XIII Seminário de Extensão e Inovação
XXVIII Seminário de Iniciação Científica e Tecnológica da UTFPR
Ciência e Tecnologia na era da Inteligência Artificial: Desdobramentos no Ensino Pesquisa e Extensão
20 a 23 de novembro de 2023 - *Campus Ponta Grossa, PR*
SEI-SICITE 2023

Condução de Veículo Autônomo usando Técnicas de Aprendizagem por Reforço

Autonomous Vehicle Driving using Reinforcement Learning Techniques

Gabriel Maestre Costa[1], Gleifer Vaz Alves[2]

RESUMO

Este artigo apresenta uma abordagem para o treinamento de veículos autônomos centrada em Aprendizagem por Reforço e sua aplicação em simulações. Utilizou-se o simulador CARLA, para criar um ambiente urbano simples, onde um veículo é treinado utilizando o algoritmo PPO (*Proximal Policy Optimization*). O veículo autônomo foi testado em alguns cenários por meio de métricas como recompensa acumulada, distância percorrida, velocidade média, tempo de duração no ambiente, etc. Os resultados obtidos indicam o potencial em explorar técnicas de aprendizagem em ambientes simulados. Almeja-se que este estudo ofereça uma integração de simulador e técnicas de Aprendizagem por Reforço para facilitar a experimentação de métricas de aprendizagem para simulação de veículos autônomos em cenários de trânsito urbano.
PALAVRAS-CHAVE: Aprendizado por Reforço; Simulador; Veículos autônomos.

ABSTRACT

This paper presents an approach to autonomous vehicle training centred on Reinforcement Learning (RL) and its application in simulations. Using the CARLA simulator, we designed a simple urban environment where a vehicle is trained using the PPO algorithm (Proximal Policy Optimization). The autonomous vehicle was tested in several scenarios using metrics such as accumulated reward, distance travelled, average speed, duration in the environment, etc. The results obtained indicate the potential of exploring learning techniques in simulated environments. We intend that our work brings an integrated solution using a simulator and techniques of RL so that one can easily test different learning techniques for the simulation of autonomous vehicles in urban traffic scenarios.
KEYWORDS: Reinforcement Learning; Simulator; Autonomous Vehicles.

INTRODUÇÃO

A condução autônoma é um dos avanços mais notáveis na indústria automobilística e tem o potencial de transformar a mobilidade urbana. No entanto, enfrenta desafios complexos, incluindo a criação de algoritmos de condução para veículos. Neste contexto, o uso de técnicas de Aprendizado por Reforço, do inglês *Reinforcement Learning* (RL), em ambientes simulados emerge como uma abordagem promissora. Este trabalho explora a integração de diferentes ferramentas para simulação e treinamento de um veículo autônomo (VA) em um ambiente simulado de trânsito urbano. Utilizou-se o seguinte: simulador CARLA; biblioteca Gym, biblioteca *Stable Baselines3* para o aprendizado;

[1] Bolsista do(a) CNPq. Universidade Tecnológica Federal do Paraná, Ponta Grossa, PR, Brasil. E-mail: gabrielmaestre@alunos.utfpr.edu.br. ID Lattes: 1375040843673618.
[2] Docente no Departamento Acadêmico de Informática. Universidade Tecnológica Federal do Paraná, Ponta Grossa, PR, Brasil. E-mail: gleifer@utfpr.edu.br. ID Lattes: 4988640748980805.

algoritmo PPO (*Proximal Policy Optimization*) e a política MLP. Por meio desta integração de ferramentas e técnicas almeja-se oferecer uma abordagem para soluções de RL na condução de VAs em ambientes simulados. De forma a explorar diferentes técnicas e políticas de aprendizagem em cenários de trânsito urbano.

APRENDIZAGEM POR REFORÇO

Aprendizagem por Reforço (RL) é uma abordagem de aprendizado que capacita agentes a tomar decisões autônomas ao interagir com um ambiente. Um agente aprende a executar ações para maximizar a acumulação de recompensas ao longo do tempo (SUTTON; BARTO, 2018). Conforme (SUTTON; BARTO, 2018), o processo de aprendizagem por reforço envolve diversos componentes fundamentais. O agente é a entidade autônoma que interage com o ambiente, tomando decisões e executando ações, este ambiente é representado pelo contexto no qual o agente opera, e é definido por estados, ações e recompensas. O estado representa a percepção atual do ambiente pelo agente, que pode ser uma imagem ou informações relevantes. As ações são escolhas feitas pelo agente para influenciar o ambiente, seguindo uma política, que por sua vez é a estratégia do agente para escolher ações com base nos estados observados. As recompensas são valores numéricos que indicam o quão benéficas foram as ações tomadas pelo agente em um determinado estado, sendo estes valores positivos ou negativos. RL segue um processo onde o agente interage com o ambiente, tomando ações que acredita ser a mais adequada com base em seu estado atual, coleta informações sobre as consequências de suas ações e atualiza sua política para melhorar seu desempenho ao longo do tempo. Ademais, (SUTTON; BARTO, 2018) explica que o desenvolvimento do RL é descrito em etapas: Observação de Estado: o agente observa o estado atual do ambiente através de percepções; Escolha de Ação: com base na política, o agente escolhe uma ação a ser executada no estado atual; Interação com o Ambiente: a ação escolhida é executada no ambiente, resultando em um novo estado e uma recompensa (positiva ou negativa) ao agente; Atualização de Política: o agente atualiza sua política com base nas recompensas obtidas, buscando maximizar a recompensa acumulada ao decorrer do tempo. Existem diversos algoritmos de RL, entre eles o PPO (*Proximal Policy Optimization*) (SCHULMAN et al., 2017). O algoritmo PPO é uma técnica projetada para treinar agentes em cenários complexos, o mesmo pertence à categoria de algoritmos de otimização de política, que se concentram em melhorar a estratégia ou política do agente para tomar decisões melhores ao interagir com o ambiente. Para tanto, o PPO atualiza gradualmente a política do agente com base nas recompensas obtidas durante o treinamento.

FERRAMENTAS UTILIZADAS

Este trabalho envolve a integração de diversas ferramentas centradas na linguagem Python.

CARLA é um simulador de código aberto, projetado para criar ambientes urbanos para testar algoritmos de controle e navegação de VAs (DOSOVITSKIY et al., 2017).

Biblioteca Stable Baselines3 é uma biblioteca de código aberto que oferece implementações de diversos algoritmos de RL, incluindo o PPO. Foi escolhida devido à sua eficiência computacional,

estabilidade e flexibilidade (RAFFIN et al., 2021).

Biblioteca Gym (OpenAI Gym) foi usada na modelagem do ambiente de treinamento para o projeto, especificando observações, ações, recompensas e critérios de término. A harmonização desta com o simulador CARLA facilitou a passagem de parâmetros para o treinamento.

Python e demais bibliotecas Python foi fundamental para viabilizar a integração de todas as ferramentas já mencionadas e ainda bibliotecas como OpenCV e Random, as quais foram usadas para tarefas como, processamento de imagem e aleatorização de cenários.

DESENVOLVIMENTO

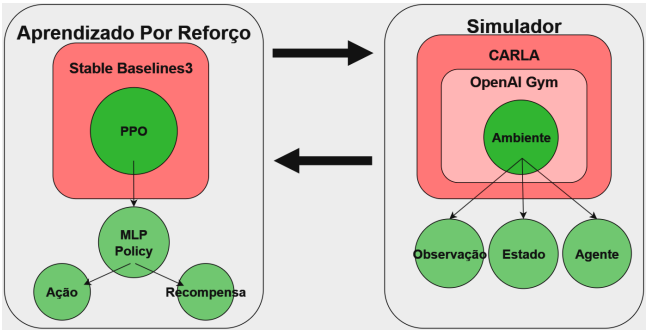
Este trabalho tem dois principais componentes.

Aprendizado por Reforço: para o desenvolvimento e e tomada de decisões foi usado o PPO, um dos algoritmos de RL da biblioteca *Stable Baselines3*, o qual tem objetivo de encontrar a melhor estratégia ou política para um agente maximizar sua recompensa cumulativa ao longo do tempo em um ambiente específico (SCHULMAN et al., 2017). Além disso, nesse algoritmo foi selecionada a política MLP (*Multilayer Perceptron*) que tem como função principal mapear o estado atual do ambiente (observações) para ações. Durante o treinamento, a política MLP é ajustada para que a ação mais apropriada seja selecionada com base no estado observado.

Simulador: Para a simulação do ambiente de trânsito urbano foi usado o simulador CARLA e a biblioteca Gym, a integração dessas ferramentas gerou um ambiente adequado para o treinamento do VA, já que o CARLA é responsável pela geração do cenário e a Gym pela configuração do ambiente e retorno das informações necessárias para o treinamento do VA.

Como é possível observar na Fig. 1, a troca de informações entre os componentes é necessária para que o treinamento do VA seja realizado, uma vez que o ambiente trará as informações necessárias que o algoritmo de aprendizado requer para efetuar o treinamento. A interação entre os componentes ocorre devido ao ciclo do aprendizado, onde o ambiente remete ao algoritmo informações essenciais e o algoritmo utiliza a política selecionada para a tomada de decisão de ação do VA, que quando é feita recebe uma recompensa pela mesma e altera diretamente o ambiente, onde este tem seus valores atualizados, e novamente destina as novas informações ao algoritmo.

Figura 1 – Componentes principais



Fonte: Elaborado pelo autor

Para o crescimento na média da coleta númerica de recompensas do agente foram testadas

diversas estruturas de recompensas e diferentes valores de parâmetros da política MLP. O espaço de ação do VA (no ambiente) foi mapeado em uma matriz $9x4$, onde existe a possibilidade de nove ações para direção do veículo e quatro possibilidades de aceleração, o balanceamento entre direção e aceleração é fundamental para que o VA não tome decisões equivocadas ao acelerar de maneira brusca e mudar de direção repentinamente, o que poderia resultar em colisão. Para a passagem do espaço de observação foi definido um vetor entre valores de 0 e 1, com diversos parâmetros que representam a imagem que foi capturada pelo sensor do VA, de modo que a política defina a melhor ação a ser tomada com determinada observação e recompensa a ser obtida naquele momento. Ainda foi desenvolvida uma espécie de prevenção para que o VA não aprenda a apenas movimentar-se em círculos, o mesmo pode avaliar essa decisão como adequada, uma vez que aumentará o seu ganho de recompensa e também o tempo de duração sem colidir com obstáculos. Logo, foi necessário aplicar uma prevenção com um contador de tempo que será acionado caso o VA mantenha a direção fixa por um período de tempo e a punição aumentará conforme o tempo de duração do VA mantendo a mesma direção. Igualmente foram implementados outros critérios de recompensas para que o VA consiga concluir seu objetivo, tais como velocidade, distância percorrida e balancemaneto entre aceleração e direção. Estes fatores foram implementados de forma que caso o VA mantenha uma velocidade moderada o mesmo será recompensado com valores posítivos, e caso esteja trafegando com velocidades inadequadas (i.e, muito alta ou muito baixa) ele será punido. Em relação a distância trafegada foi feito uma relação de distância, onde quanto maior a distância percorrida, maior será a recompensa recebida, em caso de distâncias muito baixas uma punição é aplicada. Para o balanceamento entre direção e aceleração foi realizado uma multiplicação entre os dois parâmetros e o resultado é passado como recompensa ao veículo. Essa estrutura de RL esta é responsável pela próxima tomada de decisão. Para tanto, é necessária a configuração do PPO, existem quatro principais parâmetros no trabalho: escolha da política, taxa de aprendizado, número de passos por episódio e semente. O primeiro decide qual política o algoritmo PPO irá utilizar, este possui diversas políticas, nesse trabalho foi selecionada a MLP por conta da sua ampla variedade para trabalhar com tipos de observações(SONG; SUN, 2021). O segundo parâmetro é a taxa de aprendizado, o qual controla o tamanho dos passos que o algoritmo dá ao ajustar os pesos da rede neural (no caso da política MLP) durante o treinamento. Uma taxa de aprendizado maior permite atualizações de peso mais significativas, o que pode acelerar o treinamento, mas também pode torná-lo instável. Uma taxa de aprendizado menor torna o treinamento mais estável, mas pode ser mais lento. Quanto ao número de passos por episódio, este refere-se ao número de etapas de simulação que o VA leva antes de atualizar sua política. Em um ambiente de RL uma etapa é um único ciclo de observação, ação e obtenção de recompensa. Um valor maior de passos permite que o agente colete mais informações antes de atualizar sua política, tornando as atualizações mais estáveis. No entanto, isso também pode tornar o treinamento mais lento. A semente é um valor que define o estado inicial do gerador de números aleatórios. Usar uma semente permite que os experimentos sejam reproduzíveis, ou seja se for usada a mesma semente em diferentes execuções, os mesmos resultados iniciais devem ser obtidos. Isso é útil para depuração, teste e comparação de diferentes configurações de treinamento.

EXPERIMENTOS

Nesta seção são descritos os experimentos conduzidos para avaliar o desempenho do VA com técnicas de RL em um ambiente simulado no CARLA.

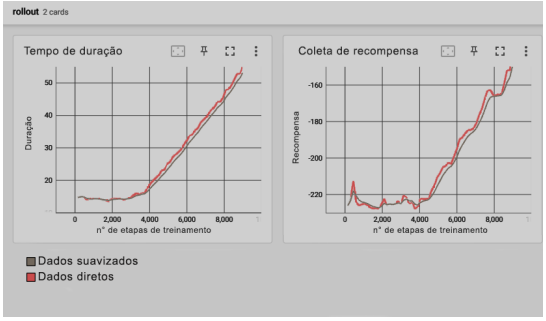
Configuração: os experimentos foram realizados em uma máquina com CPU Ryzen 7-5700U, 8GB de RAM e uma GPU Integrada; SO Windows 11; software do simulador CARLA versão 0.9.10; biblioteca Stable Baselines3 versão 2.0.0; Python 3.7.

Política de Aprendizado:

A política MLP foi escolhida devido à sua capacidade de mapear efetivamente observações em ações (SONG; SUN, 2021). No treinamento, a MLP é integrada à um algoritmo de aprendizado por reforço, como o PPO. Durante o treinamento, o VA interage com o ambiente simulado, coletando observações de estado, como dados de sensores e informações ambientais. A MLP é então treinada para aprender como mapear essas observações em ações, buscando maximizar a recompensa cumulativa ao longo do tempo. Ela é composta por várias camadas de unidades neuronais e cada camada tem pesos ajustáveis. Essas camadas podem incluir uma camada de entrada para receber as observações do ambiente, uma ou mais camadas ocultas para aprender representações intermediárias complexas e uma camada de saída que determina as ações a serem tomadas. O treinamento da MLP envolve o ajuste desses pesos por meio do algoritmo de RL para melhorar o desempenho do VA. A flexibilidade da política em questão é evidente em sua capacidade de ajustar parâmetros importantes, como a taxa de aprendizado, que controla o tamanho das atualizações de peso durante o treinamento, e a quantidade de passos por etapa de treinamento, que determina quantos passos será feito por episódio. Esses parâmetros são ajustados de acordo com as características específicas do ambiente e os objetivos do projeto. No uso posterior dessa política treinada, a MLP se torna o componente de tomada de decisão do VA, permitindo que ele escolha ações com base nas observações de estado recebidas. Em resumo, a MLP desempenha um papel central na aprendizagem do VA, ajudando-o a mapear informações do ambiente em ações relevantes e, assim, avançar na direção de um comportamento autônomo mais eficaz.

Resultados Obtidos: a Fig. 2 ilustra a relação entre a média de tempo do VA no ambiente por episódio e a média na coleta númerica de recompensa realizada pelo VA. No gráfico observa-se que o crescimento dos dois parâmetros foi praticamente exponencial, isso indica que o aprendizado do VA foi crescente e a condução do mesmo foi melhorando ao decorrer do tempo. Destaca-se que este foi o melhor resultado obtido com base em vários testes realizados de estruturação de recompensas e parâmetros. A configuração para obtenção desse resultado é: política MLP, taxa de aprendizado 0.001, 200 passos por episódio e semente aleatória. Obs: os dados coletados para (Fig. 2) utilizaram apenas 60 iterações.

Figura 2 – Alguns resultados obtidos



Fonte: Elaborado pelo autor

CONCLUSÃO

Este artigo descreve a integração de ferramentas de simulação com técnicas de RL. A combinação do simulador CARLA com bibliotecas permite que o VA aprenda a navegar em cenário urbano. Algumas estratégias de aprendizado foram exploradas buscando obter uma forma adequada de condução do VA. Como continuidade almeja-se refinamentos na política, incluindo o uso de arquiteturas de redes neurais alternativas. A adição de obstáculos dinâmicos, como pedestres e semáforos ao cenário pode enriquecer o treinamento. Pretende-se implementar um sistema multi-agente com veículos interagindo e (aprendendo) a respeitar as leis de trânsito.

Agradecimentos

Agradecimentos ao CNPq pela concessão de bolsa para realização do projeto.

REFERÊNCIAS

DOSOVITSKIY, Alexey et al. CARLA: An Open Urban Driving Simulator. In: LEVINE, Sergey; VANHOUCKE, Vincent; GOLDBERG, Ken (Ed.). **Proceedings of the 1st Annual Conference on Robot Learning**. [S.l.]: PMLR, 13–15 Nov 2017. v. 78. (Proceedings of Machine Learning Research), p. 1–16. Disponível em: .

RAFFIN, Antonin et al. Stable-Baselines3: Reliable Reinforcement Learning Implementations. **Journal of Machine Learning Research**, v. 22, n. 268, p. 1–8, 2021. Disponível em: .

SCHULMAN, John et al. Proximal Policy Optimization Algorithms. **CoRR**, abs/1707.06347, 2017. Disponível em: .

SONG, Yuda; SUN, Wen. PC-MLP: Model-based Reinforcement Learning with Policy Cover Guided Exploration. In: MEILA, Marina; ZHANG, Tong (Ed.). **Proceedings of the 38th International Conference on Machine Learning**. [S.l.]: PMLR, 18–24 Jul 2021. v. 139. (Proceedings of Machine Learning Research), p. 9801–9811. Disponível em: .

SUTTON, Richard S.; BARTO, Andrew G. **Reinforcement Learning: An Introduction**. Second. [S.l.]: The MIT Press, 2018. Disponível em: .