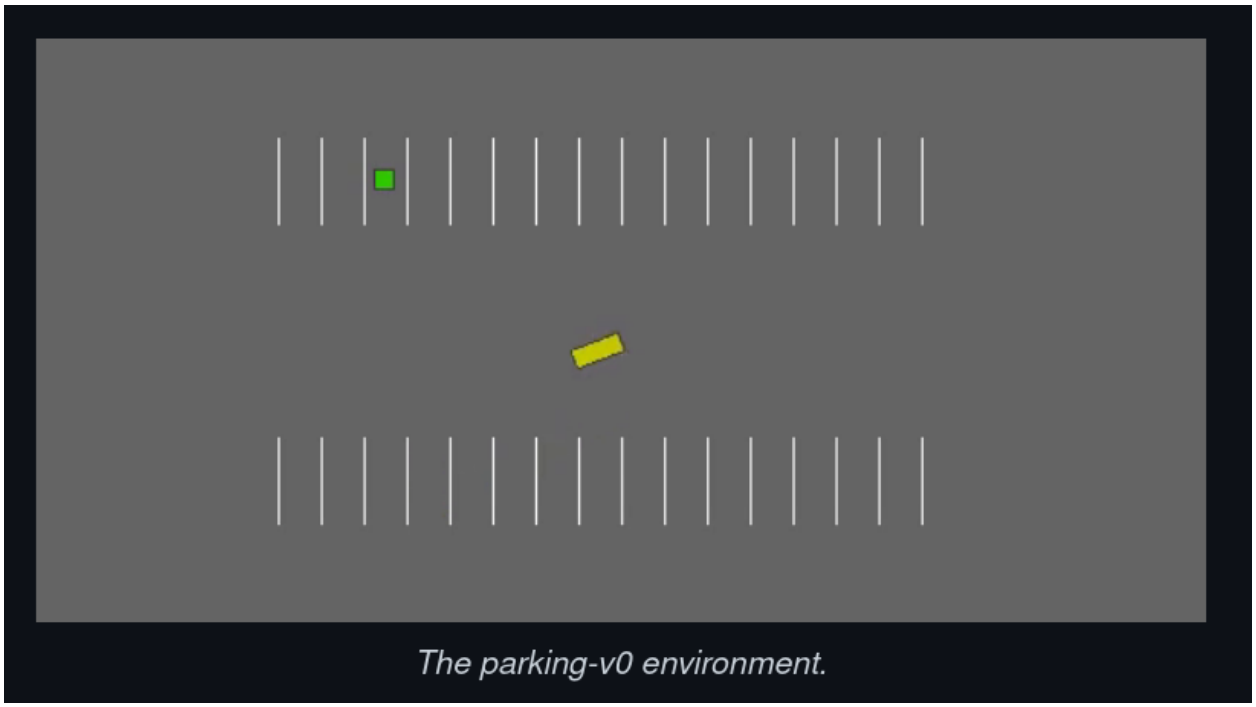
Entrega Intermediária do Projeto Final - Agentes Autônomos e Reinforcement Learning

Aluno: Antonio Fuziy
Prof: Fabrício Barth

Highway-env Auto-Parking:

Nesse ambiente a visão da implementação 2D olhando de cima do carro e seu objetivo é estacionar o carro sozinho em uma vaga determinada pelo ambiente, o agente a ser treinado é o carro, podendo se movimentar para frente, trás, além de poder se movimentar em vários ângulos para direita e para esquerda, segue uma foto do ambiente abaixo para ficar mais claro:



The parking-v0 environment.

O carro se identifica como o retângulo em amarelo e a vaga que ele deve estacionar se apresenta em verde. As recompensas durante o treino podem ser de acordo com as posições onde o carro parou, quanto mais perto da vaga o agente parar maior a recompensa, mas dependendo da posição que ele parar a recompensa pode ser negativa ou positiva. Existem alguns projetos já implementados em produção que utilizam algoritmos para estacionar carros

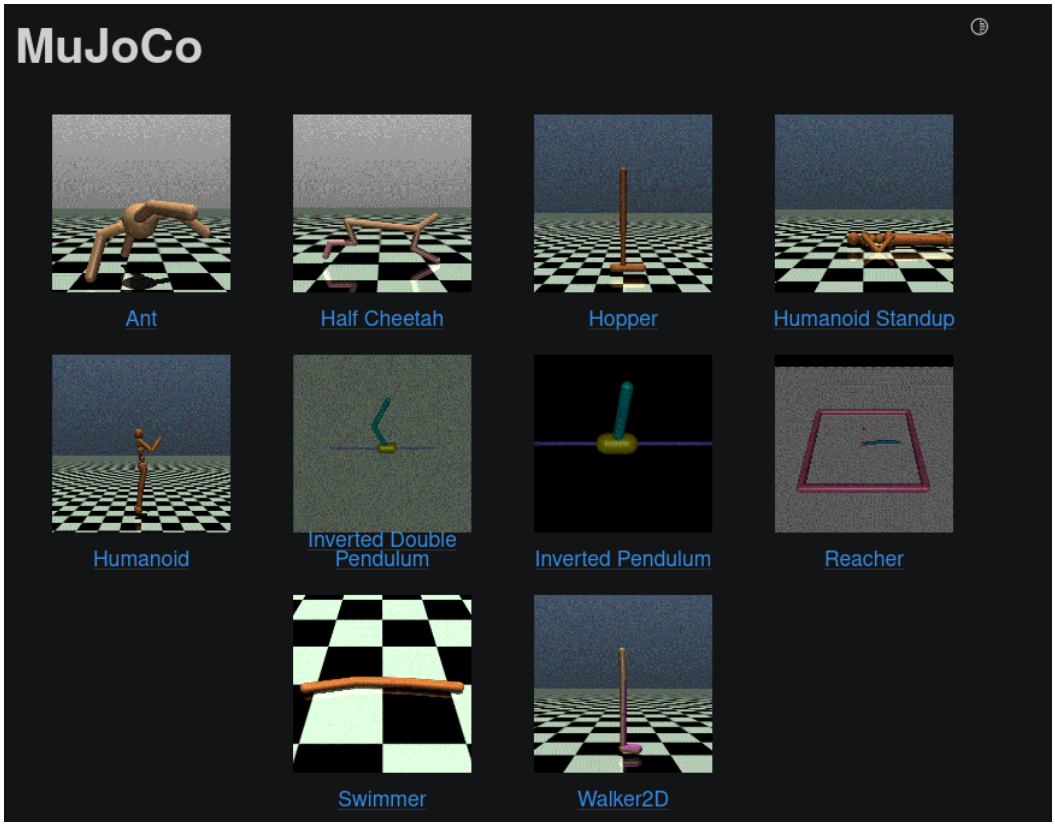
de forma autônoma, alguns projetos utilizam apenas sensores e visão computacional para estacionar o carro, porém outros usam de aprendizado de máquina para esse problema, um exemplo prático pode ser o auto-parking da Jaguar citado na referência 1. Para implementação desse problema existem alguns ambientes do Gym OpenAI que fornecem as ferramentas necessárias para resolução do problema, assim como citado na referência 2 das referências.

Referências:

1. Jaguar autonomous-parking: https://www.jaguar.com/incontrol-global/incontrol/driver-assistance/park-assist.html
2. Github: https://github.com/eleurent/highway-env
3. Rewards highway-env: https://highway-env.readthedocs.io/en/latest/rewards/index.html
4. Possible Agent Actions: https://highway-env.readthedocs.io/en/latest/actions/index.html#continuous-actions

Multi-Joint dynamics with Contact (MuJoCo):

MuJoCo é uma engine para otimizar pesquisas em robótica, animações e biomecânicas sendo necessária que a engine do MuJoCo seja instalada para uso. Para esse projeto existem vários ambientes, mas usaremos de exemplo o HumanoidStandup, o qual utiliza um modelo 3D de um humanoide, esse é o agente a ser treinado, portanto o seu objetivo é levantar o humanoide e fazer ele se equilibrar em pé aplicando-se vários torques em suas articulações, além de ser possível aplicar forças externas ao agente, as possíveis ações são de aplicar torques em várias partes do corpo nos eixos x, y e z, dentre as partes do corpo estão abdômen, quadril, joelho e vários outros. Segue uma imagem abaixo que mostra bem o ambiente:



Dentre as recompensas do agente estão o quanto o agente conseguiu levantar o humanoide, além de penalizações quando as forças de controle e os torques aplicados ao humanoide são muito altos ou quando forças externas aplicadas ao agente são muito altas fazendo o humanoide não conseguir se equilibrar. Nesse ambiente o uso de reinforcement learning pode ser mais efetivo do que

Referências:

1. HumanoidStandup, actions, env and rewards: https://www.gymlibrary.ml/environments/mujoco/humanoid_standup/

Anytrading

Esse cenário é utilizado para construção de bots de trading através de reinforcement learning utilizando três ambientes oferecidos pela OpenAI Gym, dentre eles estão ForexEnv, StocksEnv e TradingEnv, ambos citados na referência 1. O agente desse ambiente é o bot e para que ele funcione da forma correta suas ações podem ser vender, comprar ou continuar com ação, nesse caso, a função de recompensa deve ser criada pelo desenvolvedor do projeto uma vez

que se trata de um bot de trading, para isso o gym fornece um método abstrato para que o desenvolvedor crie sua função de recompensa. Atualmente existem múltiplos projetos de bots de trading utilizando AI, machine learning, web scrapping e outros métodos, porém existem alguns projetos utilizando reinforcement learning com esse mesmo ambiente, assim como o citado na referência 2.

Referências:

1. AnyTrading environment, github repository and documentation: https://github.com/AminHP/gym-anytrading
2. Trading bot using AnyTrading Gym: https://github.com/cove9988/TradingGym

Para o projeto final o ambiente de minha escolha seria o Highway-env Auto-Parking, uma vez que o ambiente construído é o mais completo e com mais documentação, além de ter uma utilidade grande para aplicações no mundo real em sistemas de carros autônomos, dessa forma acredito que o melhor primeiramente seria utilizar um algoritmo de Q-Learning ou Deep Q-Learning para treinar o agente de forma que o treinamento através do Q-Learning utilizando uma Q-table pode ser mais rápido, portanto seria mais fácil chegar a um bom progresso do projeto, e após isso utilizar uma rede neural com keras da biblioteca tensorflow a fim de desenvolver um algoritmo de Deep Q-Learning para o agente, isso seria feito assim que o ambiente estiver funcionando com o agente utilizando o Q-Learning, pois o treinamento com uma CNN tomaria mais tempo do projeto.