Agentes BDI e Aprendizagem: um mapeamento sistemático e utilização com a biblioteca MASPY

Felipe Merenda Izidorio[1], **Alexandre L. L. Mellado**[1], **André Pinz Borges**[1], **Gleifer Vaz Alves**[1]

[1]Departamento Acadêmico de Informática
Universidade Tecnológica Federal do Paraná (UTFPR)
Ponta Grossa – PR – Brasil

{felipemerenda, mellado}alunos.utfpr.edu.br

{apborges, gleifer}@utfpr.edu.br

Abstract. *Reinforcement Learning algorithms can solve sequential decision processes through repeated interactions with an environment. This approach solves complex challenges and enables technological innovations such as Autonomous Vehicles (AVs). With this in mind, this paper presents the planning, execution and conclusions of a systematic mapping of the literature on learning algorithms for AVs. One gap identified is integrating BDI Intelligent Agent architecture with Reinforcement Learning. To address this, an example is presented using the MASPY library in Python, in which a BDI agent is programmed to use learning components.*

Resumo. *Os algoritmos de Aprendizagem por Reforço são capazes de resolver processos de decisão sequenciais por meio de interações repetidas com um ambiente. Essa abordagem permite a solução de desafios complexos e possibilita inovações tecnológicas, como os Veículos Autônomos (VAs). Com isso em mente, este artigo apresenta o planejamento, execução e conclusões de um mapeamento sistemático da literatura sobre algoritmos de aprendizagem para VAs. Uma lacuna identificada é a integração de arquitetura de Agentes Inteligentes BDI com Aprendizagem por Reforço. Para abordar isso, é apresentado um exemplo usando a biblioteca MASPY em Python, em que é programado um agente BDI que utiliza componentes de aprendizagem.*

1. Introdução

Agentes inteligentes são amplamente utilizados em tomada de decisão envolvendo sistemas autônomos, em sistemas de recomendação, navegação, locomoção autônoma ou definição do fluxo de dados de rede [Gronauer and Diepold 2022]. Esse conceito é utilizado na área de aprendizado por reforço como o componente que aprende sobre um determinado ambiente. Em específico, uma das maiores vantagens deste aprendizado é sua adaptabilidade a diferentes áreas de atuação [Shakya et al. 2023].

Baseados no Modelo de Decisão de Markov (MDP), que estabelece que os estados futuros e recompensas são independentes dos estados e ações passadas, e nas Equações

de Bellman [Bellman 1957] usadas para resolver problemas de decisões sequenciais, os algoritmos de Aprendizagem por Reforço (AR) são capazes de aprimorar as tomadas de decisões por meio de interações repetidas com o ambiente [Sutton and Barto 2018]. Estes algoritmos podem solucionar desafios complexos e possibilitar várias inovações tecnológicas, como os Veículos Autônomos (VA), onde um veículo com AR é capaz de desenvolver uma política de ações ótima com base no estado atual.

A execução de um mapeamento sistemático da literatura sobre algoritmos de aprendizagem para a condução de VAs visa analisar de maneira abrangente as pesquisas existentes, identificar tendências e desafios. Este processo inclui a geração das questões de pesquisa, definição dos critérios de inclusão e exclusão, a realização de buscas em bases de dados acadêmicas, e a aplicação de métodos padronizados para avaliar a qualidade e relevância dos estudos selecionados.

Após a realização do mapeamento sistemático foi possível identificar uma lacuna (que conforme o nosso conhecimento) não foi amplamente explorada na área, a qual é a utilização da arquitetura de Agentes Inteligentes juntamente com Aprendizagem por Reforço, tendo como referência o modelo BDI [Bratman 1987], baseado em estados mentais *belief-desire-intention* para agentes deliberativos. Esse tipo de abordagem possui a sua capacidade de tomar decisões complexas, ao considerar múltiplos objetivos e ações em ambientes dinâmicos e complexos, e suas ações são executadas com base em suas crenças, desejos e intenções, promovendo mais transparência e previsibilidade.

Visando combinar um sistema multiagente BDI com aprendizado por reforço, foi desenvolvida a biblioteca MASPY, primeiramente apresentada sem aprendizagem em [Mellado et al. 2023]. Esta biblioteca, em Python, possui as abstrações necessárias para a programação de múltiplos agentes BDI com acesso a uma classe de aprendizagem para expandir suas capacidades. Este trabalho apresenta um exemplo de aprendizado sobre a movimentação em um ambiente 2D, como o agente BDI interage com a política encontrada e como as questões analisadas no mapeamento refletem no futuro da biblioteca. MASPY apresenta uma proposta semelhante à encontrada em [Bosello 2020], porém enquanto este estende o aprendizado em Jason com foco em veículos autônomos, é esperado que MASPY possibilite um desenvolvimento genérico de agentes.

O trabalho está organizado da seguinte maneira: a seção 2 apresenta o mapeamento sistemático; a seção 3 descreve a biblioteca MASPY; a seção 4 mostra um exemplo agente BDI com aprendizagem; e a seção 5 apresenta as considerações finais.

2. Mapeamento Sistemático da Literatura

Com o propósito de identificar algoritmos de aprendizagem na condução de VAs e encontrar possíveis lacunas dentro da área, foi realizado um mapeamento sistemático da literatura. A Seção 2.1 apresenta a metodologia utilizada e as etapas realizadas, a Seção 2.2 aborda sobre o planejamento, a Seção 2.3 fala sobre a execução, a Seção 2.4 faz uma análise dos resultados e, por fim, a Seção 2.5 apresenta algumas conclusões.

2.1. Metodologia de Pesquisa

Para a execução do mapeamento sistemático da literatura foi utilizada a metodologia de Kitchenham e Chartes [Kitchenham and Charters 2007], com o objetivo de identificar e classificar a pesquisa relacionada a um tópico amplo de pesquisa. Há três etapas a serem

executadas: o planejamento, que contêm a descrição do problema e uma especificação das questões de pesquisa, a execução, no intuito de identificar as pesquisas mais relevantes realizando uma extração/síntese dos estudos e uma etapa de avaliação de qualidade dos estudos, e uma análise dos resultados encontrados, respondendo às questões de pesquisa elaboradas na etapa de planejamento. Para auxiliar a realização desse mapeamento, foi utilizado a ferramenta Parsifal [Freitas and Segatto 2021].

2.2. Planejamento

O objetivo desse mapeamento é buscar por estudos relacionados a algoritmos de aprendizagem por reforço com foco na área de condução de VAs considerando 5 questões:

QP1. Quais algoritmos de aprendizagem são usados no trabalho?
QP2. É realizada uma comparação com outros algoritmos de aprendizagem?
QP3. Quais métricas de avaliação são utilizadas no trabalho?
QP4. Qual a técnica predominante nos trabalhos de condução de VAs?
QP5. Dentro da área de condução de VAs, qual temática é a mais abordada?

A partir dessas questões é possível analisar os resultados encontrados. Antes de realizar a busca dos trabalhos relacionados, é necessário definir algumas palavras-chave, na língua inglesa, relacionadas ao objetivo do mapeamento, no caso: *Reinforcement Learning* e *Autonomous Vehicle Navigation*, juntamente com os seus sinônimos, e combiná-las com os operadores lógicos AND e OR para formar uma *string* de busca. Dessa forma, pode-se executar uma busca assertiva nos repositórios de pesquisa. A *string* resultante foi: *("Reinforcement Learning"OR "Decision Making"OR "Deep Reinforcement Learning"OR "Reinforcement Learning Algorithms") AND ("Autonomous Vehicle Navigation"OR "Autonomous Vehicle Driving"OR "Autonomous Vehicle Movement Control"OR "Self-Driving Car Driving"OR "Self-Driving Car Movement Control").*

Para a realização da busca dos estudos foram usadas três bases de busca: IEEE Digital Library, Science Direct e Scopus. O motivo da escolha dessas bases de dados se deve ao fato de possuírem um grande número de artigos científicos, e por apresentar uma ferramenta de busca eficiente capaz de selecionar os estudos mais relacionados ao trabalho por meio de uma *string* de busca e filtros, como: ano de publicação, área de pesquisa, idioma, etc. Em seguida, são definidos os critérios de inclusão (CI) e exclusão (CE):

CI1. Aborda sobre algoritmos de aprendizagem (***CI2.*** ...para agentes) (***CI3.*** ...para a condução de VAs)
CE1. Estudos duplicados / ***CE2.*** Fora de escopo / ***CE3.*** Não é estudo primário

A partir desses critérios será possível eleger os trabalhos que serão usados para a etapa de qualidade. Assim, foram definidas as questões de qualidade:

QQ1. O trabalho aborda sobre algoritmos de aprendizagem por reforço?
QQ2. O trabalho faz uma comparação com outros algoritmos?
QQ3. O trabalho se baseia em alguma arquitetura de agentes?
QQ4. O trabalho aborda sobre condução de VAs?
QQ5. O trabalho utiliza o(os) algoritmo(os) para conduzir os VAs?
QQ6. O trabalho faz uma avaliação do(os) algoritmo(os) abordado(os)?

Para cada uma das questões, há três respostas possíveis: Sim, Parcialmente ou Não, com a, respectiva, pontuação: 1, 0.5 ou 0. A soma dessas pontuações corresponde ao quão útil o trabalho é para o tema abordado.

2.3. Execução

Com a etapa de Planejamento realizada, inicia-se a etapa de Execução. Por meio das bases de dados citadas e a *string* de busca, foram obtidos 159 trabalhos sobre o tema. Na sequência, foram aplicados os critérios de inclusão e exclusão definidos anteriormente. Sendo que, com a aplicação do critério CE1, 37 trabalhos foram eliminados, resultando em 122 trabalhos. Com a aplicação do CE2 foram retirados 81 trabalhos. E por fim, com o critério CE3, 7 trabalhos foram descartados. Dessa forma, foram selecionados 34 trabalhos para leitura e realização da etapa de avaliação de qualidade.

Com os trabalhos selecionados foi realizada uma leitura de cada um deles para responder às questões de qualidade e mensurar as respectivas pontuações de qualidade (*Score*), explicado anteriormente na etapa de planejamento. Com o intuito de realizar uma investigação assertiva desses estudos obtidos, somente aqueles que obtiveram uma pontuação acima de 3 pontos foram escolhidos para servirem como base para a análise dos resultados. Esses trabalhos são mostrados na Tabela 1

ID	Título	*Score*	ID	Título	*Score*
1	[Liang et al. 2023]	5.0	12	[Sun et al. 2023]	5.5
2	[Dhinakaran et al. 2024]	5.0	13	[Hook et al. 2021]	4.0
3	[Zhu and Hayashibe 2023]	5.0	14	[Hartmann et al. 2020]	5.0
4	[Arvind and Senthilnath 2019]	5.0	15	[Jacinto et al. 2023]	4.0
5	[Dar et al.]	5.0	16	[González-Miranda et al. 2023]	4.0
6	[Tiong et al. 2023]	4.0	17	[Mackay et al. 2022]	4.0
7	[Zhai et al. 2021]	4.0	18	[Yang et al. 2023]	4.0
8	[Josef and Degani 2020]	4.0	19	[Lambert et al. 2021]	4.0
9	[Bin Issa et al. 2021]	4.5	20	[Taghavifar et al. 2024]	4.0
10	[Zhang et al. 2019]	5.0	21	[Qin et al. 2024]	5.0
11	[Gao et al. 2023]	5.0	22	[Cabezas-Olivenza et al. 2023]	4.5

Tabela 1. Avaliação de qualidade dos trabalhos selecionados

2.4. Análise dos Resultados

A seguir serão respondidas às questões de pesquisa a respeito dos trabalhos da Tabela 1.

QP1. Quais algoritmos de aprendizagem são usados no trabalho? Vários algoritmos de Aprendizado por Reforço foram citados, incluindo Q-Learning, Sarsa, Deep Q-Networks e Proximal Policy Optimization. O Deep Q-Networks se destacou, combinando as vantagens do Q-Learning com o poder das Redes Neurais. Além disso, alguns trabalhos desenvolveram variações desses algoritmos mencionados.

QP2. É realizada uma comparação com outros algoritmos de aprendizagem? Dos trabalhos analisados, 15 realizaram comparações, utilizando as métricas de avaliação para determinar quais algoritmos foram mais eficazes.

QP3. Quais métricas de avaliação são utilizadas no trabalho? Os estudos adotaram diferentes métricas de avaliação de acordo com o problema escolhido. Por exemplo, em prevenção de colisões, os autores investigaram o tempo em que um veículo se moveu antes de colidir com um obstáculo. Ao longo dos testes, o veículo conseguiu aumentar esse tempo. No entanto, todos os trabalhos utilizaram um gráfico de recompensa média por episódio para mostrar a eficácia do treinamento do veículo.

QP4. Qual a técnica predominante nos trabalhos de condução de VAs? Todos os estudos abordam Aprendizado por Reforço, porém, algumas técnicas foram combinadas com outras áreas para desenvolver frameworks ou métodos para resolver problemas específicos. Por exemplo, o artigo 9 usou Redes Neurais Convolucionais para processar imagens. O artigo 21 utilizou *Social Value Orientation*, um conceito da psicologia que descreve como as pessoas avaliam e valorizam relações sociais e tomam decisões em contextos sociais, em conjunto com Aprendizado por Reforço. No total, 17 artigos incorporaram Redes Neurais em seus algoritmos, destacando a eficácia dessa técnica na condução de VAs.

QP5. Dentro da área de condução de VAs, qual temática é a mais abordada? Dentro da vasta área de condução de VAs, os estudos focaram em temas específicos, como controle de velocidade (trabalho 14), manutenção de faixa (trabalho 16), controle de estabilidade (trabalhos 1 e 22), condução em ambientes acidentados (trabalho 8), prevenção de colisão com obstáculos estáticos e dinâmicos (trabalhos 1, 4, 16 e 20), e navegação em cenários multiagentes (trabalhos 7, 12 e 13), entre outros.

2.5. Conclusões do Mapeamento

As respostas das questões levantam considerações importantes sobre os algoritmos de aprendizagem para VAs. O Deep Q-Networks, amplamente citado, é uma opção versátil para abordar os desafios de envolvendo VAs. A comparação com outros algoritmos é crucial para avaliar sua eficácia e serve como guia para estudos futuros. Métricas específicas relacionadas ao problema são mais vantajosas para solucioná-lo. A utilização de ferramentas de outras áreas pode enriquecer o aprendizado dos agentes. A área de condução de VAs tem diversas temáticas a serem exploradas, incluindo aquelas menos abordadas, que podem ser relevantes para estudos futuros.

Vale ressaltar que nenhum dos trabalhos se baseia em alguma arquitetura de agentes (QQ2). Isso é uma lacuna presente dentro do escopo de algoritmos de aprendizagem e um estudo aprofundado em algumas arquiteturas, como o modelo BDI, é uma abordagem pertinente. Durante o processo de revisão foi descoberto o trabalho de [Bosello 2020], que apresenta uma extensão da linguagem de agentes Jason com técnicas de aprendizagem de máquina. Esse trabalho apresenta, assim como a MASPY, agentes BDI com aprendizado por reforço. Porém, enquanto este tem foco no desenvolvimento de navegação autônoma, a biblioteca MASPY almeja viabilizar a criação de sistemas genéricos para agentes, como descrito a seguir.

3. MASPY: Aprendizagem e Sistema BDI

A biblioteca MASPY é dividida em cinco classes para o desenvolvimento de sistemas multiagentes, como mostra a figura 1. A classe de agentes possui as funções para as lógicas de BDI, utilizando os componentes de crenças, objetivo e planos. A classe de ambiente fornece uma abstração onde agentes podem perceber e realizar ações sobre um contexto de forma simulada. A classe de comunicação permite que agentes troquem mensagens um com os outros. A classe de aprendizagem expande as capacidades do agente além do conjunto de regras do paradigma BDI. E a classe de administrador ajudar o programador na configuração e execução do sistema além de assegurar suas funcionalidades.

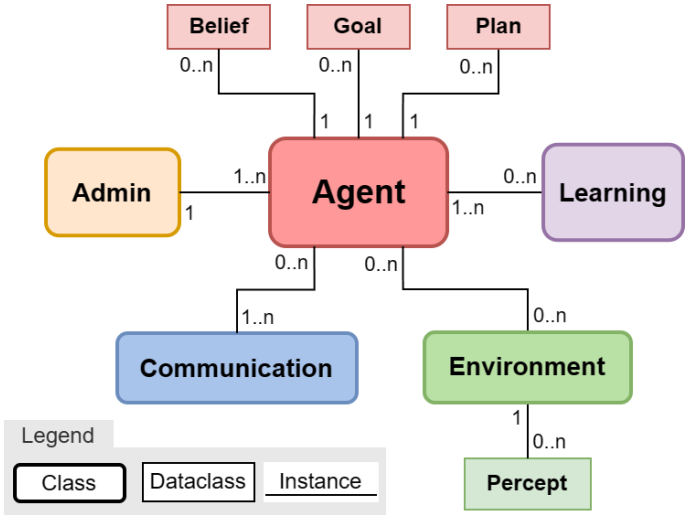

Figura 1. Diagrama das classes de MASPY

3.1. Classe de Agente

A classe de agente é a base da biblioteca MASPY. Ela compõe as abstrações para o gerenciamento de crenças e objetivos, assim como o ciclo de raciocínio que promove o comportamento BDI do agente. Durante a execução de um agente, este adquiri e perde crenças e objetivos que influenciam sua escolha de planos. Estas mudanças causam eventos que são o gatilho para que o agente realize ações.

A figura 2 apresenta como o raciocínio de cada agente transcorre durante a execução de um sistema multiagente usando MASPY. O começo do ciclo envolve a atualização de crenças. Para isso são consideradas percepções de ambiente e mensagens de outros agentes. Em seguida, os objetivos são atualizados. Neste começo do raciocínio, somente mensagens afetam os objetivos.

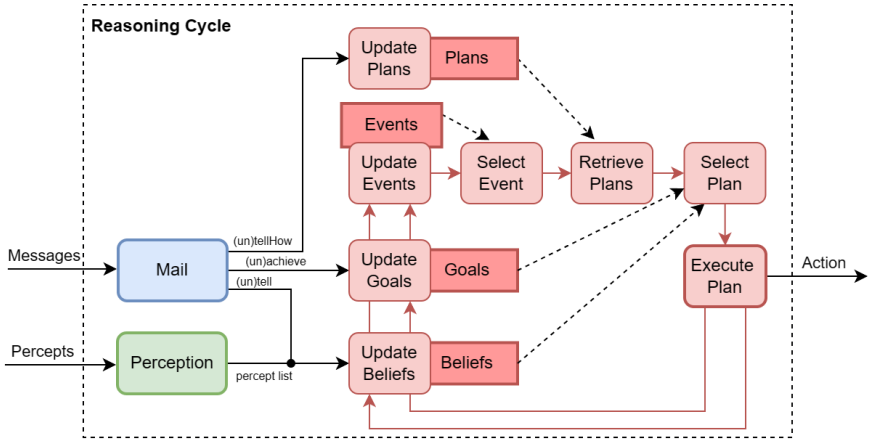

Figura 2. Ciclo de Raciocínio do Agente

As alterações feitas, seja ganho ou perda de informação, são adicionadas em uma lista de eventos. O evento mais antigo é então utilizado para recuperar planos que contém este evento como gatilho. Entre os planos encontrados, o primeiro em que seu contexto é

refletido com a lista de crenças e objetivos disponíveis é escolhido e executado. Durante a execução deste plano, ações no ambiente podem ser realizadas além de atualização de crenças e objetivos. Concluído o plano, o ciclo recomeça. Fora do ciclo principal, planos podem ser atualizados por mensagens especificas de outros agentes.

3.2. Classe de Aprendizagem

O uso da classe de aprendizagem da biblioteca MASPY é representado pela figura 3.

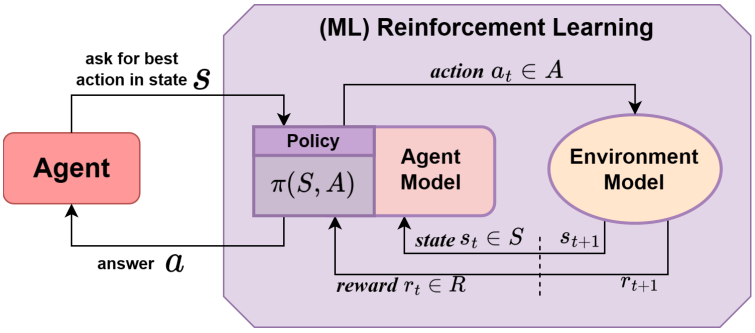

Figura 3. Diagrama de Agente utilizando um Aprendizado por Reforço

A figura 3 mostra um diagrama de como um agente pode utilizar técnicas de aprendizado por reforço. Esse agente pode requisitar o treinamento de um ambiente modelado e pedir pela melhor ação a ser tomada naquela política encontrada. Este treinamento é representado na figura pela interação entre o modelo de agente que realiza ações no ambiente, o qual retorna recompensas que atualiza a politíca do agente.

4. Exemplo de Utilização da Biblioteca

Como exemplo da utilização da MASPY com aprendizagem por reforço, será descrito um agente limpador que aprende a navegar em um ambiente 2D para alcançar todas as posições de sujeiras. O código completo está disponível em: *garbage_cleaner_learn.py*. A figura 4 apresenta um diagrama de sequência da execução e interação entre instâncias de Lrn (Aprendizado), Admin (Administrador), Agt (Agente) e Env (Ambiente).

Configuração: Lrn define seus parâmetros para o aprendizado e executa aprendizado com base nos parâmetros definidos; Lrn configura seu ambiente com o mapa da classe de ambiente; Room (Env) adiciona um *percept* para mostrar que o cômodo está sujo no momento e o Admin define seu nome como “Room”; Room adiciona um *percept* com as posições de sujeira, o mesmo que o mapa enviado para Lrn; Admin define o nome de Rbt (Agt) como “(Rbt,1)”; Rbt se conecta ao ambiente Room, define sua posição inicial e adiciona objetivo de decidir seu próximo movimento.

Execução: Admin inicia o sistema, executando a função de raciocínio de cada agente conhecido; Rbt inicia seu ciclo de raciocínio, percebendo Room; Rbt executa um plano com base em seu objetivo para decidir o movimento; Rbt solicita uma ação de Lrn que retorna a melhor ação aprendida; Rbt adiciona objetivo de mover com base na ação recebida; Essa sequência é repetida até que uma posição suja seja alcançada; Rbt percebe que está na posição de sujeira e a limpa; Room atualiza sua percepção das posições de

sujeiras; Rbt adiciona um objetivo para decidir novamente seu próximo movimento; O agente continua até que todas as posições de sujeira sejam limpas.

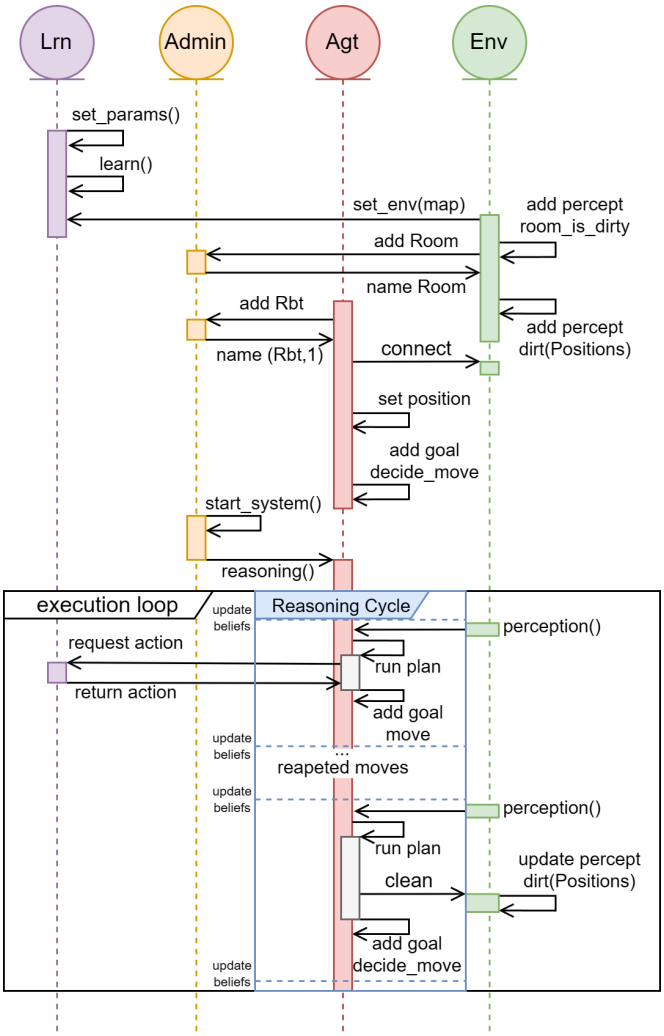

Figura 4. Diagrama de sequência da execução do exemplo

O algoritmo 1 apresenta o plano em que o agente Rbt escolhe entre mover para uma posição suja ou perceber que a limpeza acabou e deve voltar ao carregador. Para identificar que ainda existe posição suja, Rbt verifica, na linha 3, se contém uma crença "*room_is_dirty*" percebida no ambiente Room. Nas linhas 4 e 5, o agente Rbt utiliza o modelo de aprendizado pronto Lrn para requisitar a próxima ação em sua posição.

```
1     @pl(gain,Goal("decide_move"))
2     def decide_move(self,src):
3         if self.has(Belief("room_is_dirty",source="Room")):
4             state = Lrn.env.encode(*self.pos)
5             action,n_state,reward = Lrn.exec(Lrn.env, state)
6             self.add(Goal("move",(action,reward)))
7         else:
8             self.print(f"All dirt is cleaned")
9             self.add(Goal("recharge"))
```

Algoritmo 1. Plano decidir movimento

Com o algoritmo 2, Rbt executa um movimento baseado na ação e recompensa recebida de Lrn, indicado nas linhas 1 e 2. Neste exemplo, quando a recompensa é maior que zero, é determinado que o agente Rbt alcançou uma posição suja e logo ele recebe o plano de limpar esta posição. Caso contrario, Rbt requisita pela próxima ação.

```
1     @pl(gain,Goal("move",("Action","Reward")))
2     def move(self,src,action,reward):
3         match action:
4             case 0: direction = (1,0)
5             case 1: direction = (-1,0)
6             case 2: direction = (0,1)
7             case 3: direction = (0,-1)
8         self.pos = (self.pos[0]+direction[0], self.pos[1]+direction[1])
9         if reward > 0:
10            self.add(Goal("clean_dirt"))
11        else:
12            state = Lrn.env.encode(*self.pos)
13            action,n_state,reward = Lrn.exec(Lrn.env, state)
14            self.add(Goal("move",(action,reward)))
```

Algoritmo 2. Plano ação mover

5. Considerações Finais

Este artigo apresentou um mapeamento sistemático de estudos sobre a condução de veículo autônomos que utilizam aprendizagem por reforço. Essa pesquisa procurou identificar os algoritmos e técnicas mais utilizadas, os métodos para sua validação e os cenários em que são testados veículos autônomos. Também foi realizado a descrição e exemplificação de MASPY, uma biblioteca em Python para a implementação de sistemas multiagentes BDI com aprendizagem de máquina.

Entre estas questões analisadas no mapeamento encontrou-se a lacuna do desenvolvimento de agentes BDI utilizando aprendizado por reforço. MASPY, logo, apresenta a capacidade de desenvolver um sistema com os algoritmos Q-learning e Sarsa, porém não possui no momento o algoritmo mais destacado: Deep Q-Networks. Apesar de envolver um cenário de veículo autônomo, o exemplo apresenta a capacidade de movimentação em um cenário 2D utilizada para as decisões de um agente BDI.

Para o desenvolvimento futuro, é importante a validação da biblioteca MASPY. É necessário, por meio de comparações com outras bibliotecas de aprendizagem e desenvolvimento de agentes, mostrar onde se encontram as desvantagens e vantagens da MASPY. Como identificado no mapeamento, essas comparações incluem a determinação das técnicas e os cenários para a avaliação, como controle de velocidade e prevenção de colisão no caso de veículos autônomos. Capacidades futuras também incluem a implementação de novos algoritmos embutidos em MASPY para treinamento, como Deep Q-Networks, e incorporação da aprendizagem durante o próprio raciocínio dos agentes.

Referências

Arvind, C. S. and Senthilnath, J. (2019). Autonomous RL: Autonomous Vehicle Obstacle Avoidance in a Dynamic Environment using MLP-SARSA Reinforcement Learning. In *2019 IEEE 5th International Conference on Mechatronics System and Robots (ICMSR)*, pages 120–124, Singapore. IEEE.

Bellman, R. (1957). *Dynamic Programming*. Princeton University Press, Princeton, NJ, USA, 1 edition.

Bin Issa, R., Das, M., Rahman, M. S., Barua, M., Rhaman, M. K., Ripon, K. S. N., and Alam, M. G. R. (2021). Double Deep Q-Learning and Faster R-CNN-Based Autonomous Vehicle Navigation and Obstacle Avoidance in Dynamic Environment. *Sensors*, 21(4):1468. Number: 4 Publisher: Multidisciplinary Digital Publishing Institute.

Bosello, M. (2020). Integrating bdi and reinforcement learning: the case study of autonomous driving.

Bratman, M. (1987). *Intention, Plans, and Practical Reason*. Cambridge: Cambridge, MA: Harvard University Press.

Cabezas-Olivenza, M., Zulueta, E., Sanchez-Chica, A., Fernandez-Gamiz, U., and Teso-Fz-Betoño, A. (2023). Stability Analysis for Autonomous Vehicle Navigation Trained over Deep Deterministic Policy Gradient. *Mathematics*, 11(1):132. Number: 1 Publisher: Multidisciplinary Digital Publishing Institute.

Dar, S. A., Palanivel, S., and Geetha, M. K. Autonomous Taxi Driving Environment Using Reinforcement Learning Algorithms. *International Journal of Modern Education and Computer Science*, 14(3):88.

Dhinakaran, M., Rajasekaran, R., Balaji, V., Aarthi, V., and Ambika, S. (2024). Advanced Deep Reinforcement Learning Strategies for Enhanced Autonomous Vehicle Navigation Systems. In *2024 2nd International Conference on Computer, Communication and Control (IC4)*, pages 1–4.

Freitas, V. and Segatto, W. (2021). Perform Systematic Literature Reviews.

Gao, L., Wu, Y., Wang, L., Wang, L., Zhang, J., and Li, K. (2023). End-to-end autonomous vehicle navigation control method guided by the dynamic window approach. In *2023 IEEE 6th International Electrical and Energy Conference (CIEEC)*, pages 4472–4476.

González-Miranda, O., Miranda, L. A. L., and Ibarra-Zannatha, J. M. (2023). Q-Learning for Autonomous Vehicle Navigation. In *2023 XXV Robotics Mexican Congress (COMRob)*, pages 138–142.

Gronauer, S. and Diepold, K. (2022). Multi-agent deep reinforcement learning: a survey. *Artificial Intelligence Review*, pages 1–49.

Hartmann, G., Shiller, Z., and Azaria, A. (2020). Model-Based Reinforcement Learning for Time-Optimal Velocity Control. *IEEE Robotics and Automation Letters*, 5(4):6185–6192. Conference Name: IEEE Robotics and Automation Letters.

Hook, J., El-Sedky, S., De Silva, V., and Kondoz, A. (2021). Learning data-driven decision-making policies in multi-agent environments for autonomous systems. *Cognitive Systems Research*, 65:40–49.

Jacinto, E., Martinez, F., and Martinez, F. (2023). Navigation of Autonomous Vehicles using Reinforcement Learning with Generalized Advantage Estimation. *International Journal of Advanced Computer Science and Applications (IJACSA)*, 14(1). Number: 1 Publisher: The Science and Information (SAI) Organization Limited.

Josef, S. and Degani, A. (2020). Deep Reinforcement Learning for Safe Local Planning of a Ground Vehicle in Unknown Rough Terrain. *IEEE Robotics and Automation Letters*, 5(4):6748–6755. Conference Name: IEEE Robotics and Automation Letters.

Kitchenham, B. A. and Charters, S. (2007). Guidelines for performing systematic literature reviews in software engineering. Technical Report EBSE 2007-001, Keele University and Durham University Joint Report.

Lambert, R., Li, J., Wu, L.-F., and Mahmoudian, N. (2021). Robust ASV Navigation Through Ground to Water Cross-Domain Deep Reinforcement Learning. *Frontiers in Robotics and AI*, 8. Publisher: Frontiers.

Liang, J., Weerakoon, K., Guan, T., Karapetyan, N., and Manocha, D. (2023). AdaptiveON: Adaptive Outdoor Local Navigation Method for Stable and Reliable Actions. *IEEE Robotics and Automation Letters*, 8(2):648–655.

Mackay, A. K., Riazuelo, L., and Montano, L. (2022). RL-DOVS: Reinforcement Learning for Autonomous Robot Navigation in Dynamic Environments. *Sensors*, 22(10):3847. Number: 10 Publisher: Multidisciplinary Digital Publishing Institute.

Mellado, A. L. L., G., F. I., Alves, G. V., and Borges, A. P. (2023). Maspy: Towards the creation of bdi multi-agent systems. In *Proceedings of the 17th Workshop-School on Agents, Environments, and Applications (WESAAC 2023)*, pages 106–117.

Qin, J., Qin, J., Qiu, J., Liu, Q., Li, M., and Ma, Q. (2024). SRL-ORCA: A Socially Aware Multi-Agent Mapless Navigation Algorithm in Complex Dynamic Scenes. *IEEE Robotics and Automation Letters*, 9(1):143–150. Conference Name: IEEE Robotics and Automation Letters.

Shakya, A. K., Pillai, G., and Chakrabarty, S. (2023). Reinforcement learning algorithms: A brief survey. *Expert Systems with Applications*, page 120495.

Sun, Q., Zhang, L., Yu, H., Zhang, W., Mei, Y., and Xiong, H. (2023). Hierarchical Reinforcement Learning for Dynamic Autonomous Vehicle Navigation at Intelligent Intersections. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '23, pages 4852–4861, New York, NY, USA. Association for Computing Machinery.

Sutton, R. S. and Barto, A. G. (2018). *Reinforcement Learning: An Introduction*. The MIT Press, second edition.

Taghavifar, H., Wei, C., and Taghavifar, L. (2024). Socially Intelligent Reinforcement Learning for Optimal Automated Vehicle Control in Traffic Scenarios. *IEEE Transactions on Automation Science and Engineering*, pages 1–12. Conference Name: IEEE Transactions on Automation Science and Engineering.

Tiong, T., Saad, I., Teo, K. T. K., and Lago, H. B. (2023). Autonomous Vehicle Driving Path Control with Deep Reinforcement Learning. In *2023 IEEE 13th Annual Computing and Communication Workshop and Conference (CCWC)*, pages 0084–0092, Las Vegas, NV, USA. IEEE.

Yang, H., Yao, C., Liu, C., and Chen, Q. (2023). RMRL: Robot Navigation in Crowd Environments With Risk Map-Based Deep Reinforcement Learning. *IEEE Robotics*

and Automation Letters, 8(12):7930–7937. Conference Name: IEEE Robotics and Automation Letters.

Zhai, Y., Ding, B., Liu, X., Jia, H., Zhao, Y., and Luo, J. (2021). Decentralized Multi-Robot Collision Avoidance in Complex Scenarios With Selective Communication. *IEEE Robotics and Automation Letters*, 6(4):8379–8386. Conference Name: IEEE Robotics and Automation Letters.

Zhang, W., Gai, J., Zhang, Z., Tang, L., Liao, Q., and Ding, Y. (2019). Double-DQN based path smoothing and tracking control method for robotic vehicle navigation. *Computers and Electronics in Agriculture*, 166:104985.

Zhu, W. and Hayashibe, M. (2023). Autonomous Navigation System in Pedestrian Scenarios Using a Dreamer-Based Motion Planner. *IEEE Robotics and Automation Letters*, 8(6):3836–3843.