Learning Efficient End-to-end Point Goal Navigation

By Yuhang Song

[This Note is recorded from Nov 2022 to ...]

Introduction

Recent development of Reinforcement Learning algorithms shows that it can handle tasks with upper human level, it is now used frequently in many fileds for generating policies in complex environment. However, one of the most significant problem with Reinforcement Learning is the efficiency, well performing models typically requires gaint amount of training, such requirements not only becomes more concered in simple inefficient RL methods, it also brings large difficults to scenarios that potentially needs autonomous policy generation but interact with physical environment (e.g. robotics, embodied AI).

Navigation is the most common and foundameental application field for robotics, due to the powerful representation and expirence learning traditionnal ability of Deep Reinforcement Learning methods, it has been a trend of applying DRL based method on solving robotics navigation problems as a replacement of traditional navigation.

In this work, I will be focusing on applying Deep Reinforcement Learning based methods to navigation tasks, in paticular, I will focus on In-door navigation PointNav task, learning navigation behaviors end-to-end with DRL model with acceptable performance, as well as improve its training efficiency.

Background

Reinforcement Learning

1.1 Actor-Critic Reinforcement Learning

Reinforcement Learning learns a policy from environmental experiences, it gains reward signals from environment and accordingly adjust the model to maximize the expected rewards and thus formulate a policy. Value-based Reinforcement Learning define value functions that evaluate the states and actions of a markov decision process, the optimal policy is typically selected from greedy policy. Whereas the Policy-based Reinforcement Learning algorithms model the policy explicitly and optimize the policy by methods like gradient ascent.

Another group of algorithm integrates the advantages of both Value-based methods and Policy-based methods, namely Actor-Critic architectures, in Actor-Critic, we define a critic:

$$ Q_w(s,a) \approx Q^{\pi_\theta}(s,a) $$

Where the critic approximates state-action value function $Q(s,a)$, the actor approximates the policy π, there are parameterized by w and θ respectively.

Actor-critic algorithm follow an approximate policy gradient, the actor network can be updated by:

$$ \nabla_\theta J(\theta) \approx \mathbb{E}{/|pi\}{\theta}[\nabla_\theta log \pi_\theta(s,a) Q_w(s,a)] \

\nabla \theta = \alpha \nabla_\theta log \pi_\theta(s,a) Q_w(s,a) \

\theta_{t+1} \leftarrow \theta_t + \lambda \nabla\theta_t $$

The critic network is based on critic functions, here use Q-function as an example, the critic network can be updated as:

$$ \nabla_w J(w) \approx \mathbb{E}*{|pi\theta}[MSE(Q_{t},R_{t+1}+max_a Q_{t+1})] \

\nabla w = MSE(Q_{t},R_{t+1}+max_a Q_{t+1}) \

w_{t+1} \leftarrow w_t + \lambda \nabla w_t $$

Instead of leting critic to estimate state-value function, we can allow it to alternatively estimates Advantage function $A(s,a) = Q_w(s,a) - V_v(s)$ to reduce the variance. There are many alternative critic function choices.

1.2 Proximal Policy Optimization with AC-based Advantage Function

Baseline algorithm of OpenAI. PPO allows off-policy learning to policy gradient algorithm. In policy gradient, we update our policy network by compute gradient of the expected reward function with respect to policy parameters θ, and perform gradient acsent to maximize it: $$ \nabla J(\theta) = \frac{1}{N} \sum_{n=1}^N \sum_{t=1}^{T_n} R(\tau^n) \ \nabla_\theta \ log \ \pi_\theta(a_t^n | s_t^n) \

= \mathbb{E}*{|tau |backsim |pi\theta}[R(\tau^n) \ \nabla_\theta \ log \ \pi_\theta(a_t^n | s_t^n)] \

\theta \leftarrow \theta + \alpha \nabla \bar{R}|*theta $$ In PPO algorithm, instead of sampling trajectories from policy $|pi\theta$, in order to increase sample efficiency(reuse expirence), we sample trajectories from another policy $\pi_{\theta'}$ and apply a importance sampling method to correct the difference. $$ J_{\theta'}(\theta) = \mathbb{E}*{|tau |backsim |pi{\theta'}}[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta'}(a_t|s_t)} R(\tau^n)] \

\nabla J_{\theta'}(\theta) = \mathbb{E}*{|tau |backsim |pi{\theta'}}[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta'}(a_t|s_t)} R(\tau^n) \ \nabla_\theta \ log \ \pi_\theta(a_t^n | s_t^n)] $$

The reward $R(\tau^n)$ can be replaced by advantage function $A^{\theta'} (s_t,a_t)$, where we levarage the power of AC architecture the gain more suitable representation of the loss to optimize.

$$ J_{\theta'}(\theta) = \mathbb{E}*{|tau |backsim |pi{\theta'}}[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta'}(a_t|s_t)} A^{\theta'} (s_t,a_t)] \

\nabla J_{\theta'}(\theta) = \mathbb{E}*{|tau |backsim |pi{\theta'}}[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta'}(a_t|s_t)} A^{\theta'} (s_t,a_t) \ \nabla_\theta \ log \ \pi_\theta(a_t^n | s_t^n)] $$

1.2.1 PPO KL Regularzation

In addition, we need to add a regularzation term (or in TRPO, add a constrain) to the objective function to constrain the difference between two distributions, therefore the objective function becomes: $$ J(\theta) = J_{\theta'}(\theta) - \beta KL(\theta,\theta') $$ Where adaptively set the value of β, specificlly when $KL(\theta,\theta') > KL_{max}$, increase β; when $KL(\theta,\theta') < KL_{min}$, decrease β.

1.2.1 PPO Clip Method

$$ J_{clip}(\theta) \approx \sum_{s_t,a_t} min(\frac{p_\theta(a_t | s_t)}{p_{\theta'}(a_t|s_t)} A^{\theta'}(s_t,a_t), \ clip(\frac{p_\theta(a_t | s_t)}{p_{\theta'}(a_t|s_t)}, 1 - \epsilon , 1 + \epsilon) \ A^{\theta'}(s_t,a_t)) $$

Where $clip$ = $1-\epsilon$ if $\frac{p_\theta(a_t | s_t)}{p_{\theta'}(a_t|s_t)} < 1 - \epsilon$; $clip$ = $1+\epsilon$ if $\frac{p_\theta(a_t | s_t)}{p_{\theta'}(a_t|s_t)} > 1 + \epsilon$; values that fall anywhere in between $clip = \frac{p_\theta(a_t | s_t)}{p_{\theta'}(a_t|s_t)}$. This methods hence set an constrain that making sure the differences between two policy distributions changes within certain range of ϵ.

Robotics Navigation

2.1 Traditional Navigation

Navigaiton tasks for robots are defined as robots moving from a start location to a target location, figure 1 shows the traditional pipeline for navigation tasks, it make use of Simultaneous Localization and Mapping (SLAM) to generate a map for current environment, and based on it to perform gloabal mapping to guilde the global path for agent navigating to the target, integrates with local path planning for obstacle avoidance.

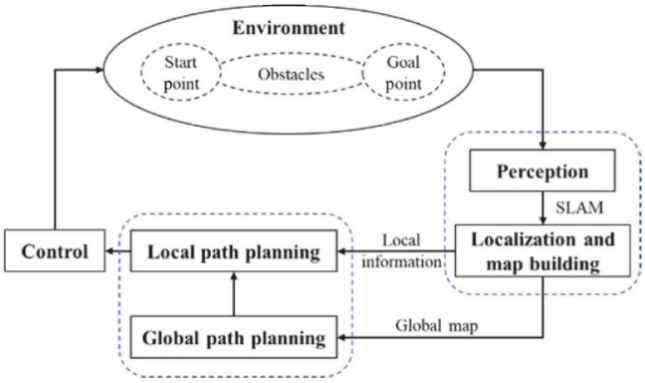

Figure 1. Traditional Navigation

However, each of the componentt above is a challenging research topic, empirially, we know that errors accumulated in each of tthe partitions will result in a significantt problem, hence lead to poor performance. In addition, the traditional navigation framework relies on a high-precision global map that is very sensitive to sensor noise, resulting in limitations in the ability to manage an unknown or dynamic environment.

2.1 Deep Reinforcement Learning Based Navigation

Unlike traditional navigation method, because of the strong representation power provided by Deep Neural Networks and policy learning ability of Reinforcementt Learning, attentions were being put into the field of directly using Reinforcement Learninig methods to learn navigation tasks end to end.

As shown in figure 2, the DRL agent replaces the localization and map building module as well as the local path planning module of the traditional navigation framework, moving toward the target point while avoiding static, dynamic, and simple structurally continuous obstacles. However, in an environment where structurally continuous obstacles are too complex, the agent may fall into a local trap. In this case, DRL requires additional Fig. 3 DRL-based navigation system. global information provided by the traditional navigation technique. As shown in Fig. 2, the global path planning module generates a series of waypoints as intermediate goal points for DRL-based navigation, which enables the integrated navigation system to realize long-distance navigation in a complex structural environment.

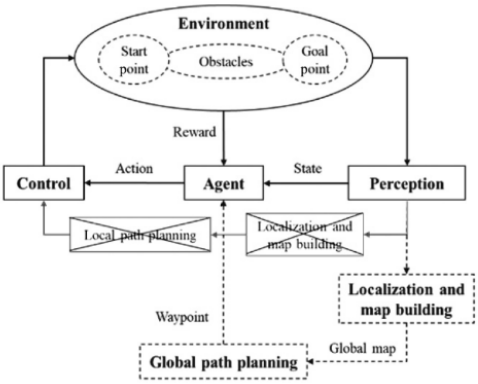

Figure 2. Traditional Navigation

Related Works

To be filled.

Expriments Setting Up

3 Point Goal Navigation

The Point Goal Navigation task defines an agent that is initialized at a random starting position and orientation in a new environment and asked to navigate to target coordinates specified relative to agents position.

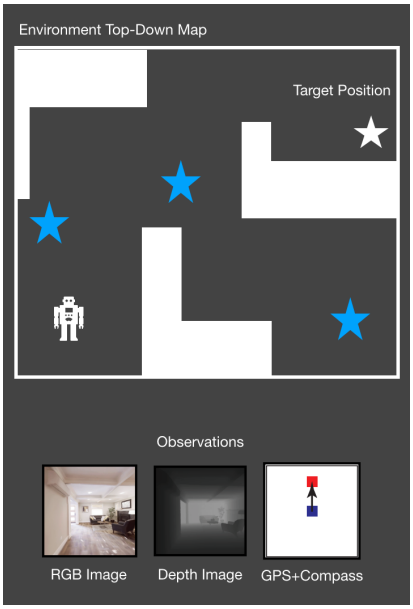

Figure 3. Point Navigation Task

No map is available (hence no global planner) and the agent must navigate using only its sensors – in our case RGB-D (or RGB) and GPS+Compass (providing current position and orientation relative to start). As shown in figure 3. The agent has 4 discrete actions to choose from, {move_forward(0.25 meters), turn_left(10 degree), turn_right(10 degree), stop}. In this work, we use SPL(Success per Legth) as our metric to evaluate thee performance, formally: $$ SPL = S \frac{l}{max(l,p)} $$ Where S is a binary indicator of whether agent reaches desired point; l be the length of shortest path between start and target; p is the length of thee path agentactually took.

References

[0] K. Zhu and T. Zhang, "Deep reinforcement learning based mobile robot navigation: A review," in Tsinghua Science and Technology, vol. 26, no. 5, pp. 674-691, Oct. 2021, doi: 10.26599/TST.2021.9010012.

[1] Tan, M., & Le, Q. (2021, July). Efficientnetv2: Smaller models and faster training. In International Conference on Machine Learning (pp. 10096-10106). PMLR.

[2] Tan, M., Chen, B., Pang, R., Vasudevan, V., Sandler, M., Howard, A., & Le, Q. V. (2019). Mnasnet: Platform-aware neural architecture search for mobile. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 2820-2828).

[3] Brock, A., De, S., Smith, S. L., & Simonyan, K. (2021, July). High-performance large-scale image recognition without normalization. In International Conference on Machine Learning (pp. 1059-1071). PMLR.

[0] K. Zhu and T. Zhang, "Deep reinforcement learning based mobile robot navigation: A review," in Tsinghua Science and Technology, vol. 26, no. 5, pp. 674-691, Oct. 2021, doi: 10.26599/TST.2021.9010012.

[1] Tan, M., & Le, Q. (2021, July). Efficientnetv2: Smaller models and faster training. In International Conference on Machine Learning (pp. 10096-10106). PMLR.

[2] Tan, M., Chen, B., Pang, R., Vasudevan, V., Sandler, M., Howard, A., & Le, Q. V. (2019). Mnasnet: Platform-aware neural architecture search for mobile. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 2820-2828).

[3] Brock, A., De, S., Smith, S. L., & Simonyan, K. (2021, July). High-performance large-scale image recognition without normalization. In International Conference on Machine Learning (pp. 1059-1071). PMLR.