XX Seminário de Pesquisas em Engenharia Mecânica
www.uff.br/petmec

1/2021

ANÁLISE DO ALGORITMO PROXIMAL POLICY OPTIMIZATION NA SIMULAÇÃO DE VELEIROS AUTÔNOMOS[1]

Rodrigo Picinini Méxas
Engenharia Mecânica
Email: rodrigopicinini@id.uff.br

Resumo: Este projeto visa o estudo de desempenho do algoritmo de aprendizado de máquina por reforço, cujo nome é *Proximal Policy Optimization* [1], na simulação de veleiros autônomos e sua resposta às diferentes direções do vento enquanto desvia de obstáculos detectados por análise de imagens. Utiliza-se para o desenvolvimento a plataforma Unity e o toolkit de aprendizado de máquina ML-Agents. A metodologia que guia o projeto pode ser aplicada de forma similar para outros problemas de aprendizado por reforço. Por meio do treinamento do agente é possível confirmar os resultados esperados pelo algoritmo.

Palavras-chave: veleiros autônomos, Unity, PPO.

1. INTRODUÇÃO

O estudo de veículos autônomos não é uma prática inédita. Desde o século passado, quando estudos sobre inteligência artificial foram iniciados, muitos entusiastas já se interessavam pelo assunto. Entretanto, devido ao baixo poder computacional, pouco podia ser feito para que os algoritmos desenvolvidos pudessem trazer resultados satisfatórios. Com o passar do tempo, novos algoritmos foram sendo desenvolvidos e o poder computacional foi aprimorado, tornando realidade o uso de técnicas de inteligência artificial em simulações que antes não eram possíveis.

A pesquisa tem como objetivo principal simular o avanço de um barco veleiro autônomo em dois ambientes distintos, utilizando-se técnicas de aprendizado de máquina, mais especificamente, aprendizado por reforço. Este aprendizado ocorre de forma que não é dito quais ações o aprendiz deve tomar e o mesmo deve descobrir quais ações levam a maior recompensa [2]. O objetivo específico pode ser descrito como a experimentação do algoritmo *Proximal Policy Optimization* nos cenários de vento favorável e vento de través com o desvio de obstáculos estáticos e móveis.

2. CONCEITOS

2.1. Barcos autônomos

Barcos autônomos são barcos que aprendem a navegar em um determinado ambiente sem que haja a necessidade de um ser humano controlá-lo. Também são conhecidos como *Unmanned Surface Vehicle* (USV), traduzido como Veículo de Superfície Não-Tribulado. Esta denominação é comumente encontrada na literatura.

Seu aprendizado normalmente se dá por meio de aprendizado por reforço. Por meio de muitas tentativas e erros, o veículo aprende como se locomover. E para obter informações do ambiente, normalmente são utilizadas câmeras e imagens como input, mas também podem ser utilizados outros tipos de sensores.

[1] Trabalho desenvolvido a partir de resultados parciais obtidos ao longo da disciplina Projeto de Graduação em Engenharia Mecânica III, esta, sob a orientação da Prof. Fabiana R. Leta. e do Prof. Esteban Clua.

2.2. PPO - Proximal Policy Optimzation

PPO, traduzido como otimização de política proximal, é um método baseado de aprendizado por reforço pela instituição Open AI. Sua ideia é evitar que uma atualização de política seja muito expressiva e cause distúrbios e erros no cálculo do modelo. Política é a estratégia utilizada para a tomada de ações [3].

Para isso utiliza um método chamado de grampear, do inglês *clip*, para evitar que a atualização da política se desvie muito da anterior. Se os valores são menores ou maiores que os extremos determinados, estes ganham o valor do extremo ultrapassado.

2.3. ML-Agents

O *Unity Machine Learning Agents Toolkit*, também conhecido como ML-Agents, é um *toolkit* de código aberto que implementa soluções de aprendizado de máquina ao Unity por meio de uma API em Python, com implementações em PyTorch, que é uma biblioteca de aprendizado de máquina. O MLAgents traz efetividade e inovação como plataforma de desenvolvimento em inteligência artificial [4]. Além do PPO, há outros algoritmos presentes no *toolkit*, como o *Soft Actor Critic* [5]. Além disso, possibilita o aprendizado por imitação, que permitem que algum humano realize previamente o que deve ser aprendido pela inteligência artificial e a forneça como entrada, o que normalmente acelera o aprendizado do modelo. E o *toolkit* também, permite o uso do aprendizado por currículo, que é utilizado para ensinar o agente em diferentes ambientes com gradualmente tornam-se mais complexos.

3. METODOLOGIA

A metodologia do projeto seguiu as etapas descritas nos subtópicos seguintes.

3.1. Revisão da bibliografia e estudo sobre o tema

Primeiramente fez-se um estudo sobre aprendizado de máquina e mais especificamente aprendizado por reforço. A partir daí foi possível a compreensão de tópicos gradativamente mais complexos, como o algoritmo estudado.

3.2. Definição das ferramentas utilizadas

Para que fosse possível desenvolver o projeto, decidiu-se as ferramentas que seriam utilizadas. Utilizou-se o Unity versão 2020.3.8f e a *release* 17 do pacote ML-Agents.

3.3. Configuração do ambiente de treinamento

Com o intuito de partir de uma simulação já implementada, e a partir dela fazer a implementação da inteligência artificial, utilizou-se um projeto de código aberto, disponível no GitHub [6]. Daí, montou-se dois cenários: obstáculos estáticos e obstáculos móveis, e fez-se alterações para a adequação ao estudo.

3.4. Definição de observações, atuadores e recompensas

Nesta etapa estabeleceu-se como o barco seria recompensado e punido durante o aprendizado, o que poderia visualizar e mover. A observação dos obstáculos se deu por análise de imagens e o veleiro pôde mover a vela e o leme, sendo recompensado por alcançar a chegada e punido por bater nos obstáculos, sair da área de treino e ir em direção contrária.

3.5. Treinamento

Fez-se o treinamento do veleiro nas situações de obstáculos estáticos e móveis. Ambos a favor do vento e com vento de través.

3.6. Análise de resultados

Os resultados obtidos foram analisados. Esta análise pode ser conferida na seção 4.

4. RESULTADOS

Aqui é possível visualizar as médias de recompensas obtidas ao longo do treinamento para os 4 casos estudados ao longo de 5 milhões de passo de tempo. Cada ponto nos gráficos é a média de 20.000 passos de tempo. Um passo de tempo representa a ação realizada. pelo veleiro. O valor máximo que pode ser obtido é 1.

4.1. Obstáculos estáticos com vento favorável

Seu gráfico de recompensas é representado pela Figura 1.

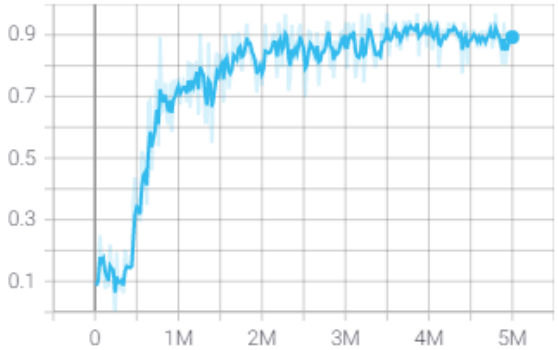

Figura 1. Gráfico de média de recompensas obtida no cenário de obstáculos estáticos com vento favorável. Fonte: Próprio autor (2021).

Última média de recompensas obtida: 0.9114. Tempo para treinamento: 4h10min24s.

4.2. Obstáculos estáticos com vento de través

Seu gráfico de recompensas é representado pela Figura 2.

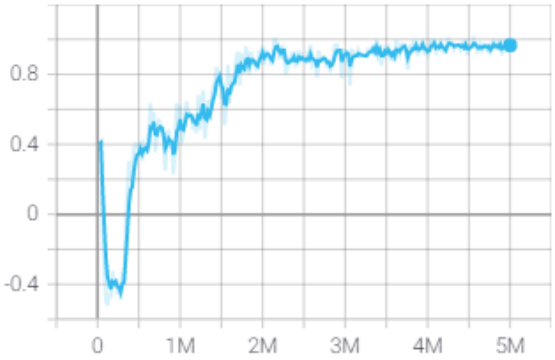

Figura 2. Gráfico de média de recompensas obtida no cenário de obstáculos estáticos com vento de través. Fonte: Próprio autor (2021).

Última média de recompensas obtida: 0.9712. Tempo para treinamento: 5h16min14s

4.3. Obstáculos móveis com vento favorável

Seu gráfico de recompensas é representado pela Figura 3.

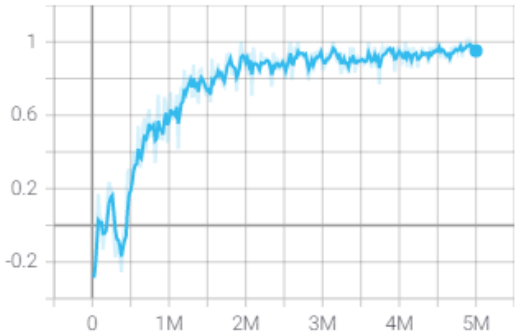

Figura 3. Gráfico de média de recompensas obtida no cenário de obstáculos móveis com vento favorável. Fonte: Próprio autor (2021).

Última média de recompensas obtida: 0.9434. Tempo para treinamento: 6h2min27s.

4.4. Obstáculos móveis com vento de través

Seu gráfico de recompensas é representado pela Figura 4.

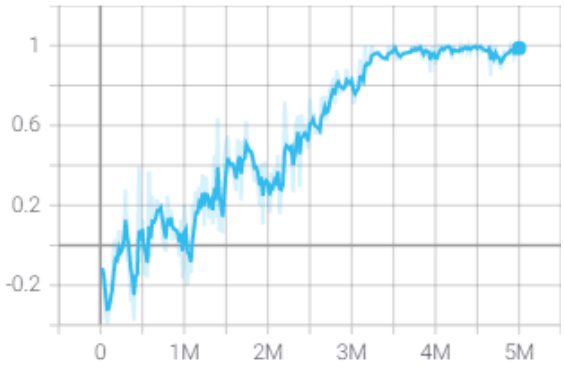

Figura 4. Gráfico de média de recompensas obtida no cenário de obstáculos móveis com vento de través. Fonte: Próprio autor (2021).

Última média de recompensas obtida: 1. Tempo para treinamento: 6h4min2s.

5. CONCLUSÕES

A presente pesquisa trouxe um estudo sobre o algoritmo Proximal Policy Optimization e sua aplicação na simulação de veleiros autônomos. O projeto foi desenvolvido na plataforma de desenvolvimento de jogos Unity e baseou-se em uma simulação de veleiros de código aberto. A partir desta simulação, modificações foram realizadas, 4 cenários diferentes foram montados e foi feita a implementação dos algoritmos de aprendizado de máquina, tendo o uso do pacote MLAgents, que permite a utilização de técnicas de inteligência artificial no Unity. Por fim, foi possível confirmar que o PPO por sua característica de limitar a atualização de sua política, conseguiu se adaptar a todos os cenários propostos.

REFERÊNCIAS

[1] Schulman, J. et al. (2017). Proximal policy optimization algorithms. arXiv.

[2] Sutton, R.; Barto, A. (2015). Reinforcement Learning: An Introduction, 2nd ed. in prog, The MIT Press, Cambridge, MA.

[3] Stapelberg, B. (2020). A survey of benchmarking frameworks for reinforcement learning. South African Computer Journal. vol 32, n 2, p. 258-292.

[4] Juliani, A. et al. (2018). Unity: A General Platform for Intelligent Agents. arXiv.

[5] Haarnoja, T. et al. (2018). Soft Actor Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. 5th International Conference on Machine Learning, ICML 2018, v. 5, p. 2976–2989

[6] Lytsus, V. (2021), Unity 3D Yacht Simulator. Disponível em

<https://github.com/vlytsus/unity-3d-boat>
Acesso em: 15 mai. 2021.