A Study of Reinforcement Learning for Self-driving RC Car using AWS DeepRacer and Unity ML-agent

Team: Timesquare

Supervisor: Young-Keun Kim

Doyeon Kim Jooho Kim Yechan Song

Introduction

- Study of Reinforcement Learning(PPO) algorithm for autonomous driving
- Design and train PPO model in simulation environment
- Implementation of PPO in real-environment on small-scaled RC car

aws

- Simulation environment provided
- Hard to customize environment
- Applies on DeepRacer car only

unity

- Can create customized environment
- Can be applied to various RC car platform
- Hard to implement on embedded processor

Theory

1. PPO

- The most popular reinforcement learning algorithms
- Easy implementation and high performance
- High data efficiency

2. Find optimal parameter θ

$$\theta \leftarrow \theta + \nabla_\theta \sum_{i=t-N+1}^{t} \frac{p_\theta(s_t, a_t)}{p_{\theta_{old}}(s_t, a_t)}$$

$$constraint: r(\theta) = \frac{p_\theta}{p_{\theta_{old}}} < \varepsilon$$

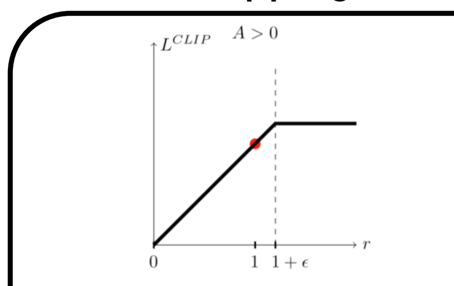
3. GAE

$$J_i = \sum_{i=t-N+1}^{t} \frac{p_\theta(s_t, a_t)}{p_{\theta_{old}}(s_t, a_t)} A_i$$

$$A_i \triangleq Q(a_i \mid s_i) - V(s_i) \approx \sum_{k=i}^{t} (\gamma\lambda)^{k-i} \delta_k$$

$$\delta_k = R_{k+1} + \gamma V(s_{k+1}) - V(s_k)$$

4.Clipping



$$J_i^{clip} \triangleq \min[ratio_i, A_i, clip(r_i 1-\varepsilon, 1+\varepsilon) A_i]$$

5. PPO Update Algorithm

0. $Initialize\ \theta, w$

$Repeat\ 1\sim4$

1. $Collect\ N\ Sample\ (sample: \{s_i, a_i, s_{i+1}\})$

$Repeat\ 2\sim3(Epoch)$

2. $Actor\ update: \theta \leftarrow \theta + \alpha \nabla_\theta \sum_{i=t-N+1}^{t} J_i^{clip}$
3. $Critic\ update: w \leftarrow w - \beta \nabla_\theta \sum_{i=t-N+1}^{t} (A_i^{GAE})^2$
4. $Clear\ the\ batch$

Part1 – AWS DeepRacer

Reward Function

Reward Function

Initializing & Updating Paramters

$H = car\ heading\ angle$
$\theta_s = steering\ angle\ of\ car$
$x_t, x_y = target\ point$
$x_{car}, y_{car} = car\ point$
$eps = error\ disired\ by\ the\ user$
$dx = x_t - x_{car},\ \ dy = y_t - y_{car}$
$\theta_t = polar(dx, dy)$
$\theta_{best\ angle} = \theta_t - H$

reward
$if(\theta_{best\ angle} - \theta_s < eps)$
get *reward*

Fig 1. Reward Function

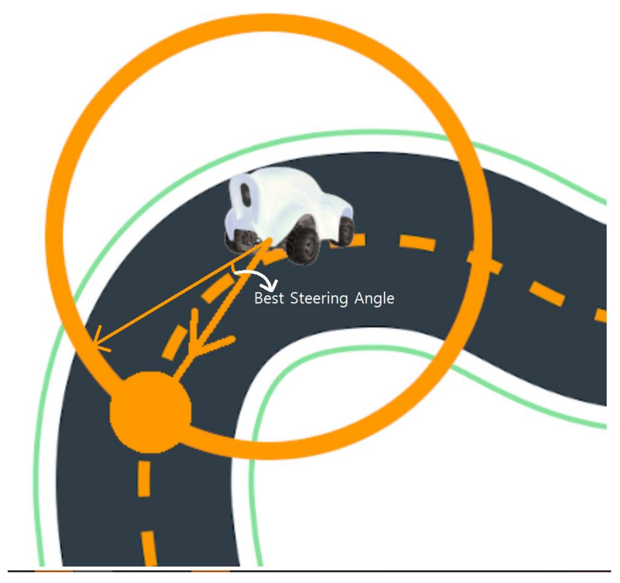


Fig 2. Schematic of Reward Function

Simulation

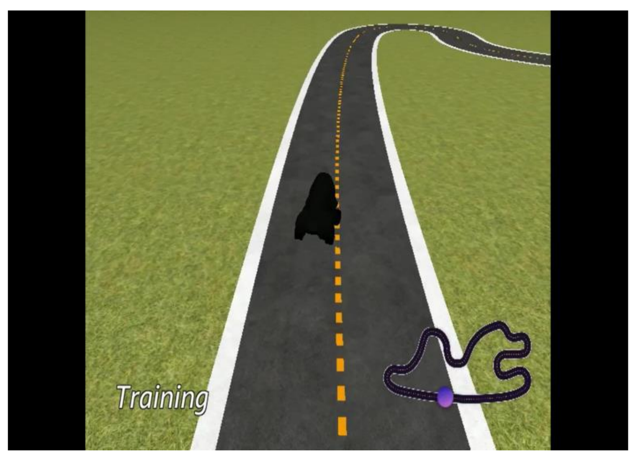


Fig 3. Simulation Program of AWS

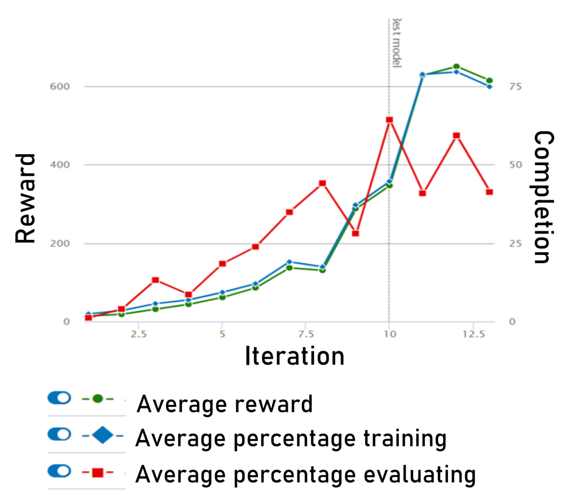


Fig 4. Result Graph of Simulation

Implementation

CPU : Intel atom Processor

Camera : 4MP(2688x1520)

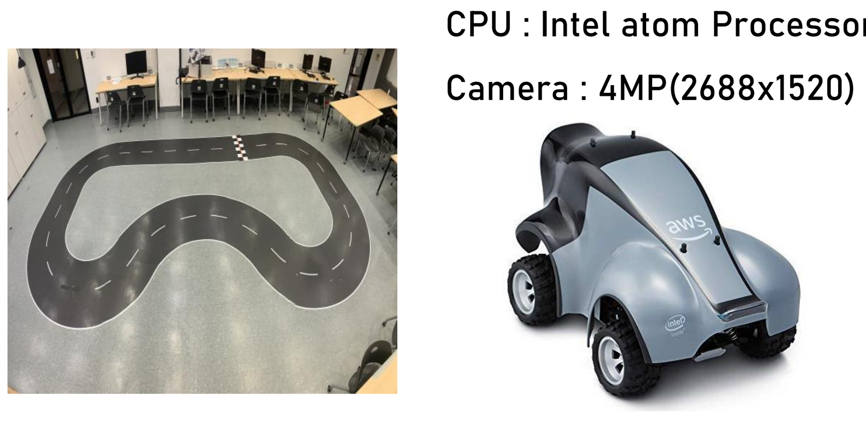


Fig 5. Track and AWS DeepRacer

Fig 6. Implementation of Driving

Part2 – Unity-ML agent

Reward & Hyperparameter

Episode end condition:

- When the agent leaves the lane
- When average reward per 10,000 steps is over 20

Reward:

- Increase by 0.01 per step if driving within lanes

Main Hyperparameters:

- beta: 0.005
- Number of hidden layers: 128
- Learning rate: 0.0003

Image Processing

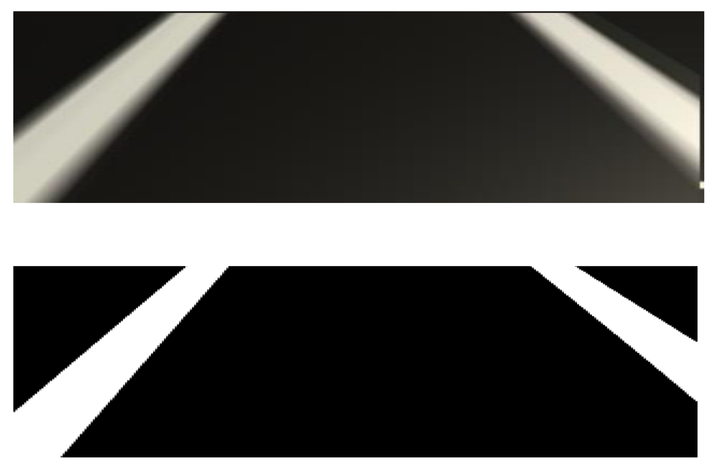


Fig 7. Using Thresholding to Make Binary Image

Simulation

Fig 8. Unity Track and Car

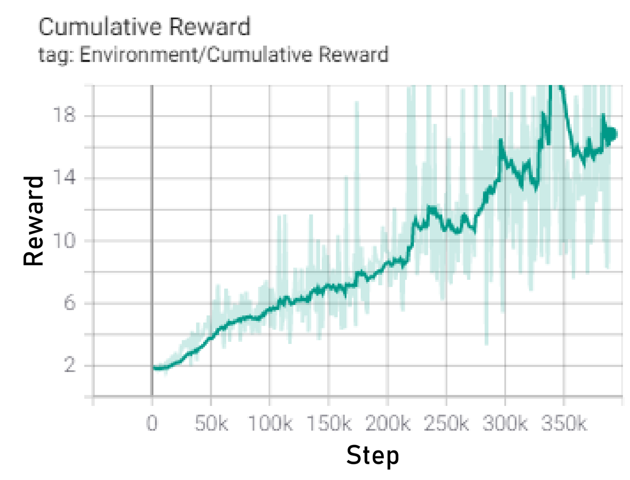


Fig 9. Result of Simulation

Future Plan

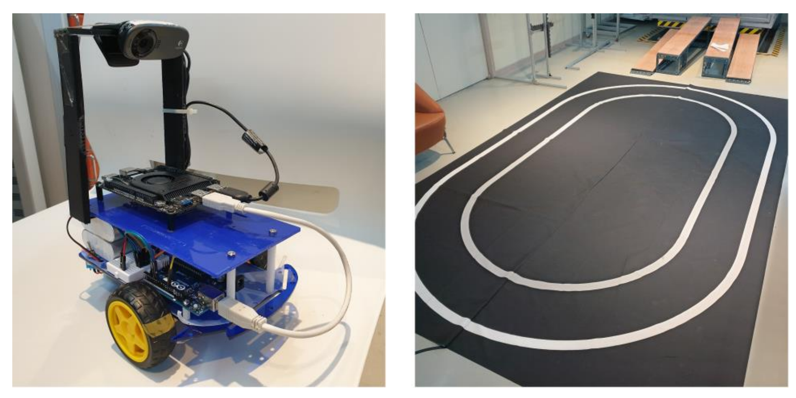


Fig 10. Track and RC car

- Implement on embedded processor
- Test driving on RC car track

Conclusion

- Studied reinforcement learning algorithm (PPO) for a simple autonomous driving
- Used AWS DeepRacer and Unity for training agent in Simulation Environment
- Deployed RL model on DeepRacer RC Car for successful driving
- Need to implement on our RC Car for future plane

References

[1] Bharathan Balaji, Sunil Mallya, Sahika Genc, SaurabhGupta, Leo Dirac, Vineet Khare, Gourav Roy, Tao Sun, Yun zhe Tao, Brian Townsend, Eddie Calleja, Sunil Muralidhara,and Dhanasekar Karuppasamy. Deepracer: Educational au tonomous racing platform for experimentation with sim2realreinforcement learning. CoRR, abs/1911.01562, 2019.

[2] John Schulman, Philipp Moritz, Sergey Levine, Michael Jor dan, and Pieter Abbeel. High-dimensional continuous con trol using generalized advantage estimation. arXiv preprintarXiv:1506.02438, 2015.

[3] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Rad ford, and Oleg Klimov. Proximal policy optimization algo rithms. arXiv preprint arXiv:1707.06347, 2017.