Instruction for Contest Week

- RC car racing competition (simulation environment)-

Introduction

The objective of this semester's term project is to implement a path planning algorithm for a RC car in environments with various difficulties. Basically, you can implement the path planning algorithm based on behavioral cloning (Project 2), but you are allowed to implement additional controller or other reinforcement learning algorithms such as DQN. Your algorithm will be evaluated in 10 unknown maps. The performance will be on the public leaderboard of project web page.

Fine-tuning algorithms

Using Gaussian Process Regression (GPR) for behavior cloning, you can fine-tune the algorithm by changing the initialization parameters of GPR. Assuming the data pairs $\tau^* = (s_0^*, a_0^*, s_1^*, a_1^*, \dots)$ follow gaussian process, the prior mean is assumed to be constant zero and the prior variance is specified by passing a kernel object. As you call **sklearn.gaussian_process.GaussianProcessRegressor.fit()**, the hyperparameters of the kernel will be optimized during fitting the model into data. You can use different kernel functions to train the GPR model.

You can also adjust the noise level α or optimizer to enhance your model's performance. To find more information, look at scikit-learn/gaussian_processes.

Alternative learning algorithms

If you want to train your agent using other deep learning or reinforcement learning algorithms, implement your own network in the Agent class. You can use additional controller or other reinforcement learning algorithms such as Deep Q-Network (DQN). Please use Pytorch for training because the RCCar uses PyTorch version 1.9.0 and CUDA version 10.2 Training in TA's simulator or RCCar is not allowed. Please refer to the PyTorch instruction manual for further details.

Environments

Training maps are not provided, so generate your own maps using **map_generator.py** for training. Evaluation maps will have different difficulty levels. In this project, your model will also be tested in more difficult environments than the provided maps.

Evaluation

Your algorithm will be evaluated in 10 unknown maps. You can see the performance and score of your model on web page's leaderboard. If you submit query following the instruction below, your model will be automatically tested in evaluation environments. After the evaluation, your contest ranking and score will be uploaded on the leaderboard. Your team's score will be calculated as the sum of 10 map scores. Final evaluation will be held by the latest submitted query until the due date of this assignment.

$$Total\ Score = (Map\ 1\ score) + \cdots + (Map\ 10\ score), (Each\ map\ score\ is\ in\ range\ 0\ to\ 10.)$$

For contest week, you should follow the steps below.

1) **Change the TEAM_NAME on the top of RLLAB_project3.py to your team's name. Otherwise, we can't grade your code**
2) Fine-tune GPR from project 2 or implement your new algorithm. Follow the commented descriptions in the code.
3) Evaluate your model. You can collect more data in various maps to obtain higher performance of your model.

Good Luck~

Submission

During contest week, submit your code via web page. To prevent long waiting times, we highly recommend you to **add a new query at least 30 minutes after your latest query**. To evaluate in your local PC, use eval_agent.py. Make sure to git pull the latest commit on the Intelligent-Systems-Project repository. To submit your code, commit your codes on github and go to our project webpage. After you sign in, send a query in Query page. Also, you can check your rank in LeaderBoard and past results in MyPage.

Due to: 2022.06.10. 23:59 KST