

UNIVERSIDADE FEDERAL DO RIO GRANDE - FURG
CENTRO DE CIÊNCIAS COMPUTACIONAIS
CURSO DE ENGENHARIA DE COMPUTAÇÃO

Dissertação de Mestrado

**Aprendizado por Reforço Profundo para Navegação Sem
Mapa de um Veículo Híbrido Aéreo-Aquático**

Ricardo Bedin Grando

Dissertação de Mestrado apresentada à pós-graduação do curso de Engenharia de Computação da Universidade Federal do Rio Grande - FURG, como requisito parcial para a obtenção do grau de Mestre em Engenharia de Computação

Orientador: Prof. Dr. Paulo Lilles Jorge Drews Jr

Rio Grande, 2021

Ficha Catalográfica

G754a Grando, Ricardo Bedin.

Aprendizado por reforço profundo para navegação sem mapa de um veículo híbrido aéreo-aquático / Ricardo Bedin Grando. – 2021.
132 f.

Dissertação (mestrado) – Universidade Federal do Rio Grande – FURG, Programa de Pós-Graduação em Computação, Rio Grande/RS, 2021.

Orientador: Dr. Paulo Lilles Jorge Drews Jr.

1. Aprendizado por Reforço Profundo 2. Navegação Sem Mapa
3. Robô Autônomo 4. Veículo Híbrido I. Drews Jr, Paulo Lilles Jorge
II. Título.

CDU 004

Catálogo na Fonte: Bibliotecário José Paulo dos Santos CRB 10/2344

ATA DE SESSÃO DE DEFESA DE DISSERTAÇÃO DE MESTRADO

Ata nº 07/2021

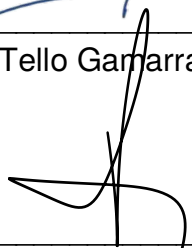
Na data de 19 de abril de 2021, às 13 horas e 30 minutos, ocorreu a Sessão de Defesa de Dissertação de Mestrado de Ricardo Bedin Grando, que apresentou a dissertação intitulada “Aprendizado por Reforço Profundo para Navegação Sem Mapa de um Veículo Híbrido Aéreo-Aquático”, realizada sob a orientação do Prof. Dr. Paulo Lilles Jorge Drews Junior. A banca examinadora foi constituída pelos Profs. Dr. Armando Alves Neto (UFMG), Dr. Daniel Fernando Tello Gamarra (UFSM) e Dra. Silvia da Silva Costa Botelho (FURG), sob a presidência do orientador. Após a apresentação do trabalho, a banca arguiu o candidato e, a seguir, deliberou pela

- (X) aprovação da Dissertação
- () aprovação da Dissertação, sugerindo modificações no texto
- () reprovação da Dissertação

Rio Grande, 19 de abril de 2021.



Prof. Dr. Armando Alves Neto (participação remota)

Prof. Dr. Daniel Fernando Tello Gamarra (participação remota)

Prof. Dra. Silvia da Silva Costa Botelho (participação remota)

Prof. Dr. Paulo Lilles Jorge Drews Junior (participação remota)
Orientador

Aos que pensam grande.

AGRADECIMENTOS

Agradeço minha família, minha mãe Arleide, meu Pai Amélio e minha irmã Juliana por toda a ajuda e suporte em mais essa conquista. Agradeço também meu orientador Paulo Drews-Jr por ter me guiado nesse trabalho de crescimento e amadurecimento. Por fim, agradeço a todos os membros da equipe Hydrone e do Nautech por terem me ajudado durante essa caminhada.

Don't try
— CHARLES BUKOWSKI

RESUMO

GRANDO, Ricardo Bedin. **Aprendizado por Reforço Profundo para Navegação Sem Mapa de um Veículo Híbrido Aéreo-Aquático**. 2021. 132 f. Dissertação (Mestrado) – Engenharia de Computação. Universidade Federal do Rio Grande - FURG, Rio Grande.

Realizar a navegação entre meios é uma tarefa desafiadora para robôs móveis híbridos, especialmente em cenários com obstáculos. Este trabalho apresenta uma abordagem baseada em aprendizado por reforço profundo (Deep-RL) para navegação autônoma de um tipo específico de robô móvel híbrido: um Veículo Híbrido Tipo Ar-Água (HUAUV). No contexto geral de robôs móveis, abordagens baseadas em informação visual são mais comuns. Entretanto, elas possuem alta sensibilidade a variação de luz e a turbidez, além de demandarem *hardware* com alta capacidade de processamento que são geralmente pesadas e difíceis de serem embarcadas em HUAUVs. Além disso, abordagens baseadas em informação visual dependem de sensores precisos para realizar uma trajetória sem colisões e o trabalho de mapeamento é inevitável para qualquer cenário. Ao invés disso, a abordagem proposta utiliza somente informação de sensores de distância e de informações relativas à localização do veículo para realizar a navegação. Resultados obtidos mostram que é possível realizar navegação sem mapa do início ao fim sem usar nenhum tipo de operação manual, somente utilizando os agentes baseados em Deep-RL. A navegação dos agentes treinados são comparados com a navegação sem mapa realizada por um algoritmo clássico - BUG2. A abordagem é baseada em dois métodos do estado da arte para navegação sem mapa de robôs terrestres: Política de Gradiente Determinístico Profundo (DDPG) e *Soft Actor-Critic* (SAC).

Palavras-chave: Aprendizado por Reforço Profundo, Navegação Sem Mapa, Robô Autônomo, Veículo Híbrido.

ABSTRACT

GRANDO, Ricardo Bedin. **Deep Reinforcement Learning for Mapless Navigation of a Hybrid Unmanned Aerial Underwater Vehicle**. 2021. 132 f. Dissertação (Mestrado) – Engenharia de Computação. Universidade Federal do Rio Grande - FURG, Rio Grande.

Performing navigation between environments is a challenging task for hybrid mobile robots, especially in scenarios with obstacles. This work presents an approach based on deep reinforcement learning (Deep-RL) for autonomous navigation of a specific type of hybrid mobile robot: a Hybrid Unmanned Aerial Underwater Vehicle (HUAUV). For the context of mobile robots in general, image-based sensing approaches are the most common. However, they have high sensibility to light and turbidity, besides the demand for high processing power hardware that are usually heavy and difficult to embed into a HUAUV. Besides that, image-based sensing methods rely on precise sensors for performing a collision-free trajectory and the map making work is inevitable for any scenario. Instead, our approach uses only information from range findings sensors and the vehicle's relative localization data to perform the navigation. Results show that it is possible to perform mapless navigation from start to finish without any manual operation, only using Deep-RL-based agents. The navigation performed by the agents is compared with the mapless navigation performed by a classic algorithm - BUG2. We based our approach in two state-of-art Deep-RL techniques for mapless navigation of terrestrial robots: Deep Deterministic Policy Gradient (DDPG) and Soft Actor Critic (SAC).

Keywords: Deep-RL, Mapless Navigation, Autonomous Robot, Hybrid Vehicle.

LISTA DE FIGURAS

1	Arquitetura e Organização do Gazebo mostrando a estrutura de um modelo e como o mesmo interage com o mundo e com demais aplicações ROS.	33
2	Comparação do robô Pioneer2AT descrito no Gazebo e real.	34
3	Veículo simulado baseado no veículo real Hydrone.	37
4	<i>Frame</i> base do veículo.	38
5	Veículo Hydrone com rotores.	39
6	Sensor de distância acoplado ao veículo.	41
7	a) Imagem gerada pelo <i>shader</i> , b) imagem acústica resultante. Apresentado em Longaray et al. 2017.	43
8	Representação gráfica referente as etapas realizadas desde a extração de informações de uma cena em OSG até a atribuição de intensidades de cada <i>bin</i> em um <i>beam</i> de sonar. Apresentado em Cerqueira et al. [2016].	43
9	Estrutura básica do ambiente descrito.	45
10	Cenários de simulação.	45
11	Exemplo de navegação sem mapa orientada a alvo.	48
12	Veículo treinando em um dos cenários (esquerda) e o modelo básico de estrutura com entradas e saídas baseado nos métodos SAC e DDPG (direita) para o contexto 2D.	53
13	Veículo treinando em um dos cenários (esquerda) e o modelo básico de estrutura com entradas e saídas baseado nos métodos SAC e DDPG (direita) para o contexto 3D.	53
14	Entradas e saídas da rede dos agentes no contexto 2D.	54
15	Estrutura da rede para o contexto 2D.	55
16	Estrutura da rede para o contexto 3D.	55
17	Recompensa ao longo dos episódios para cada abordagem no ambiente aéreo no contexto 2D.	58
18	Caminho realizado durante as 100 tentativas na primeira tarefa no contexto 2D aéreo.	60
19	Ações dos agentes realizadas durante as 100 tentativas para a primeira tarefa no contexto 2D aéreo. Em vermelho, a média do delta yaw e em azul da velocidade linear. A sombra representa o desvio padrão.	61
20	Caminho realizado durante as 100 tentativas na segunda tarefa no contexto 2D aéreo.	63

21	Ações dos agentes realizadas durante as 100 tentativas para a segunda tarefa no contexto 2D aéreo. Em vermelho a média do delta yaw e em azul da velocidade linear. A sombra representa o desvio padrão.	64
22	Caminho realizado durante as 100 tentativas na primeira tarefa no contexto 2D subaquático.	67
23	Ações dos agentes realizadas durante as 100 tentativas para a primeira tarefa no contexto 2D subaquático. Em vermelho a média do delta yaw e em azul da velocidade linear. A sombra representa o desvio padrão.	68
24	Caminho realizado durante as 100 tentativas na segunda tarefa no contexto 2D subaquático.	69
25	Ações dos agentes realizadas durante as 100 tentativas para a segunda tarefa no contexto 2D subaquático. Em vermelho a média do delta yaw e em azul da velocidade linear. A sombra representa o desvio padrão.	70
26	Recompensa ao longo dos episódios para cada abordagem no ambiente aéreo no contexto 3D.	71
27	Caminho realizado durante as 100 tentativas na primeira tarefa no contexto 3D.	74
28	Ações dos agentes realizadas durante as 100 tentativas para a primeira tarefa no contexto 3D. Em vermelho a média do delta yaw, em azul da velocidade linear e em verde a velocidade de altitude. A sombra representa o desvio padrão.	75
29	Caminho realizado durante as 100 tentativas na segunda tarefa no contexto 3D.	76
30	Ações dos agentes realizadas durante as 100 tentativas para a segunda tarefa no contexto 3D. Em vermelho a média do delta yaw, em azul da velocidade linear e em verde a velocidade de altitude. A sombra representa o desvio padrão.	77
31	Recompensa ao longo dos episódios para cada abordagem no ambiente aéreo no contexto 3D híbrido.	78
32	Caminho realizado durante 1 das 100 tentativas no contexto 3D híbrido no primeiro cenário.	79
33	Caminho realizado durante 1 das 100 tentativas no contexto 3D híbrido no segundo cenário.	81
34	Ações dos agentes realizadas durante as 100 tentativas no contexto 3D híbrido partindo do ar em direção à água. Em vermelho a média do delta yaw em azul da velocidade linear e em verde a velocidade de altitude. A sombra representa o desvio padrão.	82
35	Ações dos agentes realizadas durante as 100 tentativas no contexto 3D híbrido partindo do meio subaquático em direção ao ar. Em vermelho a média do delta yaw, em azul da velocidade linear e em verde a velocidade de altitude. A sombra representa o desvio padrão.	83
36	Recompensa ao longo dos episódios para cada abordagem no ambiente aéreo no contexto 3D.	86
37	Recompensa ao longo dos episódios para cada estrutura no contexto 2D no primeiro cenário.	87

38	Recompensa ao longo dos episódios para cada estrutura no contexto 2D no segundo cenário.	88
39	Caminho realizado durante as 100 tentativas na primeira tarefa no contexto 2D para o primeiro cenário.	90
40	Caminho realizado durante as 100 tentativas na primeira tarefa no contexto 2D para o segundo cenário.	91
41	Ações dos agentes realizadas durante as 100 tentativas para a primeira tarefa no contexto 2D no primeiro cenário. Em vermelho a média do delta yaw e em azul da velocidade linear. A sombra representa o desvio padrão.	93
42	Ações dos agentes realizadas durante as 100 tentativas para a primeira tarefa no contexto 2D no segundo cenário. Em vermelho a média do delta yaw e em azul da velocidade linear. A sombra representa o desvio padrão.	94
43	Caminho realizado durante as 100 tentativas na segunda tarefa no contexto 2D para o primeiro cenário.	96
44	Caminho realizado durante as 100 tentativas na segunda tarefa no contexto 2D para o segundo cenário.	97
45	Ações dos agentes realizadas durante as 100 tentativas para a segunda tarefa no contexto 2D no primeiro cenário. Em vermelho a média do delta yaw e em azul da velocidade linear. A sombra representa o desvio padrão.	99
46	Ações dos agentes realizadas durante as 100 tentativas para a segunda tarefa no contexto 2D no segundo cenário. Em vermelho a média do delta yaw, em azul a velocidade linear e em verde a velocidade de altitude. A sombra representa o desvio padrão.	100
47	Recompensa ao longo dos episódios para cada estrutura no contexto 3D no primeiro cenário.	101
48	Recompensa ao longo dos episódios para cada estrutura no contexto 3D no segundo cenário.	102
49	Caminho realizado durante as 100 tentativas na primeira tarefa no contexto 2D para o primeiro cenário para a abordagem DDPG.	104
50	Caminho realizado durante as 100 tentativas na primeira tarefa no contexto 2D para o primeiro cenário para a abordagem SAC.	105
51	Caminho realizado durante as 100 tentativas na primeira tarefa no contexto 2D para o segundo cenário para a abordagem DDPG.	106
52	Caminho realizado durante as 100 tentativas na primeira tarefa no contexto 2D para o segundo cenário para a abordagem SAC.	107
53	Ações dos agentes realizadas durante as 100 tentativas para a primeira tarefa no contexto 3D no primeiro cenário. Em vermelho a média do delta yaw, em azul da velocidade linear e em verde a velocidade de altitude. A sombra representa o desvio padrão.	109
54	Ações dos agentes realizadas durante as 100 tentativas para a primeira tarefa no contexto 3D no segundo cenário. Em vermelho a média do delta yaw, em azul da velocidade linear e em verde a velocidade de altitude. A sombra representa o desvio padrão.	110
55	Caminho realizado durante as 100 tentativas na primeira tarefa no contexto 2D para o primeiro cenário para a abordagem DDPG.	112

56	Caminho realizado durante as 100 tentativas na primeira tarefa no contexto 2D para o primeiro cenário para a abordagem SAC.	113
57	Caminho realizado durante as 100 tentativas na primeira tarefa no contexto 2D para o segundo cenário para a abordagem DDPG.	114
58	Caminho realizado durante as 100 tentativas na primeira tarefa no contexto 2D para o segundo cenário para a abordagem SAC.	115
59	Ações dos agentes realizadas durante as 100 tentativas para a primeira tarefa no contexto 3D no primeiro cenário. Em vermelho a média do delta yaw, em azul da velocidade linear e em verde a velocidade de altitude. A sombra representa o desvio padrão.	117
60	Ações dos agentes realizadas durante as 100 tentativas para a primeira tarefa no contexto 3D no segundo cenário. Em vermelho a média do delta yaw, em azul da velocidade linear e em verde a velocidade de altitude. A sombra representa o desvio padrão.	118
61	Caminho realizado durante as 100 tentativas com a função de ativação <i>Hardtanh</i> no contexto 2D.	120
62	Caminho realizado durante as 100 tentativas com a função de ativação <i>Hardtanh</i> no contexto 3D.	121
63	Recompensa comparativa do estudo de validação.	122
64	Comparação do tempo médio para o contexto 2D no primeiro cenário (mais estável) na primeira tarefa.	123
65	Comparação da distância média para o contexto 2D no primeiro cenário (mais estável) na segunda tarefa.	124

LISTA DE TABELAS

1	Parâmetros dos Rotores.	38
2	Parâmetros do Lidar utilizado.	42
3	Parâmetros de simulação do sonar FLS.	43
4	Parâmetros de ruído para a dinâmica do ambiente de simulação. . . .	44
5	Parâmetros de ruído Ornstein-Uhlenbeck usado na abordagem baseada em DDPG.	51
6	Hiperparâmetros utilizados no sistema de reforço.	56
7	<i>Waypoints</i> contexto 2D cenário 1.	59
8	<i>Waypoints</i> contexto 2D cenário 2.	59
9	Estatísticas da navegação orientada a alvo para o contexto 2D aéreo. .	59
10	Estatísticas da navegação sem mapa orientada a múltiplos pontos para o contexto 2D aéreo.	62
11	<i>Waypoints</i> contexto 2D subaquático cenário 1.	65
12	<i>Waypoints</i> contexto 2D cenário 2.	65
13	Estatísticas da navegação orientada a alvo para o contexto 2D subaquático.	66
14	Estatísticas da navegação sem mapa orientada a múltiplos pontos para o contexto 2D subaquático.	66
15	<i>Waypoints</i> contexto 3D cenário 1.	71
16	<i>Waypoints</i> contexto 3D cenário 2.	72
17	Estatísticas da navegação orientada a alvo para o contexto 3D.	72
18	Estatísticas da navegação sem mapa orientada a múltiplos pontos para o contexto 3D.	72
19	Estatísticas da navegação orientada a alvo para o contexto 3D híbrido. .	79
20	Estatísticas da navegação orientada a alvo para o contexto 2D da abordagem baseada em DDPG na primeira tarefa.	89
21	Estatísticas da navegação orientada a alvo para o contexto 2D da abordagem baseada em SAC na primeira tarefa.	89
22	Estatísticas da navegação orientada a alvo para o contexto 2D da abordagem baseado em DDPG na segunda tarefa.	95
23	Estatísticas da navegação orientada a alvo para o contexto 2D da abordagem baseado em SAC na segunda tarefa.	95
24	Estatísticas da navegação orientada a alvo para o contexto 3D da abordagem baseada em DDPG na primeira tarefa.	103

25	Estatísticas da navegação orientada a alvo para o contexto 3D da abordagem baseada em SAC na primeira tarefa.	103
26	Estatísticas da navegação orientada a alvo para o contexto 2D da abordagem baseado em DDPG na segunda tarefa.	111
27	Estatísticas da navegação orientada a alvo para o contexto 2D da abordagem baseada SAC na segunda tarefa.	111
28	Estatísticas da navegação orientada a alvo nas abordagens com <i>Hard-tanh</i>	119
29	Estatísticas da navegação orientada a alvo para as abordagens com <i>Hardtanh</i> na segunda tarefa.	119

LISTA DE ABREVIATURAS E SIGLAS

UV	Veículo não Tripulado
UGV	Veículo Terrestre não Tripulado
UUVs	Veículo Subaquático não Tripulado
ROVs	Veículo Remotamente Operado
HUVs	Veículo Híbrido não Tripulado
AI	Inteligência Artificial
Deep-RL	Aprendizado por Reforço Profundo
SAC	<i>Soft Actor-Critic</i>
DDPG	Política de Gradiente Determinística Profunda
GPS	Sistema de Posição Global
IMU	Unidade de Medição Inercial
SLAM	Localização Simultânea e Mapeamento
PID	Proporcional Integral e Derivativo
DoF	Graus de Liberdade
ODE	<i>Open Dynamics Engine</i>
SIFT	Transformação de recurso invariável em escala
D4PG	Política de Gradiente Determinística Distributiva Distribuída Profunda
POMDP	Processo de Decisão de Markov Parcialmente Observável
RL	Aprendizado por Reforço
DL	Aprendizado Profundo
DPG	Política do Gradiente Determinística
SPG	Política do Gradiente Estocástico
OSG	<i>Open Scene Graph</i>
OSGB	<i>Open Scene Graph Binary</i>
LSTM	<i>Long Short-Term Memory</i>
CNN	Rede Neural Convolucional

MLP	<i>Multi Layer Perceptron</i>
ROS	Sistemas Operacional de Robôs
FLS	<i>Forward Looking Sonar</i>

SUMÁRIO

1	INTRODUÇÃO	19
1.1	Contextualização	19
1.2	Motivação	21
1.3	Problema	22
1.4	Objetivo Geral	22
1.5	Objetivos Específicos	22
1.6	Contribuição Científicas	23
1.7	Seções do trabalho	23
2	ESTADO DA ARTE	24
2.1	Estado da Arte para HUAUVs	24
2.2	Estado da Arte para UAVs	27
2.3	Estado da Arte para UUVs	28
3	SETUP EXPERIMENTAL	31
3.1	Interface de Simulação	31
3.1.1	Gazebo	32
3.1.2	RotorS	35
3.1.3	UUV Sim	36
3.2	Desenvolvimento do Hydrone simulado	36
3.2.1	Descrição Quadrotor Base e Parâmetros Aéreos	37
3.2.2	Parâmetros Subaquáticos	39
3.2.3	Parâmetros de Controle	40
3.2.4	Sensor de distância - Lidar	41
3.2.5	Sensor de distância - Sonar	42
3.3	Desenvolvimento dos Ambientes de simulação	44
3.4	Algoritmo BUG2	45
4	METODOLOGIA	47
4.1	Navegação sem mapa orientada a ponto alvo	47
4.2	Aprendizado por Reforço Profundo	49
4.2.1	Abordagem baseada em DDPG - NDRL-D	50
4.2.2	Abordagem baseada em SAC - NDRL-S	51
4.3	Estrutura dos agentes	52
4.3.1	Estrutura da rede	54
4.4	Função de Reforço	55

5	RESULTADOS	57
5.1	Contexto 2D Aéreo	58
5.1.1	Navegação sem mapa orientada a alvo	59
5.1.2	Navegação sem mapa orientada a múltiplos pontos	61
5.2	Contexto 2D Subaquático	65
5.2.1	Navegação sem mapa orientada a alvo	65
5.2.2	Navegação sem mapa orientada a múltiplos pontos	66
5.3	Contexto 3D	71
5.3.1	Navegação sem mapa orientada a alvo	71
5.3.2	Navegação sem mapa orientada a múltiplos pontos	72
5.4	Contexto 3D Híbrido	78
5.5	Discussão	84
6	ESTUDO DE VALIDAÇÃO	85
6.1	Contexto 2D	87
6.1.1	Navegação sem mapa orientada a alvo	89
6.1.2	Navegação sem mapa orientada a múltiplos pontos	95
6.2	Contexto 3D	101
6.2.1	Navegação sem mapa orientada a alvo	103
6.2.2	Navegação sem mapa orientada a múltiplos pontos	111
6.3	Função de Ativação	119
6.4	Discussão	122
7	CONSIDERAÇÕES FINAIS E PERSPECTIVAS	125
	REFERÊNCIAS	126

1 INTRODUÇÃO

1.1 Contextualização

O desenvolvimento de veículos não tripulados tem-se expandido muito nos últimos anos, sendo que o baixo custo oferecido por muitos tem proporcionado inclusive o uso comercial deste tipo de veículo. Vários modelos foram propostos por pesquisadores com diferentes propósitos, tais como os modelos de [Xiang and Tian, 2011], Miranda et al. [2013], Miranda et al. [2013], Xiongkui et al. [2017], e Shen et al. [2018], Edge et al. [2020]. Esse tipo de veículo, também chamado de UV (do inglês, *Unmanned Vehicle*), pode ser definido como sendo qualquer veículo cujo controle é feito por uma pessoa à distância ou por alguma inteligência embarcada que o permita funcionar de forma autônoma, sem que algum ser humano atue dentro dele.

Os UVs podem ser categorizados de acordo com o ambiente no qual foi projetado: aéreo, terrestre e subaquático. Os UVs terrestres ou UGVs (do inglês, *Unmanned Ground Vehicles*) são os mais comuns e um exemplo desse tipo de veículo é o *Pioneer 3-AT* [MobileRobots, 2015]. Os UAVs (do inglês, *Unmanned Aerial Vehicles*) e os ROVs (do inglês, *Remotely Operated Vehicles*) são dois exemplos de UVs para o ambiente aéreo e aquático, respectivamente. O primeiro pertence à categoria de veículos aéreos não tripulados, enquanto que o segundo pertence à categoria de veículos subaquáticos não tripulados ou UUVs (do inglês, *Unmanned Underwater Vehicles*). Os UVs são muito empregados para realizar pesquisas, monitoramento e até mesmo para lazer.

O meio de atuação, portanto, define a categoria na qual um UV pertence. Entretanto, podem existir situações em que exista a necessidade de atuação ou sensoriamento em mais de um meio. A inspeção de uma plataforma de petróleo, por exemplo, precisa ser realizada na parte emersa e na parte submersa. Outro exemplo são estruturas em regiões pantanosas e/ou alagadiças, tais como a região da floresta amazônica, onde estas podem até se tornar inacessíveis ou submersas dependendo da pluviosidade de determinadas épocas do ano. Para ambos cenários, UAVs ou UUVs não são capazes de realizar a atuação ou monitoramento requisitado por completo.

Para tal, nos últimos anos tem aumentado o interesse no desenvolvimento de veículos

híbridos não tripulados HUVs (do inglês, *Hybrid Unmanned Vehicles*). Esses veículos buscam unir e explorar as vantagens e desvantagens dos UAVs, UUVs e UGVs (até por isso a distinção de híbrido). Em específico, o estudo acerca de um veículo híbrido capaz de atuar no meio aéreo e subaquático - HUAUV (do inglês, *Hybrid Unmanned Aerial Underwater Vehicle*) - tem despontado como um grande desafio na ciência devido ao contraste de atuação e percepção que os meios envolvidos demandam, onde muitos dos sensores e atuadores somente se aplicam a um determinado meio ou requerem adaptações mecânicas para um funcionamento correto.

Vários modelos de HUAUVs já foram propostos na ciência e também comercialmente. Algumas das primeiras propostas, como a de Drews et al. [2014], Yang et al. [2015] e Maia et al. [2015], mostram um aumento por esse tipo de veículo na última década e um foco por um HUAUV com uma estrutura do tipo quadrotor. Um veículo híbrido com esse tipo de formato propicia a realização de variadas tarefas, tais como inspeção e manutenção, mapeamento e coleta de dados, e também, vigilância, busca e resgate.

As tarefas de inspeção e manutenção, por exemplo, são tarefas corriqueiras em plataformas de petróleo, navios e demais estruturas localizadas em ambientes marinhos e subaquáticos. A corrosão gerada pela água e o desbalanceamento natural gerado pelo ambiente dinâmico de tais estruturas geram uma constante demanda de atuação externa a fim de corrigir problemas e realizar reparos. Essa demanda geralmente é abordada de forma separada no meio subaquático, sendo que a inspeção é geralmente realizada por ROVs e a manutenção por mergulhadores, enquanto que no meio aéreo tais tarefas são comumente realizadas por UAVs e operários com uso de andaimes, respectivamente. Outra situação relevante onde também ocorre demanda por inspeção são os gasodutos em regiões alagadas da floresta amazônica. Alguns trechos desses gasodutos ficam em localidades de difícil acesso, podendo estar submersos ou parcialmente submersos por tempo indeterminado, de acordo com a volatilidade típica desse ambiente tropical. Estes são dois exemplos de tarefas que motivam o desenvolvimento de HUAUV, pois são difíceis e custosas para serem realizadas.

No ambiente amazônico, assim como na Antártica, muitas ações de mapeamento e coleta de dados também costumam ser realizadas por pesquisadores e empresas que buscam estudar ou explorar tais ambientes de alguma forma. No ambiente antártico, especificamente, algumas dessas ações costumam ser realizadas em regiões costeiras a fim de estudar de forma mais aprofundada os derretimentos sazonais e o derretimento contínuo gerado pelo aquecimento dos oceanos. Um veículo que pudesse transitar entre os dois meios - subaquático e aéreo - poderia ser capaz de fornecer novos tipos de dados ou complementar as informações coletadas, oferecendo uma perspectiva distinta e possivelmente mais aprofundada.

Aplicações de vigilância, busca e resgate também podem ser aprimoradas usando HUAUVs. O exemplo dos meninos presos em uma caverna na Tailândia no ano de 2018 co-

moveu o mundo, apresentando grandes desafios para que os mesmos fossem resgatados. Um HUAUV teria sido de grande ajuda nesse contexto, onde muitos trechos da caverna ficaram submersos pela água das chuvas.

O motivo para que HUAUVs não tenham sido utilizados no auxílio ao resgate dos meninos na Tailândia ou que não estejam sendo utilizados para otimizar o trabalho dos operários e mergulhadores na inspeção e manutenção de estruturas marítimas se deve ao fato desses veículos ainda se encontrarem em fase de conceito e prototipação, onde vários aspectos nas partes mecânica, *hardware* e *software* ainda precisam ser definidos e/ou abordados. Dentre os problemas mais primitivos e que talvez se detenha com maior foco no escopo de *software* e *hardware* é o conceito de navegação para esse tipo de veículo. Como esse veículo navegaria de uma posição para outra no espaço, que tipo de inteligência e a informação de quais sensores que seriam utilizados para tal uma vez que a estrutura do veículo estivesse mecanicamente estável são questões a serem abordadas e discutidas.

1.2 Motivação

Saber onde estar, para onde ir e como chegar ao destino é uma forma resumida de como o escopo do problema da navegação pode ser definido [Alves et al., 2011]. Enquanto que os dois primeiros questionamentos podem ser respondidos usando um sensoriamento adequado, o terceiro questionamento para qualquer veículo autônomo precisa ser abordado utilizando algum tipo de sistema que consiga realizar um planejamento do trajeto entre o local de partida e o de chegada [Alves et al., 2011].

A discussão sobre esse sistema de planejamento compreende uma área extensa da ciência cuja solução envolve desde algoritmos matemáticos simples até complexos sistemas de AI, tais como Deep-RL. Enquanto que as abordagens mais primitivas para responder o “como chegar lá” geralmente são desenvolvidas para o contexto de um tipo de veículo específico ou até mesmo para um determinado veículo, uma abordagem usando AI tende a ser mais genérica e mais abstrata.

É possível validar essa afirmação através do número crescente de abordagens utilizando Deep-RL nesse problema para diversos veículos [Li, 2017]. Inclusive, algumas das técnicas de Deep-RL desenvolvidas, tais como SAC (do inglês *Soft Actor Critic*) [Haarnoja et al., 2018] e DDPG (do inglês *Deep Deterministic Policy Gradient*) [Lillicrap et al., 2015], possuem foco na solução de problemas que envolvem controle contínuo, inerente ao problema da navegação para qualquer veículo.

No contexto da navegação para um HUAUV, especificamente, a abordagem tende a ser mais complexa. Muitos dos sensores e atuadores funcionam de forma diferente em cada um dos ambientes ou até mesmo só funcionam em um ambiente específico, como é o caso do GPS (do inglês *Global System Position*) que é fundamental para a localização de qualquer veículo aéreo ou terrestre mas que não funciona adequadamente

quando submerso na água. Interpretar o problema da navegação para um HUAUV de forma separada para cada ambiente pode ser uma forma eficiente de resolvê-lo como um todo, uma vez que de qualquer jeito o sensoriamento do veículo se difere entre os ambientes. Nesse contexto de maior complexidade de sensoriamento e atuação, uma abordagem que seja mais genérica e que ofereça maior abstração pode ser uma forma eficiente de abordar a problemática, mesmo que realizada de forma independente para cada ambiente.

1.3 Problema

O presente trabalho busca, especificamente, realizar uma discussão a partir do escopo do problema primitivo da navegação para HUAUVs e a inteligência necessária para resolvê-lo. Parte-se de uma abordagem de Inteligência Artificial ou AI (do inglês *Artificial Intelligence*) baseada em Aprendizado por Reforço Profundo ou Deep-RL (do inglês *Deep Reinforcement Learning*) [Li, 2017] aplicado a tarefas que envolvem navegação sem mapa orientada a alvo de um HUAUV simulado a partir de um veículo real nos ambientes aéreo e subaquático.

1.4 Objetivo Geral

Propor abordagens baseadas em Deep-RL para tarefas que envolvam navegação sem mapa orientada a ponto alvo usando um HUAUV em variados cenários aéreos e subaquáticos simulados.

1.5 Objetivos Específicos

- Utilizar um sensor de distância simulado a fim de fazer o veículo "aprender" a navegar e evitar colisão com obstáculos.
- Buscar fazer uma análise comparativa entre as técnicas do estado da arte de Deep-RL propostas ao problema.
- Comparar a navegação sem mapa realizada pelos agentes de Deep-RL com a navegação realizada por um algoritmo clássico comportamental.
- Desenvolvimento de um pacote ROS contendo um veículo híbrido tipo HUAUV simulado no Gazebo.
- Navegação sem mapa aérea em 2D usando somente informações referentes à localização e sensor de distância.
- Navegação sem mapa aérea e navegação subaquática para o mesmo veículo em 3D usando somente informações referentes à localização e sensor de distância.
- Estudo validativo com variadas arquiteturas de rede dos agentes.
- Propor o uso de transferência de aprendizado da navegação aérea para a navegação

subaquática com um sonar simulado.

1.6 Contribuição Científicas

Até o momento, o presente trabalho possui duas contribuições científicas. A primeira foi realizada no Simpósio Latino Americano de Robótica 2020 (IEEE LARS 2020) [Grando et al., 2020], trabalho que envolveu a parte de navegação aérea 2D e que ficou entre os 10 melhores do evento. Outro artigo científico foi publicado na Conferência Internacional de Robótica e Automação (IEEE ICRA 2021) [Grando et al., 2021], envolvendo a navegação híbrida. Atualmente, os autores deste trabalho vem desenvolvendo uma versão estendida do artigo do LARS 2020 para o *Journal of Robotics & Intelligent Systems* (JINT) e também uma versão estendida do ICRA 2021 para o *IEEE Robotics and Automation Letters*.

1.7 Seções do trabalho

A divisão do trabalho foi feita da seguinte forma: Na sequência, o Capítulo 2 apresenta uma revisão do estado da arte sobre trabalhos pertinentes à este trabalho. Após isso, no Capítulo 3 é realizada a descrição do veículo utilizado no ambiente de simulação, bem como uma discussão de ferramentas utilizadas, adaptadas ou desenvolvidas para dar suporte ao trabalho. No Capítulo 4 é apresentada a metodologia proposta. Os resultados obtidos são apresentados no Capítulo 5, enquanto que no Capítulo 6 é apresentado um estudo validativo como complemento final ao trabalho. Por último, no Capítulo 7 as considerações finais e perspectivas futuras são discutidas.

2 ESTADO DA ARTE

Drews et al. [2014] destaca no seu trabalho um histórico acerca de HUAUVs, resumindo as primeiras ideias e protótipos de veículos até a data na qual o trabalho foi publicado. Além desse histórico, Drews et al. [2014] também realiza um estudo comparativo entre os tipos de veículos aéreos e subaquáticos já propostos e/ou desenvolvidos, dando ênfase às características que beneficiam e prejudicam cada modelo. O trabalho conclui que um veículo baseado em uma estrutura do tipo quadrotor é o mais aconselhável para o desenvolvimento de um HUAUV. Outras revisões da literatura conduzidas por Mercado et al. [2019] e Narayanan et al. [2020] também destacam o estágio inicial que o desenvolvimento desse veículo se encontra.

Esse trabalho proposto por Drews et al. [2014] é um dos primeiros acerca de HUAUVs. Atualmente, ainda existem poucos trabalhos desenvolvidos sobre HUAUVs, sendo que a maioria foca na mecânica e estrutura do veículo, havendo, dessa forma, pouco progresso no que diz respeito ao *software* e inteligência para o veículo [Ravell et al., 2018] [Maia et al., 2017] [Mercado et al., 2019]. Essa situação é um dos fatores limitantes para que estudos sobre navegação para HUAUVs, seja com mapa ou sem mapa, sejam realizados. Dessa forma, o problema da navegação sem mapa, muito explorado em robôs móveis terrestres, ainda precisa ser explorado para HUAUVs.

Levando em consideração esse contexto e a forma proposta para abordar o problema da navegação sem mapa no presente trabalho na qual se constitui de usar um HUAUV também de forma individual para cada meio de atuação, o resumo do estado da arte está dividido entre uma Seção de trabalhos relacionados, onde são discutidos com maior ênfase trabalhos sobre HUAUVs e navegação sem mapa para robôs móveis em geral, e uma Seção de estado da arte para navegação sem mapa de UAVs e UUVs. Uma conclusão geral sobre a revisão é feita no final desta Seção.

2.1 Estado da Arte para HUAUVs

Os trabalhos propostos que abordam HUAUVs até o momento o fazem de tal forma que é possível dividi-los em duas categorias: veículos bioinspirados [Stewart et al., 2018],

[Moore et al., 2018], [Tan et al., 2017] e veículos baseados em uma estrutura quadrotor [Drews et al., 2014], [Ma et al., 2018], [Maia et al., 2017], [Maia et al., 2015], [Alzu'bi et al., 2018].

Stewart et al. [2018] apresenta uma proposta de um veículo em formato de pássaro subaquático cuja locomoção se dá via um sistema de propulsão para cada meio. Moore et al. [2018] propôs um protótipo de asa fixa com somente uma hélice, similar a um avião. Tan et al. [2017] introduziu um protótipo de HUAUV de asa fixa com um único sistema de propulsão tanto para o ar como para a água. Ambas propostas buscaram validar a ideia do protótipo, não abordando o problema da navegação.

Os trabalhos mais relevantes acerca de HUAUVs, entretanto, partem de uma abordagem utilizando uma estrutura de quadrotor. Dentre algumas das similaridades que as propostas apresentam está a utilização de uma cápsula no centro do veículo onde a parte eletrônica e elétrica é encapsulada e a utilização de hélices aéreas no sistema de propulsão. Essas duas características fazem com que o veículo possa atuar e transitar entre os meios sem a necessidade de alterações mecânicas. O HUAUV descrito em simulação neste trabalho levou em consideração essa convergência de propostas, sendo portanto baseado também em uma estrutura de quadrotor.

Além do protótipo com quatro hélices aéreas e quatro hélices subaquáticas de Drews et al. [2014], Ma et al. [2018] também apresenta um protótipo com a mesma configuração de hélices. Maia et al. [2017], Maia et al. [2015] e Alzu'bi et al. [2018] apresentam propostas sem hélices subaquáticas somente aéreas, sendo que Maia et al. [2015] e Maia et al. [2017] partem de uma abordagem com oito hélices aéreas e Alzu'bi et al. [2018] utiliza somente quatro hélices aéreas. Todos os trabalhos se detêm na questões de estrutura, mecânica e controle do veículo, sem abordar o problema da navegação. O HUAUV descrito em simulação neste trabalho parte de uma abordagem com quatro hélices aéreas que também atuam no ambiente subaquático.

No contexto de navegação sem mapa em geral, Guerrero et al. [2005] foi um dos primeiros a abordar esse problema em robótica móvel através da computação de homografias entre a imagem atual de uma câmera monocular com uma imagem obtida anteriormente em uma fase de treinamento. O movimento do robô é definido pela correção das linhas extraídas da imagem na homografia. Guerrero et al. [2005] aplicou a abordagem em uma cadeira de rodas com sucesso. De modo geral, Guerrero et al. [2005] desenvolveu seu trabalho com foco para o contexto de um robô móvel terrestre baseado essencialmente em informação visual.

Guzel and Bicker [2009] também utilizou informações de imagens para realizar navegação sem mapa. Guzel and Bicker [2009] utilizou uma abordagem baseada no algoritmo SIFT (do inglês Scale Invariant Feature Transformation) [Ng and Henikoff, 2003]. Guzel and Bicker [2009] utilizou o robô terrestre Pioneer 3-DX para realizar os testes. O robô tenta entender o ambiente através da extração *features* tomadas de uma sequência de

imagens. O estudo realizado por Guzel and Bicker [2009] sugere que a combinação de ambas abordagens produz uma solução robusta para realizar navegação sem mapa com utilização de informação visual.

Os trabalhos mais recentes sobre navegação sem mapa para robôs móveis, entretanto, se baseiam em alguma abordagem que utiliza Deep-RL. Algumas das técnicas de Deep-RL já desenvolvidas, tais como DDPG e SAC, têm sido cada vez mais utilizadas em aplicações na robótica [Kober et al., 2013], [Kormushev et al., 2013], indo além do problema da navegação. Alguns dos trabalhos mais recentes e promissores na robótica envolvem alguma abordagem com Deep-RL [Gu et al., 2017], [Mahmood et al., 2018]. Alguns trabalhos recentes aplicados a robôs móveis terrestres alcançaram bons resultados na exploração de ações de controle contínuo para navegação sem mapa e também para outras tarefas como seguidor de caminho [Tai et al., 2017], [Jesus et al., 2019].

Tai et al. [2017] utilizou um robô móvel terrestre e Deep-RL para realizar navegação sem mapa em ambientes com obstáculos. O trabalho desenvolvido por Tai et al. [2017] serviu de base para o desenvolvimento deste trabalho. A estrutura do agente é semelhante, variando somente no número de entradas do sensor de distância onde somente 10 entradas de um lidar foram utilizadas. A estrutura da rede e a função de reforço também são similares. Tai et al. [2017] utilizou uma abordagem baseada em DDPG e a versão distribuída de DDPG, D4PG (do inglês Distributed Distributional Deterministic Policy Gradients). O trabalho desenvolvido por [Jesus et al., 2019] possui muitas semelhanças com que que Tai et al. [2017] desenvolveu, diferenciando-se no foco em cenários dinâmicos.

Fan et al. [2018] utilizou Deep-RL para realizar navegação sem mapa para quatro diferentes robôs móveis terrestres. Fan et al. [2018] propôs uma abordagem semelhante à de Tai et al. [2017], utilizando 512 leituras de distâncias de um lidar. Fan et al. [2018] utilizou uma versão baseada em DDPG com foco em evitar colisões em ambientes com pessoas em movimento. Fan et al. [2018] realizou o treinamento do mesmo agente em múltiplos cenários no simulador Stage [Gerkey et al., 2003]. A estrutura de rede possui duas camadas convolucionais para abstrair o número elevado de leituras de distâncias antes de concatenar em uma camada completamente conectada com as demais informações do estado atual. Uma estrutura de rede alternativa com camadas convolucionais inspirada em Fan et al. [2018] é discutida no estudo de validação.

Jin et al. [2019] abordou o problema da navegação sem mapa em cenários simulados e reais com pessoas em movimento. Jin et al. [2019] utilizou uma abordagem de RL baseada em Processo de Decisão de Markov Parcialmente Observável ou POMDP (do inglês Partially-Observable Markov Decision Process) com função de reforço que leva em consideração a distância do veículo em relação ao alvo, similar à proposta neste trabalho. Jin et al. [2019] também utilizou um Lidar 2D para obter as informações da distância. A abordagem de Jin et al. [2019] apresenta como principal contribuição uma capacidade de navegar prevendo futuros movimentos das pessoas, denominada de política de consciência

de segurança social por Jin et al. [2019].

2.2 Estado da Arte para UAVs

O problema da navegação para UAVs já foi abordado em vários trabalhos e hoje, basicamente, uma navegação autônoma entre um ponto de partida e um ponto alvo para esse tipo de veículo pode ser facilmente realizada utilizando um *mission planner* tanto em ambientes abertos (*outdoor*) quanto fechados (*indoor*). Entretanto, a tarefa da navegação sem mapa para esse tipo de veículo se torna um problema em um cenário com obstáculos, por exemplo. Os *mission planners* são até capazes de evitar a colisão com obstáculos, mas não possui a inteligência necessária para que o veículo seja capaz de desviar obstáculos e encontrar um caminho ideal até o destino pretendido pois ele não é essencialmente capaz de “entender” o cenário sem mapa.

Rodriguez-Ramos et al. [2018] propôs o uso da técnica de Deep-RL DDPG para abordar o problema de aterrissagem em uma plataforma em movimento. Apesar de não abordar diretamente o problema da navegação sem mapa para um UAV, Rodriguez-Ramos et al. [2018] possui uma contribuição interessante para o presente trabalho, uma vez que usou Deep-RL com o RotorS *framework* [Furrer et al., 2016] para um UAV no ambiente de simulação do Gazebo. A descrição da parte aérea do HUAUV utilizado neste trabalho também foi feita utilizando o RotorS *framework* e o treinamento dos agentes de Deep-RL também ocorreu no ambiente de simulação do Gazebo.

Sampedro et al. [2019], por sua vez, também propôs o uso de DDPG para um UAV, mas desta vez aplicado à tarefa de busca e resgate em um ambiente fechado. A tarefa da navegação sem mapa nesse contexto é inerente à tarefa de busca e resgate. Sampedro et al. [2019] propôs isso baseado em um UAV real e simulado e usou um alvo orientado a informação visual. Diferentemente de Sampedro et al. [2019], o presente trabalho não utiliza nenhum tipo de processamento de imagem ou informação visual para o treinamento do agente, somente informação da posição explícita do veículo e das distâncias entre o veículo e o alvo. Uma abordagem utilizando algum tipo de informação visual tende a aumentar consideravelmente o número de estados e a complexidade da rede do agente, uma vez que se, por exemplo, uma imagem com resolução de 16×16 pixels for utilizada como entrada na rede do agente, 256 estados são necessários.

Informação visual também foi utilizada no trabalho de Kang et al. [2019] para realizar navegação sem mapa, baseada no uso híbrido entre dados reais e simulados. A contribuição mais relevante de Kang et al. [2019], entretanto, está relacionado no foco que o trabalho apresentou com relação a evitar colisões com obstáculos. Este mesmo enfoque também é dado no presente trabalho, mas sem a utilização de informação visual.

2.3 Estado da Arte para UUVs

Acima da água, a maioria dos sistemas autônomos se baseiam em algum tipo de comunicação via rádio e/ou posição global (GPS). Nesse ambiente, a localização do veículo pode ser facilmente realizada, geralmente com uma boa resolução. Como consequência, a navegação entre uma posição inicial e uma posição alvo acaba sendo um problema relativamente fácil. Entretanto, embaixo da água ocorre uma rápida atenuação do sinal de GPS e rádio frequência [Paull et al., 2013] [Machado et al., 2016], o que torna o uso desses sensores bastante limitado. Nesse contexto, o problema da navegação para UUVs é muito mais desafiador quando comparado à UAVs, por exemplo. A utilização de informação acústica acaba sendo uma opção melhor no ambiente subaquático. Sensores baseados em som possuem uma performance melhor embaixo da água e acabam sendo bastante empregados para realizar o sensoriamento de UUVs [Machado et al., 2016]. Apesar de ser melhor, os sensores baseados em som possuem várias limitações como pequena largura de banda, baixa taxa de dados e alta latência.

Apesar de todo esse contexto adverso, a pesquisa envolvendo navegação e localização de UUVs aumentou bastante na última década, adentrando vários campos de pesquisa [Paull et al., 2013], [Machado et al., 2016]. Alguns estudos foram inicialmente realizados com relação à tecnologia aplicadas a sensores de inércia, onde caríssimas IMUs (do inglês *Inertial Measurement Unit*) foram desenvolvidas e aplicadas [Paull et al., 2013]. Apesar de útil, IMUs possuem performance consideravelmente limitada (mesmo as mais sofisticadas), apresentando baixo custo benefício para o ambiente desse tipo de veículo. A aplicação de Localização Simultânea e Mapeamento ou SLAM (do inglês *Simultaneous Localization And Mapping*) juntamente com uma combinação de sensores acústicos e geofísicos apresentaram novas possibilidades para que UUVs possam realizar localização e navegação de forma autônoma [Paull et al., 2013], [Machado et al., 2016].

O problema da navegação sem mapa para UUVs também têm sido abordado utilizando Deep-RL. Carlucho et al. [2018] por exemplo, usou uma abordagem baseada na técnica DDPG para realizar navegação 2D e 3D orientada a alvo de um UUV - Nessie VII [Cartwright et al., 2008] - simulado. Carlucho et al. [2018] apresenta um trabalho semelhante ao proposto nesse estudo, mais com algumas diferenças importantes. O veículo utilizado possui uma estrutura do tipo submarino, enquanto que o veículo utilizado neste trabalho possui uma estrutura baseada em quadrotor. Além disso, Carlucho et al. [2018] utiliza a posição atual do veículo como uma das informações de entrada para a rede do agente, enquanto na proposta desse trabalho essa informação não é utilizada. Carlucho et al. [2018] também não faz o uso da informação de algum sensor para detectar obstáculos, como um sonar por exemplo. Nesse contexto, onde há somente o uso do *ground truth*, o agente tende a ficar viciado no cenário no qual foi treinado, não sendo capaz de desviar um obstáculo para chegar ao alvo, muito menos ser capaz de atuar em um cenário dinâmico.

Um dos objetivos deste trabalho é fazer com que o veículo seja capaz de aprender a desviar de objetos, caso exista algum no caminho para chegar ao alvo pretendido. Yu et al. [2017] também propõe o uso de uma abordagem com DDPG, mas aplicado a tarefa de seguidor de caminho (do inglês *Path Following*) e com enfoque na comparação com um controle PID (do inglês *Proportional Integral Derivative*). A tarefa é realizada em um ambiente 2D com um UUV do tipo submarino.

Wu et al. [2019] aborda navegação sem mapa com Deep-RL em duas tarefas: prevenção de obstáculos (do inglês *Obstacle Avoidance*) e seguidor de cano (do inglês *Pipeline Following*). A primeira é realizada em um ambiente 2D enquanto que a segunda em um contexto 3D. O trabalho foi feito utilizando o veículo *Girona500* que possui 6 graus de liberdade ou DoFs (do inglês *Degrees of Freedom*) simulado no Gazebo. Para a tarefa de prevenção de obstáculos foi utilizado informação de um conjunto de sensores ultrassônicos de forma semelhante a proposta desse trabalho, enquanto que para a segunda tarefa foi utilizado informação visual de uma câmera. Assim como o presente trabalho, Wu et al. [2019] não tem a necessidade de conhecer a posição atual (*ground truth*) do veículo. O estudo acerca de evitar colisões que Wu et al. [2019] discute também é analisado neste trabalho, com a diferença de ser aplicado a um veículo baseado em quadrotor e levar em consideração o estudo em um contexto 3D também.

Zhang et al. [2020] propôs o uso de Deep-RL interativo para realizar a tarefa de seguidor de caminho em 2D para um UUV tipo submarino simulado no Gazebo. A abordagem consiste em usar reforço do ambiente e o reforço de um observador humano em uma forma híbrida. Zhang et al. [2020] salienta que essa abordagem permite um aprendizado mais rápido do que quando somente reforço do ambiente é utilizado. Assim como Carlucho et al. [2018], Zhang et al. [2020] também não utiliza informação de algum tipo de sensor, somente informações com relação à posição do veículo. Essa proposta interativa de Zhang et al. [2020], apesar de custosa do ponto de vista da demanda de ter que ter um ser humano para dar uma recompensa em tempo real, pode ser uma opção de trabalho futuro para a proposta do trabalho presente.

Conclusão, este trabalho diferencia-se dos demais trabalhos discutidos por buscar abordar o problema da navegação sem mapa para HUAUV, mesmo que sendo em simulação. Além disso, a capacidade de evitar colisão com obstáculos também é analisada levando em conta somente informação de sensores de distância descritos a partir de sensores de distância simulados. Diferentemente da maioria dos trabalhos discutidos, não há o uso de informação visual e/ou da posição cartesiana explícita do veículo, buscando obter uma estrutura de agente simples e com maior capacidade de generalização. Duas abordagens baseadas em técnicas de Deep-RL do estado da arte (DDPG e SAC) são abordadas e analisadas, diferentemente da maioria dos trabalhos que abordam navegação sem mapa utilizando somente um tipo de abordagem. Além disso, também é apresentado um estudo de validação para analisar com mais robustez as propostas apresentadas e com-

parar com propostas relacionadas, estudo não visto em outros trabalhos. Na sequência, o Capítulo 3 apresenta a descrição do veículo híbrido utilizado em simulação e o *setup* experimental do trabalho.

3 SETUP EXPERIMENTAL

Nesta Seção, é apresentado o veículo desenvolvido para a metodologia do trabalho bem como as ferramentas utilizadas para desenvolvê-lo. Também são discutidas as ferramentas utilizadas para o desenvolvimento dos agentes de Deep-RL.

3.1 Interface de Simulação

Escrever um *software* para robôs pode ser uma tarefa árdua, principalmente considerando que o escopo que envolve robótica é grande e continua crescendo. O reuso de código entre robôs não é uma tarefa trivial devido à grande variação de *hardware* existente entre eles. Além disso, a semelhança entre os códigos geralmente é muito pequena. Um mesmo *software* pode ser responsável por somente uma aplicação ou por uma série de aplicações de variados escopos, da camada de dispositivos até controle e inteligência. Ou seja, as arquiteturas de *softwares* para robôs precisam dar suporte a integração de uma vasta escala heterogênea de aplicações.

Para isso, muitos *frameworks* para o desenvolvimento de *software* foram desenvolvidos na área da robótica. Apesar de buscarem gerenciar a complexidade e facilitar a prototipagem, esses *frameworks*, de modo geral, foram desenvolvidos para propósitos específicos, talvez devido a ineficiência de outros *frameworks*, ou para enfatizar aspectos mais relevantes no processo de desenvolvimento que um determinado robô pode demandar. Partindo dessa situação que o ROS [Quigley et al., 2009] foi projetado. ROS é um *framework* que busca uma integração em larga escala e tem como algumas de suas filosofias ser modular, aberto, gratuito e multilíngue.

ROS possui suporte a muitas linguagens de programação, sendo C++ e Python as mais comumente utilizadas. Possui um micronúcleo com muitas ferramentas para compilar e rodar componentes e reusa código de muitos projetos abertos, como os de visão computacional de OpenCV [Bradski and Kaehler, 2000], planejamento de OpenRAVE [Diankov and Kuffner, 2008], navegação de Player and Stage Project [Gerkey et al., 2003], [Gerkey et al., 2001], [Vaughan et al., 2003].

Devido a todas essas características, foi utilizado a interface que ROS possui para de-

envolver várias aplicações para o presente trabalho. Mais especificamente, foi utilizado a versão Melodic do ROS juntamente com a versão 2.7 do Python, em aplicações no escopo de Rospy. A escolha da linguagem Python se deve a facilidade de uso e o suporte de várias dependências necessárias para o desenvolvimento de um trabalho que envolva AI, tal como PyTorch que facilita a criação de agentes de Deep-RL. De forma resumida, todas as aplicações desenvolvidas foram encapsuladas em um único pacote ROS ¹. Nas subsecções seguintes são descritas algumas das principais dependências do pacote ROS desenvolvido e como elas foram aplicadas no trabalho, além da descrição do pacote ROS em si.

3.1.1 Gazebo

A robótica é um área de estudo complexa e custosa que demanda muitos recursos que geralmente são difíceis de obter e manter. Partindo desse contexto, a simulação de robôs é fundamental para que muitos estudos científicos não pereçam somente na ideia e que muitas aplicações de robótica sejam efetivamente desenvolvidas. Um ambiente de simulação torna possível o desenvolvimento de robôs, o teste de aplicações e até mesmo o treinamento de complexos sistemas de inteligência em ambientes realísticos. Gazebo [Koenig and Howard, 2004] torna isso possível devido a uma grande e ativa comunidade de desenvolvedores, além de ser compatível com ROS.

Para criar esse ambiente realístico, todos os objetos simulados no Gazebo possuem massa, velocidade, fricção, e inúmeras outras propriedades físicas. Ele permite a junção de múltiplos objetos rígidos conectados via juntas, modelando uma estrutura dinâmica, onde forças lineares e angulares podem ser aplicadas para gerar locomoção e interação com o ambiente. A soma de todas as estruturas dinâmicas cria o mundo no simulador. A maioria das propriedades físicas são controláveis, das condições de luz à inércia de objetos.

3.1.1.1 Arquitetura e Ambiente

Como pode ser observado na Figura 1, o mundo no Gazebo representa o conjunto de todos os modelos e propriedades do ambiente tais como gravidade e luminosidade, onde cada modelo é composto por, ao menos, um corpo e um número de juntas e sensores. Para que o modelo e essas propriedades existam, um conjunto de bibliotecas independentes realizam uma interface para o funcionamento do simulador no nível mais baixo, prevenindo a dependência de ferramentas específicas que possam sofrer alterações e permitindo a possibilidade do emprego de melhores bibliotecas caso surjam. Por último, comandos do cliente são recebidos através de uma interface com memória compartilhada.

Gazebo utiliza, por exemplo, a biblioteca ODE (do inglês *Open Dynamics Engine* [Smith, 2001]) como uma interface para simular a dinâmica e a cinemática associada

¹https://github.com/ricardoGrando/hydrone_aerial_underwater_gazebo

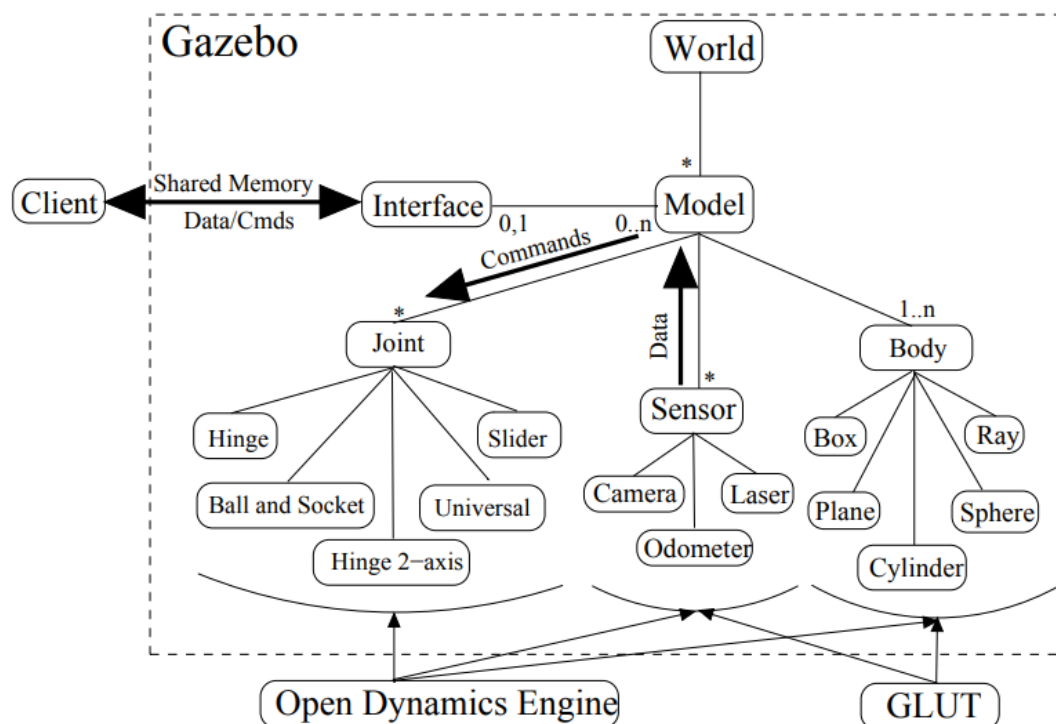


Figura 1: Arquitetura e Organização do Gazebo mostrando a estrutura de um modelo e como o mesmo interage com o mundo e com demais aplicações ROS.

Fonte: Koenig and Howard [2004].

a corpos rígidos articulados. Esta *engine* possui várias características tais como detecção de colisão, funções de massa e rotação, etc; é uma das mais utilizadas para a simulação física dentre as de código aberto.

A interface para o usuário é outro exemplo. Essa interface precisa ser sofisticada e rápida ao mesmo tempo que não pode prejudicar o processamento da capacidade de simular a dinâmica do simulador, que é a essência do Gazebo e demanda uma carga de processamento significativa. Buscando amenizar ao máximo essa condição, o simulador utiliza *OpenGL Utility Toolkit* [Shreiner et al., 2009]. OpenGL é uma biblioteca padrão para a criação de aplicações interativas em duas e três dimensões, sendo estável, escalável e permitindo a execução em GPUs. O conjunto de ferramentas de utilidades (*OpenGL Utility Toolkit*) fornece mecanismos para interação do usuário com o Gazebo através de dispositivos de entrada, tais como teclado.

Um modelo é um objeto que possui uma representação física composta por pelo menos um corpo rígido, sensores, juntas e interfaces para facilitar o fluxo de dados. Um corpo é a estrutura física mais básica, como um cubo, cilindro, esfera, etc; que possui propriedades físicas que são conectadas por juntas, que dessa forma, permitem a formação de relações cinemáticas e dinâmicas.

Para executar tarefas úteis um robô precisa de sensores. No Gazebo, um sensor é

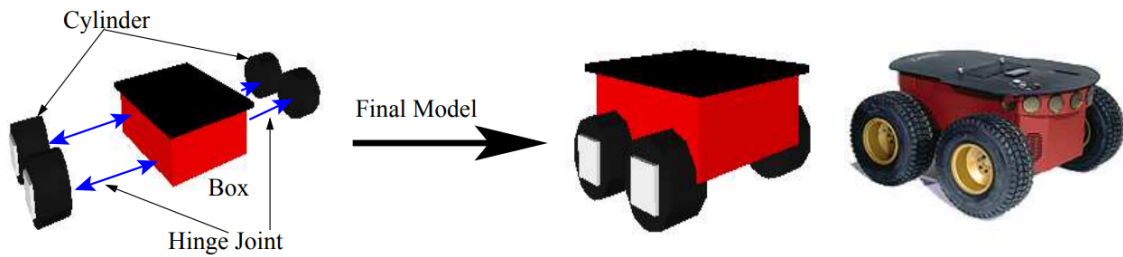


Figura 2: Comparação do robô Pioneer2AT descrito no Gazebo e real.

Fonte: Koenig and Howard [2004].

um dispositivo abstrato, sem representação física, que ganha um encorpamento somente quando descrito em um modelo. Ou seja, um sensor pode ser usado novamente por qualquer modelo. Por exemplo, um robô pode ter inúmeras câmeras desde que essas câmeras estejam descritas em modelos que compõem o modelo final deste robô.

A implementação de um modelo no Gazebo é feita utilizando a linguagem XML. Um modelo é implementado utilizando dois arquivos, sendo um de configuração (*modelo.config*) e outro com a descrição (*modelo.sdf*). O arquivo de configuração possui informações sobre o modelo tais como nome (`<name>`), versão (`<version>`) e autor (`<author>`) enquanto que no arquivo de descrição estão presentes características do modelo tais como os corpos (`<link>`) que compõem esse modelo com suas propriedades físicas: visual (`<visual>`), (`<collision>`), massa (`<mass>`), etc. Dois corpos (`<links>`) são conectados utilizando uma junta (`<joint>`) onde é definido um hierarquia entre esse dois corpos, pai (`<parent>`) e filho (`<child>`). Para robôs que possuem uma descrição muito grande, como é o caso do veículo deste trabalho, se utiliza XML Macro (Xacro) [MacroXML, 2018] para organizar a descrição em arquivos XML menores e mais legíveis.

Um exemplo de um robô descrito no Gazebo pode ser visualizado na Figura 2 onde o robô Pioneer2AT é criado através da junção de quatro cilindros e um chassi com quatro juntas de revolução.

Resumidamente, um modelo de robô no Gazebo é simplesmente uma junção entre vários corpos. Entretanto, robô com um elevado número de juntas e segmentos pode possuir muitos nós e tópicos. Além disso, o mundo onde esse robô será instanciado, do inglês *spawn*, pode necessitar possuir outras instâncias do mesmo modelo ou de modelos afins. Para efetivamente colocar o mundo de um robô no ambiente do Gazebo é utilizado o comando *roslaunch*. Com *roslaunch*, um conjunto de nós e modelos são instanciados ao menos uma vez, iniciando definitivamente o ambiente.

Para este trabalho foi descrito no Gazebo um modelo de robô do tipo HUAUV baseado em quadrotor, bem como sensores e ambientes de simulação para que o veículo possa atuar e treinar os agentes de Deep-RL. As informações sobre a descrição do veículo estão

na Seção 3.2.

3.1.2 RotorS

Para testar algoritmos em UAVs geralmente é necessário ter acesso a uma boa infraestrutura com testes de campo. A maioria dos erros que ocorrem em UAVs reais são difíceis de reproduzir e geralmente acabam por danificar o veículo. Partindo desse contexto que RotorS [Furrer et al., 2016] foi desenvolvido. RotorS é um *framework* de simulação que busca reduzir o elevado tempo que geralmente é gasto nos testes de campo e diminuir acidentes com UAVs reais. No contexto que envolve o desenvolvimento da robótica de modo geral onde o custo financeiro costuma ser um problema, ter acesso a uma ferramenta que permita de certa forma simular a infraestrutura real necessária é fundamental para que exista progresso nessa área.

RotorS fornece a descrição de uma variedade de UAVs com uma estrutura modular. A descrição consiste basicamente de um corpo, um número fixo de rotores e alguns sensores que podem ser acoplados no corpo. Todos os componentes que são encontrados em UAVs reais são replicados usando *plugins* no Gazebo, como por exemplo a dinâmica dos motores e a aerodinâmica do veículo em si. Além disso, vários sensores, como IMUs por exemplo, também podem ser acoplados na descrição do veículo. Esses sensores conseguem simular de forma fiel o ambiente real através do uso de modelos de ruído. Tanto os sensores quanto a descrição do veículo fornecem *ground truth*, abrindo espaço para facilitar o depuramento e avaliação de novos algoritmos.

Para facilitar o desenvolvimento de diferentes estratégias de controle RotorS fornece uma interface de controle simplificada baseado em um controle geométrico [Lee et al., 2010]. Essa interface de controle pode ser adaptada para qualquer veículo e consiste de um arquivo de parâmetros com informações como atitude, taxa angular e controle de posição por exemplo. A implementação deste controlador permite utilizá-lo para fazer o veículo assumir uma posição cartesiana no espaço (ativa) e para definir uma velocidade cartesiana no mesmo (passiva). A primeira opção permite que uma posição alvo ou uma sequência de posições alvo possam ser definidas, permitindo que o veículo possa navegar sem mapa. A segunda opção permite definir uma velocidade cartesiana para que o veículo possa se movimentar. O controlador foi utilizado de forma passiva pelos agentes de Deep-RL para realizar a navegação dos mesmos, podendo ser usado de forma ativa para realizar a navegação sem mapa do próprio controlador. A interface para usar cada um dos modos foi feito via ROS, com o tópico de *cmd_vel* com uma mensagem tipo *Twist* para o modo passivo e com o tópico *command/pose* com uma mensagem do tipo *PoseStamped* para o modo ativo.

Para o estudo deste trabalho foi desenvolvido um veículo com estrutura quadrotor utilizando o *framework* RotorS. Os parâmetros físicos do veículo, tais como massa e inércia, foram obtidos através da modelagem do veículo utilizando o *software* Solidworks. Para

o controle do veículo foi utilizado a interface simplificada de Lee et al. [2010] que o *framework* fornece com parâmetros que o permite funcionar não somente no ambiente aéreo por definição, mas também no ambiente subaquático que foi simulado utilizando o *framework* UUV simulator. Mais informações sobre a descrição aérea do veículo podem ser analisadas na Seção 3.2.

3.1.3 UUV Sim

Assim como para UAVs, o desenvolvimento de aplicações e algoritmos para UUVs demandam uma boa infraestrutura e testes de campo custosos do ponto de vista financeiro e de tempo. Nesse contexto, um ambiente de simulação também é fundamental para que o estudo de UUVs não seja limitado somente aos centros de pesquisa que detém boas condições financeiras. UUV simulator [Manhães et al., 2016] foi desenvolvido para simulação subaquática com o mesmo intuito que o RotorS foi desenvolvido para simulação de veículos aéreos.

UUV simulator é uma extensão do Gazebo baseada em *plugins* que modelam o efeito da hidrostática e da hidrodinâmica, bem como sensores, *thrusters*, e perturbações externas. Assim como o RotorS, ele permite descrever características do veículo para o ambiente, tais como empuxo, massa adicional, volume, etc; além de características do ambiente subaquático em si, como velocidade da corrente de água por exemplo.

Diferentemente do RotorS, o UUV sim possui um número mais reduzido de veículos descritos. Por definição, o *framework* possui somente o veículo RexROV2 [Berg, 2012] descrito como exemplo, com um plugin de controle PID específico para o veículo. A maioria dos UUVs descritos no simulador e usados em pesquisas científicas não possuem código fonte disponível, sendo que grande parte se baseiam em uma estrutura baseada em submarino.

Para o estudo do presente trabalho o mesmo veículo usado na simulação aérea foi usado na simulação subaquática. O veículo descrito no ambiente subaquático é o primeiro baseado em uma estrutura quadrotor já desenvolvido para o simulador Gazebo. Alguns parâmetros subaquáticos utilizados em simulação também foram obtidos utilizando a modelagem do veículo feita no *software* Solidworks, enquanto que outros foram estimados de forma empírica. Foi feito de tal forma pois, assim como discutido em Berg [2012], o processo para obter todos os parâmetros hidrodinâmicos não é trivial e demanda expertise avançada em *softwares* específicos. Mais informações sobre a descrição subsubaquática do veículo também estão na Seção 3.2.

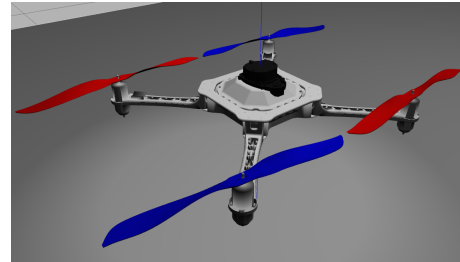
3.2 Desenvolvimento do Hydrone simulado

Como discutido previamente na Seção 3.1.1, um veículo simulado no Gazebo consiste de uma conjuntura de XML Macros, onde ocorre a descrição do veículo, e *launches*, que

permitem que a descrição seja instanciada juntamente com aplicações para criar a inteligência do veículo ou coletar dados por exemplo. Nesta Seção são discutidos informações referentes a descrição do veículo, bem como as aplicações desenvolvidas. O veículo descrito no Gazebo foi baseado em um veículo real chamado Hydrone [Horn et al., 2019]. Na Figura 3 pode-se observar o veículo real e o veículo final descrito em simulação.



(a) Veículo real.



(b) Veículo descrito.

Figura 3: Veículo simulado baseado no veículo real Hydrone.

3.2.1 Descrição Quadrotor Base e Parâmetros Aéreos

Como modelo base para a descrição do veículo foi utilizado o modelo genérico de multirotor base² que o *framework* RotorS disponibiliza. Para o *frame* base do veículo foi utilizada obteve-se do *software* Solidworks o valor de 1.93 kg para massa e a seguinte matriz de inércia:

$$I_b = \begin{bmatrix} 0,0347563 & 0,0 & 0,0 \\ 0,0 & 0,0458929 & 0,0 \\ 0,0 & 0,0 & 0,0977 \end{bmatrix}.$$

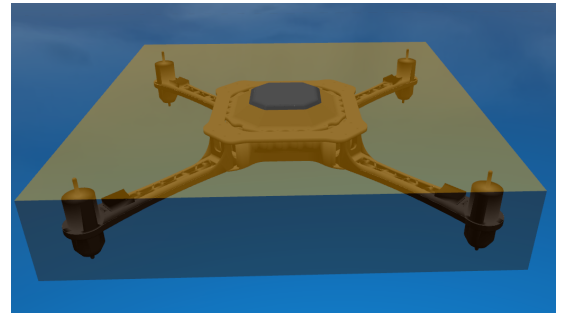
Para a *visual* do *frame* foi utilizado uma *mesh* exportada do Solidworks. Para a *collision* do *frame* foi estimado uma caixa com dimensões de 0,47 metros de lado e 0,09 metros de altura. Na Figura 4 é possível observar o visual do *frame* base com a *mesh*, *collision* e a *inertia*.

Os quatro rotores foram colocados de forma simétrica com um ângulo de 90° entre cada e a uma distância exata de 0,27 metros em relação ao centro do veículo. O sentido de rotação dos rotores é alternado a cada braço do *frame* de tal forma que na Figura 5 as hélices vermelhas giram no sentido horário enquanto que as hélices azuis giram no sentido anti-horário. As informações referentes aos rotores utilizados estão na Tabela 1, enquanto que o *frame* do veículo com os rotores acoplado pode ser visualizado na Figura 5.

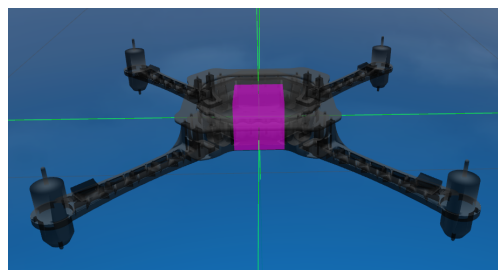
²https://github.com/ricardoGrando/hydrone_aerial_underwater_gazebo/blob/master/hydrone_aerial_underwater_description/urdf/multirotor_base.xacro



(a) Visual do frame.



(b) Collision do frame.



(c) Inertia do frame.

Figura 4: *Frame* base do veículo.

Tabela 1: Parâmetros dos Rotores.

Constante	Valor
Massa	0,005 Kg
Coefficiente de Arrasto	$8,06428e - 05$
Coefficiente de Momento	0,000001
Constante Motor	$8,54858e - 06 \text{ Kg/s}^2$
Constante de tempo superior	0,0125 s
Constante de tempo inferior	0,025 s
Rotação máxima	1838 rad/s

Para o plugin dos rotores do RotorS o rotor e a hélice são considerados o mesmo objeto. Ou seja, a massa utilizada equivale à soma das massas do rotor e da hélice. A matriz de inércia utilizada para cada rotor foi aproximada para a inércia de um cubóide da seguinte forma:

$$I_r = \begin{pmatrix} 0,0833333 * m * (y*y + z*z) & 0,0 & 0,0 \\ 0,0 & 0,0833333 * m * (x*x + z*z) & 0,0 \\ 0,0 & 0,0 & 0,0833333 * m * (x*x + y*y) \end{pmatrix}.$$

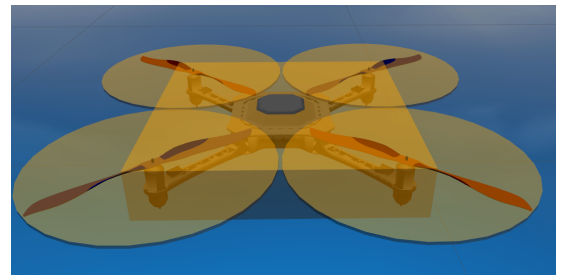
(a) *Visual* do veículo.(b) *Collision* do veículo.(c) *Joints* do veículo.(d) *Inertia* do veículo.

Figura 5: Veículo Hydrono com rotores.

Sendo a massa m de 0,005 Kg, o raio do rotor em x de 0,195 m, a altura z de 0,003 m e a largura y de 0,015 m. A massa final do veículo somado aos sensores acoplados ficou definida em 2,07675 Kg.

3.2.2 Parâmetros Subaquáticos

Os parâmetros utilizados para realizar a simulação do modelo hidrodinâmico do veículo foram obtidos utilizando o *software* Solidworks. Os parâmetros obtidos via Solidworks, entretanto, não possuem uma representação precisa. Para obter tais parâmetros, como a massa adicional e o amortecimento linear, é necessário utilizar *softwares* específicos de obtenção do modelo hidrodinâmico, tais como o Wamit [Eng et al., 2014] por exemplo. Entretanto o processo para a obtenção é complexo, como discutido em Berg [2012]. Dessa forma, para a versão do veículo até o momento foi utilizado os valores aproximados fornecidos pelo Solidworks. A descrição do modelo hidrodinâmico pode ser observada no pacote desenvolvido³.

O volume do veículo foi aproximado para 1 dm^3 , o empuxo foi aproximado para o centro real do veículo $(0, 0; 0, 0; 0, 0)$ e a caixa que representa o veículo foi definida com 0.1 m de altura e 0,47 m de largura. As matrizes referentes a massa adicional,

³https://github.com/ricardoGrando/hydrone.aerial_underwater_gazebo/blob/master/hydrone.aerial_underwater_description/urdf/hydrone.aerial_underwater_gazebo.xacro

amortecimento linear e amortecimento quadrático aproximadas pelo Solidworks podem ser observadas nas equações abaixo:

$$AddedMass = \begin{pmatrix} 1,93 & 0,0 & 0,0 & 0,0 & 0,0 & 0,0 \\ 0,0 & 1,93 & 0,0 & 0,0 & 0,0 & 0,0 \\ 0,0 & 0,0 & 1,93 & 0,0 & 0,0 & 0,0 \\ 0,0 & 0,0 & 0,0 & 0,0156 & 0,0014 & 0,0004 \\ 0,0 & 0,0 & 0,0 & 0,0014 & 0,03 & 0,0011 \\ 0,0 & 0,0 & 0,0 & 0,0004 & 0,0011 & 0,0166 \end{pmatrix},$$

$$Damping_{linear} = \begin{pmatrix} 0,0125 & 0,0 & 0,0 & 0,0 & 0,0 & 0,0 \\ 0,0 & 0,0125 & 0,0 & 0,0 & 0,0 & 0,0 \\ 0,0 & 0,0 & 0,0499 & 0,0 & 0,0 & 0,0 \\ 0,0 & 0,0 & 0,0 & 0,0125 & 0,0 & 0,0 \\ 0,0 & 0,0 & 0,0 & 0,0 & 0,0125 & 0,0 \\ 0,0 & 0,0 & 0,0 & 0,0 & 0,0 & 0,0125 \end{pmatrix},$$

$$Damping_{quadratic} = \begin{pmatrix} 0,0125 & 0,0 & 0,0 & 0,0 & 0,0 & 0,0 \\ 0,0 & 0,0125 & 0,0 & 0,0 & 0,0 & 0,0 \\ 0,0 & 0,0 & 0,0499 & 0,0 & 0,0 & 0,0 \\ 0,0 & 0,0 & 0,0 & 0,0125 & 0,0 & 0,0 \\ 0,0 & 0,0 & 0,0 & 0,0 & 0,0125 & 0,0 \\ 0,0 & 0,0 & 0,0 & 0,0 & 0,0 & 0,0125 \end{pmatrix}.$$

3.2.3 Parâmetros de Controle

O controle de posição utilizado [Lee et al., 2010] foi implementado de forma genérica no RotorS, sendo possível instanciar o controle de posição usando arquivos de configuração no *launcher* do veículo. O arquivo de configuração para o controle de posição⁴ podem ser observados no pacote ROS desenvolvido. Os valores usados para o controle de posição podem ser observados nas equações abaixo.

$$PositionGain = \begin{pmatrix} 4,0 & 4,0 & 6,0 \end{pmatrix},$$

$$VelocityGain = \begin{pmatrix} 4,7 & 4,7 & 4,7 \end{pmatrix},$$

$$AtitudeGain = \begin{pmatrix} 2,0 & 2,3 & 0,15 \end{pmatrix},$$

⁴https://github.com/ricardoGrando/hydrone_aerial_underwater_gazebo/blob/master/hydrone_aerial_underwater_gazebo/resource/lee_controller_hydrone.yaml

$$AngularRateGain = \begin{pmatrix} 0,4 & 0,52 & 0,18 \end{pmatrix}$$

3.2.4 Sensor de distância - Lidar

Um sensor de distância baseado no plugin *ray* do Gazebo foi descrito⁵. Esse sensor no mundo real pode ser representado por um Lidar no meio aéreo [Furrer et al., 2016], enquanto que um sonar ou um conjunto de sonares pode ser utilizado como representação no meio subaquático como realizado na descrição do veículo subaquático RexRov2 no Gazebo [Berg, 2012].

Na descrição feita para esse trabalho, o sensor possui resolução angular de $0,25^\circ$, mostrando 1080 leituras de distância em um alcance de 270° . Entretanto, somente 20 amostras de distância igualmente espaçadas em $13,5^\circ$ foram utilizadas como entradas para a rede dos agentes de Deep-RL. Apesar de gerar uma perda de resolução na detecção do ambiente, essa simplificação permite que o agente tenha menos estados e, portanto, consiga convergir na detecção de objetos mais rapidamente, além de gerar um menor tamanho de rede que implica em um menor demanda de capacidade de processamento e tempo médio de treino por episódio.

Como representação do *visual* foi utilizado uma *mesh*⁶ genérica para sensores de distância disponível no simulador Gazebo, enquanto que a *collision* do sensor foi abstraída para um cilindro de 0,0315 metros de comprimento e 0,055 metros de raio. A matriz de inércia for aproximada da seguinte forma:

$$InertiaLidar = \begin{pmatrix} 0,001 & 0,0 & 0,0 \\ 0,0 & 0,001 & 0,0 \\ 0,0 & 0,0 & 0,001 \end{pmatrix}.$$

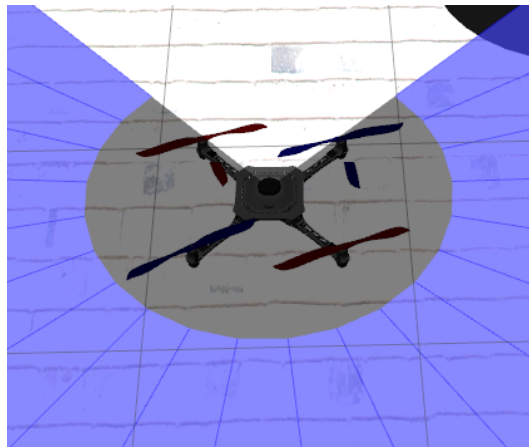


Figura 6: Sensor de distância acoplado ao veículo.

⁵https://github.com/ricardoGrando/hydrone_aerial_underwater_gazebo/blob/master/hydrone_aerial_underwater_description/urdf/component_snippets.xacro

⁶https://github.com/ricardoGrando/hydrone_aerial_underwater_gazebo/blob/master/hydrone_aerial_underwater_description/meshes/lds.stl

Tabela 2: Parâmetros do Lidar utilizado.

Constante	Valor
Massa	0,130 Kg
Angulo de escaneamento	270°
Alcance detecção mínimo	0,5 metros
Alcance detecção máximo	10,0 metros
Frequência de Atualização	5 Hz
Resolução	0,015 metros
Ruído	Gaussiano

Na Figura 6 pode-se observar o veículo com o sensor de distância acoplado na parte superior. Os raios em azul da Figura 6 representa o sinal emitido pelo sensor.

3.2.5 Sensor de distância - Sonar

Cerqueira et al. [2016] apresentou uma abordagem de sonar de imagem em simulação com baixo custo computacional. O simulador foi desenvolvido utilizando a linguagem C++ e OpenGL, com integração para rodar em simulação com o simulador Gazebo.

O simulador se baseia no uso de um OSG (do inglês *Open Scene Graph*) e uma câmera em uma cena renderizada. Três dados são obtidos a partir dessa simulação: intensidade, profundidade e distorção angular. A intensidade representa a energia acústica refletida baseada no ângulo da superfície de um objeto, a profundidade representa a distância euclidiana entre o ponto central da câmera e um ponto na superfície do objeto e por último a distorção angular representa o ângulo entre a coluna central e as colunas nas extremidades da câmera. A imagem do sonar é a junção das três imagens formadas por esses canais de dados. Na Figura 7 a imagem gerada (*shader*) e a imagem acústica corresponde são mostradas.

Todos os pixels de uma coluna possuem o mesmo ângulo e cada coluna possui um *beam*. A intensidade de cada *bin* é calculada de acordo com os canais azul e verde da imagem, azul representando a intensidade e verde a profundidade. A distância máxima é o *range* configurado para o sonar. Na Figura 8, o processo de simulação do sonar é mostrado.

Alguns parâmetros precisam ser definidos para realizar a simulação do sonar, tais como o *range*, número de *bins* e *beams* a largura e altura de *beam*. Para o presente trabalho foi utilizado um sonar do tipo *Forward Looking Sonar* (FLS) com os parâmetros da Tabela 3:

Os dois cenários de simulação desenvolvidos no Gazebo foram criados em OSG e instanciados no simulador do sonar. Um pacote adaptado do trabalho de Longaray et al. 2017 foi desenvolvido ⁷ para converter a imagem acústica do sonar em valores de distância em

⁷https://github.com/ricardoGrando/gpu_sonar_simulation

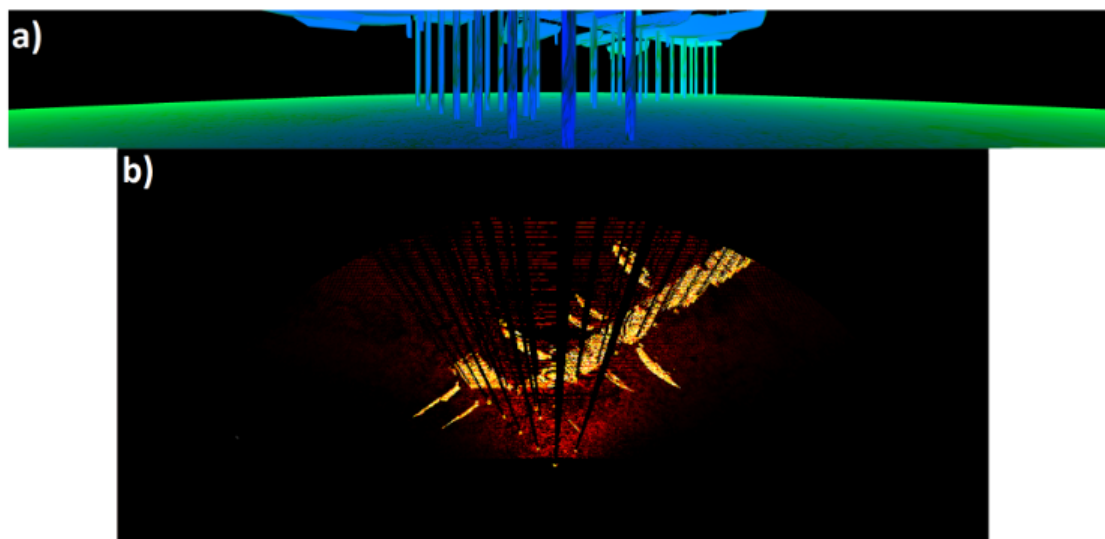


Figura 7: a) Imagem gerada pelo *shader*, b) imagem acústica resultante. Apresentado em Longaray et al. 2017.

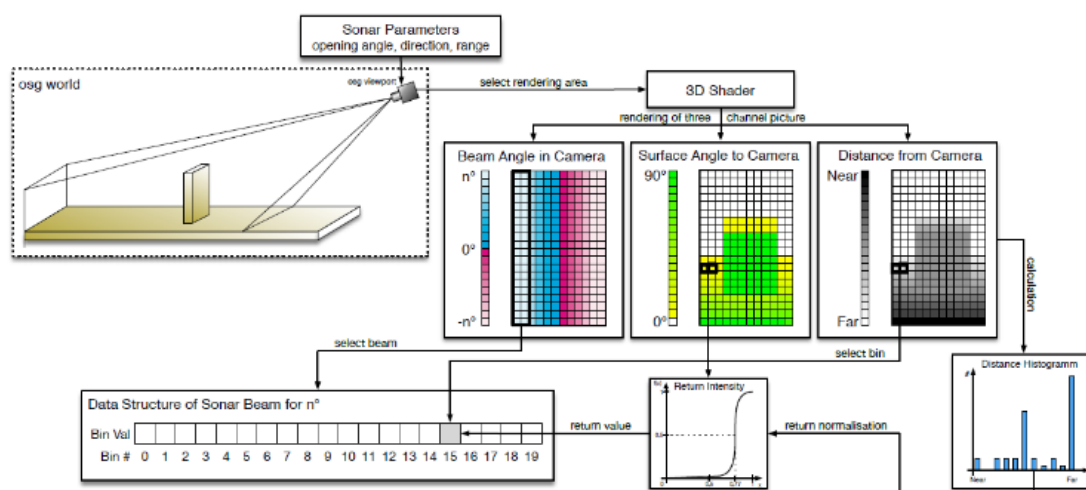


Figura 8: Representação gráfica referente as etapas realizadas desde a extração de informações de uma cena em OSG até a atribuição de intensidades de cada *bin* em um *beam* de sonar. Apresentado em Cerqueira et al. [2016].

Tabela 3: Parâmetros de simulação do sonar FLS.

Constante	Valor
Alcance	20 m
N <i>beams</i>	256
N <i>bins</i>	1000
Altura de <i>beam</i>	15
Largura de <i>beam</i>	90

metros para serem utilizados nos agentes. 20 amostras igualmente espaçados entre os 256

beams foram usadas como entrada para o estado dos agentes. É importante ressaltar que, como está descrito em mais detalhes na Seção 5 de Resultados, para os agentes no ambiente subaquático foi utilizado *transfer learning* a partir dos agentes treinados no ambiente aéreo. Além disso, o simulador de sonar foi utilizado somente na navegação subaquática no contexto 2D. No contexto híbrido foi utilizado o sensor de distância descrito pelo *plugin* de *ray* do Gazebo. Entretanto, como mostrado nos resultados, a utilização eficaz de *transfer learning* mostrou que não há grandes diferenças entre computar a distância através *plugin ray* ou através do simulador de sonar de Cerqueira et al. [2016].

3.3 Desenvolvimento dos Ambientes de simulação

Após realizar a descrição do veículo, foram desenvolvidos ambientes de simulação no Gazebo a fim de treinar e testar as abordagens de Deep-RL propostas. A estrutura básica do mundo utilizada é composta por um oceano na parte negativa do eixo Z , onde ocorre o efeito de ambos os frameworks. Na parte positiva do eixo Z somente o framework de simulação aérea RotorS atua, simulando o ambiente aéreo. O oceano possui corrente de água e no ambiente aéreo foi utilizado a influência de ventos. Na Figura 9 é possível observar essa estrutura básica, enquanto que os parâmetros da do ruído utilizado para a simulação da dinâmica no ambiente podem ser observados na Tabela 4.

Tabela 4: Parâmetros de ruído para a dinâmica do ambiente de simulação.

Constante	Valor
σ	0,15
θ	0,15

A partir dessa estrutura básica, dois cenários diferentes foram descritos utilizando modelos disponíveis no Gazebo. No primeiro cenário, um espaço com dimensões de $5 \times 5 \times 6$ metros foi descrito fazendo-se do uso do modelo *grey_wall*. Os modelos foram instanciados de forma estática de modo a criar um cenário que varia de -1 metros a 5 metros no eixo de Z , centralizado na origem para os eixos X e Y . Dessa forma o cenário representa um tanque com 1 metro de coluna de água e 5 metros de coluna de ar.

Um segundo cenário, baseado no primeiro, também foi descrito. Nesse segundo cenário, quatro cilindros com 0,6 metros de raio e 3 metros de comprimento foram adicionados à estrutura do primeiro cenário. Os quatro objetos também foram instanciados de forma estática no cenário e foram colocados em posições no cenário que os tornam possíveis obstáculos, representando dutos de perfuração de petróleo. Nesse cenário busca-se que os agentes de Deep-RL aprendam a desviar e a desviar os objetos a fim de chegar na posição de destino levando em conta um cenário mais realístico onde um veículo híbrido poderia atuar. Na Figura 10 é possível observar ambos os cenários.

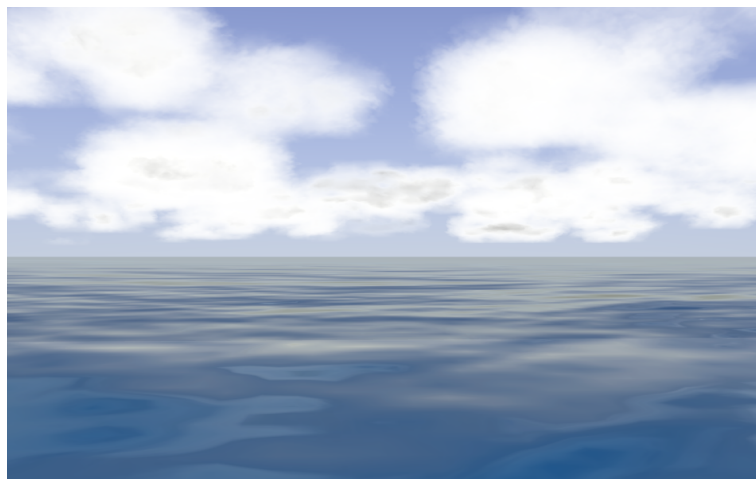
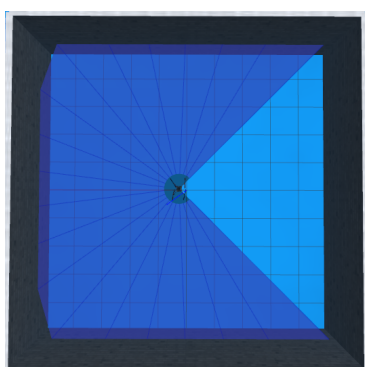
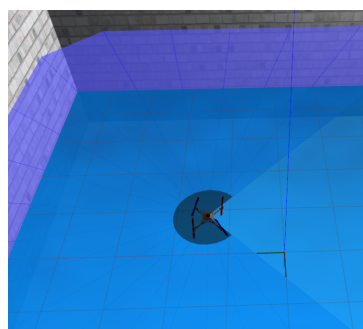


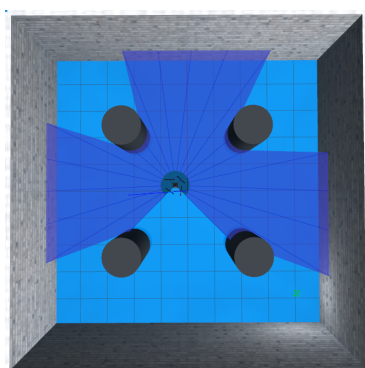
Figura 9: Estrutura básica do ambiente descrito.



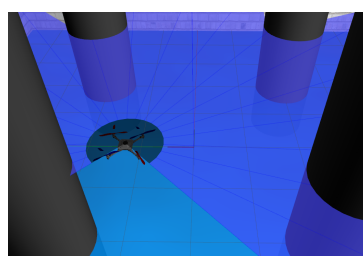
(a) Cenário 1 visto de cima.



(b) Cenário 1.



(c) Cenário 2 visto de cima.



(d) Cenário 2.

Figura 10: Cenários de simulação.

Ambos os cenários também foram descritos no *software* Blender para serem utilizados na simulação do sonar.

3.4 Algoritmo BUG2

O algoritmo foi implementado baseado na versão do algoritmo BUG2 de Marino et al. [2016]. Esse algoritmo foi idealizado para realizar a navegação de UAVs em três di-

mensões. Na versão implementada, o veículo se locomove primeiramente nos eixos X e Y , buscando o menor caminho até o alvo. Em caso de encontro com obstáculos, o veículo os desvia até chegar ao alvo. Após minimizar a distância nos eixos X e Y , o veículo se move no eixo Z , minimizando a distância final.

Para verificar que o veículo encontrou um obstáculo, é analisada a distância do menor raio do laser. Se a distância for menor que 0,5 metros, o veículo gira e contorna no sentido horário o obstáculo. Para verificar se o veículo contornou o obstáculo, analisa-se a distância do raio do laser apontado em direção ao alvo. Se a distância desse raio for maior, o veículo sair do modo contorno e vai em direção ao alvo.

4 METODOLOGIA

Neste Capítulo é apresentado a metodologia aplicada ao trabalho baseado no *setup* experimental previamente discutido.

4.1 Navegação sem mapa orientada a ponto alvo

Navegação é a ciência ou arte de guiar um robô móvel no sentido de como viajar no ambiente [Alves et al., 2011]. A essência do problema relacionado a navegação pode ser resumidamente descrita através de três questões: “Onde estou?”, “Onde vou?” e “Como chego lá?”. Os dois primeiros questionamentos estão mais fortemente relacionados ao sistema de localização e sensoramento do veículo enquanto que a última questão está relacionada à inteligência do veículo. A definição do sistema de navegação é influenciada pelas restrições impostas pelo ambiente e pelo propósito do robô em si [Alves et al., 2011]. O sistema de navegação tenta encontrar o melhor caminho possível baseado na informação coletada dos sensores que representam um mapa local ou global, geralmente representado por coordenadas cartesianas.

Partindo desse contexto, navegação sem mapa orientada a alvo pode ser definida como sendo a ação de ir de uma posição cartesiana inicial (X_i, Y_i, Z_i) no espaço a uma posição cartesiana alvo (X_t, Y_t, Z_t) no espaço. Na Figura 11 pode-se observar um exemplo de navegação sem mapa orientada a alvo em um dos cenários de simulação desenvolvido neste trabalho. Nesse exemplo, o veículo parte de uma posição inicial $(0, 0; 0, 0; 1, 0)$ até uma posição alvo final de $(4, 0; -0, 5; 1, 0)$ através de uma velocidade linear (VL_x, VL_y, VL_z) .

Além da posição cartesiana (X, Y, Z) , o veículo também assume uma orientação no espaço. Em uma mesma posição no espaço o veículo pode girar em torno dos seus eixos de rotação, assumindo diferentes orientações. Especificamente, um veículo baseado em quadrotor possui pouca atuação nos eixos de rotação X e Y devido a suas características físicas [Valavanis and Vachtsevanos, 2015]. Dessa forma, o veículo possui principalmente velocidade angular VA_z .

Navegação sem mapa é essencialmente um tipo de navegação em que nenhum mo-

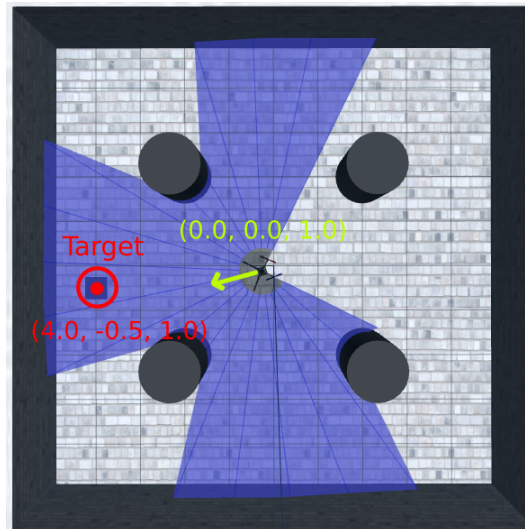


Figura 11: Exemplo de navegação sem mapa orientada a alvo.

mento um mapa do cenário é construído [Güzel, 2013]. Abordagens de navegação utilizando mapas, tais como SLAM, possuem características que fazem com que seja necessário construir o ambiente de obstáculos de forma manual. Por definição, essa é uma tarefa que consome muito tempo, tanto para construir o mapa quanto para aprimorá-lo, além de existir uma dependência muito grande na precisão dos sensores para que todo o processo funcione apropriadamente. Avanços recentes em técnicas de IA e processamento de imagem juntamente com evoluções no *hardware* permitiram que as desvantagens do uso de mapas para realizar a navegação pudessem ser otimizadas.

Navegação sem mapa é geralmente realizada utilizando informação visual de uma câmera e/ou informação de distância de um sensor de distância, podendo ser orientada a ponto alvo. Para este trabalho, ambas as tarefas propostas de navegação e *waypoint navigation* orientadas a alvo também são baseadas em um contexto sem mapa, exploradas utilizando somente informação de distância e localização em agentes de Deep-RL.

A navegação sem mapa orientada a alvo é uma das tarefas propostas para avaliar a capacidade de aprendizado dos agentes de Deep-RL neste trabalho. Durante o treinamento dos agentes, posições alvo aleatórias foram geradas para cada episódio dentro do limite físico de cada cenário desenvolvido. Busca-se com isso que o agente seja capaz de explorar todo o cenário.

Como complemento a tarefa de navegação sem mapa orientada a alvo também propõe-se uma segunda tarefa para avaliar os agentes: navegação sem mapa orientada a múltiplos alvos (do inglês *Waypoint Navigation*). Nessa tarefa, várias posições alvo amostradas a uma certa distância entre elas são definidas. Busca-se que o veículo navegue entre cada um dos pontos no espaço a fim de avaliar com maior robustez às abordagens.

4.2 Aprendizado por Reforço Profundo

Aprendizado por reforço ou RL (do inglês *Reinforcement Learning*) busca essencialmente fazer um agente interagir com um ambiente para aprender uma política ótima através de tentativa e erro [Li, 2017]. RL têm sido usado em uma de campos de pesquisa, com destaque para a área da robótica.

A integração de RL com redes neurais artificiais é bastante usado, tendo produzido resultados importantes ao longo do tempo [Li, 2017]. Com os resultados interessantes que Aprendizado Profundo ou DL (do inglês *Deep Learning*) têm conseguido obter graças a *big data*, computadores poderosos, novas técnicas de algoritmos e um aporte financeiro em crescimento, é possível observar uma renascimento de RL. Essa nova combinação de RL com redes neurais artificiais de DL é a essência que define o Aprendizado por Reforço Profundo ou Deep-RL. Deep-RL têm prevalecido nos últimos anos no avanço do desenvolvimento de jogos digitais, processamento de linguagem e robótica [Li, 2017]. Vários avanços importantes que ocorreram nas áreas da robótica e AI nos últimos anos estão relacionados à Deep-RL, tais como AlphaGo [Silver et al., 2016], computador neural diferenciável [Graves et al., 2016], Q-network [Mnih et al., 2015], etc.

DL têm ajudado RL a fazer grande progresso nos últimos anos e isso se deve ao fato de que DL proporciona engenharia automática de recursos e aprendizado de ponta a ponta de tal forma que a dependência de domínio do conhecimento é reduzida significativamente ou até mesmo removida. A engenharia de recursos costumava ser feita de forma manual, sendo lenta e ineficiente. As redes neurais artificiais de DL ofereceram a expressividade e a flexibilidade que as tarefas básicas dos avanços recentes em RL precisavam.

Vários algoritmos e técnicas de Deep-RL foram desenvolvidas com foco em aplicações na robótica ou acabaram se tornando muito úteis nessa área. Algumas das técnicas mais tradicionais, tais como o algoritmo de Q-network profundo ou DQN [Mnih et al., 2013], são capazes de produzir agentes com capacidade humana de performance, como no próprio trabalho de Mnih et al. [2015]. Entretanto, essas técnicas mais tradicionais produzem bons resultados somente em um espaço discreto de ações, sem apresentar um desempenho satisfatório em espaços complexos de observação. Sendo que muito do contexto que envolve a robótica incluindo UUVs e UAVs é baseado em espaço contínuo de ações e algoritmos tais como DQN não podem ser aplicados em domínio contínuo, outros algoritmos/técnicas precisaram ser desenvolvidas para abordar problemas desse tipo.

A forma mais comum de usar Deep-RL na robótica de modo geral se baseia em técnicas do tipo Ator-Crítico. Esses métodos são extensões dos métodos tradicionais para que Deep-RL funcione no espaço contínuo de ações. Algumas dessas técnicas foram usadas como base para desenvolver a metodologia deste trabalho e serão discutidas com mais detalhes na sequência.

4.2.1 Abordagem baseada em DDPG - NDRL-D

Silver et al. [2014] propôs um algoritmo baseado em Política do Gradiente Determinística ou DPG (do inglês Deterministic Policy Gradient) para RL como alternativa para a definição de políticas, que até então tinham um viés estocástico. DPG é o gradiente esperado da função de valor de ação, que se integra no espaço de estado; enquanto que, no caso estocástico, o gradiente de políticas se integra nos espaços de estado e de ação [Li, 2017]. Ou seja, DPG pode ser determinado de forma mais eficiente que a política do gradiente estocástico ou SPG (do inglês Stochastic Policy Gradient).

DPG é um algoritmo *off-policy* ator-crítico que aprende uma política alvo determinística através de uma política de comportamento exploratório e garante um gradiente de política imparcial com funções de aproximações. DPG foi introduzido para problemas de RL com espaço contínuo de ações. DPG apresentou superioridade à SPG em várias tarefas com grande dimensão no espaço de ações, como no controle do braço de um polvo, por exemplo.

Lillicrap et al. [2015] observou os algoritmos DQN e DPG e os estendeu criando um novo do tipo ator-crítico e *model-free* para espaço contínuo de ações. Assim como o DPG, Política do Gradiente Determinístico Profunda ou DDPG evita a otimização de uma ação a cada passo para obter uma política gulosa assim como em *Q-Learning*, tornando-o inviável no espaço de ações complexas com aproximadores de funções grandes e sem restrições, com em redes de DL.

DDPG consiste essencialmente de um método do tipo ator-crítico [Konda and Tsitsiklis, 2000] que utiliza funções de aproximações que são capazes de aprender políticas no espaço de ações contínuas. Duas redes neurais artificiais são usadas, uma para o ator e uma para o crítico. As redes produzem a ação para o estado atual do agente e geram um sinal de erro de diferença temporal a cada passo. A saída da rede crítica é a ação dada pela ator, enquanto que a saída da rede do ator é o valor real que representa uma ação escolhida para o espaço contínuo de ações.

Por ser também um algoritmo do tipo *off-policy*, DDPG aprende uma política de ator a partir de experiências de uma política de exploração. Isso é realizado adicionando ruído de um processo de ruído para a política do ator. Em outras palavras, um ruído amostrado μ' é adicionado à política da rede do ator, dado um processo de ruído $\mathcal{N}$ dessa forma:

$$\mu' = \mu(s_t) + \mathcal{N}. \quad (1)$$

$\mathcal{N}$ pode ser escolhido para atender as demandas específicas, sendo que o processo Ornstein-Uhlenbeck (OU) [Uhlenbeck and Ornstein, 1930] é geralmente escolhido para controles físicos. Para o presente trabalho o processo de ruído Ornstein-Uhlenbeck foi utilizado com os parâmetros (volatilidade σ , *drift* θ) da tabela 5 para o ruído do agente baseado em DDPG.

Tabela 5: Parâmetros de ruído Ornstein-Uhlenbeck usado na abordagem baseada em DDPG.

Constante	Valor
σ	0,5
θ	0,15

Devido à uma grande quantidade de variâncias na aproximação de uma *Q-function* resultante de muitas trajetórias simuladas temporalmente correlacionadas durante o treinamento e teste, o uso de uma memória de repetição [Schaul et al., 2015] é sugerida durante o treinamento [Schaul et al., 2015]. Como uma memória de *replay* é possível salvar os resultados da exploração durante o episódio e usar posteriormente de forma aleatória para quebrar as correlações temporais dentro de diferentes episódios para o aprendizado. Em outras palavras, isso permite um “aprendizado recente” que aumenta a velocidade de aprendizado e proporciona que uma melhor performance geral possa ser alcançada. Uma memória de *replay* com *buffer* de 50000 amostras foi utilizada no treinamento de ambos agentes neste trabalho.

Para criar uma memória de *replay* é necessário criar uma rede alvo para regular o aprendizado e a estabilidade. Essa rede é basicamente uma cópia das redes do ator e do crítico com atualizações suaves nos pesos. Os novos pesos da rede são atualizados de lentamente, levando em consideração um fator τ dessa forma:

$$\theta' = \tau\theta + (1 - \tau)\theta'. \quad (2)$$

Devido a todas as características, DDPG se tornou o cerne das aplicações de Deep-RL de robôs móveis terrestres, sendo utilizada até em UAVs e UUVs. Mais de 20 tarefas físicas simuladas de variadas dificuldades utilizando a mesma abordagem de DDPG, com mesma arquitetura de rede e hiperparâmetros foram abordadas com DDPG [Li, 2017].

Para abordagem baseada em DDPG foi dada a nomenclatura NDRL-D do inglês (Navigation Deep Reinforcement Learning Deterministic). Mais detalhes sobre a abordagem usada neste trabalho, contendo informações sobre a arquitetura de rede e hiperparâmetros, são apresentadas no Capítulo 4 de metodologia.

4.2.2 Abordagem baseada em SAC - NDRL-S

Os algoritmos de Deep-RL baseados em *model-free*, tais como DDPG, têm sido aplicados em um grande número de problemas que envolvem tarefas complexas e desafiadoras de tomada de decisão e controle. Entretanto, esses métodos geralmente apresentam dois grandes desafios: grande complexidade de amostras e propriedades de convergência frágeis, que demandam uma necessidade de refinamento metódica dos hiperparâmetros. Por consequência, esses dois desafios que algoritmos de RL baseados em *model free* apre-

sentam acabam por limitar a aplicabilidade desses métodos no mundo real.

Partindo desse contexto, surge o método *Soft Actor-Critic* (SAC) [Haarnoja et al., 2018]. SAC é um algoritmo de Deep-RL *off-policy*, ator-crítico e estocástico que aprende políticas no espaço contínuo de ações através do uso de funções de aproximação. O método SAC realiza, essencialmente, uma ponte entre algoritmos baseados em otimização de políticas estocásticas e algoritmos baseados em DDPG. O método foi desenvolvido em um formato com três ANNs: a rede do ator, a rede do crítico e também uma rede para a função de valor [Littman, 2001]. Segundo Haarnoja et al. [2018] essa divisão proporciona um treinamento mais estável. A predição das ações do estado atual é realizada por essas redes juntamente com um erro de diferença temporal a cada passo.

Além de focar somente na maximização do sistema de recompensa, o algoritmo também tenta maximizar a entropia da política, o que representa basicamente o quão imprevisível é uma determinada variável. Ou seja, o algoritmo busca ter sucesso na tarefa ao mesmo tempo que a ação é realizada de forma mais aleatória possível. Quanto mais alta a entropia de uma variável, maior o encorajamento para a exploração no agente. Uma variável que nunca teve seu valor alterado possui zero de entropia e não possui nenhum encorajamento. Esse encorajamento é feito através da definição de probabilidade igual para ações com valores similares de *Q-Value* e garantindo que uma ação particular e inconsistente na *Q-function* não faça isso falhar.

Através dessa combinação de atualizações *off-policy* com uma formulação estável de ator-crítico estocástico, o método SAC consegue atingir desempenho do estado da arte para tarefas de controle contínuo. Mais detalhes sobre a abordagem baseada em SAC usada neste trabalho, contendo informações sobre a arquitetura de rede e hiperparâmetros, são apresentadas na Seção 4. Para abordagem baseada em SAC foi dado a nomenclatura NDRL-S do inglês (Navigation Deep Reinforcement Learning Stochastic)

4.3 Estrutura dos agentes

Além de abordar o problema da navegação sem mapa, o presente trabalho também busca realizar uma comparação entre as variadas abordagens de Deep-RL utilizadas para solucionar as tarefas. Dessa forma, propõe-se desenvolver um modelo de estrutura para os agentes o mais idêntica possível, buscando evitar discrepâncias muito grandes entre cada abordagem. Na Figura 12 é possível observar a estrutura básica dos agentes para o contexto 2D, enquanto que a Figura 13 mostra a estrutura para o contexto 3D.

No contexto 2D, a rede do agente proposta possui ao total 24 entradas e 2 saídas, como pode ser observado na Figura 14a. Dos 24 valores de entrada, 20 são os valores de leitura do sensor de distância simulado, 2 valores representam a velocidade linear e a variação do ângulo de *yaw* da última ação e os 2 últimos valores representam a posição relativa do veículo e o ângulo em relação à posição alvo. O ângulo e a distância relativos foram

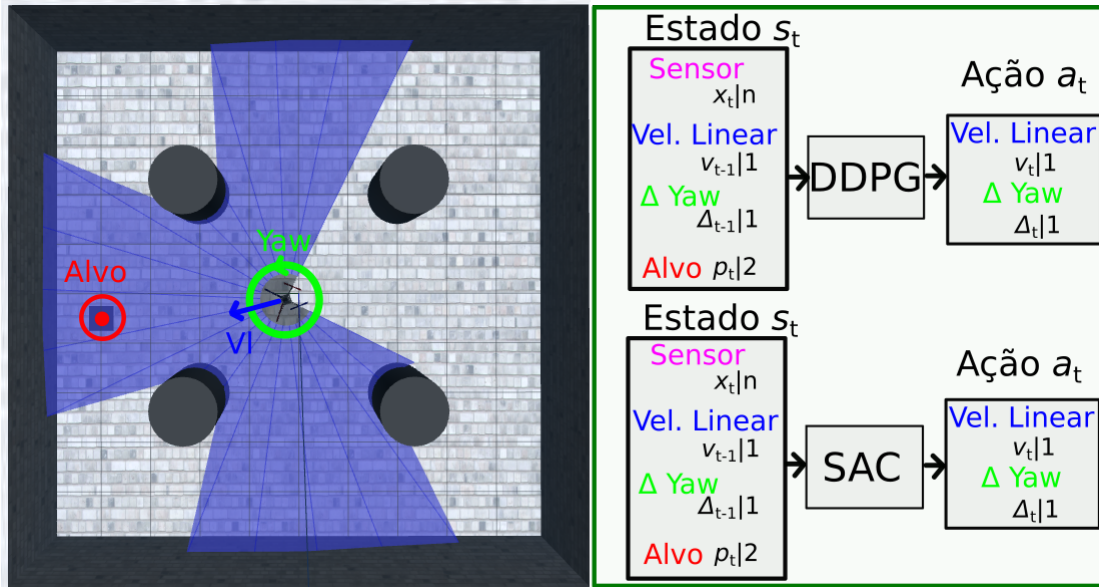


Figura 12: Veículo treinando em um dos cenários (esquerda) e o modelo básico de estrutura com entradas e saídas baseado nos métodos SAC e DDPG (direita) para o contexto 2D.

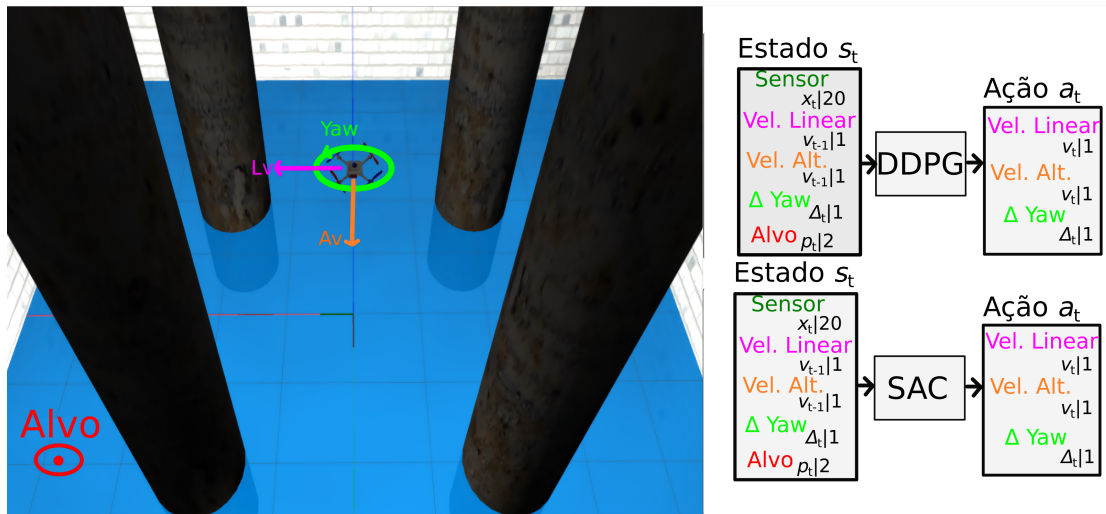


Figura 13: Veículo treinando em um dos cenários (esquerda) e o modelo básico de estrutura com entradas e saídas baseado nos métodos SAC e DDPG (direita) para o contexto 3D.

utilizados para forçar o agente a aprender a minimizá-los. As saídas da rede representam a velocidade linear e a variação do ângulo de yaw a ser aplicada no veículo.

Para o contexto 3D, um segundo ângulo relativo entre os planos Z e distância foi utilizado como informação adicional ao estado dos agentes. Além disso, para que a navegação no eixo Z fosse possível, uma terceira ação - velocidade de altitude - foi adicionado à estrutura. Como pode ser observado no Figura 14b, 26 valores são utilizados na entrada da rede que fornece as três ações para realizar a navegação no contexto 3D.

É importante ressaltar que a estrutura dos agentes (Figura 12 e 13) não muda para cada

cenário, somente para cada contexto. Dessa forma, a mesma estrutura é utilizada em cada um dos cenários. Além disso, a estrutura da rede, que será discutida com mais detalhes na sequência, é a mesma para cada cenário e ambiente.

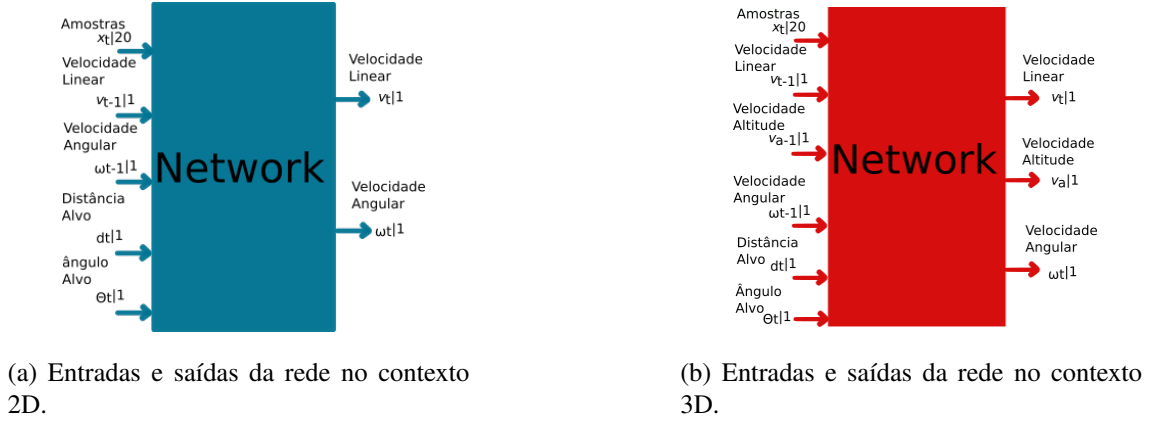


Figura 14: Entradas e saídas da rede dos agentes no contexto 2D.

4.3.1 Estrutura da rede

A estrutura de rede entre cada abordagem também foi projetada para ser o mais semelhante possível e foi baseada no trabalho do [Tai et al., 2017]. Busca-se analisar as diferenças que cada abordagem de Deep-RL em si proporciona à problemática proposta. A estrutura para a rede para ambas as abordagens no contexto 2D pode ser observada na Figura 15, enquanto que a Figura 16 mostra a estrutura para o contexto 3D. Para a rede do ator foram utilizadas 3 camadas totalmente conectadas de 512 neurônios cada, ativadas usando a função de ativação ReLU. A saída da terceira camada é conectada a um neurônio que representa a velocidade linear e a outro neurônio que representa a a variação do ângulo de *yaw* δ_{yaw} . O tipo de rede usada tanto para o ator quanto para o crítico é do tipo MLP (do inglês *Multi Layer Perceptron*).

A função de ativação de tangente hiperbólica (*Tanh*) foi utilizada nos neurônios de saída da rede. O alcance da saída da ativação *Tanh* que varia entre -1 e 1 no primeiro neurônio é escalonado para 0 e 0,25, representando a velocidade linear mínima e máxima a ser aplicado no veículo em *m/s* respectivamente. A saída do segundo neurônio é escalonado entre -0,25 e 0,25, representando a variação angular mínima e máxima a ser aplicado no veículo em *radianos* respectivamente. No contexto 3D, a terceira saída que representa a velocidade de altitude é escalonada entre -0,25 e 0,25 *m/s*. Os valores da saída são concatenados formando a ação que é enviada ao veículo.

A rede crítica, por sua vez, também possui 3 camadas completamente conectadas de 512 neurônios e ativadas usando ReLU. A entrada da ação a_t é concatenada com as entradas que representam o estado do agente.

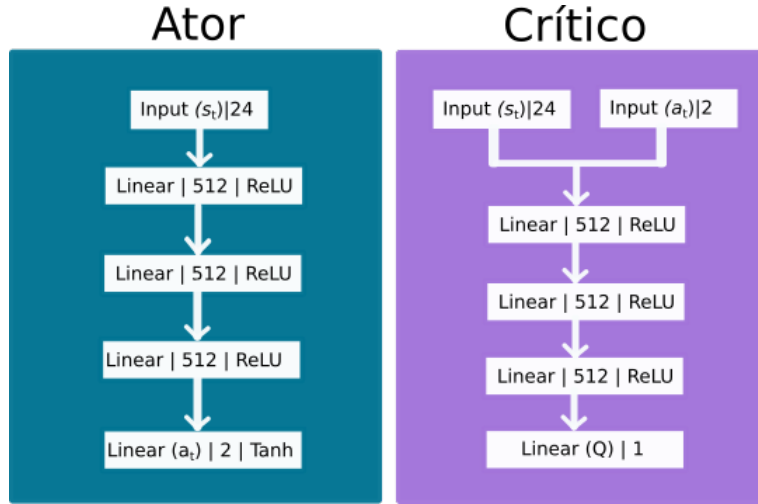


Figura 15: Estrutura da rede para o contexto 2D.

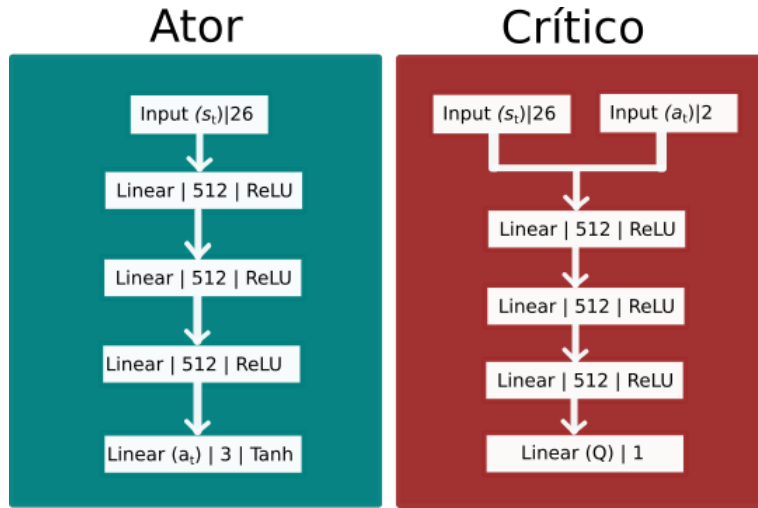


Figura 16: Estrutura da rede para o contexto 3D.

4.4 Função de Reforço

Independentemente da abordagem de Deep-RL a ser utilizada e do contexto, é necessário definir um sistema de recompensa e penalidade ou um sistema de reforço. Recompensas e penalidades são atribuídas ao agente, permitindo que ele possa aprender hiperparâmetros através dos passos de *feedforward* e *backpropagation* da rede. A estratégia de recompensa e penalidade básica proposta é a seguinte:

$$r(s_t, a_t) = \begin{cases} r_{chegada} & \text{if } d_t < c_d \\ r_{colisao} & \text{if } \min_x < c_o \end{cases} \quad (3)$$

Um reforço positivo $r_{chegada}$ é dado caso o agente alcance o alvo dentro de uma margem de c_d metros. Um reforço negativo $r_{colisao}$ é dado para o episódio caso o veículo colida com a parede ou algum dos obstáculos do cenário. A verificação é feita baseada na distância mínima da leitura do sensor de distância. Caso a distância seja menor que

uma distância de c_o metros, uma colisão é detectada. Os valores para os hiperparâmetros utilizados podem ser observados na Tabela 6.

Tabela 6: Hiperparâmetros utilizados no sistema de reforço.

Constantes	Valores
$r_{chegada}$	100
c_d	0,5m
$r_{colisao}$	-10
c_o	0,6m

Como foi abordado com mais detalhes na Seção 4.2, os agentes de Deep-RL fornecem como saída para o veículo um comando de velocidade. Por definição, os *plugins* utilizados para descrever o veículo não fornecem diretamente um tópico ou outra interface onde seja possível definir uma velocidade para o veículo, seja angular ou linear. Dessa forma, foi necessário implementar um *plugin* que permitisse que um comando de velocidade externo pudesse atuar no veículo. O *plugin* implementado¹ permite através de uma mensagem do tipo *Twist* que uma velocidade e angular em cada um dos eixos possa ser definida no veículo.

Os agentes foram estruturados de forma a fornecer uma velocidade linear e a variação do ângulo de *yaw* para o veículo, vide V_l e δ_ϕ . A velocidade linear e a variação do ângulo são convertidas e usadas para definir as velocidades cartesianas V_x e V_y da seguinte forma:

$$V_x = V_l * \cos(\phi + \delta_\phi)$$

$$V_y = V_l * \sin(\phi + \delta_\phi)$$

O ângulo ϕ representa o *yaw* do veículo. O ângulo de *yaw* é obtido através da conversão da orientação do veículo de quaternion para ângulo de Euler. A variação do ângulo de *yaw* fornecida pelo agente é somado ao ângulo de *yaw* atual, permitindo que V_x e V_y possam ser definidos a partir de uma velocidade linear V_l fornecida pela rede. As velocidades V_x e V_y são enviadas ao controlador geométrico [Lee et al., 2010] do veículo, permitindo que o veículo navegue nos eixos X e Y . Para o contexto 3D, a velocidade V_z foi utilizada como velocidade de altitude.

¹ https://github.com/ricardoGrando/hydrone_aerial_underwater_gazebo/blob/master/hydrone_aerial_underwater_ddpg/scripts/velocity_3D.py

5 RESULTADOS

Nesta Seção são apresentados os resultados obtidos no trabalho. Duas tarefas baseadas na premissa de navegação sem mapa foram desenvolvidas para analisar o grau de aprendizagem que os agentes de Deep-RL propostos conseguiram atingir. Na primeira tarefa, navegação sem mapa orientada a alvo, o veículo deve sair de uma posição no espaço e alcançar uma posição alvo no espaço. Na segunda tarefa, navegação sem mapa orientada a múltiplos pontos, pretende-se que o veículo seja capaz de passar por todos os alvos sequencialmente definidos.

Os testes do contexto 3D foram feitos primeiramente para o ambiente aéreo e depois para ambos os ambientes aéreo e subaquático de forma híbrida. Para este segundo tipo de teste somente a tarefa de navegação orientada a alvo foi realizada. A transição de meio ocorre na transição do eixo positivo para negativo de Z . Os testes no contexto 2D foram feitos somente para o ambiente aéreo como conceito de prova inicial para os agentes.

É importante ressaltar que para cada ambiente e cenário um agente de Deep-RL, seja NDRL-D ou NDRL-S, foi treinado. Um total de 100 tentativas para cada tarefa foi realizado e além de coletar o caminho percorrido durante cada tentativa informações referentes ao tempo, distância total percorrida e as ações aplicadas ao veículo também foram coletadas.

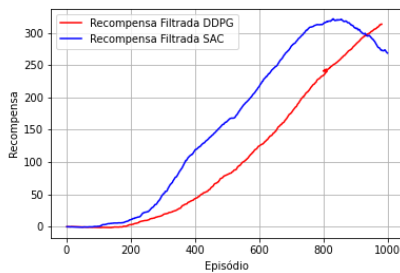
A posição inicial do veículo tanto para treinamento quanto para os testes em cada um dos cenários foi definida como sendo a posição cartesiana (0,0; 0,0; 2,5). Em todos os cenários o veículo foi posicionado com a sua parte frontal apontada em direção ao eixo positivo de X no simulador Gazebo.

Para o ambiente aéreo e subaquático no contexto 2D em nenhum momento alguma velocidade no eixo Z foi definida ou enviada para o veículo, apesar de que a posição no eixo Z do veículo possa ter sofrido alterações devido a propriedades físicas do veículo, que lhe proporciona uma característica de sistema subatuado. Esta característica é devido aos seis graus de liberdade de posição e orientação que são atuadas por somente quatro atuadores [Valavanis and Vachtsevanos, 2015]. Além disso, para o ambiente subaquático a força de empuxo também pode ter alterado a posição no eixo Z do veículo. Uma comparação com um algoritmo baseado no algoritmo BUG2 foi realizada para ambos contextos e cenários.

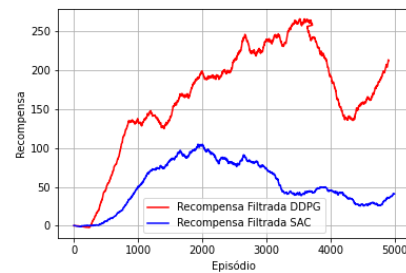
Os resultados para a navegação aérea no contexto 2D, 3D e 3D híbrida são apresentados na sequência.

5.1 Contexto 2D Aéreo

Na Figura 17 é possível observar o reforço médio obtido para cada abordagem em cada cenário. O eixo *Recompensa* representa a média móvel do reforço de 300 episódios anteriores a cada episódio, representado pelo eixo *Episódio*. O episódio termina quando o alvo é alcançado ou quando o veículo colide com a parede ou objetos. No primeiro cenário o treinamento foi realizado por 1000 episódios, enquanto que no segundo cenário o agente evoluiu por um total de 5000 episódios. Não houve um critério específico para a escolha do limite de episódios de treinamento, apenas a análise empírica do número aproximado episódios que os agentes foram capazes de realizar a navegação sem mapa.



(a) Resultado no primeiro cenário.



(b) Resultado no segundo cenário.

Figura 17: Recompensa ao longo dos episódios para cada abordagem no ambiente aéreo no contexto 2D.

De modo geral, é possível observar nas Figuras 17a e 17b como que nos primeiros episódios os agentes ainda estavam aprendendo, acumulando pouca recompensa, e como os agentes começaram a aprender a navegar até o alvo através do aumento do ganho de recompensa filtrada como a passagem dos episódios. Especificamente, pode-se observar que a maior complexidade do segundo cenário em relação ao primeiro demandou um número maior de episódios para que os agentes obtivessem uma quantidade similar de recompensa acumulada.

O desempenho dos agentes em cada tarefa está apresentado na sequência. É importante ressaltar que para a primeira tarefa o ponto alvo definido foi (2,0, 2,0, 2,5) para ambos os agentes no primeiro cenário e (3,6, 2,6, 2,5) no segundo cenário. Para a segunda tarefa a sequência de *waypoints* definidos para formar o caminho a ser seguido pode ser observado nas tabelas 7 e 8 para o primeiro e segundo cenário, respectivamente.

Tabela 7: *Waypoints* contexto 2D cenário 1.

Waypoint	Posição Cartesiana
1	(2, 0; 2, 0; 2, 5)
2	(0, 0; 3, 0; 2, 5)
3	(-2, 0; 2, 0; 2, 5)
4	(-2, 0; -2, 2; 2, 5)
5	(0, 0; -3, 0; 2, 5)
6	(2, 0; -2, 0; 2, 5)
7	(0, 0; 0, 0; 2, 5)

Tabela 8: *Waypoints* contexto 2D cenário 2.

Waypoint	Posição Cartesiana
1	(3, 6; 2.6; 2, 5)
2	(0, 0; 3.5; 2, 5)
3	(-3, 6; 3.0; 2, 5)
4	(-3, 6; 1.0; 2, 5)
5	(0, 0; 0.0; 2, 5)

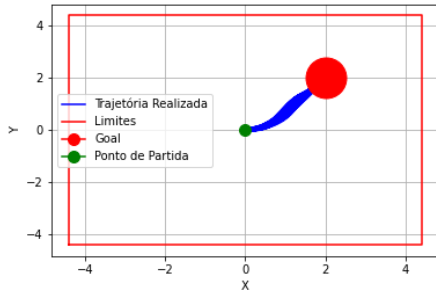
5.1.1 Navegação sem mapa orientada a alvo

Na Tabela 9 pode-se observar o total de tentativas em que a navegação foi realizada com sucesso para cada abordagem em cada cenário para a primeira tarefa. Além disso, também é mostrado o tempo médio e o desvio padrão de realização da tarefa.

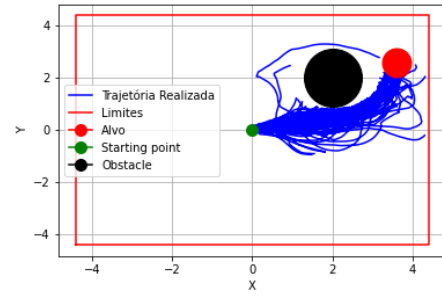
Tabela 9: Estatísticas da navegação orientada a alvo para o contexto 2D aéreo.

Cenário	Abordagem	Total Sucesso	Tempo Médio e Desvio Padrão
1	NDRL-D	100 %	13.48 ± 0.46
1	NDRL-S	93%	32.2 ± 14.52
1	BUG2	100%	16.789 ± 1.245
2	NDRL-D	35%	28.13 ± 11.06
2	NDRL-S	74%	35.617 ± 12.79
2	BUG2	27%	45.203 ± 29.019

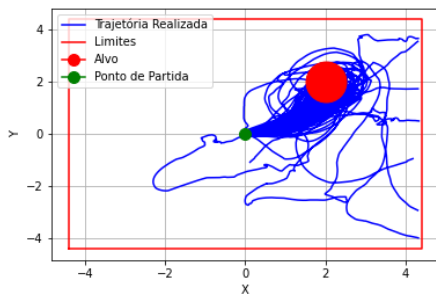
Na Figura 18 é possível observar as 100 tentativas de navegação realizada por ambos agentes e pelo algoritmo (BUG2) desenvolvido para comparação. O ponto verde representa a posição de saída do veículo, enquanto que o ponto vermelho representa o alvo. Para o segundo cenário, círculo em preto representa o duto de perfuração usado como obstáculo. O retângulo em vermelho representa os limites dos cenários vistos a partir de uma visão superior. A posição alvo para o segundo cenário foi definida de modo a forçar o veículo a ter que desviar do obstáculo, pois se caso ele fosse em linha reta ele colidiria com o obstáculo.



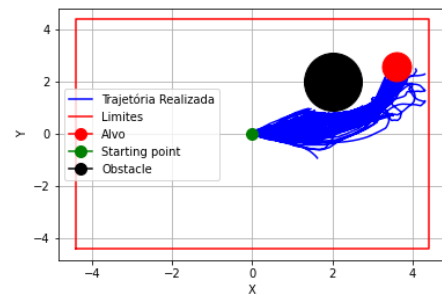
(a) NDRL-D no primeiro cenário.



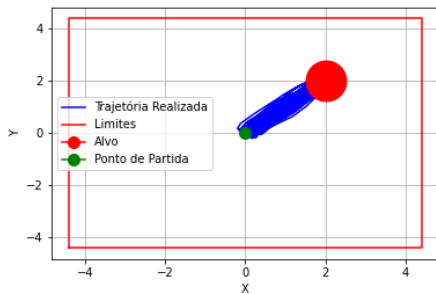
(b) NDRL-D no segundo cenário.



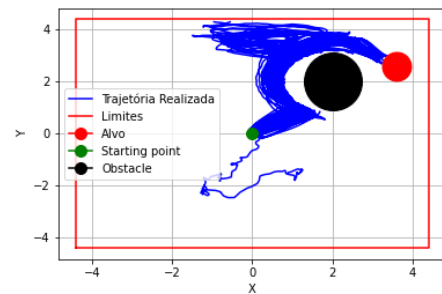
(c) NDRL-S no primeiro cenário.



(d) NDRL-S no segundo cenário.



(e) BUG2 no primeiro cenário.

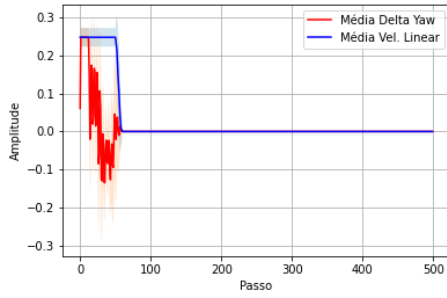


(f) BUG2 no segundo cenário.

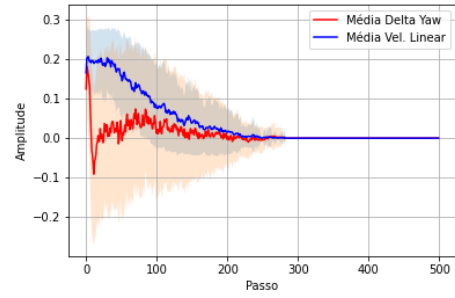
Figura 18: Caminho realizado durante as 100 tentativas na primeira tarefa no contexto 2D aéreo.

Na Figura 19 é possível observar a média de cada uma das ações de cada um dos agentes em cada passo durante a realização da tarefa. O sombreamento representa o desvio padrão.

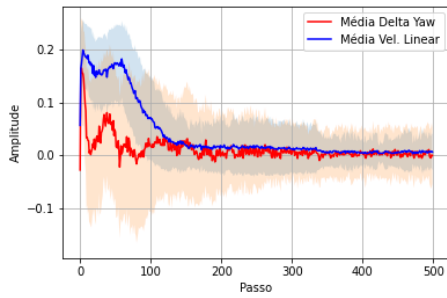
É possível concluir que ambos os agentes conseguiram realizar a tarefa nos dois cenários. No primeiro cenário o agente baseado em DDPG realizou trajetórias mais suaves em direção ao alvo, enquanto que o oposto pode ser observado no segundo cenário, onde o agente baseado em SAC foi capaz de desviar em média o obstáculo de forma menos discrepante mesmo obtendo um retorno acumulado total inferior ao agente baseado



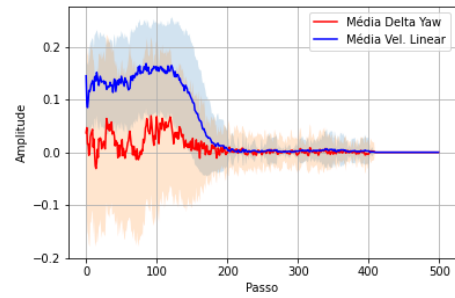
(a) NDRL-D no primeiro cenário.



(b) NDRL-D no segundo cenário.



(c) NDRL-S no primeiro cenário.



(d) NDRL-S no segundo cenário.

Figura 19: Ações dos agentes realizadas durante as 100 tentativas para a primeira tarefa no contexto 2D aéreo. Em vermelho, a média do delta yaw e em azul da velocidade linear. A sombra representa o desvio padrão.

em DDPG.

Quando comparados com a navegação sem mapa do algoritmo BUG2, os agentes de Deep-RL realizaram trajetórias mais irregulares no primeiro cenário. Entretanto, a trajetória realizada pelo algoritmo comportamental nesse cenário resultou em um número total de colisões maior que os agentes de Deep-RL. Em especial, pode-se destacar o agente NDRL-S que obteve uma performance bastante satisfatória com um percentual de navegação correta de 74%.

Além disso, é interessante analisar que apesar das diferenças fundamentais na implementação das duas abordagens, as ações a cada passo tendem a ser semelhantes. Isso pode ser observado nas Figuras 19. Pode-se observar também a característica determinística de NDRL-D ao obter um baixo desvio padrão, enquanto que o oposto pode ser observado para a abordagem determinística.

5.1.2 Navegação sem mapa orientada a múltiplos pontos

Na Tabela 10 pode-se observar o total de tentativas em que a navegação foi realizada com sucesso para cada abordagem em cada cenário para a segunda tarefa. Além do tempo médio e o desvio padrão de realização da tarefa, o percentual médio de completamento

do caminho formado pelos *waypoints* também é mostrado. Esse percentual é calculado baseado no número médio de *waypoints* que foram alcançados em cada uma das 100 tentativas. Para cada *waypoint* alcançado uma recompensa é dada.

Tabela 10: Estatísticas da navegação sem mapa orientada a múltiplos pontos para o contexto 2D aéreo.

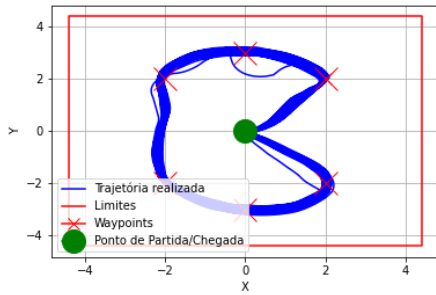
Cenário	Abordagem	Total Sucesso	Tempo Médio (s)	Porcentagem Caminho
1	NDRL-D	100%	99.37 ± 3.89	100%
1	NDRL-S	54%	164.18 ± 43.89	73.76%
1	BUG2	100%	145.24 ± 5.616	100%
2	NDRL-D	7%	55.43 ± 44.319	20.816%
2	NDRL-S	0%	36.611 ± 16.65	12.65%
2	BUG2	6%	87.328 ± 106.196	17.5%

Na Figura 20 é possível observar as 100 tentativas de navegação realizada por ambos agentes e pelo algoritmo BUG2 desenvolvido para comparação para a segunda tarefa. Os *waypoints* estão marcados na Figura em *X*.

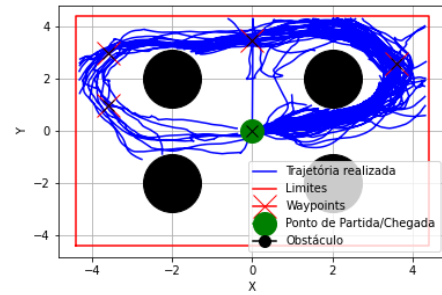
Por último, na Figura 21 também é possível observar a média de cada uma das ações de cada um dos agentes em cada passo durante a realização da segunda tarefa.

Na segunda tarefa pode-se observar que a abordagem determinística obteve melhores resultados. Pode-se destacar o tempo de realização da tarefa no primeiro cenário, onde a abordagem determinística obteve um tempo médio bastante inferior ao algoritmo BUG2, apesar de ambos terem realizada a navegação correta todas as vezes. Das trajetórias realizadas (Figuras 20), destaca-se a diferença entre a característica determinística e estocástica de cada uma das abordagens. Pode-se observar no primeiro cenário como NDRL-D apresentou trajetórias muitos similares e como NDRL-S apresentou maiores variações entre as trajetórias.

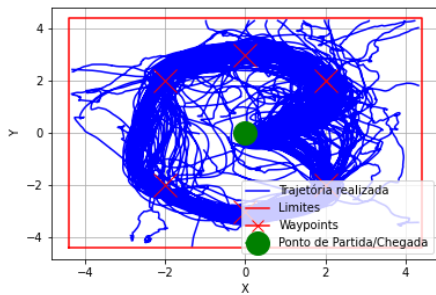
Também destaca-se a influência do vento nos resultados. O algoritmo BUG2 por exemplo não conseguiu obter bons resultados no segundo cenário devido a influência do vento, onde era esperado um desempenho próximo ao ideal desse algoritmo. Desse modo, destaca-se a capacidade de modelar a influência dos ventos que os agentes foram capazes de aprender, conseguindo deixar o veículo mais estável, obtendo melhores resultados.



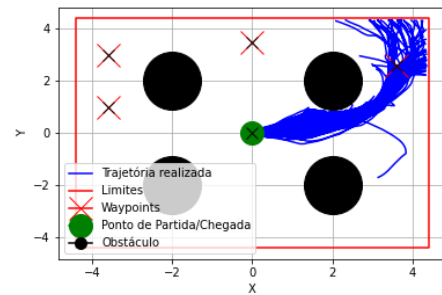
(a) NDRL-D no primeiro cenário.



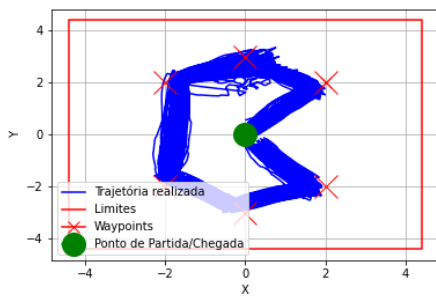
(b) NDRL-D no segundo cenário.



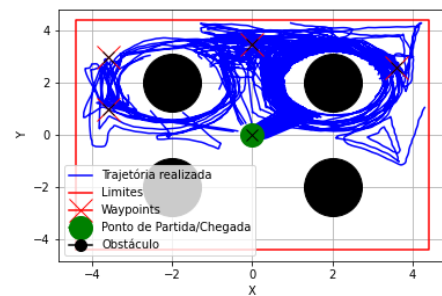
(c) NDRL-S no primeiro cenário.



(d) NDRL-S no segundo cenário.

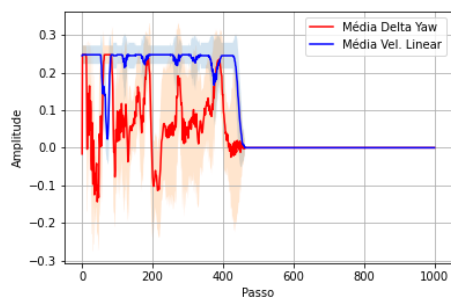


(e) BUG2 no primeiro cenário.

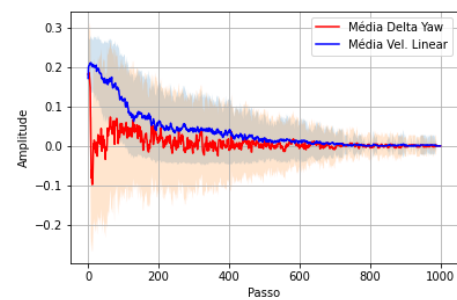


(f) BUG2 no segundo cenário.

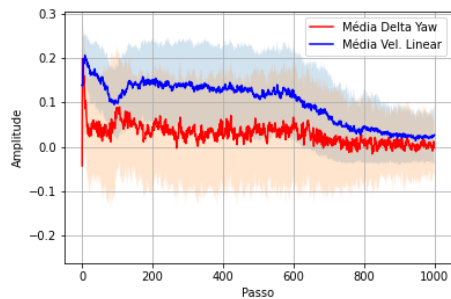
Figura 20: Caminho realizado durante as 100 tentativas na segunda tarefa no contexto 2D aéreo.



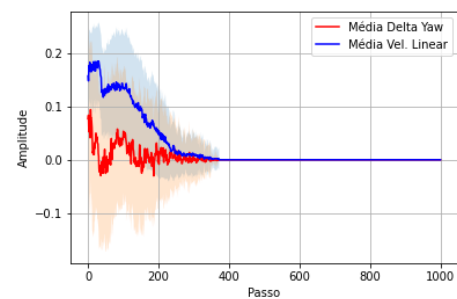
(a) NDRL-D no primeiro cenário.



(b) NDRL-D no segundo cenário.



(c) NDRL-S no primeiro cenário.



(d) NDRL-S no segundo cenário.

Figura 21: Ações dos agentes realizadas durante as 100 tentativas para a segunda tarefa no contexto 2D aéreo. Em vermelho a média do delta yaw e em azul da velocidade linear. A sombra representa o desvio padrão.

5.2 Contexto 2D Subaquático

Através de uma análise empírica, observou-se que os agentes treinados no ambiente aéreo no contexto 2D foram capazes de generalizar a ponto de apresentarem performance semelhante aos modelos treinados no ambiente subaquático, apesar de todas as diferenças entre ambientes e da informação do sensor de distância. Desta forma, ao invés de realizar o treinamento novamente, foi utilizado transferência de aprendizado para os modelos subaquáticos a partir dos modelos aéreos, sendo que os modelos treinados no ambiente aéreo foram utilizados no ambiente subaquático.

O desempenho dos agentes em cada tarefa está apresentado na sequência. É importante ressaltar que para a primeira tarefa o ponto alvo definido foi (2,0, 2,0, -0,5) para ambos os agentes no primeiro cenário e (3.6, 2.6, -0.5) no segundo cenário. Para a segunda tarefa a sequência de *waypoints* definidos para formar o caminho a ser seguido pode ser observado nas tabelas 11 e 12 para o primeiro e segundo cenário, respectivamente.

Tabela 11: *Waypoints* contexto 2D subaquático cenário 1.

Waypoint	Posição Cartesiana
1	(2, 0; 2, 0; -0, 5)
2	(0, 0; 3, 0; -0, 5)
3	(-2, 0; 2, 0; -0, 5)
4	(-2, 0; -2, 2; -0, 5)
5	(0, 0; -3, 0; -0, 5)
6	(2, 0; -2, 0; -0, 5)
7	(0, 0; 0, 0; -0, 5)

Tabela 12: *Waypoints* contexto 2D cenário 2.

Waypoint	Posição Cartesiana
1	(3, 6; 2, 6; -0, 5)
2	(0, 0; 3, 5; -0, 5)
3	(-3, 6; 3, 0; -0, 5)
4	(-3, 6; 1, 0; -0, 5)
5	(0, 0; 0, 0; -0, 5)

5.2.1 Navegação sem mapa orientada a alvo

Na Tabela 13 pode-se observar o total de tentativas em que a navegação foi realizada com sucesso para cada abordagem em cada cenário para a primeira tarefa. Além disso, também é mostrado o tempo médio e o desvio padrão de realização da tarefa.

Na Figura 22 é possível observar as 100 tentativas de navegação realizada por ambos agentes e pelo algoritmo BUG2 desenvolvido para comparação. A posição alvo para o

Tabela 13: Estatísticas da navegação orientada a alvo para o contexto 2D subaquático.

Cenário	Abordagem	Total Sucesso	Tempo Médio (s)
1	NDRL-D	100%	13.695 ± 0.75
1	NDRL-S	93%	41.09 ± 37.10
1	BUG2	100%	16.95 ± 0.95
2	NDRL-D	35%	35.151 ± 23.85
2	NDRL-S	74%	39.92 ± 37.469
2	BUG2	27%	42.13 ± 23.98

segundo cenário também foi definida de modo a forçar o veículo a ter que desviar do obstáculo, pois se caso ele fosse em linha reta ele colidiria com o obstáculo.

Na Figura 23 é possível observar a média de cada uma das ações de cada um dos agentes em cada passo durante a realização da tarefa. O sombreamento representa o desvio padrão.

No ambiente subaquático destaca-se a eficiência da transferência de aprendizado. Apesar da diferença do ambiente e das características das amostras do sonar, o desempenho dos agentes foi similar ao apresentado no ambiente aéreo no mesmo cenário. Destaca-se também o maior tempo médio para realizar a tarefa, entretanto.

5.2.2 Navegação sem mapa orientada a múltiplos pontos

Na Tabela 14 pode-se observar o total de tentativas em que a navegação foi realizada com sucesso para cada abordagem em cada cenário para a segunda tarefa.

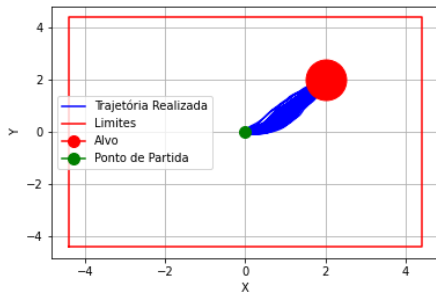
Tabela 14: Estatísticas da navegação sem mapa orientada a múltiplos pontos para o contexto 2D subaquático.

Cenário	Abordagem	Total Sucesso	Tempo Médio (s)	Porcentagem Caminho
1	NDRL-D	100%	93.84 ± 2.54	100%
1	NDRL-S	54%	267.96 ± 223.42	82.215%
1	BUG2	100%	131.49 ± 9.84	100%
2	NDRL-D	7%	54.97 ± 69.20	20.816%
2	NDRL-S	0%	81.16 ± 117.60	12.65%
2	BUG2	6%	90.14 ± 109.90	24.89%

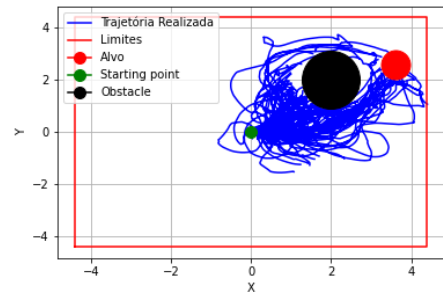
Na Figura 24 é possível observar as 100 tentativas de navegação realizada por ambos agentes e pelo algoritmo BUG2 desenvolvido para comparação para a segunda tarefa.

Por último, na Figura 25 também é possível observar a média de cada uma das ações de cada um dos agentes em cada passo durante a realização da segunda tarefa.

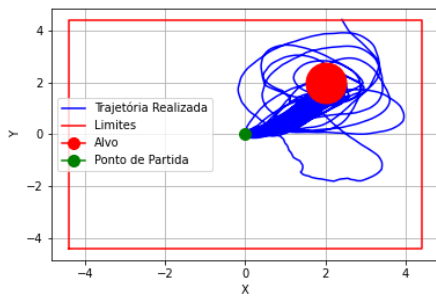
Para a segunda tarefa também destaca-se o tempo médio maior para realizar a tarefa e a semelhança das ações entre as abordagens (Figura 25). Apesar das diferenças, pode-se observar como as abordagens convergem e apresentam o mesmo espaço de ações.



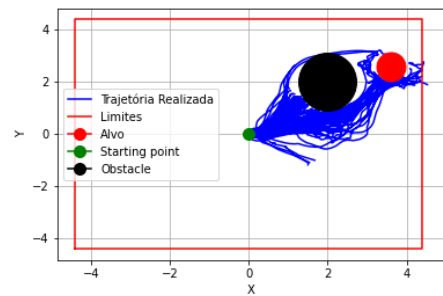
(a) NDRL-D no primeiro cenário.



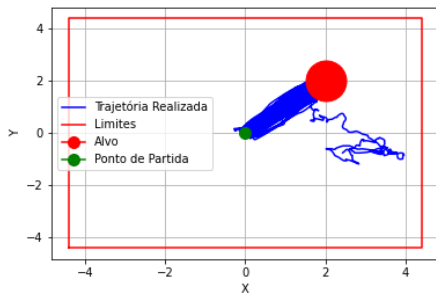
(b) NDRL-D no segundo cenário.



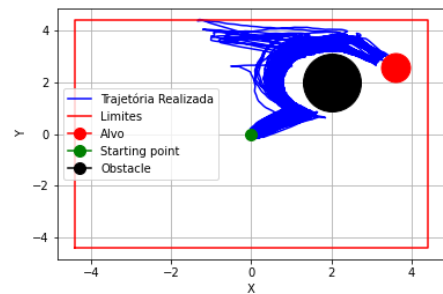
(c) NDRL-S no primeiro cenário.



(d) NDRL-S no segundo cenário.



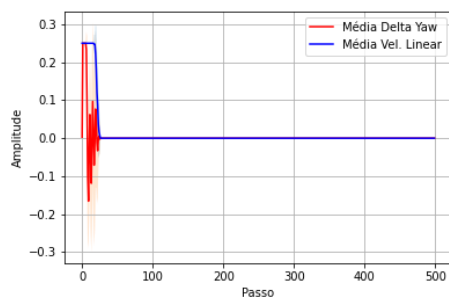
(e) BUG2 no primeiro cenário.



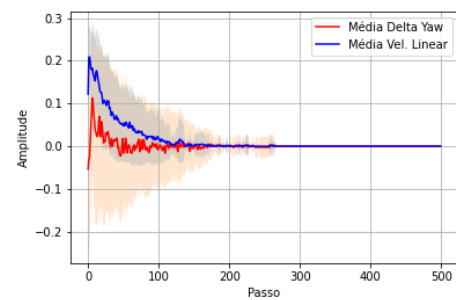
(f) BUG2 no segundo cenário.

Figura 22: Caminho realizado durante as 100 tentativas na primeira tarefa no contexto 2D subaquático.

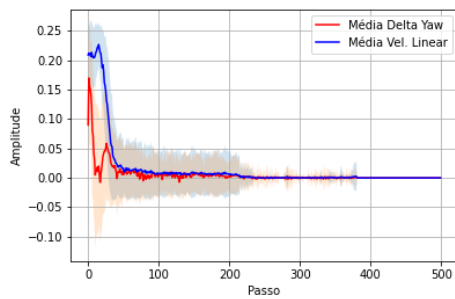
Assim como no ambiente aéreo, destaca-se a capacidade que os agentes desenvolveram de modelar a dinâmica do ambiente. Os agentes conseguiram aprender a assimilar qualquer força aplicada ao veículo, seja vento ou correnteza, e estabilizá-lo de forma a obter bons resultados em ambos os ambientes.



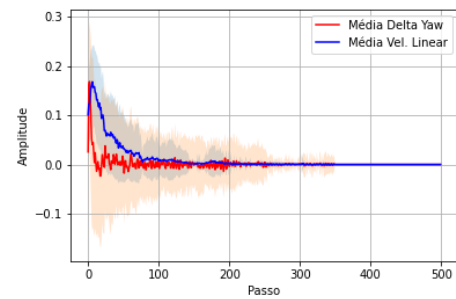
(a) NDRL-D no primeiro cenário.



(b) NDRL-D no segundo cenário.

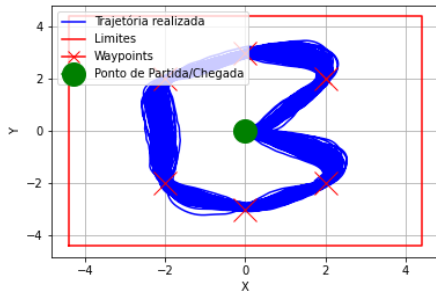


(c) NDRL-S no primeiro cenário.

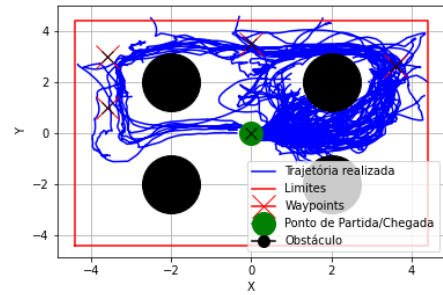


(d) NDRL-S no segundo cenário.

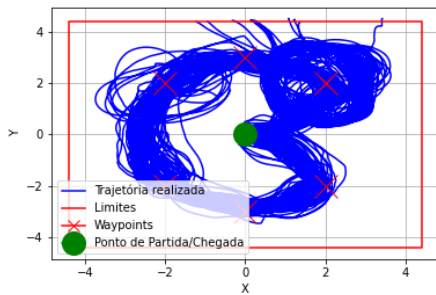
Figura 23: Ações dos agentes realizadas durante as 100 tentativas para a primeira tarefa no contexto 2D subaquático. Em vermelho a média do delta yaw e em azul da velocidade linear. A sombra representa o desvio padrão.



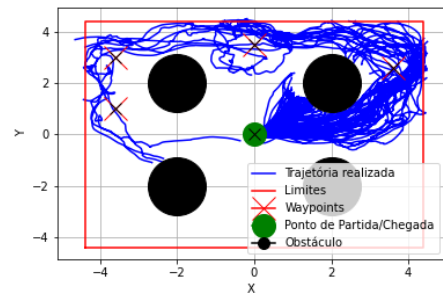
(a) NDRL-D no primeiro cenário.



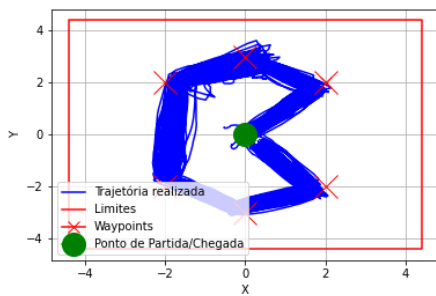
(b) NDRL-D no segundo cenário.



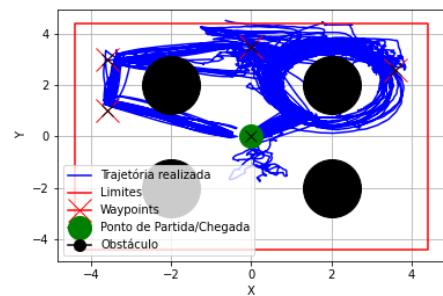
(c) NDRL-S no primeiro cenário.



(d) NDRL-S no segundo cenário.

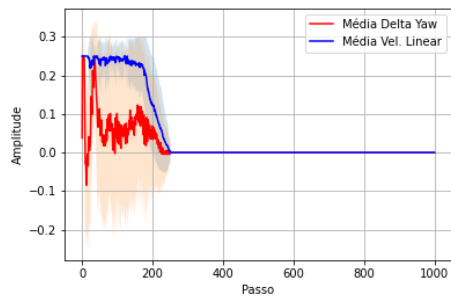


(e) BUG2 no primeiro cenário.

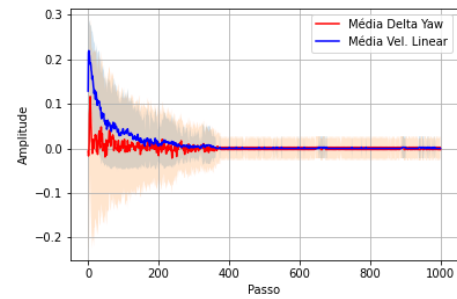


(f) BUG2 no segundo cenário.

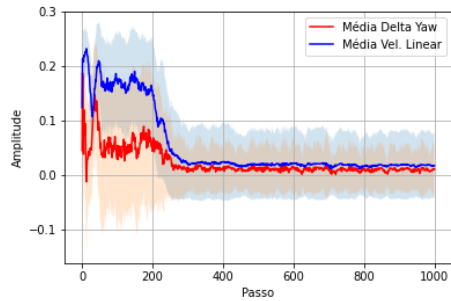
Figura 24: Caminho realizado durante as 100 tentativas na segunda tarefa no contexto 2D subaquático.



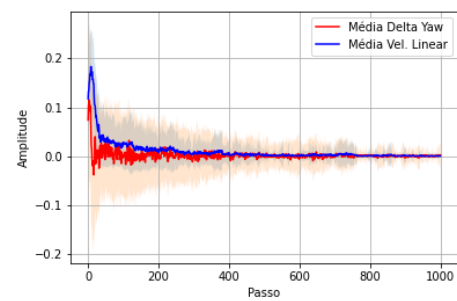
(a) NDRL-D no primeiro cenário.



(b) NDRL-D no segundo cenário.



(c) NDRL-S no primeiro cenário.

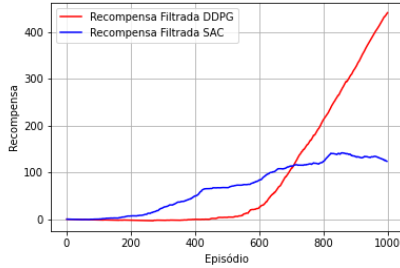


(d) NDRL-S no segundo cenário.

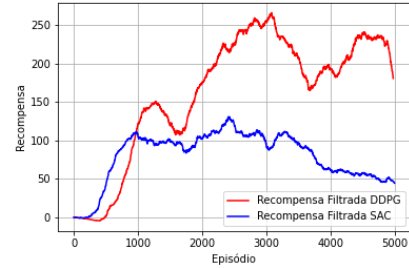
Figura 25: Ações dos agentes realizadas durante as 100 tentativas para a segunda tarefa no contexto 2D subaquático. Em vermelho a média do delta yaw e em azul da velocidade linear. A sombra representa o desvio padrão.

5.3 Contexto 3D

Na Figura 26 é possível observar o reforço médio obtido para cada abordagem em cada cenário. No primeiro cenário o treinamento também foi realizado por 1000 episódios, enquanto que no segundo cenário o agente evoluiu por um total de 5000 episódios.



(a) Abordagens no primeiro cenário.



(b) Abordagens no segundo cenário.

Figura 26: Recompensa ao longo dos episódios para cada abordagem no ambiente aéreo no contexto 3D.

O desempenho dos agentes em cada tarefa está apresentado na sequência. É importante ressaltar que para a primeira tarefa o ponto alvo definido foi (2,0; 2,0; 1,5) para ambos os agentes no primeiro cenário e (3,6; 2,6; 1,5) no segundo cenário. Para a segunda tarefa a sequência de *waypoints* definidos para formar o caminho a ser seguido pode ser observado nas tabelas 15 e 16 para o primeiro e segundo cenário, respectivamente.

Tabela 15: *Waypoints* contexto 3D cenário 1.

Waypoint	Posição Cartesiana
1	(1, 0; 1, 0; 2, 5)
2	(0, 0; 2, 0; 3, 0)
3	(-2, 0; 2, 0; 2, 0)
4	(-2, 0; -2, 0; 2, 5)
5	(0, 0; -2, 0; 2, 0)
6	(1, 0; -1, 0; 3, 0)
7	(0, 0; 0, 0; 2, 5)

5.3.1 Navegação sem mapa orientada a alvo

Na Tabela 17 pode-se observar o total de tentativas em que a navegação foi realizada com sucesso para cada abordagem em cada cenário para a primeira tarefa. O tempo médio e o desvio padrão de realização da tarefa também são amostrados.

Na Figura 27 é possível observar as 100 tentativas de navegação realizada por ambos agentes e pelo algoritmo BUG2 desenvolvido para comparação.

Tabela 16: *Waypoints* contexto 3D cenário 2.

Waypoint	Posição Cartesiana
1	(3, 6; 2, 6; 1, 5)
2	(0, 0; 3, 5; 2, 0)
3	(-3, 6; 3, 0; 3, 0)
4	(-3, 6; 1, 0; 2, 5)
5	(0, 0; 0, 0; 2, 5)

Tabela 17: Estatísticas da navegação orientada a alvo para o contexto 3D.

Cenário	Abordagem	Total Sucesso	Tempo Médio (s)
1	NDRL-D	100%	13.697 ± 0.85
1	NDRL-S	100%	27.038 ± 18.673
1	BUG2	100%	21.92 ± 1.341
2	NDRL-D	17%	49.276 ± 38.32
2	NDRL-S	19%	39.43 ± 22.21
2	BUG2	36%	51.93 ± 26.83

Na Figura 28 é possível observar a média de cada uma das ações de cada um dos agentes em cada passo durante a realização da tarefa. O sombreamento representa o desvio padrão.

Para o contexto 3D, destaca-se a maior dificuldade que esse contexto apresenta com relação à navegação 2D. Ambas as abordagens obtiveram um total de sucesso inferior em ambos os cenários quando comparadas ao algoritmo BUG2.

5.3.2 Navegação sem mapa orientada a múltiplos pontos

Na Tabela 18 pode-se observar o total de tentativas em que a navegação foi realizada com sucesso para cada abordagem em cada cenário para a segunda tarefa.

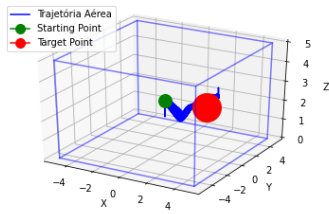
Tabela 18: Estatísticas da navegação sem mapa orientada a múltiplos pontos para o contexto 3D.

Cenário	Abordagem	Total Sucesso	Tempo Médio (s)	Porcentagem Caminho
1	NDRL-D	77%	76.99 ± 29.23	86.44%
1	NDRL-S	10%	135.220 ± 61.154	47.084%
1	BUG2	100%	145.94 ± 19.45	100%
2	NDRL-D	0%	39.8 ± 28.85	2.24%
2	NDRL-S	0%	42.44 ± 31.20	3.469%
2	BUG2	3%	90.95 ± 93.07	11.42%

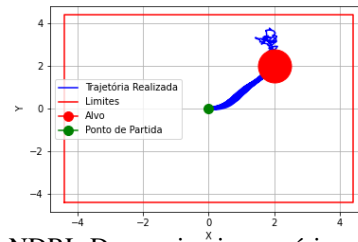
Na Figura 29 é possível observar as 100 tentativas de navegação realizada por ambos agentes e pelo algoritmo tradicional (BUG) desenvolvido para comparação para a segunda tarefa. Os *waypoints* estão marcados na Figura em *X*.

Por último, na Figura 30 também é possível observar a média de cada uma das ações de cada um dos agentes em cada passo durante a realização da segunda tarefa.

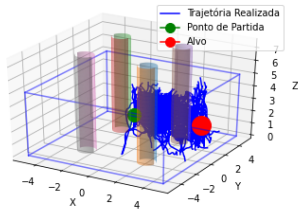
A segunda tarefa apresentou similaridades com a primeira tarefa em termos de desempenho. Ambos agentes e o algoritmo BUG2 apresentaram um total de sucesso ruim. O mesmo ocorreu para a porcentagem do caminho percorrido, onde ambos percorreram um percentual médio menor. Destaque negativo no segundo cenário, onde o percentual ficou próximo a zero. É importante ressaltar a dificuldade do cenário com obstáculos e a influência do vento no resultado.



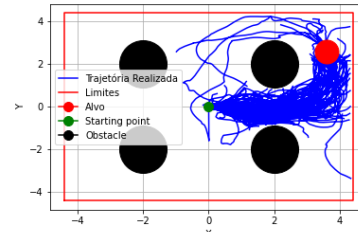
(a) NDRL-D no primeiro cenário.



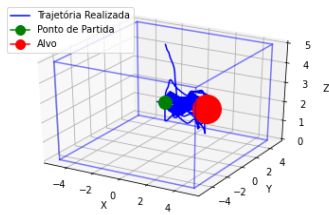
(b) NDRL-D no primeiro cenário visto de cima.



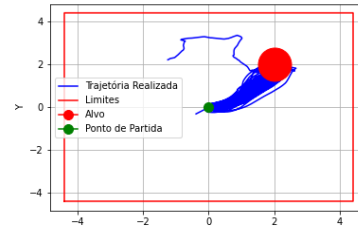
(c) NDRL-D no segundo cenário.



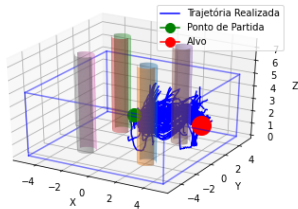
(d) NDRL-D no segundo cenário visto de cima.



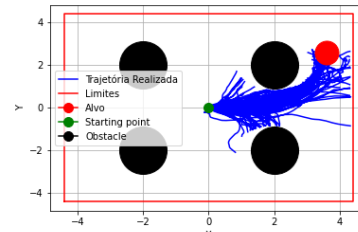
(e) NDRL-S no primeiro cenário.



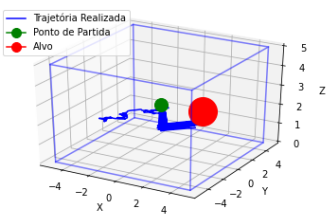
(f) NDRL-S no primeiro cenário visto de cima.



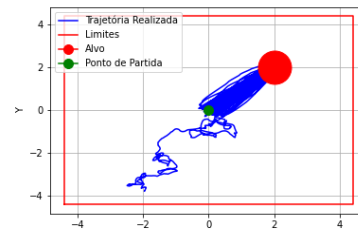
(g) NDRL-S no segundo cenário.



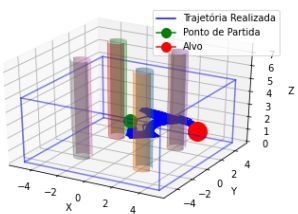
(h) NDRL-S no segundo cenário visto de cima.



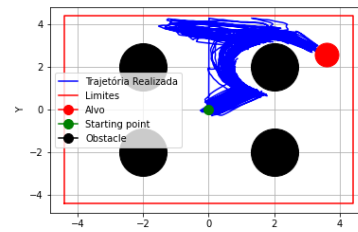
(i) BUG2 no primeiro cenário. /



(j) BUG2 no primeiro cenário visto de cima.

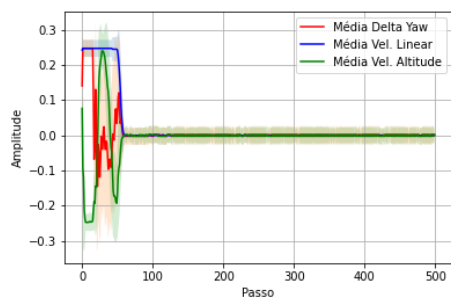


(k) BUG2 no segundo cenário. /

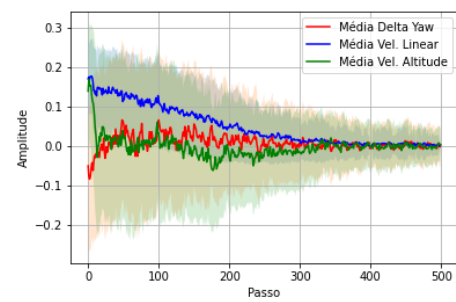


(l) BUG2 no segundo cenário visto de cima.

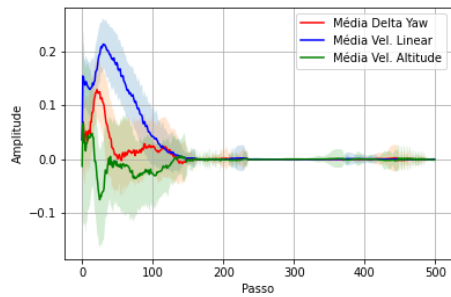
Figura 27: Caminho realizado durante as 100 tentativas na primeira tarefa no contexto 3D.



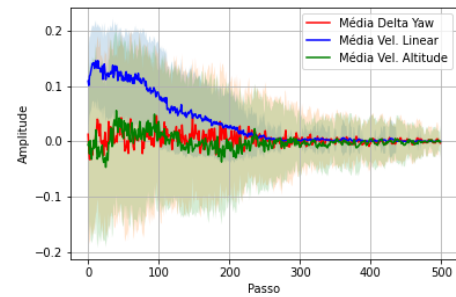
(a) NDRL-D no primeiro cenário.



(b) NDRL-D no segundo cenário.

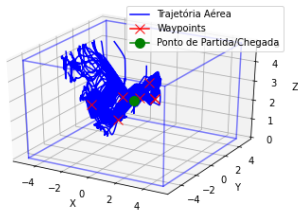


(c) NDRL-S no primeiro cenário.

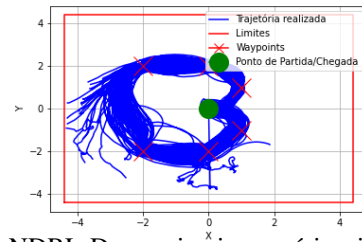


(d) NDRL-S no segundo cenário.

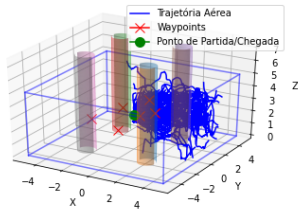
Figura 28: Ações dos agentes realizadas durante as 100 tentativas para a primeira tarefa no contexto 3D. Em vermelho a média do delta yaw, em azul da velocidade linear e em verde a velocidade de altitude. A sombra representa o desvio padrão.



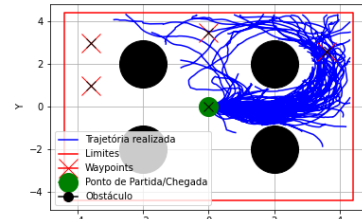
(a) NDRL-D no primeiro cenário.



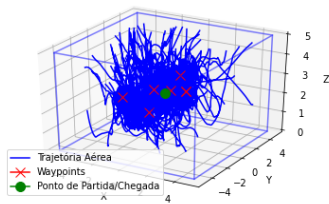
(b) NDRL-D no primeiro cenário visto de cima.



(c) NDRL-D no segundo cenário.



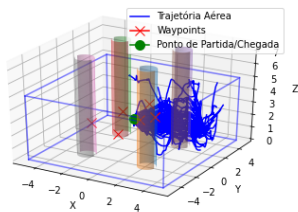
(d) NDRL-D no segundo cenário visto de cima.



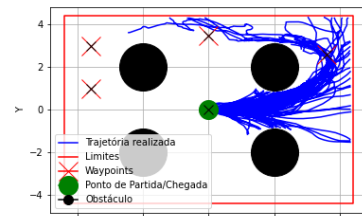
(e) NDRL-S no primeiro cenário.



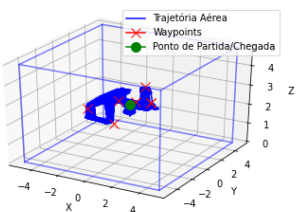
(f) NDRL-S no primeiro cenário visto de cima.



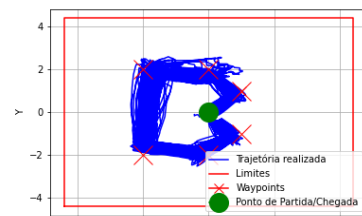
(g) NDRL-S no segundo cenário.



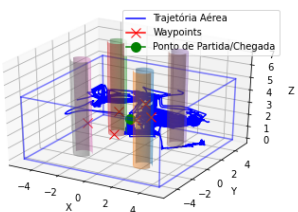
(h) NDRL-S no segundo cenário visto de cima.



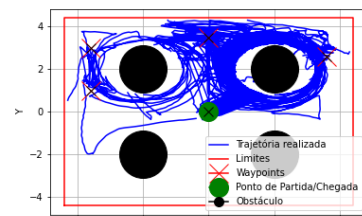
(i) BUG2 no primeiro cenário. /



(j) BUG2 no primeiro cenário visto de cima.

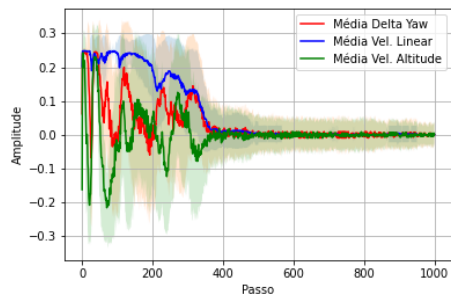


(k) BUG2 no segundo cenário. /

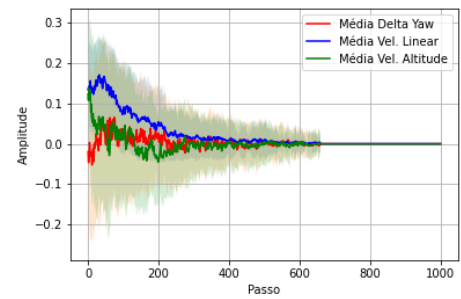


(l) BUG no segundo cenário visto de cima.

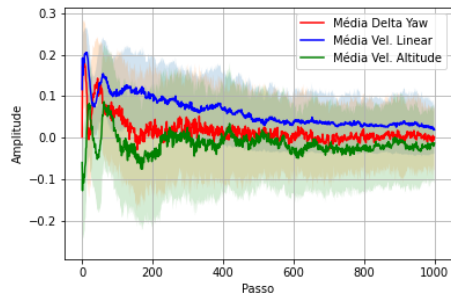
Figura 29: Caminho realizado durante as 100 tentativas na segunda tarefa no contexto 3D.



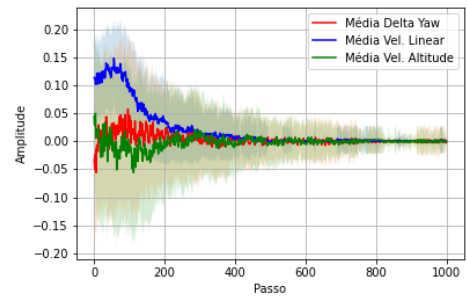
(a) NDRL-D no primeiro cenário.



(b) NDRL-D no segundo cenário.



(c) NDRL-S no primeiro cenário.

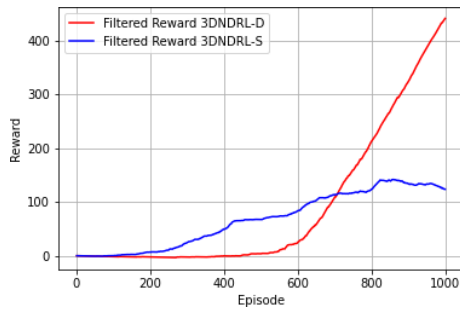


(d) NDRL-S no segundo cenário.

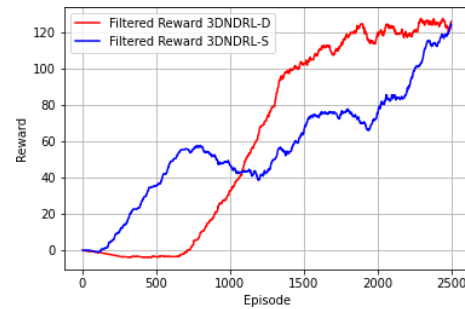
Figura 30: Ações dos agentes realizadas durante as 100 tentativas para a segunda tarefa no contexto 3D. Em vermelho a média do delta yaw, em azul da velocidade linear e em verde a velocidade de altitude. A sombra representa o desvio padrão.

5.4 Contexto 3D Híbrido

O reforço médio obtido para cada abordagem em cada cenário pode ser observado na Figura 26. Assim como nos contextos anteriores, o treinamento também foi realizado por 1000 episódios no primeiro cenário, enquanto que no segundo cenário o agente evoluiu por um total de 5000 episódios.



(a) Abordagens no primeiro cenário.



(b) Abordagens no segundo cenário.

Figura 31: Recompensa ao longo dos episódios para cada abordagem no ambiente aéreo no contexto 3D híbrido.

O desempenho dos agentes em cada tarefa está apresentado na sequência. É importante ressaltar que somente a primeira tarefa foi realizada para o contexto híbrido dada a maior complexidade na navegação entre meios diferentes. Entretanto, essa tarefa foi realizada partindo de pontos iniciais distintos (um aéreo e um subaquático), buscando analisar com maior robustez a transição de meio. Para o primeiro cenário, a posição aérea de partida foi definida como sendo a coordenada $(0,0; 0,0; 2,5)$ no mundo do Gazebo, enquanto que a posição subaquática de partida foi definida como sendo $(2,0; 3,0; -1,0)$. A posição alvo para o ponto de partida aéreo é a posição de partida subaquática, enquanto que o inverso ocorre para o ponto de partida subaquático. Dessa forma, em ambos os testes o veículo é obrigado a sair de um meio e transitar para o alvo que está no outro meio. Para o segundo cenário o mesmo foi definido, mudando-se somente as posições nas quais foram definidas como sendo $(0,0; 0,0; 2,5)$ e $(3,6; -2,4; -1,0)$, respectivamente. Busca-se também avaliar a capacidade de evitar colisão com obstáculos nesse cenário.

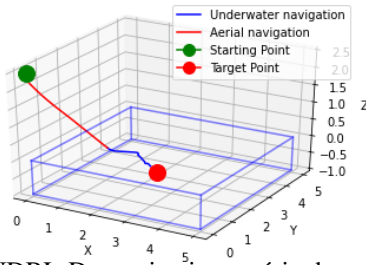
Na Tabela 19 pode-se observar o total de tentativas em que a navegação foi realizada com sucesso para cada abordagem em cada cenário. O tempo médio e o desvio padrão de realização da tarefa também são amostrados.

Na Figura 32 é possível observar 1 das 100 tentativas de navegação realizada por ambos agentes e pelo algoritmo BUG2, no primeiro cenário. Figura 33 mostra o mesmo para o segundo cenário.

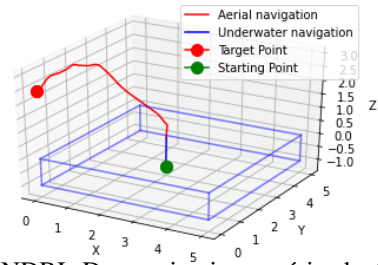
Nas Figuras 34 e 35 é possível observar a média e o desvio padrão de cada uma das ações de cada um dos agentes em cada passo durante a realização da tarefa.

Tabela 19: Estatísticas da navegação orientada a alvo para o contexto 3D híbrido.

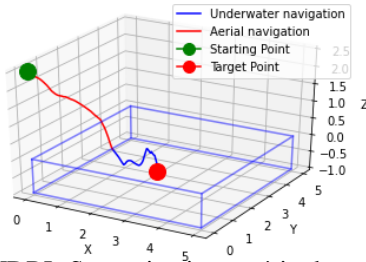
Cenário	Abordagem	Tempo Médio Ar (s)	Tempo Médio Água (s)	Total Sucesso
1	Ar-água-NDRL-D	15.29 ± 2.40	33.05 ± 10.91	96 %
1	Ar-água-NDRL-S	39.81 ± 23.16	23.64 ± 19.63	72 %
1	Ar-água-BUG2	31.963 ± 1.65	21.05 ± 0.199	100 %
1	Água-Ar-NDRL-D	18.78 ± 1.21	6.10 ± 0.17	97 %
1	Água-Ar-NDRL-S	33.98 ± 26.15	17.36 ± 12.87	75 %
1	Água-Ar-BUG2	32.74 ± 3.33	3.86 ± 0.28	100 %
2	Ar-água-NDRL-D	20.06 ± 13.92	59.83 ± 22.49	54 %
2	Ar-água-NDRL-S	60.88 ± 30.25	17.38 ± 16.10	37 %
2	Ar-água-BUG2	48.94 ± 28.03	12.55 ± 8.26	47 %
2	Água-Ar-NDRL-D	53.56 ± 31.28	5.98 ± 1.31	57 %
2	Água-Ar-NDRL-S	29.98 ± 14.48	6.61 ± 0.82	71 %
2	Água-Ar-BUG2	117.266 ± 79.54	3.775 ± 0.31	32 %



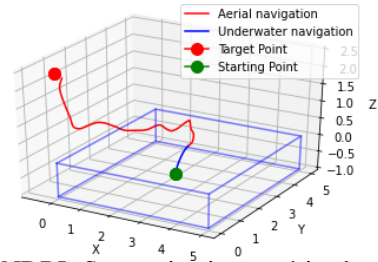
(a) NDRL-D no primeiro cenário do ar para a água.



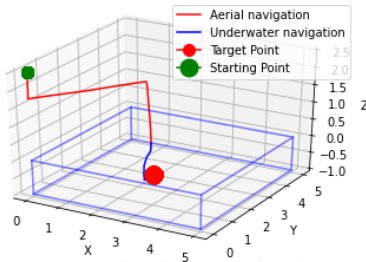
(b) NDRL-D no primeiro cenário da água para o ar.



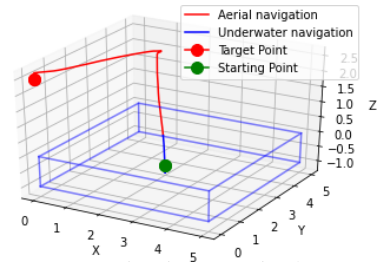
(c) NDRL-S no primeiro cenário do ar para a água.



(d) NDRL-S no primeiro cenário da água para o ar.



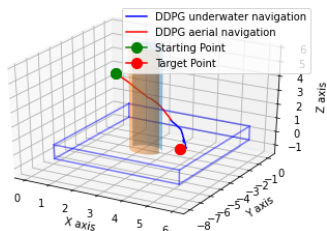
(e) BUG2 no primeiro cenário do ar para a água.



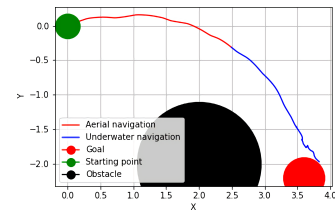
(f) BUG2 no primeiro cenário da água para o ar.

Figura 32: Caminho realizado durante 1 das 100 tentativas no contexto 3D híbrido no primeiro cenário.

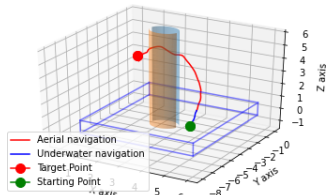
No contexto híbrido destaca-se a robustez dos agentes ao serem capazes de navegar do ar para a água e vice-versa com pequena variação no desempenho. Destaca-se também o total de sucesso das abordagens de Deep-RL quando comparadas com o BUG2 no segundo cenário. Assim como no contexto 2D ambas as abordagens foram mais eficazes em média no desvio dos obstáculos para chegar ao alvo em um ambiente diferente do ponto de partida. Nesse contexto fica ainda mais claro as características de cada abordagem no que diz respeito às ações a cada passo (Figura 34 e Figura 35). Pode-se observar novamente as características determinísticas e estocásticas e a semelhança no espaço de ações final.



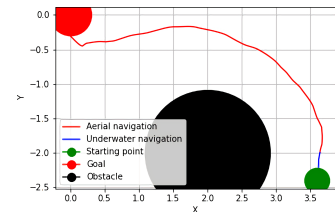
(a) NDRL-D no segundo cenário do ar para a água.



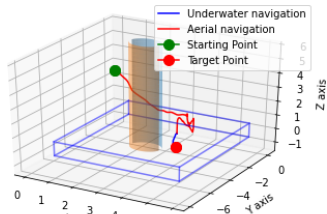
(b) NDRL-D no segundo cenário do ar para a água visto de cima.



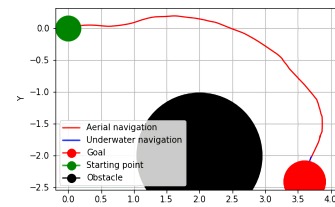
(c) NDRL-D no segundo cenário da água para o ar.



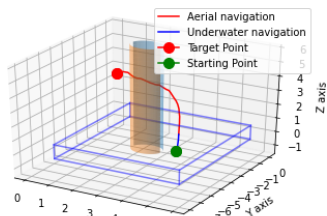
(d) NDRL-D no segundo cenário da água para o ar visto de cima.



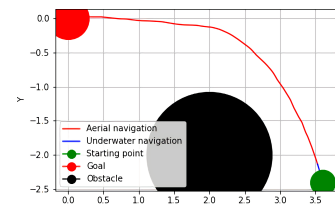
(e) NDRL-S no segundo cenário do ar para a água.



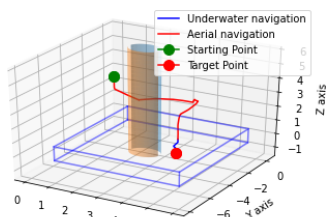
(f) NDRL-S no segundo cenário do ar para a água visto de cima.



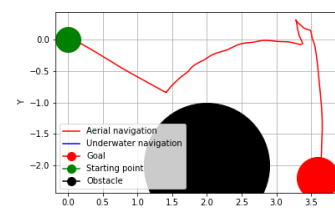
(g) NDRL-S no segundo cenário da água para o ar.



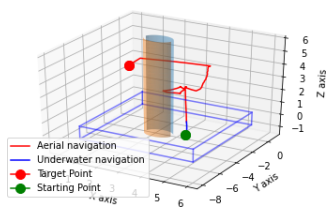
(h) NDRL-S no segundo cenário da água para o ar visto de cima.



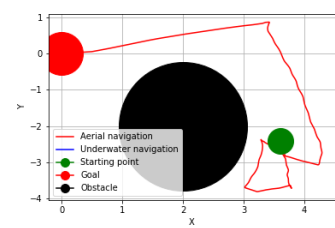
(i) BUG2 no segundo cenário do ar para a água.



(j) BUG2 no segundo cenário do ar para a água visto de cima.

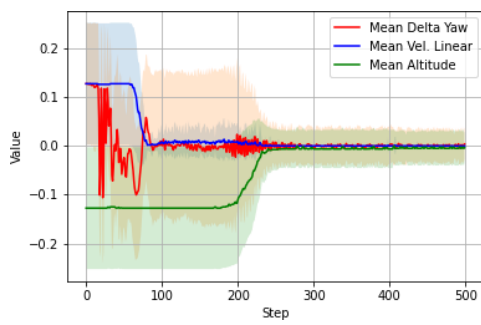


(k) BUG2 no segundo cenário da água para o ar.

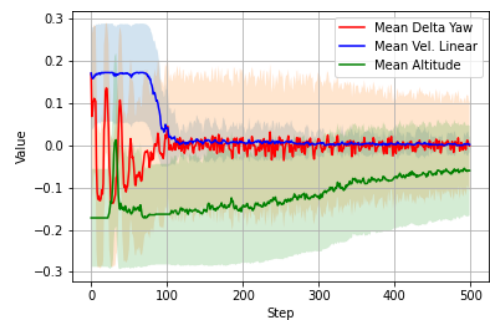


(l) BUG2 no segundo cenário da água para o ar visto de cima.

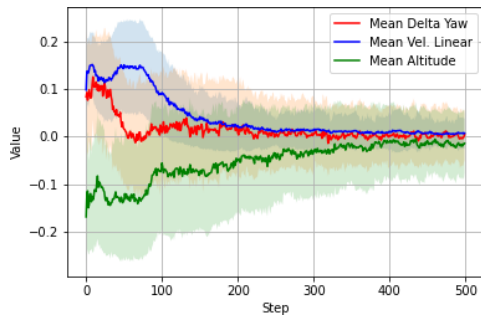
Figura 33: Caminho realizado durante 1 das 100 tentativas no contexto 3D híbrido no segundo cenário.



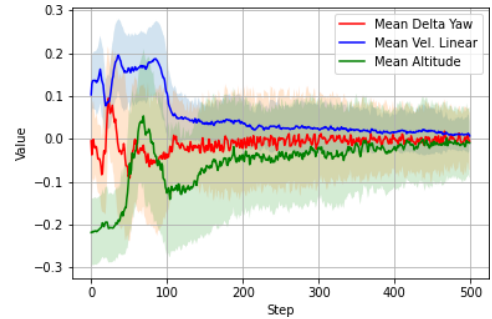
(a) NDRL-D no primeiro cenário.



(b) NDRL-D no segundo cenário.

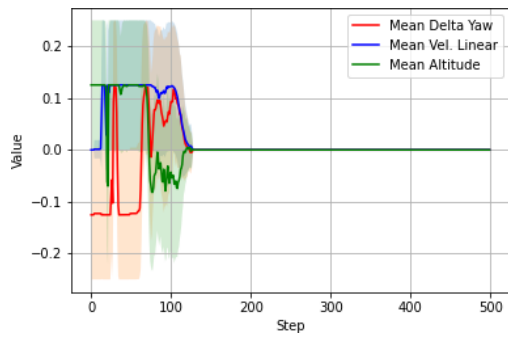


(c) NDRL-S no primeiro cenário.

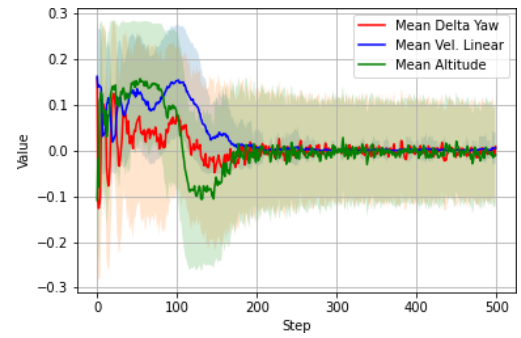


(d) NDRL-S no segundo cenário.

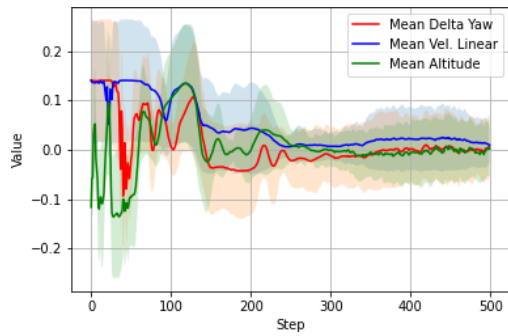
Figura 34: Ações dos agentes realizadas durante as 100 tentativas no contexto 3D híbrido partindo do ar em direção à água. Em vermelho a média do delta yawem azul da velocidade linear e em verde a velocidade de altitude. A sombra representa o desvio padrão.



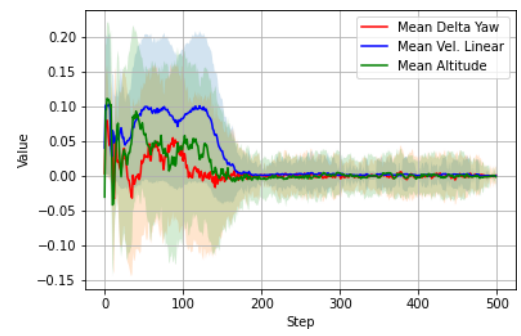
(a) NDRL-D no primeiro cenário.



(b) NDRL-D no segundo cenário.



(c) NDRL-S no primeiro cenário.



(d) NDRL-S no segundo cenário.

Figura 35: Ações dos agentes realizadas durante as 100 tentativas no contexto 3D híbrido partindo do meio subaquático em direção ao ar. Em vermelho a média do delta yaw, em azul da velocidade linear e em verde a velocidade de altitude. A sombra representa o desvio padrão.

5.5 Discussão

De modo geral, a extensiva validação das abordagens de Deep-RL desenvolvidas mostra um apropriado comportamento de ambos os agentes nos variados contextos e tarefas testados. Os agentes foram capazes de aprender as dificuldades e características de cada ambiente, realizando a navegação sem mapa em duas dimensões no ar e na água e em três dimensões com transição de meio entre os ambientes. Além disso, os agentes desenvolvidos foram capazes de aprender a detectar e desviar de obstáculos a fim de evitar colisões. Foram analisados duas abordagens baseadas em dois algoritmos poderosos do estado da arte - SAC e DDPG -, que foram capazes de entender o comportamento complexo dos ambientes - incluindo simulação de ventos - em simulação e que podem ser usados no mundo real em um veículo real se garantido o sensoriamento e dados da localização relativa do veículo de forma correta.

Especificamente, também é possível observar de modo geral que a abordagem baseada em DDPG apresentou um desempenho melhor do que a abordagem baseada em SAC no primeiro e mais simples cenário contendo a simulação de ventos e o tanque de água. Entretanto, no segundo cenário com os dutos de perfuração simulados, é possível observar de modo geral que a melhor política de exploração da abordagem baseada em SAC fez essa abordagem ter uma performance similar ou até melhor em alguns casos, como no contexto híbrido e mais importante. Por último, é interessante destacar que apesar do fato de que as duas abordagens possuem diferenças relevantes em suas respectivas implementações, as ações geradas por ambas a cada passo de teste quando quantizada a média de 100 tentativas são similares, principalmente no contexto híbrido e no segundo e mais complexo cenário.

6 ESTUDO DE VALIDAÇÃO

A fim de analisar com maior detalhes as abordagens de Deep-RL desenvolvidas neste trabalho, é proposto um estudo de validação de ambas as abordagens com variadas configurações de rede. A estrutura de rede proposta para ambos os contextos (15 e 16) foi desenvolvida inspirada em modelos de trabalhos relacionados que obtiveram bons resultados [Tai et al., 2017], [Jesus et al., 2019], [Rodriguez-Ramos et al., 2018]. Propõe-se uma comparação com outros modelos de MLP (do inglês *Multi Layer Perceptron*) com um número maior e menor de camadas escondidas. Além disso, também propõe-se uma comparação com um modelo de rede utilizando uma rede LSTM (do inglês *Long Short Term Memory*) e uma rede convolucional.

O estudo proposto foi realizado em ambos os contexto 2D e 3D para ambas as tarefas propostas no ambiente aéreo. Nas Figuras 36 são apresentadas as estruturas de rede dos modelos desenvolvidos para o estudo de validação deste trabalho para o contexto 3D. Para o contexto 2D, as mesmas estruturas de rede são propostas, alterando-se somente o número de entrada e saídas (vide Figura 15). Para o modelo convolucional, especificamente, foi utilizado um número maior de amostras do sensor de distância onde um total de 270 amostras foi utilizado, ao invés das 20 amostras para os demais modelos.

Apesar do foco do estudo ser nos tipos e estruturas de rede, um estudo em relação ao tipo de função de ativação também é proposto. Um modelo utilizando a função de ativação *Hardtanh* ao invés de *Tanh* na camada de saída é comparada a estrutura original com três camadas completamente conectadas. Nesse estudo em específico, busca-se analisar o impacto de utilizar uma função de ativação linear no intervalo de normalização ao invés da curva hiperbólica convencional. Os resultados obtidos para cada contexto e estrutura são apresentados na sequência.

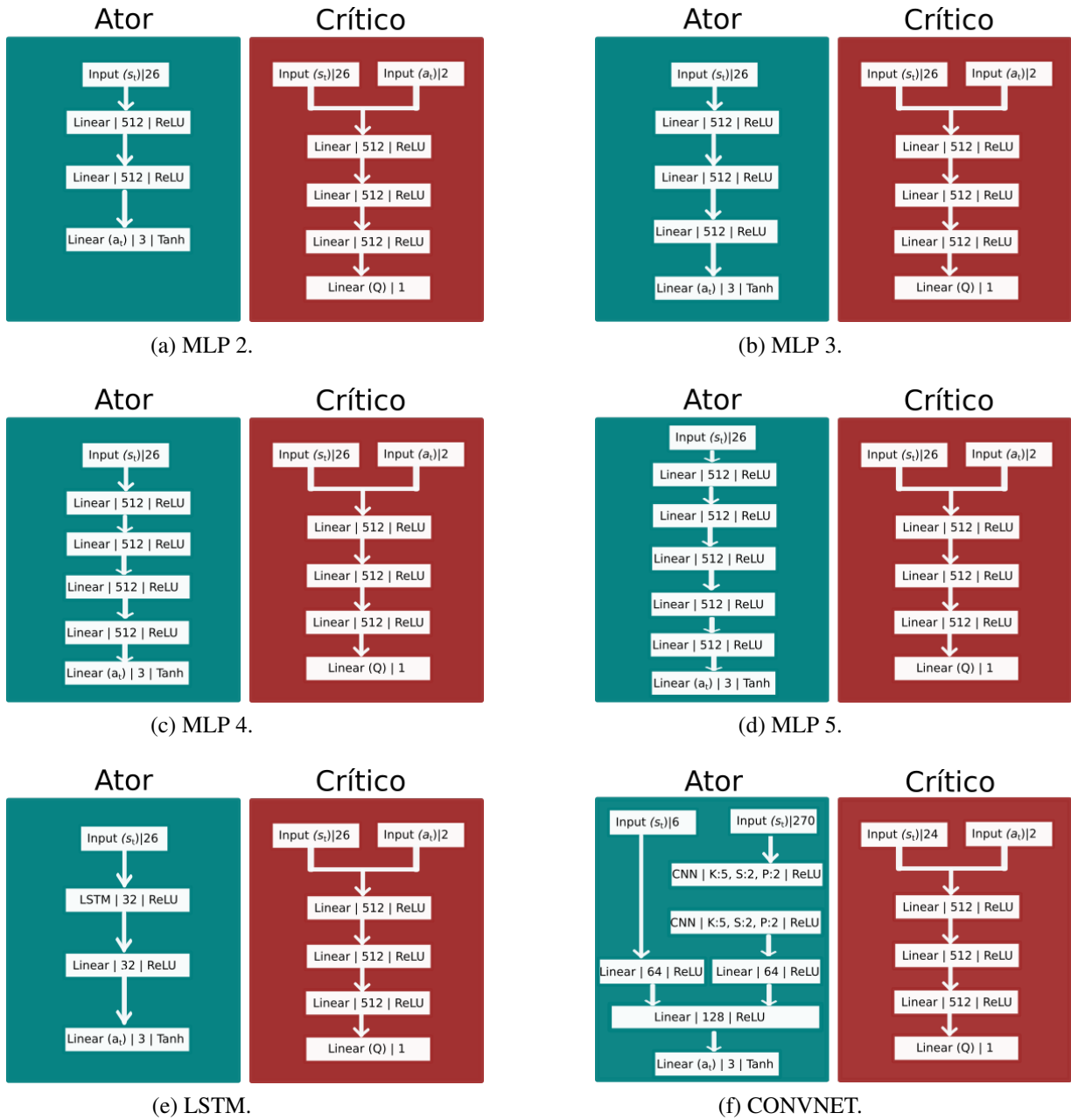
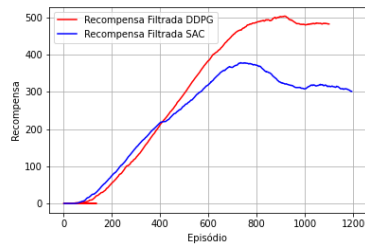


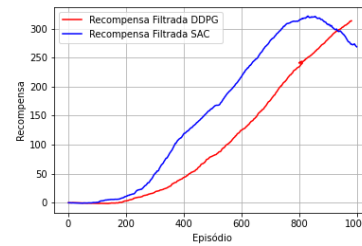
Figura 36: Recompensa ao longo dos episódios para cada abordagem no ambiente aéreo no contexto 3D.

6.1 Contexto 2D

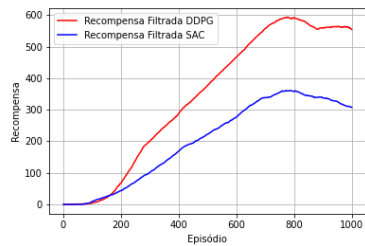
Nas Figuras 37 é possível observar o reforço médio obtido para cada abordagem para o primeiro cenário, enquanto que na Figura 38 o reforço médio no segundo cenário é mostrado.



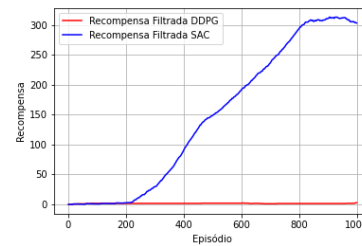
(a) MLP com duas camadas escondidas.



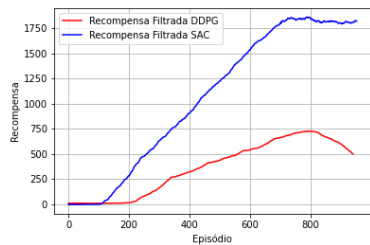
(b) MLP com três camadas escondidas.



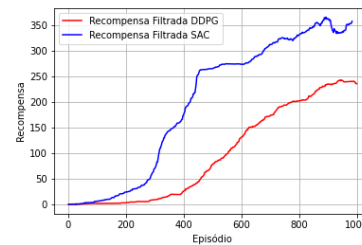
(c) MLP com quatro camadas escondidas.



(d) MLP com cinco camadas escondidas.

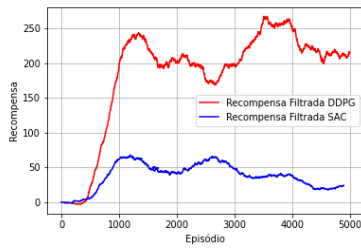


(e) Rede LSTM.

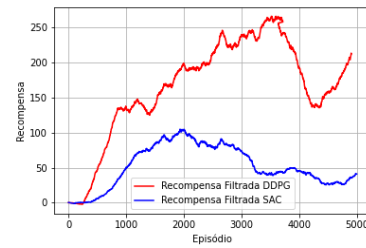


(f) Rede convolucional.

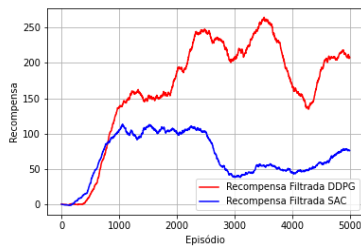
Figura 37: Recompensa ao longo dos episódios para cada estrutura no contexto 2D no primeiro cenário.



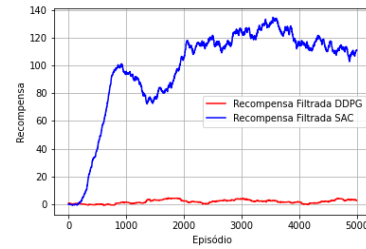
(a) MLP com duas camadas escondidas.



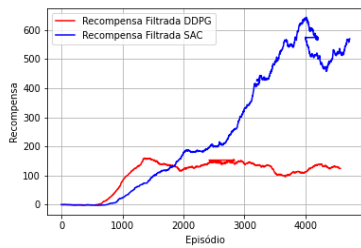
(b) MLP com três camadas escondidas.



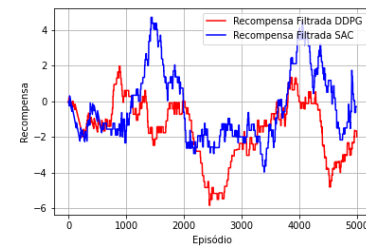
(c) MLP com quatro camadas escondidas.



(d) MLP com cinco camadas escondidas.



(e) Rede LSTM.



(f) Rede convolucional.

Figura 38: Recompensa ao longo dos episódios para cada estrutura no contexto 2D no segundo cenário.

6.1.1 Navegação sem mapa orientada a alvo

Nas Tabelas 20 e 21 pode-se observar o total de tentativas em que a navegação foi realizada com sucesso para cada abordagem em cada cenário para a primeira tarefa. Além disso, também é mostrado o tempo médio e o desvio padrão de realização da tarefa.

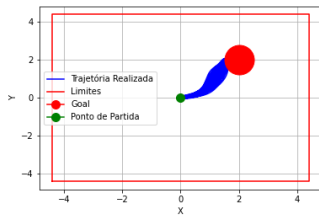
Tabela 20: Estatísticas da navegação orientada a alvo para o contexto 2D da abordagem baseada em DDPG na primeira tarefa.

Cenário	Abordagem	Total Sucesso	Tempo Médio (s)
1	NDRL-D 2	100%	18.54 ± 1.21
1	NDRL-D 3	100%	13.48 ± 0.46
1	NDRL-D 4	100%	13.442 ± 0.96
1	NDRL-D 5	0%	102.812 ± 10.849
1	NDRL-D LSTM	95%	58.657 ± 65.295
1	NDRL-D CONV	84%	48.329 ± 48.522
2	NDRL-D 2	65%	30.711 ± 11.343
2	NDRL-D 3	35%	28.13 ± 11.06
2	NDRL-D 4	39%	29.9 ± 14.62
2	NDRL-D 5	0%	92.547 ± 17.86
2	NDRL-D LSTM	21%	44.06 ± 26.49
2	NDRL-D CONV	0%	98.77 ± 82.11

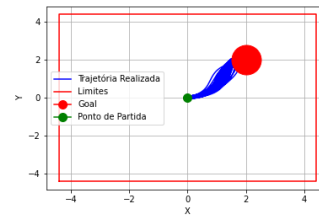
Tabela 21: Estatísticas da navegação orientada a alvo para o contexto 2D da abordagem baseada em SAC na primeira tarefa.

Cenário	Abordagem	Total Sucesso	Tempo Médio (s)
1	NDRL-S 2	100%	18.38 ± 1.05
1	NDRL-S 3	93%	32.2 ± 14.52
1	NDRL-S 4	100%	16.16 ± 1.63
1	NDRL-S 5	97%	24.87 ± 14.896
1	NDRL-S LSTM	100%	19.22 ± 4.37
1	NDRL-S CONV	100%	15.233 ± 1.023
2	NDRL-S 2	13%	39.99 ± 26.32
2	NDRL-S 3	74%	35.617 ± 12.79
2	NDRL-S 4	60%	33.978 ± 11.84
2	NDRL-S 5	32%	25.011 ± 12.29
2	NDRL-S LSTM	35%	39.87 ± 20.71
2	NDRL-S CONV	0%	34.01 ± 14.67

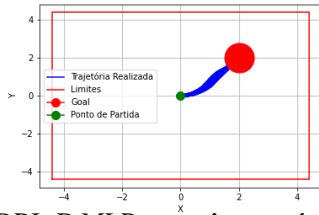
Nas Figuras 39 e 40 é possível observar as 100 tentativas de navegação realizada por ambos agentes no primeiro e segundo cenários, respectivamente.



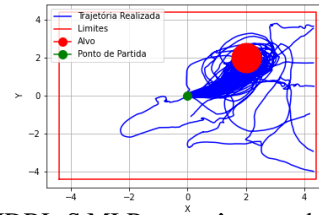
(a) NDRL-D MLP com duas camadas escondidas.



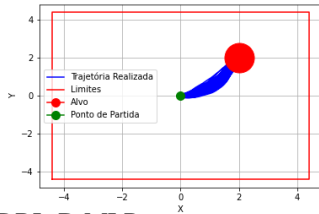
(b) NDRL-S MLP com duas camadas escondidas.



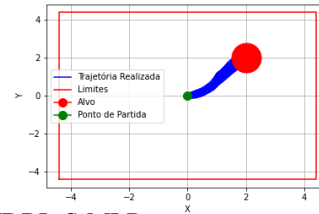
(c) NDRL-D MLP com três camadas escondidas.



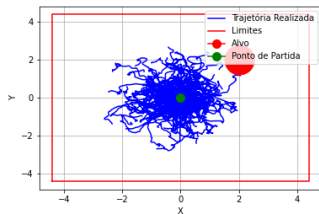
(d) NDRL-S MLP com três camadas escondidas.



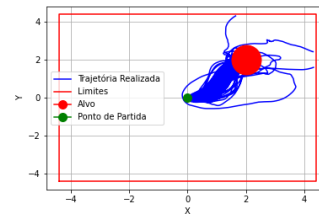
(e) NDRL-D MLP com quatro camadas escondidas.



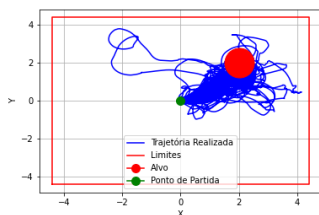
(f) NDRL-S MLP com quatro camadas escondidas.



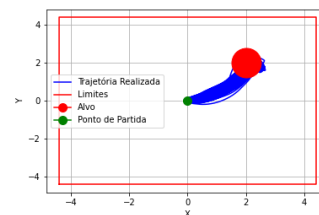
(g) NDRL-D MLP com cinco camadas escondidas.



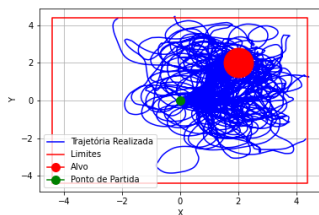
(h) NDRL-S MLP com cinco camadas escondidas.



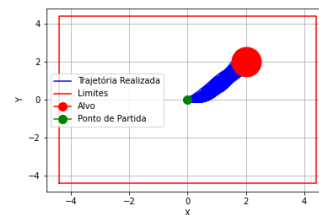
(i) NDRL-D LSTM.



(j) NDRL-S LSTM.

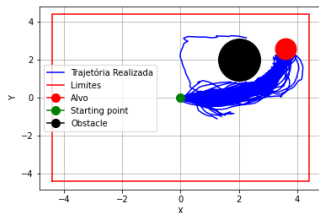


(k) NDRL-D Convolucionar.

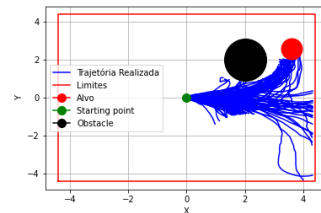


(l) NDRL-S Convolucionar.

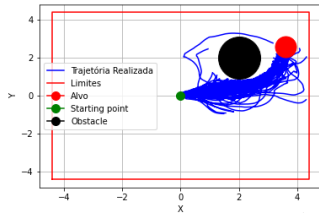
Figura 39: Caminho realizado durante as 100 tentativas na primeira tarefa no contexto 2D para o primeiro cenário.



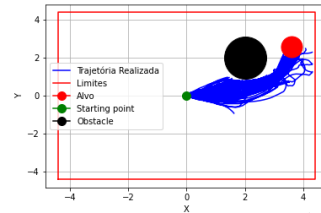
(a) NDRL-D MLP com duas camadas escondidas.



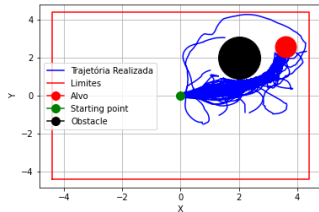
(b) NDRL-S MLP com duas camadas escondidas.



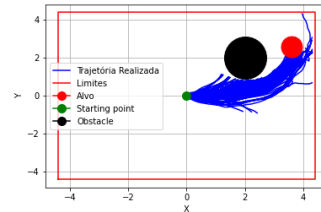
(c) NDRL-D MLP com três camadas escondidas.



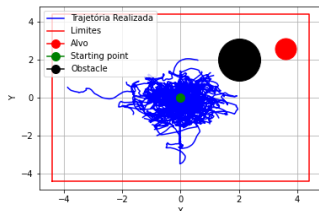
(d) NDRL-S MLP com três camadas escondidas.



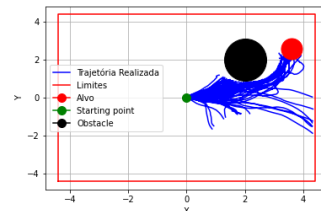
(e) NDRL-D MLP com quatro camadas escondidas.



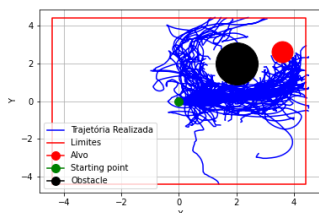
(f) NDRL-S MLP com quatro camadas escondidas.



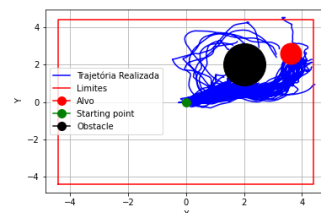
(g) NDRL-D MLP com cinco camadas escondidas.



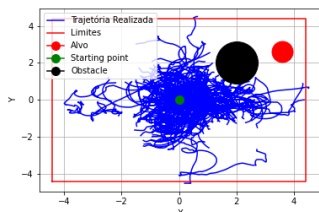
(h) NDRL-S MLP com cinco camadas escondidas.



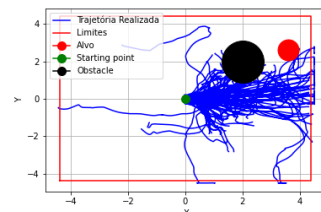
(i) NDRL-D LSTM.



(j) NDRL-S LSTM.



(k) DDPG convolucional.

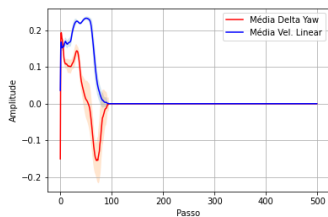


(l) NDRL-S convolucional.

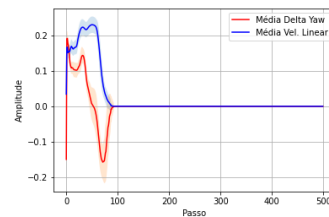
Figura 40: Caminho realizado durante as 100 tentativas na primeira tarefa no contexto 2D para o segundo cenário.

Nas Figuras 41 e 42 é possível observar a média de cada uma das ações de cada um dos agentes em cada passo durante a realização da tarefa para cada um dos respectivos cenários.

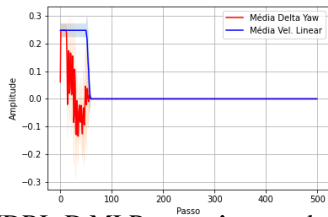
Para a primeira tarefa destaca-se o bom desempenho que ambos os agentes tiveram. Em especial destaca-se as abordagens convolucional e LSTM que obtiveram bons resultados no primeiro cenário. Das abordagens de MLP, destaca-se a maior capacidade de aprendizado da abordagem estocástica na variação do tamanho da rede em comparação com a abordagem determinística. É importante ressaltar que os melhores resultados foram obtidos com a estrutura de rede com três camadas escondidas, que foi a estrutura escolhida neste trabalho. É importante ressaltar também que apesar das abordagens convolucionais e LSTM terem apresentado bom desempenho, elas precisam ser melhores exploradas a fim de obter melhores resultados.



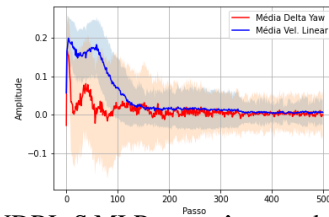
(a) NDRL-D MLP com duas camadas escondidas.



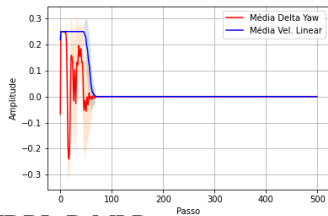
(b) NDRL-S MLP com duas camadas escondidas.



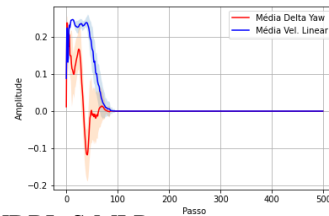
(c) NDRL-D MLP com três camadas escondidas.



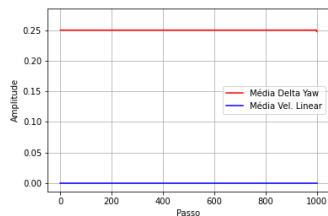
(d) NDRL-S MLP com três camadas escondidas.



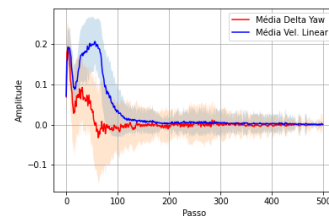
(e) NDRL-D MLP com quatro camadas escondidas.



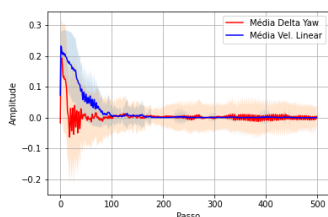
(f) NDRL-S MLP com quatro camadas escondidas.



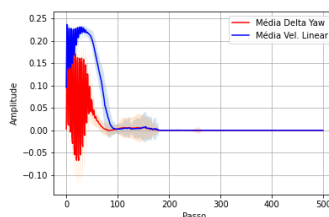
(g) NDRL-D MLP com cinco camadas escondidas.



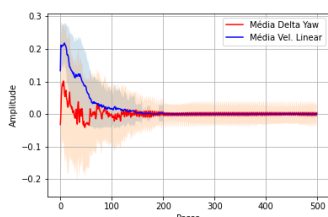
(h) NDRL-S MLP com cinco camadas escondidas.



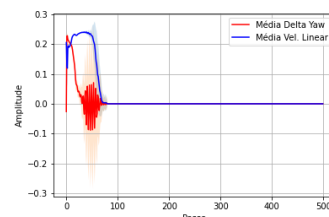
(i) NDRL-D LSTM.



(j) NDRL-S LSTM.

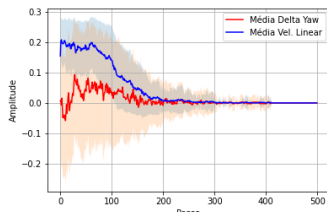


(k) NDRL-D convolucional.

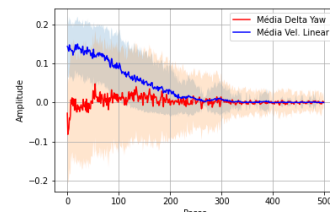


(l) NDRL-S convolucional.

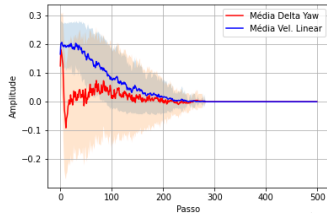
Figura 41: Ações dos agentes realizadas durante as 100 tentativas para a primeira tarefa no contexto 2D no primeiro cenário. Em vermelho a média do delta yaw e em azul da velocidade linear. A sombra representa o desvio padrão.



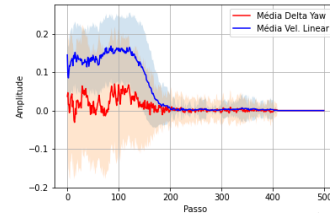
(a) NDRL-D MLP com duas camadas escondidas.



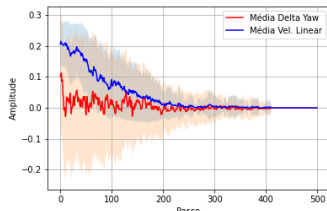
(b) NDRL-S MLP com duas camadas escondidas.



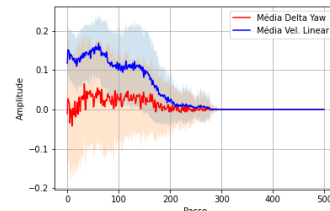
(c) NDRL-D MLP com três camadas escondidas.



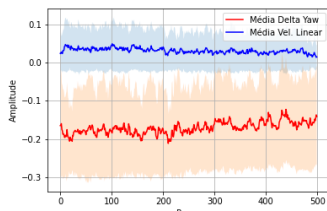
(d) NDRL-S MLP com três camadas escondidas.



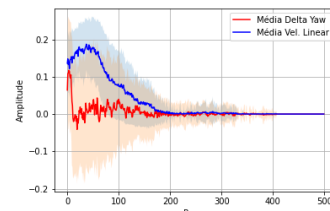
(e) NDRL-D MLP com quatro camadas escondidas.



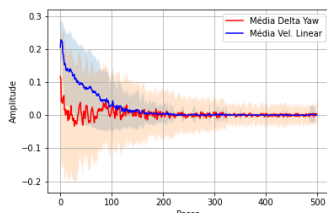
(f) NDRL-S MLP com quatro camadas escondidas.



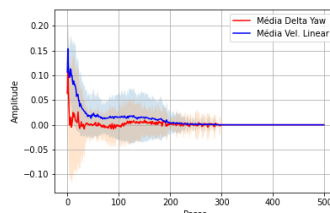
(g) NDRL-D MLP com cinco camadas escondidas.



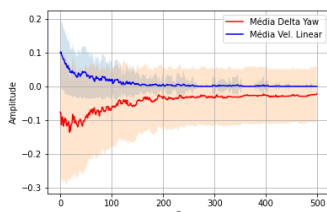
(h) NDRL-S MLP com cinco camadas escondidas.



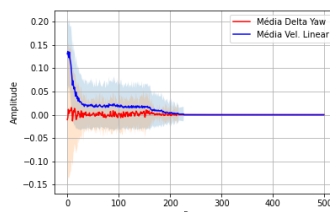
(i) NDRL-D LSTM.



(j) NDRL-S LSTM.



(k) NDRL-D convolucional.



(l) NDRL-S convolucional.

Figura 42: Ações dos agentes realizadas durante as 100 tentativas para a primeira tarefa no contexto 2D no segundo cenário. Em vermelho a média do delta yaw e em azul da velocidade linear. A sombra representa o desvio padrão.

6.1.2 Navegação sem mapa orientada a múltiplos pontos

Nas Tabelas 22 e 23 pode-se observar o total de tentativas em que a navegação foi realizada com sucesso para cada abordagem em cada cenário para a segunda tarefa.

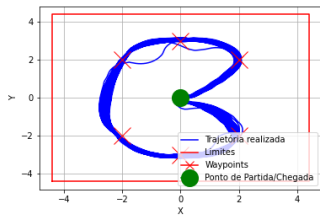
Tabela 22: Estatísticas da navegação orientada a alvo para o contexto 2D da abordagem baseado em DDPG na segunda tarefa.

Cenário	Abordagem	Total Sucesso	Tempo Médio (s)	Percentual Percorrido
1	NDRL-D 2	100%	97.22 ± 1.79	100%
1	NDRL-D 3	100%	99.37 ± 3.89	100%
1	NDRL-D 4	95%	135.84 ± 29.146	97.376%
1	NDRL-D 5	0%	203.168 ± 1.849	0%
1	NDRL-D LSTM	84%	248.04 ± 167.75	87.75%
1	NDRL-D CONV	39%	382.637 ± 211.715	67.89%
2	NDRL-D 2	1%	118.278 ± 75.77	46.734%
2	NDRL-D 3	7%	55.43 ± 44.319	20.816%
2	NDRL-D 4	0%	45.09 ± 35.45	10%
2	NDRL-D 5	0%	97.54 ± 16.79	0%
2	NDRL-D LSTM	0%	75.06 ± 53.6	7.142%
2	NDRL-D CONV	0%	127.50 ± 74.74	0.61%

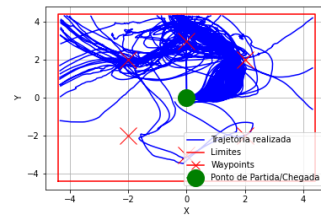
Tabela 23: Estatísticas da navegação orientada a alvo para o contexto 2D da abordagem baseado em SAC na segunda tarefa.

Cenário	Abordagem	Total Sucesso	Tempo Médio (s)	Percentual Percorrido
1	NDRL-S 2	2%	131.07 ± 88.29	32.21%
1	NDRL-S 3	54%	164.18 ± 43.89	73.76%
1	NDRL-S 4	68%	155.54 ± 40.36	83.09%
1	NDRL-S 5	48%	152.418 ± 50.384	74.63%
1	NDRL-S LSTM	100%	118.55 ± 11.16	100%
1	NDRL-S CONV	0%	209.74 ± 2.051	50.43%
2	NDRL-S 2	0%	53.379 ± 39.343	11.224%
2	NDRL-S 3	0%	36.611 ± 16.65	12.65%
2	NDRL-S 4	0%	54.508 ± 42.59	14.285%
2	NDRL-S 5	9%	73.56 ± 59.489	28.775%
2	NDRL-S LSTM	1%	62.92 ± 46.63	12.85%
2	NDRL-S CONV	0%	35.6 ± 15.30	0.204%

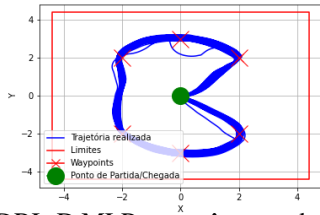
Nas Figuras 43 e 44 é possível observar as 100 tentativas de navegação realizada por ambos agentes no primeiro e segundo cenários, respectivamente.



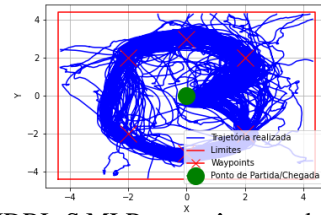
(a) NDRL-D MLP com duas camadas escondidas.



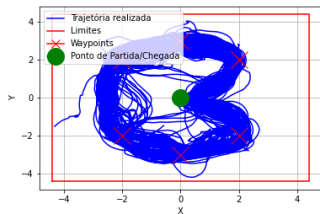
(b) NDRL-S MLP com duas camadas escondidas.



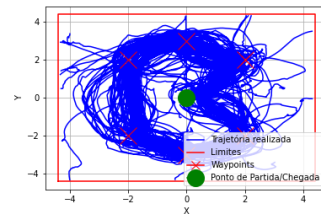
(c) NDRL-D MLP com três camadas escondidas.



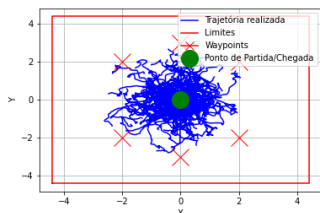
(d) NDRL-S MLP com três camadas escondidas.



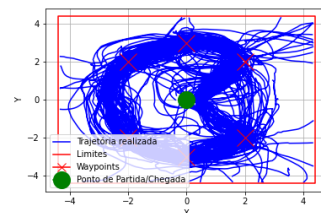
(e) NDRL-D MLP com quatro camadas escondidas.



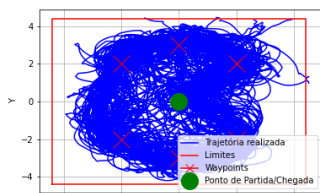
(f) NDRL-S MLP com quatro camadas escondidas.



(g) NDRL-D MLP com cinco camadas escondidas.



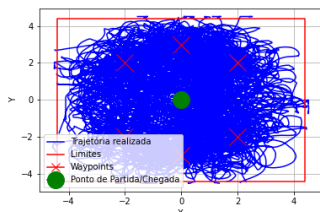
(h) NDRL-S MLP com cinco camadas escondidas.



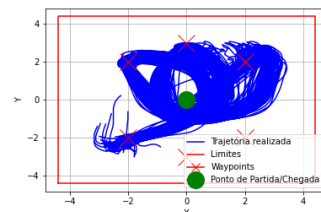
(i) NDRL-D LSTM.



(j) NDRL-S LSTM.

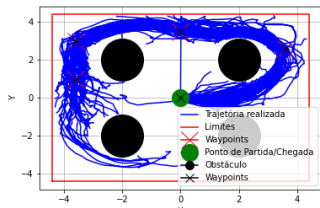


(k) NDRL-D Convolucionar.

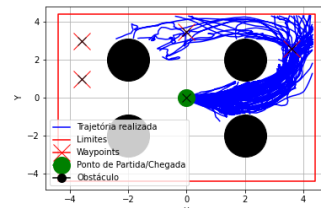


(l) NDRL-S Convolucionar.

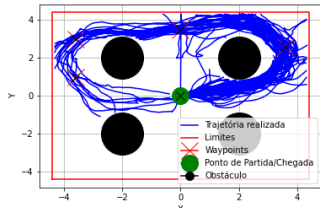
Figura 43: Caminho realizado durante as 100 tentativas na segunda tarefa no contexto 2D para o primeiro cenário.



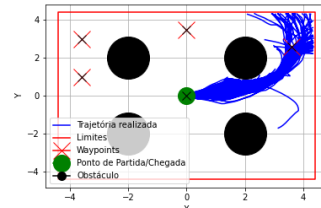
(a) NDRL-D MLP com duas camadas escondidas.



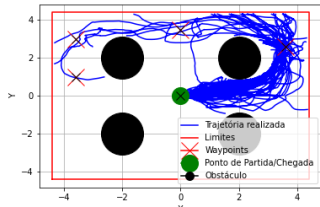
(b) NDRL-S MLP com duas camadas escondidas.



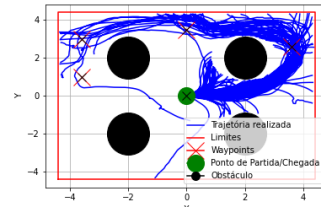
(c) NDRL-D MLP com três camadas escondidas.



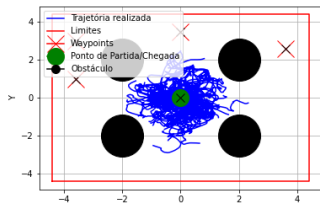
(d) NDRL-S MLP com três camadas escondidas.



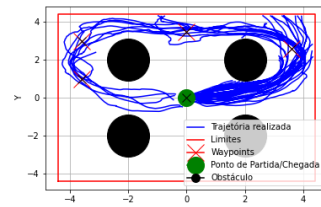
(e) NDRL-D MLP com quatro camadas escondidas.



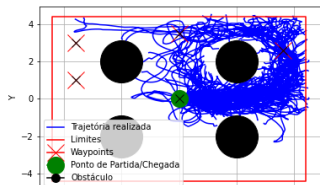
(f) NDRL-S MLP com quatro camadas escondidas.



(g) NDRL-D MLP com cinco camadas escondidas.



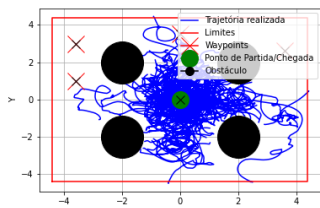
(h) NDRL-S MLP com cinco camadas escondidas.



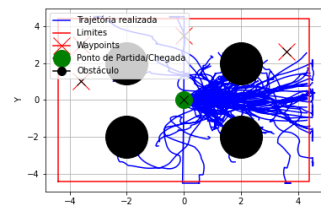
(i) NDRL-D LSTM.



(j) NDRL-S LSTM.



(k) NDRL-D convolucional.

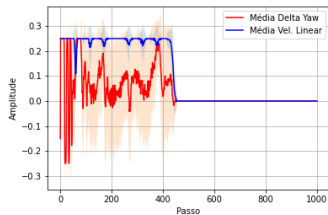


(l) NDRL-S convolucional.

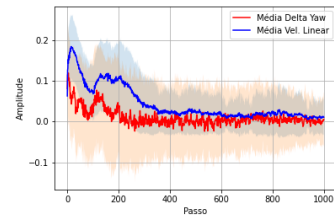
Figura 44: Caminho realizado durante as 100 tentativas na segunda tarefa no contexto 2D para o segundo cenário.

Nas Figuras 45 e 46 é possível observar a média de cada uma das ações de cada um dos agentes em cada passo durante a realização da tarefa para cada um dos respectivos cenários.

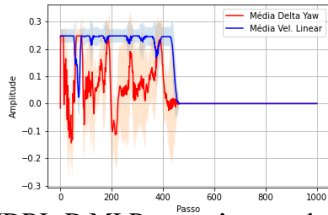
Na segunda tarefa destaca-se o percentual médio percorrido. Para a abordagem determinística esse percentual tende a diminuir com o aumento do número de camadas escondidas, enquanto que o oposto ocorreu com a abordagem estocástica. Destaca-se também o bom resultado obtido pelas abordagens com LSTM, que apesar da simplicidade (apenas 32 células) obtiveram resultados até melhores que as redes MLP. Em especial para a abordagem estocástica, que obteve o melhor resultado entre todas.



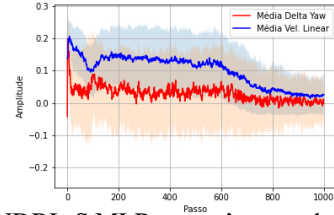
(a) NDRL-D MLP com duas camadas escondidas.



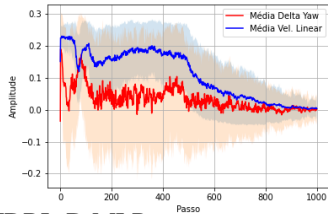
(b) NDRL-S MLP com duas camadas escondidas.



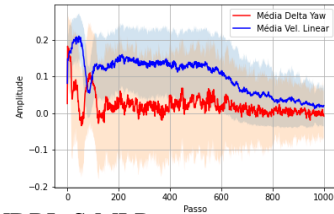
(c) NDRL-D MLP com três camadas escondidas.



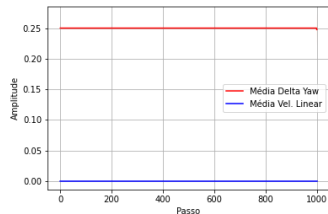
(d) NDRL-S MLP com três camadas escondidas.



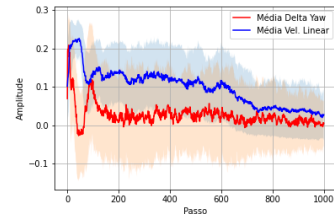
(e) NDRL-D MLP com quatro camadas escondidas.



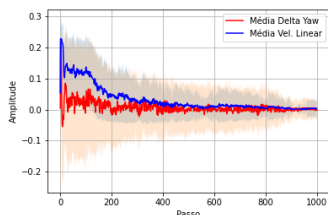
(f) NDRL-S MLP com quatro camadas escondidas.



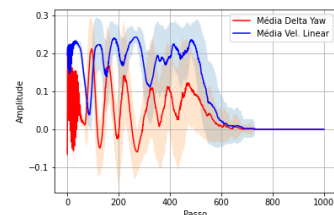
(g) NDRL-D MLP com cinco camadas escondidas.



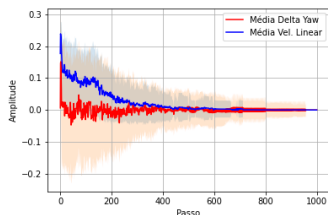
(h) NDRL-S MLP com cinco camadas escondidas.



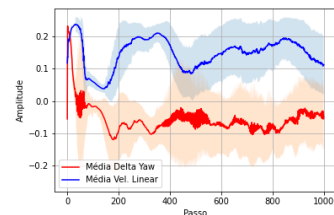
(i) NDRL-D LSTM.



(j) NDRL-S LSTM.

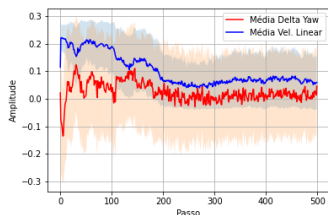


(k) NDRL-D convolucional.

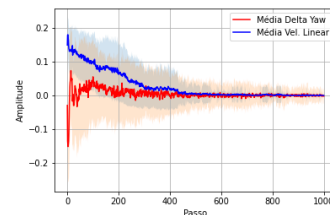


(l) NDRL-S convolucional.

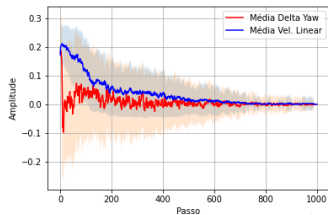
Figura 45: Ações dos agentes realizadas durante as 100 tentativas para a segunda tarefa no contexto 2D no primeiro cenário. Em vermelho a média do delta yaw e em azul da velocidade linear. A sombra representa o desvio padrão.



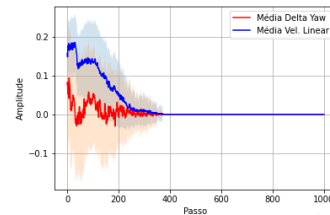
(a) NDRL-D MLP com duas camadas escondidas.



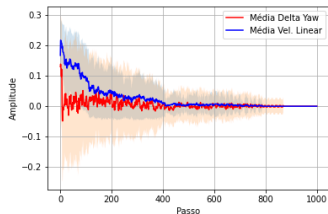
(b) NDRL-S MLP com duas camadas escondidas.



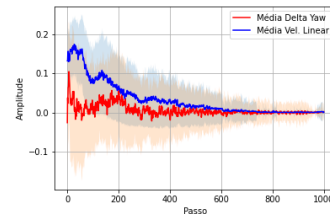
(c) NDRL-D MLP com três camadas escondidas.



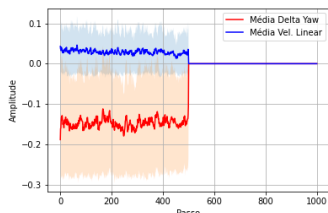
(d) NDRL-S MLP com três camadas escondidas.



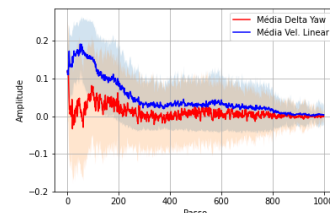
(e) NDRL-D MLP com quatro camadas escondidas.



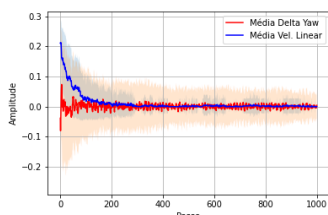
(f) NDRL-S MLP com quatro camadas escondidas.



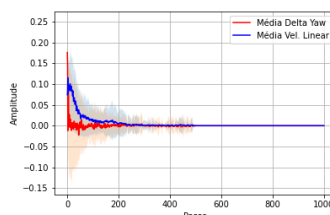
(g) NDRL-D MLP com cinco camadas escondidas.



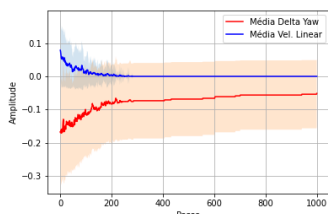
(h) NDRL-S MLP com cinco camadas escondidas.



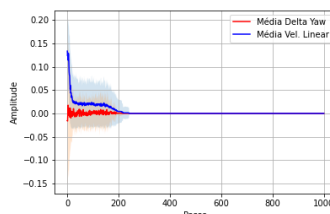
(i) NDRL-D LSTM.



(j) NDRL-S LSTM.



(k) NDRL-D convolucional.

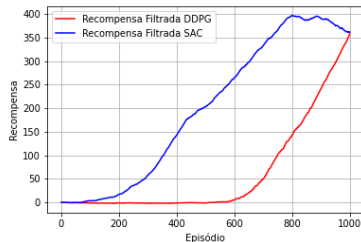


(l) NDRL-S convolucional.

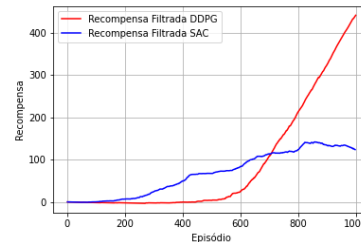
Figura 46: Ações dos agentes realizadas durante as 100 tentativas para a segunda tarefa no contexto 2D no segundo cenário. Em vermelho a média do delta yaw, em azul a velocidade linear e em verde a velocidade de altitude. A sombra representa o desvio padrão.

6.2 Contexto 3D

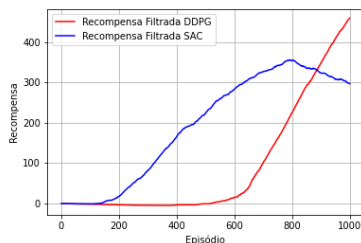
Nas Figuras 47 é possível observar o reforço médio obtido para cada abordagem para o primeiro cenário, enquanto que na Figura 48 o reforço médio no segundo cenário é mostrado.



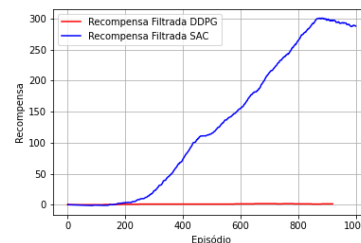
(a) MLP com duas camadas escondidas.



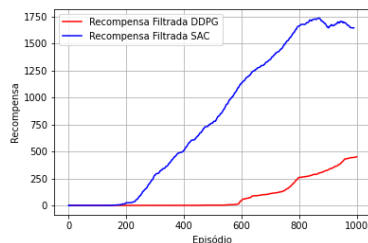
(b) MLP com três camadas escondidas.



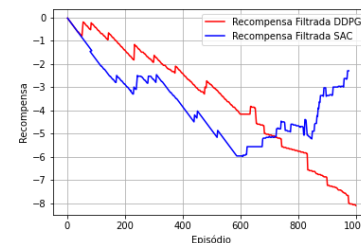
(c) MLP com quatro camadas escondidas.



(d) MLP com cinco camadas escondidas.

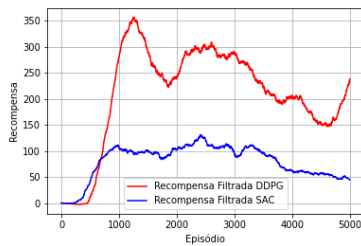


(e) Rede LSTM.

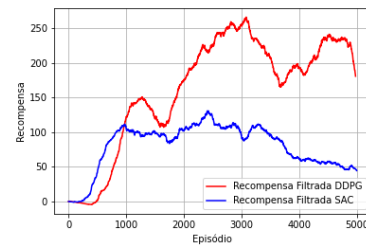


(f) Rede convolucional.

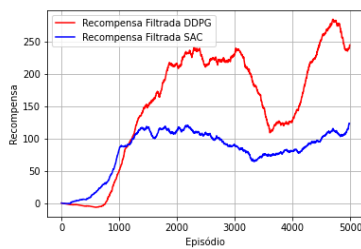
Figura 47: Recompensa ao longo dos episódios para cada estrutura no contexto 3D no primeiro cenário.



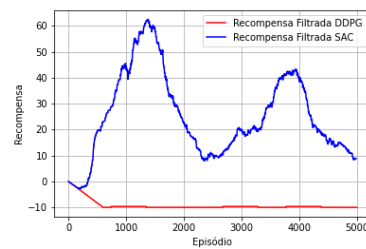
(a) MLP com duas camadas escondidas.



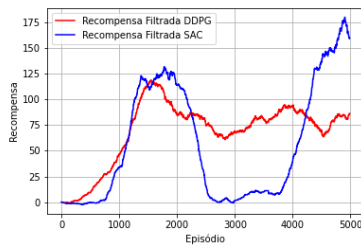
(b) MLP com três camadas escondidas.



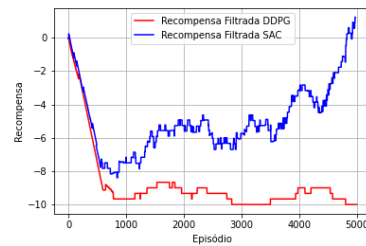
(c) MLP com quatro camadas escondidas.



(d) MLP com cinco camadas escondidas.



(e) Rede LSTM.



(f) Rede convolucional.

Figura 48: Recompensa ao longo dos episódios para cada estrutura no contexto 3D no segundo cenário.

6.2.1 Navegação sem mapa orientada a alvo

Nas Tabelas 24 e 25 pode-se observar o total de tentativas em que a navegação foi realizada com sucesso para cada abordagem em cada cenário para a primeira tarefa. Além disso, também é mostrado o tempo médio e o desvio padrão de realização da tarefa.

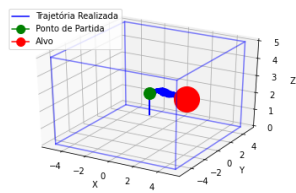
Tabela 24: Estatísticas da navegação orientada a alvo para o contexto 3D da abordagem baseada em DDPG na primeira tarefa.

Cenário	Abordagem	Total Sucesso	Tempo Médio (s)
1	NDRL-D 2	100%	13.54 ± 1.00
1	NDRL-D 3	100%	13.697 ± 0.85
1	NDRL-D 4	100%	13.735 ± -0.941
1	NDRL-D 5	0%	20.21 ± 13.93
1	NDRL-D LSTM	100%	13.81 ± 1.22
1	NDRL-D CONV	0%	26.66 ± 17.83
2	NDRL-D 2	8%	23.187 ± 10.30
2	NDRL-D 3	17%	49.276 ± 38.32
2	NDRL-D 4	24%	29.76 ± 15.85
2	NDRL-D 5	0%	17.93 ± 5.727
2	NDRL-D LSTM	61%	31.26 ± 16.43
2	NDRL-D CONV	0%	18.56 ± 10.57

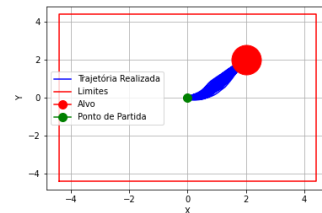
Tabela 25: Estatísticas da navegação orientada a alvo para o contexto 3D da abordagem baseada em SAC na primeira tarefa.

Cenário	Abordagem	Total Sucesso	Tempo Médio (s)
1	NDRL-S 2	100%	25.89 ± 10.39
1	NDRL-S 3	100%	27.038 ± 18.673
1	NDRL-S 4	91%	26.008 ± 15.20
1	NDRL-S 5	100%	16.706 ± 2.674
1	NDRL-S LSTM	100%	56.48 ± 25.71
1	NDRL-S CONV	0%	68.75 ± 51.81
2	NDRL-S 2	5%	28.218 ± 16.553
2	NDRL-S 3	19%	39.43 ± 22.21
2	NDRL-S 4	29%	50.373 ± 40.765
2	NDRL-S 5	8%	51.58 ± 23.25
2	NDRL-S LSTM	5%	37.44 ± 24.72
2	NDRL-S CONV	0%	32.78 ± 17.62

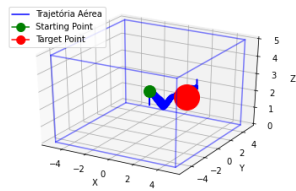
Nas Figuras 49, 50, 51 e 52 é possível observar as 100 tentativas de navegação realizada por ambos agentes no primeiro e segundo cenários e para cada abordagem, respectivamente.



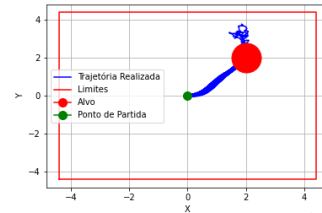
(a) NDRL-D MLP com duas camadas escondidas.



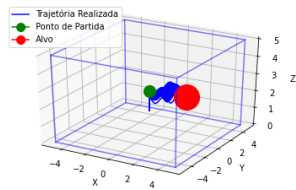
(b) NDRL-D MLP com duas camadas escondidas visto de cima.



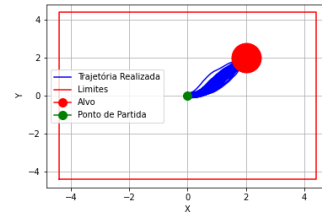
(c) NDRL-D MLP com três camadas escondidas.



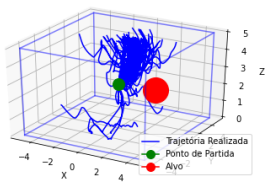
(d) NDRL-D MLP com três camadas escondidas visto de cima.



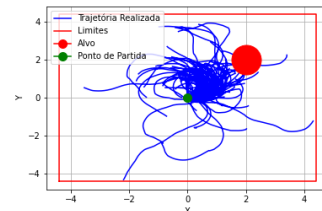
(e) NDRL-D MLP com quatro camadas escondidas.



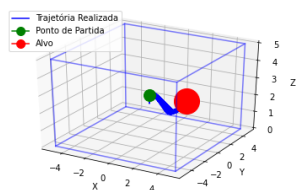
(f) NDRL-D MLP com quatro camadas escondidas visto de cima.



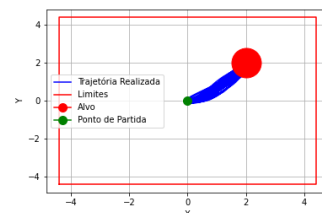
(g) NDRL-D MLP com cinco camadas escondidas.



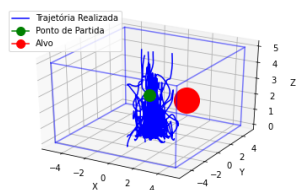
(h) NDRL-D MLP com cinco camadas escondidas visto de cima.



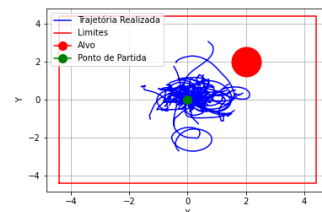
(i) NDRL-D LSTM.



(j) NDRL-D LSTM visto de cima.

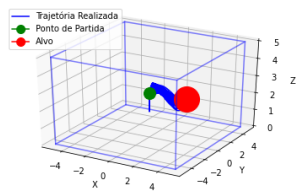


(k) NDRL-D convolucional.

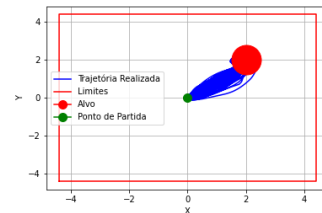


(l) NDRL-D convolucional visto de cima.

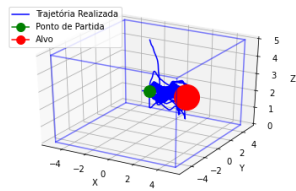
Figura 49: Caminho realizado durante as 100 tentativas na primeira tarefa no contexto 2D para o primeiro cenário para a abordagem DDPG.



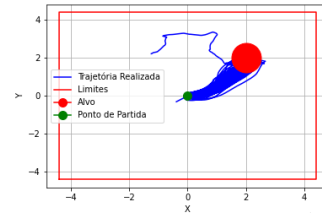
(a) NDRL-S MLP com duas camadas escondidas.



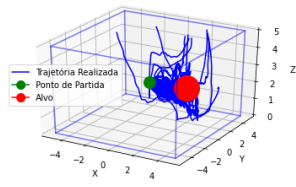
(b) NDRL-S MLP com duas camadas escondidas visto de cima.



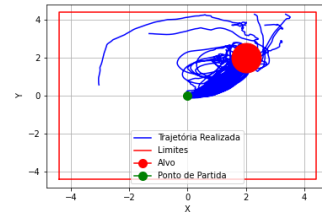
(c) NDRL-S MLP com três camadas escondidas.



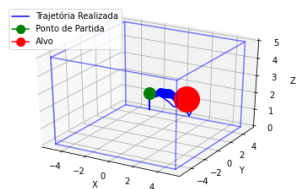
(d) NDRL-S MLP com três camadas escondidas visto de cima.



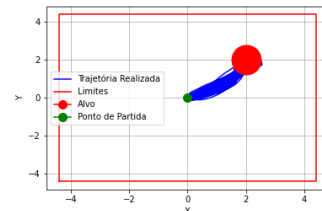
(e) NDRL-S MLP com quatro camadas escondidas.



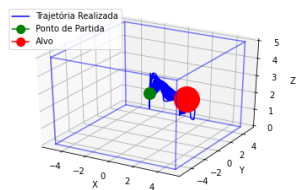
(f) NDRL-S MLP com quatro camadas escondidas visto de cima.



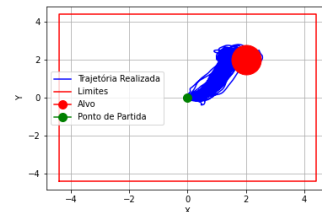
(g) NDRL-S MLP com cinco camadas escondidas.



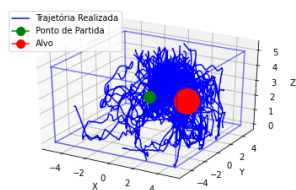
(h) NDRL-S MLP com cinco camadas escondidas visto de cima.



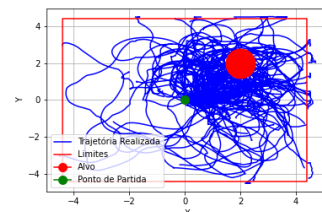
(i) NDRL-S LSTM.



(j) NDRL-S LSTM visto de cima.

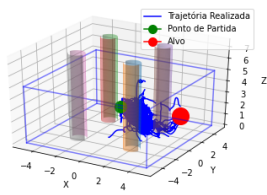


(k) NDRL-S convolucional.

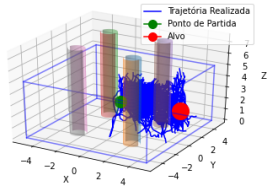


(l) NDRL-S convolucional visto de cima.

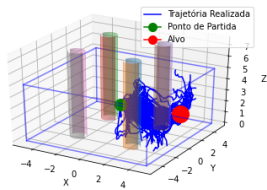
Figura 50: Caminho realizado durante as 100 tentativas na primeira tarefa no contexto 2D para o primeiro cenário para a abordagem SAC.



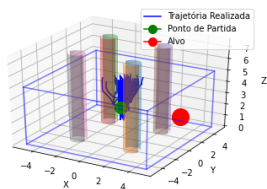
(a) NDRL-D MLP com duas camadas escondidas.



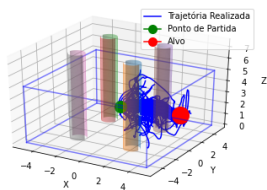
(c) NDRL-D MLP com três camadas escondidas.



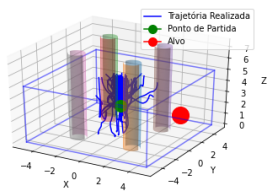
(e) NDRL-D MLP com quatro camadas escondidas.



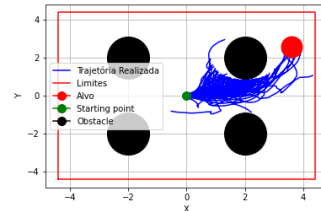
(g) NDRL-D MLP com cinco camadas escondidas.



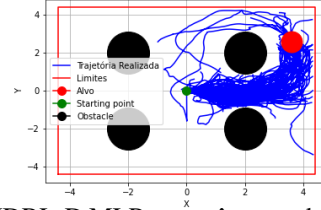
(i) NDRL-D LSTM.



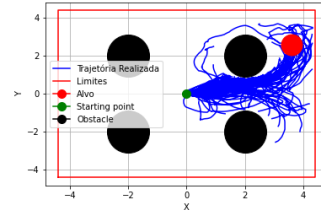
(k) NDRL-D convolucional.



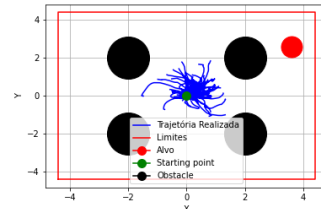
(b) NDRL-D MLP com duas camadas escondidas visto de cima.



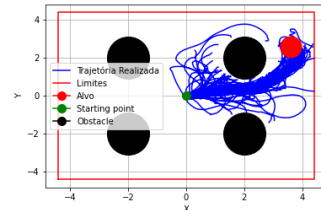
(d) NDRL-D MLP com três camadas escondidas visto de cima.



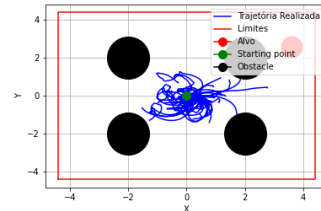
(f) NDRL-D MLP com quatro camadas escondidas visto de cima.



(h) NDRL-D MLP com cinco camadas escondidas visto de cima.

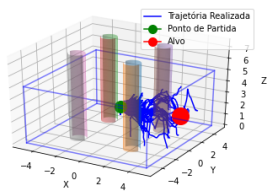


(j) NDRL-D LSTM visto de cima.

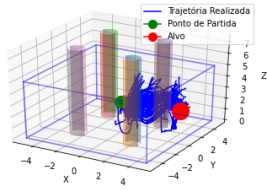


(l) NDRL-D convolucional visto de cima.

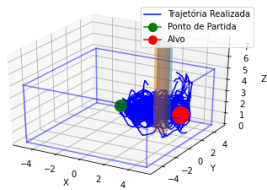
Figura 51: Caminho realizado durante as 100 tentativas na primeira tarefa no contexto 2D para o segundo cenário para a abordagem DDPG.



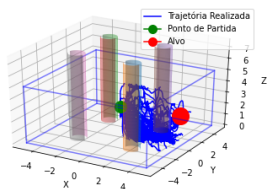
(a) NDRL-S MLP com duas camadas escondidas.



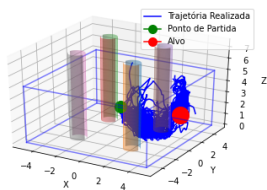
(c) NDRL-S MLP com três camadas escondidas.



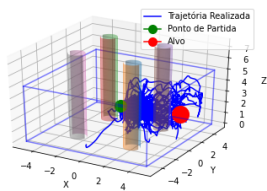
(e) NDRL-S MLP com quatro camadas escondidas.



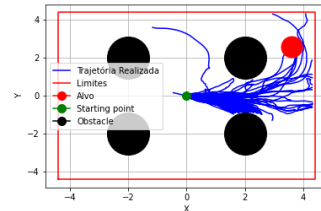
(g) NDRL-S MLP com cinco camadas escondidas.



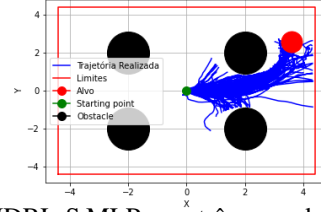
(i) NDRL-S LSTM.



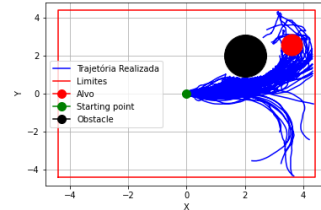
(k) NDRL-S convolucional.



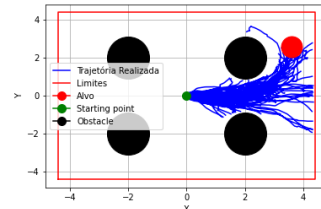
(b) NDRL-S MLP com duas camadas escondidas visto de cima.



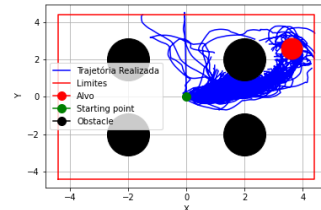
(d) NDRL-S MLP com três camadas escondidas visto de cima.



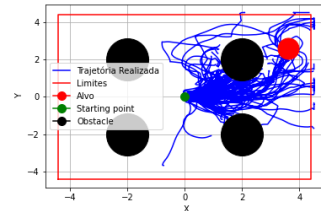
(f) NDRL-S MLP com quatro camadas escondidas visto de cima.



(h) NDRL-S MLP com cinco camadas escondidas visto de cima.



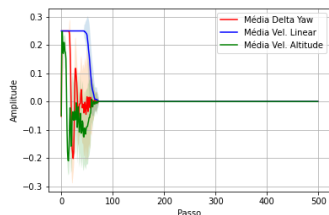
(j) NDRL-S LSTM visto de cima.



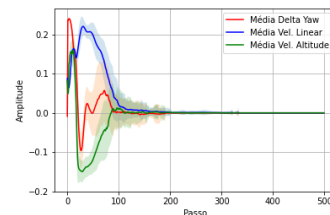
(l) NDRL-S convolucional visto de cima.

Figura 52: Caminho realizado durante as 100 tentativas na primeira tarefa no contexto 2D para o segundo cenário para a abordagem SAC.

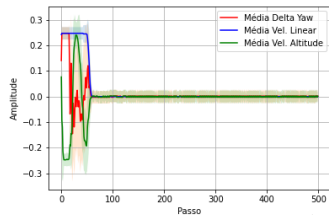
Nas Figuras 53 e 54 é possível observar a média de cada uma das ações de cada um dos agentes em cada passo durante a realização da tarefa para cada um dos respectivos cenários.



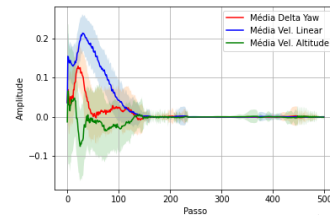
(a) NDRL-D MLP com duas camadas escondidas.



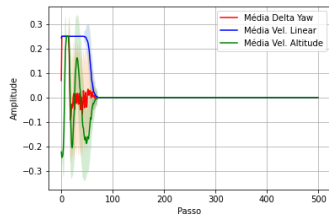
(b) NDRL-S MLP com duas camadas escondidas.



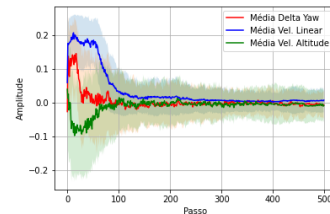
(c) NDRL-D MLP com três camadas escondidas.



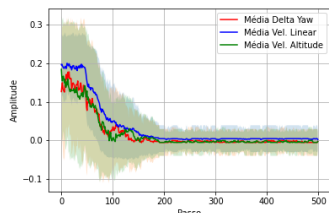
(d) NDRL-S MLP com três camadas escondidas.



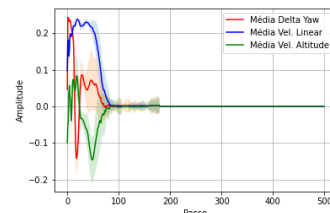
(e) NDRL-D MLP com quatro camadas escondidas.



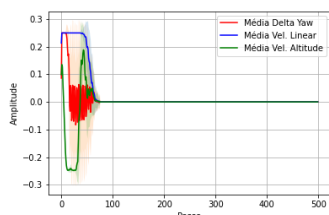
(f) NDRL-S MLP com quatro camadas escondidas.



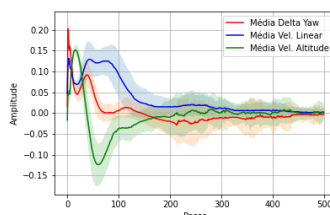
(g) NDRL-D MLP com cinco camadas escondidas.



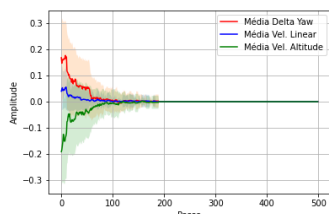
(h) NDRL-S MLP com cinco camadas escondidas.



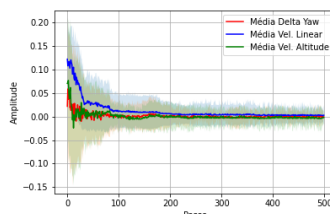
(i) NDRL-D LSTM.



(j) NDRL-S LSTM.

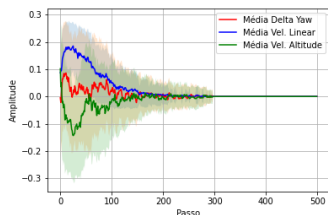


(k) NDRL-D convolucional.

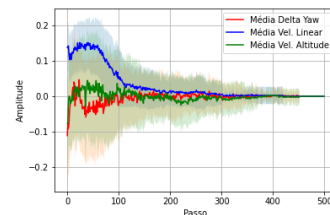


(l) NDRL-S convolucional.

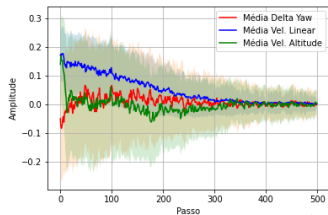
Figura 53: Ações dos agentes realizadas durante as 100 tentativas para a primeira tarefa no contexto 3D no primeiro cenário. Em vermelho a média do delta yaw, em azul da velocidade linear e em verde a velocidade de altitude. A sombra representa o desvio padrão.



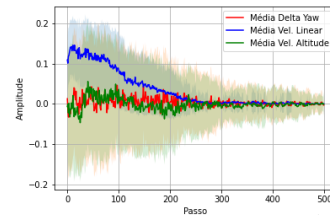
(a) NDRL-D MLP com duas camadas escondidas.



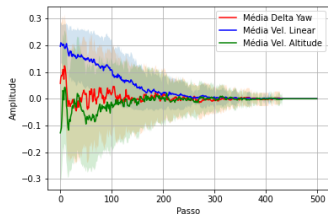
(b) NDRL-S MLP com duas camadas escondidas.



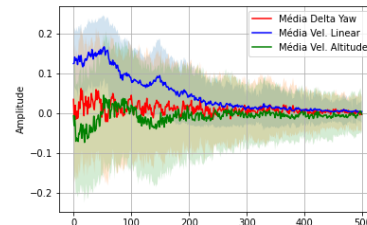
(c) NDRL-D MLP com três camadas escondidas.



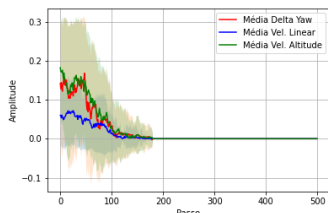
(d) NDRL-S MLP com três camadas escondidas.



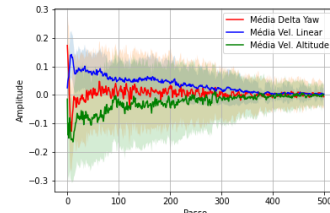
(e) NDRL-D MLP com quatro camadas escondidas.



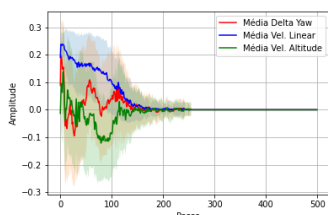
(f) NDRL-S MLP com quatro camadas escondidas.



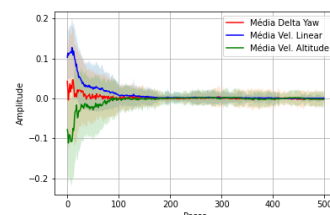
(g) NDRL-D MLP com cinco camadas escondidas.



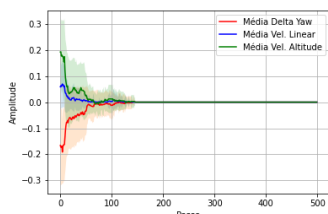
(h) NDRL-S MLP com cinco camadas escondidas.



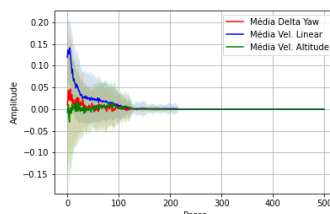
(i) NDRL-D LSTM.



(j) NDRL-S LSTM.



(k) NDRL-D convolucional.



(l) NDRL-S convolucional.

Figura 54: Ações dos agentes realizadas durante as 100 tentativas para a primeira tarefa no contexto 3D no segundo cenário. Em vermelho a média do delta yaw, em azul da velocidade linear e em verde a velocidade de altitude. A sombra representa o desvio padrão.

6.2.2 Navegação sem mapa orientada a múltiplos pontos

Nas Tabelas 26 e 27 pode-se observar o total de tentativas em que a navegação foi realizada com sucesso para cada abordagem em cada cenário para a segunda tarefa.

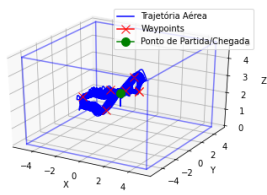
Tabela 26: Estatísticas da navegação orientada a alvo para o contexto 2D da abordagem baseado em DDPG na segunda tarefa.

Cenário	Abordagem	Total Sucesso	Tempo Médio (s)	Percentual Percorrido
1	NDRL-D 2	100%	72.90 ± 2.21	100%
1	NDRL-D 3	77%	76.99 ± 29.23	86.44%
1	NDRL-D 4	99%	90.120 ± 19.90	98.396%
1	NDRL-D 5	0%	19.529 ± 20.32	1.02%
1	NDRL-D LSTM	99%	90.44 ± 6.02	98.97%
1	NDRL-D CONV	0%	20.53 ± 11.1790	0.62%
2	NDRL-D 2	0%	29.194 ± 23.434	2.24%
2	NDRL-D 3	0%	39.8 ± 28.85	2.24%
2	NDRL-D 4	0%	38.41 ± 31.38	4.897%
2	NDRL-D 5	0%	17.95 ± 6.486	0%
2	NDRL-D LSTM	0%	34.9 ± 25.37	8.36%
2	NDRL-D CONV	0%	21.01 ± 18.51	0%

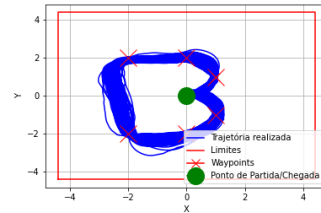
Tabela 27: Estatísticas da navegação orientada a alvo para o contexto 2D da abordagem baseada SAC na segunda tarefa.

Cenário	Abordagem	Total Sucesso	Tempo Médio (s)	Percentual Percorrido
1	NDRL-S 2	0%	114.01 ± 67.66	37.463%
1	NDRL-S 3	10%	135.220 ± 61.154	47.084%
1	NDRL-S 4	52%	140.94 ± 50.30	68.22%
1	NDRL-S 5	26%	173.124 ± 48.973	55.685%
1	NDRL-S LSTM	94%	155.82 ± 60.69	95.34%
1	NDRL-S CON	0%	121.21 ± 73.98	1.603%
2	NDRL-S 2	0%	25.054 ± 15.52	0.204%
2	NDRL-S 3	0%	42.44 ± 31.20	3.469%
2	NDRL-S 4	3%	61.382 ± 55.434	16.938%
2	NDRL-S 5	1%	57.499 ± 28.62	3.06%
2	NDRL-S LSTM	0%	34.9 ± 25.37	1.224%
2	NDRL-S CONV	0%	29.46 ± 12.94	0%

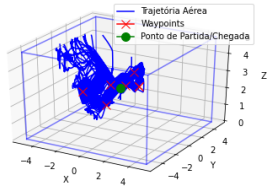
Nas Figuras 55, 56, 57 e 58 é possível observar as 100 tentativas de navegação realizada por ambos agentes no primeiro e segundo cenários e para cada abordagem, respectivamente.



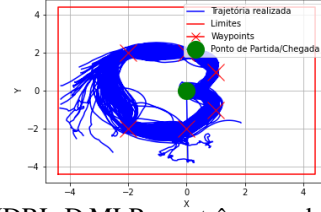
(a) NDRL-D MLP com duas camadas escondidas.



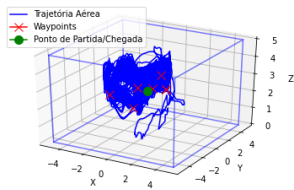
(b) NDRL-D MLP com duas camadas escondidas visto de cima.



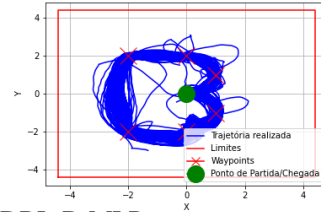
(c) NDRL-D MLP com três camadas escondidas.



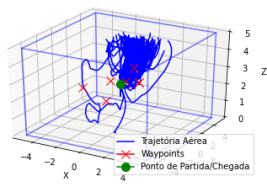
(d) NDRL-D MLP com três camadas escondidas visto de cima.



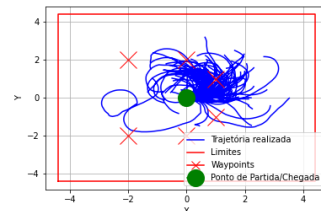
(e) NDRL-D MLP com quatro camadas escondidas.



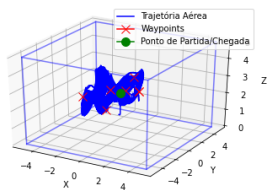
(f) NDRL-D MLP com quatro camadas escondidas visto de cima.



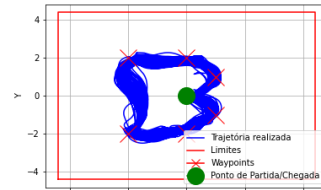
(g) NDRL-D MLP com cinco camadas escondidas.



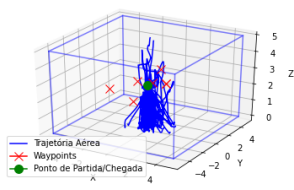
(h) NDRL-D MLP com cinco camadas escondidas visto de cima.



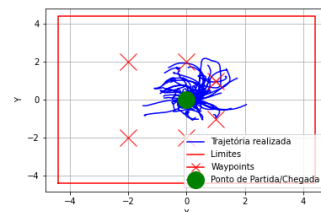
(i) NDRL-D LSTM.



(j) DDPG LSTM visto de cima.

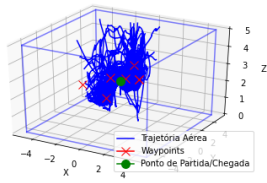


(k) NDRL-D convolucional.

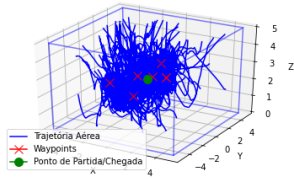


(l) NDRL-D convolucional visto de cima.

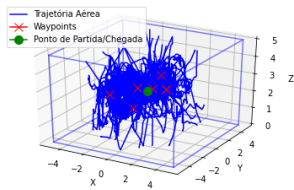
Figura 55: Caminho realizado durante as 100 tentativas na primeira tarefa no contexto 2D para o primeiro cenário para a abordagem DDPG.



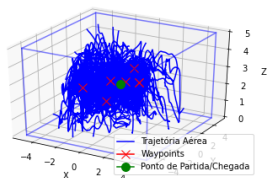
(a) NDRL-S MLP com duas camadas escondidas.



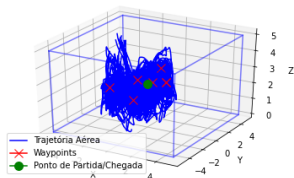
(c) NDRL-S MLP com três camadas escondidas.



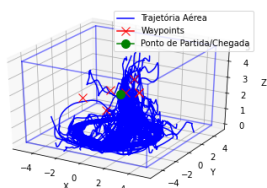
(e) NDRL-S MLP com quatro camadas escondidas.



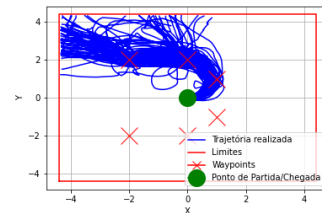
(g) NDRL-S MLP com cinco camadas escondidas.



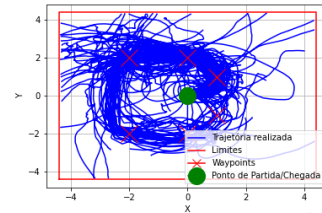
(i) NDRL-S LSTM.



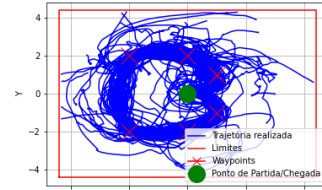
(k) NDRL-S convolucional.



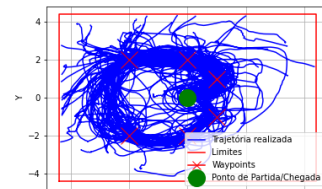
(b) NDRL-S MLP com duas camadas escondidas visto de cima.



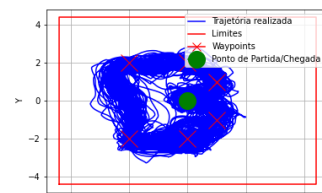
(d) NDRL-S MLP com três camadas escondidas visto de cima.



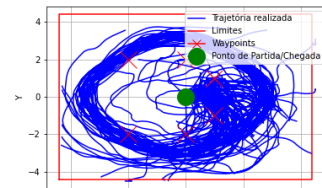
(f) NDRL-S MLP com quatro camadas escondidas visto de cima.



(h) NDRL-S MLP com cinco camadas escondidas visto de cima.

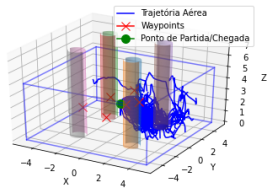


(j) NDRL-S LSTM visto de cima.

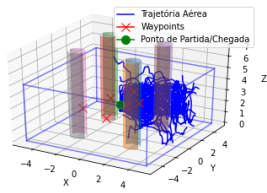


(l) NDRL-S convolucional visto de cima.

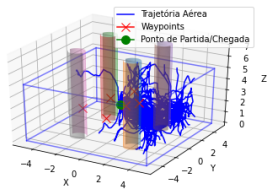
Figura 56: Caminho realizado durante as 100 tentativas na primeira tarefa no contexto 2D para o primeiro cenário para a abordagem SAC.



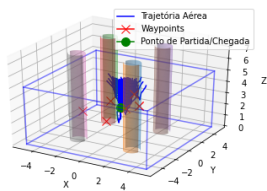
(a) NDRL-D MLP com duas camadas escondidas.



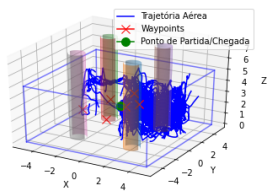
(c) NDRL-D MLP com três camadas escondidas.



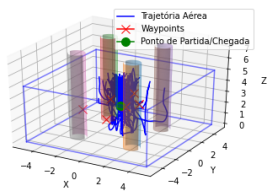
(e) NDRL-D MLP com quatro camadas escondidas.



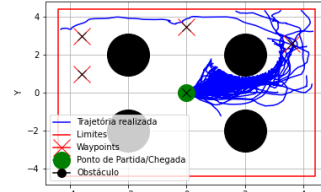
(g) NDRL-D MLP com cinco camadas escondidas.



(i) NDRL-D LSTM.



(k) NDRL-D convolucional.



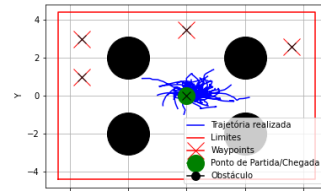
(b) NDRL-D MLP com duas camadas escondidas visto de cima.



(d) NDRL-D MLP com três camadas escondidas visto de cima.



(f) NDRL-D MLP com quatro camadas escondidas visto de cima.



(h) NDRL-D MLP com cinco camadas escondidas visto de cima.

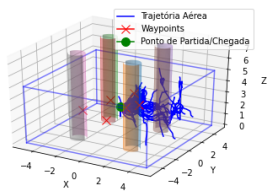


(j) NDRL-D LSTM visto de cima.

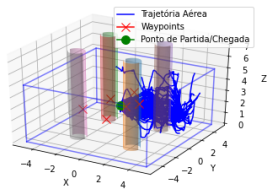


(l) NDRL-D convolucional visto de cima.

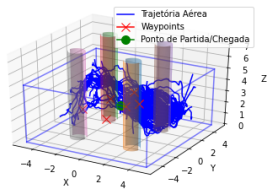
Figura 57: Caminho realizado durante as 100 tentativas na primeira tarefa no contexto 2D para o segundo cenário para a abordagem DDPG.



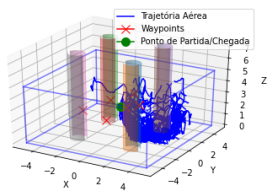
(a) NDRL-S MLP com duas camadas escondidas.



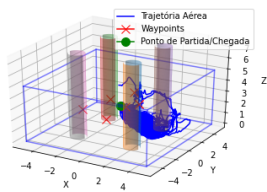
(c) NDRL-S MLP com três camadas escondidas.



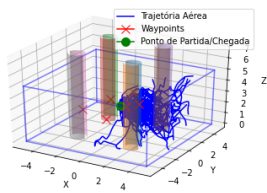
(e) NDRL-S MLP com quatro camadas escondidas.



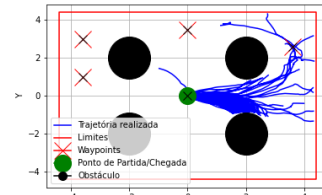
(g) NDRL-S MLP com cinco camadas escondidas.



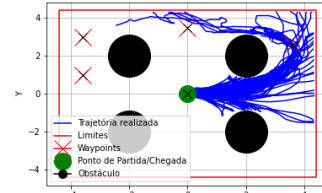
(i) NDRL-S LSTM.



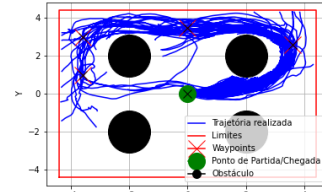
(k) NDRL-S convolucional.



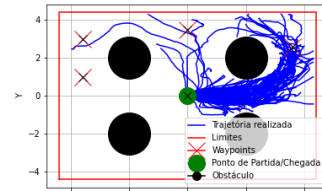
(b) NDRL-S MLP com duas camadas escondidas visto de cima.



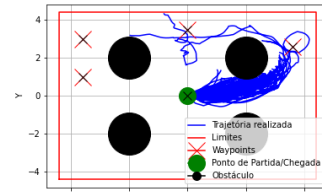
(d) NDRL-S MLP com três camadas escondidas visto de cima.



(f) NDRL-S MLP com quatro camadas escondidas visto de cima.



(h) NDRL-S MLP com cinco camadas escondidas visto de cima.



(j) NDRL-S LSTM visto de cima.



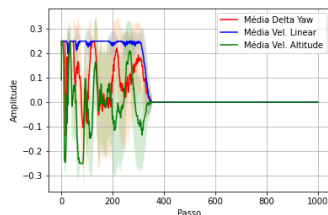
(l) NDRL-S convolucional visto de cima.

Figura 58: Caminho realizado durante as 100 tentativas na primeira tarefa no contexto 2D para o segundo cenário para a abordagem SAC.

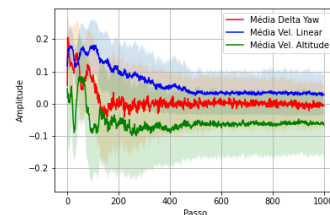
Nas Figuras 59 e 60 é possível observar a média de cada uma das ações de cada um dos agentes em cada passo durante a realização da tarefa para cada um dos respectivos cenários.

O contexto 3D merece destaque para a abordagem convolucional. Para ambas as tarefas e em ambos os cenários os agentes tiveram desempenhos muito ruins. O destaque positivo mais uma vez foi a estrutura com LSTM na abordagem estocástica que obteve os melhores resultados da série.

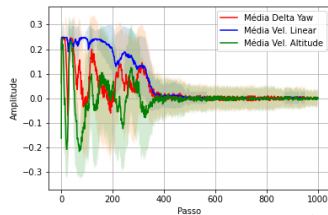
De modo geral, pode-se concluir também, levando em consideração o contexto 2D, que os melhores resultados foram obtidos com a estrutura de MLP com três camadas e que os resultados tendem a melhorar com o aumento do número de camadas escondidas para a abordagem estocástica, enquanto que o oposto ocorre com a abordagem determinística. Destaca-se também o potencial das abordagens com LSTM para o futuro e o limite do problema proposto que foi atingido com a versão convolucional. Para o segundo caso, um estudo mais aprofundado com CNNs precisa ser realizado.



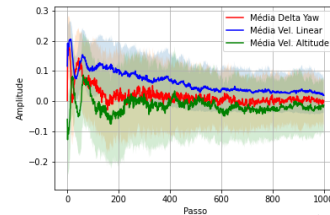
(a) NDRL-D MLP com duas camadas escondidas.



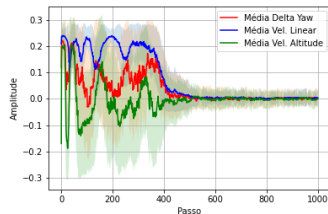
(b) NDRL-S MLP com duas camadas escondidas.



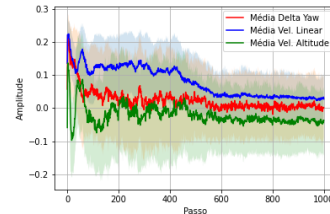
(c) NDRL-D MLP com três camadas escondidas.



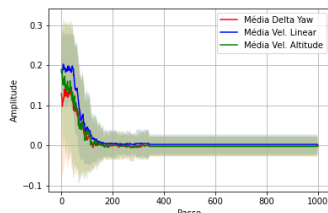
(d) NDRL-S MLP com três camadas escondidas.



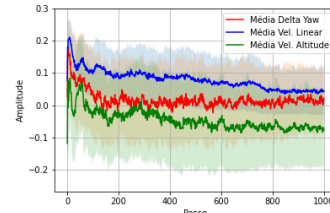
(e) NDRL-D MLP com quatro camadas escondidas.



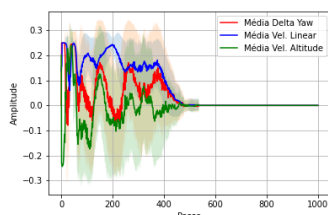
(f) NDRL-S MLP com quatro camadas escondidas.



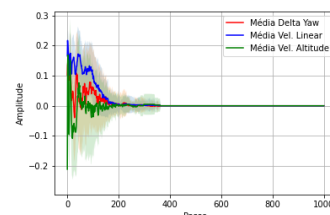
(g) NDRL-D MLP com cinco camadas escondidas.



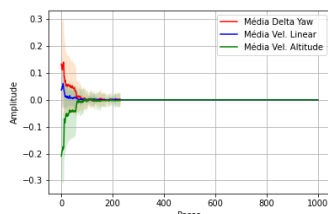
(h) NDRL-S MLP com cinco camadas escondidas.



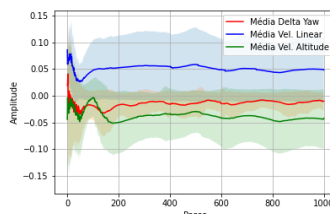
(i) NDRL-D LSTM.



(j) NDRL-S LSTM.

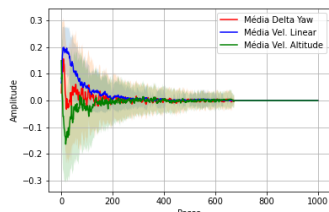


(k) NDRL-D convolucional.

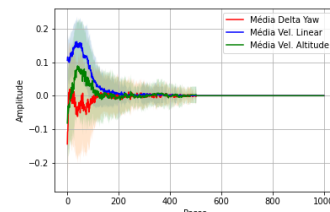


(l) NDRL-S convolucional.

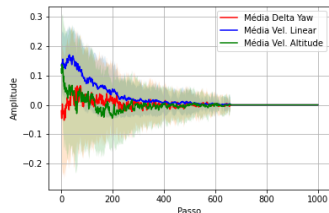
Figura 59: Ações dos agentes realizadas durante as 100 tentativas para a primeira tarefa no contexto 3D no primeiro cenário. Em vermelho a média do delta yaw, em azul da velocidade linear e em verde a velocidade de altitude. A sombra representa o desvio padrão.



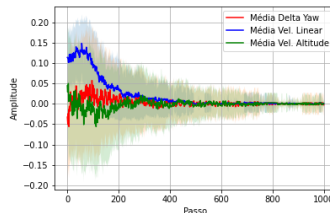
(a) NDRL-D MLP com duas camadas escondidas.



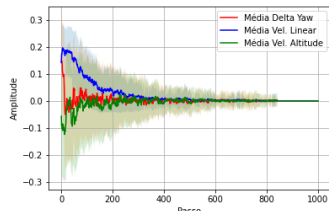
(b) NDRL-S MLP com duas camadas escondidas.



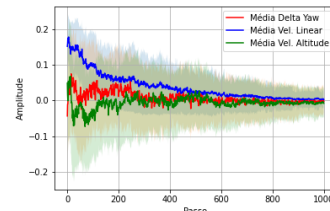
(c) NDRL-D MLP com três camadas escondidas.



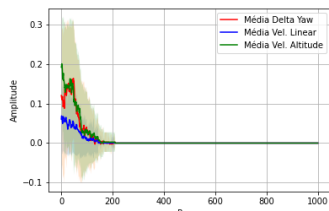
(d) NDRL-S MLP com três camadas escondidas.



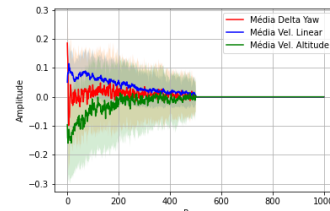
(e) NDRL-D MLP com quatro camadas escondidas.



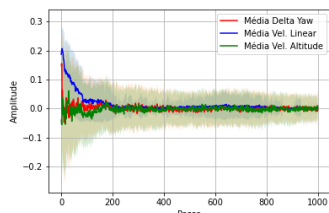
(f) NDRL-S MLP com quatro camadas escondidas.



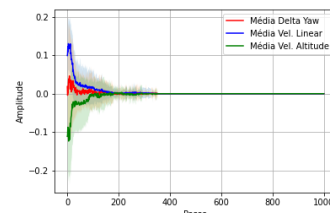
(g) NDRL-D MLP com cinco camadas escondidas.



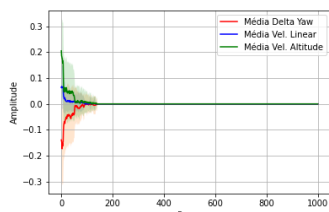
(h) NDRL-S MLP com cinco camadas escondidas.



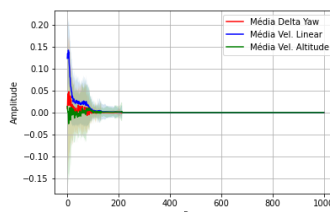
(i) NDRL-D LSTM.



(j) NDRL-S LSTM.



(k) NDRL-D convolucional.



(l) NDRL-S convolucional.

Figura 60: Ações dos agentes realizadas durante as 100 tentativas para a primeira tarefa no contexto 3D no segundo cenário. Em vermelho a média do delta yaw, em azul da velocidade linear e em verde a velocidade de altitude. A sombra representa o desvio padrão.

6.3 Função de Ativação

Visando um estudo comparativo específico para a função de ativação utilizada na saída das ANNs, foi desenvolvido um modelo que utiliza a função de ativação *Hardtanh*. Essa função foi escolhida pois possui os mesmos limiares da função *Tanh*, diferenciando-se basicamente por ser linear no intervalo de atuação. Dessa forma, busca-se fazer uma comparação entre um função linear e uma não linear. Nas tabelas 28 e 29 é possível observar as estatísticas das 100 tentativas para para cada tarefa, enquanto que nas nas Figuras 61 e 62 pode-se observar a navegação realizada por ambos os agentes nos contextos 2D e 3D, respectivamente.

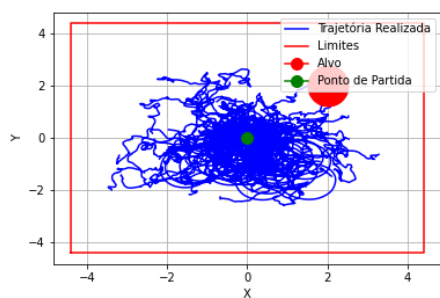
Tabela 28: Estatísticas da navegação orientada a alvo nas abordagens com *Hardtanh*.

Cenário	Abordagem	Total Sucesso	Tempo Médio (s)
1	NDRL-D 2D	0%	167.61 ± 127.48
1	NDRL-D 3D	0%	20.46 ± 13.546
1	NDRL-S 2D	0%	96.329 ± 30.443
1	NDRL-S 3D	0%	11.05 ± 0.41

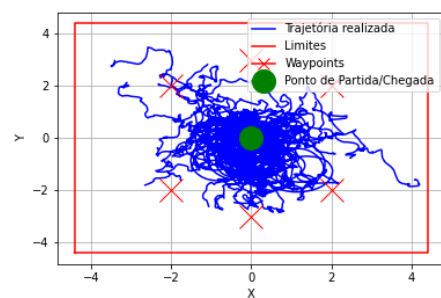
Tabela 29: Estatísticas da navegação orientada a alvo para as abordagens com *Hardtanh* na segunda tarefa.

Cenário	Abordagem	Total Sucesso	Tempo Médio (s)	Percentual Percorrido
1	NDRL-D 2D	0%	158.78 ± 126.75	0%
1	NDRL-D 3D	0%	19.842 ± 12.053	0%
1	NDRL-S 2D	0%	202.34 ± 1.75	0%
1	NDRL-S 3D	0%	11.12 ± 0.43	0%

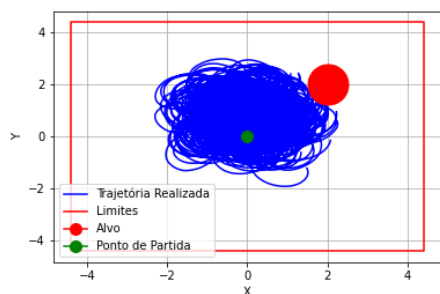
Observa-se que ambos os agentes e em ambos os contextos e cenários não foram capazes de aprender a realizar a tarefa. Com uma função linear as abordagens não foram capazes de convergir. Um estudo mais aprofundado com mais funções de ativações também precisa ser realizado para analisar com mais detalhes.



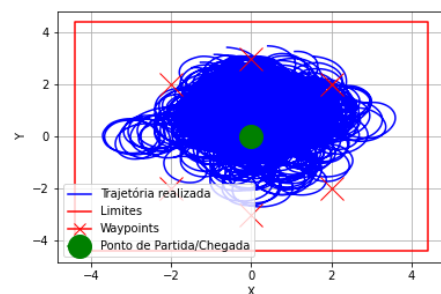
(a) NDRL-D no contexto 2D na primeira tarefa.



(b) NDRL-D no contexto 2D na segunda tarefa.

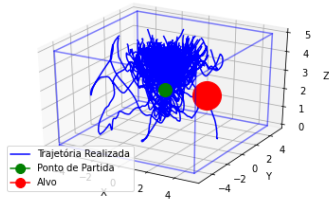


(c) NDRL-S no contexto 2D na primeira tarefa.

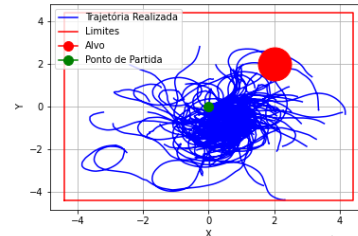


(d) NDRL-S no contexto 2D na segunda tarefa.

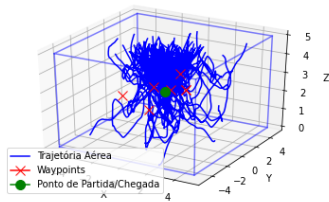
Figura 61: Caminho realizado durante as 100 tentativas com a função de ativação *Hard-tanh* no contexto 2D.



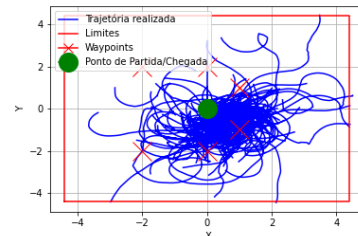
(a) NDRL-D no contexto 3D na primeira tarefa.



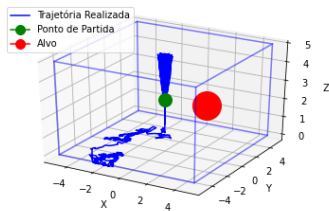
(b) NDRL-D no contexto 3D na primeira tarefa visto de cima.



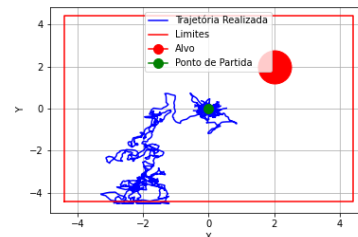
(c) NDRL-D no contexto 3D na segunda tarefa.



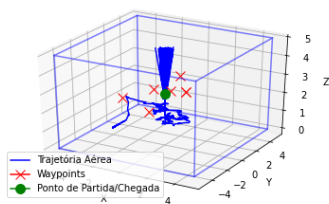
(d) NDRL-D no contexto 3D na segunda tarefa visto de cima.



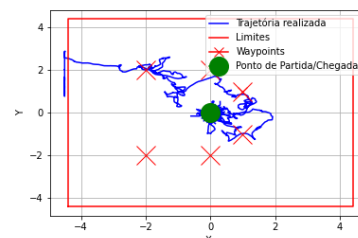
(e) NDRL-S no contexto 3D na primeira tarefa.



(f) NDRL-S no contexto 3D na primeira tarefa visto de cima.



(g) NDRL-S no contexto 3D na segunda tarefa.



(h) NDRL-S no contexto 3D na segunda tarefa visto de cima.

Figura 62: Caminho realizado durante as 100 tentativas com a função de ativação *Hard-tanh* no contexto 3D.

6.4 Discussão

De modo geral, a extensiva validação dos variados modelos criados e testados mostra que ambos os agentes são flexíveis com relação ao tipo de ANNs utilizada. Pode-se concluir que a abordagem baseada em DDPG possui um melhor desempenho em um cenário sem obstáculos, enquanto que o contrário ocorre com as abordagens baseada em SAC. Na Figura 63 é apresentada a recompensa final de cada modelo em cada contexto e cenário.

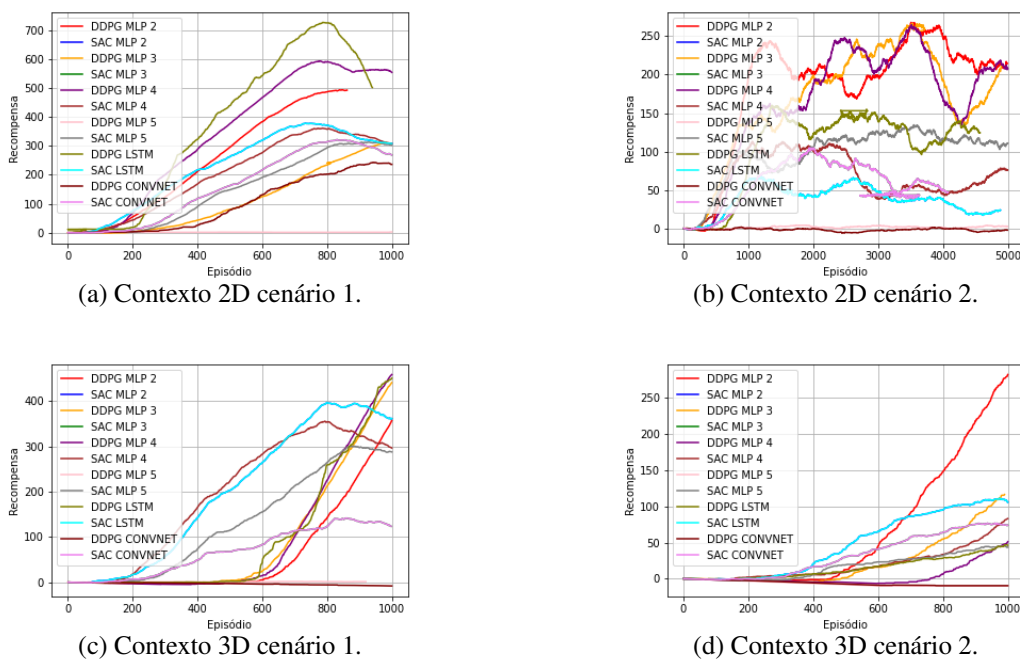


Figura 63: Recompensa comparativa do estudo de validação.

Pode-se observar que quanto maior e mais complexa a rede, maior tende a ser o reforço médio, como por exemplo para os modelos com LSTM e CNN. É importante ressaltar que isso se deve ao fato de um número maior de navegações ser realizado a cada episódio e não que os modelos são melhores ou piores. Todos os modelos com reforço médio superior a 100 podem ser considerados funcional. O número maior de navegações é devido ao passo mais lento com redes mais complexas, permitindo que a tarefa seja concluída e otimizada com um número menor de ações. Isso reforça ainda mais a importância de focar em um sistema recompensatório simples como o proposto neste trabalho.

Na Figura 65 é possível observar a comparação do tempo médio para a realização da primeira no contexto 2D, contexto onde as abordagens apresentaram resultados médio próximos ao máximo possível para todas as estruturas.

É interessante observar na Figura 65 como elas apresentam as características de cada abordagem com mais detalhes. É possível observar que a abordagem SAC possui um tempo médio semelhante entre as estruturas, enquanto que a abordagem baseada em

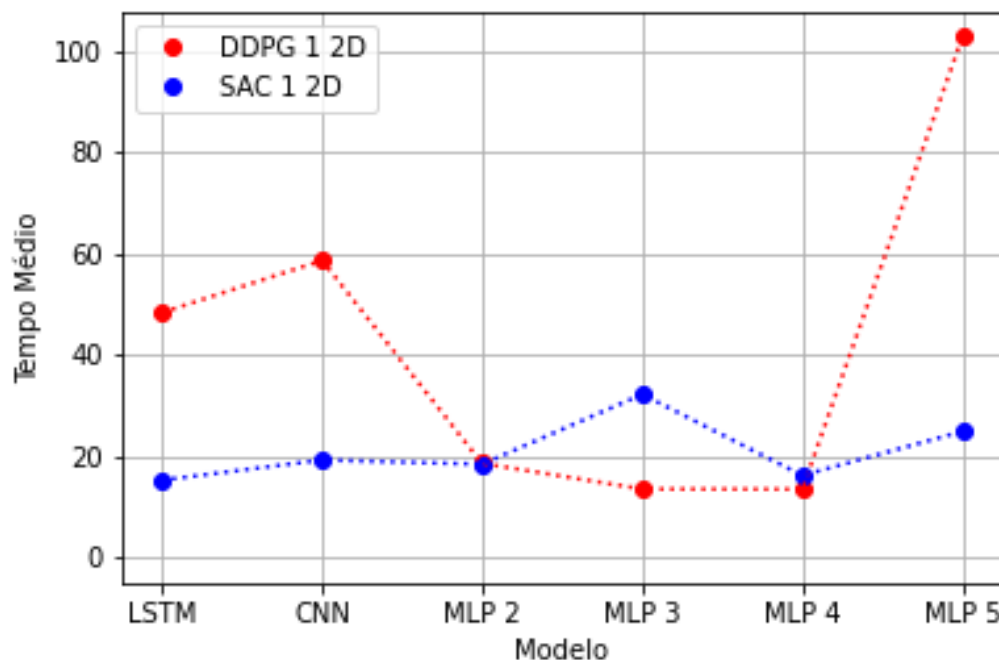


Figura 64: Comparação do tempo médio para o contexto 2D no primeiro cenário (mais estável) na primeira tarefa.

DDPG varia com maior intensidade. Isso é devido a maior capacidade de generalização que o método com viés estocástico como a SAC proporciona, ao mesmo tempo que DDPG pode ser muito bom para estruturas específicas. De modo geral, com relação ao tempo é possível concluir que a abordagem baseada em SAC tende a ser em média um pouco maior e mais previsível, enquanto que o oposto ocorre com abordagens baseadas em DDPG.

Na Figura 65 pode-se observar também a distância média para a segunda tarefa no segundo cenário, também no contexto 2D por ser mais de modo geral estável entre as estruturas.

Dessa ilustração é interessante observar ainda com mais detalhes as características de cada abordagem. Pode-se observar como a abordagem baseada em DDPG possui um melhor desempenho com 2 camadas e como o desempenho cai de forma exponencial com o aumento da complexidade da rede. Enquanto isso, a abordagem SAC apresenta melhores resultados com estruturas de rede mais complexas, aumentando o desempenho conforme o aumento do número de camadas por exemplo. Isso se deve a capacidade de generalização e criação de maiores gradientes que o método baseado em com viés estocástico SAC possui. De modo geral, pode-se concluir que quanto maior a rede melhor tende a ser o desempenho de agentes baseados em SAC, enquanto que o oposto ocorre com DDPG.

O limite para a complexidade, entretanto, parece ser próximo ao modelo convolucional proposto. Como pode-se observar na Figura 65 e também nos resultados para o

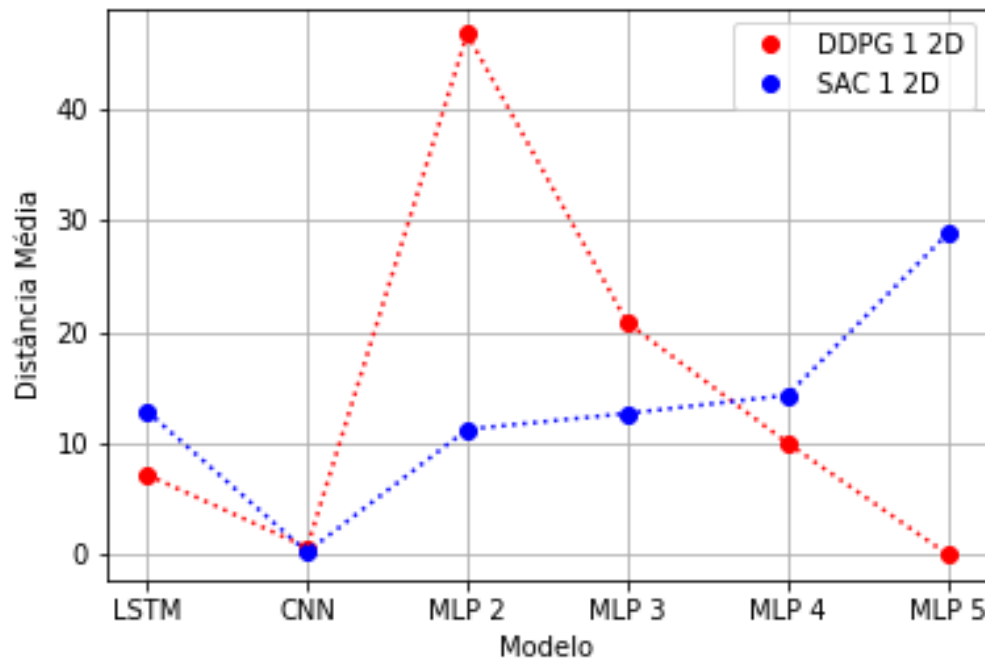


Figura 65: Comparação da distância média para o contexto 2D no primeiro cenário (mais estável) na segunda tarefa.

contexto 3D, ambas as abordagens com CNN não conseguiram aprender a realizar as tarefas. A solução para isso pode ser redes contrastivas [Srinivas et al., 2020]. A utilização de redes contrastivas com Deep-RL pode ser um caminho não só para resolver esse problema com CNNs, mas também para otimizar a problemática do trabalho como um todo.

Com relação a função de ativação, observa-se que os agentes não foram capazes de aprender usando uma função de ativação linear *Hardtanh*, enquanto que apresentaram convergência com *Tanh*.

7 CONSIDERAÇÕES FINAIS E PERSPECTIVAS

Neste trabalho, foi proposto abordagens baseadas em Deep-RL para realizar navegação sem mapa de um veículo híbrido capaz de atuar no ar e na água. Um HUAUV com estrutura de quadrotor baseado em um veículo real foi descrito utilizando *plugins* de simulação aérea e subaquática disponíveis para simulação de robôs. Duas abordagens baseadas em DDPG e SAC foram desenvolvidas e aplicadas a duas tarefas em variados ambientes e cenários no contexto 2D, 3D e híbrido e comparadas com o algoritmo BUG2.

Com os resultados obtidos, pode-se verificar que abordagens baseadas em Deep-RL para navegação sem mapa, tradicionalmente usadas em robôs móveis terrestres, podem ser usadas para fazer a inteligência de navegação para HUAUVs. A estrutura de agente e rede proposta, utilizando somente informação de um sensor de distância e a informação da localização do veículo, mostrou ter potencial para realizar navegação sem mapa orientada a alvo e navegação sem mapa orientada a múltiplos alvos. Sem utilizar uma abordagem baseada em informação visual, que demanda um *hardware* com alta capacidade de processamento que é geralmente custoso em peso e difícil de ser embarcado em HUAUVs de pequeno porte, tal como o HUAUV Hydrone, a abordagem simples e com poucos estados proposta mostrou-se ser capaz realizar a navegação do veículo e fazê-lo ser inteligente o suficiente para evitar colisões com objetos em um contexto 2D. Apesar de promissores, os resultados obtidos até o momento possuem bastante espaço para serem aprimorados como mostrado com mais detalhes no estudo de validação.

Para a sequência do desenvolvimento do trabalho propõe-se várias perspectivas. Primeiro propõe-se uma modelagem subaquática mais completa pois como mostrados nos resultados da parte subaquática do trabalho há muito espaço para isso ser aprimorado. Isso, entretanto, depende de um estudo futuro mais aprofundado. Além disso, propõe-se a criação e testes em ambientes mais complexos e com obstáculos dinâmicos, o que ainda não foi possível. Por último, propõe-se a utilização de abordagens que utilizem uma combinação de estruturas de rede com LSTM e com CNN mas com aprendizado contrastivo. Essas parecem ser duas vertentes interessantes no futuro próximo para o problema abordado neste trabalho.

REFERÊNCIAS

Silas FR Alves, Joao M Rosario, Humberto Ferasoli Filho, LK Rincon, RA Yamasaki, and Alejandra Barrera. Conceptual bases of robot navigation modeling, control and applications. *Advances in Robot Navigation*, page 26, 2011.

Hamzeh Alzu'bi, Iyad Mansour, and Osamah Rawashdeh. Loon copter: Implementation of a hybrid unmanned aquatic–aerial quadcopter with active buoyancy control. *Journal of field Robotics*, 35(5):764–778, 2018.

Viktor Berg. Development and commissioning of a dp system for rovs sf 30k. Master's thesis, Institutt for marin teknikk, 2012.

Gary Bradski and Adrian Kaehler. Opencv. *Dr. Dobb's journal of software tools*, 2000.

Ignacio Carlucho, Mariano De Paula, Sen Wang, Bruno V Menna, Yvan R Petillot, and Gerardo G Acosta. Auv position tracking control using end-to-end deep reinforcement learning. In *OCEANS 2018 MTS/IEEE Charleston*, pages 1–8. IEEE, 2018.

J Cartwright, N Johnson, BC Davis, Z Qiang, A Linares, A Enoch, G Lemaitre, H Roth, and Y Petillot. Nessie iii autonomous underwater vehicle for sauc-e 2008. *Proceedings of The 10th Unmanned Underwater Vehicle Showcase*, 2008.

Rômulo Cerqueira, Tiago Trocoli, Gustavo Neves, Luciano Oliveira, Sylvain Joyeux, Jan Albiez, and Robotics Innovation Center. Custom shader and 3d rendering for computationally efficient sonar simulation. In *29th Conference on Graphics, Patterns and Images-SIBGRAPI*, 2016.

Rosen Diankov and James Kuffner. Openrave: A planning architecture for autonomous robotics. *Robotics Institute, Pittsburgh, PA, Tech. Rep. CMU-RI-TR-08-34*, 79, 2008.

Paulo LJ Drews, Armando Alves Neto, and Mario FM Campos. Hybrid unmanned aerial underwater vehicle: Modeling and simulation. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4637–4642. IEEE, 2014.

Chelsey Edge, Sadman Sakib Enan, Michael Fulton, Jungseok Hong, Jiawei Mo, Kimberly Barthelemy, Hunter Bashaw, Berik Kallevig, Corey Knutson, Kevin Orpen, et al. Design and experiments with loco auv: A low cost open-source autonomous underwater vehicle. *arXiv preprint arXiv:2003.09041*, 2020.

You-Hong Eng, Cheng-Siong Chin, and Michael Wai-Shing Lau. Added mass computation for control of an open-frame remotely-operated vehicle: Application using wamit and matlab. *Journal of Marine Science and Technology*, 22(4):405–416, 2014.

Tingxiang Fan, Xinjing Cheng, Jia Pan, Dinesh Manocha, and Ruigang Yang. Crowd-move: Autonomous mapless navigation in crowded scenarios. *arXiv preprint arXiv:1807.07870*, 2018.

Fadri Furrer, Michael Burri, Markus Achtelik, and Roland Siegwart. Rotors—a modular gazebo mav simulator framework. In *Robot Operating System (ROS)*, pages 595–625. Springer, 2016.

Brian Gerkey, Richard T Vaughan, and Andrew Howard. The player/stage project: Tools for multi-robot and distributed sensor systems. In *Proceedings of the 11th international conference on advanced robotics*, volume 1, pages 317–323, 2003.

Brian P Gerkey, Richard T Vaughan, Kasper Stoy, Andrew Howard, Gaurav S Sukhatme, and Maja J Mataric. Most valuable player: A robot device server for distributed control. In *Intelligent Robots and Systems, 2001. Proceedings. 2001 IEEE/RSJ International Conference on*, volume 3, pages 1226–1231. IEEE, 2001.

Ricardo B Grando, Junior C de Jesus, and Paulo LJ Drews-Jr. Deep reinforcement learning for mapless navigation of unmanned aerial vehicles. In *2020 Latin American Robotics Symposium (LARS), 2020 Brazilian Symposium on Robotics (SBR) and 2020 Workshop on Robotics in Education (WRE)*, pages 1–6. IEEE, 2020.

Ricardo Bedin Grando, Junior Costa de Jesus, Victor Augusto Kich, Alisson Henrique Kolling, Nicolas Pieper Bortoluzzi, Pedro Miranda Pinheiro, Armando Alves Neto, and Paulo Lilles Jorge Drews Jr. Deep reinforcement learning for mapless navigation of a hybrid aerial underwater vehicle with medium transition. *arXiv preprint arXiv:2103.12883*, 2021.

Alex Graves, Greg Wayne, Malcolm Reynolds, Tim Harley, Ivo Danihelka, Agnieszka Grabska-Barwińska, Sergio Gómez Colmenarejo, Edward Grefenstette, Tiago Ramalho, John Agapiou, et al. Hybrid computing using a neural network with dynamic external memory. *Nature*, 538(7626):471–476, 2016.

Shixiang Gu, Ethan Holly, Timothy Lillicrap, and Sergey Levine. Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates. In *2017 IEEE international conference on robotics and automation (ICRA)*, pages 3389–3396. IEEE, 2017.

José Jesús Guerrero, Ruben Martinez-Cantin, and Carlos Sagüés. Visual map-less navigation based on homographies. *Journal of Robotic Systems*, 22(10):569–581, 2005.

Mehmet Serdar Güzel. Autonomous vehicle navigation using vision and mapless strategies: a survey. *Advances in Mechanical Engineering*, 5:234747, 2013.

Mehmet Serdar Guzel and Robert Bicker. Vision based mapless navigation system for mobile robots using scale invariant features and optical flow, 2009.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *arXiv preprint arXiv:1801.01290*, 2018.

Alexandre C Horn, Pedro M Pinheiro, Cesar B Silva, Armando Alves Neto, and Paulo LJ Drews-Jr. A study on configuration of propellers for multirotor-like hybrid aerial-aquatic vehicles. In *2019 19th International Conference on Advanced Robotics (ICAR)*, pages 173–178. IEEE, 2019.

Junior C Jesus, Jair A Bottega, Marco ASL Cuadros, and Daniel FT Gamarra. Deep deterministic policy gradient for navigation of mobile robots in simulated environments. In *2019 19th International Conference on Advanced Robotics (ICAR)*, pages 362–367. IEEE, 2019.

Jun Jin, Nhat M Nguyen, Nazmus Sakib, Daniel Graves, Hengshuai Yao, and Martin Jagersand. Mapless navigation among dynamics with social-safety-awareness: a reinforcement learning approach from 2d laser scans. *arXiv preprint arXiv:1911.03074*, 2019.

Katie Kang, Suneel Belkhale, Gregory Kahn, Pieter Abbeel, and Sergey Levine. Generalization through simulation: Integrating simulated and real data into deep reinforcement learning for vision-based autonomous flight. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 6008–6014. IEEE, 2019.

Jens Kober, J Andrew Bagnell, and Jan Peters. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*, 32(11):1238–1274, 2013.

Nathan Koenig and Andrew Howard. Design and use paradigms for gazebo, an open-source multi-robot simulator. In *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No. 04CH37566)*, volume 3, pages 2149–2154. IEEE, 2004.

Vijay R Konda and John N Tsitsiklis. Actor-critic algorithms. In *Advances in neural information processing systems*, pages 1008–1014. Citeseer, 2000.

Petar Kormushev, Sylvain Calinon, and Darwin G Caldwell. Reinforcement learning in robotics: Applications and real-world challenges. *Robotics*, 2(3):122–148, 2013.

Taeyoung Lee, Melvin Leok, and N Harris McClamroch. Geometric tracking control of a quadrotor uav on se (3). In *49th IEEE conference on decision and control (CDC)*, pages 5420–5425. IEEE, 2010.

Yuxi Li. Deep reinforcement learning: An overview. *arXiv preprint arXiv:1701.07274*, 2017.

Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

Michael L Littman. Value-function reinforcement learning in markov games. *Cognitive systems research*, 2(1):55–66, 2001.

Zongcheng Ma, Jinfu Feng, and Jian Yang. Research on vertical air–water trans-media control of hybrid unmanned aerial underwater vehicles based on adaptive sliding mode dynamical surface control. *International Journal of Advanced Robotic Systems*, 15(2): 1729881418770531, 2018.

Matheus Machado, Guilherme Zaffari, Pedro Ballester, Paulo Drews-Jr, and Silvia Botelho. A topological descriptor of forward looking sonar images for navigation and mapping. In *Robotics*, pages 120–134. Springer, 2016.

MacroXML. Macro XML. <http://wiki.ros.org/xacro>, 2018. [Online; accessed 16-August-2018].

A Rupam Mahmood, Dmytro Korenkevych, Gautham Vasan, William Ma, and James Bergstra. Benchmarking reinforcement learning algorithms on real-world robots. *arXiv preprint arXiv:1809.07731*, 2018.

Marco M Maia, Parth Soni, and Francisco J Diez. Demonstration of an aerial and submersible vehicle capable of flight and underwater navigation with seamless air-water transition. *arXiv preprint arXiv:1507.01932*, 2015.

Marco M Maia, Diego A Mercado, and F Javier Diez. Design and implementation of multirotor aerial-underwater vehicles with experimental results. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 961–966. IEEE, 2017.

Musa Morena Marcusso Manhães, Sebastian A Scherer, Martin Voss, Luiz Ricardo Douat, and Thomas Rauschenbach. Uuv simulator: A gazebo-based package for underwater intervention and multi-robot simulation. In *OCEANS 2016 MTS/IEEE Monterey*, pages 1–8. IEEE, 2016.

Roberto Marino, Fulvio Mastrogiovanni, Antonio Sgorbissa, and Renato Zaccaria. A minimalistic quadrotor navigation strategy for indoor multi-floor scenarios. In *Intelligent Autonomous Systems 13*, pages 1561–1570. Springer, 2016.

D Mercado, M Maia, and F Javier Diez. Aerial-underwater systems, a new paradigm in unmanned vehicles. *Journal of Intelligent & Robotic Systems*, 95(1):229–238, 2019.

Mario Miranda, Pierre-Philippe Beaujean, Edgar An, and Manhar Dhanak. Homing an unmanned underwater vehicle equipped with a dusbl to an unmanned surface platform: A feasibility study. In *2013 OCEANS-San Diego*, pages 1–10. IEEE, 2013.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518 (7540):529–533, 2015.

Adept MobileRobots. Pioneer 3-at (2015). URL <http://www.mobilerobots.com/ResearchRobots/P3AT.aspx>, 2015.

Joseph Moore, Andrew Fein, and William Setzler. Design and analysis of a fixed-wing unmanned aerial-aquatic vehicle. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1236–1243. IEEE, 2018.

A Narayanan, P Rajeshirke, A Sharma, and K Pestonjamas. Survey of the emerging bio-inspired unmanned aerial underwater vehicles. In *IOP Conference Series: Materials Science and Engineering*, volume 810, page 012078. IOP Publishing, 2020.

Pauline C Ng and Steven Henikoff. Sift: Predicting amino acid changes that affect protein function. *Nucleic acids research*, 31(13):3812–3814, 2003.

Liam Paull, Sajad Saeedi, Mae Seto, and Howard Li. Auv navigation and localization: A review. *IEEE Journal of Oceanic Engineering*, 39(1):131–149, 2013.

Morgan Quigley, Ken Conley, Brian Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y Ng. Ros: an open-source robot operating system. In *ICRA workshop on open source software*, volume 3, page 5. Kobe, Japan, 2009.

Diego Alberto Mercado Ravell, Marco Moreno Maia, and Francisco Javier Diez. Modeling and control of unmanned aerial/underwater vehicles using hybrid control. *Control Engineering Practice*, 76:112–122, 2018.

Alejandro Rodriguez-Ramos, Carlos Sampedro, Hriday Bavle, Ignacio Gil Moreno, and Pascual Campoy. A deep reinforcement learning technique for vision-based autonomous multirotor landing on a moving platform. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1010–1017. IEEE, 2018.

Carlos Sampedro, Alejandro Rodriguez-Ramos, Hriday Bavle, Adrian Carrio, Paloma de la Puente, and Pascual Campoy. A fully-autonomous aerial robot for search and rescue applications in indoor environments using learning-based techniques. *Journal of Intelligent & Robotic Systems*, 95(2):601–627, 2019.

Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay. *arXiv preprint arXiv:1511.05952*, 2015.

Xueqiang Shen, Jiwei Fan, and Haiqing Wang. Design and simulation of eight-rotor unmanned aerial vehicle based on hybrid control system. *International Journal of Aerospace Engineering*, 2018, 2018.

Dave Shreiner, Bill The Khronos OpenGL ARB Working Group, et al. *OpenGL programming guide: the official guide to learning OpenGL, versions 3.0 and 3.1*. Pearson Education, 2009.

David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. 2014.

David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

Russell Smith. Open dynamics engine. <http://www.ode.org/>, 2001.

Aravind Srinivas, Michael Laskin, and Pieter Abbeel. Curl: Contrastive unsupervised representations for reinforcement learning. *arXiv preprint arXiv:2004.04136*, 2020.

William Stewart, Warren Weisler, Marc MacLeod, Thomas Powers, Aaron Defreitas, Richard Gritter, Mark Anderson, Kara Peters, Ashok Gopalarathnam, and Matthew Bryant. Design and demonstration of a seabird-inspired fixed-wing hybrid uav-uuv system. *Bioinspiration & biomimetics*, 13(5):056013, 2018.

Lei Tai, Giuseppe Paolo, and Ming Liu. Virtual-to-real deep reinforcement learning: Continuous control of mobile robots for mapless navigation. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 31–36. IEEE, 2017.

Yu Herng Tan, Rob Siddall, and Mirko Kovac. Efficient aerial–aquatic locomotion with a single propulsion system. *IEEE Robotics and Automation Letters*, 2(3):1304–1311, 2017.

George E Uhlenbeck and Leonard S Ornstein. On the theory of the brownian motion. *Physical review*, 36(5):823, 1930.

Kimion P Valavanis and George J Vachtsevanos. *Handbook of unmanned aerial vehicles*, volume 1. Springer, 2015.

Richard T Vaughan, Brian P Gerkey, and Andrew Howard. On device abstractions for portable, reusable robot code. In *Intelligent Robots and Systems, 2003.(IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*, volume 3, pages 2421–2427. IEEE, 2003.

Hui Wu, Shiji Song, Yachu Hsu, Keyou You, and Cheng Wu. End-to-end sensorimotor control problems of auvs with deep reinforcement learning. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5869–5874. IEEE, 2019.

Haitao Xiang and Lei Tian. Development of a low-cost agricultural remote sensing system based on an autonomous unmanned aerial vehicle (uav). *Biosystems engineering*, 108(2): 174–190, 2011.

He Xiongkui, Jane Bonds, Andreas Herbst, and Jan Langenakens. Recent development of unmanned aerial vehicle for plant protection in east asia. *International Journal of Agricultural and Biological Engineering*, 10(3):18–30, 2017.

Xingbang Yang, Tianmiao Wang, Jianhong Liang, Guocai Yao, and Miao Liu. Survey on the novel hybrid aquatic–aerial amphibious aircraft: Aquatic unmanned aerial vehicle (aquauav). *Progress in Aerospace Sciences*, 74:131–151, 2015.

Runsheng Yu, Zhenyu Shi, Chaoxing Huang, Tenglong Li, and Qiongqiong Ma. Deep reinforcement learning based optimal trajectory tracking control of autonomous underwater vehicle. In *2017 36th Chinese Control Conference (CCC)*, pages 4958–4965. IEEE, 2017.

Qilei Zhang, Jinying Lin, Qixin Sha, Bo He, and Guangliang Li. Deep interactive reinforcement learning for path following of autonomous underwater vehicle. *IEEE Access*, 8:24258–24268, 2020.