artificial Neural Network

Training target

- Target function:

$$f(x) = x^2$$

- Database:
  - $x = \{0, 1, \cdots, 9\}$ adding noise with normal distribution($\mu = 0, \sigma = 0.2$).
  - 100 datas for each point(total 1,000 datas).
- Configuration:
  - 2 hidden layer, each layer with 50 neuros.
  - Two different learning rate($\alpha = 0.01, 0.001$).

Training result

- Overfitting happend at $\alpha = 0.01$.

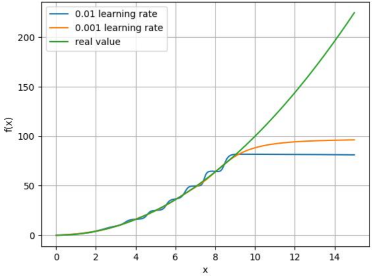


Figure 6: $f(x)$ vs x

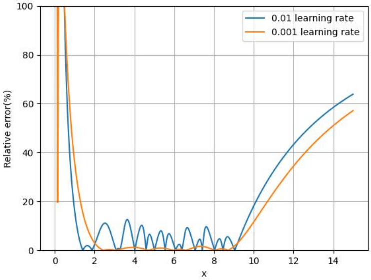


Figure 7: Relative error

Reinforcement Learning Algorithm

- Model-free:
  1. Does not require a model of the environment.
  2. It’s an explicitly a trial-and-error method.
- Model-base:
  1. Require a model of the environment.
  2. Will predict the future state.

Q-learning

- Create a Q-table for each state and action.
- Find out the next action to maximizes the Q-value in the Q-table.

Initialized

Q-Table		Actions					
		South (0)	North (1)	East (2)	West (3)	Pickup (4)	Dropoff (5)
States	0	0	0	0	0	0	0
	. . .	. . .	. . .	. . .	. . .	. . .	. . .
	327	0	0	0	0	0	0
	. . .	. . .	. . .	. . .	. . .	. . .	. . .
	499	0	0	0	0	0	0

Training

Q-Table		Actions					
		South (0)	North (1)	East (2)	West (3)	Pickup (4)	Dropoff (5)
States	0	0	0	0	0	0	0
	. . .	. . .	. . .	. . .	. . .	. . .	. . .
	328	-2.30108105	-1.97092096	-2.30357004	-2.20591839	-10.3607344	-8.5583017
	. . .	. . .	. . .	. . .	. . .	. . .	. . .
	499	9.96984239	4.02706992	12.96022777	29	3.32877873	3.38230603

Figure 8: Q-table

Q-learning

The iteration formula for Q-value

$$Q^{new}(s_t, a_t) = Q(s_t, a_t) + \alpha(r_t + \gamma \max_a Q(s_{t+1}, a))$$

The term r_t is the current reward from the environment, and $\max_a Q(s_{t+1}, a)$ is the maximum Q-value that can be obtain from next state s_{t+1}

Q-learning

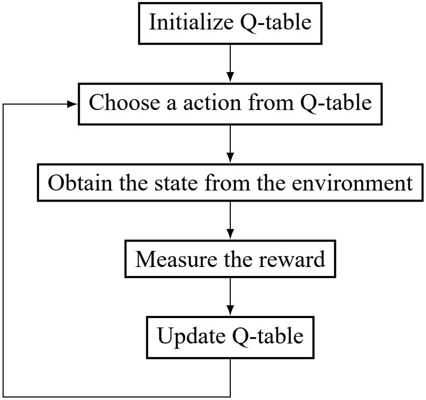


Figure 9: Q-learning flow chart

Q-learning

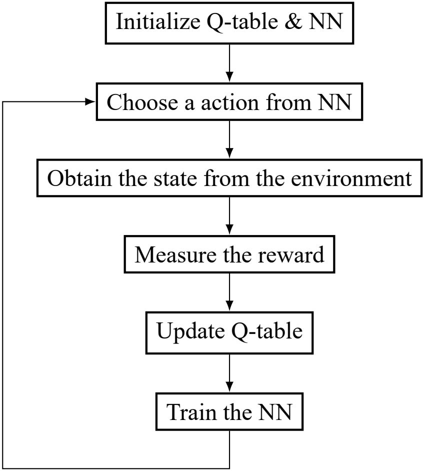


Figure 10: Q-learning flow chart using NN

Expectation

1. Realize on the inverse pendulum system.
2. Realize on the fix wing UAV.
3. Find more different algorithm.

Q&A

References

[1] W. Koch, "Flight controller synthesis via deep reinforcement learning," *CoRR*, vol. abs/1909.06493, 2019.

[2] W. Koch, R. Mancuso, R. West, and A. Bestavros, "Reinforcement learning for UAV attitude control," *CoRR*, vol. abs/1804.04154, 2018.