Teach a Quadcopter How to Fly

Meets Specifications

Awesome

You did a great job.

Suggestion

I recommend checking the Google Python Style Guide, there are great tips about how to improve coding, in general:
https://google.github.io/styleguide/pyguide.html
About this specific project there are some tips here: https://github.com/WittmannF/quadcopter-best-practices
There is a paper "Control of a Quadrotor with Reinforcement Learning": https://arxiv.org/abs/1707.05110
There is also some videos about how to build and program a Quadcopter: https://www.youtube.com/watch?v=g7SjuTzH44c&list=PLQVvvaa0QuDfp4GmCdhbPhPwWt4O_TBmB

Define the Task, Define the Agent, and Train Your Agent!

The `agent.py` file contains a functional implementation of a reinforcement learning algorithm.

Awesome

You did a great job in the DDPG (Deep Deterministic Policy Gradients).
There is an article about it here: http://pemami4911.github.io/blog/2016/08/21/ddpg-rl.html

The `Quadcopter_Project.ipynb` notebook includes code to train the agent.

Awesome

Your code in the notebook was concise. It is great to use the files outside the notebook for the hard coding so the notebook is clean.

Plot the Rewards

A plot of rewards per episode is used to illustrate how the agent learns over time.

Awesome

The reward per episode is really good. Your algorithm performed very well.

Reflections

The submission describes the task and reward function, and the description lines up with the implementation in `task.py`. It is clear how the reward function can be used to guide the agent to accomplish the task.

Awesome

Great description about the task and reward.
It is critical to avoid the vanishing gradient problem and you did it very well. A scaling here is a alternative to help in the learning.
A good video discussing reward is here: https://www.youtube.com/watch?v=0R3PnJEisqk
Here there is a way to deal with vanishing gradient: https://www.dlology.com/blog/how-to-deal-with-vanishingexploding-gradients-in-keras/

The submission provides a detailed description of the agent in `agent.py`.

Awesome

Very good description of the DDPG and the challenges you have faced.

The submission discusses the rewards plot. Ideally, the plot shows that the agent has learned (with episode rewards that are gradually increasing). If not, the submission describes in detail various attempted settings (hyperparameters and architectures, etc) that were tested to teach the agent.

Awesome

You got a great performance. It is interesting to notice the importance of the reward function in the final performance.

A brief overall summary of the experience working on the project is provided, with ideas for further improving the project.

Awesome

It is great to know your experience and challenges.
In a previous version of this project you had an simulator to test the quadcopter but to set up it was had an time consuming. If you are interested, the Robotics Nanodegree has a lot of challenges like this one.

DOWNLOAD PROJECT

RETURN TO PATH