

UNIVERSIDADE FEDERAL DE SANTA MARIA  
CENTRO DE TECNOLOGIA  
CURSO DE GRADUAÇÃO EM ENGENHARIA ELÉTRICA

Lucas Sordi Rambo

**CONSTRUÇÃO E ANÁLISE EXPERIMENTAL DE UM BRAÇO  
ROBÓTICO DE CINCO GRAUS DE LIBERDADE**

Santa Maria, RS  
2024

Lucas Sordi Rambo

## **CONSTRUÇÃO E ANÁLISE EXPERIMENTAL DE UM BRAÇO ROBÓTICO DE CINCO GRAUS DE LIBERDADE**

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Engenharia Elétrica da Universidade Federal de Santa Maria (UFSM, RS), como requisito parcial para obtenção do grau de **Engenheiro Eletricista**. Defesa realizada por videoconferência.

Orientador: Prof. Anselmo Rafael Cukla

Santa Maria, RS  
2024

**Lucas Sordi Rambo**

**CONSTRUÇÃO E ANÁLISE EXPERIMENTAL DE UM BRAÇO ROBÓTICO DE CINCO  
GRAUS DE LIBERDADE**

Trabalho de Conclusão de Curso apresentado ao  
Curso de Graduação em Engenharia Elétrica da  
Universidade Federal de Santa Maria (UFSM, RS),  
como requisito parcial para obtenção do grau de  
**Engenheiro Eletricista.**

**Aprovado em 16 de janeiro de 2024:**

---

**Anselmo Rafael Cukla, Dr. Eng. (UFSM)**  
**(Presidente/Orientador)**

---

**Daniel Fernando Tello Gamarra, Dr. (UFSM) (videoconferência)**

---

**Leonardo Ramos Emmendorfer, Dr. (UFSM) (videoconferência)**

Santa Maria, RS  
2024

## AGRADECIMENTOS

O apoio e a orientação que recebi durante a realização desse projeto foram fundamentais para aprimorar meu desenvolvimento profissional e acadêmico.

Agradeço à Universidade Federal de Santa Maria (UFSM), ao curso de Engenharia Elétrica e, em especial, ao meu orientador Prof. Dr. Anselmo Rafael Cukla, por me oferecerem um ambiente enriquecedor, que contribuiu para a formação de habilidades técnicas, capacidade de aprender a aprender e construção de resiliência.

Agradeço ao grupo TauraBots pela disponibilização de recursos e materiais para a construção do protótipo, assim como pela equipe sempre disponível para auxiliar. Agradeço ao Lucas Strapazzon pelo trabalho conjunto no desenvolvimento do projeto mecânico.

Agradeço ao Dr. Adriano Longo, da Fábrica CT, pelo apoio nas etapas de impressão 3D e construção do robô.

Agradeço ao meu amigo Igor Dal Forno pela ajuda na implementação dos componentes eletrônicos.

Agradeço aos meus pais Maurício Rambo e Teresinha Vitória Sordi Rambo pelo apoio incondicional, pelos valores ensinados e por estarem sempre disponíveis.

Agradeço à minha namorada Maria Laura Martins Silva pelo amor e paciência durante esse período. Tua paixão e comprometimento incentivam-me a alcançar a excelência.

Com apreço e reconhecimento,

Lucas Sordi Rambo.

## RESUMO

### CONSTRUÇÃO E ANÁLISE EXPERIMENTAL DE UM BRAÇO ROBÓTICO DE CINCO GRAUS DE LIBERDADE

AUTOR: Lucas Sordi Rambo  
Orientador: Anselmo Rafael Cukla

Braços robóticos possuem amplo uso na indústria e consolidado estudo na literatura científica. Existem ferramentas *open source* capazes de lidar com a simulação e execução de movimentos nos robôs físicos, é o caso do ROS, Gazebo, MoveIt, entre outros. Neste trabalho, é analisada a estrutura e o funcionamento de um braço robótico comercial, o Viper X 300, da *Interbotix*, para servir como base na construção de um modelo similar. Esse robô possui capacidades de programação avançada que podem ser adaptadas para utilização por outros modelos de robôs. O objetivo da pesquisa foi construir um robô manipulador utilizando técnicas de manufatura aditiva e atuadores DYNAMIXEL, validando a sua cinemática. Para isso, foram realizadas as configurações de *software* e *hardware* adequadas. Além disso, foi desenvolvida uma análise experimental da garra do efetuador, juntamente com a avaliação da execução dos movimentos programados no robô físico.

**Palavras-chave:** Robô manipulador. Cinemática direta. Manufatura aditiva. Servomotores DYNAMIXEL.

## ABSTRACT

### CONSTRUCTION AND EXPERIMENTAL ANALYSIS OF A 5-DOF ROBOT ARM

AUTHOR: Lucas Sordi Rambo  
ADVISOR: Anselmo Rafael Cukla

Robotic arms are widely used in industry and have been extensively studied in scientific literature. There are open source tools capable of handling the simulation and execution of movements in physical robots, such as ROS, Gazebo, MoveIt, among others. This paper analyzes the structure and operation of a commercial robotic arm, the Viper X 300, from Interbotix, to serve as a basis for building a similar model. This robot has advanced programming capabilities that can be adapted for use by other robot models. The aim of the research was to build a manipulator robot using additive manufacturing techniques and DYNAMIXEL actuators, validating its kinematics. To do this, the appropriate software and hardware configurations were made. In addition, an experimental analysis of the effector's gripper was carried out, along with an evaluation of the execution of the programmed movements on the physical robot.

**Keywords:** Robot manipulator. Forward kinematics. Additive manufacturing. DYNAMIXEL smart actuator.

## LISTA DE FIGURAS

Figura 1 – Representações dos tipos de juntas. ....	17
Figura 2 – Manipulador do tipo SCARA. ....	17
Figura 3 – Sistema de coordenadas com eixos de posicionamento e ângulos de orientação. ....	18
Figura 4 – Componentes de um robô industrial de seis graus de liberdade. ....	19
Figura 5 – Sistema de controle tradicional. ....	20
Figura 6 – Exemplificação da técnica de Fabricação por Filamento Fundido. ....	25
Figura 7 – Ensaio de tração para os materiais PLA, ABS e PETG. ....	26
Figura 8 – Robô <i>open source</i> da <i>BCN3D</i> . ....	27
Figura 9 – Robô <i>Dobot Magician</i> da fabricante <i>Dobot</i> . ....	28
Figura 10 – Metodologia. ....	31
Figura 11 – Braço robótico Viper X 300. ....	33
Figura 12 – Modelo CAD da base do robô. ....	35
Figura 13 – Mecanismo giratório da base. ....	36
Figura 14 – Modelo CAD do ombro do robô. ....	37
Figura 15 – Modelo CAD do braço e antebraço do robô. ....	37
Figura 16 – Modelo CAD do cotovelo do robô. ....	38
Figura 17 – Modelo CAD do punho do robô. ....	38
Figura 18 – Modelo CAD do garra do robô. ....	39
Figura 19 – Vista explodida da garra. ....	39
Figura 20 – Processo de impressão 3D. ....	40
Figura 21 – Visualização da tela inicial do Cura. ....	41
Figura 22 – Visualização do fatiamento realizado no Cura. ....	41
Figura 23 – Identificação dos principais componentes da Creality 3D CR-10S V2. ....	42
Figura 24 – Identificação do <i>horn</i> e da comunicação serial no motor. ....	45
Figura 25 – Identificação do ID de cada motor no projeto CAD. ....	47
Figura 26 – Placa OpenCR e apontamento dos componentes mais relevantes para o projeto. ....	48
Figura 27 – Pinagem da comunicação RS-485. ....	49
Figura 28 – Esquema de ligação em cadeia dos motores Dynamixel. ....	49
Figura 29 – Esquema de conexão dos componentes. ....	50
Figura 30 – Interface do DYNAMIXEL Wizard 2.0. ....	51
Figura 31 – Interface do DYNAMIXEL SDK utilizado pelo Arduino IDE. ....	52
Figura 32 – Arquitetura de <i>software</i> para manipuladores da <i>Interbotix</i> (IRROS). ....	55
Figura 33 – Dispositivo de aquisição no ensaio de aperto máximo ....	57
Figura 34 – Calibração da célula de carga com pesos padrões. ....	58

Figura 35 – Configuração do diagrama cinemático. ....	59
Figura 36 – Planejamento de trajetória utilizando Movelt. ....	75
Figura 37 – Execução de trajetória utilizando Movelt. ....	75
Figura 38 – Modelo CAD da estrutura completa do robô. ....	76
Figura 39 – Placa de alimentação. ....	77
Figura 40 – Montagem dos componentes elétricos do manipulador. ....	77
Figura 41 – Registro do processo de montagem da garra. ....	78
Figura 42 – Estrutura do manipulador totalmente montada. ....	79
Figura 43 – Comparativo em simulação do robô construído e do Viper X 300. ....	80
Figura 44 – Simulação de movimentos com ambos os robôs. ....	81
Figura 45 – Simulação de agarre e movimentação de objeto com o robô construído. .	82
Figura 46 – Comparação para a posição <i>home</i> na interface do Movelt e no robô físico. 83	
Figura 47 – Execução de uma sequência de movimentos com o robô físico. ....	84
Figura 48 – Execução de movimento para pega de objeto. ....	85
Figura 49 – Garra realizando pega de uma maçã. ....	85
Figura 50 – Garra realizando pega de uma caneta. ....	86
Figura 51 – Garra realizando pega de um copo de vidro. ....	86
Figura 52 – Aquisição de dados do fechamento da garra. ....	87
Figura 53 – Resposta da célula de carga de acordo com o fechamento e reabertura da garra. ....	87



## LISTA DE TABELAS

TABELA 1 – Relação dos parâmetros de Denavit-Hartenberg. ....	22
TABELA 2 – Especificações do robô <i>Moveo</i> . ....	28
TABELA 3 – Especificações do robô <i>Dobot Magician</i> . ....	28
TABELA 4 – Especificações do robô Viper X 300. ....	33
TABELA 5 – Especificações do servomotor DYNAMIXEL XM540-W270-T. ....	34
TABELA 6 – Especificações do servomotor DYNAMIXEL XM430-W350-T. ....	34
TABELA 7 – Especificações da impressora Creality 3D CR-10S V2. ....	42
TABELA 8 – Especificações dos parâmetros de impressão adotados. ....	43
TABELA 9 – Especificações do servomotor DYNAMIXEL MX-106R. ....	45
TABELA 10 – Especificações do servomotor DYNAMIXEL MX-64R. ....	46
TABELA 11 – Relação dos motores utilizados e respectivos IDs. ....	46
TABELA 12 – Parâmetros de Denavit-Hartenberg para o Viper X 300. ....	60
TABELA 13 – Equações da posição do efetuador final, de acordo com Denavit-Hartenberg para o Viper X 300. ....	62
TABELA 14 – Parâmetros de Denavit-Hartenberg para o robô construído. ....	63
TABELA 15 – Equações da posição do efetuador final, de acordo com Denavit-Hartenberg para o robô construído. ....	64
TABELA 16 – Comparativo de dimensões do Viper X 300 e do robô construído. ....	87
TABELA 17 – Comparativo de especificações do Viper X 300 e do robô construído. ...	88

## LISTA DE ABREVIATURAS

CAD	<i>Computer-aided design</i>
STL	<i>STeroLithography</i>
ABS	Acrilonitrila butadieno estireno
PLA	Poliácido láctico
PETG	Tereftalato de Polietileno Glicol Modificado
TPU	Poliuretano Termoplástico
ROS	<i>Robot Operating System</i>
FFF	Fabricação por Filamento Fundido
DH	Denavit-Hartenberg
MCU	Microcontrolador
ID	Identificador
PWM	Modulação por Largura de Pulso
IDE	Ambiente de Desenvolvimento Integrado
SDK	Kit de Desenvolvimento de Software
API	Interface de Programação de Aplicações

## SUMÁRIO

<b>1</b>	<b>INTRODUÇÃO</b>	<b>12</b>
1.1	MOTIVAÇÃO	13
1.2	OBJETIVOS DO ESTUDO	13
1.3	DESCRIÇÃO DO PROBLEMA	14
1.4	PRINCIPAIS CONTRIBUIÇÕES	14
1.5	ORGANIZAÇÃO DO TRABALHO	15
<b>2</b>	<b>REVISÃO BIBLIOGRÁFICA</b>	<b>16</b>
2.1	FUNDAMENTAÇÃO TEÓRICA	16
<b>2.1.1</b>	<b>Manipuladores</b>	<b>16</b>
<b>2.1.2</b>	<b>Juntas e Elos</b>	<b>16</b>
<b>2.1.3</b>	<b>Graus de Liberdade</b>	<b>17</b>
2.2	CINEMÁTICA DIRETA	19
<b>2.2.1</b>	<b>Sistema de Referência</b>	<b>20</b>
<b>2.2.2</b>	<b>Definição dos Parâmetros</b>	<b>21</b>
<b>2.2.3</b>	<b>Procedimentos</b>	<b>22</b>
2.3	EFETUADORES	23
2.4	MANUFATURA ADITIVA	23
2.5	ATUADORES	26
2.6	ESTADO DA ARTE	27
<b>2.6.1</b>	<b>Robôs comerciais e de código aberto</b>	<b>27</b>
<b>2.6.2</b>	<b>Construção de manipuladores na literatura acadêmica</b>	<b>29</b>
<b>3</b>	<b>MATERIAIS E MÉTODOS</b>	<b>31</b>
3.1	METODOLOGIA	31
3.2	DESCRIÇÃO DO VIPER X 300	33
3.3	CONCEPÇÃO E PROJETO MECÂNICO	35
<b>3.3.1</b>	<b>Base</b>	<b>35</b>
<b>3.3.2</b>	<b>Ombro</b>	<b>36</b>
<b>3.3.3</b>	<b>Braço e Antebraço</b>	<b>36</b>
<b>3.3.4</b>	<b>Cotovelo</b>	<b>37</b>
<b>3.3.5</b>	<b>Punho</b>	<b>38</b>
<b>3.3.6</b>	<b>Garra</b>	<b>38</b>
3.4	PROCEDIMENTOS DE IMPRESSÃO 3D	40
3.5	MOTORES DYNAMIXEL	44
3.6	MICROCONTROLADOR OPENCN	47
3.7	MALHA ELÉTRICA	48

3.8	COMPONENTES DE SOFTWARE.....	50
3.8.1	<b>DYNAMIXEL Wizard 2.0</b> .....	<b>50</b>
3.8.2	<b>DYNAMIXEL SDK</b> .....	<b>51</b>
3.8.3	<b>ROS</b> .....	<b>52</b>
3.8.4	<b>Movelt</b> .....	<b>54</b>
3.8.5	<b>Arquitetura de <i>software</i> para manipuladores da <i>Interbotix</i></b> .....	<b>54</b>
3.9	BANCADA EXPERIMENTAL.....	56
<b>4</b>	<b>MECANISMOS DE CONTROLE</b> .....	<b>59</b>
4.1	DIAGRAMA CINEMÁTICO.....	59
4.2	PARÂMETROS DH PARA VIPER X 300.....	60
4.3	PARÂMETROS DH PARA O ROBÔ CONSTRUÍDO.....	62
4.4	CINEMÁTICA VIA PRODUTO DE EXPONENCIAIS.....	64
4.5	SISTEMA DE CONTROLE E OPERAÇÃO DOS MOTORES.....	67
4.6	PROGRAMAÇÃO.....	69
4.6.1	<b>Configuração dos motores</b> .....	<b>69</b>
4.6.2	<b>URDF</b> .....	<b>70</b>
4.6.3	<b>Controladores de posição e trajetória</b> .....	<b>72</b>
4.6.4	<b>SRDF</b> .....	<b>72</b>
4.6.5	<b>Matrizes Cinemáticas</b> .....	<b>73</b>
4.7	GERAÇÃO DE TRAJETÓRIAS.....	73
<b>5</b>	<b>RESULTADOS E DISCUSSÃO</b> .....	<b>76</b>
5.1	PROJETO CAD COMPLETO.....	76
5.2	MONTAGEM DA PARTE ELÉTRICA.....	76
5.3	MONTAGEM MECÂNICA.....	78
5.4	SIMULAÇÕES.....	78
5.5	EXECUÇÃO DOS MOVIMENTOS.....	79
5.6	VALIDAÇÃO EXPERIMENTAL.....	80
5.6.1	<b>Demonstrações de pega</b> .....	<b>81</b>
5.6.2	<b>Esforço máximo da garra</b> .....	<b>83</b>
5.7	COMPARATIVO FINAL COM O VIPER X 300.....	84
5.8	LIMITAÇÕES E CONDIÇÃO DE OPERAÇÃO.....	86
<b>6</b>	<b>CONCLUSÃO</b> .....	<b>89</b>
6.1	TRABALHOS FUTUROS.....	90
	<b>REFERÊNCIAS BIBLIOGRÁFICAS</b> .....	<b>91</b>
	<b>APÊNDICE A – COMANDO DE DOIS ROBÔS SIMULTANEAMENTE</b> .....	<b>94</b>
	<b>APÊNDICE B – AGARRAR E MOVIMENTAR OBJETO (<i>PICK AND PLACE</i>)</b> ...	<b>95</b>

## 1 INTRODUÇÃO

Tornou-se imperativa a utilização de robôs com níveis cada vez maiores de complexidade para a realização de tarefas específicas e repetitivas nos mais diversos ramos do setor industrial. Porém, os primeiros robôs industriais datam da década de 50 e 60 e surgiram no contexto das linhas de montagem de automóveis. A prática está intrinsecamente associada à redução de custos, aumento da produtividade, aumento da qualidade e segurança. Na sociedade contemporânea, por exemplo, não é mais possível imaginar a produção em massa de veículos sem a aplicação de manipuladores robóticos soldando, pintando, serrando e instalando módulos elétricos no chão de fábrica das montadoras de automóveis. De forma semelhante, qualquer indústria de manufatura em geral pode ser beneficiada pelos processos de automação.

No ano de 2021 haviam mais de 3,5 milhões de robôs industriais operando em fábricas ao redor do mundo. O que representa um aumento de 517.385 robôs em comparação com 2020 (IFR, 2022). Com isso, nota-se que a capacidade de automação representa um fator determinante para toda a cadeia produtiva, tendo estreita relação com os ciclos econômicos e com a capacidade de produção de mercadorias de uma fábrica.

A robótica vem apresentando alto crescimento também como linha de pesquisa. São relevantes as aplicações: desde algoritmos de inteligência artificial até a construção de novos formatos mecânicos para os robôs, que estão se tornando mais flexíveis e modulares. As fronteiras que determinam o que um robô é capaz de executar estão se expandindo. Isso se deve tanto pela pretensão de atender as necessidades do setor industrial quanto para inserir os sistemas robóticos em novos campos.

Dentre as inovações, estão os modelos de atuadores mais modernos (motores responsáveis pela movimentação das juntas de um robô industrial). Esses motores são considerados inteligentes, uma vez que contam com sensores embutidos para fornecer *feedback* das principais grandezas elétricas e mecânicas em tempo real e com uma placa de controle, que facilita a sua configuração. Assim, é possível dispor de diferentes plataformas para o controle do sistema, utilizando-se tanto *softwares* próprios dos motores quanto as linguagens de programação mais comuns para o interfaceamento.

Um modelo comercial de manipulador robótico, com foco em aplicações de pesquisa e de educação, mas que conta com um *hardware* robusto e diversas funcionalidades de *software* é o Viper X 300, produzido pela *Interbotix*. Ele é um exemplo de braço robótico construído com motores inteligentes, modelos DYNAMIXEL, da empresa Robotis. Podendo ser programado com plataformas como Python, MATLAB, Moveit, Gazebo, entre outras. Destaca-se o fato da disponibilização *open source* tanto das suas características físicas quanto de demonstrações de código para funções básicas.

O trabalho apresentado neste documento visa relatar os processos envolvidos na

construção e validação de um robô manipulador baseado nas especificações do Viper X 300 e aproveitando as ferramentas disponíveis pelo fabricante para a simulação e controle do robô, através de um ambiente virtual. O robô construído conta com uma garra flexível, cinco graus de liberdade, atuadores inteligentes também modelos DYNAMIXEL e capacidade de ser programado para executar um amplo espectro de tarefas.

Observa-se que este projeto, em especial a concepção e desenho das estruturas mecânicas, foram desenvolvidos em conjunto com o Lucas Strappazon, mestrando em Engenharia Mecânica pela UFSM.

## 1.1 MOTIVAÇÃO

A realização deste projeto partiu do desejo de trabalhar em uma aplicação prática de robótica, na qual fosse possível construir um dispositivo físico funcional. Mais importante ainda do que chegar em um produto final, a intenção era poder experimentar todos os estágios de construção de um robô: definição do objetivo, planejamento e *design*, seleção dos componentes, montagem mecânica, instalação dos módulos eletrônicos, programação, testes e ajustes. Este trabalho tem ainda como motivação a construção de um robô de plataforma aberta com cinemática semelhante à do Viper X 300. Dessa forma, é possível utilizar as configurações fornecidas pelo fabricante para validar os movimentos dos atuadores e a capacidade do robô em seguir trajetórias.

## 1.2 OBJETIVOS DO ESTUDO

O principal objetivo deste trabalho consiste em construir e analisar experimentalmente um robô manipulador, atendendo às características de flexibilidade, resistência mecânica e capacidade de execução de movimentos conforme programados.

Em vista do exposto, os objetivos específicos são:

- Pesquisar as arquiteturas mais atuais de manipuladores robóticos;
- Especificar os requisitos de projeto (mecânicos e elétricos);
- Explorar as diferentes possibilidades proporcionadas pela manufatura aditiva para a construção de protótipos;
- Executar a construção física do robô;
- Implementar um sistema de controle apropriado para a realização dos movimentos;

- Desenvolver um algoritmo para a geração de trajetórias do robô manipulador;
- Validar os movimentos do robô utilizando as ferramentas disponibilizadas pelo fabricante do Viper X 300, afim de verificar o grau de similaridade entre a cinemática dos robôs.

### 1.3 DESCRIÇÃO DO PROBLEMA

A construção do robô manipulador carrega consigo alguns questionamentos que nortearam o desenvolvimento do projeto. Portanto, as perguntas a seguir servem como temas de fundo para as propostas apresentadas no decorrer do documento:

- É possível montar um robô manipulador eficiente e com dimensões similares a um modelo comercial com materiais acessíveis?
- Como configurar um sistema de controle capaz de planejar movimentos complexos de manipulação?
- Quais são os melhores critérios para estabelecer uma comparação entre robôs?
- É possível desenvolver um robô, que, com poucas modificações, seja adaptável a diferentes tarefas?

Muitos dos desafios enfrentados no desenvolvimento de um robô no meio acadêmico também estão presentes no setor industrial. Adaptabilidade, capacidade de execução de movimentos complexos e diminuição de custos de produção são temas atuais na área de robótica. Logo, o presente trabalho também busca explorar possíveis caminhos para responder essas indagações.

### 1.4 PRINCIPAIS CONTRIBUIÇÕES

A principal contribuição deste trabalho é oferecer uma plataforma de *hardware* robótico completa e pronta para ser utilizada em aplicações dentro do laboratório de robótica da Universidade Federal de Santa Maria (UFSM). Sendo possível usufruir das configurações cinemáticas e controle de movimentos fornecidos pelo fabricante de um robô comercial, no caso o robô Viper X 300. A partir deste robô manipulador, é possível desenvolver cenários de visão computacional e *Bin Picking*, que consiste em um problema central de robótica, na qual é realizado o manuseio de objetos em posições aleatórias e executar

múltiplas outras tarefas de manipulação. É importante destacar que os procedimentos formais descritos definem um guia para a construção de robôs manipuladores e estabelecem critérios para a comparação entre sistemas robóticos, que podem ser utilizados quando do desenvolvimento de outros modelos de robôs.

## 1.5 ORGANIZAÇÃO DO TRABALHO

Os capítulos posteriores estão dispostos da seguinte forma.

No capítulo 2, é apresentada uma revisão bibliográfica acerca de robótica, cinemática de robôs, componentes de sistemas robóticos e manufatura aditiva. Seguida da apresentação do estado da arte de robôs manipuladores a nível de pesquisa e de uso industrial.

No capítulo 3, são apresentados os aspectos construtivos do robô desenvolvido, incluindo a descrição das escolhas dos componentes, a explicação do projeto mecânico, a análise de esforços, os procedimentos de impressão 3D e a montagem da estrutura.

No capítulo 4, é apresentado o mecanismo de controle do braço robótico, detalhando o sistema de controle proposto, o planejamento dos movimentos, as bibliotecas utilizadas para a programação do dispositivo, a geração das trajetórias e as demais configurações do robô.

No capítulo 5, são apresentados e analisados os resultados experimentais. São relatados resultados tanto de validação da garra para o manuseio de objetos sensíveis, quanto a validação das trajetórias executadas no robô em comparativo com as simuladas no Viper X 300.

No capítulo 6, são apresentadas as conclusões e sugestões para a continuidade da pesquisa.



## 2 REVISÃO BIBLIOGRÁFICA

Para descrever um robô manipulador, é essencial abordar os seguintes tópicos: descrição de posição e orientação, caracterização dos elementos do robô e estruturas cinemáticas. Esses conceitos fundamentais para embasar teoricamente esta pesquisa são discutidos nas próximas seções, acompanhados do estado da arte no campo do desenvolvimento de robôs manipuladores.

### 2.1 FUNDAMENTAÇÃO TEÓRICA

#### 2.1.1 Manipuladores

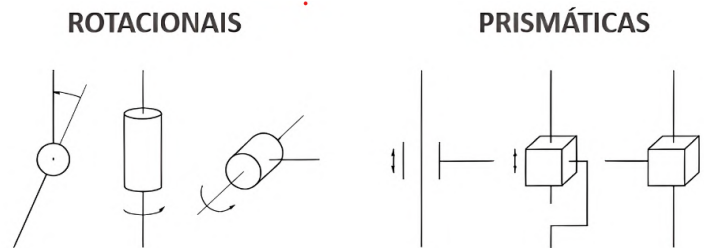
O manipulador robótico é um sistema mecânico composto por elos, juntas e atuadores. Esse conjunto é utilizado na movimentação de ferramentas e peças entre pontos ou seguindo curvas de trajetórias, podendo ser reconfiguráveis conforme a necessidade (FERREIRA; ROMANO; MACHADO, 2002).

#### 2.1.2 Juntas e Elos

As juntas são as partes móveis, normalmente atuadas por motores, e têm o papel de movimentar os elos, que são as partes rígidas do robô. Os elos são caracterizados por elementos estruturais que possuem alta rigidez quando submetidos a forças de flexão e/ou de torção, sendo que os materiais mais comuns para a composição da estrutura de elos são aço, alumínio e ligas metálicas. Esses elementos conectam as juntas e possibilitam a montagem do sistema mecânico. Por sua vez, as juntas são elementos que permitem o movimento relativo entre um elo específico e outro adjacente. Elas podem ser classificadas de acordo com a possibilidade de movimentação, sendo rotacionais (*revolute joints*) quando permitem movimento rotativo ou prismáticas (*prismatic joints*) quando possibilitam o movimento linear (SPONG; HUTCHINSON; VIDYASAGAR, 2005). A Figura 1 apresenta os tipos de representações mais comuns das juntas em um diagrama.

Além disso, é comum representar o movimento das juntas por meio de setas direcionais (indicadas pelo símbolo  $\theta$ ), já que a junta é responsável pela mobilidade do manipulador, como mostrado na Figura 2 através de um robô SCARA (*Selective Compliance Assembly Robot*). É possível observar que esse robô SCARA possui três juntas de rotação e uma junta prismática. Esse modelo é caracterizado por sua mecânica estrutural que ofe-

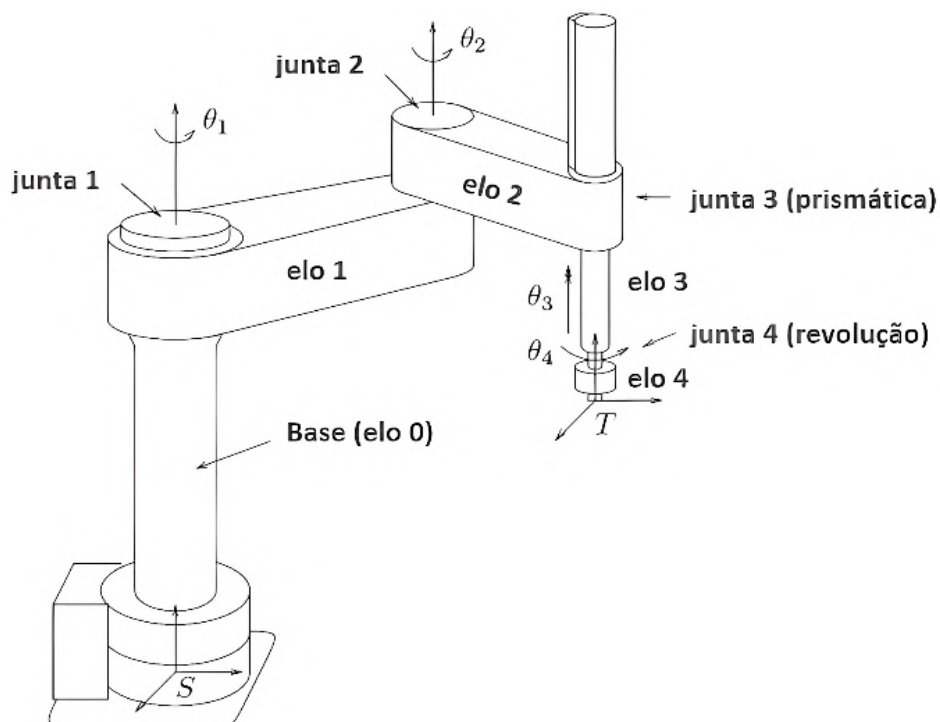
Figura 1 – Representações dos tipos de juntas.



Fonte: Adaptado de (SCIAVICCO; SICILIANO, 2001).

rece alta rigidez em cargas verticais. Também conforme ilustrado na Figura 2, o primeiro elo é denominado base ( $S$ ) e é fixo, enquanto o elo na outra extremidade, conhecido como efetuador ( $T$ ), é onde geralmente são acopladas diversas ferramentas.

Figura 2 – Manipulador do tipo SCARA.



Fonte: Adaptado de (MURRAY; LI; SASTRY, 2017).

### 2.1.3 Graus de Liberdade

No estudo da robótica, é de grande interesse determinar a localização dos diversos elementos do manipulador e dos objetos do seu ambiente no espaço tridimensional. Para descrever a posição e orientação de um corpo no espaço, sempre é atrelado rigidamente ao objeto um sistema de coordenadas, também chamado de sistema de referência

(CRAIG, 2005).

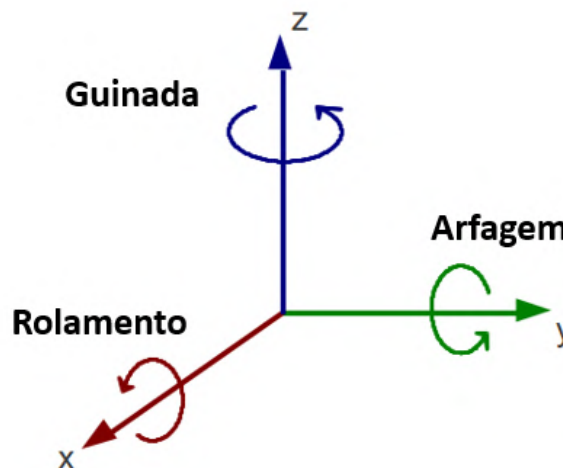
Os graus de liberdade referem-se à capacidade de movimento e orientação de um corpo rígido no espaço. Em geral, para posicionar e orientar um objeto arbitrariamente no espaço tridimensional, são necessários seis graus de liberdade, dos quais três são responsáveis pelo posicionamento de um ponto e três pela orientação do objeto em relação a um eixo de referência.

O movimento do punho possui uma nomenclatura específica, descrita por:

- *Roll* ou rolamento: rotação do punho ao redor do braço;
- *Pitch* ou arfagem: rotação vertical do punho, para cima ou para baixo;
- *Yaw* ou guinada: rotação horizontal do punho, para a esquerda ou para a direita.

A Figura 3 ilustra esse sistema.

Figura 3 – Sistema de coordenadas com eixos de posicionamento e ângulos de orientação.



Fonte: Adaptado de (XMARKLABS, 2017).

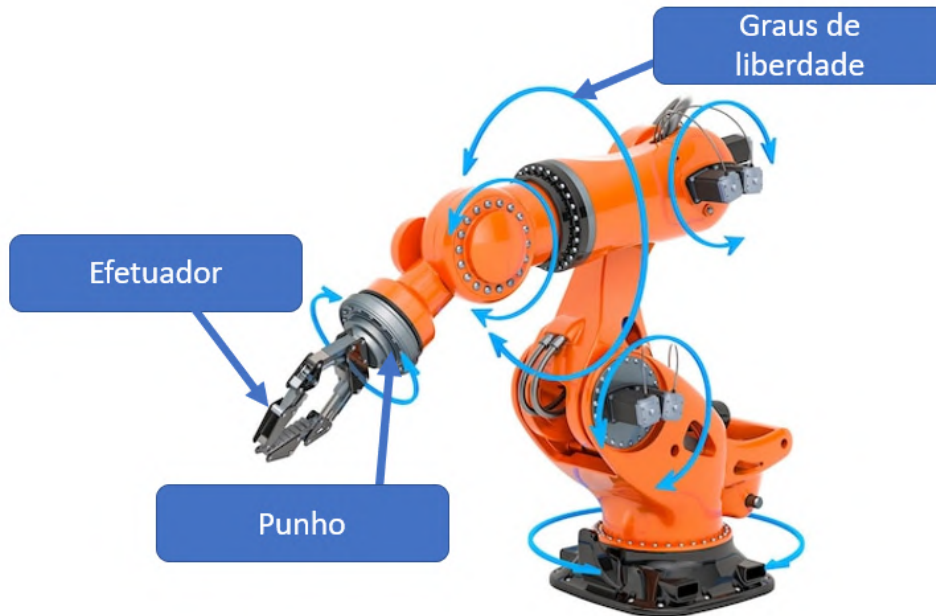
Na robótica, os graus de liberdade estão relacionados ao número de variáveis independentes necessárias para determinar a localização e a orientação de todas as peças do mecanismo (FERREIRA; ROMANO; MACHADO, 2002). Com isso, o objetivo é permitir que a ferramenta alcance um ponto qualquer dentro de um espaço limitado ao redor do braço.

A esse espaço limitado, dá-se o nome de volume de trabalho de um robô. Ou seja, refere-se à área do ambiente que o efetuator final é capaz de alcançar. A forma e o tamanho desse volume dependem da estrutura do manipulador, bem como da presença de limitações nas articulações mecânicas. Assim, a principal tarefa do braço é posicionar o punho, que é responsável pela orientação do atuador final.

A Figura 4 é uma ilustração que apresenta alguns dos componentes de um robô industrial de seis graus de liberdade. São mostrados o efetuator, que consiste em uma

garra com dois dedos responsável pela execução final da tarefa determinada e o punho, ao qual o efetuador é acoplado.

Figura 4 – Componentes de um robô industrial de seis graus de liberdade.



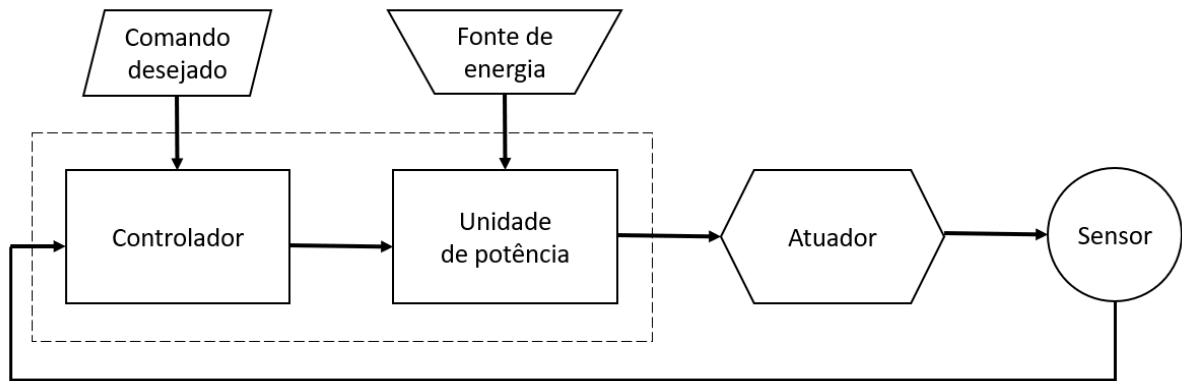
Fonte: Adaptado de (MACHNATA, 2023).

A estrutura mecânica do robô está em permanente interação com o ambiente, respondendo tanto aos esforços mecânicos quanto aos comandos de um sistema de controle. A Figura 5 apresenta a formatação tradicional do controle de um sistema robótico. Nota-se que ela contém os principais componentes de um sistema de controle. O controlador é responsável por decidir que medida de atuação deve ser imposta, de forma a corresponder ao comando desejado. A unidade de potência fornece energia ao atuador. O atuador gera deslocamento linear ou rotacional. E o sensor produz um sinal de medição que é repassado ao controlador (SANTOS, 2004).

## 2.2 CINEMÁTICA DIRETA

A cinemática busca tratar da descrição dos movimentos de pontos, corpos e grupos de objetos, sem se preocupar com a análise das suas causas, ou seja, desconsiderando as forças que dão origem aos movimentos. Na robótica, um manipulador pode ser considerado, sob o ponto de vista mecânico, como uma cadeia cinemática de corpos rígidos (os elos) conectados por juntas. A base do robô é a extremidade fixa da cadeia, enquanto que o efetuador fica livre para se mover pelo espaço e está localizado na outra extremidade. A composição dos movimentos elementares de cada elo em relação ao anterior resulta na

Figura 5 – Sistema de controle tradicional.



Fonte: Adaptado de (SANTOS, 2004).

movimentação completa da estrutura.

A fim de lidar com a geometria complexa de um manipulador, são fixados sistemas de referência às várias partes do mecanismo e depois são descritas as relações entre eles. Assim, é possível determinar como a localização dos sistemas de referência se altera à medida que o mecanismo se articula (CRAIG, 2005).

O problema da cinemática direta, para um sistema com juntas rotacionais, consiste em definir uma função  $f$  tal que:

$$x = f(\theta) \quad (2.1)$$

Em que,  $x$  representa a posição e/ou orientação do efetuador em relação à base e  $\theta$  representa o ângulo das juntas. Dessa forma, a cinemática direta busca determinar a localização da garra (ou de cada elo do manipulador) com referência à base, dadas as variáveis das juntas.

Uma das principais metodologias para a determinação da cinemática direta de manipuladores é a notação de Denavit-Hartenberg (DH). Trata-se de um método sistemático para descrever a posição e orientação relativa entre dois elos consecutivos. Ele se baseia no estabelecimento de convenções para os parâmetros das juntas e dos elos do manipulador. O método indica que qualquer robô pode ser descrito atribuindo-se quatro valores para cada elo. Dois para descrever o elo em si e dois para descrever a conexão do elo com um vizinho.

### 2.2.1 Sistema de Referência

É essencial na notação de Denavit-Hartenberg a fixação adequada do sistema de referência (também conhecido como *frame*) em cada um dos elos. De forma geral, um

robô com  $n$  graus de liberdade é formado por  $n$  juntas e  $n + 1$  elos. Os elos são numerados começando com o valor 0 para a base e  $n$  para o elo do efetuador. Já as juntas iniciam com valor 1, na junta que conecta a base ao primeiro elo móvel e aumenta continuamente até  $n$ . Por exemplo, o elo  $i$  é conectado entre o elo inferior  $i - 1$  pela junta  $i$  e conectado com o elo posterior  $i + 1$  pela junta  $i + 1$  (PREEZ, 2014). Para um sistema de coordenadas local em cada elo, o método propõe que os eixos sejam estabelecidos como:

- O eixo  $z_i$  alinhado com eixo da junta  $i + 1$ ;
- O eixo  $x_i$  estipulado pela normal comum entre o eixo  $z_{i-1}$  e o eixo  $z_i$ , apontando de  $z_{i-1}$  para  $z_i$ ;
- O eixo  $y_i$  obtido como uma consequência da regra da mão direita.

### 2.2.2 Definição dos Parâmetros

Existem quatro parâmetros que definem a geometria de um elo  $i$ . Eles são expressos como:

- $a_i$ : é o comprimento do elo. Trata-se da distância entre os eixos  $z_i$  e  $z_{i+1}$ , medida ao longo do eixo  $x_i$ ;
- $\alpha_i$ : indica a torção do elo. Trata-se do ângulo de rotação entre os eixos  $z_i$  e  $z_{i+1}$ , medida em  $x_i$ ;
- $d_i$ : é a distância ao longo do eixo de junta. Trata-se da distância entre os eixos  $x_{i-1}$  e  $x_i$ , medida ao longo de  $z_i$ ;
- $\theta_i$ : é o ângulo de junta. Trata-se do ângulo de rotação entre os eixos  $x_{i-1}$  e  $x_i$ , medido ao longo do eixo  $z_i$ .

Aplicando esses conceitos na prática, vale a seguinte caracterização:

- Se a junta  $i$  for de rotação:  $d_i = \text{fixo}$  e  $\theta_i = \text{variável}$ ;
- Se a junta  $i$  for prismática:  $d_i = \text{variável}$  e  $\theta_i = \text{fixo}$ .

### 2.2.3 Procedimentos

A finalidade da representação DH é apresentar um enfoque matricial para descrever, de forma matemática, as transformações de translação e rotação geradas pelos pares articulação-junta. A Tabela 1 exemplifica como é realizado o procedimento para a aplicação dos parâmetros descritos anteriormente. Cada elo (*link*) corresponde a uma linha da tabela com seus parâmetros associados.

Tabela 1 – Relação dos parâmetros de Denavit-Hartenberg.

<b>Elo (i)</b>	$a_i$	$\alpha_i$	$d_i$	$\theta_i$
<b>1</b>	$a_1$	$\alpha_1$	$d_1$	$\theta_1$
<b>2</b>	$a_2$	$\alpha_2$	$d_2$	$\theta_2$
...	...	...	...	...
<b>j</b>	$a_j$	$\alpha_j$	$d_j$	$\theta_j$
...	...	...	...	...
<b>n</b>	$a_n$	$\alpha_n$	$d_n$	$\theta_n$

Fonte: Adaptado de (PREEZ, 2014).

Essa abordagem resulta na obtenção de uma matriz de transformação homogênea, montada a partir dos parâmetros de DH e da equação:

$$T_i^{i-1} = \begin{bmatrix} \cos(\theta_i) & -\text{sen}(\theta_i)\cos(\alpha_i) & \text{sen}(\theta_i)\text{sen}(\alpha_i) & a_i\cos(\theta_i) \\ \text{sen}(\theta_i) & \cos(\theta_i)\cos(\alpha_i) & -\cos(\theta_i)\text{sen}(\alpha_i) & a_i\text{sen}(\theta_i) \\ 0 & \text{sen}(\alpha_i) & \cos(\alpha_i) & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2.2)$$

Para um manipulador de quatro graus de liberdade, por exemplo, terão que ser definidas quatro matrizes iguais a da Equação (2.2). Assim, com base nas etapas mencionadas anteriormente, é possível determinar uma matriz de transformação homogênea para cada elo.

Essas matrizes são multiplicadas sequencialmente para obter a matriz de transformação global do robô, que descreve a posição e a orientação do efetuador final em relação à base do robô, conforme a equação:

$$T_n^0 = T_1^0 T_2^1 T_3^2 \dots T_n^{n-1} \quad (2.3)$$

Essas são as equações básicas para a análise cinemática de manipuladores. Elas expressam as relações entre os sistemas de referência. Portanto, como visto nesta seção, o método de DH é constituído, basicamente, de quatro etapas:

1. Estabelecer um sistema de coordenadas local em cada elemento;

2. Utilizar o sistema de coordenada local para definir os parâmetros de DH correspondentes a cada elemento;
3. Substituir esses parâmetros na matriz homogênea genérica para obter a matriz específica para cada elemento;
4. Multiplicar as matrizes do passo anterior para obter a matriz de transformação global.

### 2.3 EFETUADORES

Efetadores são os dispositivos que definem a funcionalidade dos robôs manipuladores. Também conhecidos como garras, permitem a execução de tarefas específicas, como a soldagem, pintura ou manipulação de peças. De acordo com (CRAIG, 2005), os efetadores podem ser classificados em quatro tipos:

- **Garras rígidas:** são aquelas que não possuem movimentos relativos nos elementos terminais, sendo utilizadas principalmente em tarefas de carregamento e descarregamento de paletes e caixas;
- **Ferramentas:** são empregadas em tarefas específicas como soldagem e pintura;
- **Garras não mecânicas:** possibilitam a manipulação por meio de ímãs, sistemas de sucção, entre outros dispositivos não mecânicos;
- **Garras mecânicas:** atuam em contato direto com os objetos com um mecanismo de pega conhecido como dedos. Pode ser rígida ou flexível.

Os estudos sobre garras robóticas flexíveis estão ganhando destaque devido à sua capacidade de facilitar a manipulação de objetos pequenos, frágeis e com formas irregulares. Essas garras são capazes de se adaptar a diferentes configurações cinemáticas.

### 2.4 MANUFATURA ADITIVA

A técnica de manufatura aditiva é conhecida popularmente como impressão 3D. É definida como uma forma de fabricar objetos em que se adiciona material a partir de um código de programação inserido na máquina, a qual irá imprimir a peça camada por camada (MICALLEF, 2015).

Em comparação com os processos tradicionais de fabricação, a construção via impressão 3D oferece as seguintes vantagens (VOLPATO, 2021):



- Rápida, prática e de fácil acesso;
- Produção de objetos leves, o que reduz custos;
- Pouco desperdício de material;
- Liberdade geométrica, ou seja, capacidade de construir objetos de geometrias variadas e produtos com grande detalhamento;

Conforme argumentado por (GU, 2015), desde a introdução da manufatura aditiva na década de 80, as indústrias de manufatura têm aproveitado de forma significativa as suas vantagens para produzir pequenas quantidades de peças com geometrias complexas de maneira ágil.

Todavia, essa tecnologia também apresenta as seguintes limitações (VOLPATO, 2021):

- Precisão e acabamento superficial inferiores aos processos convencionais como a usinagem;
- Mais adequada para pequena escala. Ao considerar grandes lotes, é lenta e mais cara;
- Limitação em relação aos materiais que podem ser empregados. Em geral, as propriedades não são as mesmas dos materiais em processos tradicionais.

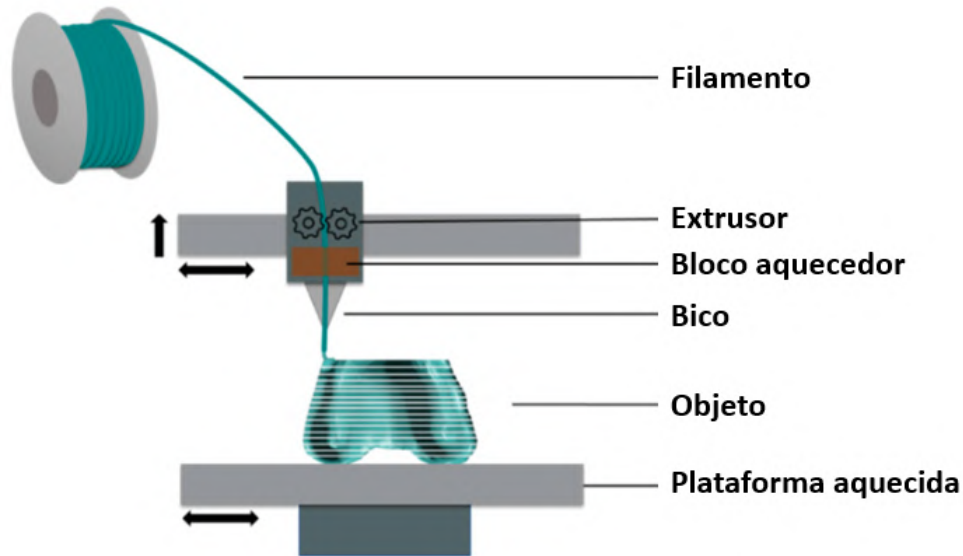
Esses aspectos colocam a manufatura aditiva como uma técnica apropriada para a elaboração de protótipos, que necessitam de rápidas e acessíveis provas de conceito e constante iteração do projeto mecânico. Os sistemas de impressão 3D utilizam como base uma representação geométrica tridimensional dos objetos a serem fabricados. O formato mais empregado atualmente é o *STereoLithography* (STL). A modelagem CAD (*Computer-aided design*) é a principal ferramenta para a geração das geometrias (VOLPATO, 2021).

Um dos métodos mais robustos e economicamente atrativos de impressão 3D é a Fabricação por Filamento Fundido (FFF). Esse processo foi patenteado pela empresa *Stratasys* na década de 90 e se popularizou em 2009 com a expiração da patente (VOLPATO, 2021). Na FFF, um material termoplástico alimenta um "bico" aquecido, onde o filamento é derretido e então depositado camada por camada para criar o objeto (BURKHARDT et al., 2020).

A Figura 6 ilustra esses procedimentos. O extrusor traciona o filamento entre os componentes rolantes, empurrando o material em direção ao "bico", que se aquece em temperatura controlada. Outro elemento importante do processo é a plataforma aquecida. Essa superfície serve para a deposição do material fundido e colabora com a adesão da primeira camada de impressão.

Na técnica de FFF são usualmente empregados filamentos de polímeros, mais especificamente os termoplásticos. Destacam-se os seguintes tipos de materiais:

Figura 6 – Exemplificação da técnica de Fabricação por Filamento Fundido.



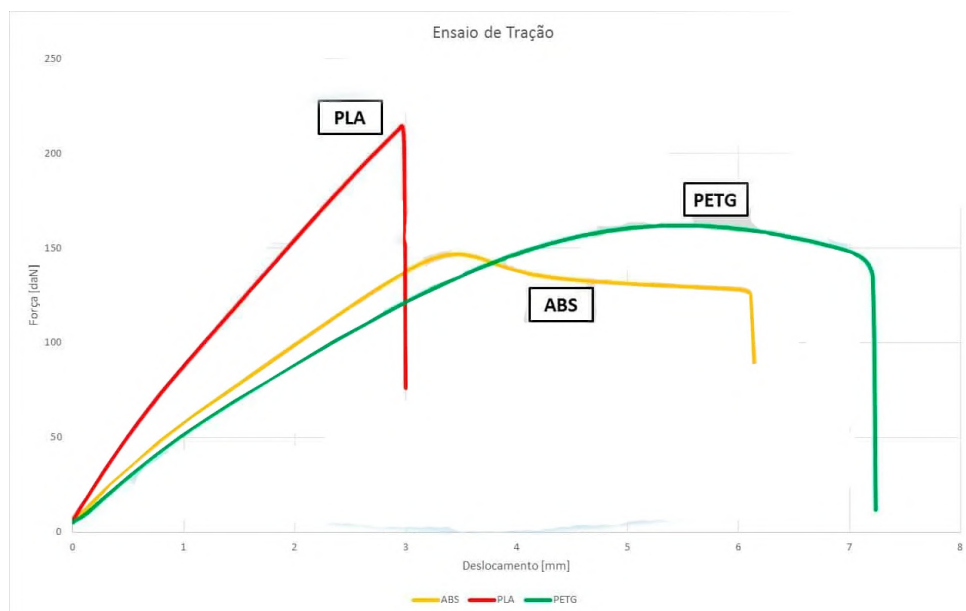
Fonte: Adaptado de (BURKHARDT et al., 2020).

- **PLA** (ácido polilático): É biodegradável, promove uma alta qualidade visual, boa aderência à mesa e entre camadas e elevada dureza superficial. Porém, não possui boa resistência térmica, não podendo ser exposto ao sol, por exemplo. É fácil de imprimir e tem tendência menor de entupir a extrusora. Pode ser usado em impressora aberta ou fechada, com ou sem mesa aquecida e não requer altas temperaturas para a extrusão.
- **ABS** (acrilonitrila butadieno estireno): É feito de produtos derivados de petróleo. Suas características são: resistência ao impacto, boa resistência térmica e a produtos químicos, durabilidade e baixa visibilidade de camada proporcionando um acabamento liso. Necessita de maior temperatura para atingir o ponto de fusão. Durante o processo de impressão, ocorre a geração de fumaça considerada tóxica, por isso deve-se operar a impressora 3D em uma área ventilada, sendo indicado o uso em impressoras fechadas e com mesa aquecida.
- **PETG** (tereftalato de polietileno glicol modificado): É gerado pela copolimerização do PET, o que confere maior estabilidade e menor ponto de fusão. Combina os aspectos de facilidade de impressão do PLA com algumas das propriedades mecânicas do ABS. É resistente, durável e tem uma boa resistência química. Adequado para peças mecânicas e protótipos funcionais. Pode ser utilizado em impressoras abertas ou fechadas, preferencialmente com mesa aquecida.
- **TPU** (poliuretano termoplástico): É um material versátil, elástico, durável e resistente ao impacto. Apresenta boa flexibilidade em variadas temperaturas e alta elasticidade em toda a sua faixa de dureza. Recomenda-se que seja impresso em uma veloci-

dade de impressão mais baixa a fim de evitar uma extrusão inadequada. Pode ser utilizado em impressoras tanto abertas quanto fechadas, mas indica-se o uso em mesa aquecida.

A Figura 7, exibe o resultado de um ensaio de tração realizado com amostras de material da fabricante 3DLAB, cujo filamento foi adquirido para a construção do robô. Conforme a análise das curvas, o PETG é um material nobre, capaz de suportar aproximadamente 162 kg, absorvendo com isso 7,2 mm de deformação antes da ruptura. Sendo assim, ele suporta mais carga e é mais dúctil que o ABS. Já quando comparado ao PLA, não suporta tanta carga estática, porém aceita uma deformação maior. Sob o ponto de vista mecânico, portanto, o PETG é o que possui melhor resistência.

Figura 7 – Ensaio de tração para os materiais PLA, ABS e PETG.



Fonte: (3DLAB, 2017).

## 2.5 ATUADORES

Os atuadores são os responsáveis pela movimentação dos sistemas robóticos, seja na forma de translação ou rotação das juntas seja na ativação do efetuator. Os servomotores são um tipo de atuadores utilizados em aplicações que exigem controle de posição preciso. São uma combinação de um motor de corrente contínua (DC) e um sistema de *feedback* de posição, geralmente por meio de um potenciômetro. Assim, o controlador, que geralmente é do tipo PID (Proporcional Integral Derivativo) monitora constantemente o erro entre a posição atual e a posição desejada e ajusta a corrente enviada para o motor de forma a minimizar esse erro.

## 2.6 ESTADO DA ARTE

A capacidade e tecnologia dos manipuladores robóticos tem evoluído rapidamente, impulsionadas por avanços em áreas como eletrônica, inteligência artificial, ciência dos materiais e mecânica.

### 2.6.1 Robôs comerciais e de código aberto

Na área de robótica educacional e de pesquisa acadêmica, são utilizados modelos de robôs mais simples, que buscam aliar um orçamento acessível à possibilidade de implementação de tarefas básicas no manipulador. Nesse cenário, existem modelos *open source* como o manipulador *Moveo*, que pode ser visualizado na Figura 8. Ele foi desenvolvido pela *BCN3D* com a finalidade de se tornar um material didático de simples replicação e difundir a utilização da manufatura aditiva. Os materiais necessários para a fabricação deste robô são disponibilizados na plataforma *Github*. Isso inclui os modelos STL para a impressão 3D, os modelos CAD, a lista de materiais, software e manual de instruções (BCN3D, 2016b). Entre as especificações desse produto, destacam-se as apresentadas na Tabela 2.

Figura 8 – Robô *open source* da *BCN3D*.



Fonte: (BCN3D, 2016a).

Também existem no mercado manipuladores comerciais com a mesma finalidade,

Tabela 2 – Especificações do robô *Moveo*.

Característica	Especificação
Graus de Liberdade	5
Carga	500g
Alcance	400mm
Atuador	Motor de passo
Material	Impressão 3D
Controlador	Arduino Mega 2560

Fonte: Autor, 2023.

como o *Dobot Magician*, apresentado na Figura 9. Ele é produzido pela fabricante *Dobot* com o objetivo de ser um manipulador multifuncional para a educação prática de conceitos de robótica. Para isso, oferece a execução de tarefas como desenho, escrita e gravação à laser (DOBOT, 2023). As principais especificações desse modelo são mostradas na Tabela 3.

Figura 9 – Robô *Dobot Magician* da fabricante *Dobot*.

Fonte: (MINIPA, 2016).

Tabela 3 – Especificações do robô *Dobot Magician*.

Característica	Especificação
Graus de Liberdade	4
Carga	500g
Alcance	320mm
Atuador	Servomotor
Material	Alumínio
Controlador	Dobot Integrated Controller

Fonte: Autor, 2023.

### 2.6.2 Construção de manipuladores na literatura acadêmica

Em (COSTA; MACHADO; CARNEIRO, 2020) é realizada a implementação e validação de um modelo de braço robótico *open source*, o Thor. A construção do protótipo foi desenvolvida a partir dos arquivos CAD disponibilizados, sendo alterado o tipo de material de impressão. Optou-se pela utilização do ABS no lugar do PLA do modelo original. Foram efetuadas também mudanças no sistema eletrônico, com o uso de dois microcontroladores Arduino Mega. A validação se deu pela movimentação do robô. Não foram apresentados, no entanto, a programação de trajetórias e análise em simulação.

Em (PURDON; SETATI; MARAIS, 2021) também é construído um braço robótico a partir de um modelo *open source*. São apresentados os aspectos de montagem, conforme o material do projeto. Alguns problemas na fabricação das peças 3D foram encontrados e foram discutidas algumas soluções de melhorias. O robô foi testado de forma a validar as especificações do manual. Também foi proposta uma interface de programação, a partir da movimentação direta das juntas.

O desenvolvimento de braços robóticos através de métodos de impressão 3D representa um avanço significativo na robótica. O artigo (BANSAL et al., 2023) fornece experiências práticas sobre a fabricação de braços robóticos utilizando impressão 3D com o material PLA e relata desafios usuais da técnica como manipulação de peças grandes e parâmetros de configuração. No projeto do robô deste trabalho, optou-se pela utilização do material PETG, que oferece vantagens adicionais como maior resistência, ao mesmo tempo em que requer um cuidado maior que a impressão com o PLA.

O HydraX (KRIMPENIS; PAPAPASCHOS; BONTARENKO, 2020) é um braço robótico de 6 graus de liberdade, impresso em 3D com o material PETG. O artigo inclui além dos aspectos construtivos, testes de cinemática direta para determinar a localização do efetuador final e a repetibilidade do robô.

Em (LAURITZEN, 2023) é proposta uma plataforma de simulação para o ReactorX-150, um robô fabricado pela Interbotix. O objetivo do trabalho foi integrar o dispositivo com o ROS em uma plataforma Linux para permitir funcionalidades como controle de posição e trajetória e a aplicação da cinemática direta e inversa. Foram utilizadas tecnologias como ROS, Gazebo, RViz e MoveIt. A dissertação apresenta uma documentação extensa a respeito do robô ReactorX-150 e seus pacotes de controle. O autor utiliza o método de Denavit-Hartenberg para apresentar a solução da cinemática direta do braço robótico. São apresentados também aspectos detalhados comparando a simulação do robô via MoveIt e códigos Python com a implementação no robô físico. O projeto desenvolvido se relaciona bastante com essa proposta, com a diferença de que a implementação não foi realizada com o robô da Interbotix, mas sim com uma versão de construção própria.

Os manipuladores estão se tornando mais flexíveis em termos de *design* e funcionalidades. Mecanismos de articulação flexíveis, contando com braços de segmentos suaves e sensores integrados, estão sendo desenvolvidos para permitir movimentos mais naturais

e uma interação mais segura com o ambiente e com humanos. Em análise realizada por (SHINTAKE et al., 2018), é enfatizado que as garras do tipo Fin Ray são classificadas no grupo de rigidez ou deformação controlada e são amplamente utilizadas na manipulação de uma variedade de itens, como esferas, ovos, verduras, bulbos de flores, brinquedos, entre outros. Essas características têm impulsionado pesquisas recentes na aplicação dessas garras em contextos domésticos, que envolvem a manipulação de objetos com diferentes formas, tamanhos e pesos. Para o manuseio de frutas, verduras e produtos frágeis, (ELGENEIDY et al., 2019) propõe uma garra do tipo Fin Ray. O autor projetou a separação das nervuras internas, com um efeito crescente de ângulo, tentando manter o mesmo espaçamento entre os nervos da garra, obtendo um aumento significativo da força de sustentação dos objetos.

O protótipo desenvolvido implementa a construção de um robô manipulador, a partir da reconstituição e adaptação das peças de um robô comercial. Propondo, inclusive, um efetuador flexível, com vistas à pega de materiais sensíveis. A realização de testes mecânicos experimentais, a utilização de plataformas de programação de movimentos e a geração de trajetórias para a validação são considerados os diferenciais deste trabalho.

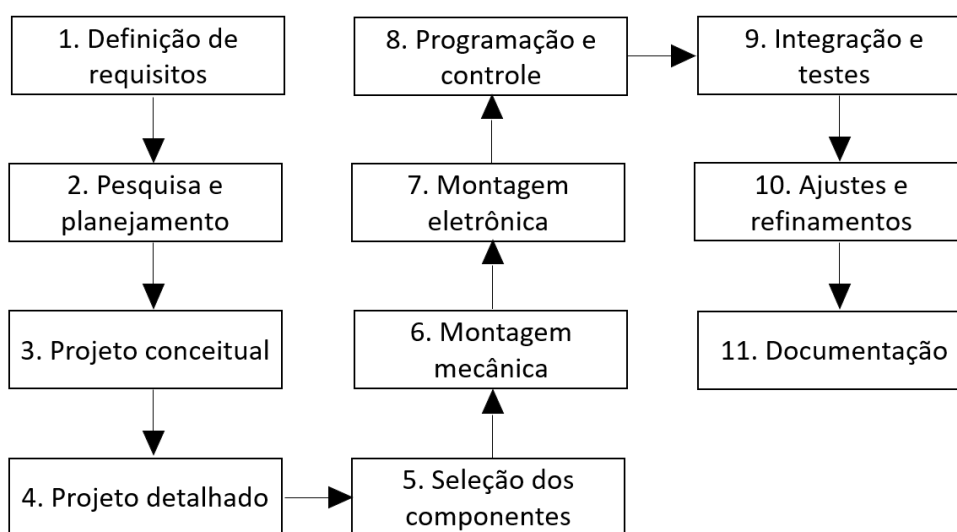
### 3 MATERIAIS E MÉTODOS

Nesta seção, serão explorados os elementos construtivos do robô manipulador. Incluindo as características relacionadas à sua estrutura e funcionalidades. Dessa forma, serão percorridos aspectos de projeto como materiais utilizados na construção da estrutura física, atuadores, sistemas elétricos e esquemas de montagem.

#### 3.1 METODOLOGIA

A metodologia proposta para a realização deste trabalho é ilustrada na Figura 10. Essa estrutura engloba as áreas de conhecimento e estágios envolvidos no processo, desde a concepção do sistema até a validação em um cenário real.

Figura 10 – Metodologia.



Fonte: Autor, 2023.

1. Definição de requisitos: O objetivo é desenvolver um robô multipropósito, com capacidade de carga em torno de 1kg e que possa executar tarefas complexas de manipulação. Ele deve ser composto por cinco graus de liberdade, permitindo maior diversidade de trajetórias. Sua construção, a nível de protótipo, deve ser simples e acessível.
2. Pesquisa e Planejamento: Foi realizada uma pesquisa sobre as tecnologias e modelos disponíveis. Assim como as melhores práticas de construção de robôs. O robô manipulador Viper X 300, fabricado pela empresa *Interbotix* e com foco nos mercados de pesquisa e educação foi o modelo que mais se alinhou aos objetivos do



projeto. Então, ele foi extensivamente analisado durante a pesquisa e considerado como base de conhecimento e de comparação para o sistema construído.

3. Projeto conceitual: Nesta etapa foram desenvolvidos os esboços iniciais. Foram considerados aspectos de mecânica, eletrônica, *software* e integração. Com isso, os requisitos também ficaram mais claros para a continuidade do projeto.
4. Projeto detalhado: Foi desenvolvido o projeto mecânico e, conseqüentemente, os modelos 3D das peças, diagramas de circuitos, entre outras especificações técnicas. Também foram avaliados os modelos matemáticos da estrutura, principalmente a cinemática.
5. Seleção dos componentes: Para a construção completa são necessários sensores, motores, controladores, fixadores, ferramentas e demais dispositivos. Priorizou-se a utilização de atuadores e microcontroladores do próprio grupo de pesquisa. Foi feita a aquisição dos novos componentes e organização dos materiais.
6. Montagem mecânica: A estrutura mecânica se constitui de peças de alumínio e derivadas de impressão 3D. Foram realizados os procedimentos para a impressão e montagem de cada uma das partes do robô.
7. Montagem eletrônica: Foram instalados e conectados os componentes eletrônicos, como a placa de controle, a alimentação, os cabos e os motores.
8. Programação e controle: Envolveu a programação dos atuadores, a geração de trajetórias e o desenvolvimento de algoritmos para a movimentação do robô.
9. Integração e testes: Foram realizados testes de todas as funcionalidades. Assim como a validação das trajetórias e a análise dos esforços da garra. Compararam-se os resultados obtidos com a simulação do Viper X 300.
10. Ajustes e refinamentos: Depois dos testes preliminares, procedeu-se com a realização de ajustes para melhorar o desempenho do robô. Tanto em relação à mudança no projeto de algumas peças quanto alterações no *software*.
11. Documentação: Todas as etapas anteriores geraram grande quantidade de conhecimento, na forma de especificações, códigos e manuais. Esse processo de construção foi documentado, tanto neste documento quanto em outros repositórios de livre acesso.

### 3.2 DESCRIÇÃO DO VIPER X 300

No mercado de robôs manipuladores para educação e pesquisa, um dos equipamentos de maior destaque são os da linha *XSeries Arm*, da *Interbotix*. O modelo Viper X 300 oferece cinco graus de liberdade e é comercializado por \$4,895.95, em agosto de 2023 (Figura 11).

Figura 11 – Braço robótico Viper X 300.



Fonte: (TROSSEN ROBOTICS, 2023).

Conforme as informações apresentadas em (TROSSEN ROBOTICS, 2023), esse equipamento fornece uma capacidade de carga de 750g e é construído com alumínio extrudado de 20 mm x 40 mm, assim como suportes de alumínio. O braço fica disposto em um rolamento de giro para maior estabilidade e precisão. Os componentes eletrônicos são cobertos com um escudo de acrílico transparente, para evitar impactos. Há, também, a possibilidade de customização do efetuador, a partir de impressão 3D. Na Tabela 4, estão resumidas as principais especificações.

Tabela 4 – Especificações do robô Viper X 300.

Característica	Especificação
Graus de Liberdade	5
Carga	750g
Alcance	750mm
Atuador	Servomotor
Material	Alumínio e Impressão 3D
Controlador	DYNAMIXEL U2D2

Fonte: Autor, 2023.

Em relação ao *hardware*, esse modelo possui uma fonte de alimentação de 12V e

10A, oito servomotores e uma placa microcontroladora. A placa é a DYNAMIXEL U2D2, a qual permite operar e controlar motores do tipo DYNAMIXEL.

São utilizados dois modelos de motores, o DYNAMIXEL XM540-W270-T e o DYNAMIXEL XM430-W350-T. O robô é composto por seis motores do primeiro modelo, responsáveis pela movimentação nos pontos de maior criticidade de carga. E por dois motores do segundo modelo, um para executar a rotação da garra e outro para a sua abertura e fechamento. As especificações desses servomotores são indicadas nas Tabelas 5 e 6.

Tabela 5 – Especificações do servomotor DYNAMIXEL XM540-W270-T.

Modelo	<b>DYNAMIXEL XM540-W270-T</b>
Microcontrolador	ARM CORTEX-M3 (72 [MHz], 32Bit)
Sensor de posição	Contactless absolute encoder (12Bit, 360°)
Motor	Coreless
Baudrate	9.600 [bps] - 4,5 [Mbps]
Algoritmo de controle	PID
Resolução	4096 [pulse/rev]
Backlash	15 [arcmin] (0,25°)
Massa	165 [g]
Gear Ratio	272,5 : 1
Torque	10,6 [N.m] (a 12,0 [V], 4,4 [A])
Velocidade a vazio	30 [rev/min] (a 12,0 [V])
Comunicação serial	TTL

Fonte: Adaptado de (TROSSEN ROBOTICS, 2023).

Tabela 6 – Especificações do servomotor DYNAMIXEL XM430-W350-T.

Modelo	<b>DYNAMIXEL XM430-W350-T</b>
Microcontrolador	ARM CORTEX-M3 (72 [MHz], 32Bit)
Sensor de posição	Contactless absolute encoder (12Bit, 360°)
Motor	Coreless
Baudrate	9.600 [bps] - 4,5 [Mbps]
Algoritmo de controle	PID
Resolução	4096 [pulse/rev]
Backlash	15 [arcmin] (0,25°)
Massa	82 [g]
Gear Ratio	353,5 : 1
Torque	4.1 [N.m] (a 12,0 [V], 2,3 [A])
Velocidade a vazio	46 [rev/min] (a 12,0 [V])
Comunicação serial	TTL

Fonte: Adaptado de (TROSSEN ROBOTICS, 2023).

Já em relação ao *software*, esse modelo se caracteriza por ser multiplataforma. É possível operar e programar movimentos no robô utilizando pacotes que são construídos em cima das bibliotecas de baixo nível dos motores DYNAMIXEL. Nesse sentido, pode-se utilizar ROS (*Robotic Operating System*), com as interfaces de programação em Python

ou MATLAB. Também é possível testar diferentes cenários com o Gazebo, que é um ambiente de simulação que oferece representações realísticas e possui interfaces gráficas e programáveis. Dentre os pacotes disponibilizados pela *Interbotix*, também estão arquivos de configuração e demonstrações de uso para o Moveit, que é um dos *softwares* mais utilizados para manipulação, incorporando capacidades de planejamento de movimentos, percepção 3D, cinemática, controle e navegação.

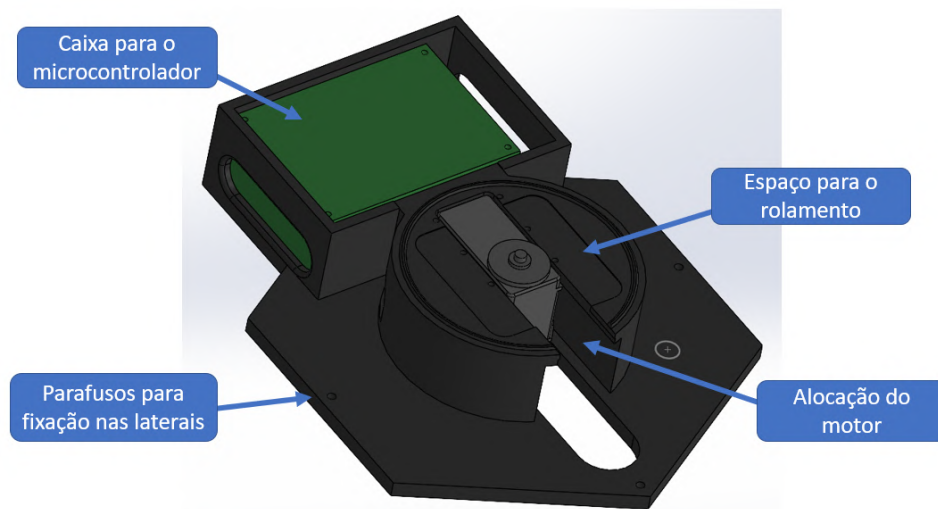
### 3.3 CONCEPÇÃO E PROJETO MECÂNICO

Robôs manipuladores possuem sua anatomia inspirada em grande medida no corpo humano, tanto na estrutura física quanto na forma de movimentação. Sendo assim, pode ser estabelecida uma analogia entre o robô desenvolvido e os componentes de um braço humano. Por isso, a concepção e projeto mecânico serão subdivididos nas seguintes partes: base, ombro, braço e antebraço, cotovelo, punho e garra.

#### 3.3.1 Base

A base é a parte inferior do robô manipulador, que fornece suporte e mobilidade, sendo capaz de sustentar os demais componentes e permitir o movimento em diferentes direções. A base projetada é uma base fixa, ou seja, pode ser parafusada em uma mesa ou qualquer outra superfície estável (Figura 12).

Figura 12 – Modelo CAD da base do robô.



Fonte: Autor, 2023.

Definiu-se um compartimento para que o microcontrolador pudesse ser armaze-

nado, protegendo-o de danos e facilitando o acesso para manutenção. Esse espaço fica agregado à estrutura da base e a partir dele são feitas as conexões com os motores e com a fonte de alimentação.

A rotação da base é de responsabilidade do primeiro motor, que fica localizado na parte interna da estrutura. Acima dele, é acoplado um mecanismo de rolamento, que é um suporte giratório comercial (Figura 13). Essa peça transmite o giro do *horn* do motor para os outros componentes durante a operação do robô.

Figura 13 – Mecanismo giratório da base.



Fonte: Autor, 2023.

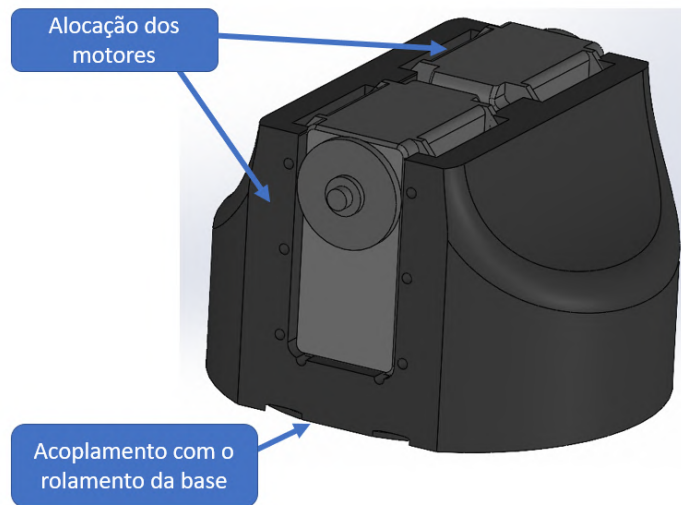
### 3.3.2 Ombro

O ombro é um mecanismo articulado importante para o robô manipulador, uma vez que desempenha a movimentação mais exigente da estrutura. Para garantir esse processo, são posicionados dois motores presos à peça (Figura 14). Na parte inferior, o ombro é conectado ao rolamento giratório da base.

### 3.3.3 Braço e Antebraço

O braço e o antebraço são idênticos. Sendo o braço localizado entre o ombro e o cotovelo. E o antebraço localizado entre o cotovelo e o punho. A estrutura é formada por dois componentes (Figura 15). A primeira parte é uma haste, em formato de "Y", que é acoplada ao *horn* dos motores (mostrada em preto na Figura 15). Assim, ela transmite o movimento de rotação do motor para gerar um novo posicionamento do robô. Já o segundo componente é um perfil de alumínio. Seus atributos de leveza para facilitar o manuseio e a instalação, resistência capaz de suportar cargas consideráveis, durabilidade e baixo custo são fatores determinantes para a utilização no projeto.

Figura 14 – Modelo CAD do ombro do robô.



Fonte: Autor, 2023.

Figura 15 – Modelo CAD do braço e antebraço do robô.

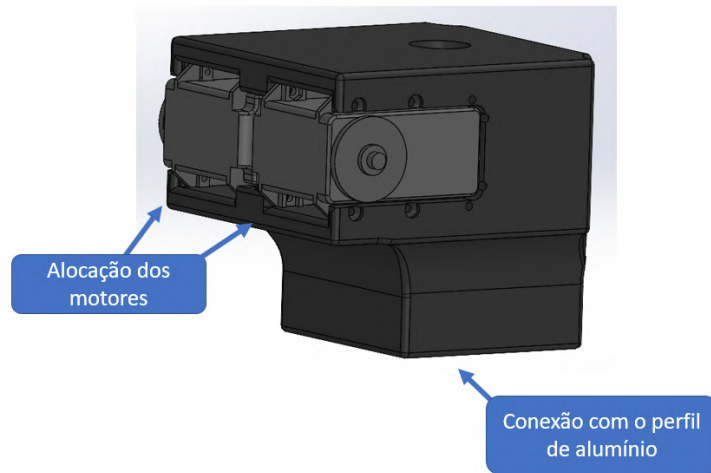


Fonte: Autor, 2023.

### 3.3.4 Cotovelo

O cotovelo é a peça responsável pela ligação entre o braço e antebraço do manipulador (Figura 16). Nessa junta, também são colocados dois motores, já que é necessário uma capacidade maior de torque para suportar a exigência de carga.

Figura 16 – Modelo CAD do cotovelo do robô.



Fonte: Autor, 2023.

### 3.3.5 Punho

O punho agrega dois graus de liberdade no robô, proporcionando tanto a movimentação no sentido vertical da garra, quanto a sua rotação (Figura 17).

Figura 17 – Modelo CAD do punho do robô.



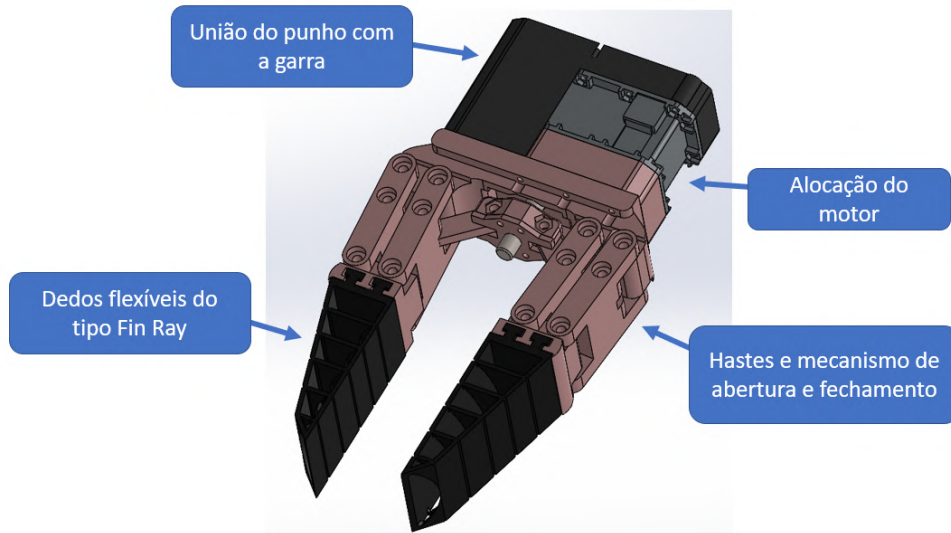
Fonte: Autor, 2023.

### 3.3.6 Garra

A garra, também denominada *gripper* é o componente com maior nível de detalhamento da estrutura (Figura 18). O projeto se destina a agarrar objetos de diferentes dimensões e geometrias, especialmente objetos sensíveis. Por isso, ela possui dois dedos flexíveis do tipo Fin Ray, que se adaptam ao formato do objeto, proporcionando uma

aderência eficiente. As nervuras em formato triangular nos dedos auxiliam o processo de deformação, fazendo com que haja maior deformação no centro e menor nas pontas e base. Essa composição amplia a área de contato com o objeto, proporcionando firmeza ao segurá-lo e evitando danos.

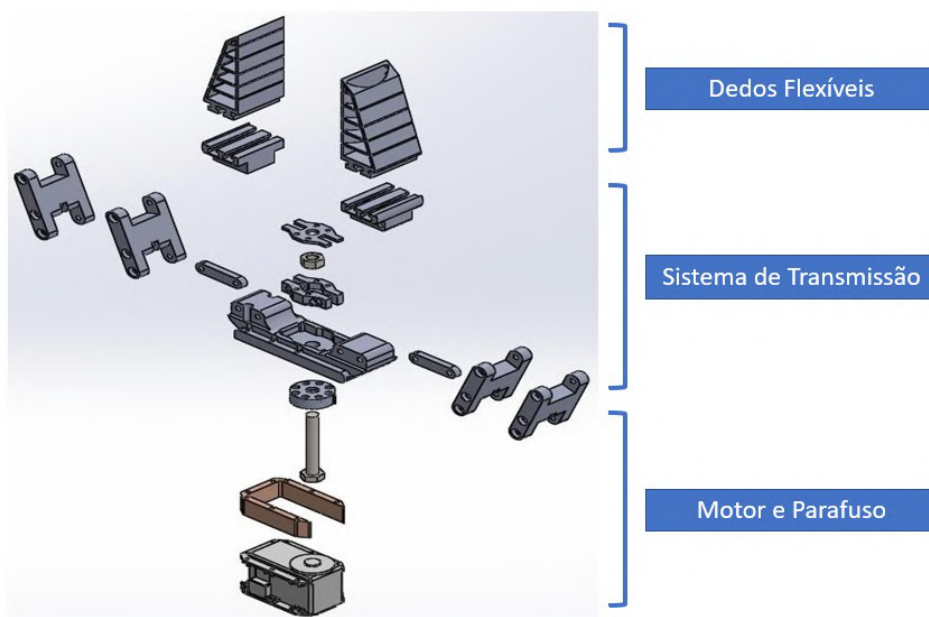
Figura 18 – Modelo CAD do garra do robô.



Fonte: Autor, 2023.

As outras peças da garra são rígidas e constituem o suporte dos dedos, os sistemas de transmissão de movimento e a união com o punho, conforme vista explodida da garra, na Figura 19.

Figura 19 – Vista explodida da garra.



Fonte: Autor, 2023.



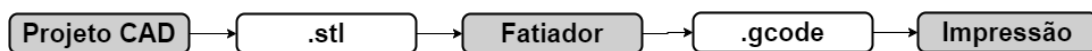
O mecanismo de abertura e fechamento funciona de forma tal que:

1. O motor é acionado, girando o *horn*;
2. Um parafuso está acoplado ao *horn* e rotaciona junto com ele;
3. O movimento do parafuso é transmitido para os dedos, através do sistema de hastes.

### 3.4 PROCEDIMENTOS DE IMPRESSÃO 3D

Com exceção do perfil de alumínio, que compõe o braço e o antebraço, e o rolamento giratório da base, todas as outras peças da estrutura mecânica são fabricadas via manufatura aditiva. O processo de impressão 3D das peças segue a sequência apresentada na Figura 20.

Figura 20 – Processo de impressão 3D.



Fonte: Autor, 2023.

Com base nos componentes estruturais do manipulador examinados na seção anterior, o projeto CAD é exportado em um arquivo no formato *.stl*. Esse arquivo é posteriormente importado em um *software* de fatiamento. O Cura é um *software* de código aberto desenvolvido pela Ultimaker com esse propósito. Devido à facilidade de acesso, suporte em impressoras variadas e possibilidade de utilização de configurações avançadas, essa foi a ferramenta selecionada para uso neste trabalho. A Figura 21 exibe a tela inicial desse programa ao abrir o arquivo referente à base do robô.

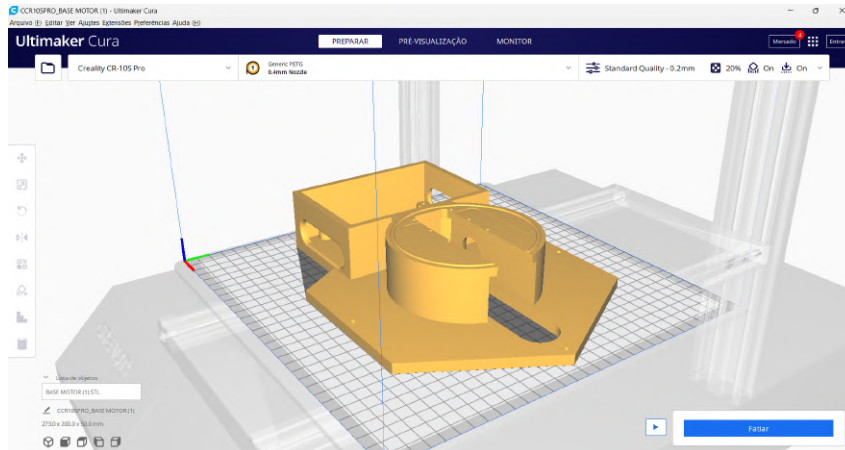
Nota-se na Figura 21 que a peça é retratada sobre a própria mesa de impressão. Além disso, o *software* permite a seleção de diferentes configurações, como o tipo de material, a largura da camada, atribuição de suporte, entre outras particularidades.

Após a inserção da configuração desejada, é necessário fatiar. O resultado desse processo de fatiamento é exibido na Figura 22. A malha de fatiamento formada expressa o caminho que o bico de impressão irá percorrer desde o início até a finalização da fabricação. No canto inferior direito, verificam-se fatores como o tempo de impressão e a quantidade de material a ser utilizado.

Como consequência do fatiamento, é gerado um arquivo *.gcode*, que é armazenado em um cartão de memória e levado diretamente até o equipamento no momento da impressão.

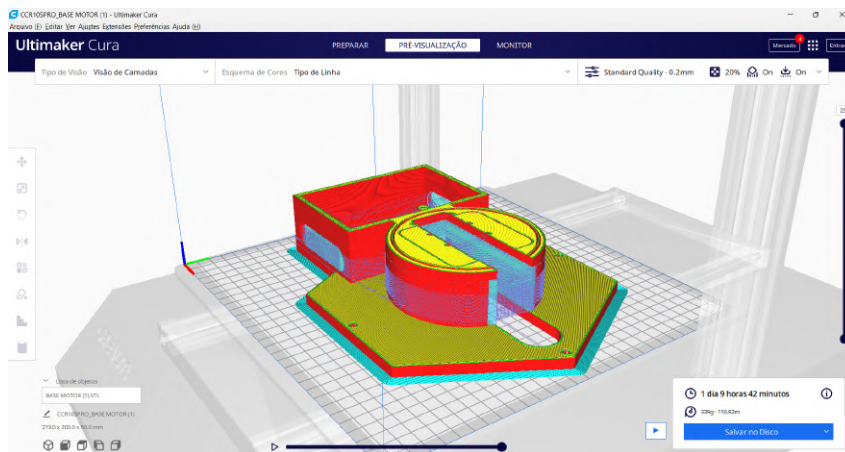
A preparação do material, o procedimento de impressão e o refino das peças construídas foram realizados no espaço do laboratório Fábrica CT no Centro de Tecnologia

Figura 21 – Visualização da tela inicial do Cura.



Fonte: Autor, 2023.

Figura 22 – Visualização do fatiamento realizado no Cura.



Fonte: Autor, 2023.

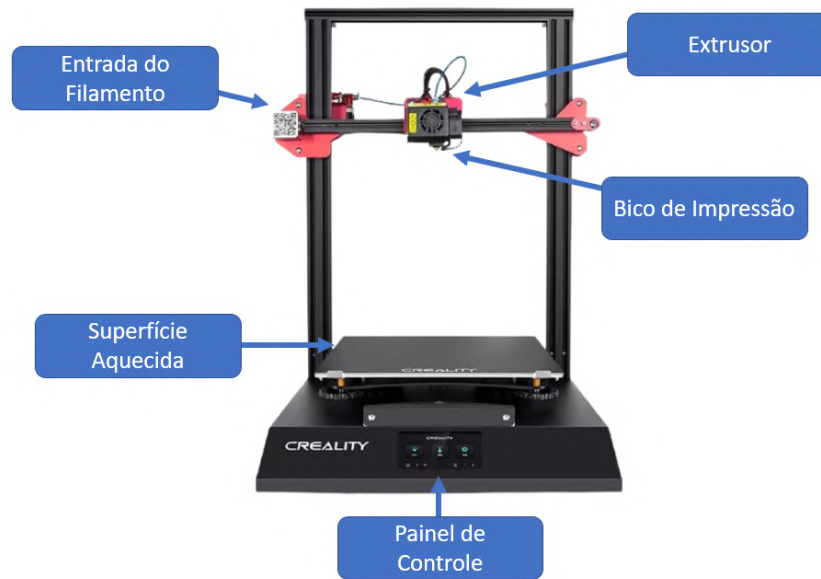
da UFSM. A impressora utilizada foi a Creality 3D CR-10S V2. Esse modelo apresenta excelente relação custo-benefício e versatilidade (Figura 23).

A Tabela 7 compreende algumas das principais características dessa impressora. Convém destacar a disponibilidade de mesa aquecida até 100 °C, a capacidade de uso de diferentes tipos de materiais e o fato de ser uma impressora aberta.

A escolha dos materiais mais apropriados para o caso deste trabalho foi embasada nessas características específicas de resistência, flexibilidade, transparência, temperatura de impressão e acabamento. Para as peças rígidas, deseja-se um material com excelente resistência mecânica, como o PETG. O projeto da garra determina a construção de dois dedos do tipo Fin Ray. Nesse caso, deseja-se um material flexível, que proporcione capacidade de deformação para que a garra se molde ao objeto que será agarrado. O material mais adequado para essa finalidade é, deste modo, o TPU.

Como forma de avaliar as configurações no *software* Cura que promoveriam os

Figura 23 – Identificação dos principais componentes da Creality 3D CR-10S V2.



Fonte: Adaptado de (CREALITY, 2021).

Tabela 7 – Especificações da impressora Creality 3D CR-10S V2.

Modelo	<b>Creality 3D CR-10S V2.</b>
Tecnologia	FDM
Precisão	$\pm 0,1$ mm
Espessura da camada	0,1 a 0,4 mm
Volume de impressão	300x300x400 mm
Número de extrusores	1
Temperatura do extrusor	$\leq 260$ °C
Temperatura da mesa	$\leq 100$ °C
Diâmetro do bico	0,4 mm
Velocidade de impressão	$\leq 180$ mm/s
Materiais compatíveis	PLA, TPU, PETG, etc

Fonte: Adaptado de (CREALITY, 2021).

melhores resultados de impressão, foram realizados testes com alguns protótipos. Os testes se destinam a ponderar sobre três fatores: calibração dos parâmetros, temperatura e velocidade de impressão:

- **Calibração dos parâmetros:** com o objetivo de garantir que as dimensões projetadas são as mesmas que as encontradas na medição do componente fabricado, foi impresso um cubo, com 20 mm em cada dimensão. Ao verificar que as dimensões correspondiam, confirmou-se a calibração correta da impressora;
- **Temperatura:** como a temperatura pode influenciar tanto o acabamento quanto a resistência da peça, foram impressas torres de temperatura, que consistem em torres formadas por pequenas seções impressas em temperaturas diferentes, de acordo

com o intervalo de temperaturas permitidas para o material. A observação dos estratos revela qual temperatura gera melhor desempenho;

- **Velocidade:** em geral, quanto mais devagar o processo, melhor a qualidade. Porém, como muitas peças do projeto levavam várias horas de impressão, é necessário estabelecer qual a melhor relação entre a velocidade e a qualidade, para minimizar o tempo dispendido. Dessa forma, foi realizado um teste usando uma pequena haste. Esse mesmo objeto foi impresso algumas vezes, em diferentes velocidades.

A análise desses fatores permitiu a determinação das configurações apropriadas para os componentes do projeto. A Tabela 8 relaciona os principais parâmetros utilizados tanto para o PETG quanto para o TPU.

Tabela 8 – Especificações dos parâmetros de impressão adotados.

Material	PETG	TPU
Altura da camada	0,2 mm	0,2 mm
Largura de Extrusão	0,4 mm	0,4 mm
Espessura de Parede	2,0 mm	0,8 mm
Espessuras Superior e Inferior	2,0 mm	0,8 mm
Camadas Superior e Inferior	10	4
Passar a Ferro	Desabilitado	Desabilitado
Densidade do Preenchimento	25 %	70 %
Padrão do Preenchimento	Grade	Triângulos
Temperatura de Impressão	242 °C	240 °C
Temperatura da Camada Inicial	250 °C	240 °C
Temperatura da Mesa	72 °C	60 °C
Velocidade de Impressão	60 mm/s	30 mm/s
Velocidade da Camada Inicial	30 mm/s	30 mm/s
Retração	Habilitado	Desabilitado
Refrigeração de Impressão	Habilitado	Habilitado
Suporte	Conforme Necessidade	Conforme Necessidade
Aderência à Mesa	Brim	Brim
Sequência de Impressão	Conforme Necessidade	Conforme Necessidade

Fonte: Autor, 2023.

A fabricação via manufatura aditiva muitas vezes envolve um ciclo iterativo de impressão, análise e ajustes. Através de experimentação e aprendizado contínuos, é possível aprimorar o processo e obter melhores resultados ao longo do tempo. No decorrer das impressões, notaram-se alterações que poderiam ser feitas no projeto. Essas alterações se referem tanto a peças que depois de impressas não correspondiam às necessidades do robô, quanto a peças que poderiam ter seu *design* otimizado. Com isso, também puderam ser propostas novas configurações de impressão que trariam maior robustez.

A aderência do objeto à mesa é um fator verificado logo na impressão das primeiras camadas e que determina se a peça será manufaturada corretamente. Algumas vezes,

ocorreu o desprendimento da peça, comprometendo a conclusão do processo. Para solucionar essa questão, foram avaliados aspectos como a seleção de Brim, que consiste em uma ou várias camadas finas de material que são criadas em torno da base do objeto, expandindo a área de contato com a mesa. Além disso, foi adotado o aumento da temperatura da mesa e a diminuição da velocidade da camada inicial.

Em alguns casos, é necessário adicionar suportes temporários durante a impressão, para garantir a estabilidade das partes suspensas da peça. Esses suportes ajudam a evitar problemas de adesão nessas áreas críticas. A orientação adequada da peça também é um fator a considerar, pois pode melhorar a aderência e até evitar a utilização de suporte.

### 3.5 MOTORES DYNAMIXEL

Os motores DYNAMIXEL são um tipo de servomotor muito utilizado em aplicações robóticas. Produzidos pela empresa coreana Robotis e conhecidos por sua alta precisão e controle avançado. Fornecem dados sobre o funcionamento do motor em tempo real, com informações sobre temperatura, posição, carga e grandezas elétricas (corrente e tensão).

Possuem diversos modos de operação além do controle por PWM (Modulação por Largura de Pulso), como o controle por posição, velocidade e corrente. Esses modos de operação tornam os motores DYNAMIXEL versáteis e adequados para uma ampla gama de aplicações. Além disso, esses motores podem ser conectados em cadeia através de um barramento de comunicação compartilhado, garantindo o controle de uma série de motores atuando em conjunto. Eles também são endereçáveis. Isso faz com que cada motor seja identificado e controlado individualmente dentro de um sistema ou rede que possui diversos motores, através de um ID (Identificador) específico. Esses servomotores são compostos por quatro elementos básicos: motor DC, placa de controle interna, conjunto de engrenagens e um encoder absoluto 360°.

A *Control Table* é uma estrutura de dados fundamental em atuadores DYNAMIXEL, funcionando como uma tabela de registros que armazena configurações relacionadas ao funcionamento do motor. Cada parâmetro na *Control Table* possui um endereço que pode ser lido ou modificado pelo usuário.

No manipulador desenvolvido, optou-se por utilizar dois modelos de motores. O MX-106R, nos locais de maior exigência de carga e o MX-64R, para a abertura e fechamento da garra. Um disco metálico, mais conhecido como *horn*, vai acoplado na engrenagem do servomotor. Ele possibilita a fixação aos elementos mecânicos da estrutura, como as peças de impressão 3D, que foram projetadas de forma a acomodar os parafusos para colocação do *horn* e permitir a transmissão dos movimentos. A Figura 24 ilustra o posicionamento do *horn* e identifica o conector destinado a comunicação serial.

Figura 24 – Identificação do *horn* e da comunicação serial no motor.



Fonte: Adaptado de (ROBOTIS, 2023).

Existem dois padrões de comunicação nos motores DYNAMIXEL. A comunicação serial pode ser TTL ou RS-485. A primeira é realizada com três pinos e a última com quatro pinos. A letra "T" no final do nome do modelo de cada motor se refere a TTL, já a letra "R" indica RS-485. O detalhamento das especificações do DYNAMIXEL MX-106R e do DYNAMIXEL MX-64R são dados nas Tabelas 9 e 10, respectivamente.

Tabela 9 – Especificações do servomotor DYNAMIXEL MX-106R.

Modelo	<b>DYNAMIXEL MX-106R</b>
Microcontrolador	ARM CORTEX-M3 (72 [MHz], 32Bit)
Sensor de posição	Contactless absolute encoder (12Bit, 360°)
Motor	Coreless
Baudrate	7.834 [bps] - 3 [Mbps]
Algoritmo de controle	PID
Resolução	4096 [pulse/rev]
Backlash	20 [arcmin] (0,33°)
Massa	153 [g]
Gear Ratio	225 : 1
Stall Torque	8,4 [N.m] (a 12,0 [V], 5,2 [A])
Velocidade a vazio	45 [rev/min] (a 12,0 [V])
Comunicação serial	RS-485

Fonte: Adaptado de (ROBOTIS, 2023).

Pode ser estabelecida uma comparação entre os modelos de servomotores utilizados na construção deste manipulador e os aplicados no Viper X 300. Primeiramente, os dois modelos do Viper X 300 são do tipo TTL, enquanto os do projeto são do tipo RS-485. Para as juntas que exigem maior capacidade, como a base, o ombro, o cotovelo e o ângulo do punho, o Viper X 300 utiliza o DYNAMIXEL XM540-W270-T e o projeto realizado utiliza o DYNAMIXEL MX-106R. Os dois modelos possuem características semelhantes, sendo o XM540-W270-T parte de uma linha mais recente e robusta de motores, com torque de 10,6 Nm a 12 V e 4,4 A e massa de 165 g. Pode ser comandado no modo de controle por

Tabela 10 – Especificações do servomotor DYNAMIXEL MX-64R.

Modelo	<b>DYNAMIXEL MX-64R</b>
Microcontrolador	ARM CORTEX-M3 (72 [MHz], 32Bit)
Sensor de posição	Contactless absolute encoder (12Bit, 360°)
Motor	Coreless
Baudrate	7.834 [bps] - 3 [Mbps]
Algoritmo de controle	PID
Resolução	4096 [pulsos/rev]
Backlash	20 [arcmin] (0,33°)
Massa	126 [g]
Gear Ratio	200 : 1
Stall Torque	6,0 [N.m] (a 12,0 [V], 4,1 [A])
Velocidade a vazio	63 [rev/min] (a 12,0 [V])
Comunicação serial	RS-485

Fonte: Adaptado de (ROBOTIS, 2023).

corrente e possui maior flexibilidade para cabeamento. Já o MX-106R entrega um torque de 8,4 Nm a 12 V e 5,2 A e possui massa de 153 g.

Nas juntas de rotação do punho e na garra, o Viper X 300 utiliza o DYNAMIXEL XM-430-W350-T, que possui torque de 4,1 Nm a 12 V e 2,3 A e massa de 82 g. No caso do projeto do manipulador, optou-se por utilizar o MX-106R para a rotação do punho, uma vez que havia disponibilidade. E o DYNAMIXEL MX-64R para a garra, que entrega um torque de 6 Nm a 12 V e 4,1 A e possui massa de 126 g.

Como citado anteriormente, o ID é um número exclusivo atribuído a cada motor no barramento de comunicação. Ele permite ao sistema distinguir e enviar comandos para um motor em específico, ignorando os outros motores conectados. O ID pode ser definido como um número de 0 até 252. Na Tabela 11, é apresentada a designação dos IDs de cada motor, junto com a informação de onde está alocado.

Tabela 11 – Relação dos motores utilizados e respectivos IDs.

ID	Nomenclatura no Software	Alocação	Modelo	Rolamento Secundário
1	<i>waist</i>	Base	MX-106R	Não
2	<i>shoulder</i>	Ombro	MX-106R	Não
3	<i>shoulder_shadow</i>	Ombro	MX-106R	Não
4	<i>elbow</i>	Cotovelo	MX-106R	Não
5	<i>elbow_shadow</i>	Cotovelo	MX-106R	Não
6	<i>wrist_angle</i>	Ângulo do Punho	MX-106R	Sim
7	<i>wrist_rotate</i>	Rotação do Punho	MX-106R	Sim
8	<i>gripper</i>	Garra	MX-64R	Sim

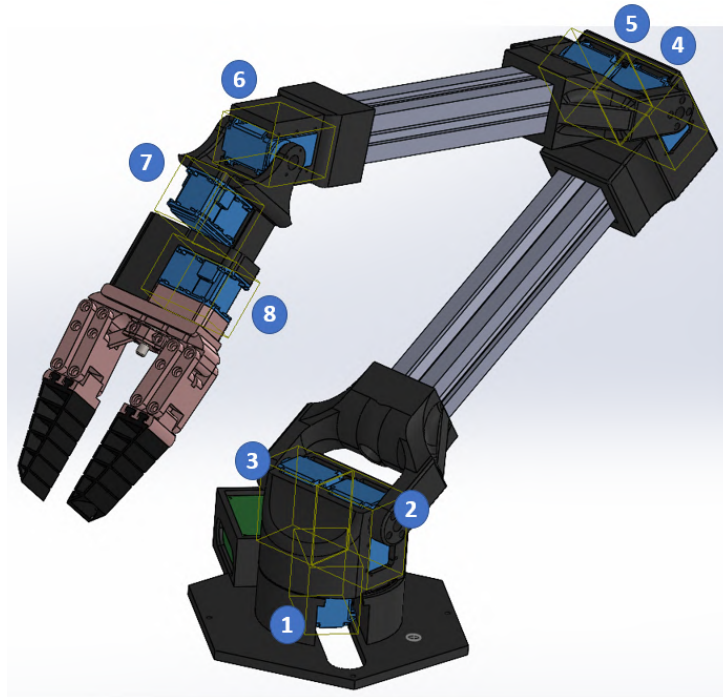
Fonte: Autor, 2023.

Além do *horn*, presente em todos os motores, em algumas juntas também foi utilizado um rolamento secundário no lado oposto, também referido como *bearing set*. Ele

é incorporado quando há fixação de peças nos dois lados de um mesmo motor e serve para que o movimento seja transmitido de maneira mais suave, fazendo com que ambas as conexões acompanhem a rotação.

A Figura 25 relaciona o ID de cada motor com a sua alocação dentro da estrutura completa do robô manipulador.

Figura 25 – Identificação do ID de cada motor no projeto CAD.



Fonte: Autor, 2023.

### 3.6 MICROCONTROLADOR OPENCR

O OpenCR é um microcontrolador desenvolvido pela Robotis, projetado especialmente para aplicações de robótica. Uma das principais características é que foi desenvolvido para ser um sistema embarcado de ROS (*Robot Operating System*).

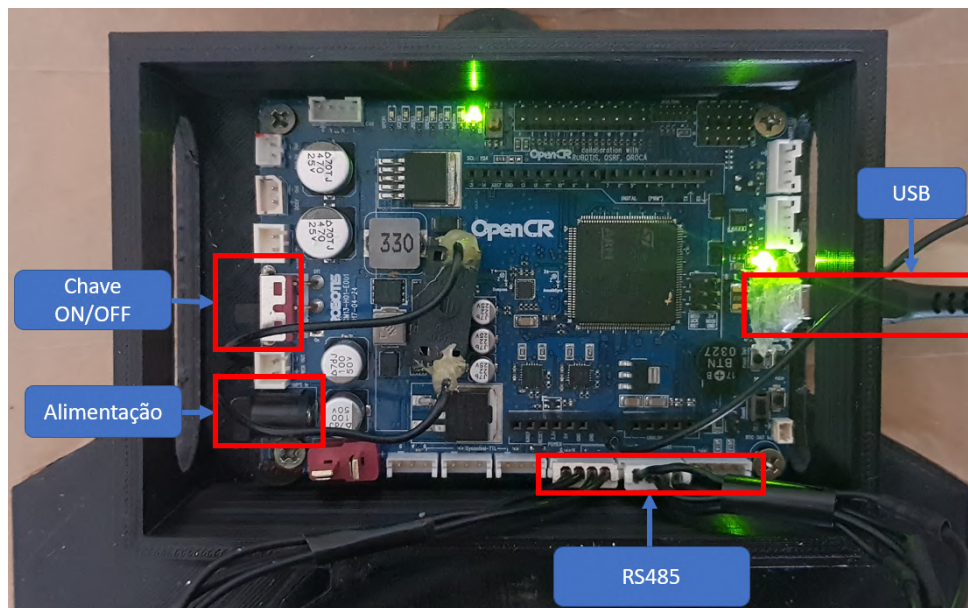
A placa de desenvolvimento combina um *chip* STM32F7 baseado no ARM Cortex-M7 de 32 bits, com recursos de conectividade e controle. O OpenCR possui uma variedade de interfaces de comunicação, como UART, SPI e I2C e interfaces dedicadas para controlar motores DYNAMIXEL.

O dispositivo possui versatilidade de ambientes de desenvolvimento. É programável usando a IDE do Arduino, na qual podem ser escritos códigos em C++ e utilizadas as bibliotecas para controlar os componentes conectados. Também pode ser programado via ferramentas tradicionais de *firmware* ou acessado diretamente pelo ROS.



As entradas RS-485 são dedicadas à conexão dos motores DYNAMIXEL. A chave ON/OFF serve para ligar e desligar a placa. A entrada de alimentação, que por padrão é de 12V e 4,5A, pode ser utilizada para alimentar os motores DYNAMIXEL conectados. Essa funcionalidade foi utilizada nos testes e configurações individuais de cada motor. Porém, na estrutura do robô completo, o OpenCR é alimentado em 5V pela conexão USB. E é utilizado um barramento externo para alimentação dos motores. O pino de *ground* da placa também foi utilizado para aterramento no circuito completo do manipulador. O OpenCR fica armazenado em uma caixa própria, acoplada na base do robô (Figura 26).

Figura 26 – Placa OpenCR e apontamento dos componentes mais relevantes para o projeto.



Fonte: Autor, 2023.

### 3.7 MALHA ELÉTRICA

Conforme mencionado anteriormente, os motores DYNAMIXEL se comunicam através do protocolo RS-485. Nessa rede, os conectores possuem quatro pinos: terra (GND), alimentação (VDD), entrada de dados (DADOS +) e saída de dados (DADOS -). A Figura 27 esclarece a respectiva pinagem.

Os motores utilizados possuem os conectores RS-485 em ambos os lados. Essa característica permite que eles sejam conectados em cadeia (ligação em série). Visto que o ID é único para cada motor da rede, os pacotes de instruções podem ser endereçados diretamente para cada um dos motores, mesmo que circulem pela rede inteira (Figura 28).

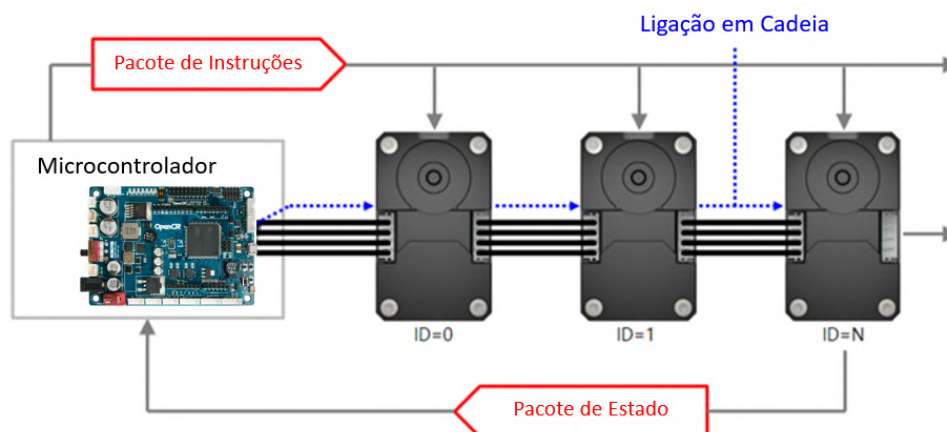
Os pacotes de instruções são enviados pelo microcontrolador, o qual também recebe um retorno na forma de um pacote de estado, que indica funções como a posição do

Figura 27 – Pinagem da comunicação RS-485.

Item	RS-485
Pinagem	1 : GND 2 : VDD 3 : DADOS + 4 : DADOS -
Diagrama	
Conector	

Fonte: Adaptado de (ROBOTIS, 2023).

Figura 28 – Esquema de ligação em cadeia dos motores Dynamixel.

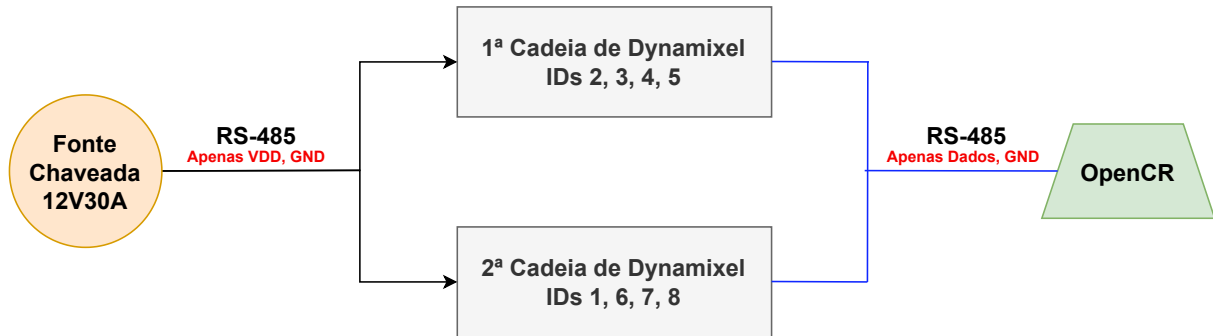


Fonte: Adaptado de (ROBOTIS, 2023).

motor, a temperatura, a corrente, entre outras propriedades.

Dessa forma, pode ser estabelecida a conexão completa dos motores do manipulador. Os oito motores foram subdivididos em duas cadeias (Figura 29). A primeira é composta por quatro motores MX-106R, cujos IDs são 2, 3, 4, 5. E a segunda consiste em três motores MX-106R e um motor MX-64R, cujos IDs são 1, 6, 7, 8. Nesse sistema, o OpenCR troca, através do protocolo RS-485, apenas dados (DADOS+ e DADOS -) e *ground*. Enquanto isso, a alimentação dos motores é feita por uma fonte chaveada externa de 12V e 30A, a qual é conectada na outra extremidade da cadeia.

Figura 29 – Esquema de conexão dos componentes.



Fonte: Autor, 2023.

### 3.8 COMPONENTES DE SOFTWARE

A arquitetura de *software* de um robô manipulador da *Interbotix* é projetada para aproveitar ao máximo as capacidades dos motores DYNAMIXEL. A integração com os motores permite que o *software* controle com precisão os movimentos do robô, receba *feedback* em tempo real e configure parâmetros específicos do motor, como velocidade e posição.

Essa arquitetura é modular, dividida em camadas que separam o controle de baixo nível dos motores, a lógica de alto nível do manipulador e as interfaces com o usuário e outros sistemas. Isso permite que desenvolvedores e pesquisadores personalizem e expandam o *software*, adaptando-o às necessidades específicas de suas aplicações.

Primeiramente, serão apresentadas as configurações utilizando as ferramentas disponibilizadas pela Robotis para os motores DYNAMIXEL (DYNAMIXEL Wizard e DYNAMIXEL SDK). Após, serão exploradas as propriedades dos pacotes disponibilizados pela *Interbotix* (implementados no ROS e Moveit).

#### 3.8.1 DYNAMIXEL Wizard 2.0

O DYNAMIXEL Wizard 2.0 é uma ferramenta otimizada projetada especificamente para gerenciar os motores DYNAMIXEL em diversos sistemas operacionais. Esta ferramenta oferece uma série de recursos trabalhar com esses motores, como: atualização de *firmware* do DYNAMIXEL, diagnóstico de problemas ou falhas, configuração e teste de parâmetros, plotagem de dados em tempo real e geração e monitoramento de pacotes de dados (ROBOTIS, 2023).

Com base nesse instrumento, procedeu-se com a configuração dos IDs de cada um dos motores, assim como os parâmetros mais básicos. Ele também permite a movimentação dos motores para testes simples, especialmente quando se deseja lidar com um único

atuador.

A Figura 30 exibe a interface do programa, destinada à configuração e monitoramento. No lado esquerdo, há uma lista de atuadores conectados, cada um com seu respectivo ID. Enquanto no centro, uma tabela detalha os diversos parâmetros do atuador selecionado, incluindo informações como *Baud Rate*, *Drive Mode* e *Temperature Limit*. À direita, métricas em tempo real, como *Position*, e *Voltage* são apresentadas junto a botões para selecionar modo de controle, torque e acionamento do LED. Verifica-se também uma imagem modelo do motor na parte inferior direita.

Figura 30 – Interface do DYNAMIXEL Wizard 2.0.

Address	Item	Decimal	Hex	Actual
0	Model Number	321	0x0141	MX-106(2.0)
2	Model Information	0	0x00000000	
6	Firmware Version	42	0x2A	
7	ID	1	0x01	ID 1
8	Baud Rate (Bus)	3	0x03	1 Mbps
9	Return Delay Time	0	0x00	0 [µsec]
10	Drive Mode	4	0x04	
11	Operating Mode	3	0x03	Position control
12	Secondary(Shadow) ID	255	0xFF	Secondary ID 255 disable
13	Protocol Type	2	0x02	Protocol 2.0
20	Homing Offset	0	0x00000000	0.00 [°]
24	Moving Threshold	10	0x0000000A	2.29 [rev/min]
31	Temperature Limit	80	0x50	80 [°C]
32	Max Voltage Limit	160	0x0080	16.00 [V]
34	Min Voltage Limit	95	0x005F	9.50 [V]
36	PWM Limit	885	0x0375	100.00 [%]
38	Current Limit	2847	0x07FF	6877.92 [mA]
44	Velocity Limit	131	0x00000083	30.00 [rev/min]
48	Max Position Limit	4095	0x00000FFF	359.91 [°]
52	Min Position Limit	2583	0x000009C7	219.99 [°]
63	Shutdown	52	0x34	
64	Torque Enable	0	0x00	OFF
65	LED	0	0x00	OFF

Fonte: Autor, 2023.

### 3.8.2 DYNAMIXEL SDK

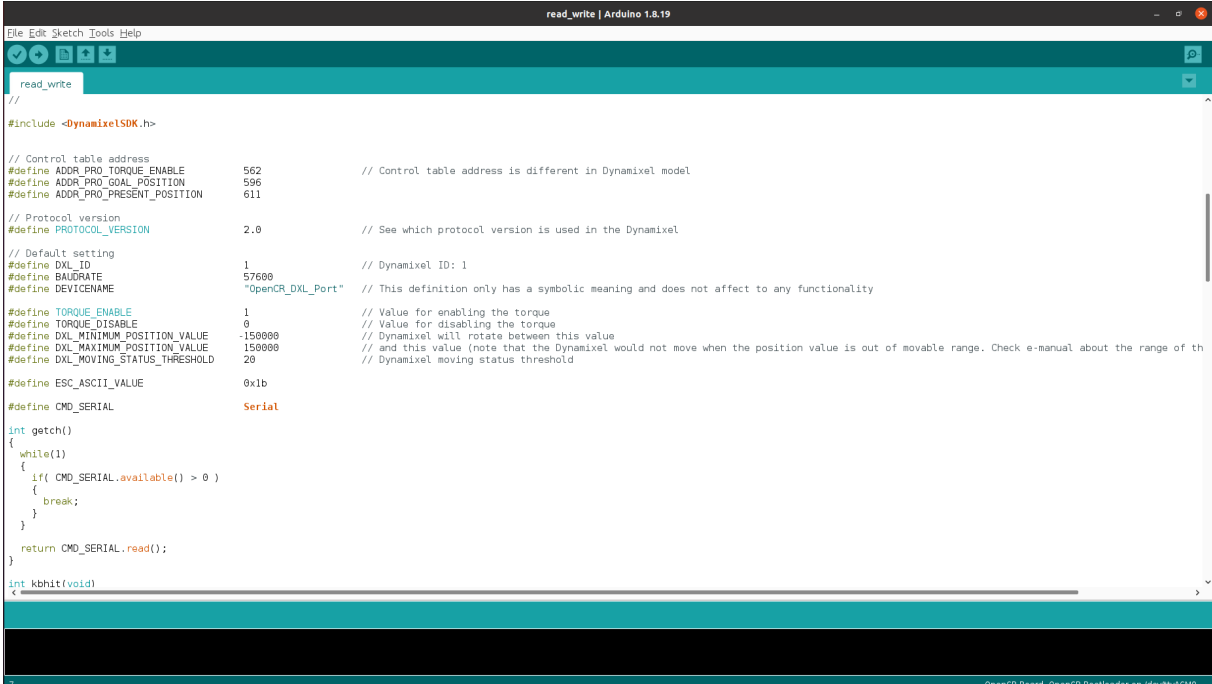
O DYNAMIXEL SDK é um kit de desenvolvimento de *software* projetado pela Robotis para facilitar o controle dos motores DYNAMIXEL através da comunicação por pacotes. É compatível com todas as séries e protocolos DYNAMIXEL, além de ser versátil em termos de dispositivos e sistemas operacionais em que pode ser utilizado. Entre as linguagens de programação suportadas, constam: C, C++, C#, Python, Java, MATLAB, LabView e ROS.

O SDK fornece funções para acessar e modificar a *Control Table*, assim como habilitar ou desabilitar recursos específicos dos atuadores e trabalhar com os modos de controle definindo posição, velocidade, corrente e torque desejados. Dessa forma, viabiliza o esta-

beleciamento de lógicas de movimentação personalizadas para o conjunto de motores de um sistema robótico.

Como demonstração de utilização, a Figura 31 mostra o ambiente de desenvolvimento do Arduino, com um código-fonte básico de leitura e escrita nos atuadores DYNAMIXEL. A partir da biblioteca DYNAMIXEL SDK são definidos diversos endereços da tabela de controle e o nome do dispositivo usado, que é o OpenCR. O código é estruturado para escrever uma posição desejada e, em seguida, continuar lendo a posição atual até que o motor pare de se mover.

Figura 31 – Interface do DYNAMIXEL SDK utilizado pelo Arduino IDE.



```

read_write | Arduino 1.8.19
File Edit Sketch Tools Help
read_write
//
#include <DynamixelSDK.h>

// Control table address
#define ADDR_PRO_TORQUE_ENABLE 562 // Control table address is different in Dynamixel model
#define ADDR_PRO_GOAL_POSITION 596
#define ADDR_PRO_PRESENT_POSITION 611

// Protocol version
#define PROTOCOL_VERSION 2.0 // See which protocol version is used in the Dynamixel

// Default setting
#define DXL_ID 1 // Dynamixel ID: 1
#define BAUDRATE 57600
#define DEVICENAME "OpenCR_DXL_Port" // This definition only has a symbolic meaning and does not affect to any functionality

#define TORQUE_ENABLE 1 // Value for enabling the torque
#define TORQUE_DISABLE 0 // Value for disabling the torque
#define DXL_MINIMUM_POSITION_VALUE 150000 // Dynamixel will rotate between this value
#define DXL_MAXIMUM_POSITION_VALUE 150000 // and this value (note that the Dynamixel would not move when the position value is out of movable range. Check e-manual about the range of th
#define DXL_MOVING_STATUS_THRESHOLD 20 // Dynamixel moving status threshold

#define ESC_ASCII_VALUE 0x1b

#define CMD_SERIAL Serial

int getch()
{
  while(1)
  {
    if( CMD_SERIAL.available() > 0 )
    {
      break;
    }
  }
  return CMD_SERIAL.read();
}

int kbhit(void)
<
7
OpenCR Board, OpenCR Bootloader on /dev/ttyACM0

```

Fonte: Autor, 2023.

Verifica-se que o DYNAMIXEL Wizard e o DYNAMIXEL SDK se adaptam a diferentes etapas do estágio de desenvolvimento. O DYNAMIXEL Wizard é ótimo para configuração e testes iniciais, conta com interface gráfica, e pode ser utilizado para alterar parâmetros de um motor de maneira simples e direta. Por outro lado, o DYNAMIXEL SDK viabiliza aplicações mais complexas, como o movimento sincronizado entre múltiplos servos, além de proporcionar a flexibilidade de um ambiente de programação.

### 3.8.3 ROS

Um sistema robótico pode ser formado por uma variedade de componentes, como motores, sensores, *software* e baterias. Para que o robô consiga operar de maneira ade-

quada, todas essas partes devem funcionar em conjunto. O *Robot Operating System* ou ROS é um *software* de código aberto que permite a integração desses componentes usando um modelo de mensagens de publicação/assinatura. Além disso, ele suporta interfaces de *hardware* para os principais componentes robóticos como câmeras, LIDARs e controladores. Essa arquitetura modular também garante flexibilidade para trabalhar com componentes de diferentes fornecedores ao mesmo tempo (ROS, 2023).

É importante notar que o ROS não é um sistema operacional convencional, voltado para gerenciamento e agendamento. Em vez disso, é um conjunto de bibliotecas e ferramentas projetadas para facilitar o desenvolvimento de sistemas robóticos.

Os módulos ROS podem ser escritos em C++, Python, MATLAB, Java, entre outras linguagens. Como a arquitetura é distribuída, programas podem ser executados em vários computadores e se comunicarem pela rede através do protocolo TCP/IP.

Os principais conceitos necessários para a compreensão do ROS são:

- **Nó:** programa executável de propósito único, que é compilado, executado e gerenciado individualmente;
- **ROS Master:** é o nó mestre, que gerencia a comunicação entre nós;
- **Tópico:** é o fluxo de mensagem através do qual os nós se comunicam, podendo publicar ou se inscrever em um tópico;
- **Mensagem:** é a estrutura de dados que define o tipo de um tópico;
- **Serviço:** é composto por um servidor, que anuncia o serviço, e um cliente, que acessa esse serviço. Nesse caso, o robô deve esperar até que o serviço termine para fazer outra tarefa;
- **Ação:** é semelhante ao serviço, porém, uma ação é uma chamada assíncrona para as funcionalidades de outro nó, portanto o robô pode fazer outra tarefa enquanto executa aquela ação;
- **Rviz:** é a ferramenta de visualização 3D para o ROS, utilizada para visualizar informações de sensores e o estado de um robô, tanto em simulação, quanto em tempo real, caso esteja conectado.
- **Simulador Gazebo:** serve para simulação 3D de sistemas dinâmicos, contendo um banco de dados com diferentes robôs e ambientes.

Um programa ROS é constituído por vários nós independentes agrupados em pacotes. Cada nó é uma representação abstrata de um componente do sistema, seja ele um sensor, um atuador ou outro elemento. Com isso, programas individuais se comunicam por API definida (*peer-to-peer*). Além disso, existem as funcionalidades de estruturas

como serviço e ação. Todos esses fatores tornam o ROS adaptável às necessidades do usuário.

### 3.8.4 MoveIt

O MoveIt é um *framework* para o planejamento de movimentos em robótica, amplamente utilizado para tarefas de manipulação. É capaz de gerar trajetórias e executá-las em controladores, resolver cinemática inversa, conectar-se a sensores de percepção 3D e verificar colisões (PickNik Robotics, 2023). Essas funcionalidades fazem com que ele seja aplicado tanto na indústria quanto em instituições de pesquisa.

No MoveIt há um nó central chamado *move\_group* com o qual os usuários podem interagir através de suas ações e serviços usando uma GUI no Rviz (visualizador ROS) ou por meio de código, em C++ ou Python. Em Python, o pacote *moveit\_commander* é usado para acessar as capacidades do *move\_group*. Existem três maneiras de planejar e executar movimentos: definindo uma configuração alvo para as juntas, definindo uma pose alvo para o efetuador final ou planejando um caminho cartesiano. O resultado do planejamento de movimentos é uma mensagem *RobotTrajectory*, que contém uma sequência de *JointTrajectoryPoints*. Cada ponto tem valores de juntas, velocidades, acelerações e esforços, além do tempo para alcançar aquele ponto. Essa mensagem é então enviada aos controladores ROS para executar a trajetória planejada (Malvido Fresnillo et al., 2023).

O amplo uso dessa ferramenta é viabilizado por arquivos de configuração específicos, que permitem descrever um robô no MoveIt. Os dois principais arquivos são o URDF e o SRDF, que serão abordados nas subseções (4.6.2) e (4.6.4), respectivamente.

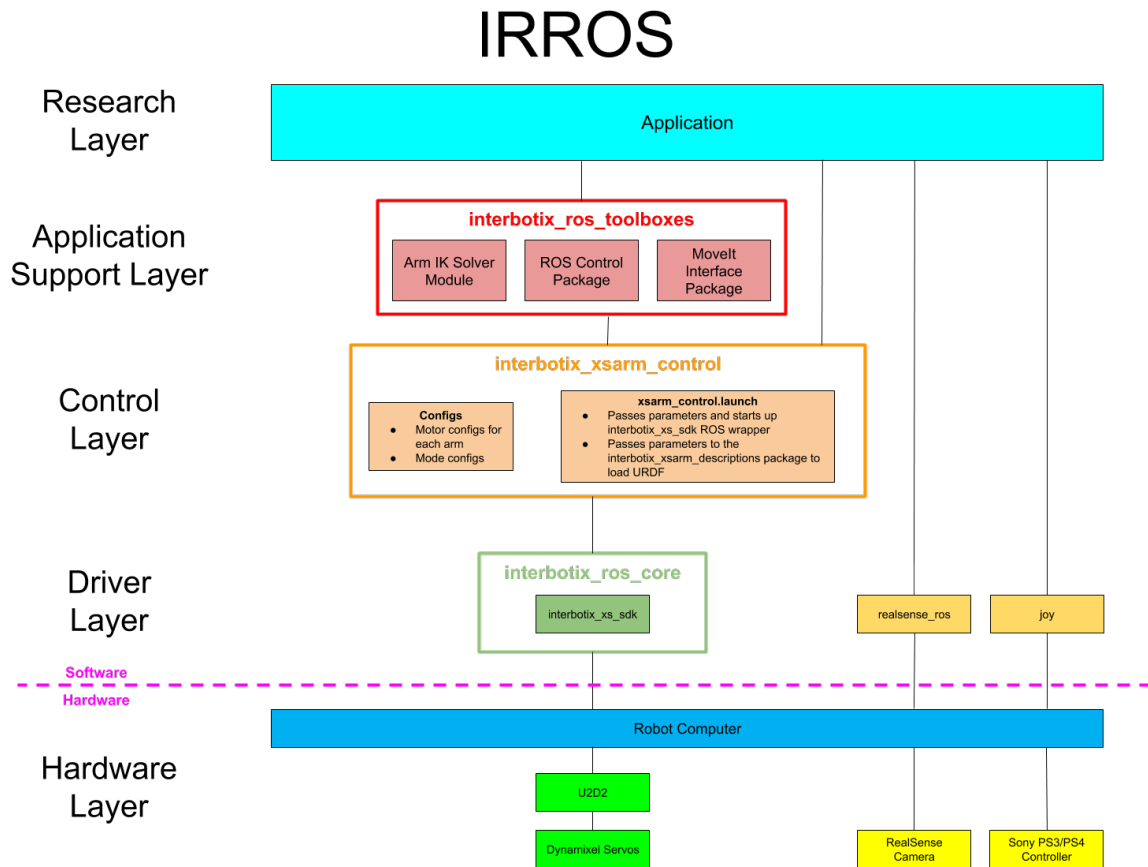
### 3.8.5 Arquitetura de *software* para manipuladores da *Interbotix*

Os robôs da *Interbotix* são voltados para o mercado de pesquisa e educação. Dessa forma, eles procuram fornecer uma abstração para aspectos de configuração de *software* e *hardware*, permitindo que o usuário trabalhe em uma camada de alto nível para o desenvolvimento de aplicações baseadas nos seus manipuladores robóticos. E essa é uma característica presente na sua arquitetura.

A *Interbotix* propõe uma estrutura para facilitar a integração dos atuadores de baixo nível com o código de nível superior denominada de *Interbotix Research Robotics Open Standard* (IRROS). Essa abordagem envolve um padrão de nomenclatura e construção dos pacotes que é aplicado para os diferentes robôs da empresa (Wiznitzer et al., 2023).

Existem cinco camadas dentro dessa estrutura que conferem níveis de abstração particulares e que estabelecem a interface entre *software* e *hardware* (Figura 32).



Figura 32 – Arquitetura de *software* para manipuladores da *Interbotix* (IRROS).

Fonte: (Wiznitzer et al., 2023).

Abaixo, são descritos os seus aspectos fundamentais:

- **Hardware Layer** (Camada de Hardware): Trata da configuração física, especificamente quais atuadores e sensores estão conectados com o robô. Os braços da *Interbotix* são construídos com os motores da série X da DYNAMIXEL e com uma placa U2D2 para comunicação e execução dos comandos de controle;
- **Driver Layer** (Camada de driver): Contém as interfaces do ROS para atuadores e sensores, permitindo enviar e receber dados desses dispositivos. Ela abstrai o código de baixo nível, como protocolos seriais e endereços. O repositório *interbotix\_ros\_core*, que engloba o *interbotix\_xs\_sdk*, está incluído nessa camada. Esse pacote foi construído em cima do DYNAMIXEL SDK, disponibilizado pela Robotis e apresentado na subseção (3.8.2);
- **Control Layer** (Camada de Controle): no caso dos manipuladores, refere-se ao pacote *interbotix\_xsarm\_control*, que facilita o controle e a configuração. Esse pacote possui dois componentes principais: o diretório *config*, que armazena arquivos com



parâmetros para os atuadores e sensores do robô, como por exemplo o nome das juntas; e o arquivo de inicialização, que transmite esses parâmetros para os atuadores/sensores. A vantagem é que todas as variáveis que precisam ser alteradas ficam alocadas em apenas um lugar;

- **Application Support Layer** (Camada de suporte de aplicação): Essa camada existe para fornecer módulos Python ou pacotes ROS de suporte para facilitar o trabalho em um nível mais elevado. É aqui que os arquivos relacionados ao MoveIt podem ser encontrados, assim como um solucionador de cinemática inversa;
- **Research Layer** (Camada para Pesquisa): É onde o usuário final programa o próprio código para manipulação, navegação, visão computacional, aprendizado de máquina ou outras aplicações. Os *scripts* dentro do diretório de exemplos podem ser descritos como código dessa camada.

O repositório da *Interbotix* no *Github* e a documentação especificando os produtos da empresa são fontes para compreender mais profundamente essa estrutura (Wiznitzer et al., 2023).

Em relação aos aspectos de *hardware* do robô manipulador desenvolvido neste trabalho, foram alterados os modelos dos motores e do controlador. Para os motores, em vez da série X, foram utilizados motores da série MX, conforme já abordado na subseção (3.5) e para o controlador, em vez do U2D2, foi utilizado o OpenCR, como visto na subseção (3.6).

Já em relação aos aspectos de *software*, foram necessárias algumas configurações adicionais e adaptações dos pacotes, de forma a conseguir utilizar os recursos disponibilizados pela *Interbotix*. Esses aspectos serão vistos na subseção (4.6), a seguir, que trata de programação.

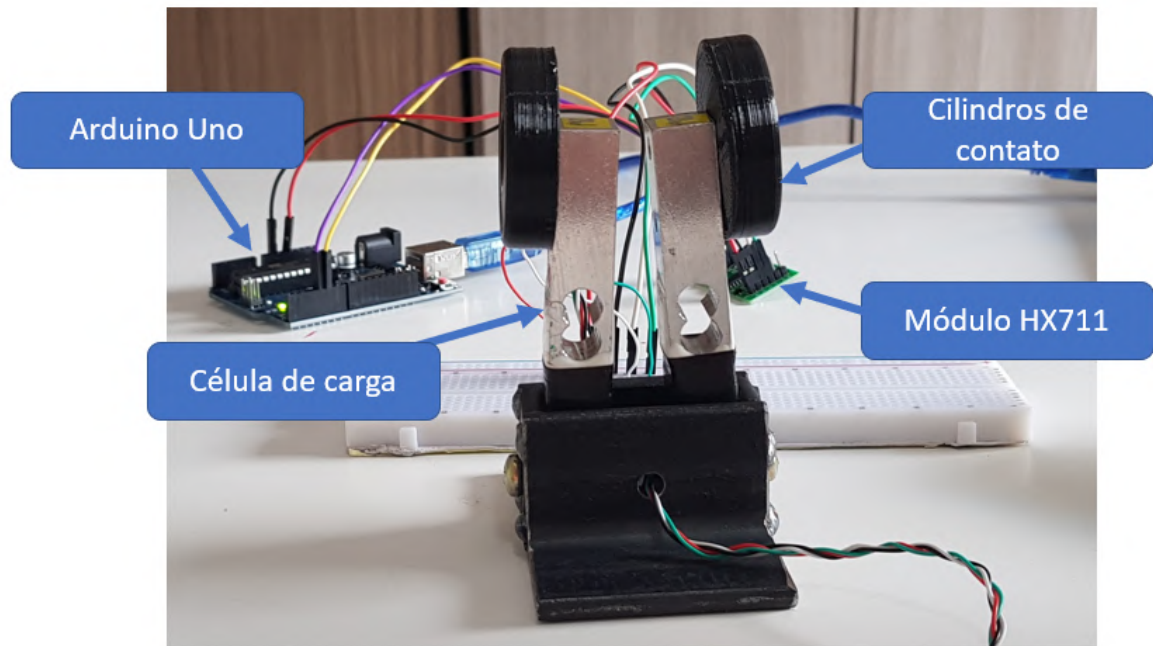
Cabe notar que há diferentes níveis de abstração que podem ser adotados conforme as necessidades do usuário. Próximo ao *hardware*, há o DYNAMIXEL SDK, que permite comandar, por exemplo, as posições das juntas do robô diretamente. Por outro lado, nas camadas de aplicação, com a utilização do MoveIt, é possível desenvolver aplicações mais complexas, em que se deseja, por exemplo, definir a trajetória que o robô deve executar para agarrar uma peça detectada por uma câmera conectada ao sistema. A arquitetura apresentada viabiliza a transição entre essas diferentes camadas.

### 3.9 BANCADA EXPERIMENTAL

Para fins de validação experimental dos esforços da garra, foi concebida uma estrutura por meio da qual fosse possível medir a força executada pela garra e simular o

comportamento real de movimentação de um objeto. O ensaio consistiu em realizar o fechamento e abertura da garra, aplicando o máximo esforço do motor. Esse ensaio utilizou um suporte de aço para fixação de duas células de carga. Na Figura 33 é apresentado o dispositivo construído.

Figura 33 – Dispositivo de aquisição no ensaio de aperto máximo



Fonte: Autor, 2023.

A bancada para testes contou com os seguintes materiais:

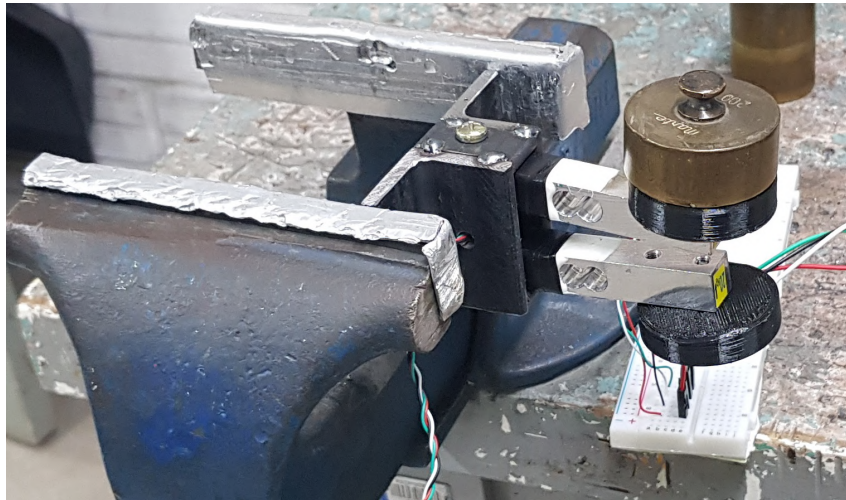
- Suporte de aço para fixação das células de carga;
- Duas células de carga: sensor de peso com carga nominal de 20kg, de liga de alumínio, com tensão de funcionamento de 5V;
- Dois cilindros impressos em tecnologia 3D, no material PETG, para ser a superfície de contato entre a garra e a célula de carga;
- Microcontrolador Arduino Uno: com tempo de leitura de  $100\mu s$  e *baudrate* de 57.600;
- Módulo HX711 para condicionamento dos sinais da célula de carga para o microcontrolador. Trata-se de um conversor analógico digital para balanças.

Vale notar que é necessário ter apenas uma célula de carga ativa para realizar as medições, uma vez que, devido a suas características construtivas, a força aplicada pelos dois dedos da garra é igual. Sendo assim, a segunda célula foi utilizada com o intuito de manter o outro dedo da garra estável e sob o mesmo regime de esforço.

A seguir, será detalhado o procedimento realizado desde a calibração do sensor até a aquisição final dos resultados.

Primeiramente, foram utilizados pesos padronizados para a calibração da célula de carga (Figura 34). A estrutura composta pela célula de carga e suporte de aço foi fixada em uma morsa de bancada. Assim, diferentes pesos padrões foram medidos via célula de carga, para comparação com o valor na balança de precisão e estimativa da função de calibração.

Figura 34 – Calibração da célula de carga com pesos padrões.



Fonte: Autor, 2023.

Como resultado do procedimento de calibração, foram obtidos os dados necessários para gerar uma função de interpolação. Posteriormente, foi utilizado o software software MATLAB, em específico, o seu recurso 'polyfit'. A partir disso, obteve-se o seguinte resultado:

$$f = -0.01x - 0.6823 \quad (3.1)$$

A Equação 3.1 é a função de interpolação que representa o ajuste de calibração dos dados da medição.

A leitura dos dados pelo Arduino foi feita com um *script* baseado na biblioteca do HX711 ADC, que permite a aquisição dos dados da célula de carga. Dessa forma, pode ser medida a força que a garra executa quando da sua abertura e fechamento em uma superfície. Um filtro de média móvel também foi aplicado no *script*, para evitar ruídos de medição e estabilizar os resultados encontrados. Por fim, os dados de aquisição podem ser verificados no monitor serial do Arduino IDE.

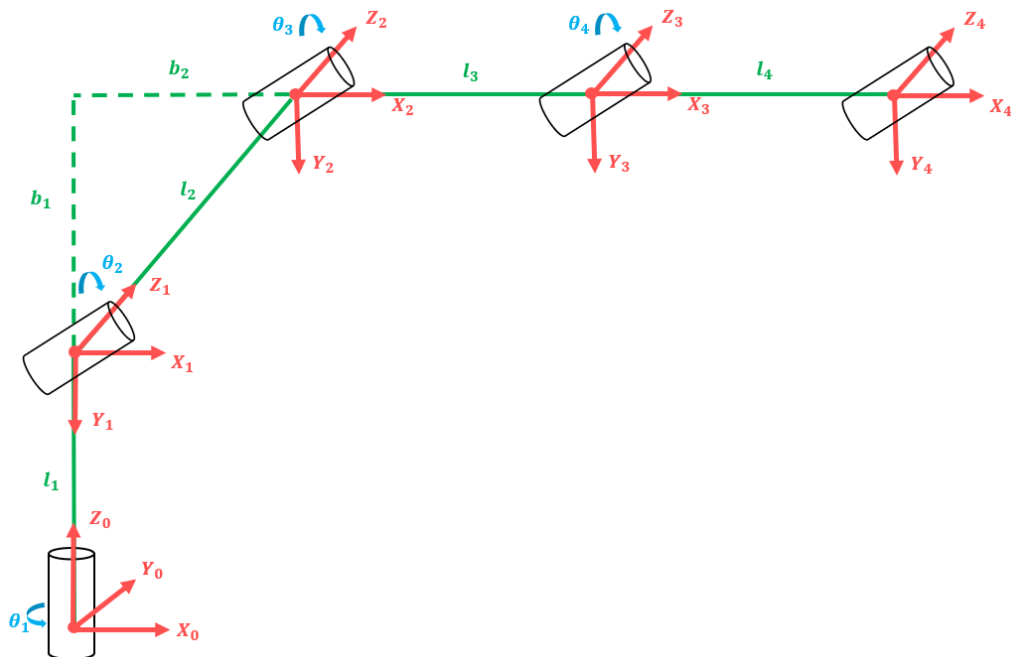
## 4 MECANISMOS DE CONTROLE

O controle é um aspecto essencial de um dispositivo robótico. Nesse capítulo, serão desenvolvidos os procedimentos de cálculo da cinemática direta do manipulador. Após, será apresentada a implementação dos algoritmos que permitem a movimentação conforme a trajetória desejada para o robô.

### 4.1 DIAGRAMA CINEMÁTICO

A cinemática direta pode ser determinada de acordo com a convenção de Denavit-Hartenberg, conforme detalhado na Seção 2.2. O primeiro procedimento é a definição de sistemas de referência em cada um dos elos (Figura 35). Com isso, são representadas as coordenadas espaciais  $Z_i$ ,  $X_i$  e  $Y_i$  a partir das quais serão encontrados os parâmetros dos elos. Sendo que a variável  $l$  se refere ao comprimento do elo. Na Figura 35, é apresentada a disposição cinemática conforme os parâmetros DH. Ela serve tanto para o Viper X 300 quanto o robô desse projeto.

Figura 35 – Configuração do diagrama cinemático.



Fonte: Autor, 2023.

## 4.2 PARÂMETROS DH PARA VIPER X 300

Em (LAURITZEN, 2023) é realizada a obtenção dos parâmetros DH do ReactorX-150, outro robô da fabricante *Interbotix*. Essa tese serviu como material de referência para os procedimentos elaborados a seguir no documento. Assim, convém ressaltar que a formulação DH propõe o estabelecimento dos seguintes parâmetros:

- $\theta_i$ : rotação em torno de  $Z_{n-1}$ ;
- $\alpha_i$ : rotação em torno de  $X_n$  de  $Z_{n-1}$  para  $Z_n$ ;
- $d_i$ : deslocamento ao longo de  $Z_{n-1}$ ;
- $a_i$  deslocamento ao longo de  $X_n$ .

As informações do desenho técnico do Viper X 300 indicam os valores dos comprimentos como:  $l_1 = 126,75$ ,  $l_2 = 305,94$ ,  $l_3 = 300$ ,  $l_4 = 204,21$ ,  $b_1 = 300$  e  $b_2 = 60$  (TROSSEN ROBOTICS, 2023). E esses são os dados necessários para corresponder ao diagrama cinemático.

Na Figura 35, apenas o eixo  $Z_0$  aponta em uma direção diferente, ou seja, ele está girado em  $-90^\circ$ , girando em torno de  $X_1$  para corresponder a  $Z_1$ . Verifica-se que há um deslocamento entre os *frames* um e dois no eixo  $y$ . Esse deslocamento é indicado pela posição em que os motores (e, portanto, as juntas) são alocados no robô. Isso significa que  $z_2$  e  $z_3$  são girados de forma a compensar o deslocamento. Esse aspecto é considerado na tabela DH a partir do ângulo calculado na Equação 4.1:

$$\arcsin\left(\frac{300}{305,94}\right) = 78,69^\circ \quad (4.1)$$

Com base nesses procedimentos, são preenchidos os parâmetros de DH na Tabela 12.

Tabela 12 – Parâmetros de Denavit-Hartenberg para o Viper X 300.

<b>Elo (i)</b>	$a_i$	$d_i$	$\alpha_i$	$\theta_i$
<b>1</b>	0	126,75	$-90^\circ$	$\theta_1$
<b>2</b>	305,94	0	$0^\circ$	$\theta_2 - 78,69^\circ$
<b>3</b>	300	0	$0^\circ$	$\theta_3 + 78,69^\circ$
<b>4</b>	204,21	0	$0^\circ$	$\theta_4$

Fonte: Autor, 2023.

O próximo passo é inserir os parâmetros na matriz de transformação homogênea,

de forma a obter como resultado as quatro matrizes a seguir.

$$T_0^1 = \begin{bmatrix} c(\theta_1) & 0 & -s(\theta_1) & 0 \\ s(\theta_1) & 0 & c(\theta_1) & 0 \\ 0 & -1 & 0 & 126,75 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4.2)$$

$$T_1^2 = \begin{bmatrix} c(\theta_2 - 78,69^\circ) & -s(\theta_2 - 78,69^\circ) & 0 & 305,94 \cdot c(\theta_2 - 78,69^\circ) \\ s(\theta_2 - 78,69^\circ) & c(\theta_2 - 78,69^\circ) & 0 & 305,94 \cdot s(\theta_2 - 78,69^\circ) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4.3)$$

$$T_2^3 = \begin{bmatrix} c(\theta_3 + 78,69^\circ) & -s(\theta_3 + 78,69^\circ) & 0 & 300 \cdot c(\theta_3 + 78,69^\circ) \\ s(\theta_3 + 78,69^\circ) & c(\theta_3 + 78,69^\circ) & 0 & 300 \cdot s(\theta_3 + 78,69^\circ) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4.4)$$

$$T_3^4 = \begin{bmatrix} c(\theta_4) & -s(\theta_4) & 0 & 204,21 \cdot c(\theta_4) \\ s(\theta_4) & c(\theta_4) & 0 & 204,21 \cdot s(\theta_4) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4.5)$$

As matrizes anteriores são, então multiplicadas para chegar na matriz H:

$$H = T_0^1 \cdot T_1^2 \cdot T_2^3 \cdot T_3^4 \quad (4.6)$$

$$H = \begin{bmatrix} H_{11} & H_{12} & -s(\theta_1) & x_e \\ H_{21} & H_{22} & c(\theta_1) & y_e \\ -s(\theta_2 + \theta_3 + \theta_4) & -c(\theta_2 + \theta_3 + \theta_4) & 0 & z_e \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4.7)$$

$$H_{11} = 0.5 \cdot c(\theta_2 - \theta_1 + \theta_3 + \theta_4) + 0.5 \cdot c(\theta_1 + \theta_2 + \theta_3 + \theta_4) \quad (4.8)$$

$$H_{12} = -0.5 \cdot s(\theta_2 - \theta_1 + \theta_3 + \theta_4) - 0.5 \cdot s(\theta_1 + \theta_2 + \theta_3 + \theta_4) \quad (4.9)$$

$$H_{21} = 0.5 \cdot s(\theta_1 + \theta_2 + \theta_3 + \theta_4) - 0.5 \cdot s(\theta_2 - \theta_1 + \theta_3 + \theta_4) \quad (4.10)$$

$$H_{22} = 0.5 \cdot c(\theta_1 + \theta_2 + \theta_3 + \theta_4) - 0.5 \cdot c(\theta_2 - \theta_1 + \theta_3 + \theta_4) \quad (4.11)$$

Os componentes  $x_e$ ,  $y_e$ ,  $z_e$  definem as posições do efetuador do robô manipulador e são dados na Tabela 13.

Tabela 13 – Equações da posição do efetuador final, de acordo com Denavit-Hartenberg para o Viper X 300.

<b>Equações da posição do efetuador final (Viper X 300)</b>	
$x_e$	$150,0 \cdot c(\theta_1 + \theta_2 + \theta_3)$
	$+102,105 \cdot c(\theta_2 - 1,0 \cdot \theta_1 + \theta_3 + \theta_4)$
	$+102,105 \cdot c(\theta_1 + \theta_2 + \theta_3 + \theta_4)$
	$+152,97 \cdot c(\theta_1 - 1,0 \cdot \theta_2 + 1,373)$
	$+150,0 \cdot c(\theta_2 - 1,0 \cdot \theta_1 + \theta_3)$
	$+152,97 \cdot c(\theta_1 + \theta_2 - 1,373)$
$y_e$	$152,97 \cdot s(\theta_1 + \theta_2 - 1,373)$
	$+150,0 \cdot s(\theta_1 + \theta_2 + \theta_3)$
	$-102,105 \cdot s(\theta_2 - 1,0 \cdot \theta_1 + \theta_3 + \theta_4)$
	$+102,105 \cdot s(\theta_1 + \theta_2 + \theta_3 + \theta_4)$
	$+152,97 \cdot s(\theta_1 - 1,0 \cdot \theta_2 + 1,373)$
$z_e$	$-150,0 \cdot s(\theta_2 - 1,0 \cdot \theta_1 + \theta_3)$
	$126,75 - 305,94 \cdot s(\theta_2 - 1,373)$
	$-300,0 \cdot s(\theta_2 + \theta_3)$
	$-204,21 \cdot s(\theta_2 + \theta_3 + \theta_4)$

Fonte: Autor, 2023.

#### 4.3 PARÂMETROS DH PARA O ROBÔ CONSTRUÍDO

O robô construído neste trabalho segue o mesmo diagrama cinemático da Figura 35, portanto os cálculos da cinemática são similares aos do Viper X 300 elaborados na seção anterior.

As informações medidas no robô físico indicam os valores dos comprimentos como:

$$l_1 = 115, l_2 = 395,38, l_3 = 430, l_4 = 320, b_1 = 385,0 \text{ e } b_2 = 90.$$

O ângulo, nesse caso, é calculado na Equação 4.12:

$$\arcsin\left(\frac{385}{395,38}\right) = 76,84^\circ \quad (4.12)$$

São, então, preenchidos os parâmetros de DH na Tabela 14.

O próximo passo é inserir os parâmetros na matriz de transformação homogênea,

Tabela 14 – Parâmetros de Denavit-Hartenberg para o robô construído.

Elo (i)	$a_i$	$d_i$	$\alpha_i$	$\theta_i$
1	0	115,0	$-90^\circ$	$\theta_1$
2	395,38	0	$0^\circ$	$\theta_2 - 76,84^\circ$
3	430,0	0	$0^\circ$	$\theta_3 + 76,84^\circ$
4	320,0	0	$0^\circ$	$\theta_4$

Fonte: Autor, 2023.

de forma a obter como resultado as quatro matrizes a seguir.

$$T_0^1 = \begin{bmatrix} c(\theta_1) & 0 & -s(\theta_1) & 0 \\ s(\theta_1) & 0 & c(\theta_1) & 0 \\ 0 & -1 & 0 & 115,0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4.13)$$

$$T_1^2 = \begin{bmatrix} c(\theta_2 - 76,84^\circ) & -s(\theta_2 - 76,84^\circ) & 0 & 395,38 \cdot c(\theta_2 - 76,84^\circ) \\ s(\theta_2 - 76,84^\circ) & c(\theta_2 - 76,84^\circ) & 0 & 395,38 \cdot s(\theta_2 - 76,84^\circ) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4.14)$$

$$T_2^3 = \begin{bmatrix} c(\theta_3 + 76,84^\circ) & -s(\theta_3 + 76,84^\circ) & 0 & 430 \cdot c(\theta_3 + 76,84^\circ) \\ s(\theta_3 + 76,84^\circ) & c(\theta_3 + 76,84^\circ) & 0 & 430 \cdot s(\theta_3 + 76,84^\circ) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4.15)$$

$$T_3^4 = \begin{bmatrix} c(\theta_4) & -s(\theta_4) & 0 & 320,0 \cdot c(\theta_4) \\ s(\theta_4) & c(\theta_4) & 0 & 320,0 \cdot s(\theta_4) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4.16)$$

As matrizes anteriores são, então multiplicadas para chegar na matriz H:

$$H = T_0^1 \cdot T_1^2 \cdot T_2^3 \cdot T_3^4 \quad (4.17)$$

$$H = \begin{bmatrix} H_{11} & H_{12} & -s(\theta_1) & x_e \\ H_{21} & H_{22} & c(\theta_1) & y_e \\ -s(\theta_2 + \theta_3 + \theta_4) & -c(\theta_2 + \theta_3 + \theta_4) & 0 & z_e \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4.18)$$



$$H_{11} = 0.5 \cdot c(\theta_2 - \theta_1 + \theta_3 + \theta_4) + 0.5 \cdot c(\theta_1 + \theta_2 + \theta_3 + \theta_4) \quad (4.19)$$

$$H_{12} = -0.5 \cdot s(\theta_2 - \theta_1 + \theta_3 + \theta_4) - 0.5 \cdot s(\theta_1 + \theta_2 + \theta_3 + \theta_4) \quad (4.20)$$

$$H_{21} = 0.5 \cdot s(\theta_1 + \theta_2 + \theta_3 + \theta_4) - 0.5 \cdot s(\theta_2 - \theta_1 + \theta_3 + \theta_4) \quad (4.21)$$

$$H_{22} = 0.5 \cdot c(\theta_1 + \theta_2 + \theta_3 + \theta_4) - 0.5 \cdot c(\theta_2 - \theta_1 + \theta_3 + \theta_4) \quad (4.22)$$

Novamente, os componentes  $x_e$ ,  $y_e$ ,  $z_e$  definem as posições do efetuador do robô manipulador e são dados na Tabela 15.

Tabela 15 – Equações da posição do efetuador final, de acordo com Denavit-Hartenberg para o robô construído.

<b>Equações da posição do efetuador final (Robô construído)</b>	
$x_e$	$215,0 \cdot c(\theta_1 + \theta_2 + \theta_3)$
	$+160,0 \cdot c(\theta_2 - 1,0 \cdot \theta_1 + \theta_3 + \theta_4)$
	$+160,0 \cdot c(\theta_1 + \theta_2 + \theta_3 + \theta_4)$
	$+197,69 \cdot c(\theta_1 - 1,0 \cdot \theta_2 + 1,341)$
	$+215,0 \cdot c(\theta_2 - 1,0 \cdot \theta_1 + \theta_3)$
	$+197,69 \cdot c(\theta_1 + \theta_2 - 1,341)$
$y_e$	$197,69 \cdot s(\theta_1 + \theta_2 - 1,341)$
	$+215,0 \cdot s(\theta_1 + \theta_2 + \theta_3)$
	$-160,0 \cdot s(\theta_2 - 1,0 \cdot \theta_1 + \theta_3 + \theta_4)$
	$+160,0 \cdot s(\theta_1 + \theta_2 + \theta_3 + \theta_4)$
	$+197,69 \cdot s(\theta_1 - 1,0 \cdot \theta_2 + 1,341)$
	$-215,0 \cdot s(\theta_2 - 1,0 \cdot \theta_1 + \theta_3)$
$z_e$	$115,0 - 395,38 \cdot s(\theta_2 - 1,341)$
	$-430,0 \cdot s(\theta_2 + \theta_3)$
	$-320,0 \cdot s(\theta_2 + \theta_3 + \theta_4)$

Fonte: Autor, 2023.

#### 4.4 CINEMÁTICA VIA PRODUTO DE EXPONENCIAIS

A cinemática direta também pode ser obtida utilizando-se produto de exponenciais para fazer o mapeamento dos elos da cadeia cinemática. Trata-se de um método alternativo à parametrização de Denavit-Hartenberg. Essa é a abordagem que a *Interbotix*

aplica no *software* dos seus robôs e fornece na sua documentação técnica. Uma das suas vantagens é a facilidade de ser processada computacionalmente.

Esse método resulta na obtenção de duas matrizes:

- A matriz **M** é a configuração do efetuador quando o robô está na sua posição *home* (zero). Essa posição significa que todas as juntas estão na posição 0;
- A matriz **Slist** representa os eixos de rotação das juntas no espaço quando o robô está em sua posição *home*. É formatada para ter cada eixo como uma coluna.

Para esse método, é necessário definir apenas dois referenciais (*frame*), o da base e o do efetuador. *Space frame* é o nome dado para o elo que servirá como a localização da base do robô ou elo 0 e *body frame* é o nome dado para o elo que servirá como a localização do efetuador.

Novamente, será tomado como base o diagrama cinemático apresentado na Figura 35. A cinemática por produto de exponenciais pode ser, então, obtida através da sequência de passos elaborada a seguir, de acordo com os procedimentos indicados no livro *Modern Robotics* (LYNCH; PARK, 2017).

Como os eixos tanto da base quanto do efetuador estão alinhados, há a correspondência entre eles na parte dedicada à rotação da matriz M:

- (1, 0, 0) indica que o eixo *x* do frame do efetuador está alinhado com o eixo *x* da base;
- (0, 1, 0) indica que o eixo *y* do frame do efetuador está alinhado com o eixo *y* da base;
- (0, 0, 1) indica que o eixo *z* do frame do efetuador está alinhado com o eixo *z* da base.

O *frame* do efetuador possui um deslocamento de  $b_2 + l_3 + l_4$  na coordenada *x*, 0 na coordenada *y* e  $l_1 + b_1$  na coordenada *z*. Por fim, devem ser adicionados zeros e 1 para deixar a matriz quadrada.

O preenchimento da matriz M completa é dado na Equação 4.23.

$$M = \begin{bmatrix} 1 & 0 & 0 & b_2 + l_3 + l_4 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & l_1 + b_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4.23)$$

Considerando  $\vec{w}$  como o vetor unitário na direção positiva do eixo de junta (velocidade angular),  $\vec{q}$  qualquer ponto arbitrário no eixo de junta escrito em termos das coorde-

nadas no *space frame* e  $\vec{v}$  o vetor unitário na direção de translação positiva, é estabelecida a relação de produto vetorial:

$$\vec{v} = -\vec{w} \times \vec{q} \quad (4.24)$$

Assim, a matriz completa para a matriz *Slist* consiste na Equação 4.25.

$$Slist = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & -l_1 & 0 & 0 \\ 0 & 1 & 0 & -l_1 - b_1 & 0 & b_2 \\ 0 & 1 & 0 & -l_1 - b_1 & 0 & b_2 + l_3 \\ 1 & 0 & 0 & 0 & l_1 + b_1 & 0 \end{bmatrix}^T \quad (4.25)$$

Dessa forma, foram obtidas as duas matrizes, resolvidas para variáveis simbólicas. Aplicando os dados referentes ao Viper X 300, conforme apresentados na seção anterior e agora convertidos de *mm* para *m*, verifica-se que:

$$M = \begin{bmatrix} 1 & 0 & 0 & 0,536494 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0,42705 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4.26)$$

$$Slist = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & -0,12705 & 0 & 0 \\ 0 & 1 & 0 & -0,42705 & 0 & 0,05955 \\ 0 & 1 & 0 & -0,42705 & 0 & 0,35955 \\ 1 & 0 & 0 & 0 & 0,42705 & 0 \end{bmatrix}^T \quad (4.27)$$

Essas matrizes correspondem às informadas na documentação técnica do Viper X 300, disponíveis em (TROSSEN ROBOTICS, 2023).

Já aplicando as dimensões do protótipo do robô construído, obtêm-se:

$$M = \begin{bmatrix} 1 & 0 & 0 & 0,605 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0,5 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4.28)$$

$$Slist = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & -0,115 & 0 & 0 \\ 0 & 1 & 0 & -0,5 & 0 & 0,09 \\ 0 & 1 & 0 & -0,5 & 0 & 0,52 \\ 1 & 0 & 0 & 0 & 0,5 & 0 \end{bmatrix}^T \quad (4.29)$$

Na Seção 4.6.5, será visto como ambas as matrizes são importantes para a definição da estrutura do robô no *software* e compõem os aspectos necessários para a correta simulação e utilização das bibliotecas da *Interbotix*.

#### 4.5 SISTEMA DE CONTROLE E OPERAÇÃO DOS MOTORES

Os motores DYNAMIXEL utilizados no projeto estavam originalmente com o Protocolo 1.0 instalado. Esse é um protocolo mais antigo da linha MX. Dessa forma, o *firmware* dos motores foi atualizado para o Protocolo 2.0, através da interface do DYNAMIXEL Wizard. Esse protocolo é mais estável e seguro que o Protocolo 1.0, além de permitir recursos adicionais de configuração.

Cada motor DYNAMIXEL pode operar segundo diferentes modos de operação. Esses modos possuem características e possibilidades de aplicação em cenários específicos (ROBOTIS, 2023). O primeiro a ser explorado, que é considerado padrão, é o modo de posição. Ele apresenta características particulares que o diferenciam e também o assemelham aos servos usuais. Permite que o motor seja direcionado para um ângulo específico dentro do seu intervalo de movimento, ou seja, possibilita a definição da posição das juntas. Nesse modo, a capacidade de giro é de até  $360^\circ$ , sendo estabelecidas uma posição limite máxima e uma posição limite mínima, que correspondem ao seu intervalo de operação. Isso pode ser especialmente útil em aplicações onde o movimento do motor precisa ser restrito a uma certa faixa de ângulos, seja por questões de segurança, projeto ou funcionalidade. É adequado para a maioria das aplicações, em específico para robôs articulados, em que cada junta rotaciona menos de  $360^\circ$ .

O modo de posição estendida é uma variação do modo tradicional de posição oferecido pelos motores DYNAMIXEL. Enquanto mantém semelhanças, ele incorpora características adicionais que o tornam útil para aplicações que exigem maior flexibilidade e amplitude de movimento. Permite que o motor seja direcionado para uma posição específica. Porém, uma das características distintivas é a capacidade de executar múltiplas rotações, ao contrário de um movimento único de  $360^\circ$ , neste modo, o motor pode girar até 512 voltas em torno do seu eixo, gerando movimentos rotativos contínuos. Sendo assim, as restrições angulares não são utilizadas no modo de posição estendida.

O modo de velocidade dos motores DYNAMIXEL oferece uma abordagem diferente para o controle do movimento, focando na regulação da velocidade do motor em vez de sua posição. Apresenta características que o tornam especialmente útil para aplicações como robôs móveis com rodas. No modo de velocidade, o motor é capaz de executar uma rotação contínua e sem limitações. O usuário pode definir uma velocidade específica que deseja que o motor alcance e o DYNAMIXEL ajustará sua operação para atender a esse comando. Esse tipo de controle é essencial em aplicações em que a taxa de movimento é

mais crítica do que a posição final.

O modo de controle por PWM é uma abordagem especializada de controle para motores DYNAMIXEL que se baseia na modulação direta da largura de pulso, operando de maneira similar a um motor de corrente contínua. Nesse caso, o motor é capaz de realizar uma rotação contínua sem limitações de ângulo ou posição. Possui a vantagem de controlar a força aplicada pelo motor, de forma que ela possa ser ajustada de acordo com as necessidades da aplicação.

Também existe o modo de controle baseado em corrente, que controla o torque em vez da velocidade ou posição e permite rotações múltiplas. Adicionalmente, o modo de controle de posição baseado em corrente é uma alternativa que suporta até 512 voltas e controla a posição e a corrente (torque) simultaneamente.

No decorrer do projeto, foram testados os diferentes modos de controle para a operação dos motores individualmente. Já para a estrutura do robô manipulador em si, foi aplicado o modo de controle de posição para os motores das juntas (IDs 1 a 7), uma vez que dessa forma é possível direcionar o motor para a posição desejada e limitar a rotação, para evitar que o robô seja levado para posições que comprometeriam a sua estrutura, podendo, por exemplo, romper o cabeamento interno. Na garra, por outro lado, foi utilizado o controle por PWM, que permite um ajuste fino da força do motor. Essas escolhas também são embasadas nas definições padrões estabelecidas nos pacotes de *software* da *Interbotix*, que garantem o funcionamento adequado do sistema.

Com o objetivo de aumentar a performance de uma junta, mais de um motor pode ser combinado para trabalhar em conjunto na mesma junta. Esse é o caso do ombro e do cotovelo do manipulador, que são compostos por dois motores. Assim, torna-se necessário sincronizar de maneira adequada os dois atuadores. Deve-se assegurar que os motores operem de forma a não comprometer ou forçar a estrutura em que estão inseridos, respeitando a dinâmica e os ângulos de instalação.

O envio de comandos separadamente para os motores pode fazer com que eles fiquem ligeiramente deslocados, e uma descoordenação, mesmo que mínima, pode resultar em pressões indevidas sobre a estrutura. Por isso, é utilizado o método de definir um ID Secundário (*Shadow ID*) no endereço 12 de um dos motores da junta. Assim, o motor secundário irá receber comandos do motor principal e atuar em sincronia com ele. Essa solução não requer configurações adicionais de *hardware* e *software*, apenas a configuração interna do próprio motor.

Complementarmente, a configuração de *Drive mode* no endereço 10 permite escolher que os motores operem tanto no modo normal (ambos girando no mesmo sentido) quanto no modo reverso (girando em sentidos opostos). Essa última opção foi a utilizada, já que é particularmente útil nas juntas do ombro e cotovelo, na qual, pela forma em que foram dispostos, os motores precisam operar em direções contrárias.

## 4.6 PROGRAMAÇÃO

Para que o robô desenvolvido pudesse funcionar de maneira similar ao robô original da *Interbotix*, ele deve ter a mesma convenção de nomenclatura e documentação. Além disso, deve conter os arquivos adaptados com base nas suas especificações. O robô foi denominado de *lucasrobot* em todos os arquivos. A listagem contendo a sequência que deve ser realizada é encontrada em (Wiznitzer et al., 2023) e sintetizada abaixo:

1. Criar um arquivo de configuração dos motores no formato YAML;
2. Criar um arquivo URDF descrevendo a estrutura do robô e adicionar o projeto mecânico das peças;
3. Fornecer os controladores de posição e trajetória para o ROS/Gazebo;
4. Criar um arquivo SRDF para o MoveIt;
5. Adicionar as matrizes cinemáticas.

Os procedimentos relacionados a cada um desses passos serão abordados nas subseções a seguir.

### 4.6.1 Configuração dos motores

A configuração dos motores é dada por um arquivo em formato YAML contido no diretório *interbotix\_xsarm\_control/config*. Esse arquivo começa estabelecendo a porta USB em que o robô está conectado. Também indica a ordem das juntas do robô e requer dados sobre o *gripper*. O Código 4.1 demonstra um exemplo da parte mais relevante desse arquivo, em que são definidos os parâmetros para os motores. Nesse caso, é mostrado para a junta da base (*waist*), mas o processo é similar para as demais juntas.

Código 4.1: Reprodução parcial do arquivo de configuração dos motores.

```

1 motors:
2   waist:
3     ID: 1
4     Baud_Rate: 3
5     Return_Delay_Time: 0
6     Drive_Mode: 0
7     Velocity_Limit: 131
8     Min_Position_Limit: 0
9     Max_Position_Limit: 4095
10    Secondary_ID: 255

```

Os parâmetros definidos são referentes às características do motor dentro do sistema. O *ID* é a identificação do servo, que vai de 1 a 251. O *Baud\_Rate* indica a velocidade em que a comunicação serial ocorre. O *Return\_Delay\_Time* refere-se ao tempo que o servo demora para enviar uma resposta a um pacote recebido. O *Drive\_Mode* indica o sentido de rotação do motor. O parâmetro *Velocity\_Limit* atribui o limite de velocidade, sendo 131 o valor máximo. Já *Min\_Position\_Limit* e *Max\_Position\_Limit*, definem o alcance da junta, importantes de serem limitados caso a junta não possa rotacionar 360°. Finalmente, *Secondary\_ID* é usado quando uma junta é formada por dois motores, em que um segue os comandos do outro. Para essa junta, o valor de 255 desabilita a funcionalidade. Porém, o *Secondary\_ID* foi aplicado no ombro e no cotovelo.

#### 4.6.2 URDF

Para que seja possível simular um sistema robótico o mais próximo possível do seu aspecto real, é necessário agregar todas as características físicas do robô em um mesmo modelo digital. O URDF (*Unified Robot Description Format*) é o formato mais utilizado para isso. Ele permite descrever a estrutura, cinemática, dinâmica e atributos visuais de robôs. Foi introduzido como um formato padrão no ROS e posteriormente ganhou suporte em diversas outras ferramentas de simulação (ROS, 2023). O arquivo *.urdf* descrevendo o modelo também pode contar com outro arquivo, que representa o conjunto de malhas, que são arquivos da aparência física dos elos do robô, ou seja, arquivos geralmente em formato *.STL* correspondendo às peças do sistema. O arquivo URDF se refere a esse arquivo com as malhas (*meshes*) geométricas dos diferentes elos usando caminhos relativos para a pasta.

Entre os aspectos que podem ser definidos com o URDF estão:

- **Estrutura:** permite a descrição das conexões entre as diferentes partes, como juntas e elos. As juntas podem ser descritas como fixas, contínuas, giratórias, deslizantes, entre outras.
- **Geometria:** inclui informações sobre a geometria dos componentes do robô, como sua forma, tamanho e posição. Também é possível especificar informações visuais, como cores e texturas, facilitando a simulação e visualização do robô em ambientes virtuais.
- **Propriedades cinemáticas e dinâmicas:** contém informações sobre as cadeias cinemáticas do robô, incluindo modelos de colisão e limites das juntas. Além disso, agrega as propriedades dinâmicas, como centro de massa e inércia.

A *Interbotix* disponibiliza os arquivos URDF dos seus robôs em formato Xacro (XML

macro). Esse formato é uma extensão do URDF que permite a parametrização e reutilização de componentes, tornando a descrição do robô mais modular e fácil de manter. Verifica-se que, como o URDF é baseado na linguagem de marcação XML, a representação consiste em uma série de *tags*, as quais podem ser agrupadas.

O arquivo URDF referente ao robô Viper X 300 foi adaptado de forma a considerar as propriedades específicas do robô desenvolvido. O Código 4.2 é apresentada a descrição para a junta da base. Os outros componentes foram expressos com estrutura similar.

Código 4.2: Reprodução parcial do arquivo URDF

```

1  <link name="$(arg robot_name)/$(arg base_link_frame)">
2    <visual>
3      <origin rpy="0 0 ${pi/2}" xyz="-0.045 0 0"/>
4      <geometry>
5        <mesh filename="package://interbotix_xsarm_descriptions/meshes/
6          lucasrobot_meshes/lucasrobot_1_base.stl" scale="0.001 0.001 0.001"/>
7      </geometry>
8      <material name="interbotix_black"/>
9    </visual>
10   <collision>
11     <origin rpy="0 0 ${pi/2}" xyz="-0.045 0 0"/>
12     <geometry>
13       <mesh filename="package://interbotix_xsarm_descriptions/meshes/
14         lucasrobot_meshes/lucasrobot_1_base.stl" scale="0.001 0.001 0.001"/>
15     </geometry>
16   </collision>
17   <inertial>
18     <origin rpy="0 0 ${pi/2}" xyz="-0.0534774000 -0.0005625750
19     0.0205961000"/>
20     <mass value="0.969034" />
21     <inertia ixx="0.0060240000" iyy="0.0017000000" izz="0.0071620000"
22     ixy="0.0000471300" ixz="0.0000038510" iyz="-0.0000841500" />
23   </inertial>
24 </link>
25
26 <joint name="waist" type="revolute">
27   <axis xyz="0 0 1"/>
28   <limit effort="10" lower="${-pi + pi_offset}" upper="${pi -
29     pi_offset}" velocity="${pi}"/>
30   <origin rpy="0 0 0" xyz="0 0 0.05"/>
31   <parent link="$(arg robot_name)/$(arg base_link_frame)"/>
32   <child link="$(arg robot_name)/shoulder_link"/>
33   <dynamics friction="0.1"/>
34 </joint>

```

As peças mecânicas do robô desenvolvido foram salvas em formato *.STL* e inseridas no diretório correspondente. Elas são denominadas de *meshes* na documentação.



Dessa forma, a simulação do robô irá contar com as mesmas peças que foram impressas para compor o robô físico.

### 4.6.3 Controladores de posição e trajetória

É necessário expressar as configurações de um controlador para a posição e para a trajetória no Gazebo/ROS, que também corresponda ao valor dos controladores dos próprios motores das juntas. Tanto para o caso da posição quanto da trajetória, foram mantidos os mesmos parâmetros que o do Viper X 300.

O controlador de posição para a junta da base é mostrado no Código 4.3. É possível verificar aspectos como o tópico que publica o estado das juntas do robô, assim como o valor dos ganhos proporcional, integral e derivativo do controlador.

Código 4.3: Reprodução parcial do arquivo do controlador de posição

```

1 waist_controller:
2   type: effort_controllers/JointPositionController
3   joint: waist
4   pid: {p: 100, i: 0.0, d: 0.0}

```

A configuração apresentada é similar às definições para as demais juntas e para o controlador de trajetória.

### 4.6.4 SRDF

O arquivo SRDF (*Semantic Robot Description Format*) é utilizado no contexto do MoveIt. Serve para complementar o URDF com informações adicionais como grupos de movimento das juntas e a matriz de colisões permitidas. Enquanto o URDF define a estrutura física do robô (como elos e juntas), o SRDF adiciona informações semânticas sobre a estrutura do robô, que são essenciais para o planejamento de movimento.

No Código 4.4 é apresentada uma parte do SRDF, referente ao estado do grupo das juntas para a posição *Home* do robô. Essa é a posição em que todos os ângulos de junta estão definidos como 0.

Código 4.4: Reprodução parcial do arquivo SRDF

```

1 <group_state name="Home" group="arm">
2   <joint name="elbow" value="0" />
3   <joint name="shoulder" value="0" />
4   <joint name="waist" value="0" />
5   <joint name="wrist_angle" value="0" />
6   <joint name="wrist_rotate" value="0" />

```

```
7 </group_state>
```

Além disso, no SRDF estão definidas outras posições, como a de *Sleep*, os parâmetros de abertura e fechamento da garra e as condições para desabilitar a checagem de colisões.

#### 4.6.5 Matrizes Cinemáticas

As matrizes cinemáticas que foram obtidas na Seção 4.4, devem ser inseridas no arquivo *mr\_descriptions.py*, já que, para os *scripts* de programação, será utilizada a linguagem Python. O Código 4.5 expressa o formato em que essas matrizes foram definidas.

Código 4.5: Reprodução parcial do arquivo *mr\_descriptions.py*

```
1 class lucasrobot:
2     Slist = np.array([[0.0, 0.0, 1.0, 0.0, 0.0, 0.0],
3                     [0.0, 1.0, 0.0, -0.12705, 0.0, 0.0],
4                     [0.0, 1.0, 0.0, -0.42705, 0.0, 0.05955],
5                     [0.0, 1.0, 0.0, -0.42705, 0.0, 0.35955],
6                     [1.0, 0.0, 0.0, 0.0, 0.42705, 0.0]]).T
7
8     M = np.array([[1.0, 0.0, 0.0, 0.536494],
9                 [0.0, 1.0, 0.0, 0.0],
10                [0.0, 0.0, 1.0, 0.42705],
11                [0.0, 0.0, 0.0, 1.0]])
```

Realizadas todas essas alterações no repositório, seguindo as convenções de nomenclatura e estrutura da *Interbotix*, o robô pode ser carregado nas simulações e também controlado como dispositivo físico.

## 4.7 GERAÇÃO DE TRAJETÓRIAS

Até o momento, observou-se uma ampla variedade de linguagens de programação e ferramentas disponíveis para o controle do robô. No contexto deste projeto, a API Python (biblioteca *interbotix\_xs\_modules*) fornecida pela *Interbotix* foi a mais empregada na geração de trajetórias. Essa API facilita a movimentação do robô de maneira prática e eficiente, demonstrando ser uma escolha conveniente para as necessidades do trabalho.

O primeiro passo para usar a API é inicializar uma instância do manipulador, conforme o Código 4.6. Assim, é especificado um manipulador de nome "*lucasrobot*", que tem um braço "*arm*" e uma garra "*gripper*".

Código 4.6: Instanciar manipulador.

```
1 bot = InterbotixManipulatorXS("lucasrobot", "arm", "gripper")
```

Os movimentos podem ser definidos de diferentes formas. A primeira é movendo o efetuador para uma posição específica (Código 4.7). Esse comando define a posição do efetuador final (*end-effector*) do braço robótico nas coordenadas "x" e "z".

Código 4.7: Movimentação para posição específica.

```
1 bot.arm.set_ee_pose_components(x=0.3, z=0.2)
```

Outra forma, é estabelecendo uma movimentação diretamente para a junta (Código 4.8). Nesse comando, a junta da base gira (*waist*) gira em 90 graus ( $\pi/2$  radianos).

Código 4.8: Movimentação da junta.

```
1 bot.arm.set_single_joint_position("waist", np.pi/2.0)
```

Também é possível executar um trajeto cartesiano (Código 4.9). Nesse caso, o efetuador final se move em um trajeto cartesiano, deslocando-se 0.1 metros para frente (direção "x") e -0.16 metros para baixo (direção de "z").

Código 4.9: Trajeto cartesiano.

```
1 bot.arm.set_ee_cartesian_trajectory(x=0.1, z=-0.16)
```

Além de inserir parâmetros numéricos, é possível utilizar configurações que foram previamente alocadas no arquivo SRDF, como a abertura da garra (Código 4.10). Nesse caso, o método "open()" abre a garra e o método "close()" fecha.

Código 4.10: Abrir a garra.

```
1 bot.gripper.open()
```

Da mesma forma, é possível levar para posições como a *Home*, que foi definida no Código 4.4 e agora pode ser chamada como no Código 4.11.

Código 4.11: Ir para posição *Home*.

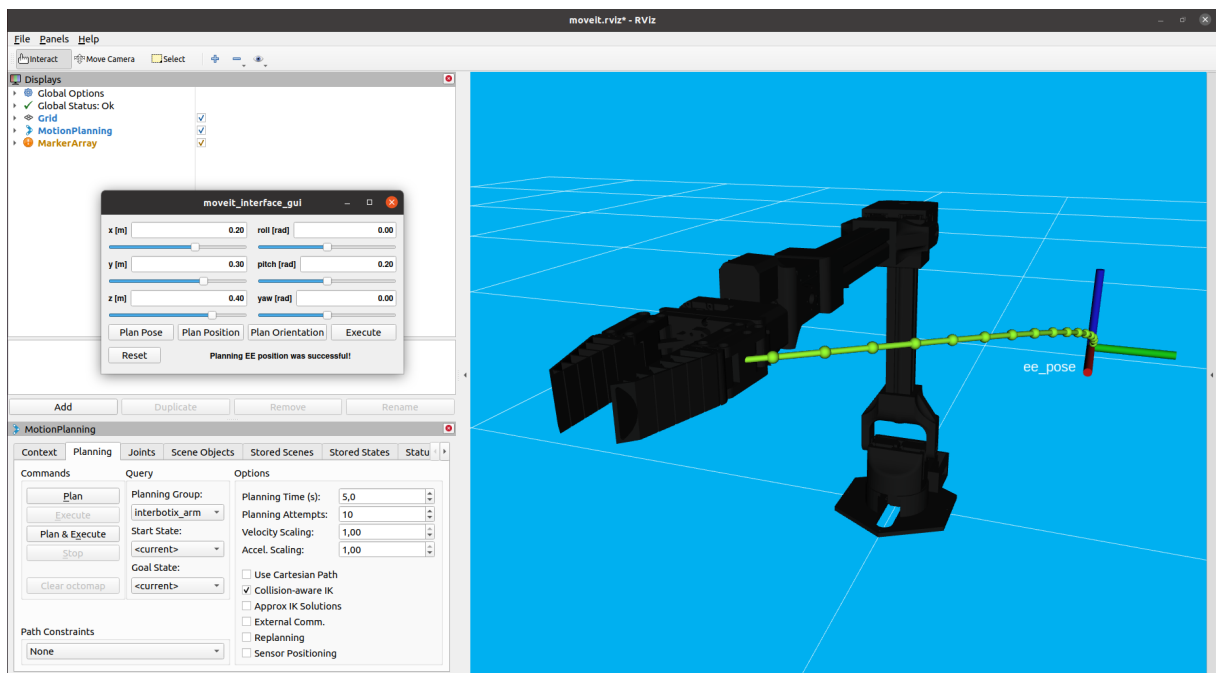
```
1 bot.arm.go_to_home_pose()
```

Todas essas funções apresentadas permitem a criação de uma variedade de trajetórias para o robô, por meio de comandos simples na linguagem Python. Uma simulação utilizando essas funcionalidades será demonstrada na seção a seguir.

Alternativamente, o planejamento de trajetórias pode ser executado diretamente no MoveIt, a partir da inserção de uma coordenada desejada para o efetuador do robô. Observa-se na Figura 36 a trajetória gerada pelo MoveIt para que o robô se desloque da posição *Home* para uma posição dada por  $x = 0.20$  m,  $y = 0.30$  m,  $z = 0.40$  m,  $\text{roll} = 0.00$  rad,  $\text{pitch} = 0.20$  rad,  $\text{yaw} = 0.00$  rad.

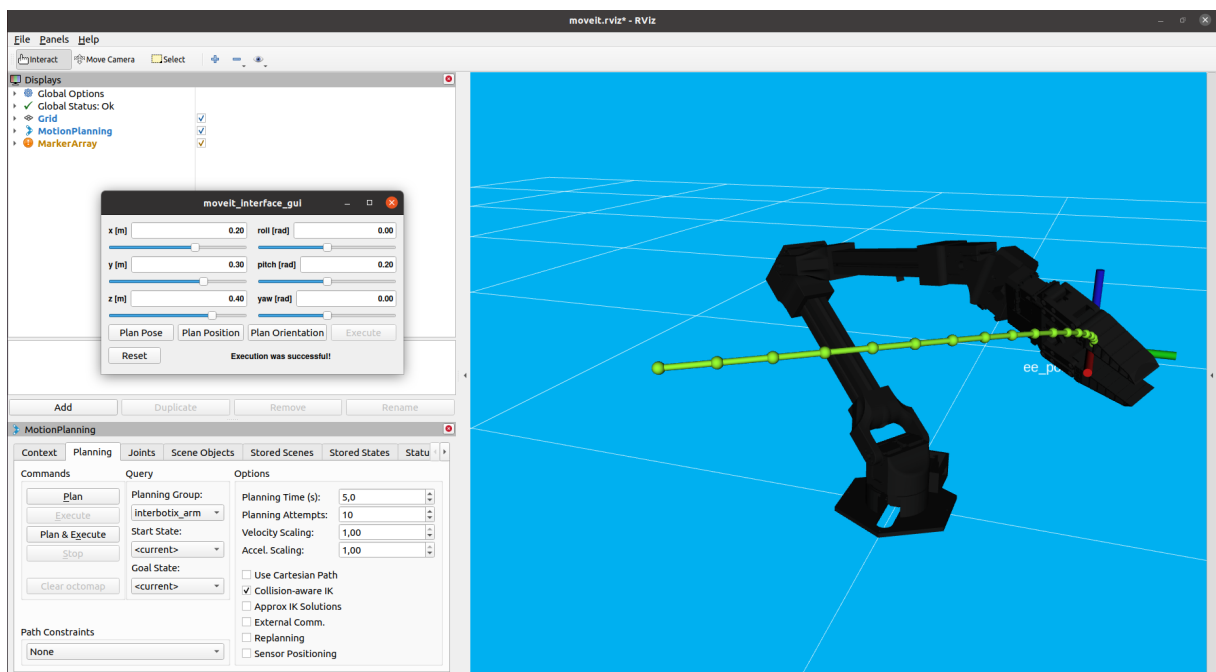
Como o planejamento para essa trajetória foi realizado com sucesso, é possível proceder com a sua execução. O resultado da posição a ser alcançada consta na Figura 37.

Figura 36 – Planejamento de trajetória utilizando MoveIt.



Fonte: Autor, 2023.

Figura 37 – Execução de trajetória utilizando MoveIt.



Fonte: Autor, 2023.

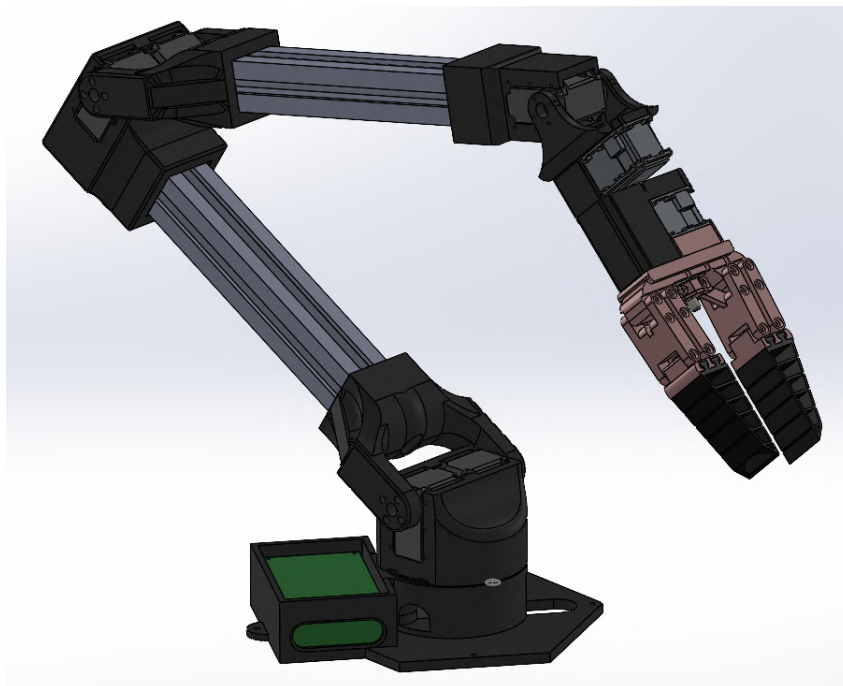
## 5 RESULTADOS E DISCUSSÃO

Neste capítulo são apresentados os resultados da implementação do robô físico, demonstrando a execução de movimentos e a validação experimental da garra proposta. Também será avaliado o sistema construído em comparação com o Viper X 300 e discutidas as oportunidades de melhoria.

### 5.1 PROJETO CAD COMPLETO

A conexão de todos os componentes projetados e dispostos gera a estrutura completa em CAD do manipulador, apresentada na Figura 38.

Figura 38 – Modelo CAD da estrutura completa do robô.



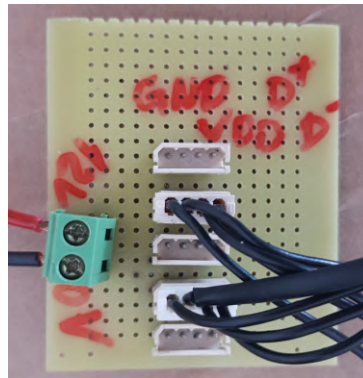
Fonte: Autor, 2023.

### 5.2 MONTAGEM DA PARTE ELÉTRICA

A partir da especificação técnica de uma fonte de 12V, que é a tensão padrão para os motores DYNAMIXEL, definiu-se um modelo adequado de fonte. O equipamento adquirido foi uma fonte chaveada de 12V e 30A.

Também foi desenvolvida uma placa de alimentação montada em fenolite perfurada (Figura 39). Essa placa recebe a tensão da fonte chaveada e estabelece um barramento para alimentar um conjunto de conectores RS-485. Isso permite que os cabos dos DYNAMIXEL sejam plugados diretamente a essa placa. É importante ressaltar que apenas os pinos de VDD e GND do protocolo RS-485 estão sendo utilizados nessa placa.

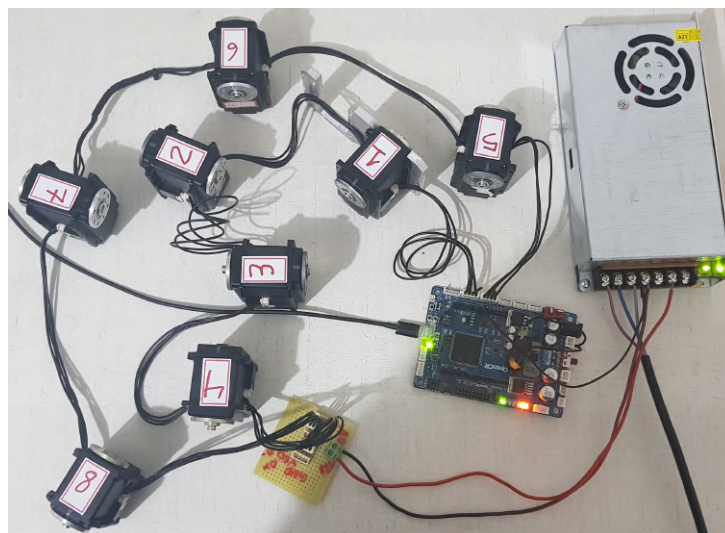
Figura 39 – Placa de alimentação.



Fonte: Autor, 2023.

Com o objetivo de testar todos os componentes elétricos e definir as configurações adequadas para os motores, realizou-se um teste do conjunto formado pelos motores, placa de alimentação, OpenCR e fonte chaveada (Figura 40). Com isso, ficou validado o funcionamento da ligação em cadeia e a conexão dos componentes antes da implementação na estrutura mecânica.

Figura 40 – Montagem dos componentes elétricos do manipulador.



Fonte: Autor, 2023.

O cabeamento entre as conexões dos motores foi efetuado utilizando cabos próprios da linha DYNAMIXEL. No entanto, devido à curta extensão da maioria deles, foi necessário alongar os cabos para que se adequassem às dimensões do robô.



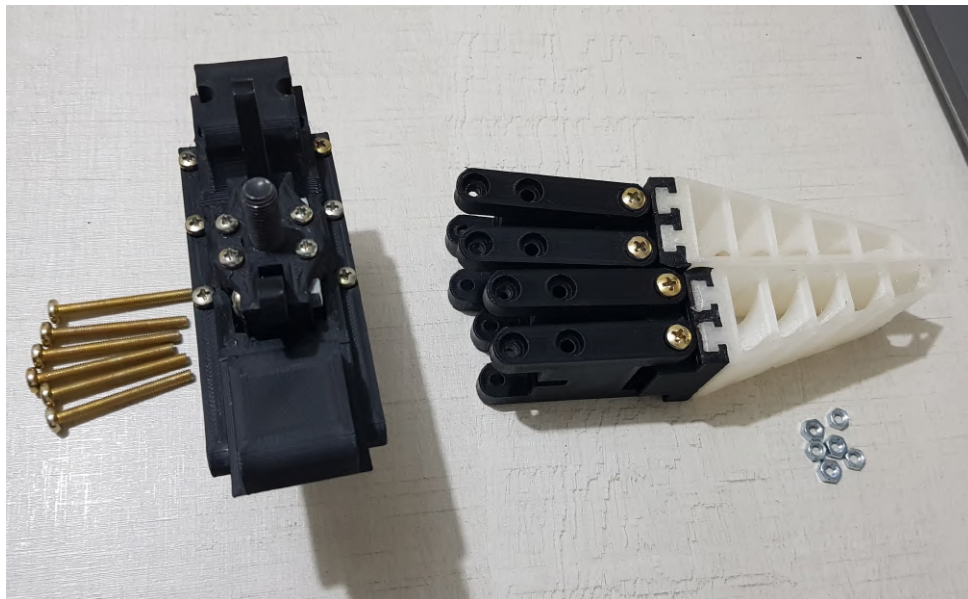
### 5.3 MONTAGEM MECÂNICA

Depois de finalizada a impressão das peças e o desenvolvimento das conexões elétricas, foi montada a estrutura mecânica do manipulador. O processo começou com a fixação da base em uma superfície rígida e nivelada. Em seguida, foram acoplados os atuadores nas posições pré-determinadas.

Dessa forma, cada segmento do manipulador foi montado, conectando as peças impressas em 3D com os elementos fixadores. Os parafusos, porcas e arruelas adequados foram selecionados e apertados de acordo com as especificações, assegurando que todas as conexões fossem estáveis. Durante todo o processo de montagem, foram realizados testes para verificar se os movimentos das articulações ocorriam corretamente.

A garra, cuja montagem é registrada na Figura 41, é uma das etapas mais importantes do sistema, porque requer o encaixe de peças com grande detalhamento em uma sequência específica.

Figura 41 – Registro do processo de montagem da garra.



Fonte: Autor, 2023.

A estrutura mecânica do robô foi concluída de acordo com o modelo (Figura 42). Assim, todos componentes foram construídos, e o robô está com sua estrutura física completa.

### 5.4 SIMULAÇÕES

É possível realizar a simulação do robô construído junto com o Viper X 300 no ROS, utilizando as aplicações da *Interbotix*. A Figura 43 demonstra a vista superior dos

Figura 42 – Estrutura do manipulador totalmente montada.



Fonte: Autor, 2023.

dois robôs na posição *Sleep*. A visualização é dada no *software* Rviz.

Foi realizada uma simulação executando os mesmos movimentos tanto no Viper X 300 quanto no robô construído. O algoritmo escrito em Python que gera a trajetória para ambos os robôs se encontra no APÊNDICE A - COMANDO DE DOIS ROBÔS SIMULTANEAMENTE. A Figura 44 apresenta os registros dessa simulação. As funções utilizadas para a programação são similares às demonstradas na Seção 4.6, que trata como gerar trajetórias com a linguagem Python.

O código do APÊNDICE B - AGARRAR E MOVIMENTAR OBJETO (*PICK AND PLACE*) apresenta uma trajetória completa gerada para o robô construído nesse projeto. Esse algoritmo foi baseado no exemplo *bartender.py* da *Interbotix*, que emula o agarre de um recipiente e execução de um movimento para servir. O registro do cenário resultante é apresentado na Figura 45.

## 5.5 EXECUÇÃO DOS MOVIMENTOS

O robô físico desenvolvido e corretamente configurado permite a programação de movimentos de maneira similar ao Viper X 300, com as funcionalidades da *Interbotix*. É estabelecido, dessa forma, um paralelo entre a interface RViz que executa os algoritmos do MoveIt e o robô físico (Figura 46).

Através da configuração realizada para o ROS/MoveIt, o sistema consegue ler as posições dos motores em tempo real e atualizar na interface. Além disso, os comandos



Figura 43 – Comparativo em simulação do robô construído e do Viper X 300.



Fonte: Autor, 2023.

propostos no Movelt são enviados para os motores e executados no robô físico. Na Figura 46 é demonstrado esse cenário para o momento em que o manipulador está na sua posição *home*.

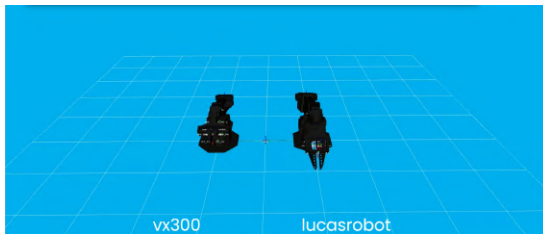
O robô físico conta com flexibilidade para se movimentar conforme desejado pelo usuário. Na Figura 47, é executada uma sequência de movimentos para demonstrar a rotação das juntas em diferentes graus de liberdade.

A possibilidade de agarrar objetos dentro do espaço de trabalho do robô é destacada na Figura 48, em que ocorre a pega de um objeto de uso cotidiano. Para executar essa trajetória, é necessária a atuação coordenada de todas as juntas do robô.

## 5.6 VALIDAÇÃO EXPERIMENTAL

Nesta seção, será apresentado o procedimento de validação experimental com duas abordagens. A primeira é por meio da demonstração de funcionamento da garra robótica ao ser submetida à pega de objetos de uso cotidiano. A segunda é um estudo de

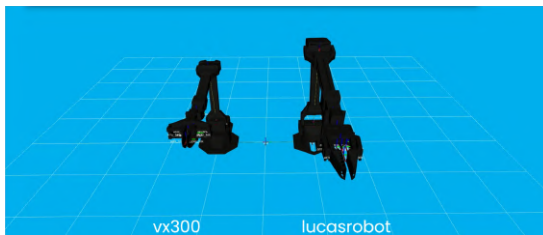
Figura 44 – Simulação de movimentos com ambos os robôs.



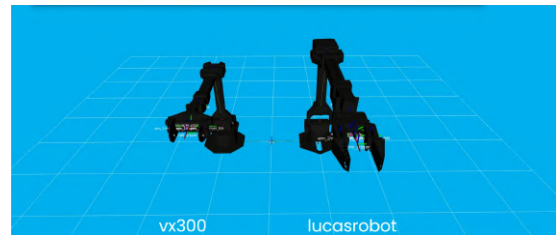
(a) Início na posição *Sleep*.



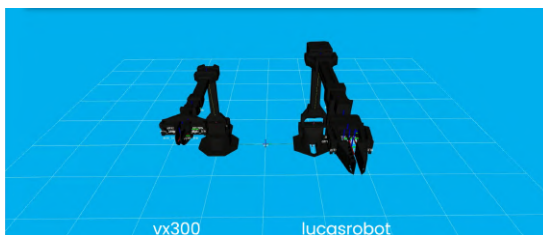
(b) Movimentação para coordenadas  $x = 0,3$  e  $z = 0,2$ .



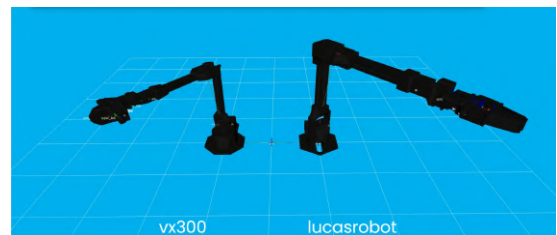
(c) Posição *Home*.



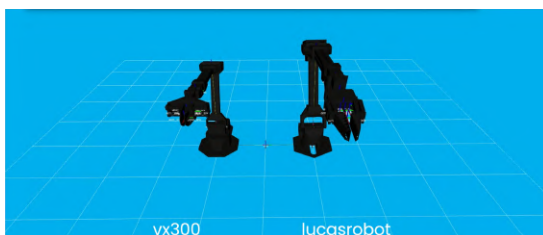
(d) Abertura das garras.



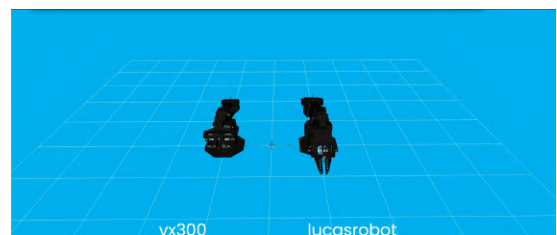
(e) Fechamento das garras.



(f) Movimentação em  $45^\circ$  da junta da base.



(g) Retorno para a posição *Home*.



(h) Retorno para a posição *Sleep*.

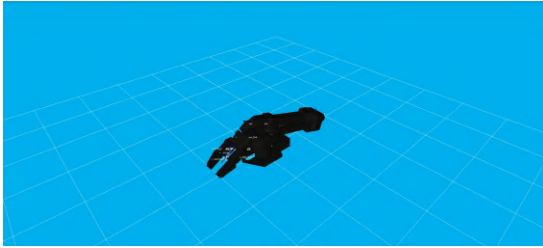
Fonte: Autor, 2023.

esforço máximo que pode ser realizado pela garra.

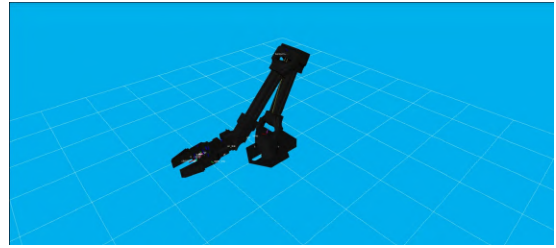
### 5.6.1 Demonstrações de pega

Na elaboração e desenvolvimento dos dedos da garra, optou-se pelo uso de nervuras internas. Esta abordagem serve para otimizar a manipulação de objetos, focando em maximizar a deformação em áreas críticas, como no centro do dedo, enquanto se minimiza

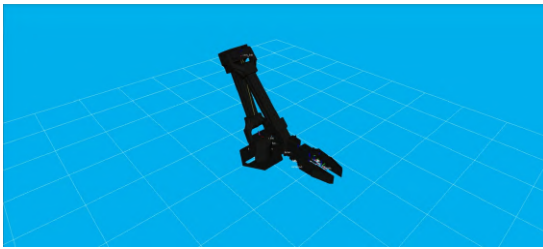
Figura 45 – Simulação de agarre e movimentação de objeto com o robô construído.



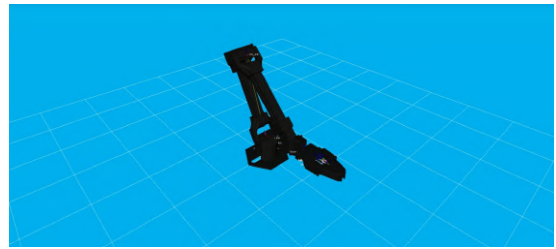
(a) Início na posição *Sleep*.



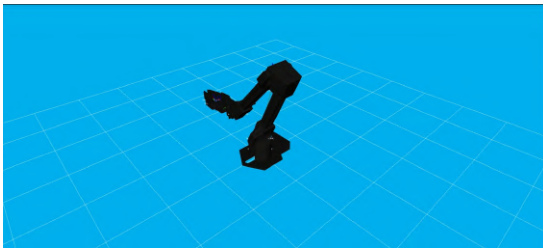
(b) Movimentação para coordenadas  $x = 0,3$  e  $z = 0,2$ .



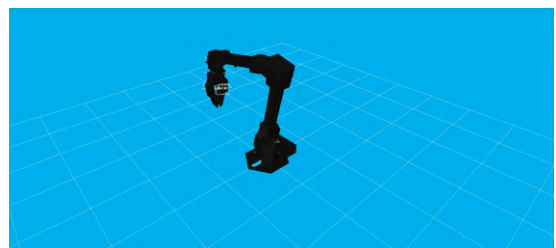
(c) Posição de pega com rotação para  $90^\circ$  na junta da base.



(d) Fechamento da garra.



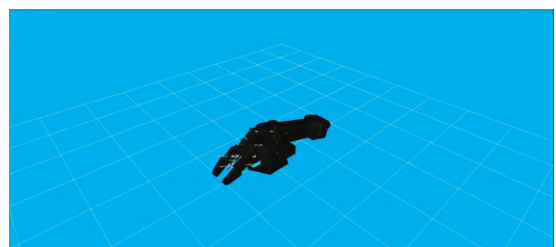
(e) Posição de entrega, com rotação para  $-90^\circ$  na junta da base.



(f) Movimento de  $pitch = 1,5$  (servir).



(g) Retorno para a posição de  $90^\circ$  na junta da base e abertura da garra.



(h) Retorno para a posição *Sleep*.

Fonte: Autor, 2023.

nas extremidades, conforme ilustrado na Figura 49, em que ocorre um ajuste da garra para se adaptar ao formato de uma maçã. Foram consideradas dois tipos de pegadas: a pega superior e o agarre lateral do objeto.

Essa diferenciação na capacidade de deformação dos dedos amplia significativamente a área de contato com os objetos. Como resultado, observa-se um aumento na aderência, o que é vital para o manuseio eficaz e seguro.

Figura 46 – Comparação para a posição *home* na interface do MoveIt e no robô físico.



Fonte: Autor, 2023.

Adicionalmente, os dedos apresentam centralizadores verticais, que colaboram ao segurar objetos cilíndricos, como caneta, garfo, colher, entre outros. Esse aspecto pode ser verificado na Figura 50, em que foi executada a pega de uma caneta marcadora.

Os esforços de manipulação considerados para esta pesquisa foram determinados com base no manuseio de elementos do cotidiano, de forma a validar a aderência e a suavidade do agarre sem causar danos ao objeto. Para esse propósito, foi escolhido um copo de vidro com paredes 3 mm de espessura (Figura 51). No ensaio, foi aplicado o máximo de força com a garra, a fim de avaliar a possibilidade de ruptura do copo de vidro. Com esse cenário, a garra robótica conseguiu executar a pega com firmeza e não houve danos ao objeto, conforme era desejado.

### 5.6.2 Esforço máximo da garra

A partir dos procedimentos demonstrados na Seção 3.9 referente à Bancada Experimental no Capítulo de Materiais e Métodos, foi possível realizar a análise de esforços da garra. A Figura 52 apresenta a garra se deformando no processo de fechamento para pegar o cilindro de testes.

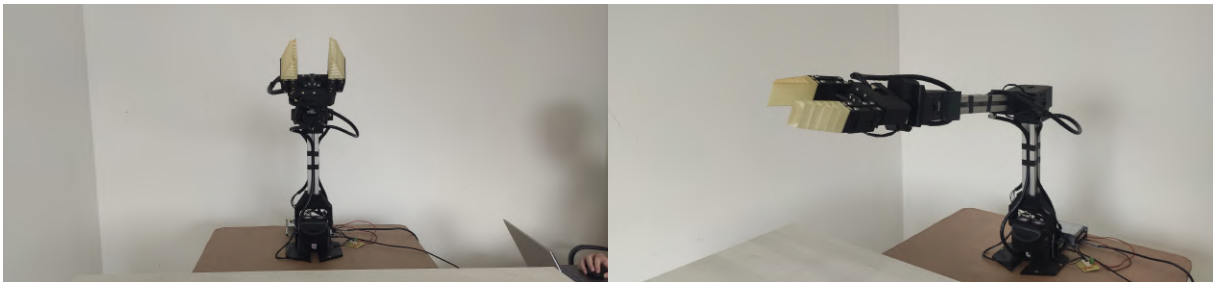
O sinal da força é então adquirido via célula de carga e passa por um posterior processamento. A Figura 53 expressa o comportamento de resposta da célula de carga de acordo com o fechamento, chegando a atingir valores próximos de 40 N. Após o pico, ponto em que ocorre fechamento total, a garra é aberta até a amplitude total, não havendo mais contato com a célula de carga.

Figura 47 – Execução de uma sequência de movimentos com o robô físico.



(a) Rotação de  $-90^\circ$  na junta da base.

(b) Rotação de  $-45^\circ$  na junta do cotovelo.



(c) Rotação de  $45^\circ$  na junta do cotovelo.

(d) Rotação de  $45^\circ$  no punho.



(e) Rotação de  $45^\circ$  na junta do punho e de  $45^\circ$  na junta da base.

(f) Rotação de  $-45^\circ$  na junta do punho.

Fonte: Autor, 2023.

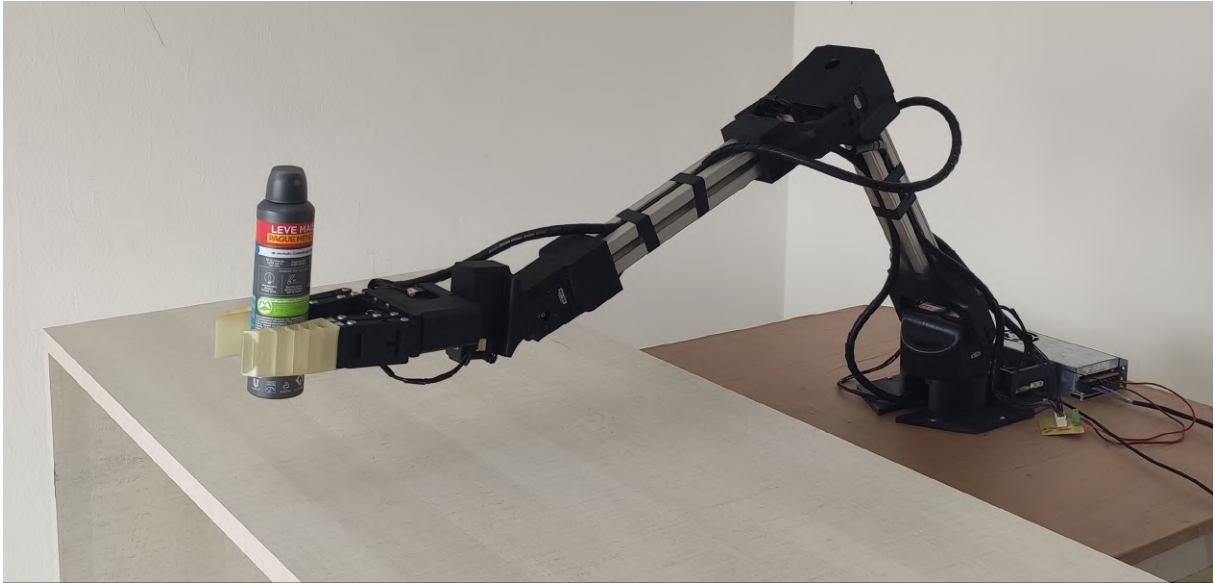
## 5.7 COMPARATIVO FINAL COM O VIPER X 300

A estrutura da cadeia cinemática é a mesma tanto para o Viper X 300 quanto para o robô construído, conforme observado na Figura 35. Por outro lado, os comprimentos dos elos são diferentes. A relação evidenciando essa diferença entre as dimensões é apontada na Tabela 16. Destaca-se que o elo referente ao segundo perfil de alumínio ( $l_3$ ) e o elo da garra ( $l_4$ ) correspondem à maior disparidade.

Com base nos aspectos relatados ao longo desse documento, é possível estabelecer um comparativo completo das especificações dos dois robôs, como visto na Tabela 17. Nota-se que o robô comercial Viper X 300 apresenta uma massa menor e maior capacidade de carga. Enquanto isso, o robô construído ficou um pouco mais pesado, porém possibilita a utilização do mesmo ambiente de programação e possui os mesmos graus de

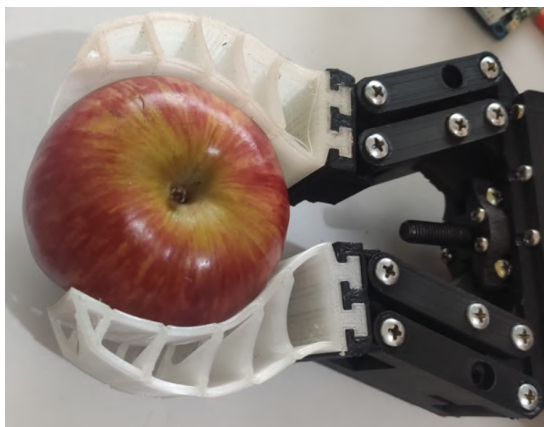


Figura 48 – Execução de movimento para pega de objeto.



Fonte: Autor, 2023.

Figura 49 – Garra realizando pega de uma maçã.



(a) Pega na posição lateral.



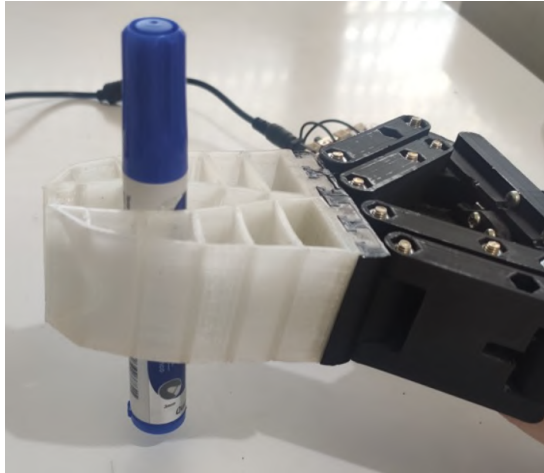
(b) Pega na posição superior.

Fonte: Autor, 2023.

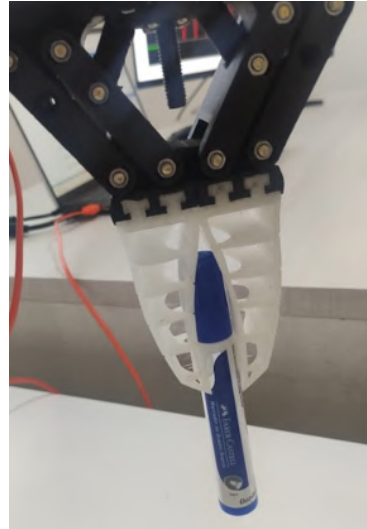
liberdade, além de contar com uma garra flexível.

No robô construído foram encontradas algumas limitações de movimentação, principalmente na junta do ombro. Isso faz com que o alcance desse robô seja menor que o Viper X 300 e que as trajetórias executadas tenham que ser mais simples.

Figura 50 – Garra realizando pega de uma caneta.



(a) Pega na posição lateral.



(b) Pega na posição superior.

Fonte: Autor, 2023.

Figura 51 – Garra realizando pega de um copo de vidro.



(a) Pega na posição lateral.



(b) Pega na posição superior.

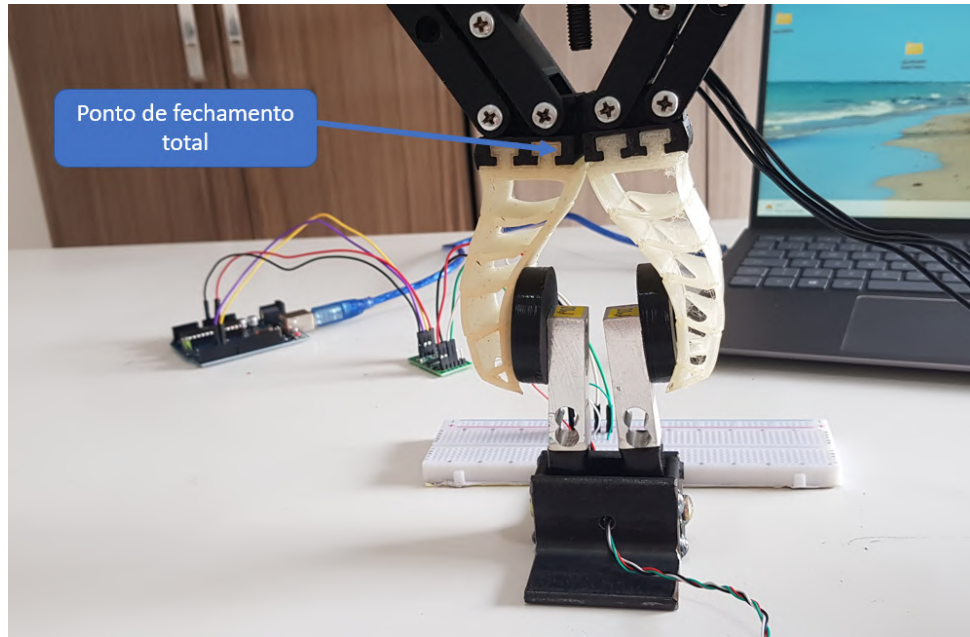
Fonte: Autor, 2023.

## 5.8 LIMITAÇÕES E CONDIÇÃO DE OPERAÇÃO

Nos testes executados com o robô físico construído, foram identificados pontos de melhoria para a performance do sistema. Verificou-se que a condição de limitação acabou sendo limitada por dois aspectos:

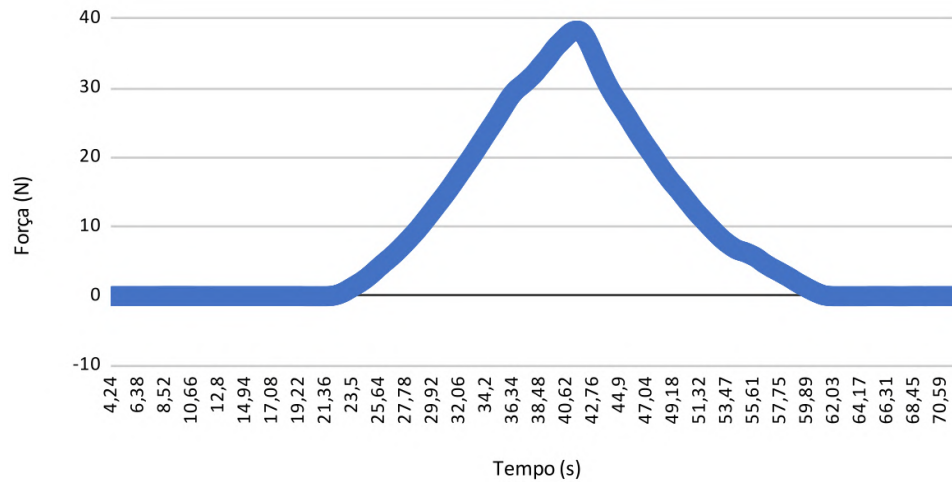
1. **Dimensões:** O robô proposto neste trabalho foi baseado no Viper X 300, porém, suas dimensões, tanto em relação à massa quanto em relação ao comprimento

Figura 52 – Aquisição de dados do fechamento da garra.



Fonte: Autor, 2023.

Figura 53 – Resposta da célula de carga de acordo com o fechamento e reabertura da garra.



Fonte: Autor, 2023.

Tabela 16 – Comparativo de dimensões do Viper X 300 e do robô construído.

Dimensão	Viper X 300 (mm)	Robô construído (mm)
$l_1$	126,75	115
$l_2$	305,94	395,38
$l_3$	300	430
$l_4$	204,21	320
$b_1$	300	385
$b_2$	60	90

Fonte: Autor, 2023.



Tabela 17 – Comparativo de especificações do Viper X 300 e do robô construído.

Especificação	Viper X 300	Robô construído
Massa	3,63 kg	4,2 kg
Graus de Liberdade	5	5
Número de servomotores	8	8
Microcontrolador	DYNAMIXEL U2D2	OpenCR
Garra	Rígida	Flexível
Ambiente de programação	<i>Interbotix ROS Packages</i>	<i>Interbotix ROS Packages</i>

Fonte: Autor, 2023.

dos elos, são maiores. Essa característica fez com que os motores não conseguissem atuar para alcançar posições críticas, limitando a capacidade de movimentação. Notou-se isso, pelo sinal de *overload* nos motores do cotovelo e do ombro, quando a requisição de torque era elevada.

2. **Colisão:** Conforme foram executados *scripts* de movimentação, verificou-se que o robô algumas vezes vinha a colidir com a mesa à qual estava parafusado. Isso acontecia tanto pelo tamanho da garra quanto pelos últimos elos do robô serem longos.

A solução sugerida para esses problemas é a diminuição dos elos do robô (em torno de 10 cm), através do corte do perfil de alumínio. Dessa forma, não seria necessário refazer as peças de impressão 3D. Isso pode tornar o robô mais leve e facilitar o desempenho dos motores. Com a implementação dessa mudança, o braço robótico pode ter, efetivamente, um resultado similar ao do Viper X 300 na execução de trajetórias complexas.

## 6 CONCLUSÃO

Neste trabalho, foram detalhadas as etapas de desenvolvimento, construção e validação de um robô manipulador. O projeto abrangeu desde o desenho das peças mecânicas até a integração de componentes eletrônicos. As peças foram criadas utilizando tecnologia de manufatura aditiva, e os parâmetros de impressão, que afetam a qualidade final, foram analisados. O robô foi montado em um suporte fixo para otimizar seu espaço de trabalho. A configuração do *software* foi adaptada para se alinhar com as bibliotecas da *Interbotix*, permitindo a utilização de suas funcionalidades. Realizaram-se testes tanto em simulação quanto com o robô físico. Além disso, identificaram-se oportunidades de aprimoramento e conduziu-se uma validação experimental, focando especialmente na garra do robô.

Os objetivos estabelecidos neste estudo, que incluíam investigar a arquitetura de manipuladores robóticos na literatura acadêmica e os modelos da *Interbotix*, foram alcançados. Realizou-se uma análise, destacando os componentes de *hardware* e *software* desses sistemas. Durante a fase de construção, o robô foi montado, respeitando os requisitos mecânicos e elétricos previstos no projeto. A implementação física foi acompanhada pelo uso de algoritmos de controle, capazes de gerenciar as movimentações do robô. Ademais, foi validada a aplicação das ferramentas fornecidas pela *Interbotix* no robô construído. A análise da cinemática direta, tanto no modelo quanto nos robôs da *Interbotix*, revelou um grau significativo de similaridade entre eles. Os robôs possuem dimensões diferentes, mas são compostos por uma cadeia cinemática similar. Os arquivos de montagem e os códigos apresentados e desenvolvidos neste trabalho estão documentados no *Github*, que pode ser acessado pelo link: <https://github.com/lucrambo>.

Durante a realização deste estudo, emergiram desafios significativos associados aos aspectos físicos do robô, particularmente na fase de montagem e execução de movimentos. Foi identificada a necessidade de ajustes nas peças após a impressão para garantir a conformidade com as especificações do projeto. Observou-se também que a capacidade de movimentação efetiva do robô, em comparação com os modelos teóricos, era limitada por fatores como a relação entre o torque dos motores e as dimensões físicas da estrutura.

A implementação prática do projeto revelou ótimos resultados. Ficou demonstrado que o robô é capaz de realizar as trajetórias propostas. Os aprimoramentos identificados ao longo do processo representaram fontes valiosas de aprendizado. Tais experiências sublinham a importância da experimentação prática na engenharia e contribuem significativamente para a formação de capacidade de execução enquanto profissional da área.

## 6.1 TRABALHOS FUTUROS

Como sugestões para a continuação desse projeto destacam-se a implementação de um sistema de visão computacional em conjunto com o robô e a substituição das peças de impressão 3D por componentes de alumínio, consolidando uma versão definitiva.

## REFERÊNCIAS

3DLAB. *Propriedades dos materiais para impressora 3D: Análise do gráfico do ensaio de tração*. 3DLAB, 2017. Acesso em 17 de junho 2023. Disponível em: <<https://3dlab.com.br/propriedades-dos-materiais-para-impressora-3d/>>.

BANSAL, N. et al. Robotic arm by using 3d printing and polylactic acid (pla) wire challenges and solutions. In: **2023 1st International Conference on Intelligent Computing and Research Trends (ICRT)**. [S.l.: s.n.], 2023. p. 1–11.

BCN3D. **BCN3D MOVEO**: A fully open source 3d printed robot arm. BCN3D, 2016. Acesso em 17 de junho 2023. Disponível em: <<https://www.bcn3d.com/bcn3d-moveo-the-future-of-learning-robotic-arm/>>.

\_\_\_\_\_. **BCN3D-Moveo: A fully Open Source 3D printed robot arm**. [S.l.]: GitHub, 2016. <https://github.com/BCN3D/BCN3D-Moveo>.

BURKHARDT, F. et al. Pandemic-driven development of a medical-grade, economic and decentralized applicable polyolefin filament for additive fused filament fabrication. **Molecules**, v. 25, n. 24, 2020. ISSN 1420-3049. Disponível em: <<https://www.mdpi.com/1420-3049/25/24/5929>>.

COSTA, J.; MACHADO, T. A. U.; CARNEIRO, M. Implementation and validation of thor 3d printed open source robotic arm. **IEEE Latin America Transactions**, v. 18, n. 05, p. 907–913, 2020.

CRAIG, J. J. **Introduction to Robotics: Mechanics and Control**. Pearson/Prentice Hall, 2005. (Addison-Wesley series in electrical and computer engineering: Control engineering). ISBN 9780131236295. Disponível em: <<https://books.google.com.br/books?id=ZJkOSgAACAAJ>>.

CREALITY. **CR-10S Pro V2 3D Printer**: User manual. CREALITY 3D, 2021. Acesso em 10 de julho 2023. Disponível em: <<https://manuals.plus/wp-content/sideoads/creality-cr-10s-pro-v2-3d-printer-manual-original.pdf>>.

DOBOT. 2023. <https://www.dobot-robots.com/products/education/magician.html>. Acessado: 2024-01-04.

ELGENEIDY, K. et al. Characterising 3d-printed soft fin ray robotic fingers with layer jamming capability for delicate grasping. In: . [S.l.: s.n.], 2019. p. 143–148.

FERREIRA, N.; ROMANO, V.; MACHADO, J. T. **Robótica Industrial Aplicação na Indústria de Manufatura e de Processos**. [S.l.: s.n.], 2002. ISBN 85-212-0315-2,.

GU, D. **Laser Additive Manufacturing of High-Performance Materials**. Springer, 2015. ISBN 9783662460900. Disponível em: <<https://books.google.com.br/books?id=Q5nfzQEACAAJ>>.

IFR. **IFR Presents World Robotics Report 2022**: The new world robotics report shows an all-time high of 517,385 new industrial robots installed in 2021 in factories around the world. Automation.com, 2022. Acesso em 30 de dezembro 2022. Disponível em: <<https://www.automation.com/en-us/articles/october-2022/ifr-presents-world-robotics-report-2022>>.

KRIMPENIS, A. A.; PAPAPASCHOS, V.; BONTARENKO, E. Hydrax, a 3d printed robotic arm for hybrid manufacturing. part i: Custom design, manufacturing and assembly. **Procedia Manufacturing**, v. 51, p. 103–108, 2020. ISSN 2351-9789. 30th International Conference on Flexible Automation and Intelligent Manufacturing (FAIM2021). Disponível em: <<https://www.sciencedirect.com/science/article/pii/S2351978920318710>>.

LAURITZEN, C. **Development of a simulation platform and testing of a 5 degrees of freedom ReactorX 150 robotic arm manipulator**. 2023. Dissertação (Mestrado) — University of South-Eastern Norway, 2023.

LYNCH, K.; PARK, F. **Modern Robotics**. Cambridge University Press, 2017. ISBN 9781107156302. Disponível em: <<https://books.google.com.br/books?id=YUkTnQAACAAJ>>.

MACHNATA. **Braço robótico com renderização 3d de robô industrial de 6 eixos**: Ilustração. Free Pik, 2023. Acesso em 18 de maio 2023. Disponível em: <[https://br.freepik.com/fotos-premium/braco-robotico-com-renderizacao-3d-de-robotico-industrial-de-6-eixos\\_37520058.htm](https://br.freepik.com/fotos-premium/braco-robotico-com-renderizacao-3d-de-robotico-industrial-de-6-eixos_37520058.htm)>.

Malvido Fresnillo, P. et al. Extending the motion planning framework moveit with advanced manipulation functions for industrial applications. **Robotics and Computer-Integrated Manufacturing**, v. 83, p. 102559, 2023. ISSN 0736-5845. Disponível em: <<https://www.sciencedirect.com/science/article/pii/S0736584523000352>>.

MICALLEF, J. **Beginning Design for 3D Printing**. Apress, 2015. ISBN 9781484209479. Disponível em: <<https://books.google.com.br/books?id=O982jwEACAAJ>>.

MINIPA. **Dobot Magician**: Robô educacional inteligente. MINIPA, 2016. Acesso em 17 de junho 2023. Disponível em: <<https://www.minipa.com.br/categoria/3/linha-educacional-robotica/dobot-magician>>.

MURRAY, R.; LI, Z.; SASTRY, S. **A Mathematical Introduction to Robotic Manipulation**. CRC Press, 2017. ISBN 9781138440166. Disponível em: <<https://books.google.com.br/books?id=ntQltAEACAAJ>>.

PickNik Robotics. **MoveIt**: Documentation. PickNik Robotics, 2023. Acesso em 28 de outubro de 2023. Disponível em: <<https://moveit.ros.org>>.

PREEZ, R. D. **3D 6-DOF Serial Arm Robot**: Kinematics and implementation in linuxcnc. [s.n.], 2014. SA-CNC-CLUB. Disponível em: <<http://alvarestech.com/temp/RoboAsealRB6S2-Fiat/CinematicaExemplosManuaisConfigurador-DH-EMC/EMC-6-axis-serial-robot-2.pdf>>.

PURDON, K.; SETATI, T.; MARAIS, S. Manufacturing and evaluation of the open-source ar3 robot arm for educational uses. In: **2021 Rapid Product Development Association of South Africa - Robotics and Mechatronics - Pattern Recognition Association of South Africa (RAPDASA-RobMech-PRASA)**. [S.l.: s.n.], 2021. p. 01–05.

ROBOTIS. **Robotis e-manual**: Products, software, parts. ROBOTIS, 2023. Acesso em 15 de julho 2023. Disponível em: <<https://emanual.robotis.com/>>.

ROS. **Robot Operating System**: Documentation. ROS, 2023. Acesso em 28 de outubro de 2023. Disponível em: <<https://wiki.ros.org>>.

SANTOS, V. M. **Robótica Industrial**: Apontamentos teóricos. [s.n.], 2004. Universidade de Aveiro - Departamento de Engenharia Mecânica. Disponível em: <<http://www.ece.ufrgs.br/~rventura/RoboticalIndustrial.pdf>>.

SCIAVICCO, L.; SICILIANO, B. **Modelling and Control of Robot Manipulators**. Springer London, 2001. (Advanced Textbooks in Control and Signal Processing). ISBN 9781852332211. Disponível em: <<https://books.google.com.br/books?id=v9PLbcYd9aUC>>.

SHINTAKE, J. et al. Soft robotic grippers. **Advanced Materials**, v. 30, n. 29, p. 1707035, 2018. Disponível em: <<https://onlinelibrary.wiley.com/doi/abs/10.1002/adma.201707035>>.

SPONG, M.; HUTCHINSON, S.; VIDYASAGAR, M. **Robot Modeling and Control**. Wiley, 2005. ISBN 9780471649908. Disponível em: <<https://books.google.com.br/books?id=cPhvxwEACAAJ>>.

TROSSEN ROBOTICS. **Viper X 300 Robot Arm**: Description of the robot arm. Trossen Robotics, 2023. Acesso em 24 de junho 2023. Disponível em: <<https://www.trossenrobotics.com/viperx-300-robot-arm.aspx>>.

VOLPATO, N. **Manufatura aditiva: tecnologias e aplicações da impressão 3D**. Editora Blucher, 2021. ISBN 9788521211518. Disponível em: <<https://books.google.com.br/books?id=ni9dDwAAQBAJ>>.

Wiznitzer et al. **Interbotix ROS Manipulators**: Bsd-3-clause. Interbotix, 2023. Acesso em 09 de agosto 2023. Disponível em: <[https://github.com/Interbotix/interbotix\\_ros\\_manipulators](https://github.com/Interbotix/interbotix_ros_manipulators)>.

XMARKLABS. **Roll, Pitch, Yaw**: Rotation angles. XMarkLabs, 2017. Acesso em 18 de maio 2023. Disponível em: <<https://xmarklabs.com/blog/2017/08/mysteries-tracking-rollpitchyaw/>>.

## APÊNDICE A – COMANDO DE DOIS ROBÔS SIMULTANEAMENTE

Código A.1: Reprodução do código para comandar dois robôs simultaneamente.

```
1 import math
2 import rospy
3 from threading import Thread
4 from interbotix_xs_modules.arm import InterbotixManipulatorXS
5
6 def robot_1():
7     robot_1 = InterbotixManipulatorXS(robot_model="vx300", robot_name="
8     arm_1", moving_time=1.0, gripper_pressure=1.0, init_node=False)
9     robot_1.arm.set_ee_pose_components(x=0.3, z=0.2)
10    robot_1.arm.go_to_home_pose()
11    robot_1.gripper.open()
12    robot_1.gripper.close()
13    robot_1.arm.set_single_joint_position("waist", -math.pi/4.0)
14    robot_1.arm.set_single_joint_position("waist", 0)
15    robot_1.arm.go_to_sleep_pose()
16
17 def robot_2():
18     robot_2 = InterbotixManipulatorXS(robot_model="lucasrobot",
19     robot_name="arm_2", moving_time=1.0, gripper_pressure=1.0, init_node=
20     False)
21    robot_2.arm.set_ee_pose_components(x=0.3, z=0.2)
22    robot_2.arm.go_to_home_pose()
23    robot_2.gripper.open()
24    robot_2.gripper.close()
25    robot_2.arm.set_single_joint_position("waist", math.pi/4.0)
26    robot_2.arm.set_single_joint_position("waist", 0)
27    robot_2.arm.go_to_sleep_pose()
28
29 def main():
30     rospy.init_node("xsarm_dual")
31     Thread(target=robot_1).start()
32     Thread(target=robot_2).start()
33
34 if __name__ == '__main__':
35     main()
```

## APÊNDICE B – AGARRAR E MOVIMENTAR OBJETO (*PICK AND PLACE*)

Código B.1: Reprodução do código para agarrar e movimentar um objeto.

```
1 from interbotix_xs_modules.arm import InterbotixManipulatorXS
2 import numpy as np
3
4 def main():
5     bot = InterbotixManipulatorXS("lucasrobot", "arm", "gripper")
6     bot.arm.set_ee_pose_components(x=0.3, z=0.2)
7     bot.arm.set_single_joint_position("waist", np.pi/2.0)
8     bot.gripper.open()
9     bot.gripper.close()
10    bot.arm.set_single_joint_position("waist", -np.pi/2.0)
11    bot.arm.set_ee_cartesian_trajectory(pitch=1.5)
12    bot.arm.set_ee_cartesian_trajectory(pitch=-1.5)
13    bot.arm.set_single_joint_position("waist", np.pi/2.0)
14    bot.gripper.open()
15    bot.arm.go_to_home_pose()
16    bot.arm.go_to_sleep_pose()
17
18 if __name__ == '__main__':
19     main()
```



NUP: 23081.159102/2023-88

Prioridade: Normal

**Homologação de ata de defesa de TCC e estágio de graduação**

125.322 - Bancas examinadoras de TCC: indicação e atuação

**COMPONENTE**

Ordem	Descrição	Nome do arquivo
9	Trabalho de Conclusão de Curso	TCC_LUCAS_BRAÇO_ROBÓTICO_REVISADO.pdf

**Assinaturas**

**23/01/2024 11:07:01**

ANSELMO RAFAEL CUKLA (PROFESSOR DO MAGISTÉRIO SUPERIOR (Ativo))  
07.54.00.00.0.0 - DEPARTAMENTO DE PROCESSAMENTO DE ENERGIA ELÉTRICA - DPEE

**23/01/2024 11:37:40**

DIEGO BERLEZI RAMOS (Coordenador(a) de Curso)  
07.09.02.00.0.0 - CURSO DE ENGENHARIA ELÉTRICA - CEELE



Código Verificador: 3767351

Código CRC: 55ae7f37

Consulte em: <https://portal.ufsm.br/documentos/publico/autenticacao/assinaturas.html>

