Trabalho de Conclusão de Curso

Squalo - Um framework de ASV para coleta de lixo

de **Hyago Procopio Brito**

orientado por
Prof. Mestre Glauber Rodrigues Leite

Universidade Federal de Alagoas
Instituto de Computação
Maceió, Alagoas
12 de Dezembro de 2022

UNIVERSIDADE FEDERAL DE ALAGOAS
Instituto de Computação

SQUALO - UM FRAMEWORK DE ASV PARA COLETA DE LIXO

Trabalho de Conclusão de Curso submetido ao Instituto de Computação da Universidade Federal de Alagoas como requisito parcial para a obtenção do grau de Engenheiro de Computação.

Hyago Procopio Brito

Orientador: Prof. Mestre Glauber Rodrigues Leite

Banca Avaliadora:

Ícaro Bezerra Queiroz de Araújo	Prof. Dr., UFALL
José Henrick Viana Ramalho	Prof. Msc., IFAL

Maceió, Alagoas
12 de Dezembro de 2022

Catalogação na fonte
Universidade Federal de Alagoas
Biblioteca Central
Divisão de Tratamento Técnico
Bibliotecária: Taciana Sousa dos Santos – CRB-4 – 2062

B862s Brito, Hyago Procopio.
Squalo – um framework de ASV para coleta de lixo / Hyago Procopio Brito. – 2022.
35 f. : il. color.

Orientador: Glauber Rodrigues Leite.
Monografia (Trabalho de Conclusão de Curso em Engenharia da Computação) – Universidade Federal de Alagoas. Instituto de Computação. Maceió, 2022.

Bibliografia: f. 34-35.

1. ASV. 2. *Framework.* 3. Simulação robótica. I. Título.

CDU: 004

Dedicatória

Dedico este trabalho aos meus pais, meus maiores incentivadores, pela educação que me deram e pelo amor que me dedicaram.

Hyago Procopio Brito,

Agradecimentos

Agradeço, primeiramente aos meus pais, por terem acreditado em mim, sempre investindo tudo que podiam e não podiam nos meus estudos. Agradeço à minha noiva, por ter sempre me apoiado, estado ao meu lado nos momentos bons e ruins. Agradeço aos meus professores e orientadores, Glauber Rodrigues e Ícaro Araújo, por terem me guiado e instruído da melhor forma possível. Por fim, agradeço aos meus amigos e familiares que fizeram parte dessa trajetória, contribuindo diretamente ou indiretamente para que esse trabalho se realizasse.

01 de Dezembro de 2022, Maceió,
Alagoas

Hyago Procopio Brito

A melhor maneira de prever o futuro é inventá-lo.
Alan Kay

Resumo

A água é um recurso precioso para a humanidade, mas, desde muito tempo, vem sofrendo com a poluição, desde produtos químicos, lixo sólido até os mais diversos tipos de resíduos. Devido a isso, se faz necessário monitorar e proteger esse recurso. Neste estudo, usando uma tecnologia de prototipação 3D, uma proposta de veículo autônomo de superfície (ASV) foi desenvolvida. Esse veículo é capaz de coletar lixo na superfície da água ao mesmo tempo em que coleta dados utilizando sensores. Entretanto, o projeto de robôs envolve tarefas de programação complexas, como sensoreamento, percepção, navegação e controle, além de problemas relacionados a eletrônica, mecânica e instrumentação. No domínio das aplicações marítmas, a simulação física, como a flutuação na água é uma especificidade, também complexa. Um *framework* robótico foi desenvolvido para dar suporte ao desenvolvimento de experimentação dos algoritmos em um ambiente de simulação confiável. O ciclo de vida do *framework* é descrito do seguinte modo: Dadas as coordenadas do resíduo, o controlador recebe os dados para processamento e computa a navegação para chegar ao destino, gerando comandos de força a serem aplicados a cada motor do veículo. Como prova de conceito, um MPC (*Model Predictive Controller*) não linear foi implementado utilizando as funções do *framework*, provendo métricas de controle padrão para testar diferentes configurações.

***Palavras-chave*: *Framework*; *Simulação robótica*; *ASV*.**

Abstract

Water is a precious resource for humankind, but for a long time, it has suffered from pollution from chemical products and solid waste to various types of residues. Because of that, it became necessary to monitor and protect this resource. In this study, using 3D prototyping technology, an autonomous surface vehicle (ASV) design was proposed. This vehicle can collect waste on the water surface and acquire data from water with sensors. However, along with problems concerning electronics, mechanics, and instrumentation, robot design involves complex programming tasks, such as sensing, perception, navigation, and control. Furthermore, simulated physics like water floating are specific to the domain. A robotic framework was developed to support the development and experimentation of such algorithms in a reliable simulation environment. The framework lifecycle is described as follows: Given the waste coordinates, the controller receives the data to process and compute the navigation to reach the destination, generating force commands to be applied in each engine of the vehicle. As concept proof, a nonlinear MPC (Model Predictive Controller) was implemented using the framework functions, providing standard control metrics to test different configurations.

***Keywords*: *Framework*; *Simulation*; *ASV*.**

Lista de Figuras

Lista de Tabelas

Lista de Símbolos

x Coordenada x do robô no plano

y Coordenada y do robô no plano

v Velocidade linear do robô

θ Orientação do robô

w Velocidade angular do robô

M Massa do robô

d Distância do eixo dos motores ao centro de massa

L Distância do motor ao eixo central do robô

J Momento de inércia em relação ao centro de massa

F_R Força do motor direito

F_L Força do motor esquerdo

V_{Ly} Velocidade do motor esquerdo no eixo Y

V_{Ry} Velocidade do motor direito no eixo Y

V_{Lx} Velocidade do motor esquerdo no eixo X

V_{Rx} Velocidade do motor direito no eixo Y

C Centro de massa do robô

V Velocidade resultante do robô

Yc Coordenada Y do centro de massa

Xc Coordenada X do centro de massa

F_{wL} Força do motor esquerdo no eixo Y

F_{wR} Força do motor direito no eixo Y

F_{uL} Força do motor esquerdo no eixo X

F_{uR} Força do motor direito no eixo X

v_w Velocidade local no eixo Y

v_u Velocidade local no eixo X

a_w Aceleração local no eixo Y

a_u Aceleração local no eixo X

X_r Deslocamento linear do robô

A Centro do eixo entre os motores

Xm Coordenadas do objetivo

Xd Coordenadas do robô

Fm_R Força do motor direito

Fm_L Força do motor esquerdo

Fx Força no eixo X

Fy Força no eixo Y

Fz Força no eixo Z

k Momento

p Horizonte máximo de predição

x_d Coordenada X desejada

y_d Coordenada Y desejada

θ_d Orientação desejada

v_d Velocidade linear desejada

w_d Velocidade angular desejada

e Diferença entre o valor de erro medido no robô e o valor de referência

Lista de Abreviaturas

IC Instituto de Computação.

MPC Model Predictive Control

USV Unmaned Surface Vehicle

ASV Autonomous Surface Vehicle

MIMO Multiple Input, Multiple Output

IAE Integral do módulo do erro

ISE Integral dos erros ao quadrado

ITAE Integral do módulo do erro vezes o tempo

Sumário

Capítulo 1

Introdução

Embora a água seja um recurso imprescindível para a humanidade, a forma que temos lidado com este precioso recurso não tem sido a melhor possível. Os resíduos na água vão de plástico Jambeck et al. (2015), metais, vidros, entre outros materiais que tem origem nos continentes Ministério do Meio Ambiente (2019). Estudos desenvolvidos na área, identificaram que, por exemplo, o Rio Danúbio na Europa, despeja cerca de 530 a 1500 toneladas de plástico no Mar Negro anualmente Lebreton et al. (2017). É um número alto, especialmente quando se considera que essa quantidade é de apenas um rio. Estima-se que a quantidade total de lixo despejada nos oceanos anualmente é algo em torno de 1.15 e 2.41 milhões de toneladas Lebreton et al. (2017).

Com o nível da poluição dos rios e oceanos alto, o desenvolvimento de veículos de superfície não tripulados (USV), e, consequentemente, veículos de superfície autônomos (ASV) passam a ser necessários. Diversas instituições já desenvolveram ou desenvolvem pesquisas na área, desde universidades, até mesmo militares, envolvendo os mais diversos campos, desde estudos da qualidade da água Dunbabin et al. (2009) e Chang et al. (2021a), estudo de trajetórias e desenvolvimento de novas tecnologias Bibuli et al. (2012) e até mesmo coletar o lixo da superfície da água Chang et al. (2021b). Ainda assim, mesmo com diversos estudos com esses veículos, diversas outras pesquisas foram desenvolvidas utilizando ambientes simulados através de diferentes *softwares* de renderização e simulação 3D, auxiliando no desenvolvimento de protótipos e tecnologias. Alguns estudos como Paravisi et al. (2019a) buscaram criar um ambiente simulado onde os veículos não tripulados operavam buscas em cenários de desastres bem como um sistema de trajetória variável em um ambiente com obstáculos, enquanto Sehgal and Cernea (2010a) buscou desenvolver um *framework* para simulação da comunicação de veículos não tripulados submersos na água.

Desenvolver soluções com robôs e veículos autônomos demanda tempo, de forma que muitas vezes é necessário reservar grande parte do trabalho em atividades que não estão relacionadas com a tecnologia que se pretende construir. Por exemplo, testar algoritmos de movimentação exigem que toda uma pilha de navegação, envolvendo mapeamento,

localização, controle e outros aspectos, estejam implementados em um certo grau de abstração. Testar um novo sistema de controle exige que simulações reais já estejam disponíveis e funcionais. Apesar de existirem diversos simuladores robóticos, em um domínio de aplicação mais específico, como o proposto neste estudo, as opções ficam ainda mais escassas.

Este trabalho visa a criação de um *framework* de veículo de superfície autônomo que seja capaz de coletar resíduos da superfície, onde seu modelo em 3D tem capacidade para instalação de sensores para coleta de dados de qualidade da água. O modelo em 3D foi desenvolvido utilizando o *software* AutoDesk Fusion 360 Fusion360 (2022), enquanto que o *framework* desenvolvido foi testado utilizando o Coppelia Robotics (2022) atráves de uma simulação.

1.1 Objetivos

1.1.1 Objetivo geral

Desenvolver um *framework* de veículo autônomo de superfície que seja capaz de navegar e coletar resíduos na superfície com capacidade para instalação de sensores de coleta de dados. O sistema deve ser capaz de receber uma informação (posição do resíduo) e ser capaz de iniciar uma trajetória até o objeto, coletando-o.

1.1.2 Objetivos específicos

- Estudo de um modelo matemático de robô com propulsão diferencial;
- Implementação de um controlador (MPC) para validação do framework;
- Desenvolvimento de um sistema de comunicação entre a simulação e o controlador em tempo real;
- Análise do comportamento do robô utilizando o MPC na simulação.

1.2 Organização do texto

O estudo está organizado em capítulos descrevendo cada etapa do desenvolvimento do trabalho, onde são fornecidas informações teóricas e a respeito da implementação da solução. No Capítulo 2 é apresentado o estado da arte, detalhando os trabalhos já desenvolvidos na área. No Capítulo 3, encontra-se a definição do problema, onde o modelo é definido, bem como suas equações. O Capítulo 4 apresenta a proposta do *framework* bem como seu diagrama de blocos e outras informações importantes do sistema proposto. O Capítulo 5 detalha a implementação do MPC para validação do *framework*. No Capítulo 6 são

discutidos os resultados obtidos com os cenários de simulação utilizados, de acordo com os objetivos propostos. Por fim, o Capítulo 7 conclui o trabalho, apontando dificuldades encontradas e trabalhos futuros.

Capítulo 2

Estado da Arte

Diversas instituições desenvolveram pesquisas e projetos na área dos veículos de superfície não tripulados (USV) e veículos de superfície autônomos (ASV), desde universidades e grandes empresas, até os militares. As pesquisas desenvolvidas envolveram os mais variados campos, desde estudos da qualidade da água Dunbabin et al. (2009) e Chang et al. (2021a), desenvolvimento e estudo conjunto de novas tecnologias Caccia et al. (2009) e até coleta de resíduos na superfície da água Chang et al. (2021a).

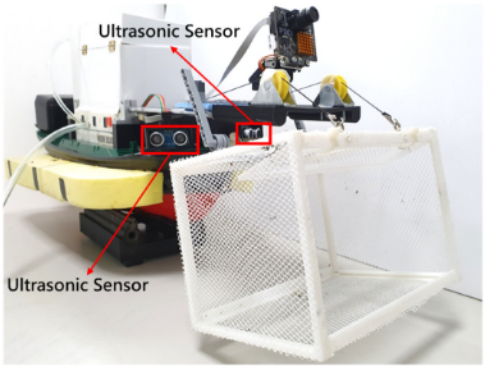
Ainda no campo do desenvolvimento desses veículos, diversas pesquisas foram desenvolvidas utilizando ambientes simulados através de diferentes programas de renderização 3D, auxiliando no desenvolvimento de protótipos e tecnologias. Alguns estudos, como por exemplo em Paravisi et al. (2019b) visaram criar um ambiente simulado envolvendo o uso de drones de superfície em desastres, com o objetivo de busca. Sehgal and Cernea (2010b) desenvolveu um *framework* para simular comunicação com drones em baixo da água. Svec et al. (2012) buscou desenvolver um sistema de trajetória variável num ambiente com obstáculos.

Dunbabin et al. (2009) desenvolveu um ASV com o propósito de estudos da qualidade da água, o Lake Wivenhoe, onde o protótipo possui capacidade de navegação autônoma mesmo fora do campo de visão, além de contar com painéis solares, o que o torna capaz de operar por até 24h. Devido ao seu design, como pode ser visto na Figura 2.1(a), possui capacidade de operação mesmo com influência de ventos de até 15 nós (aproximadamente 28 km/h). Uma desvantagem deste modelo, é o custo para ser produzido, já que possui painéis solares, bem como o seu tamanho, 16 pés (aproximadamente 5 metros) de comprimento.

Ainda falando de estudos relacionados à qualidade da água, Chang et al. (2021a) desenvolveu um USV, MF-USV (figura 2.1(b)), que, além do monitoramento da qualidade, também possui capacidade de remoção de objetos da superfície da água, e desvio de obstáculos. Utilizando visão computacional, o veículo é capaz de detectar um objeto na superfície, ajustar a própria direção e coletá-lo.

No campo da pesquisa e desenvolvimento de novas tecnologias, o USV Charlie (fi-

(a) ASV Lake Wivenhoe



(b) MF-USV

(c) Charlie USV (configuração de 2009)

Figura 2.1: Veículos de superfície desenvolvidos

gura 2.1(c)), foi um dos principais veículos. Caccia et al. (2009) descreve sua arquitetura, sistema de controle e fala ainda sobre diversas pesquisas que foram desenvolvidas utilizando-o, tais como pesquisas de sensoriamento marinho, estudos envolvendo possíveis tráfegos de veículos submarinos, desenvolvimento de técnicas de de navegação autônoma na superfície da água, entre diversas outras pesquisas.

No campo da simulação, diversos estudos foram desenvolvidos com o propósito de criar ou melhorar tecnologias que auxiliassem no desenvolvimento de diversos protótipos. Paravisi et al. (2019b) visou utilizar diferentes simuladores e ambientes para criar um framework capaz de gerar uma simulação de desastre natural e o uso de veículos de superfície nessas situações. O *framework* desenvolvido é capaz de simular as ações do vento e correnteza nesse tipo de veículo.

Sehgal and Cernea (2010b) buscou desenvolver um *framework* para comunicação com veículos subaquáticos, uma vez que para o desenvolvimento desse tipo de veículo se faz necessário uma boa qualidade de comunicação.

Svec et al. (2012) desenvolveu um sistema de trajetória capaz de seguir um alvo em movimento enquanto desvia de obstáculos. Durante o estudo foram utilizadas ferramentas de simulação e em seguida foi desenvolvido um ambiente físico com o intuito de validar o estudo desenvolvido.

Os trabalhos descritos acima funcionam muito bem, porém foram desenvolvidos, em sua maioria, em projetos relativamente grandes, com suporte de grandes centros de pesquisa. Entretanto, atualmente, com o avanço da tecnologia 3D, desenvolver protótipos de ASV se tornou mais prático e barato. O protótipo proposto neste trabalho visa ser um ASV de baixo custo, de fácil acesso, que pode ser impresso em 3D, capaz de coletar resíduos na superfície da água durante o trajeto que pode ser previamente definido ou controlado manualmente. Utilizando de tecnologias de simulação, o protótipo proposto pode ser testado, bem como o sistema de trajetória utilizado por ele. O que proporciona praticidade para quem deseje replicar o estudo com outro modelo proposto.

Capítulo 3

Definição

Neste capítulo, será apresentada a fundamentação para a descrição de um robô móvel para aplicações de controle utilizando um modelo de sistema de segunda ordem, onde são consideradas as restrições de movimento. Com isso, será apresentado ao final um modelo que permite a estimativa de posições e velocidades futuras de um robô a partir da aplicação de comandos de força.

3.1 Configuração do robô

Choset et al. (2005) define como configuração $\boldsymbol{q} \in \mathbb{Q}$ de um robô a especificação completa da posição de todos os pontos do sistema, onde $\mathbb{Q}$ é o conjunto de todas as possíveis configurações do robô. Desta forma, a quantidade mínima de parâmetros necessários para descrever a configuração do robô chama-se *graus de liberdade*. Assim, pode-se representar uma especificação completa do robô através de sua posição central, se o raio for conhecido.

Roland (2004) afirma que um robô precisa de mecanismos para que possa se mover num ambiente, onde são levadas em consideração características desse ambiente, como por exemplo, rigidez, atrito, etc. Quando comparadas as estratégias de locomoção (por articulações, e por rodas), a primeira, por sua vez, exige mais complexidade mecânica, do que a locomoção por rodas. Entretanto, ambas as opções podem ser capaz de trabalhar em um nível mais alto de assimilação, recebendo comandos de velocidade ou aceleração no sistema. Por se tratar de um robô aquático diferencial (isto é, sem leme para direcionar o movimento), pode-se afirmar que os fundamentos da movimentação do robô serão similares ao de um robô diferencial com rodas, tornando mais simples a modelagem do seu movimento.

3.2 Modelagem dinâmica de robôs

É sabido que, para que um objeto saia do repouso e inicie um movimento, é necessária a aplicação de uma força que seja suficiente para vencer as demais que o mantém em repouso (como o atrito, por exemplo). Com robôs, o princípio é o mesmo, e no caso dos robôs diferenciais, dependendo da força aplicada pelos motores, a trajetória pode ser bem diferente de uma simples linha reta.

A partir disso, considerando o exemplo de um robô diferencial com rodas, o movimento desse robô será feito a partir dessas rodas que irão girar em velocidades iguais ou diferentes, com isso o robô irá adquirir uma trajetória, podendo ser curva, em linha reta, ou até mesmo um giro no próprio eixo, entretanto, esse robô não poderá se mover lateralmente ou numa diagonal devido às restrições não holonômicas. Segundo Siciliano et al. (2009), uma restrição holonômica, é, normalmente o resultado de conexões mecânicas entre vários corpos do sistema, já uma restrição não holonômica, reduz a mobilidade do sistema mecânico. Ou seja, uma vez que os motores são fixos, os seus vetores de força gerada terão sua componente lateral iguais a zero como é ilustrado na Figura 3.1. Isso impossibilita que o robô deslize lateralmente, ou na diagonal.

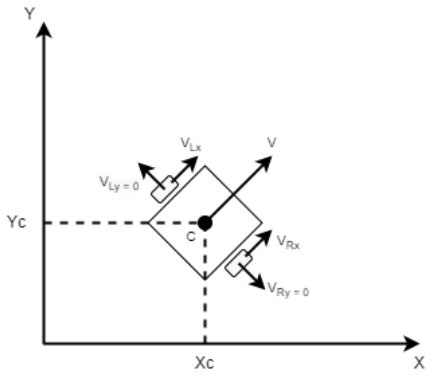


Figura 3.1: Ilustração de um robô diferencial com rodas

3.3 Modelo do barco diferencial

De acordo com Ahmad Abu Hatab (2013), a primeira etapa para a modelagem de um sistema de direção diferencial numa abordagem *Newton-Euler* é desenhar o diagrama de corpo livre, para descobrir quais as forças que atuam no robô, como é mostrado na figura 3.2(a).

Seguindo o método descrito em Ahmad Abu Hatab (2013), e considerando as restrições não-holonômicas de um robô diferencial, aplicando alguns ajustes, foram obtidas as equações:

$$\dot{x} = v\cos\theta$$

$$\dot{y} = v\sin\theta$$

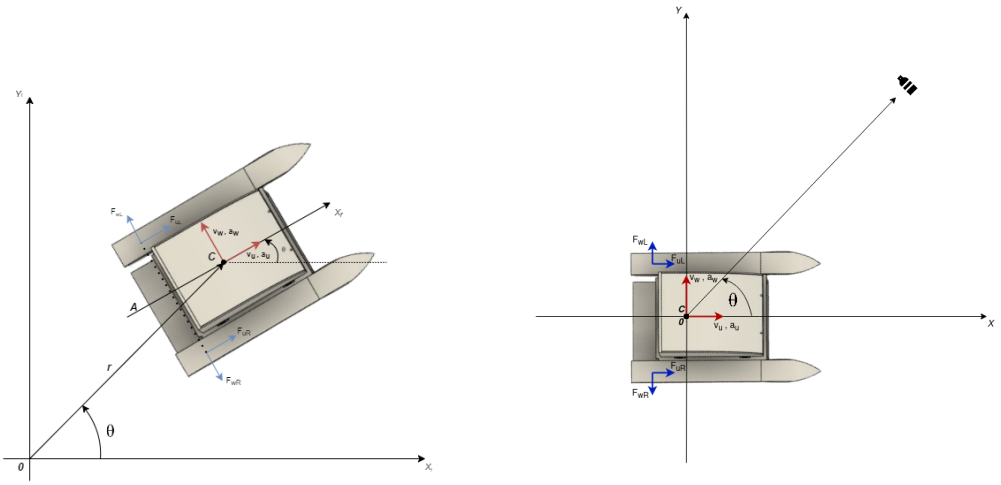
$$\dot{\theta} = w$$

$$\dot{v} = d \cdot w^2 + \frac{1}{M} \cdot (F_R + F_L)$$

$$\dot{w} = \frac{L}{M \cdot d^2 + J} \cdot (F_R - F_L) - \frac{M \cdot d}{M \cdot d^2 + J} \cdot v \cdot w$$

Onde M é a massa do robô, J é o momento de inercia em relação ao centro de massa, L, a distância do motor ao eixo central do robô, d é a distancia do eixo dos motores até o centro de massa, v e w são as velocidades linear e angular.

A partir do diagrama de corpo livre como referência, foi necessário adaptá-lo ao problema sendo modelado. Na figura 3.2(b), é exibido o diagrama de corpo livre final do modelo descrito pelas equações acima. Neste modelo, é considerado que o robô sempre estará na origem do sistema, facilitando assim a obtenção de uma trajetória até o objetivo.



(a) Diagrama de corpo livre de referência (b) Diagrama de corpo livre utilizado

Figura 3.2: Diagramas de corpo livre

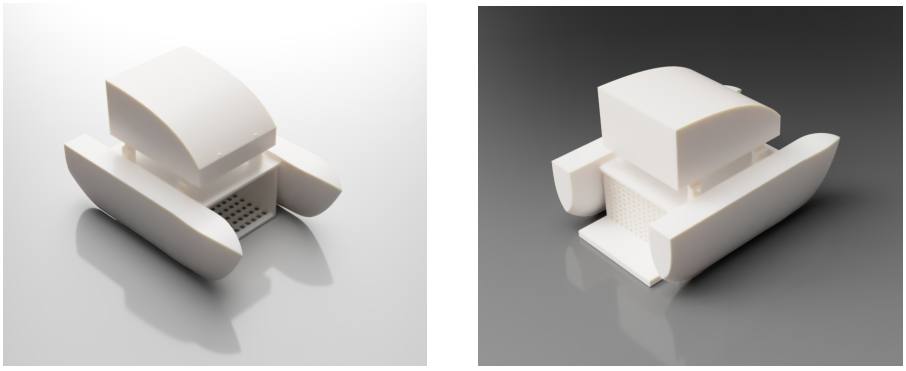
Capítulo 4

Framework Proposto

Primeiramente, é necessário ressaltar que uma simulação é diferente um robô real. O *framework* foi desenvolvido baseado numa simulação do Coppelia Robotics (2022), e, com isso, os comandos utilizados durante o processo, simulam a posição real e força que os motores poderiam aplicar.

4.1 O modelo

O modelo em 3D foi desenvolvido inteiramente no *software* AutoDesk Fusion360 Fusion360 (2022), de modo que todas as suas partes possam ser inspecionadas individualmente e montadas facilmente. O protótipo em escala real deve ter cerca de 01 metro de comprimento, 70 cm de largura e 65 cm de altura. Com esse tamanho, o robô deve ser capaz de coletar resíduos de tamanho considerável na superfície da água. O seu design foi inspirado em soluções já existentes, como por exemplo RanMarine (2022), que desenvolveu um produto comercial com alto custo voltado à coleta de lixo de forma autônoma, entretanto, o protótipo possui o diferencial de ser desenvolvido com a proposta de ser um robô autônomo de baixo custo.



(a) Vista da proa (b) Vista da popa

Figura 4.1: Design 3D do robô Squalo

Nas Figuras 4.1(a) e 4.1(b), é possível ver o protótipo proposto. Na proa, é possível ver uma plataforma, onde podem ser instalados diferentes sensores para coleta de dados. As baterias para o funcionamento do robô podem ser instaladas nas canoas, o hardware de controle na caixa no topo do robô, e os resíduos são coletados pela caixa vazada entre as canoas.

4.2 O *Framework*

Após o desenvolvimento do diagrama de corpo livre ilustrado na Figura 3.2(b), deu-se o inicio do desenvolvimento do *framework*, que por sua vez, tem 4 módulos (ilustrados na Figura 4.2). O primeiro módulo, o *Controller*, ou Controlador, recebe a posição atual do robô e a posição atual do objetivo, ressaltando que a posição do robô será sempre a origem, e portanto, a posição do objetivo é que varia conforme a simulação progride e as forças são aplicadas. A saída do Controlador são as forças a serem aplicadas em cada motor (direito e esquerdo).

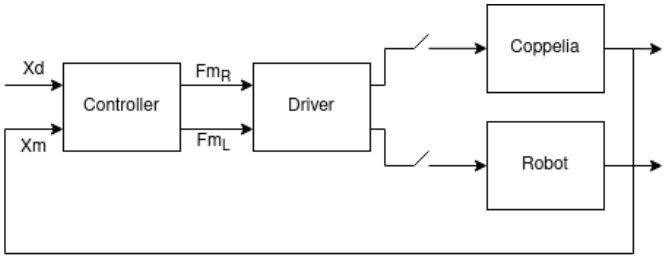


Figura 4.2: Diagrama de blocos do *framework*

Com essas forças calculadas, o Driver entra em ação, ele é que faz a comunicação entre o Controlador e a simulação, nele, os *scripts* fazem requisições à simulação obtendo a posição atual do objetivo, atual velocidade do robô, (através dos comandos $sim.getObjectPosition$ e $sim.getObjectVelocity$) entre outras informações importantes para o Controlador. É nele também que as forças são aplicadas na simulação, através de comandos como o $sim.addForce(obj, [x, y, z], [Fx, Fy, Fz])$ que é utilizado duas vezes, pois são dois motores, uma para o motor esquerdo, e uma para o motor direito. As forças são enviadas e aplicadas no robô, na posição estimada onde os motores ficariam. O primeiro parâmetro *obj* faz referência ao robô, enquanto que o segundo, $[x, y, z]$ refere-se ao local onde as forças serão aplicadas, nesse caso, os valores utilizados foram: [-0.3, -0.2, -0.1] e [-0.3, 0.2, -0.1] e por último, o vetor $[Fx, Fy, Fz]$ possui as forças a serem aplicadas nos motores, e, como foi discutido acima, apenas a componente Fx é utilizada, devido às restrições não-holonômicas.

Os últimos dois módulos são apenas os resultados das aplicações das forças, que podem ser a simulação como o *framework* foi desenvolvido, ou um robô real, com as devidas adaptações de código.

Capítulo 5

Estudo de Caso

Para validar o *framework* desenvolvido, foi implementado um controlador MPC para que os testes pudessem ocorrer. Neste capítulo, é dada uma breve definição do MPC, bem como a equação de otimização utilizada, um detalhamento acerca do ambiente de simulação, além da apresentação dos cenários utilizados nos testes para validação.

5.1 *Model Predictive Control (MPC)*

O *Model Predictive Control*, ou apenas MPC, é um modelo de controlador, onde, com o uso de um modelo, o comportamento do sistema pode ser previsto dentro de um horizonte máximo. Em outras palavras, dado o momento k, o modelo é capaz de simular a ordem de estados $k + p$, onde p é o horizonte máximo de predição. Uma de suas vantagens é a facilidade de implementação em sistemas MIMO *(multiple input, multiple output)*, que são os tipos mais comuns na robótica, bem como o caso do *framework* proposto.

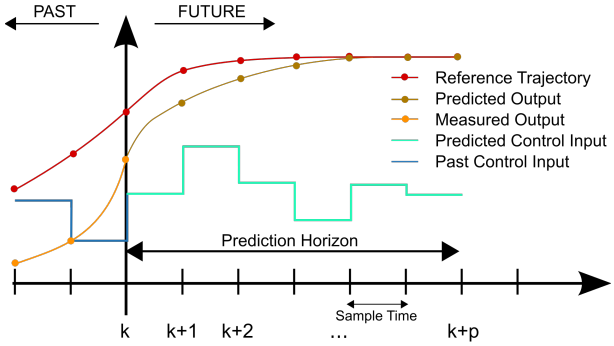


Figura 5.1: Ilustração do MPC

Existem alguns tipos de MPC, como, linear e não-linear (que é o tipo do *framework* proposto). Para a implementação do MPC, foi utilizado a biblioteca do-mpc Do-mpc (2022), que é uma biblioteca na linguagem Python que facilita o desenvolvimento do MPC, simplificando os passos necessários durante o processo.

A equação de otimização utilizada no *framework* está logo abaixo, onde suas restrições não-holonômicas estão sendo consideradas como citado no capítulo anterior.

$$\sum_{i=0}^{N}(\dot{x}(k+i)-\dot{x}_d(k+i))^2+(\dot{y}(k+i)-\dot{y}_d(k+i))^2+ \\ 70\cdot(\dot{\theta}(k+i)-\dot{\theta}_d(k+i))^2+(\dot{v}(k+i)-\dot{v}_d(k+i))^2+ \\ (\dot{w}(k+i)-\dot{w}_d(k+i))^2 \quad (5.1)$$

Após muitas etapas de teste e otimização de parâmetros, foi necessário priorizar a trajetória até o objetivo, aplicando um ganho no ângulo entre o robô e o seu objetivo, em detrimento das demais características, como as velocidades linear e angular, por exemplo. O valor encontrado para aplicação de ganho nesse parâmetro (θ) foi de 70. Com esse ajuste, o robô foi capaz de chegar ao objetivo com trajetórias simples na maioria dos cenários testados. Entretanto, em alguns casos, devido à inércia, o robô finalizava o arco da trajetória um pouco antes do objetivo, obrigando-o a executar mais ações para chegar ao final do percurso, como será ilustrado nos cenários mais à frente.

5.2 Simulação

Esta seção trata do ambiente de simulação utilizado para teste do *framework*, descrevendo a simulação da superfície e do veículo. Na Figura 5.2, é ilustrado um dos cenários para que se possa dar uma visualização geral do ambiente de simulação.

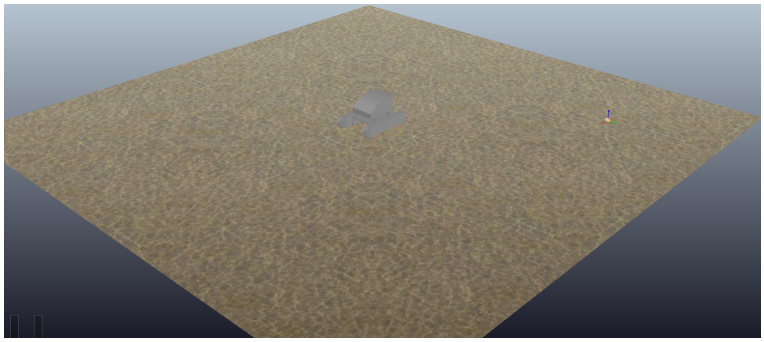


Figura 5.2: Ambiente de simulação

5.2.1 *Water Simulation*

Não existe no Coppelia um objeto que simule a água de fato, como presente em softwares de simulação fluídica (*Computational Fluid Dynamics*), seja a nível de partícula ou de volume de controle. No entanto, o simulador permite estabelecer, de forma algorítmica, forças atuantes para imitar o comportamento de flutuação de objetos em um fluido.

O objeto *Water Simulation* foi obtido e adaptado do projeto disponível no Github VREP-Boat-Simulation O'Meadhra (2015), onde foi desenvolvido um *script* de forma que o chão da simulação do Coppelia Robotics (2022) se torna "água", simulando sua física. Esse objeto é capaz de detectar objetos declarados como barcos na hierarquia da cena utilizada, os divide em elementos de volume, recebendo suas propriedades básicas (como geometria, volume, etc), calcula o coeficiente de arrasto para aquele robô, e por último, calcula, baseado nas propriedades do robô, as forças de arrasto e de flutuação para aplicação enquanto o robô está se movendo.

5.2.2 Simulação do robô

Para simular o robô, o seu modelo em 3D foi desenvolvido (exibido nas imagens 4.1) utilizando o *software* Autodesk Fusion360 Fusion360 (2022). O modelo foi desenvolvido para ter em torno de 01 metro de comprimento, 70 cm de largura e 65 cm de altura. Para sua simulação, um modelo STL foi gerado dentro do *software*, e inserido na *Water Simulation*, citada anteriormente.

Para simulação dos motores, o comando $sim.addForce$ foi usado usado na através da API ZeroMQ ZeroMQ (2022). Para aplicação das forças, foi necessário o uso de posições específicas, pois deveriam simular, aproximadamente, o local onde os motores ficariam no robô, os valores foram apresentados na Seção 4.2.

5.3 Cenários

Esta seção ilustra alguns dos cenários utilizados para os testes do *framework*, foram selecionados três deles mais genéricos, que ilustram os demais cenários testados, já que seriam apenas variações.

5.3.1 Cenário 01

O primeiro cenário testado é uma situação básica, o objeto está posicionado no ponto (3,0) do espaço, mostrado na Figura 5.3. Este é o caso mais simples, o robô deve apenas andar em linha reta até o objetivo. O vídeo de sua simulação pode ser encontrado aqui.

5.3.2 Cenário 02

O segundo, é um pouco mais complicado de executar, o objeto está na diagonal do robô, posicionado em (3, -3), mostrado na Figura 5.4. Nesse caso, o robô deve performar um arco à sua frente até o objetivo, uma vez que não é possível se mover em diagonal devido às restrições não-holonômicas. O vídeo de sua simulação pode ser encontrado aqui.

5.3.3 Cenário 03

Por último, o cenário mais difícil, onde o objetivo está posicionado em (-3, 3), mostrado na Figure 5.5. Neste caso, é esperada uma trajetória similar à letra U, onde o robô iniciará o movimento já com uma curva acentuada em direção ao objetivo. O vídeo de sua simulação pode ser encontrado aqui.

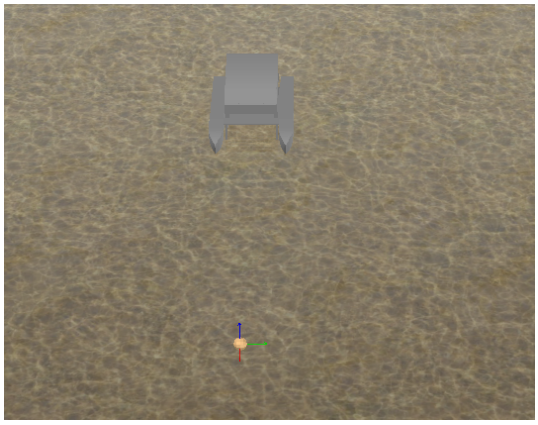


Figura 5.3: Ponto inicial do cenário 01

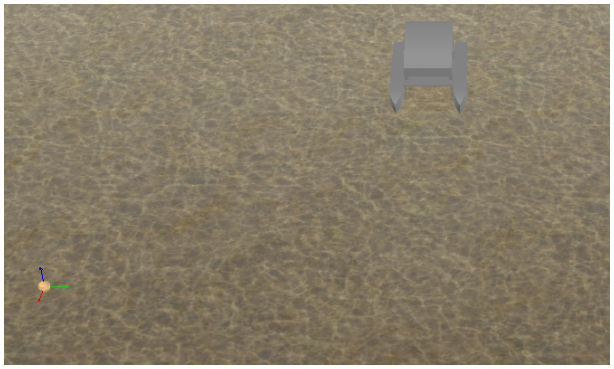


Figura 5.4: Ponto inicial do cenário 02

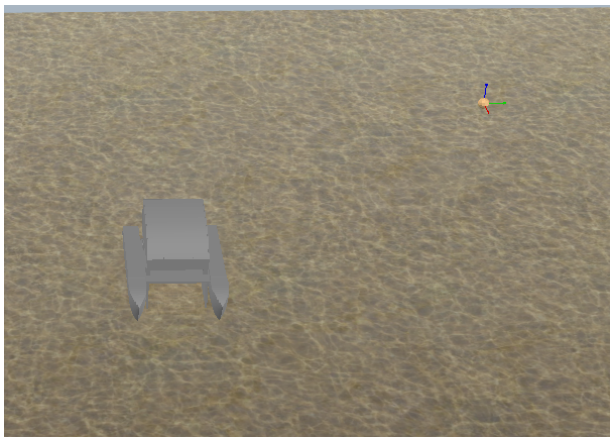


Figura 5.5: Ponto inicial do cenário 03

5.4 Métricas de desempenho

Para avaliação de técnicas desenvolvidas usando o *framework*, foram implementadas chamadas para alguns índices de desempenho usados na avaliação de sistemas de controle em malha fechada. Foram eles: IAE, ISE e ITAE. Com os índices citados, qualquer pessoa ao tentar reproduzir o *framework* poderá realizar análises e ajustes nos parâmetros, facilitando a sintonia do controle.

Nesses índices, *e(t)* representa a diferença entre o valor medido no robô, e o valor de referência esperado em cada instante *t*. Mario Cesar M. Massa de Campos (2010)

O IAE (Integral do módulo do erro) apresenta o acúmulo de erro no período de tempo. Seu cálculo, definido em Mario Cesar M. Massa de Campos (2010), está logo abaixo:

$$\int |e(t)|dt$$

O ISE (Integral dos erros ao quadrado) apresenta a média do sinal do erro no período de tempo, apresentando um peso maior a erros com valores maiores que 1, consequentemente apresentando peso menor em erros inferiores a esse valor. Pode ser calculado através da equação definida em Mario Cesar M. Massa de Campos (2010):

$$\int e^2(t)dt$$

No ITAE (Integral do módulo do erro vezes o tempo) é apresentado o valor do erro em relação ao tempo, entretanto, nele, são minimizados os erros iniciais, dando maior importância aos erros de estado estacionário. Apresenta uma equação um pouco mais elaborada, definida em Mario Cesar M. Massa de Campos (2010):

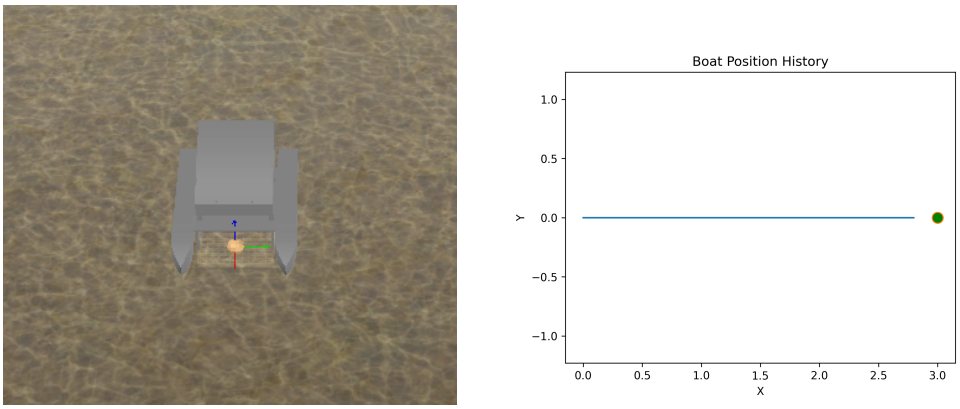
$$\int t \cdot |e(t)|dt$$

Capítulo 6

Resultados

Neste capítulo, serão discutidos os resultados obtidos na seção 5.2. Os cenários utilizados apenas ilustram alguns dos muitos que foram utilizados durante os testes do *framework*. Neles, o objetivo foi colocado de maneira que os outros casos fossem apenas variações, cobrindo de maneira genérica os casos de objetos à frente, diagonal e atrás do robô.

Começando pelo Cenário 01 (5.3.1), como ilustrado pela Figura 6.1, o robô não apresentou dificuldades, e seguiu o comportamento esperado. Partindo da posição (0, 0), o robô foi capaz de andar em linha reta até seu objetivo (ponto verde representado na Figura 6.1(b)).

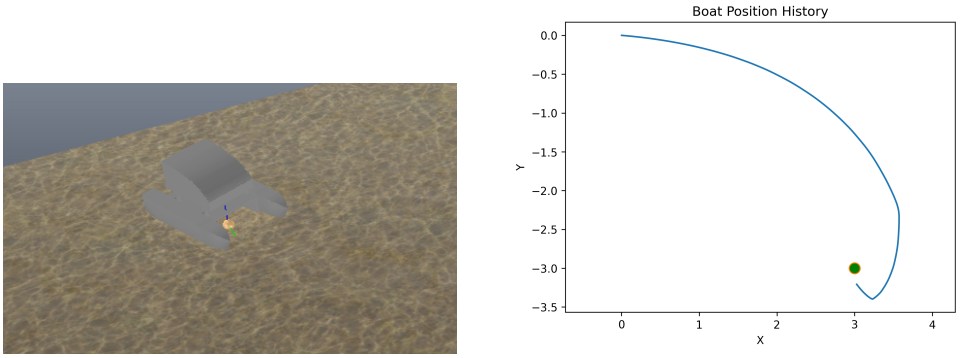


(a) Posição final (b) Trajetória apresentada

Figura 6.1: Resultado do Cenário 01

No segundo cenário (5.3.2), era esperado um arco à frente do robô, entretanto, foi apresentada uma certa dificuldade em controlar a inércia do movimento, errando o objetivo por muito pouco. Mesmo assim, o robô foi capaz de ajustar sua trajetória, executando um giro, e chegando ao objetivo corretamente. Sua posição final, e sua trajetória são mostradas na Figura 6.2.

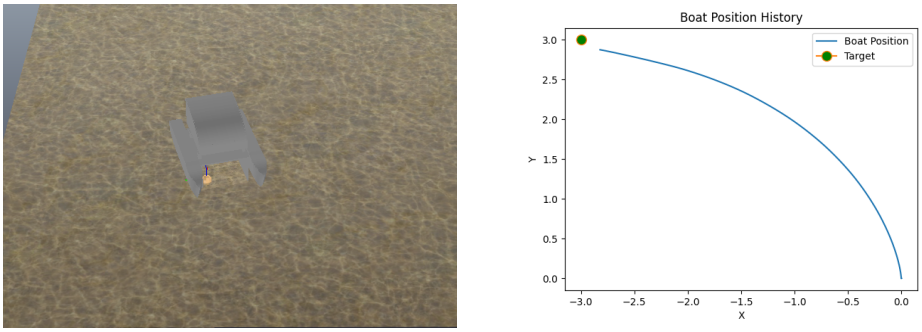
Por último, o cenário 03 (5.3.3), mais complexo, onde o robô deveria performar uma trajetória simular a um U. Neste cenário, o robô foi capaz de executar um giro no próprio eixo, até o ponto onde foi capaz de acelerar em um arco até o objetivo. O que deveria



(a) Posição final (b) Trajetória apresentada

Figura 6.2: Resultado do Cenário 02

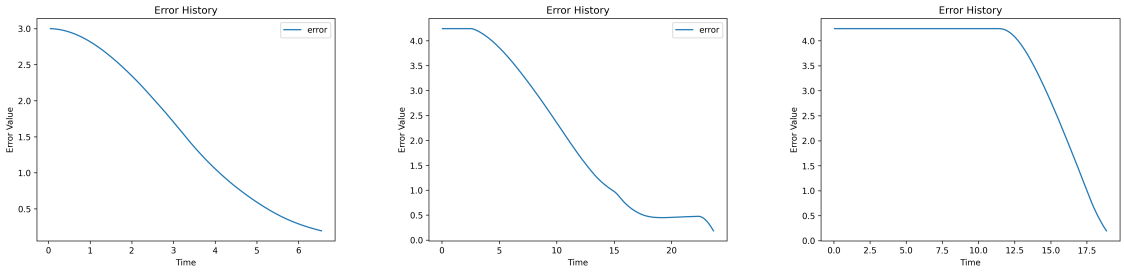
ser o caso mais complexo de ser performado, foi um dos melhores resultados. A trajetória pode ser vista na Figura 6.3.



(a) Posição final (b) Trajetória apresentada

Figura 6.3: Resultado do Cenário 03

Abaixo, temos os gráficos de erro apresentado pelos 3 cenários:



(a) Cenário 01 (b) Cenário 02 (c) Cenário 03

Figura 6.4: Gráficos de erro nos 3 cenários

Como pode ser visto na Figura 6.4(a), no primeiro cenário, não foi apresentada muita dificuldade para o sistema, iniciando uma aceleração, com um ponto de inflexão ao iniciar a desaceleração do robô.

Cenários	IAE	ISE	ITAE
Cenário 01	1.544	3.831	10.165
Cenário 02	2.065	6.461	14.537
Cenário 03	3.563	14.081	28.521

Tabela 6.1: Tabela de métricas

Na Figura 6.4(b), podemos ver que o sistema apresentou uma certa dificuldade bem próximo do objetivo, que pode ser vista no gráfico da trajetória apresentada na Figura 6.2(b), onde o robô acaba passando um pouco do objetivo e efetua uma curva, retornando e alcançando-o.

Por último, temos o gráfico do erro do cenário 3, na Figura 6.4(c), onde é possível ver que boa parte do tempo, o erro se mantém alto, (durante a curva no próprio eixo), e então há uma queda brusca, chegando ao objetivo.

Na Tabela 6.1, são apresentados as métricas IAE, ISE e ITAE, onde é possível notar que, nos cenários 01 e 02, os valores apresentados foram relativamente baixos em todos os índices, entretanto, no cenário 03, os índices apresentaram valores acima, justificados pelo tempo que o robô levou girando no próprio eixo, com a intenção de rotacionar em direção ao objetivo.

Capítulo 7

Conclusão

Nesse trabalho, foi apresentado Squalo, um *framework* autônomo para coleta de lixo na superfície da água. O protótipo foi proposto para ser de fácil reprodução com custo relativamente baixo e capaz de ser impresso em impressoras 3D.

O *framework* desenvolvido teve bons resultados. A simulação base utilizada para desenvolvimento do framework não foi desenvolvida para ser utilizada com um controlador robusto como o MPC, mesmo assim, o *framework*, com equações que não consideram a flutuação na água, apenas o seu deslocamento, foi capaz de não apenas flutuar, mas de ir até os objetivos designados, mesmo com pertubações não modeladas, como demonstrado no Cenário 02 (5.3.2), onde o robô acabou afetado pela inércia, mas se recuperou e foi até o objetivo.

Em relação às limitações, primeiro, é necessário que a relação entre a força do motor e a tensão seja diretamente convertida durante um ciclo interno de controle para assegurar essa conversão. Surgiram incertezas relativas à inércia do robô, uma vez que sua geometria foi aproximada para uma forma mais simples. Também, devido ao foco no *framework*, e não no controle, nenhum estudo mais profundo foi adiante durante o ajuste dos parâmetros utilizados para assegurar a melhor performance e estabilidade possível. Entretanto, considerando os resultados das simulações, podemos concluir que Squalo é capaz de coletar resíduos na superfície da água de forma autônoma, com um sistema de baixo custo, e com um *framework* capaz de ser adaptado para outros modelos de robôs.

Como trabalhos futuros, uma vez que o objetivo do robô deve ser estabelecido primariamente, técnicas de detecção de resíduos na água devem ser estudadas para detecção e computação automática do objetivo, incluindo testes de trajetórias envolvendo padrões de busca e patrulhamento, além de testes em um robô real.

Os códigos, modelos e simulações estão disponíveis no repositório do GitHub.

Bibliografia

Ahmad Abu Hatab, R. D. (2013). Dynamic modelling of differential-drive mobile robots using lagrange and newton-euler methodologies: A unified framework.

Bibuli, M., Caccia, M., Lapierre, L., and Bruzzone, G. (2012). Guidance of unmanned surface vehicles: Experiments in vehicle following. *IEEE Robotics Automation Magazine*, 19(3):92–102.

Caccia, M., Bibuli, M., Bruzzone, G., Bruzzone, G., Bono, R., and Spirandelli, E. (2009). Charlie, a testbed for usv research. *IFAC Proceedings Volumes*, 42(18):97–102. 8th IFAC Conference on Manoeuvring and Control of Marine Craft.

Chang, H.-C., Hsu, Y.-L., Hung, S.-S., Ou, G.-R., Wu, J.-R., and Hsu, C. (2021a). Autonomous water quality monitoring and water surface cleaning for unmanned surface vehicle. *Sensors*, 21(4).

Chang, H.-C., Hsu, Y.-L., Hung, S.-S., Ou, G.-R., Wu, J.-R., and Hsu, C. (2021b). Autonomous water quality monitoring and water surface cleaning for unmanned surface vehicle. *Sensors*, 21(4):1102.

Choset, H., Lynch, K. M., Hutchinson, S., Kantor, G. A., and Burgard, W. (2005). *Principles of Robot Motion: Theory, Algorithms, and Implementations*. MIT Press.

Do-mpc (2022). https://www.do-mpc.com/.

Dunbabin, M., Grinham, A., and Udy, J. (2009). An autonomous surface vehicle for water quality monitoring. *Proceedings of the 2009 Australasian Conference on Robotics and Automation, ACRA 2009*, 13.

Fusion360, A. (2022). https://www.autodesk.com/products/fusion-360/.

Jambeck, J. R., Geyer, R., Wilcox, C., Siegler, T. R., Perryman, M., Andrady, A., Narayan, R., and Law, K. L. (2015). Plastic waste inputs from land into the ocean. *Science*, 347(6223):768–771.

Lebreton, L. C., van der Zwet, J., Damsteeg, J.-W., Slat, B., Andrady, A., and Reisser, J. (2017). River plastic emissions to the world's oceans. *Nature Communications*, 8(1).

Mario Cesar M. Massa de Campos, H. C. G. T. (2010). Controles típicos de equipamentos e processos industriais.

Ministério do Meio Ambiente (2019). Plano nacional de combate ao lixo no mar. https://www.gov.br/mma/pt-br/assuntos/agendaambientalurbana/combate-ao-lixo-no-mar.

O'Meadhra, C. (2015). https://github.com/comeadhra/VREP-Boat-Simulator.

Paravisi, M., H. Santos, D., Jorge, V., Heck, G., Gonçalves, L. M., and Amory, A. (2019a). Unmanned surface vehicle simulator with realistic environmental disturbances. *Sensors*, 19(5).

Paravisi, M., Santos, D., Jorge, V., Heck, G., Gonçalves, L., and Amory, A. (2019b). Unmanned surface vehicle simulator with realistic environmental disturbances. *Sensors*, 19:1068.

RanMarine (2022). https://www.ranmarine.io/.

Robotics, C. (2022). https://www.coppeliarobotics.com/.

Roland, S. (2004). *Introduction to Autonomous Mobile Robots.* MIT Press.

Sehgal, A. and Cernea, D. (2010a). A multi-auv missions simulation framework for the usarsim robotics simulator. In *Proceedings of the 18th Mediterranean Conference on Control and Automation (MED'10).*

Sehgal, A. and Cernea, D. (2010b). A multi-auv missions simulation framework for the usarsim robotics simulator. *18th Mediterranean Conference on Control and Automation, MED'10*, pages 1188–1193.

Siciliano, B., Sciavicco, L., Villani, L., and Oriolo, G. (2009). *Robotics, Modelling, Planning and Control*. Springer.

Svec, P., Thakur, A., Shah, B. C., and Gupta, S. K. (2012). Usv trajectory planning for time varying motion goals in an environment with obstacles. Volume 4: 36th Mechanisms and Robotics Conference, Parts A and B:1297–1306.

ZeroMQ, C. R. (2022). https://www.coppeliarobotics.com/helpFiles/en/zmqRemoteApiOverview.htm.