Comunicação Distribuída para Robôs Autônomos via DDS: Uma Abordagem Cooperativa com Deep-Q Networks

Carlos Daniel de Sousa Bezerra e Alisson Assis Cardoso e Flávio Henrique Teles Vieira

***Resumo*— Este artigo propõe uma arquitetura de comunicação para sistemas multi-robôs, permitindo a distribuição eficiente de estados durante o processo de aprendizado por reforço aplicado à navegação autônoma. A solução desenvolvida emprega comunicação distribuída baseada em middleware, utilizando o Data Distribution Service (DDS) para o compartilhamento de transições de estado entre os agentes, por meio de uma abordagem denominada $MARLQ_{DDS}$. A proposta é validada por meio de simulações no ambiente CoppeliaSim, integradas ao ROS2, e os resultados demonstram um ganho de aproximadamente 23,33% na taxa de sucesso em tarefas de navegação autônoma, quando comparado ao modelo DDQN sem comunicação entre agentes.**

***Palavras-Chave*— Middleware, DDS, Deep Reinforcement Learning, Multi-Agent, Robotic communication.**

***Abstract*— This article proposes a communication architecture for multi-robot systems, enabling efficient state distribution during the reinforcement learning process applied to autonomous navigation. The developed solution employs distributed communication based on middleware, using the Data Distribution Service (DDS) to share state transitions among agents through an approach called $MARLQ_{DDS}$. The proposed method is validated through simulations in the CoppeliaSim environment integrated with ROS2, and the results demonstrate an improvement of approximately 23.33% in the success rate for autonomous navigation tasks when compared to a baseline DDQN model without inter-agent communication.**

***Keywords*— Middleware, DDS, Deep Reinforcement Learning, Multi-Agent, Robotic communication.**

I. Introdução

A comunicação eficiente entre robôs móveis autônomos é um fator crítico para a navegação em ambientes dinâmicos e complexos. Em cenários multi-robô, a troca de informações melhora a eficiência da exploração, a tomada de decisão e a execução de tarefas colaborativas [1], [2]. Nesse contexto, destaca-se o controle descentralizado, no qual cada robô atua de forma autônoma, sem depender de um agente central [3].

Técnicas de comunicação entre agentes possibilitam a coordenação distribuída, embora desafios como latência e confiabilidade da transmissão ainda limitem sua aplicação. Estratégias recentes têm explorado o aprendizado por reforço multiagente para permitir que os robôs aprendam a partir de experiências próprias e compartilhadas [4].

Em ambientes sem infraestrutura tradicional (como redes celulares), garantir comunicação eficiente entre robôs torna-se mais desafiador [5], [6]. Assim, são necessárias soluções locais que viabilizem o aprendizado colaborativo e a sincronização de ações em tempo real.

Este trabalho propõe um algoritmo de comunicação multi-robô baseado no Serviço de Distribuição de Dados (*Data Distribution Service* - DDS), voltado ao compartilhamento eficiente de experiências entre agentes móveis em tarefas com ações semelhantes. A abordagem é fundamentada no aprendizado por reforço *off-policy* com redes *Deep-Q*, seguindo o método DEE-MARL [7], no qual robôs compartilham transições de estado por meio de estruturas denominadas memorias de repetição (*replay buffer*).

A principal **contribuição** deste trabalho é a proposta de implementação e validação prática de um protocolo de comunicação robótica confiável para sistemas multiagente autônomos, compatível com arquiteturas modernas como o ROS2. O protocolo utiliza o **middleware DDS** para viabilizar o compartilhamento eficiente de experiências entre robôs, incluindo dados sensoriais reais como: imagens RGB-D, sensores de odometria, recompensas e ações durante o processo de aprendizagem.

Avalia-se ainda requisitos de QoS (como latência e entrega de pacotes) referentes à arquitetura proposta. Diferentemente de abordagens anteriores da literatura, que assumem redes ideais em simulações, o presente estudo valida o protocolo em ambiente realista, utilizando o ROS2, medindo assim o desempenho da comunicação com métricas práticas. Além disso, a proposta considera o compartilhamento explícito de estados em algoritmos de aprendizado por reforço *off-policy* (DDQN), o que acelera a convergência e a eficiência da navegação colaborativa. Além das simulações em ambiente CoppeliaSim, foi desenvolvida uma infraestrutura complementar em ROS2 [8], que adota DDS de forma nativa.

Este artigo está estruturado da seguinte forma: a Seção 2 aborda os Trabalhos Relacionados; a Seção 3 descreve a Proposta de comunicação distribuída; a Seção 4 apresenta os Resultados experimentais; e a Seção 5 traz as Conclusões e perspectivas futuras.

II. Trabalhos Relacionados

A comunicação entre robôs autônomos tem sido amplamente estudada, especialmente em ambientes dinâmicos, onde

Carlos Daniel Bezerra, Departamento de Áreas IV, Instituto Federal de Goiás (IFG), Goiânia, carlos.daniel@ifg.edu.br; Flávio Vieira, CERISE (Centro de Excelência em Redes Inteligentes Sem Fio e Serviços Avançados) Escola de Engenharia Elétrica e da Computação, Universidade Federal de Goiás (UFG), e-mail: flavio_vieira@ufg.br; Alisson Cardoso, CERISE, Escola de Engenharia Elétrica e da Computação, Universidade Federal de Goiás (UFG), e-mail: alsnac@ufg.br. Este trabalho foi parcialmente financiado por Fundação de Amparo à Pesquisa do Estado de Goiás (FAPEG).

a troca de informações é essencial para a eficiência da navegação. Em [6], os autores propõem o uso da tecnologia LoRa para transmitir dados sensoriais em robôs móveis afastados do centro de comando, caracterizando uma abordagem *Robot-to-Infrastructure* (R2I). A solução apresentada permite a compressão e envio de dados do sensor LiDAR por meio de um protocolo de banda estreita, viabilizando o monitoramento remoto mesmo em áreas com baixa conectividade, com baixo consumo de energia.

Diferentemente disso, o presente trabalho concentra-se na comunicação robô-para-robô (R2R), em um ambiente colaborativo. Utiliza-se o DDS (*Data Distribution Service*) [9] para a troca de experiências de aprendizado entre os agentes durante a navegação, promovendo aceleração no processo de aprendizado por reforço multiagente.

O algoritmo *Deep Double Q-Network* (DDQN) [10] foi originalmente desenvolvido para mitigar a superestimação de valores Q no tradicional algoritmo DQN [11], é amplamente utilizado em controle e navegação de agentes únicos. Com o crescimento dos sistemas multiagentes, surgiram adaptações do DDQN voltadas à execução descentralizada e aprendizado cooperativo. No entanto, observa-se uma lacuna na literatura quanto à definição de protocolos eficazes para a troca de experiências estruturadas entre agentes.

Grande parte dos estudos em MARL (*Multiagent Reinforcement Learning*) focam na colaboração via recompensas compartilhadas, mas poucos abordam explicitamente como os agentes podem trocar experiências completas (estado, ação, recompensa, próximo estado) em redes descentralizadas. Essa troca pode ser implícita ou explícita, sendo esta última mais promissora para acelerar o aprendizado coletivo. Na prática, mecanismos de comunicação R2R (robot-to-robot) são comumente utilizados não apenas para troca de informações, mas para habilitar ou automatizar tarefas cooperativas.

Em [4], os autores propõem um método de comunicação com dois níveis de otimização: um nível superior, com compartilhamento de recompensas entre vizinhos, e um nível inferior, com políticas ajustadas localmente. Embora promova maior coordenação, a comunicação proposta limita-se à troca de recompensas, sem detalhar um protocolo estruturado.

Neste trabalho, propõe-se o compartilhamento de experiências completas entre robôs, utilizando o DDS como *middleware* de comunicação. A especificação do algoritmo de comunicação, aliada à implementação prática de rede distribuída, representa uma contribuição em relação às abordagens anteriores.

III. Aprendizado por Reforço e o Protocolo de Comunicação Multi-robô

O sistema de Aprendizado por Reforço (*Reinforcement Learning - RL*) tem como objetivo fornecer conhecimento a um agente por meio de interações em um ambiente. As qualidades das ações desse agente são medidas por meio da função de recompensa, ou função valor. A função valor de estado-ação $Q(s, a)$ quantifica as recompensas obtidas por esse agente em relação aos seus estados e ações, tanto no curto quanto no longo prazo. Dessa forma, o agente aprende políticas de ação π que maximizam a função valor. A função $Q(s, a)$ é uma variação do algoritmo de aprendizado temporal proposto por Bellman e é dada por:

$$Q^{\pi}(s, a) = r + \gamma \ \max Q^{\pi}(s', a') \tag{1}$$

onde r é a recompensa imediata obtida e γ é um fator de desconto para recompensas futuras, s, s', a e a' são os estados e ações presentes e futuros, respectivamente.

A função $Q(s, a)$ pode ser aproximada por redes neurais profundas no que é conhecido como *Deep Q-Network (DQN)* [11] . No entanto, o algoritmo DQN convencional sofre com o problema de superestimação dos valores Q, o que pode levar a um aprendizado instável e a políticas subótimas. Para mitigar esse problema, foi proposta a abordagem *Double Deep Q-Network (DDQN)* [10], que separa a seleção da ação e a avaliação da ação em duas redes neurais diferentes. No DDQN, a função de atualização da rede Q é modificada para reduzir essa superestimação:

$$Q^{\pi}(s, a) = r + \gamma Q(s', \arg\max_{a'} Q(s', a'; \theta); \theta') \tag{2}$$

onde:

- θ representa os pesos da **rede principal**, que é utilizada para selecionar a melhor ação no próximo estado, por meio de $\arg\max_{a'} Q(s', a'; \theta)$;
- θ' representa os pesos da **rede-alvo**, que é usada para avaliar o valor da ação selecionada.

Essa estratégia melhora a estabilidade do treinamento e permite que os agentes aprendam de forma mais eficiente.

O protocolo de comunicação proposto sistematiza o aprendizado por reforço multiagente por meio da troca de experiências entre robôs (R2R). A hipótese central é que, ao inserir transições de estados ótimas no *memory replay* de robôs vizinhos com políticas semelhantes, pode-se enriquecer e acelerar aprendizagem de maneira colaborativa. Esta seção descreve: i) o ambiente de simulação, ii) o formato das mensagens DDS e iii) a política de compartilhamento de estados.

A. Descrição do Ambiente de Simulação

Para testes e validação do protocolo proposto, utiliza-se o robô Pioneer 3DX, um modelo amplamente empregado em pesquisas de navegação autônoma. O robô é equipado com sensores visuais e não visuais, sendo eles: Câmera RGB-D, utilizada para percepção visual do ambiente + profundidade da cena; Odometria (IMU + Encoder), responsável por fornecer estimativas de posição e velocidade do robô.

A simulação é realizada no *software* CoppeliaSim (V-REP), onde um cenário virtual foi modelado para testar a comunicação multi-robô. O cenário, ilustrado na Figura 1, consiste em uma indústria contendo quatro estações de trabalho (*workspaces*), onde dois robôs devem aprender a se movimentar de forma autônoma por essas salas desempenhando missões semelhantes, porém com políticas distintas. Além disso os robôs devem navegar entre as estações de trabalho sem colisões. As missões específicas para cada robô são descritas a seguir:

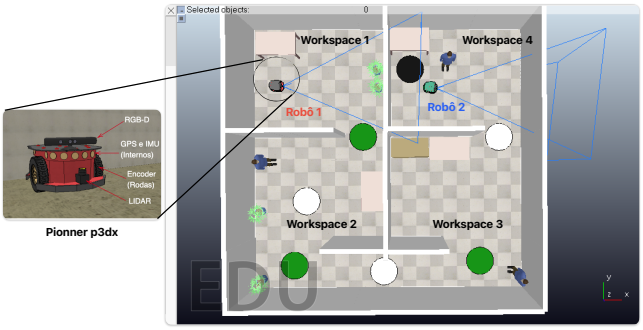

Fig. 1. Ambiente de simulação no CoppeliaSim contendo quatro estações de trabalho.

- Robô 1 (Vermelho): Parte da Estação de Trabalho 1 e deve alcançar a Estação de Trabalho 4, evitando obstáculos e colisões ao longo do percurso.
- Robô 2 (Azul): Parte da Estação de Trabalho 4 e deve alcançar a Estação de Trabalho 1, também evitando colisões com o ambiente e outros agentes.

Esses trajetos inversos permitem avaliar a eficiência da comunicação distribuída no compartilhamento de informações para navegação autônoma multi-robô.

Cada robô possui uma rede neural embarcada que executa o algoritmo DDQN [10]. Essa rede neural processa tanto as imagens da câmera RGB-D quanto os dados de odometria, realizando um processo de **fusão de sensores**, seguindo a estratégia sistemática de fusão descrita no trabalho de [12]. Vale ressaltar que as recompensas obtidas nesse ambiente significam a soma da atuação dos dois robôs no ambiente. À medida que os robôs ultrapassam a marca de **10 pontos** de recompensa acumulada, observa-se uma tendência de convergência para a máxima taxa de sucesso (aproximadamente 30 pontos). Por essa razão, nos resultados deste trabalho, define-se essa faixa como a **zona de sucesso**. A eficiência da navegação é avaliada com base na capacidade dos agentes em ultrapassar esse limiar. A recompensa individual pode ser calculada como:

$$r = \begin{cases} dist_{k-1} - dist_k & \text{se ativo} \\ -1, & \text{se colidiu} \\ +1, & \text{se alcançou o alvo} \end{cases} \quad (3)$$

$dist_k$ e $dist_{k-1}$ são respectivamente as distâncias entre o robô e o alvo atual (círculos verde ou branco na imagem) e a distância no *step* anterior. Quanto mais o robô se movimenta em relação ao alvo, maior a sua recompensa obtida.

O algoritmo descrito nesse artigo é denominado de $MARL_{Q_{DDS}}$ (*Multiagent Reinforcement Learning Double Deep Q-Network with Q-DDS*). Para garantir a eficiência na transmissão e processamento das informações, os estados compartilhados são encapsulados em mensagens estruturadas e organizadas em três categorias principais, como mostra a Tabela I.

B. Encapsulamento e Transmissão de Experiências

Os dados sensoriais coletados pelos robôs como imagens RGB-D, odometria, ações tomadas e recompensas obtidas são organizados em estruturas padronizadas de dados e encapsulados em mensagens DDS. Cada mensagem segue um formato composto por um cabeçalho (*header*) com identificador e carimbo temporal, um corpo de dados (*payload*) contendo a transição (s, a, r, s'), e um código de verificação (CRC) para garantir a integridade da comunicação.

TABELA I
FORMATO DAS MENSAGENS DDS

Tipo	Campos Principais	Tópico DDS
Imagem RGB-D	ID do robô, timestamp, imagem RGB e imagem de profundidade (Base64)	/robot/sensor/rgbd
Odometria	ID do robô, timestamp, posição (x, y, θ), velocidade (linear e angular)	/robot/sensor/odometry
Experiências	$(s, a, r, s', done)$	/robot/experience

A seleção das experiências a serem compartilhadas é realizada localmente por cada robô. A **rede** Q, executada localmente, avalia as transições armazenadas na memória local, atribuindo um valor de utilidade com base na função $Q(s, a)$. As transições de maior relevância (isto é, com maiores valores de Q) são então publicadas no tópico `/robot/experience` do DDS.

A responsabilidade por distribuir essas experiências otimizadas entre os demais agentes recai sobre o Serviço de Reprodução de Experiências (***Memory Replay Service***), um serviço distribuído implementado no DDS que gerencia o intercâmbio e a replicação das transições entre os robôs (ver Figura 2). A lógica de priorização dessas experiências é detalhada no Algoritmo 1 - $MARL_{Q_{DDS}}$.

C. Política de Compartilhamento de Experiências

A política de compartilhamento entre robôs segue um fluxo cíclico orientado pela avaliação local de transições e por um serviço de gerenciamento de memória distribuído. As regras principais são descritas a seguir:

- A partir do episódio 30 (fase intermediária de aprendizado) e a cada 5 episódios, cada robô avalia suas experiências locais e seleciona as 10 transições mais relevantes com base na função $Q(s, a)$ (rede Q);
- As transições selecionadas são publicadas no tópico DDS `/robot/experience`;
- O *Memory Replay Service*, implementado sobre DDS, atua simultaneamente como *subscriber* e *publisher* (assinante e publicador), escutando esse tópico e redirecionando as experiências recebidas para as memórias dos demais agentes conectados.

O computador usado para os testes possui as seguintes configurações: MD Ryzen 5 5600X, GeForce RTX 3060 TI 8GB, 24GB DDR4, SSD M.2 240GB. A próxima seção descreve os resultados obtidos com a proposta de comunicação.

IV. RESULTADOS

Esta seção apresenta os resultados obtidos com a implementação do sistema multiagente e a validação do protocolo DDS. Inicialmente, avalia-se o impacto do compartilhamento de estados no desempenho do aprendizado por reforço, em simulação no CoppeliaSim. Posteriormente, mensuram-se métricas de rede no uso prático do DDS via ROS2. São executadas **150 épocas** de treinamento.

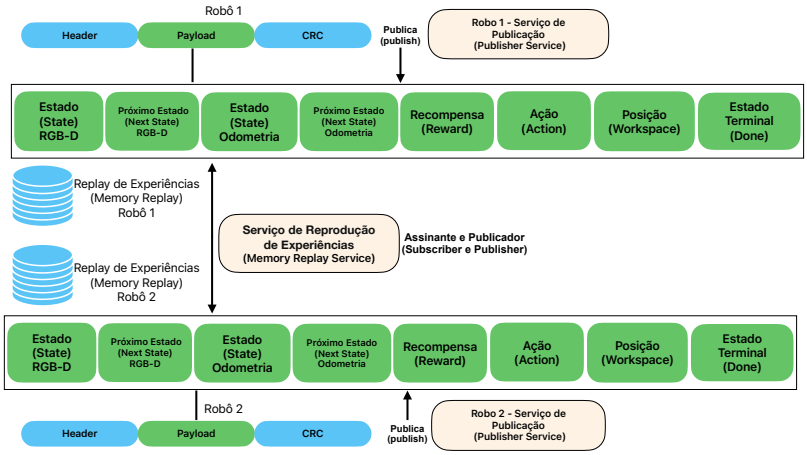

Fig. 2. Estrutura do *frame* DDS para transmissão de estados.

Inicializar cada robô i com rede Q_i e buffer de experiências B_i;
Definir a política π_i baseada em Q_i e os parâmetros de exploração;
Definir os tópicos DDS para troca de estados e melhores ações;
for $e = 0$ *até* E **do**
  for *cada robô* i **do**
    Observar estado atual s_i e selecionar ação $a_i \sim \pi_i(s_i)$;
    Executar a_i, observar novo estado s_i' e recompensa r_i;
    Armazenar transição (s_i, a_i, r_i, s_i') em B_i;
    if $e > 30$ *e* $e\%5 == 0$ **then**
      Selecionar as k melhores ações locais com maior valor $Q_i(s, a)$;
      Publicar essas ações e seus valores estimados no tópicos /robot/experience;
      Receber melhores ações estimadas de outros robôs via DDS;
      Atualizar o buffer B_i com as ações distribuídas pelo ***memory replay service***;
    end
  end
  for *cada robô* i **do**
    Amostrar um minibatch de experiências (s, a, r, s') de B_i;
    Atualizar parâmetros da rede Q_i via aprendizado DDQN:;
      $y = r + \gamma \cdot Q_{\text{target}}(s', \arg\max_{a'} Q(s', a'))$;
      Minimizar a perda: $\mathcal{L} = (y - Q(s, a))^2$;
  end
end

Algorithm 1: $MARL_{Q_{DDS}}$: Protocolo de Comunicação Multi-Robô com Compartilhamento da Rede Q via DDS

A Figura 3 compara o desempenho entre o $MARL_{Q_{DDS}}$ (linha vermelha) e o Pure DDQN (linha azul), onde não há troca de experiências entre os robôs. Além disso, o método proposto também é comparado com uma estratégia de distribuição de experiências aleatórias, sem passar pela rede Q local ($MARL - Rand$).

Os resultados mostram a evolução da recompensa média ao longo das execuções bem-sucedidas para os dois métodos avaliados. Observa-se que o sistema $MARL_{Q_{DDS}}$, ao empregar uma estratégia de compartilhamento da rede Q através do *memory replay service*, atinge recompensas significativamente mais elevadas e estáveis ao longo do tempo. O vídeo disposto em `bit.ly/41QKjAj` apresenta a performance do modelo $MARL_{Q_{DDS}}$ no ambiente de simulação. Em termos de tempo de execução, na Figura 4, verifica-se que o $MARL_{Q_{DDS}}$ permanece ativo por períodos mais longos que os demais.

Embora isso possa sugerir maior custo computacional, tal

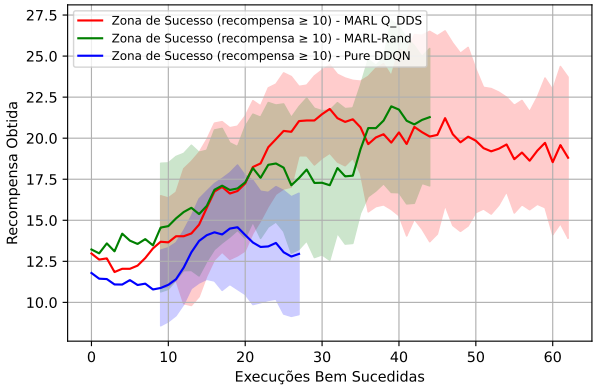

Fig. 3. Comparativo entre curvas de aprendizado: $MARL_{Q_{DDS}}$, $MARL-Rand$ e Pure DDQN

aumento está diretamente associado ao **prolongamento da atuação no ambiente**, permitindo que os agentes explorem e concluam missões de maior complexidade. Essa permanência é consequência direta da efetividade da comunicação entre os agentes, diferentemente do Pure DDQN, que tende a encerrar sua atuação de forma prematura. A Tabela II apresenta o comparativo das técnicas $MARL_{Q_{DDS}}$, ($MARL_{rand}$) e o Pure DDQN. Verifica-se que o algoritmo proposto $MARL_{Q_{DDS}}$ provê uma taxa de sucesso consideravelmente maior do que as outras abordagens consideradas.

TABELA II
RESUMO COMPARATIVO ENTRE MARL E DQN PURO NA FASE DE TREINAMENTO.

Parâmetro	$MARL_{Q_{DDS}}$	MARL-Rand	DDQN
Tempo médio (s)	154.95	119.27	111.38
Desvio padrão (s)	97.06	78.12	57.06
Taxa de sucesso	**42.00%**	24.00%	18.67%
Execuções bem-sucedidas	**63**	36	28

A. Validação da Comunicação DDS

Como o CoppeliaSim não suporta DDS, validou-se a comunicação distribuída no ROS2, que adota DDS como mid-

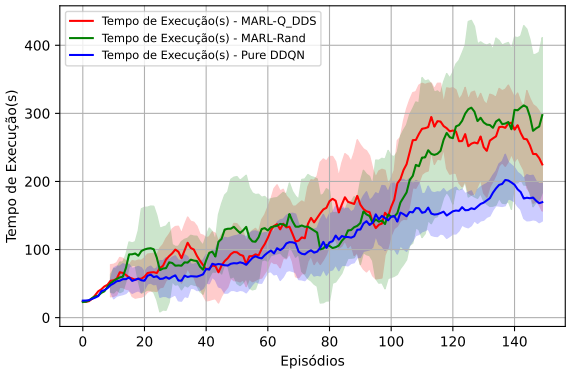

Fig. 4. Comparativo entre curvas de tempo de execução: $MARL_{Q_{DDS}}$, $MARL - Rand$ e Pure DDQN

dleware nativo. Para isso, os robôs compartilham transições completas de aprendizado (s, s', a, r, $done$) com um nó servidor de memória, conforme o esquema da Figura 2. A Tabela III resume as métricas da comunicação: a vazão média foi de $173.058bps$, com 208 pacotes trocados e latência média de $0,5s$. Essa avaliação foi realizada sobre uma rede Wi-Fi de $2.4GHz$.

TABELA III
MÉTRICAS DA COMUNICAÇÃO DDS

Métrica	**Valor**
Vazão média (bps)	173058.62
Pacotes recebidos	208
Tamanho médio do pacote (bytes)	86529.00
Latência média (ms)	500

O trabalho de [13] aborda alguns requisitos de latência para sistemas robóticos móveis de onde se pode inferir que a latência atingida na abordagem proposta pode atender adequadamente aplicações do tipo MARL (*off-policy*), cujo controle é descentralizado e executado na borda (*Edge-Control*) e uma vez que a comunicação entre agentes é usada para enriquecimento das memórias de experiência, e não para ações imediatas.

V. CONCLUSÕES

Os experimentos realizados demonstraram a viabilidade do uso do DDS como protocolo de comunicação para troca de experiências em um sistema multi-robô baseado em aprendizado por reforço profundo. A vazão de dados média foi de $173kbps$ com uma latência de $0.5s$. Isso é, uma rede teoricamente veloz para os sensores considerados. A implementação utilizando ROS2 permitiu avaliar as métricas de comunicação, validando a eficiência da transmissão dos estados e transições dos agentes durante a navegação.

Os resultados indicam, também, que o compartilhamento de experiências entre os robôs pode melhorar significativamente a convergência do aprendizado, permitindo que os agentes aprendam políticas mais rapidamente do que abordagens isoladas, como o *Pure DDQN*. No entanto, algumas limitações foram identificadas, incluindo o uso excessivo de memória durante as simulações, que impediu a continuidade dos testes por períodos mais longos. O $MARL_{Q_{DDS}}$ proposto superou em termos de performance o MARL-Rand e o DDQN (*Pure DDQN*) apresentando uma taxa de sucesso superior: 18% em relação ao MARL-Rand e 23.33% em relação ao *Pure DDQN*. Dessa forma, como estudos futuros, recomenda-se aprimorar a seleção das experiências compartilhadas, por exemplo, através de uma rede neural "crítica".

VI. AGRADECIMENTOS

Os autores agradecem ao Centro de Excelência em Redes Inteligentes Sem Fio e Serviços Avançados (CERISE) e à Fundação de Amparo à Pesquisa do Estado de Goiás (FAPEG) pelo apoio e financiamento à pesquisa.

REFERÊNCIAS

[1] I. Jawhar, N. Mohamed, J. Wu, and J. Al-Jaroodi, "Networking of multi-robot systems: Architectures and requirements," *Journal of Sensor and Actuator Networks*, vol. 7, no. 4, 2018. [Online]. Available: https://www.mdpi.com/2224-2708/7/4/52

[2] A. A. Rusu, S. G. Colmenarejo, C. Gulcehre, G. Desjardins, J. Kirkpatrick, R. Pascanu, V. Mnih, K. Kavukcuoglu, and R. Hadsell, "Policy distillation," 2016. [Online]. Available: https://arxiv.org/abs/1511.06295

[3] J. Jiang, K. Su, and Z. Lu, "Fully decentralized cooperative multi-agent reinforcement learning: A survey," 2024. [Online]. Available: https://arxiv.org/abs/2401.04934

[4] Y. Yi, G. Li, Y. Wang, and Z. Lu, "Learning to share in multi-agent reinforcement learning," 2022. [Online]. Available: https://arxiv.org/abs/2112.08702

[5] M. P. Manuel, M. Faied, and M. Krishnan, "A novel lora lpwan-based communication architecture for search rescue missions," *IEEE Access*, vol. 10, pp. 57 596–57 607, 2022.

[6] C. D. d. S. Bezerra, A. A. Cardoso, and F. H. T. Vieira, "Utilizing autoencoders for latent representation and efficient transmission of lidar data via lora in ros," *IEEE Internet of Things Journal*, vol. 12, no. 5, pp. 4579–4590, 2025.

[7] K. Zhang, Z. Yang, H. Liu, T. Zhang, and T. Başar, "Fully decentralized multi-agent reinforcement learning with networked agents," 2018. [Online]. Available: https://arxiv.org/abs/1802.08757

[8] S. Macenski, T. Foote, B. Gerkey, C. Lalancette, and W. Woodall, "Robot operating system 2: Design, architecture, and uses in the wild," *Science Robotics*, vol. 7, no. 66, May 2022. [Online]. Available: http://dx.doi.org/10.1126/scirobotics.abm6074

[9] J. Zhang, X. Yu, S. Ha, J. Peña Queralta, and T. Westerlund, "Comparison of middlewares in edge-to-edge and edge-to-cloud communication for distributed ros 2 systems," *Journal of Intelligent amp; Robotic Systems*, vol. 110, no. 4, 2024. [Online]. Available: http://dx.doi.org/10.1007/s10846-024-02187-z

[10] H. van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double q-learning," 2015. [Online]. Available: https://arxiv.org/abs/1509.06461

[11] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," 2013. [Online]. Available: https://arxiv.org/abs/1312.5602

[12] C. D. de Sousa Bezerra, F. H. Teles Vieira, and D. P. Queiroz Carneiro, "Autonomous robotic navigation approach using deep q-network late fusion and people detection-based collision avoidance," *Applied Sciences*, vol. 13, no. 22, 2023. [Online]. Available: https://www.mdpi.com/2076-3417/13/22/12350

[13] S. B. Kamtam, Q. Lu, F. Bouali, O. C. L. Haas, and S. Birrell, "Network latency in teleoperation of connected and autonomous vehicles: A review of trends, challenges, and mitigation strategies," *Sensors*, vol. 24, no. 12, 2024. [Online]. Available: https://www.mdpi.com/1424-8220/24/12/3957