

## **INTERFACE PARA POSICIONAMENTO DE MANIPULADOR COM 5 GRAUS DE LIBERDADE EM APLICAÇÕES SUBMARINAS**

Lucas Oliveira Vivian

Projeto de Graduação



# **INTERFACE PARA POSICIONAMENTO DE MANIPULADOR COM 5 GRAUS DE LIBERDADE EM APLICAÇÕES SUBMARINAS**

**Aluno: Lucas Oliveira Vivian**

**Orientador: Mauro Speranza Neto**

**Co-orientadora: Vivian Suzano Medeiros**

Trabalho apresentado com requisito parcial à conclusão do curso de Engenharia de Controle e Automação na Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro, Brasil.

## Agradecimentos

Primeiramente, agradeço aos meus pais, Ricardo e Gilza, que sempre me encorajaram e incentivaram a seguir a carreira de engenharia me proporcionando desde pequeno, do meu pai, uma visão e conhecimento técnicos, me apresentando os primeiros conceitos de eletrônica e sempre me incluindo em seus projetos. Da minha mãe, toda a criatividade e imaginação em projetos mão na massa, artes e reutilização de materiais nos projetos. Obrigado por me tornarem o homem que sou hoje.

À minha irmã, Lara, pelas boas risadas ao longo dos anos, por ser minha grande companhia em discussões, consumo e idas a eventos da cultura pop.

À toda minha família, avós, tios, primos e amigos, por todo o apoio, companheirismo e por todos os encontros que são sempre inesquecíveis.

À minha melhor amiga e parceira de vida, Natália, que desde os tempos do colégio escuta minhas reclamações e dificuldades de estudar, que nunca desistiu de mim e esteve presente em todos os momentos provendo o maior apoio possível e me incentivando com seu amor e carinho.

Aos melhores amigos Wanderson Assis e Marcelo Ramalho com uma amizade de quase duas décadas sempre me proporcionando boas risadas e ideias, além do apoio incondicional em todas as etapas da minha vida.

Às minhas melhores amigas Luiza Freire, Maria Eduarda Neves e Maria Manuela Coelho por todos os momentos que tivemos juntos, é sempre bom estar com vocês. Em especial à minha grande amiga Marcela Santos que apesar de não estar mais conosco, sempre foi e sempre será uma grande inspiração na minha vida.

À minha segunda família, Nicho, que a cada dia cresce mais, Adulto, Agnelo, Aleixo, Àvila, Caruso, Creatina, Feder, Luiz, Marcelo, Matias, Martins, Neves, Nunes, Paulo, Península, Ramalho, Seci, Tom e VH, só tenho a agradecer a vocês por essa amizade, vocês são como irmãos para mim.

Ao meu grande e eterno companheiro de Controle e Automação, do início ao fim da faculdade, Marcelo Rocha, durante todas as matérias, sempre um ajudando ao outro, é uma honra finalmente estar me formando ao seu lado.

À Yan Dalton, pela imensa amizade criada nesses últimos anos de PUC, pela companhia nas madrugadas durante a pandemia sendo crucial na resolução dos problemas deste projeto e por me inspirar a continuar os meus projetos, fazendo com que eu deseje ser um profissional cada vez melhor.

Menções honrosas aos grandes amigos: Bruno Fischer, Bruno Franco, Guilherme Fernandes, Lucas Ramos, Arthur Patrício, Vinicius Henriques, Maria Elisa, Guilherme Xavier, Alcides, Rodrigo Nogueira, Matheus Baliú, grupo Bribinha e à Space Factory Co. por todos os grandes momentos proporcionados antes e durante a pandemia.

À Paul Falstad por desenvolver o software responsável por tornar incrivelmente mais fácil a compreensão e aumentar o meu interesse ainda mais por circuitos eletrônicos.

Ao Centro de Pesquisa em Tecnologia de Inspeção (CPTI), em especial Miguel Freitas, pela oportunidade e aos funcionários, técnicos e estagiários pelos conhecimentos compartilhados e pelas boas risadas.

Ao meu orientador, Mauro Speranza Neto, pela oportunidade e apoio na elaboração deste projeto assim como as abordagens da teoria em casos práticos como o de veículos de Fórmula 1 em suas matérias, que tornaram ainda mais interessante o aprendizado de Controle e Automação.

À minha co-orientadora, Vivian Suzano Medeiros, pela oportunidade, pela enorme paciência em reexplicar conceitos, pela companhia durante todos os testes, por todas as reuniões, pela grande ajuda na hora de resolver os problemas e por fim por servir como imensa inspiração dentro da área de Controle e Automação bem como no quesito profissional do que é ser um engenheiro.

À Pontifícia Universidade Católica do Rio de Janeiro por todos esses anos de atmosfera incrível com seus laboratórios, restaurantes, vila, bibliotecas e o campus como um todo. Carregarei as experiências e amizades aqui adquiridas por toda a vida.

À minha terceira família, Equipe Reptiles, que merece uma página própria de agradecimentos por ter proporcionado uma das maiores experiências da minha vida, formando meu caráter, melhorando meu lado pessoal e profissional, só tenho a agradecer pela oportunidade, por todas as gerações que vieram antes de mim e por todas que virão. É impossível agradecer um por um, mas é necessário agradecer à aqueles que mais tiveram impacto na minha vida.

Primeiramente os capitães Vendramini e Percalço que não ousaram desistir de mim e foram cruciais na criação e implementação do subsistema da eletrônica no baja. Em segundo lugar, todo o pessoal do subsistema da eletrônica: Cunha, Darze, Sid, Gi, Marina, Davi e Luig, só tenho a agradecer por tudo que me ensinaram e por elevarem o nível da eletrônica tornando-a uma das melhores do Brasil, eu não poderia estar mais orgulhoso.

Ao professor José Parise por trazer o baja para a PUC-Rio, pelo apoio e pela inspiração. Ao professor José Paulo que não foi apenas um imenso suporte à equipe mas também um grande amigo pessoal sempre discutindo filosofias e como companhia para escutar um bom Rock'n Roll.

Aos grandes amigos Millhouse, Nico, Baiano, Samurai, Henrique, Godinho, Erik, Caipira, Lelê, Biel, Lucilady, Jessica, Wang, Ju, Lukita, Turba, Rulli, Mendonça, Rafa, Richard, Rod, Turnes, Paladino, Ju Emery, Machadinho, Tristão, Bruno, Ana, André, Bigo, Carol, Diogo, Monte Santo, Dedão, Motta, Flávia, Pohl, Stélio, Leo, Godoy, Han, Portari, Luísa, Marcão, Matheus Jorge, Milantinha, Nico2, Shrek, Zé, Fei e a todos os outros integrantes da Reptiles por toda a amizade, carinho e experiência proporcionados no laboratório, nas competições e em todas as resenhas.

*"Try not. Do or do not. There is no try."*

-Yoda



## Resumo

O uso de manipuladores robóticos em operações *subsea* permite a instalação e a operação de diversos equipamentos de inspeção necessários para manter a segurança do processo de produção de petróleo no Brasil. Tipicamente, os manipuladores disponíveis no mercado são operados através de uma interface de controle com *joystick*, sendo dependente então da precisão humana.

Neste trabalho é realizada a simulação da cinemática direta e inversa do manipulador ARM 5E Mini de cinco graus de liberdade, bem como a elaboração de uma interface de controle de posicionamento, permitindo o controle individual dos ângulos de cada junta do manipulador ou da trajetória desejada do *end-effector*, dispensando o uso do *joystick*. Além disso, a interface também conta com leitura de dados como temperatura, corrente, posição e velocidade das juntas para constante monitoramento do sistema.

A implementação da interface de controle desenvolvida neste trabalho irá viabilizar o uso do manipulador ARM 5E Mini em operações *subsea* realizadas pelo Centro de Pesquisa em Tecnologia de Inspeção. Todos os programas foram desenvolvidos em MATLAB e Python e foram validados em testes experimentais com o manipulador.

**Palavras-chave:** Manipuladores Robóticos, Cinemática Direta, Cinemática Inversa, Notação Denavit-Hartenberg, Ferramentas Submarinas

## **FIVE-DEGREE-OF-FREEDOM MANIPULATOR POSITIONING INTERFACE FOR SUBSEA APPLICATIONS**

### **Abstract**

The use of robotic manipulators in subsea operations allows the installation and operation of several inspection equipment needed to maintain the security of oil's production process in Brazil. Typically, the available manipulators in the market are operated through a control interface with a joystick, being dependent on human precision.

In this project, a simulation of the forward and inverse kinematics for the ARM 5E Mini manipulator of five degrees of freedom is performed, as well as the elaboration of positioning control interface, allowing the individual control of the angles of each joint of the manipulator or the desired trajectory of the end-effector, dispensing the use of the joystick.

In addition, the interface also features data reading such as temperature, current, position and velocity of the joints for constant system monitoring.

The implementation of the control interface developed in the work will enable the use of the manipulator in subsea operations carried out by the Inspection Technology Research Center at PUC-Rio (CPTI PUC-Rio). All the algorithms were developed in Matlab and Python 2.7 and were validated through experimental tests with the manipulator.

**Keywords: Robotic Manipulators, Forward Kinematics, Inverse Kinematics, Denavit-Hartenberg Notation, Subsea Tools**

## Lista de Figuras

1	ROV Shepard Beachcomber H800 com dois manipuladores BM5E Micro [1]. . . . .	3
2	Fatores que afetam a performance do manipulador subaquático [2]. . . . .	4
3	Na linha de cima, da esquerda para direita, um dos primeiros autômatos [3] e o livro "Eu, Robô", que introduz as leis [4]. Na linha de baixo, respectivamente, <i>Unimate</i> , o primeiro robô industrial [5] e um robô aspirador atual [6]. . . . .	5
4	Manipulador planar de 3 juntas rotativas [7]. . . . .	6
5	Representação das juntas rotativa e prismática em 2D e 3D [8]. . . . .	6
6	Representação dos graus de liberdade de um sistema tridimensional [9]. . . . .	6
7	Diagrama com todos os seis graus de liberdade do manipulador Viper 6-DOF [10]. . . . .	7
8	Exemplo de diagrama da cadeia cinemática [8]. . . . .	7
9	Sistema de coordenada $o1x1y1$ rotacionado em $\theta$ graus em relação ao sistema $o0x0y0$ [8]. . . . .	8
10	Robô manipulador PUMA com suas juntas e elos [11]. . . . .	10
11	Definição dos parâmetros Denavit-Hartenberg em sistemas [8]. . . . .	11
12	ARM 5E Mini com juntas nomeadas. . . . .	15
13	Ferramenta de controle manual do manipulador e fonte de alimentação do sistema. . . . .	16
14	Conexão em "T". . . . .	16
15	Vaso de pressão. . . . .	17
16	Garra cortadora - agarrador padrão. . . . .	17
17	Captura de tela do <i>software</i> de controle provido pela ECA [12]. . . . .	18
18	<i>Software</i> Fabricante - Motor 1 ( <i>Shoulder</i> ) parado [12]. . . . .	19
19	<i>Software</i> Fabricante - Motor 1 ( <i>Shoulder</i> ) movimentando [12]. . . . .	20
20	Área de trabalho com ponto de fixação do manipulador e computadores utilizados. . . . .	21
21	Representação do manipulador com os sistemas de coordenadas já arbitrados em cada junta. . . . .	22
22	Parâmetros Denavit-Hartenberg teóricos. . . . .	22
23	Parâmetros DH obtidos empiricamente. . . . .	23
24	<i>Workspace</i> manipulador. . . . .	24
25	Vista superior <i>workspace</i> manipulador. . . . .	25
26	Manipulador na posição inicial com a função <i>teach</i> do <i>toolbox</i> de Peter Corke. . . . .	25
27	<i>Workspace</i> plotado sobre a interface <i>teach</i> . . . . .	26
28	Exemplos da interface <i>teach</i> com o <i>workspace</i> . . . . .	26
29	Teste manipulador posição 30° em cada junta. . . . .	28
30	Resultado da otimização para o primeiro teste. . . . .	28
31	Teste manipulador configurado com juntas em 20°, 45° e 90°. . . . .	29
32	<i>Print</i> das informações recebidas do manipulador exibidas no terminal. . . . .	30
33	<i>Print</i> da rotina de perguntas no terminal para enviar mensagem para o manipulador. . . . .	31
34	Manipulador posição 1 Matlab e real. . . . .	32
35	Manipulador posição 2 Matlab e real. . . . .	32
36	Manipulador posição 3 Matlab e real. . . . .	32
37	<i>Print</i> da interface de trajetória . . . . .	34
38	Posicionamento dos componentes do teste. . . . .	35
39	Posicionamento manipulador. . . . .	35
40	Conjunto de pontos da trajetória quadrado. . . . .	36
41	Mesma trajetória desejada percorrida 10 vezes pelo manipulador, em conjunto com a demarcação dos pontos desejados. . . . .	36
42	Trajétórias comparadas: Teórica (Pontos em losango azul), Otimizada (trajetória em laranja), Real (trajetória em amarelo). . . . .	37
43	Coordenadas $x$ e $y$ comparadas: Teórica, Otimizada, Real. . . . .	37
44	Valor real e desejado em AD das três juntas do manipulador. . . . .	38
45	Trajetória triângulo. . . . .	39
46	Coordenadas comparadas: Teórica, Otimizada, Real. . . . .	39
47	Trajetória triangular: coordenadas $x$ e $y$ comparadas (teórica, otimizada e real). . . . .	40
48	Trajetória círculo. . . . .	40
49	Coordenadas comparadas: Teórica, Otimizada, Real. . . . .	41
50	Coordenadas $x$ e $y$ comparadas: Teórica, Otimizada, Real. . . . .	41
51	Conceito da interface de controle de trajetória do manipulador. . . . .	44
52	Conceito da interface de controle - seção de aviso antes do teste. . . . .	44
53	Conceito da interface de controle - seção de cinemática inversa. . . . .	45

## Sumário

<b>1</b>	<b>Introdução</b>	<b>1</b>
a	Motivação	1
b	Problema	1
c	Objetivos	1
<b>2</b>	<b>ROV e Técnicas de Inspeção Submarinas</b>	<b>3</b>
<b>3</b>	<b>Manipuladores Robóticos</b>	<b>5</b>
a	Juntas, elos e graus de liberdade	5
b	Análise cinemática	7
c	Transformação homogênea	8
d	Cinemática direta	9
e	Parâmetros Denavit-Hartenberg	10
f	Cinemática inversa	13
<b>4</b>	<b>ARM 5E Mini</b>	<b>15</b>
a	Especificações	15
b	Software fabricante	17
c	Protocolo de comunicação	18
1	Algoritmo para envio dos dados	18
2	Algoritmo para leitura dos dados	19
d	Estação de trabalho	21
<b>5</b>	<b>Simulações</b>	<b>22</b>
a	Parâmetros Denavit-Hartenberg do ARM 5E Mini	22
b	<i>Workspace</i> do manipulador	24
c	Cinemática direta	25
d	Cinemática inversa	27
<b>6</b>	<b>Testes Experimentais</b>	<b>30</b>
a	Interface de comunicação	30
b	Controle de posição das juntas	31
c	Controle de posição do efector final	33
d	Controle de trajetória	33
e	Preparação para testes	34
f	Trajetoórias	36
g	Custo computacional	42
<b>7</b>	<b>Conclusões e Trabalhos Futuros</b>	<b>43</b>
a	Orientação e ação da ferramenta	43
b	Interface	43
<b>8</b>	<b>Referências</b>	<b>46</b>
<b>A</b>	<b>Apêndice</b>	<b>48</b>
a	Protocolo Manipulador [12]	48
1	Envia dados para manipulador	48
2	Recebe dados do manipulador	50

## 1 Introdução

Ao longo dos anos, diferentes formas de energia foram sendo utilizadas para sustentar diferentes atividades do ser humano. Ainda na história antiga, já havia o uso de óleo como fonte para lampiões entre outros, assim como poços de petróleo datando de 347 DC. Contudo, é só em 1847 que a indústria de óleo e gás inicia sua ascensão a partir das extrações de petróleo a partir do carvão e é na virada do século que as atuais grandes companhias dessa indústria surgem.

Em seguida há a aparição da primeira perfuração na água e o primeiro escopo do que viria a ser uma plataforma petrolífera. Avançando para os dias atuais, existem mais de doze mil plataformas espalhadas pelo mundo extraíndo petróleo para os mais diversos usos descobertos pelo ser humano. A fim de manter o processo de extração mais seguro e estável possível é que surgem as tecnologias de inspeção submarina, sempre à procura de falhas, desgastes ou qualquer possível fator que possa pôr em risco a retirada de petróleo do leito marinho.

À medida que a tecnologia foi avançando, novas formas de inspeção foram surgindo e junto com elas, a robótica, com o objetivo de facilitar o trabalho do homem em realizar atividades em mar aberto e minimizando os riscos de vida inerentes a atividades de risco como, por exemplo, inspeção de tubulações submarinas e *risers* flexíveis (dutos de diversas camadas utilizados para transportar o petróleo extraído do leito marinho). Em termos de inspeção, é possível monitorar o estado das estruturas através de sensoriamento buscando imperfeições ou falhas, bem como realizar reparos como cortes, soldas, entre outros.

É nesse contexto que o uso de manipuladores entra em vigor, reproduzindo os movimentos de um braço humano auxiliando na inspeção de dutos, seja reparando, servindo de apoio para outro robô ou mergulhador, ou para buscar e apanhar objetos que estejam no fundo do mar.

### a Motivação

Como visto em [2], manipuladores subaquáticos de uso comercial geralmente possuem um sistema de teleoperação totalmente dependente no operador localizado no navio ou na plataforma. Este observa a situação através de câmeras e sistemas de sonar e toma decisões de movimento através de *joysticks*, controlando assim o manipulador.

Justamente por depender da precisão humana para resposta do manipulador é que o sistema fica sujeito a erros, uma vez que depende dos reflexos do operador, tempo de resposta do sistema, entre outros fatores durante a operação. Para determinadas atividades seria possível implementar um algoritmo baseado na cinemática do manipulador que o posicione no ponto desejado percorrendo a melhor trajetória possível, automatizando o processo e o tornando livre de ação humana.

### b Problema

São necessárias algumas etapas para substituir o uso do *joystick* por um sistema de posicionamento automático de manipuladores robóticos. Inicialmente deve-se entender e modelar a cinemática do manipulador em conjunto com suas limitações mecânicas, aplicar conceitos matemáticos para fazer com que a trajetória seja a melhor possível, além de implementar o protocolo de comunicação entre computador e o braço mecânico.

### c Objetivos

O objetivo deste trabalho é desenvolver um sistema de controle automático de posição para um braço robótico de 5 graus de liberdade utilizado em operações submarinas. O sistema de controle deve permitir que o usuário escolha uma trajetória desejada para o manipulador através de um conjunto de pontos desejados, expressos em coordenadas 3D no espaço e referenciados em relação a base do manipulador. O software de controle irá determinar os ângulos necessários de cada junta do braço robótico para rastrear a trajetória desejada em tempo real, a partir da solução da cinemática inversa do manipulador.

Primeiramente, serão introduzidos conceitos básicos de cinemática e dos mecanismos que compõem o manipulador e, a seguir, será apresentada a modelagem dos parâmetros de Denavit-Hartenberg do sistema, necessários para a solução da cinemática direta e inversa do manipulador. Por fim, será apresentado o software de controle de posicionamento do manipulador para um ponto desejado no espaço.

Para os testes experimentais, será utilizado o manipulador ARM 5E Mini, fabricado pela empresa ECA Robotics [12] e adquirido pelo Centro de Pesquisa em Tecnologia de Inspeção (CPTI PUC-Rio) para auxílio

em operações de inspeção submarina. A validação do modelo e da cinemática do manipulador é realizada através de simulações em Matlab. A partir das simulações, é elaborado um algoritmo em Python para a solução da cinemática inversa e a comunicação serial com o manipulador, utilizando um protocolo específico fornecido pelo fabricante do braço robótico.

## 2 ROV e Técnicas de Inspeção Submarinas

Ao longo dos anos de exploração submarina de petróleo houve uma evolução considerável nas tecnologias de inspeção que garantem o funcionamento seguro do processo de transporte e extração de petróleo submarino. Exemplos de estruturas marítimas que necessitam de inspeção periódica são *risers* flexíveis, dutos submarinos, cascos de navio, correntes e amarras de plataformas de petróleo, entre outros. Em todos os exemplos mencionados, um dos principais métodos de inspeção é através do uso de ROVs, do inglês *Remotely Operated Underwater Vehicle* (ou Veículo Submarino Operado Remotamente), que realiza inspeções teleoperadas através de comandos realizados por um operador humano.

Essa inspeção pode ser feita através do uso de câmeras, sensores de ultrassom, raio-X ou outra forma de sensoriamento que esteja em contato com a superfície da estrutura, na busca por imperfeições e eventuais danos. Para assistir nas inspeções, reparos ou até na sua estabilização quando embaixo d'água, grande parte dos ROVs faz uso de um ou mais manipuladores podendo ser hidráulicos ou elétricos, como pode ser observado na Figura 1.



Figura 1: ROV Shepard Beachcomber H800 com dois manipuladores BM5E Micro [1].

No caso do Centro de Pesquisa em Tecnologia de Inspeção, o uso de manipulador tem como o objetivo auxiliar em operações de instalação e utilização do veículo autônomo AURI (do inglês *Autonomous Underwater Riser Inspection Tool*), responsável por fazer a inspeção visual externa de *risers* flexíveis. Esse tipo de tubulação, em particular, se encontra em condições de operação críticas, sujeitos à fadiga, abrasão, impacto, entre outros, necessitando de constante monitoramento [2].

O manipulador é a ferramenta mais capacitada para executar intervenções submarinas. Estes são compostos por uma sequência de corpos rígidos interconectados por juntas articuladas podendo ser rotativas ou prismáticas conectando os corpos ao efector final ("end effector") e neste podendo ser acopladas diferentes ferramentas como garras.

Na Figura 2, é possível observar um diagrama de todas as partes que compõem um manipulador e a sua relação com a interface do piloto.

Para que o manipulador possa operar em águas profundas e lidar com condições adversas no ambiente submarino, materiais específicos são utilizados na sua construção. Além disso, dependendo da tarefa que irão executar, os manipuladores submarinos devem atender a determinados requisitos, como por exemplo, o tamanho da área de atuação em que eles operam, capacidade de levantamento e torque de pulso, entre outros.

O tamanho de um manipulador é descrito pelo seu alcance, que está diretamente relacionado à amplitude de movimento das juntas, que determinam a área de trabalho (ou "workspace") do manipulador, ou seja, o conjunto de pontos que podem ser alcançados pelo seu efector final.

Geralmente, manipuladores submarinos são projetados com 3 a 6 graus de liberdade, sem considerar a mobilidade da garra. Os graus de liberdade de um manipulador são o número de coordenadas necessárias para especificar a sua configuração. Eles são projetados para atender diferentes necessidades, sendo 3 graus de liberdade o suficiente para atingir uma posição arbitrária no espaço e 6 para também incluir a orientação do efector final. Em toda e qualquer aplicação de manipuladores subaquáticos é necessário desenvolver a sua modelagem cinemática e algoritmos de controle que serão explicados mais à frente.



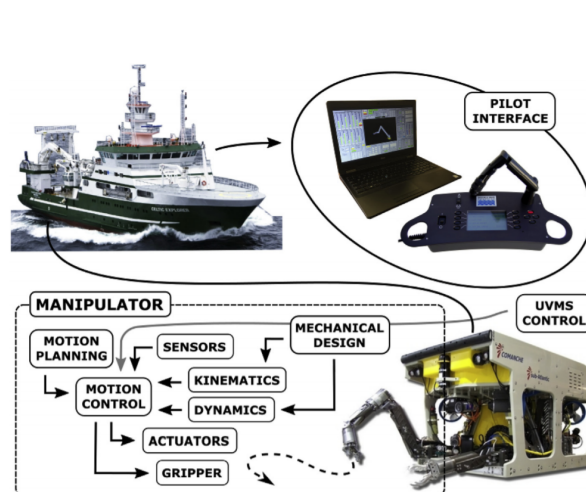


Figura 2: Fatores que afetam a performance do manipulador subaquático [2].

No geral, existem dois tipos de manipuladores, os hidráulicos e os elétricos. No caso desse projeto, foi utilizado um manipulador elétrico que, ao contrário dos hidráulicos, possui posicionamento mais acurado e controle de força e torque.

Outro contratempo nos manipuladores hidráulicos é o vazamento de uma pequena quantidade de fluido hidráulico, problema que é praticamente impossível de resolver, além do alto custo do sistema completo, que inclui acessórios como bomba hidráulica, reservatório, filtros, reguladores, válvulas, etc.

No caso dos manipuladores elétricos, para impedir a entrada de água salgada, os atuadores são preenchidos com óleo, o que também auxilia na lubrificação e no arrefecimento. Além disso, a vantagem principal de manipuladores com atuação elétrica é a capacidade de precisão do movimento e controle de torque/força. Em contrapartida, para aplicações industriais que necessitem maior velocidade de movimento e força/torque, manipuladores hidráulicos são mais adequados.

Modelos comerciais de manipuladores subaquáticos são comumente controlados remotamente por um operador localizado na superfície, que observa a operação através de câmeras e/ou sistemas de sonar enquanto toma decisões de movimentação. Essa dependência de um operador humano é o que motiva esse projeto, com o objetivo de automatizar a movimentação do manipulador com o intuito de reduzir eventuais fontes de erro na operação. A seguir, serão introduzidos conceitos mecânicos que fazem parte do processo de elaboração do modelo matemático que descreve a movimentação do manipulador.



## 3 Manipuladores Robóticos

Desde o século XV existem projetos mecânicos que são considerados o início da robótica [7]. Começando pelos autômatos, passando pelas leis da robótica de *Asimov*, até chegar no primeiro robô industrial (*Unimate*) e os atuais robôs residenciais, a robótica está cada vez mais presente no cotidiano do ser humano. Exemplos podem ser observados na Figura 3.



Figura 3: Na linha de cima, da esquerda para direita, um dos primeiros autômatos [3] e o livro "Eu, Robô", que introduz as leis [4]. Na linha de baixo, respectivamente, *Unimate*, o primeiro robô industrial [5] e um robô aspirador atual [6].

Atualmente, a robótica está presente em diversos lugares, desde indústrias até as casas comuns. Um exemplo muito comum empregado nas indústrias é o robô manipulador, com o simples objetivo de realizar o "trabalho forçado" de transportar um objeto de um ponto a outro remetendo a um braço mecânico. Contudo, o robô manipulador não se limita apenas a esse uso podendo ser utilizado com outras ferramentas acopladas ao seu efector final para realização de tarefas como soldagem, corte de materiais, entre outras.

A extremidade do manipulador é chamada de efector final, pois se encontra conectado geralmente à uma garra ou ferramenta para determinada tarefa que será realizada pelo manipulador. Os componentes básicos de um robô manipulador consistem nos elos mecânicos, atuadores, sensores, controladores e interface com o usuário.

A atuação dos manipuladores pode ser elétrica, hidráulica ou pneumática. O sensoreamento geralmente é feito através de *encoders* e potenciômetros, para medir a posição angular de cada junta. Além disso, pode ser medida a velocidade, a temperatura, bem como forças e momentos exercidos por cada junta.

### a Juntas, elos e graus de liberdade

Todo manipulador possui dois componentes básicos em sua estrutura: as juntas e os elos, em inglês, *joints* e *links* como pode ser observado na Figura 4.

Juntas são os componentes mecânicos responsáveis por unir os elos e se movimentar de determinada forma de acordo com o seu tipo. Os tipos mais comuns são as rotativas, que permitem um movimento rotacional linear entre dois elos, e as prismáticas, que permitem um movimento linear entre dois elos. Suas representações podem ser observadas na Figura 5. Os elos, por sua vez, são os corpos rígidos conectados por essas juntas e que complementam nas características de alcance e atuação do manipulador.

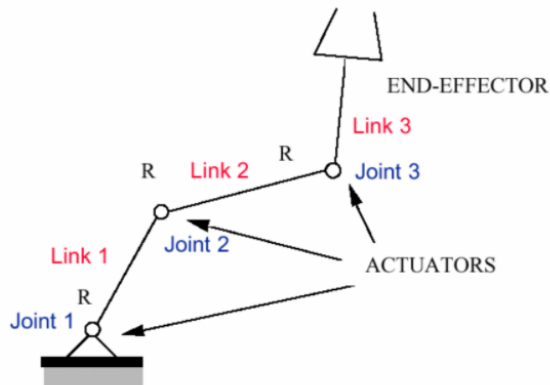


Figura 4: Manipulador planar de 3 juntas rotativas [7].

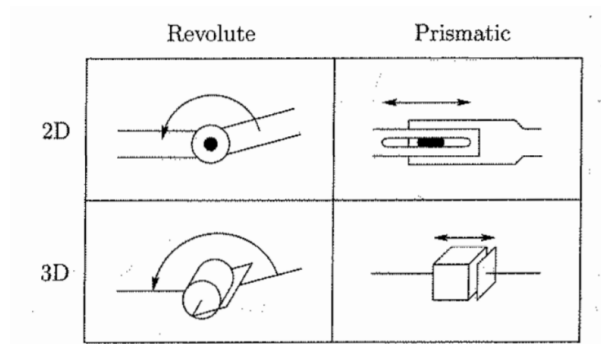


Figura 5: Representação das juntas rotativa e prismática em 2D e 3D [8].

Os graus de liberdade são o menor número de variáveis independentes para especificar a configuração de um corpo. Cada movimento realizado pelo objeto, seja de translação ou rotação, configura um grau de liberdade. No caso da Figura 4, é possível dizer que o manipulador possui 3 graus de liberdade devido ao número de juntas rotativas que permitem 3 rotações diferentes. Exemplos da representação dos graus de liberdade podem ser observados nas Figuras 6 e 7.

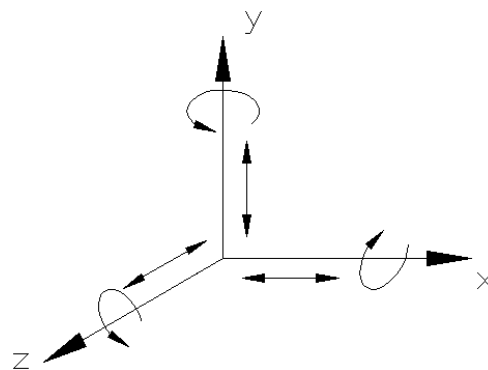


Figura 6: Representação dos graus de liberdade de um sistema tridimensional [9].

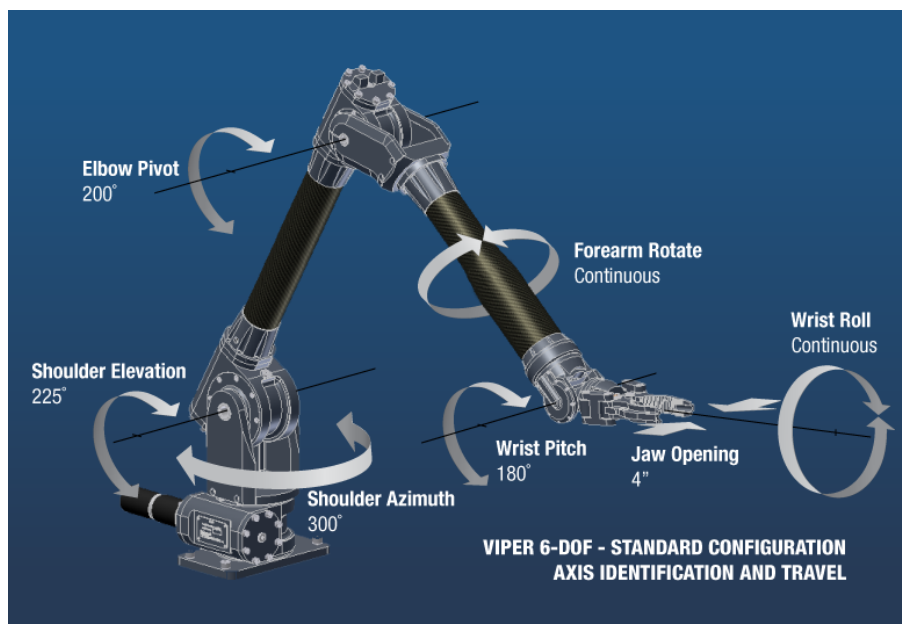


Figura 7: Diagrama com todos os seis graus de liberdade do manipulador Viper 6-DOF [10].

O diagrama da Figura 7 também pode ser chamado de cadeia cinemática. Trata-se da representação simbólica, como na Figura 8, do sistema de elos conectados pelas juntas do manipulador robótico em uma determinada configuração.

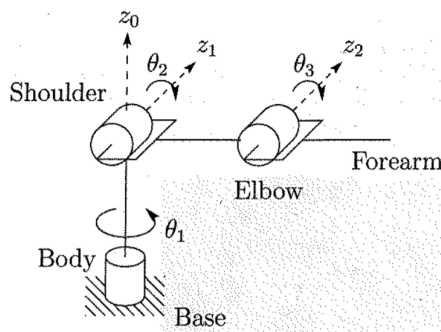


Figura 8: Exemplo de diagrama da cadeia cinemática [8].

Conhecendo os graus de liberdade, suas dimensões e parâmetros de juntas, em conjunto com conceitos algébricos apresentados a seguir, é possível elaborar o modelo cinemático do manipulador.

## b Análise cinemática

O problema da cinemática tem como propósito descrever o movimento do manipulador sem levar em consideração as forças e torques que causam o movimento se tornando portanto uma descrição apenas geométrica [8].

Este problema pode ser dividido em duas partes: a cinemática direta e a cinemática inversa. A primeira, traduz os ângulos desejados das juntas em uma posição no espaço para o efetor final e possui um algoritmo mais simples dependendo apenas de equações algébricas.

Já a cinemática inversa realiza o contrário, dado um ponto desejado no espaço, o algoritmo computa quais são os ângulos necessários de cada junta para que essa posição possa ser atingida. Esse caso já é mais complexo uma vez que os ângulos passam a ser as variáveis e o sistema se torna não-linear, sendo necessário métodos de otimização para resolvê-lo.

A metodologia utilizada para resolver e aplicar ambos os métodos no manipulador deste projeto será

abordada mais à frente. A seguir são apresentadas algumas definições necessárias para o entendimento da modelagem cinemática do manipulador.

### c Transformação homogênea

Transformações homogêneas combinam operações de rotação e translação em uma única matriz [13] e serão utilizadas para encontrar as equações cinemáticas do manipulador rígido mais à frente neste projeto. Para entender melhor o que são transformações homogêneas, é necessário entender o conceito de transformações isométricas.

Transformações isométricas (do inglês *Rigid Motions*) são qualquer tipo de movimentação de pontos em um plano de forma que a distância e posição relativa entre esses pontos se mantenha a mesma. Dois exemplos de transformação são a rotação e translação.

Considerando dois sistemas de coordenadas, 0 e 1 na Figura 9, a rotação de um sistema para outro pode ser obtida rotacionando o sistema 1 em  $\theta$  graus com relação ao sistema 0.

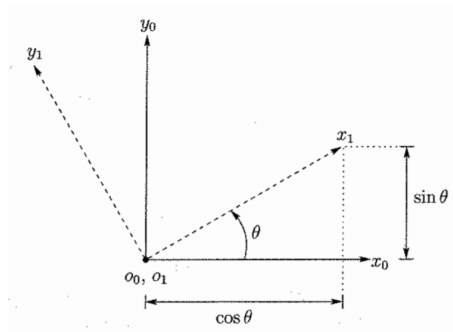


Figura 9: Sistema de coordenada  $o_1x_1y_1$  rotacionado em  $\theta$  graus em relação ao sistema  $o_0x_0y_0$  [8].

Uma das maneiras de se representar essa rotação, de acordo com [8], é através dos vetores de coordenadas dos eixos do sistema 1 em relação ao sistema 0.

$$R_1^0 = [x_1^0 \mid y_1^0] \quad (1)$$

Onde  $R_1^0$  é a matriz de rotação do sistema 1 para o sistema 0. Os vetores, por sua vez, são dados por:

$$x_1^0 = \begin{bmatrix} \cos \theta \\ \sin \theta \end{bmatrix}, \quad y_1^0 = \begin{bmatrix} -\sin \theta \\ \cos \theta \end{bmatrix} \quad (2)$$

Além disso, existe uma abordagem alternativa de representação dos termos da matriz que deixa mais trivial a adaptação para três dimensões. Esta alternativa se dá pela projeção dos eixos e origem do sistema 1 nos eixos e origem do sistema 0, através do produto escalar, entre os vetores unitários que representam esses sistemas. Portanto, adaptando na matriz (1).

$$R_1^0 = \begin{bmatrix} x_0 \cdot x_1 & y_0 \cdot x_1 \\ x_0 \cdot y_1 & y_0 \cdot y_1 \end{bmatrix} \quad (3)$$

No caso de um sistema de três dimensões, o eixo  $z$  passa a ser levado em consideração. Adaptando a equação (3).

$$R_1^0 = \begin{bmatrix} x_1 \cdot x_0 & y_1 \cdot x_0 & z_1 \cdot x_0 \\ x_1 \cdot y_0 & y_1 \cdot y_0 & z_1 \cdot y_0 \\ x_1 \cdot z_0 & y_1 \cdot z_0 & z_1 \cdot z_0 \end{bmatrix} \quad (4)$$

Usando como exemplo uma rotação do sistema 1 feita em torno do eixo  $z$  do sistema 0 em  $\theta$  graus, os produtos escalares são dados por:

$$x_1 \cdot x_0 = \cos \theta, \quad y_1 \cdot x_0 = -\sin \theta \quad (5)$$

$$x_1 \cdot y_0 = \sin \theta, \quad y_1 \cdot y_0 = \cos \theta \quad (6)$$

$$z_1 \cdot z_0 = 1 \quad (7)$$

Os outros produtos escalares são zero e portanto a matriz de rotação tridimensional entre dois sistemas se dá por:

$$R_1^0 = \begin{bmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (8)$$

Dado um ponto  $p$  em um sistema de coordenadas 1, é possível representá-lo no sistema de coordenadas 0 a partir da seguinte equação:

$$p_0 = R_1^0 p_1 + d_0 \quad (9)$$

Onde  $R_1^0$  é a matriz de rotação entre os sistemas de coordenadas 0 e 1,  $p_1$  são as coordenadas do ponto  $p$  no sistema 1, e  $d_0$  é a translação entre os sistemas 0 e 1.

Essa representação foi feita considerando apenas dois sistemas de coordenadas, à medida que outros sistemas são considerados na representação de  $p$ , esta começa a se tornar mais complexa.

No caso da translação, a representação pode ser feita na forma matricial através de uma matriz identidade 4x4 com um termo na última coluna indicando a quantidade de unidades transladadas em relação a cada eixo ( $x$ ,  $y$  ou  $z$ ):

$$T = \begin{bmatrix} 1 & 0 & 0 & x \\ 0 & 1 & 0 & y \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (10)$$

Para sistemas mais complexos, surgem as transformações homogêneas para simplificar a transformação entre sistemas de coordenadas, compostas pelas operações de rotação e translação em uma única matriz, definida por:

$$H = \begin{bmatrix} R & d \\ 0 & 1 \end{bmatrix} \quad (11)$$

Onde  $R$  é a matriz 3x3 de rotação e  $d$  a matriz coluna de translação.

Deve-se estabelecer múltiplos sistemas de coordenadas para representar as posições e orientações de cada corpo rígido no sistema. A partir disso, definem-se transformadas que permitem representar o movimento de um corpo em relação ao outro e assim, extrair as equações de cinemática direta, como será visto a seguir.

## d Cinemática direta

O problema da cinemática direta consiste em determinar a posição e a orientação do efector final a partir dos ângulos de cada junta do manipulador. A posição do efector final está diretamente ligada a movimentação das juntas bem como as suas limitações e as dimensões dos elos.

Assumindo que cada junta tem um grau de liberdade, a ação de cada junta pode ser descrita por único número real: o ângulo de rotação, no caso de uma junta rotativa, e um deslocamento, no caso de uma junta prismática.

Em um robô com  $n$  juntas, tem-se  $n + 1$  elos, já que cada junta conecta dois elos. Nomeiam-se os elos de 0 a  $n$  e as juntas de 1 a  $n$ , sendo 0 o sistema de coordenadas fixo na base do manipulador. A cada junta, associa-se uma variável  $q$  que pode representar um ângulo  $\theta$  no caso de uma junta de rotação ou um deslocamento  $d$  no caso de uma junta prismática. Um exemplo de robô com as descrições e numerações dos seus elos e juntas pode ser observado na Figura 10.

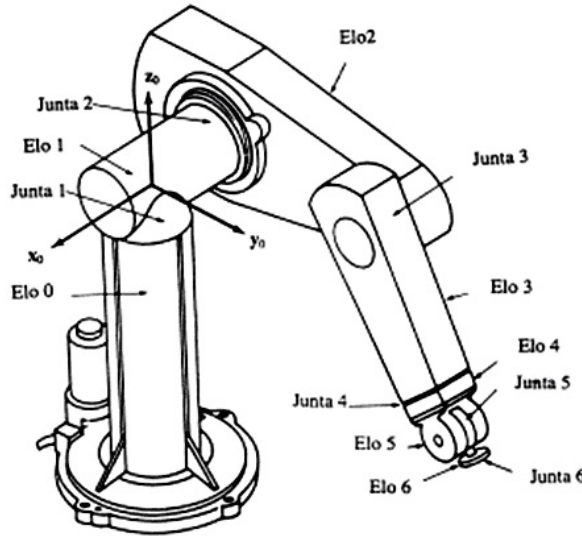


Figura 10: Robô manipulador PUMA com suas juntas e elos [11].

Para que seja realizada a análise cinemática, deve-se adotar um sistema de coordenadas para cada junta, sendo o primeiro o sistema de referência 0, e o último, o sistema de coordenadas solidário ao efector final do manipulador. Para cada sistema de coordenadas  $i$ , existe uma matriz  $A_i$  de transformação homogênea com a posição e orientação daquele sistema de coordenadas em relação ao seu anterior, a qual é expressa por:

$$A_i^{i-1} = \begin{bmatrix} R_i^{i-1} & o_i^{i-1} \\ 0 & 1 \end{bmatrix} \quad (12)$$

Sendo  $R_i^{i-1}$  a matriz de rotação 3x3 do sistema  $i$  para o sistema  $i-1$ , e  $o_i^{i-1}$  é o vetor com as coordenadas  $x$ ,  $y$  e  $z$  da origem do sistema  $i$  em relação ao sistema  $i-1$ .

A matriz final de transformação entre o eixo 0 e o eixo do efector final é dada pelo produto entre as matrizes  $A_i$  de cada junta obtendo  $H$ :

$$H = T_n^0 = A_1^0(q_1) \cdot A_2^1(q_1) \cdots A_n^{n-1}(q_n) \quad (13)$$

Obtêm-se então a matriz de transformação final da base do manipulador para o efector final:

$$H = \begin{bmatrix} R_n^0 & o_n^0 \\ 0 & 1 \end{bmatrix} \quad (14)$$

Onde  $R_n^0$  é a matriz 3x3 de rotação da última junta  $n$  em relação ao sistema de referência 0 e o vetor  $o_n^0$  representa a posição da última junta em relação à base.

Para simplificar a resolução do problema da cinemática direta, existem convenções como a notação Denavit-Hartenberg para representar cada junta, e será abordada a seguir.

## e Parâmetros Denavit-Hartenberg

A notação de Denavit-Hartenberg é usada para representar cada elo do manipulador através de dimensões referentes ao seu sistema de coordenadas em relação ao elo anterior. Essa convenção é útil em sistemas que possuem mais de dois elos em planos diferentes.

Essas dimensões são os parâmetros que compõem a notação:  $a$ ,  $\alpha$ ,  $d$  e  $\theta$  - comprimento do elo, torção do elo, *offset* do elo e ângulo de junta, respectivamente. Dentre esses, o único parâmetro variável é  $\theta$  no caso de uma junta de rotação ou  $d$  no caso de uma junta prismática.

Na Figura 11, os elos  $i$  e  $i-1$  estão conectados pela junta  $i$  e é possível perceber que em cada junta  $i$  existe um sistema de coordenadas  $(x_{i-1}, y_{i-1}, z_{i-1})$ . A relação entre os elos pode ser descrita através da

posição relativa e da orientação de um sistema para o outro, dadas pelos parâmetros de DH (Denavit-Hartenberg).

Além disso, existe uma linha que passa pelo ponto  $O_i$ , origem do sistema de coordenadas  $i$ , e pelo eixo  $z_{i-1}$ , denominada normal comum.

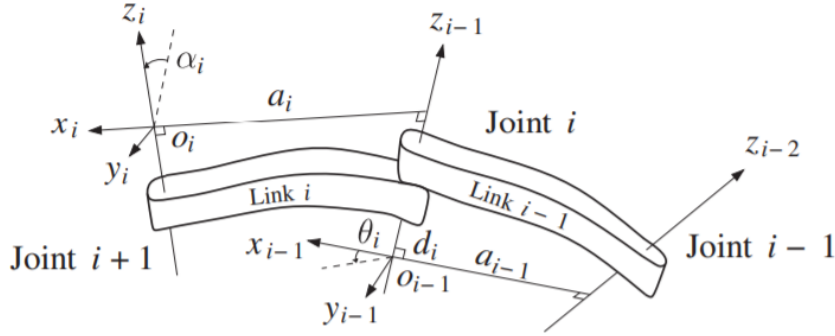


Figura 11: Definição dos parâmetros Denavit-Hartenberg em sistemas [8].

Os parâmetros de DH são encontrados através de relações entre os sistemas de coordenadas de duas juntas, seguindo um determinado procedimento. Para que esses parâmetros possam ser encontrados, é necessário atribuir um sistema de coordenadas para cada junta seguindo o passo-a-passo abaixo, de acordo com [7].

- 1) O eixo  $z_{i-1}$  é coincidente com o eixo de movimento da junta  $i$ ;
- 2) O eixo  $x_i$  é normal ao eixo  $z_{i-1}$  e ao eixo  $z_i$  ou a normal comum se  $z_i // z_{i-1}$ ;
- 3) O eixo  $y_i$  completa o sistema ortonormado direto.

Uma vez determinados os sistemas de coordenadas de cada junta, a definição de cada parâmetro DH pode ser obtida a partir das seguintes relações [14]:

$$\theta_i = \angle(x_{i-1}, x_i) |_{z_{i-1}} \quad (15)$$

$$\alpha_i = \angle(z_{i-1}, z_i) |_{x_i} \quad (16)$$

$$d_i = (O_{i-1}, z_{i-1} \cap x_i) |_{z_{i-1}} \quad (17)$$

$$a_i = (z_{i-1} \cap x_i, O_i) |_{x_i} \quad (18)$$

Com os parâmetros DH determinados, é necessário agora determinar a forma genérica da matriz (12) em função desses parâmetros. Para isso, é necessário fazer o produto entre quatro matrizes que compõem as transformações isométricas necessárias para transformar a posição e a orientação de um sistema em outro. Abreviando as funções trigonométricas cosseno e seno para  $c$  e  $s$  respectivamente:

Matriz de rotação básica de  $\theta$  graus em relação ao eixo  $z$ :

$$\begin{bmatrix} c_{\theta_i} & -s_{\theta_i} & 0 & 0 \\ s_{\theta_i} & c_{\theta_i} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (19)$$

Matriz de translação de  $d$  unidades de medida em relação ao eixo  $z$ :

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (20)$$



Matriz de translação de  $a$  unidades de medida em relação ao eixo  $x$ :

$$\begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (21)$$

Matriz de rotação básica de  $\alpha$  graus em relação ao eixo  $x$ :

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c_{\alpha_i} & -s_{\alpha_i} & 0 \\ 0 & s_{\alpha_i} & c_{\alpha_i} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (22)$$

Obtendo então a matriz  $A$  genérica:

$$A_i^{i-1} = \begin{bmatrix} c_{\theta_i} & -s_{\theta_i} c_{\alpha_i} & s_{\theta_i} c_{\alpha_i} & a_i c_{\theta_i} \\ s_{\theta_i} & c_{\theta_i} c_{\alpha_i} & -c_{\theta_i} s_{\alpha_i} & a_i s_{\theta_i} \\ 0 & s_{\alpha_i} & c_{\alpha_i} & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (23)$$

Utilizando um manipulador com quatro graus de liberdade, por exemplo, é necessária a conversão entre cinco sistemas de coordenadas, do sistema referência 0 até o efector final. Portanto, são necessárias quatro matrizes:  $A_1^0$ ,  $A_2^1$ ,  $A_3^2$  e  $A_4^3$  e o produto entre elas resulta na matriz de transformação da última junta, o efector final, para o sistema de referência na base do manipulador:

$$A_4^0 = \begin{bmatrix} a_{ex}^0 & g_{ex}^0 & r_{ex}^0 & x_e^0 \\ a_{ey}^0 & g_{ey}^0 & r_{ey}^0 & y_e^0 \\ a_{ez}^0 & g_{ez}^0 & r_{ez}^0 & z_e^0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (24)$$

Dessa matriz é possível extrair tanto a posição quanto a orientação do efector final. Desconsiderando a última linha da matriz, a matriz 3x3 com as três primeiras colunas são a orientação de arfagem, guinada e rolagem do efector final e os elementos da última coluna são as coordenadas  $x_e$ ,  $y_e$  e  $z_e$  do efector final, formando o conjunto de equações do problema da cinemática direta.

Conhecendo os parâmetros DH, é possível resolver esse problema, ou seja, descobrir a posição e orientação do efector final para determinadas posições das juntas, aplicando essas posições nas equações encontradas e, por fim, determinar a posição e orientação do efector final em relação aos ângulos de suas juntas.

Adotando um exemplo literal de um manipulador de dois graus de liberdade, a matriz com os parâmetros DH se daria por:

$$P_{DH} = \begin{bmatrix} \theta_1 & d_1 & \alpha_1 & a_1 \\ \theta_2 & d_2 & \alpha_2 & a_2 \end{bmatrix} \quad (25)$$

A transformação do sistema 0 de referência para o efector final se dará pelo produto entre duas matrizes com esses parâmetros aplicados. Observando apenas a última coluna, tem-se as equações de posição do efector final da cinemática direta.

$$x_e = a_1 \cos \theta_1 + a_2 \cos \theta_1 \cos \theta_2 + d_2 \sin \alpha_1 \sin \theta_1 - a_2 \cos \alpha_1 \sin \theta_1 \sin \theta_2 \quad (26)$$

$$y_e = a_1 \sin \theta_1 - d_2 \sin \alpha_1 \cos \theta_1 + a_2 \cos \theta_2 \sin \theta_1 + a_2 \cos \alpha_1 \cos \theta_1 \sin \theta_2 \quad (27)$$

$$z_e = d_1 + d_2 \cos \alpha_1 + a_2 \sin \alpha_1 \sin \theta_2 \quad (28)$$

Nesse trabalho serão utilizadas apenas as equações de posição para o problema da cinemática direta e inversa. No capítulo 5 será feita uma abordagem prática desses conceitos.



## f Cinemática inversa

Com o problema da cinemática direta resolvido, ou seja, encontrar a posição e orientação do efector final dados os ângulos de cada junta, agora deve-se resolver o problema da cinemática inversa, em que, dada uma posição  $(x_e, y_e, z_e)$  para o efector final, encontram-se os ângulos necessários nas juntas para que seja possível alcançá-la.

Uma das maneiras de se resolver esse problema é analiticamente, isolando as variáveis  $\theta$  nas equações da cinemática direta para resolver o sistema. Contudo, como pode ser observado nas equações (26), (27) e (28) acima, o isolamento das variáveis e a resolução do problema nem sempre ocorre de maneira trivial.

Nesse caso, é melhor adotar métodos numéricos iterativos, em que são sugeridos valores para as variáveis, avaliando a cada iteração se o método convergiu para o desejado, de acordo com tolerâncias impostas. Em contraste com as soluções analíticas, essa abordagem não é precisa, mas atinge boas aproximações e compensa pela complexidade de levantar a solução analítica.

Normalmente, esses métodos iterativos utilizam a matriz jacobiana do manipulador, composta pelas derivadas parciais das posições do efector final com respeito aos ângulos das juntas. Ela representa a relação infinitesimal entre os deslocamentos das juntas e a localização do efector final para uma determinada configuração do manipulador.

$$J = \begin{bmatrix} \frac{dx_e}{d\theta_1} & \dots & \frac{dx_e}{d\theta_n} \\ \frac{dy_e}{d\theta_1} & \dots & \frac{dy_e}{d\theta_n} \\ \frac{dz_e}{d\theta_1} & \dots & \frac{dz_e}{d\theta_n} \end{bmatrix} \quad (29)$$

Uma forma numérica de resolver o problema da cinemática inversa é formulá-lo como um problema de otimização em que as variáveis são os ângulos das juntas e o objetivo é minimizar a soma dos quadrados das diferenças entre o valor proposto e o valor real da posição do efector final. Essas diferenças também são chamadas de resíduos e esse método é conhecido como mínimos quadrados. Esse problema pode ser formulado da seguinte forma:

$$\begin{aligned} &\underset{q}{\text{minimize}} && ||f(q) - ee_d||^2 \\ &\text{subject to} && q_{min} \leq q \leq q_{max} \end{aligned}$$

Onde  $q$  é o conjunto de ângulos de cada junta do manipulador  $q = [\theta_1 \dots \theta_n]^T$ ;  $q_{min}$  e  $q_{max}$  são os limites superiores e inferiores de cada junta;  $f(q)$  são as equações da cinemática direta do manipulador, que retornam a posição do efector final  $(x_e, y_e, z_e)$  para um determinado conjunto de ângulos  $q$ ; e  $ee_d$  é a posição desejada para o efector final do manipulador.

Para esta formulação, a função objetivo do problema é minimizar a soma dos quadrados das diferenças dos resíduos, ou seja, a diferença entre as equações de cinemática direta e as coordenadas do ponto desejado. As equações a seguir mostram a função objetivo para este caso:

$$f(q) = \begin{bmatrix} x_e(q) - x_{e_d} \\ y_e(q) - y_{e_d} \\ z_e(q) - z_{e_d} \end{bmatrix} \quad (30)$$

$$F = ||f(q) - ee_d||^2 = (x_e(q) - x_{e_d})^2 + (y_e(q) - y_{e_d})^2 + (z_e(q) - z_{e_d})^2 \quad (31)$$

Sendo  $x_{e_d}$ ,  $y_{e_d}$  e  $z_{e_d}$  as coordenadas do efector final a serem inseridas como ponto desejado. As coordenadas  $x_e(q)$ ,  $y_e(q)$  e  $z_e(q)$  são obtidas a partir das equações cinemáticas do manipulador para um conjunto de ângulos  $q = [\theta_1 \dots \theta_n]^T$ .

A otimização das variáveis depende de um palpite inicial  $x_0$  para servir de ponto de partida para a otimização.

Por se tratar de um problema físico, o manipulador possui restrições mecânicas reais e que devem ser consideradas durante a otimização para que o resultado seja uma configuração possível. Essas restrições são os limites inferior e superior da otimização e impõem a condição de que a variável deve permanecer dentro deste intervalo.

Assim, a cada iteração tenta-se obter a melhor aproximação para o ponto desejado através da função objetiva, baseando-se nas restrições impostas.

Contudo, nem sempre a otimização é um sucesso e podem ocorrer as chamadas singularidades, problemas existentes em manipuladores robóticos. Elas ocorrem geralmente quando o manipulador parte de uma configuração em que se encontra totalmente esticado e no movimento seguinte acaba gerando esse problema. No modelo cinemático isso se traduz para quando o determinante da matriz jacobiana (29) é zero.

$$\det(J) = 0 \quad (32)$$

Em termos de comportamento do manipulador, isso pode resultar em grandes mudanças de ângulos das juntas, fazendo com que o manipulador se mova descontroladamente com movimentos bruscos. Geralmente, as singularidades [15] podem representar um problema na solução da cinemática inversa em métodos que utilizam a pseudo-inversa da matriz jacobiana, que não é o caso deste projeto. O método utilizado será abordado na Seção 5.

## 4 ARM 5E Mini

O manipulador utilizado neste projeto foi o modelo ARM 5E Mini da fabricante ECA Robotics, que pode ser observado na Figura 12, líder em modelos comerciais de manipuladores elétricos. Este manipulador foi adquirido pelo CPTI, como apresentado anteriormente, para servir de auxílio a ROV's e ao veículo autônomo de inspeção externa de *risers* flexíveis, AURI.

### a Especificações

Este manipulador é de 5 funções, sendo as três primeiras (*Slew*, *Shoulder* e *Elbow*) que, apesar de utilizarem atuadores lineares, estes pivotam o elo respectivo em torno da junta, tornando-se portanto juntas rotativas. A quarta função (*Jaw*), também é uma junta rotativa que influencia diretamente na orientação do efector final, e a última junta é a função de abrir e fechar a ferramenta acoplada no efector final.

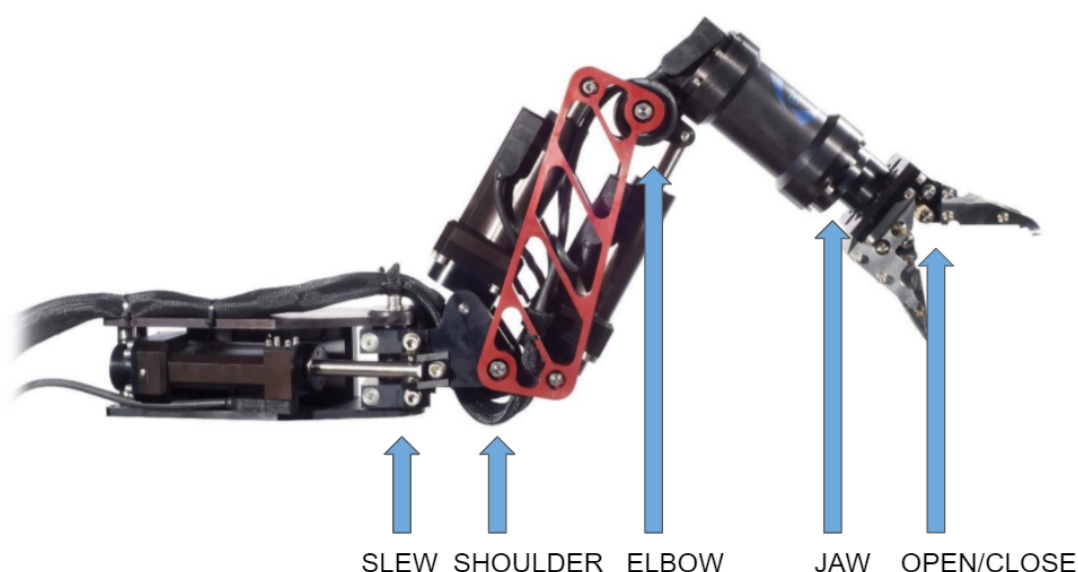


Figura 12: ARM 5E Mini com juntas nomeadas.

Limites de cada junta:

*SLEW* - 120°

*SHOULDER* - 90°

*ELBOW* - 145°

*JAW* - Rotação contínua (360°)

*JAW OPEN/CLOSE* - Abertura máxima de 140 mm ponta a ponta

Outras características:

- Alcance máximo: 850 mm
- Capacidade de elevação: 25 Kg no alcance máximo alimentado por 4 Ampères
- Peso: 15 Kg na água e 23 Kg no ar
- Alimentação: 24 - 36 Vdc
- Profundidade máxima: 300 metros quando preenchido com ar e 6000 metros quando preenchido com óleo
- Tipo de controle: *Joystick* USB em conjunto com *software* no computador, apresentado na Figura 13(a).
- Tipo de comunicação: Serial RS485 ou RS232

Para a alimentação elétrica do manipulador, foi utilizada uma fonte de bancada Minipa MPC-3005 DC a 24 volts com limitação de 4 A como a da Figura 13(b).



(a) Joystick USB



(b) Fonte de bancada Minipa MPC-3005 DC

Figura 13: Ferramenta de controle manual do manipulador e fonte de alimentação do sistema.

Para que fosse possível enviar/ler dados através da serial com algoritmos próprios enquanto o manipulador era controlado pelo *joystick* e o *software* do fabricante, foi necessário criar um "T" para os conectores, como pode ser visto na Figura 14.

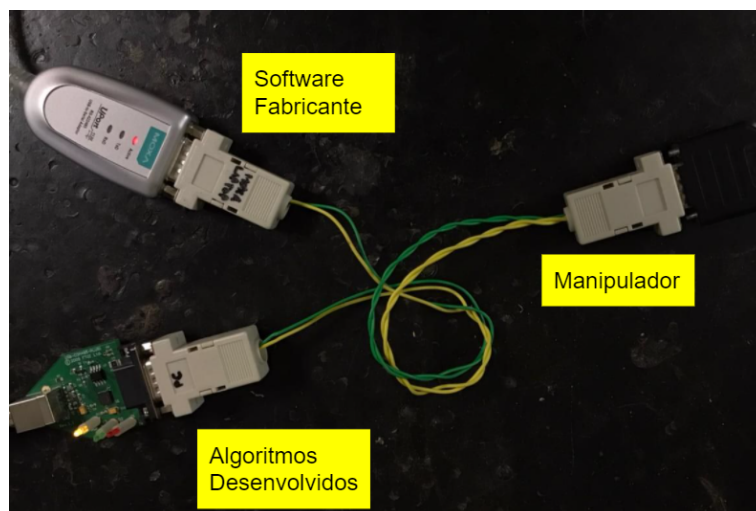


Figura 14: Conexão em "T".

Além disso, o manipulador acompanha um vaso de pressão, apresentado na Figura 15, contendo o módulo de comunicação serial, assim como os *drivers* de atuação de cada motor, essenciais para a operação do braço.

A ferramenta utilizada no efector final é a garra cortadora padrão, apresentada na Figura 16.



Figura 15: Vaso de pressão.

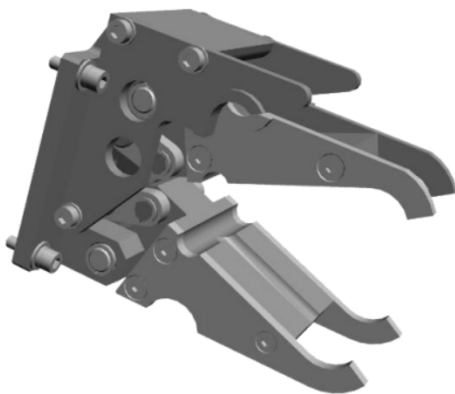


Figura 16: Garra cortadora - agarrador padrão.

## b Software fabricante

A empresa fabricante do manipulador, ECA Robotics, disponibiliza um software para controle do manipulador através do *joystick*. Nele é possível visualizar dados como temperatura do sistema, posição e velocidade de cada motor do manipulador. A Figura 17 mostra um *screenshot* da interface do software de controle, onde é possível visualizar todos os dados disponíveis.

Nem todos os valores exibidos tem uma unidade associada e, portanto, interpretar alguns desses dados para a elaboração dos algoritmos foi um dos desafios desse projeto.



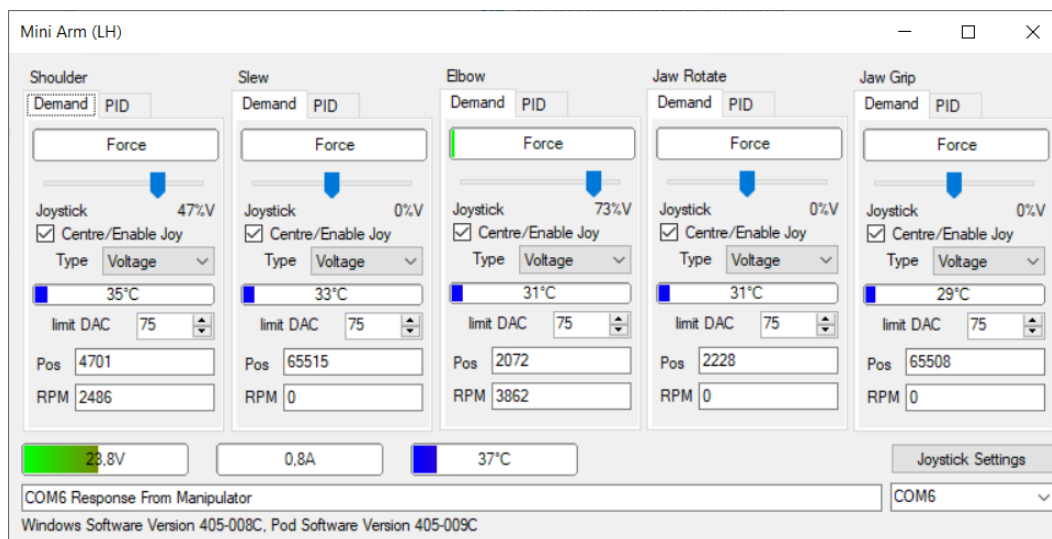


Figura 17: Captura de tela do software de controle provido pela ECA [12].

### c Protocolo de comunicação

O ARM 5E Mini possui o seu próprio protocolo de comunicação composto por 51 bytes para troca de informações com o computador. Quando o computador está enviando dados para o manipulador, geralmente contém as demandas de cada motor, ou seja, o tipo de movimento a ser realizado: se será baseado em tensão, posição, velocidade, etc. No caso do manipulador enviando para o computador, será uma mensagem de *feedback* com os valores lidos pelos sensores em relação à temperaturas, consumo de corrente e posição. O protocolo está dividido em três grandes categorias: *Master Values*, Motor e Bytes Gerais.

A *Master Values* são os valores lidos pelos sensores em relação ao manipulador como um todo, e são eles temperatura, tensão e corrente do mesmo. Já o motor é dividido em 9 bytes de forma que, na relação *PC* → *Manipulador*, contém o tipo da demanda, a demanda, o limite de velocidade e o limite do consumo de corrente, e na relação *Manipulador* → *PC* contém os valores dos sensores naquele motor apresentando posição, velocidade, corrente e temperatura. Por último, os bytes gerais são os que garantem a integridade da mensagem enviada seguindo o protocolo: byte de início de mensagem, byte de final de mensagem e *checksum* para checar a soma final dos valores da mensagem.

Ter conhecimento do protocolo utilizado foi crucial para a elaboração do algoritmo de controle de trajetória do manipulador por cinemática direta/inversa uma vez que é desejado inserir o valor do ângulo para cada junta. O protocolo fornecido no manual do fabricante pode ser encontrado no apêndice a.

#### 1 Algoritmo para envio dos dados

Para o envio de dados, decidiu-se trabalhar principalmente em cima da demanda de posição dado que este trabalho tem o objetivo de posicionar o manipulador em um determinado ponto através da resolução de sua cinemática inversa. O desafio foi saber interpretar como o protocolo lida com o valor de posição e como esse dado corresponde ao ângulo da junta.

No manual do fabricante as duas únicas informações disponíveis são que o valor da posição possui 16 bits de tamanho (portanto podendo variar de 0 a 65535) e que o valor da posição exibido no software do fabricante corresponde à contagem de pulso da rotação do motor, detectada pelos sensores de efeito Hall em cada motor.

Como não havia mais nenhum dado que pudesse levar a uma relação entre a contagem de pulso e o ângulo da junta, foi feito um teste empírico para descobrir qual o valor máximo de pulsos em uma junta com abertura máxima observando o valor no software do fabricante. O valor lido foi parecido para todas as juntas em sua abertura máxima, em torno de 20320. Com esse valor é possível fazer um mapeamento em relação ao ângulo de determinada junta, como pode ser observado na equação (33).

$$posicao = \frac{20320 \cdot angulo}{angulo_{max}} \quad (33)$$

## 2 Algoritmo para leitura dos dados

No caso do algoritmo de leitura dos dados do manipulador no computador, houveram três percalços a serem superados durante sua elaboração. Eles foram encontrados durante o teste com um algoritmo apenas de leitura dos sensores do manipulador que mostrava a posição e a velocidade variando sem apresentar nenhum padrão, a corrente variando de forma desprezível durante o movimento do manipulador e a temperatura só variando durante o movimento.

O primeiro problema foi resolvido com uma modificação no protocolo. A ordenação dos bytes durante a descompactação da mensagem do motor passou a ser do tipo *Little-Endian*, em que o byte menos significativo é armazenado no menor endereço. Isso fez com que tanto a posição quanto a velocidade passassem a variar com conforme o movimento do manipulador.

Para os outros dois problemas foi necessária uma análise mais minuciosa da mensagem que estava sendo recebida pelo computador. Foram utilizadas duas situações em cima do mesmo motor (*Shoulder*), parado e em movimento:

Situação: manipulador parado

Motor data: 01 01 00 00 00 40 0f 00 00

01 – prefixo  
01 00 – posição  
00 00 – velocidade  
40 0f – corrente  
00 – temperatura  
00 – *padding*

É possível observar na Figura 18 que o byte de temperatura do motor 1 encontra-se zerado quando no *software* isto não é verdade (o mesmo está a 29°).

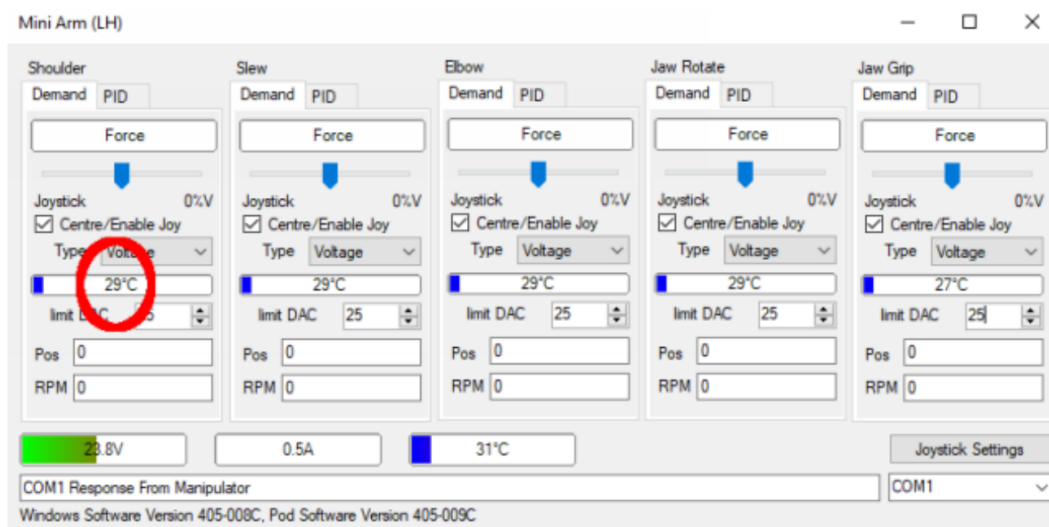


Figura 18: *Software* Fabricante - Motor 1 (*Shoulder*) parado [12].

Situação: motor 1 (*Shoulder*) em movimento

Motor data: 01 14 0c 22 0f 40 11 14 00

01 – prefixo  
14 0c – posição  
22 0f – velocidade  
40 11 – corrente  
14 – temperatura  
00 – *padding*

Aplicando a fórmula de temperatura, encontrada no manual do manipulador [12], no byte correspondente (0x14 em decimal 20), obtém-se:

$$T = ((8\text{BitTemperature} / 255) * 3.3) / 0.0066101694915254237288135593220339$$

$$T = 39^\circ$$

Pelo *software* do manipulador é possível perceber que o resultado também não corresponde. Além disso, o comportamento da temperatura ser zero quando o manipulador se encontra parado e variar quando o mesmo está se movimentando, é o comportamento da corrente e, possivelmente, a corrente é a temperatura já que se encontra sempre em torno do mesmo valor.

A partir de uma análise mais minuciosa da variação de cada byte, foi possível observar que a corrente (composta por 2 bytes sendo então o tipo de data *short*), estava com a informação dividida em duas e no meio o byte em questão correspondia na verdade à temperatura (composta apenas por um byte).

Observando a Figura 19, é possível comprovar que os bytes menos significativos da corrente e da temperatura estão invertidos. Já o byte mais significativo da corrente pode ser desprezado, dado que não se altera, sendo então a segunda modificação no protocolo.

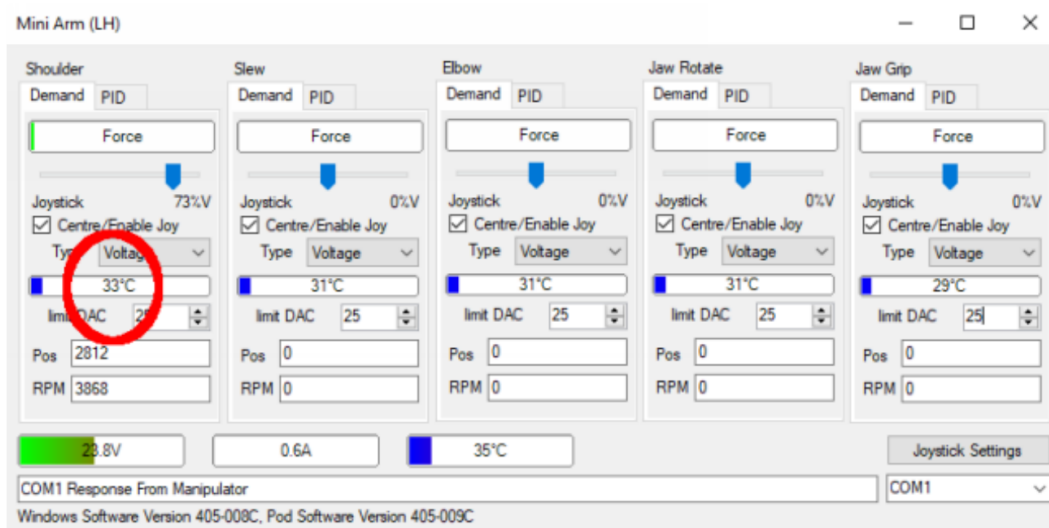


Figura 19: Software Fabricante - Motor 1 (*Shoulder*) movimentando [12].

Motor 1 – em movimento – *Current*: 40 11

Utilizando o byte menos significativo da corrente:

$$T = ((8\text{BitTemperature} / 255) * 3.3) / 0.0066101694915254237288135593220339$$

$$T = 33^\circ$$

Segunda inspeção:

Primeira mensagem no formato hexadecimal com manipulador parado, dados do motor 1:

01 00 00 00 00 00 14 00 00

Segunda mensagem em hexadecimal com manipulador parado, dados do motor 1:

01 00 00 00 00 40 14 00 00



Como pode ser observado acima, existe uma diferença de 16.384 (em decimal) entre os valores do AD da corrente. Assim, foi decidido alterar o formato de desempacotamento da mensagem do manipulador na parte dos motores:

Anteriormente: '1B3H1Bx'

- 1 Byte de *Start*
- 3 Shorts (Posição, Velocidade e Corrente)
- 1 Byte de Temperatura
- 1 byte de *padding*

Formato atual: '1B2H3Bx'

- 1 Byte de *Start*
- 2 Shorts (Posição e Velocidade)
- 1 Byte desprezado (Mais significativo da corrente, desprezado)
- 1 Byte de Temperatura
- 1 Byte de corrente (Menos significativo da corrente)
- 1 byte de *padding*

Com isso, todos os problemas de visualização dos dados foram resolvidos e o algoritmo de leitura pronto para ser utilizado.

## d Estação de trabalho

Para os diferentes testes desse projeto, o manipulador foi aparafusado a caibros de madeira e estes presos com grampos C a um carrinho de ferramentas ao lado da mesa do operador, como mostra a Figura 20.



Figura 20: Área de trabalho com ponto de fixação do manipulador e computadores utilizados.

## 5 Simulações

Seguindo agora para a parte prática deste projeto, os conceitos vistos anteriormente serão aplicados no manipulador ARM 5E Mini com o objetivo final de realizar uma simples trajetória pré-definida. A seguir, será feito o levantamento de seus parâmetros DH, sua área de trabalho, suas equações de cinemática direta, bem como a resolução da cinemática inversa.

### a Parâmetros Denavit-Hartenberg do ARM 5E Mini

Para que as simulações e testes cinemáticos possam ser realizados, é necessário encontrar os parâmetros de DH do manipulador. Para isso, basta seguir o procedimento visto na Seção 4e em relação a atribuição dos eixos de referência das juntas e levantamento dos parâmetros correspondentes. A Figura 21 mostra os sistemas de coordenadas de cada junta e ilustra os parâmetros DH para o manipulador ARM 5E Mini, que são apresentados na Figura 22.

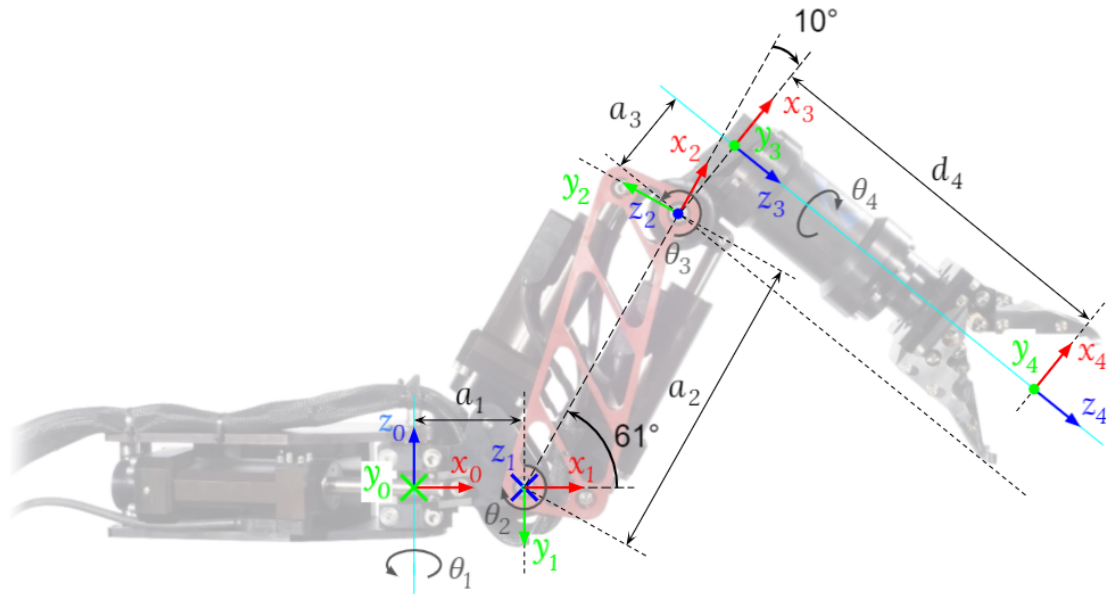


Figura 21: Representação do manipulador com os sistemas de coordenadas já arbitrados em cada junta.

Elo	$\theta$	$\alpha$	$a$	$d$
1	$0^\circ$	$-90^\circ$	$a_1$	0
2	$-61^\circ$	$180^\circ$	$a_2$	0
3	$-10^\circ$	$90^\circ$	$a_3$	0
4	$0^\circ$	$0^\circ$	0	$d_4$

Figura 22: Parâmetros Denavit-Hartenberg teóricos.

Para encontrar o equacionamento da posição do efetor final, coloca-se os parâmetros  $\theta_i$  de cada junta como variáveis, pois estes serão os parâmetros a serem determinados a partir da resolução da cinemática direta. A partir dos parâmetros DH obtidos, as matrizes de transformação de cada junta  $i$  em relação a junta anterior  $i - 1$  para o manipulador ARM 5E Mini são:

$$A_1^0 = \begin{bmatrix} c_{\theta_1} & 0 & s_{\theta_1} & a_1 c_{\theta_1} \\ s_{\theta_1} & 0 & -c_{\theta_1} & a_1 s_{\theta_1} \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (34)$$

$$A_2^1 = \begin{bmatrix} c_{\theta_2} & -s_{\theta_1} & 0 & a_2 c_{\theta_2} \\ s_{\theta_2} & c_{\theta_2} & 0 & a_2 s_{\theta_2} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (35)$$

$$A_3^2 = \begin{bmatrix} c_{\theta_3} & 0 & -s_{\theta_3} & a_3 c_{\theta_3} \\ s_{\theta_3} & 0 & c_{\theta_3} & a_3 s_{\theta_3} \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (36)$$

$$A_4^3 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_4 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (37)$$

A partir da multiplicação dessas matrizes, obtém a matriz de transformação final do efector final até o sistema de referência na base do manipulador:

$$H = A_1^0 \cdot A_2^1 \cdot A_3^2 \cdot A_4^3 = A_0^4 = \begin{bmatrix} c_{\theta_2+\theta_3} c_{\theta_1} & -s_{\theta_1} & -s_{\theta_2+\theta_3} c_{\theta_1} & c_{\theta_1} (a_1 + a_3 c_{\theta_2+\theta_3} - d_4 s_{\theta_2+\theta_3} + a_2 c_{\theta_2}) \\ c_{\theta_2+\theta_3} s_{\theta_1} & c_{\theta_1} & -s_{\theta_2+\theta_3} s_{\theta_1} & s_{\theta_1} (a_1 + a_3 c_{\theta_2+\theta_3} - d_4 s_{\theta_2+\theta_3} + a_2 c_{\theta_2}) \\ s_{\theta_2+\theta_3} & 0 & c_{\theta_2+\theta_3} & d_4 c_{\theta_2+\theta_3} + a_3 s_{\theta_2+\theta_3} + a_2 s_{\theta_2} \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (38)$$

Inicialmente, como o manual do fabricante não possuía o projeto CAD completo do manipulador e nem os parâmetros DH, buscou-se trabalhos publicados que haviam utilizado o mesmo modelo de manipulador para reaproveitar esses parâmetros. Contudo, os únicos trabalhos encontrados utilizavam um modelo diferente, o ARM 5E, que apesar de ser diferente, serviu de base devido à semelhança do posicionamento das juntas dos dois manipuladores [16, 17].

No caso desse projeto, o manipulador se encontra fixado em uma bancada e sua posição inicial é totalmente recolhido, como é possível observar na Figura 20. Essa configuração faz com que o manipulador só possa se mover em um sentido, do contrário ele já se encontra no seu limite mecânico.

Dessa forma, os parâmetros  $\alpha$  e a escolha dos eixos nas juntas tiveram de ser atribuídos levando em consideração essa configuração inicial e o limite do movimentação do manipulador. No caso dos parâmetros  $d$  e  $a$ , como não temos disponível o CAD para obter as medidas exatas, foram utilizados instrumentos de medição como paquímetro e escala para obter a medida empiricamente no próprio manipulador. A partir desse procedimento, é possível observar na Figura 23 as medidas obtidas do manipulador ARM 5E Mini.

Link	Parâmetros Medidos			
	Theta (graus)	d (mm)	a (mm)	alfa (graus)
1	0	0	115.8	-90
2	-61	0	315	180
3	-10	0	92	90
4	0	407	0	0

Figura 23: Parâmetros DH obtidos empiricamente.

A partir da matriz apresentada em (38), tem-se as equações da cinemática direta:

$$x_e = \cos(\theta_1) \cdot (a_1 + a_3 \cdot \cos(\theta_2 - \theta_3) - d_4 \cdot \sin(\theta_2 - \theta_3) + a_2 \cdot \cos(\theta_2)) \quad (39)$$

$$y_e = \sin(\theta_1) \cdot (a_1 + a_3 \cdot \cos(\theta_2 - \theta_3) - d_4 \cdot \sin(\theta_2 - \theta_3) + a_2 \cdot \cos(\theta_2)) \quad (40)$$

$$z_e = -a_3 \cdot \sin(\theta_2 - \theta_3) - d_4 \cdot \cos(\theta_2 - \theta_3) - a_2 \cdot \sin(\theta_2) \quad (41)$$

## b Workspace do manipulador

Como mencionado anteriormente, manipuladores possuem inúmeras juntas e elos de diversos tipos e tamanhos, proporcionando o alcance do efector final em diferentes pontos. Com isso, a fim de conhecer todo o potencial de alcance do manipulador, é necessário conhecer a sua área de trabalho (*workspace*), ou seja, todos os pontos que o manipulador é capaz de atingir.

Para encontrar o *workspace* do manipulador serão utilizadas as equações de cinemática direta de posição  $x_e$ ,  $y_e$  e  $z_e$  do efector final, encontradas na matriz de transformação final (38). Com elas, é possível encontrar toda a área de atuação do manipulador utilizando uma faixa de valores para os ângulos das juntas.

Utilizando o algoritmo *DHRobotliteral*, é possível gerar as equações da cinemática direta com os parâmetros  $\theta_i$  como variáveis simbólicas e utilizar o comando *linspace* do Matlab para substituí-los por uma faixa de valores linearmente distribuídos representando a excursão total de cada junta, entre o seu limite inferior e superior.

O número de valores adotado na faixa foi de 50, pois foi o suficiente para a visualização do *workspace* sem comprometer o processamento do computador onde foi realizada a simulação. A partir da faixa de ângulos para cada junta, é possível simular todas as possíveis configurações do manipulador, resolvendo as equações cinemáticas para todas as configurações possíveis das juntas, e gerando todas as possibilidades de posicionamento do efector final, ou seja, o seu *workspace*. As equações cinemáticas do manipulador estão apresentadas nas equações (39), (40) e (41).

Além disso, é necessário levar em consideração o *offset* inicial do manipulador, ou seja, a posição inicial das juntas, que no caso desse projeto, é a junta *slew* posicionada totalmente para a direita e as juntas *shoulder* e *elbow*, totalmente recolhidas dada a fixação do manipulador como visto anteriormente na Figura 20.

$$\theta_1 = \theta_1 - \pi/2 \quad (42)$$

$$\theta_2 = \theta_2 - \pi/2 \quad (43)$$

$$\theta_3 = \theta_3 - 75 \cdot \pi/2 \quad (44)$$

O algoritmo usado para gerar a área de trabalho pode ser encontrado em [18].

A Figura 24 mostra o *plot* de todos os pontos resultantes das faixas de ângulos das juntas, obtendo a área de atuação do manipulador. É possível observar que a área de trabalho do manipulador tem uma aparência esférica dado que todas as suas juntas são rotativas. Na Figura 25 é possível visualizar mais facilmente o alcance máximo do manipulador no eixo  $x$ , que condiz aproximadamente com o do manual do fabricante (850 mm).

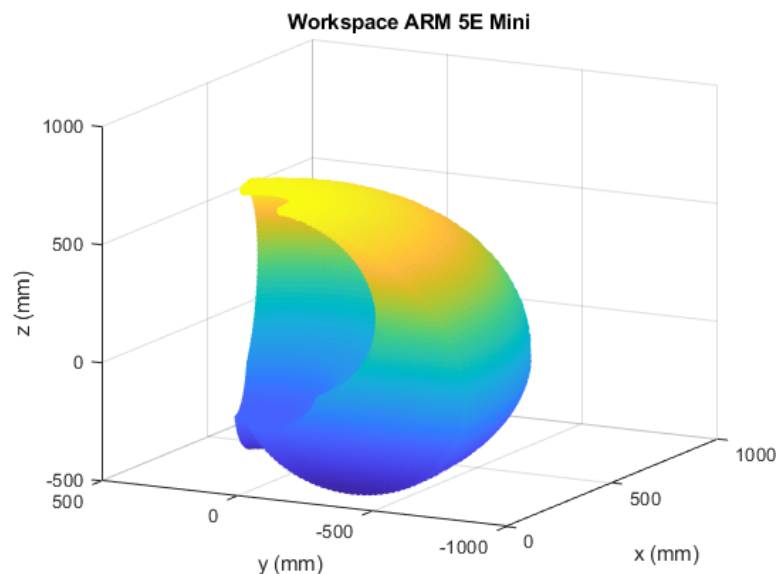


Figura 24: *Workspace* manipulador.

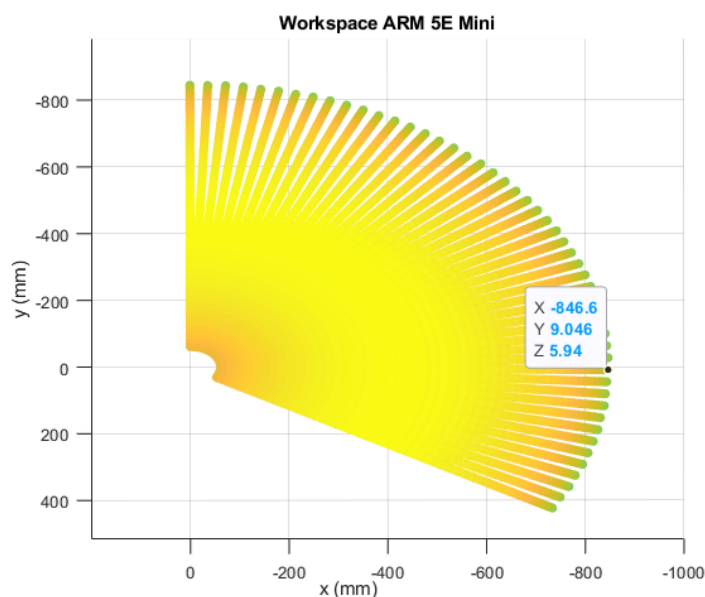


Figura 25: Vista superior *workspace* manipulador.

### c Cinemática direta

Para a simulação do posicionamento do manipulador no Matlab, foi utilizada a *toolbox* de robótica do Peter Corke [19] que, através dos parâmetros declarados de Denavit-Hartenberg que estão na Figura 23, cria os links do manipulador e monta uma simulação interativa, com o uso do comando *teach*. Isso permite que o usuário varie os ângulos das juntas, movimentando o efector final, demonstrando os conceitos de cinemática direta. Na Figura 26, é possível observar essa interface de controle em que há as coordenadas do efector final, sua orientação e um *slider* para variar a posição de cada junta, permitindo visualizar em tempo real a movimentação do manipulador. O código utilizado pode ser encontrado em [18].

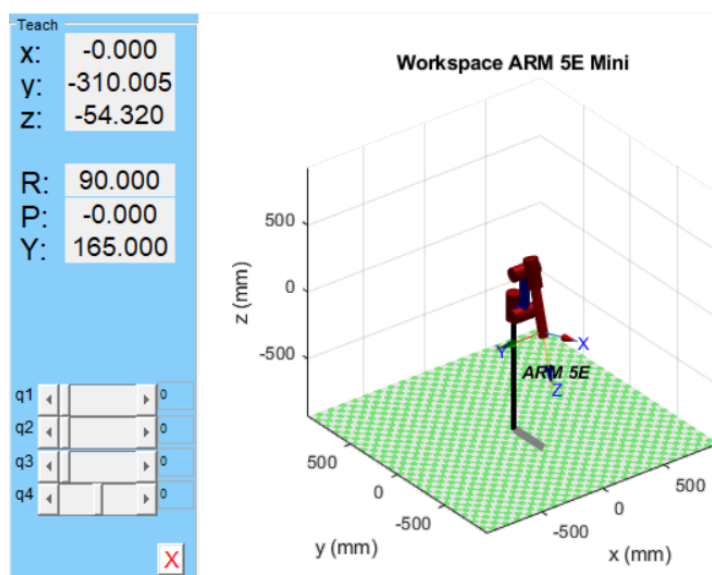


Figura 26: Manipulador na posição inicial com a função *teach* do *toolbox* de Peter Corke.

É possível validar a elaboração do algoritmo do *workspace* da Figura 24 plotando-o juntamente com o desenho simulado do manipulador, como mostra a Figura 27. A Figura 28 mostra exemplos de diversas configurações do manipulador sobrepostas com o *workspace* teórico. Observe que a posição do efector final em todos os casos coincide com o *workspace*, como esperado.



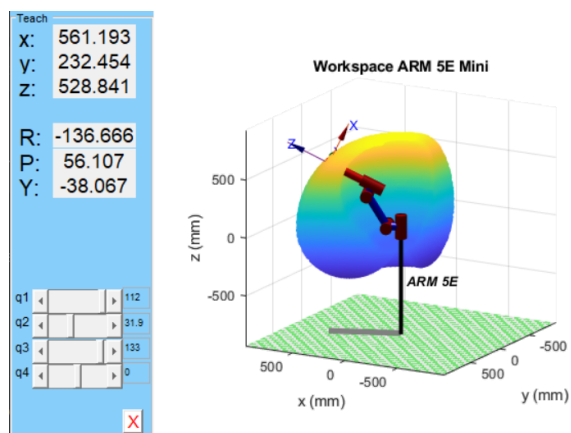


Figura 27: *Workspace* plotado sobre a interface *teach*.

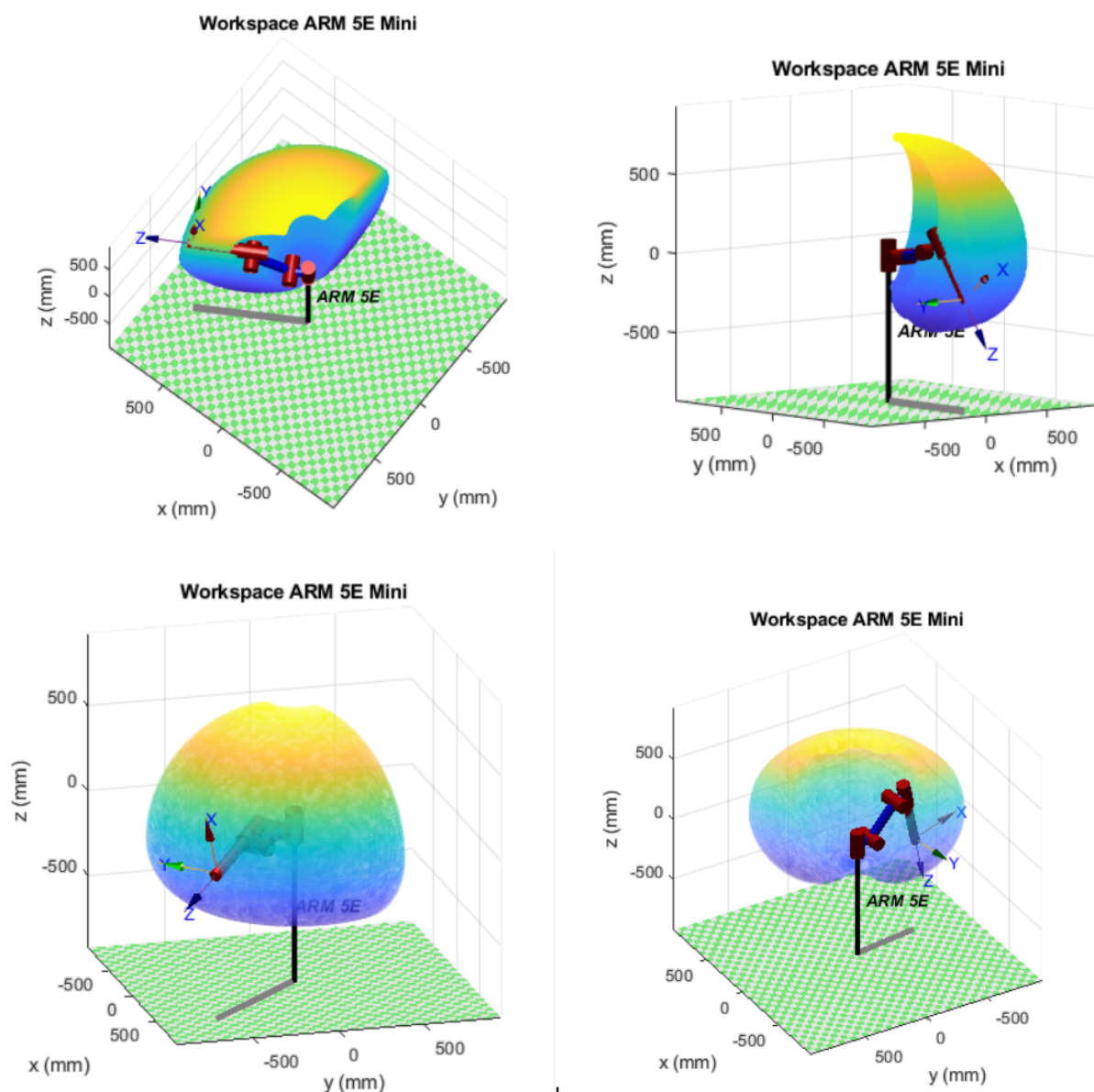


Figura 28: Exemplos da interface *teach* com o *workspace*.

#### d Cinemática inversa

Agora que as equações da cinemática direta do manipulador foram validadas através de simulações no Matlab, dá-se início aos testes da cinemática inversa, sendo este um problema mais complexo, já que as incógnitas passam a ser os ângulos das juntas em (39), (40) e (41) e faz-se necessário o uso de métodos numéricos iterativos para a resolução do problema.

O método escolhido para a solução da cinemática inversa do manipulador foi o dos mínimos quadrados, onde a função objetivo do problema se dá pela soma dos quadrados das diferenças entre o ponto desejado e ponto resultante, como visto na Seção 3f.

Para a solução de um problema de mínimos quadrados não-linear com restrições nas variáveis, é recomendado o uso da função `lsqnonlin` do Matlab [20]. Ela recebe como parâmetros a função a ser minimizada, um palpite inicial e os limites superior e inferior das variáveis, e retorna o valor otimizado para as variáveis e o erro da otimização. Essa função possui como algoritmo *default* o *Trust Region Reflective*. O Matlab recomenda este algoritmo [21] quando existe apenas uma restrição de variáveis, que no caso deste projeto, são os limites das juntas. Contudo, ele só pode ser utilizado se o sistema de equações não-lineares não for subdeterminado, ou seja, houverem menos equações do que variáveis no sistema, condição cumprida pelo sistema aqui utilizado.

A função `lsqnonlin`, do Matlab, resolve um problema da forma:

$$\min_x ||f(x)||_2^2 = (f_1(x))^2 + f_2(x)^2 + \dots + f_n(x)^2$$

onde o usuário deve fornecer como entrada o vetor  $f(x) = [f_1(x) \ f_2(x) \ \dots \ f_n(x)]^T$ , e opcionalmente os limites superiores e inferiores das variáveis  $x$ .

Para a cinemática inversa do manipulador, o vetor  $f(x)$  é composto pelas equações (39), (40) e (41), subtraídas do ponto desejado para o efector final, da forma:

$$f(\theta_1, \theta_2, \theta_3) = \begin{bmatrix} \cos(\theta_1) \cdot (a_1 + a_3 \cdot \cos(\theta_2 - \theta_3)) - d_4 \cdot \sin(\theta_2 - \theta_3) + a_2 \cdot \cos(\theta_2) - x_e \\ \sin(\theta_1) \cdot (a_1 + a_3 \cdot \cos(\theta_2 - \theta_3)) - d_4 \cdot \sin(\theta_2 - \theta_3) + a_2 \cdot \cos(\theta_2) - y_e \\ -a_3 \cdot \sin(\theta_2 - \theta_3) - d_4 \cdot \cos(\theta_2 - \theta_3) - a_2 \cdot \sin(\theta_2) - z_e \end{bmatrix} \quad (45)$$

Além disso, devem ser levado em consideração o *offset* da posição do manipulador e os limites mecânicos das juntas. É válido ressaltar que este problema de otimização é composto apenas por três variáveis ( $\theta_1, \theta_2, \theta_3$ ) porque a orientação do efector final não está sendo levada em consideração, apenas a posição.

O *offset* se dá pela posição inicial em que o manipulador se encontra e deve ser subtraído das variáveis antes de aplicá-las na função objetivo, da seguinte forma:

$$\theta_1 = \theta_1 - \pi/2 \quad (46)$$

$$\theta_2 = \theta_2 - \pi/2 \quad (47)$$

$$\theta_3 = \theta_3 - 75 \cdot \pi/2 \quad (48)$$

O limite mecânico das juntas representa o limite inferior e superior da otimização e é dado pelos vetores  $l_i$  e  $l_s$  respectivamente:

$$l_i = [0 \ 0 \ 0] \quad (49)$$

$$l_s = [120 \ 90 \ 45] \quad (50)$$

Para validação do método, foi feito um teste comparativo escolhendo os ângulos das juntas *slew*, *shoulder* e *elbow* como  $30^\circ$  na interface do Peter Corke, como pode ser observado na Figura 26, e fornecendo a posição do efector final nesta configuração como entrada para o algoritmo da cinemática inversa. A Figura 29 mostra o resultado da cinemática direta e a Figura 30 mostra a solução da otimização para o ponto desejado (233.752, -404.871, -96.522). Convertendo o resultado da otimização para graus, obtém-se exatamente  $\theta_1 = 30.0000^\circ, \theta_2 = 29.9999^\circ, \theta_3 = 30.0000^\circ$ . Note que o resultado encontrado condiz com os ângulos esperados com um erro muito pequeno, que pode ser considerado desprezível.

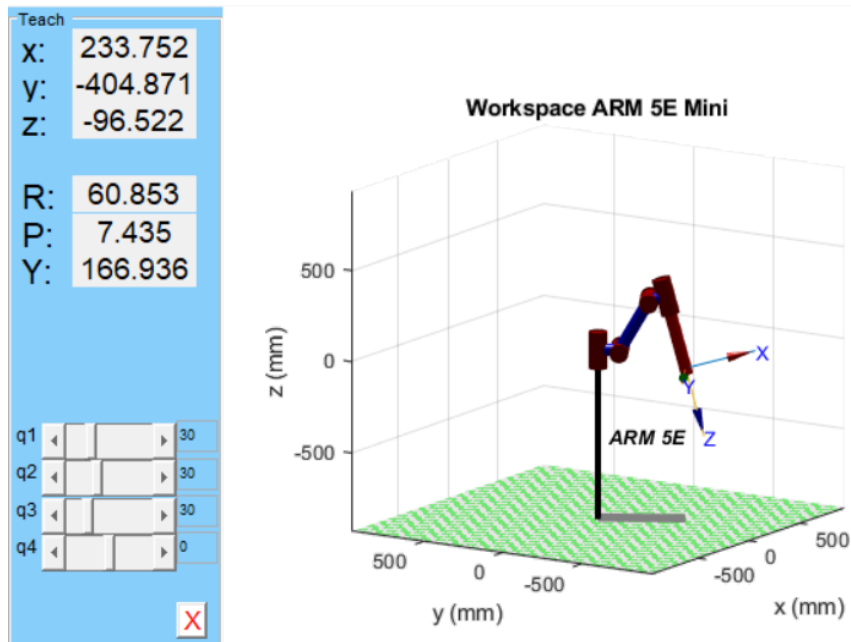


Figura 29: Teste manipulador posição 30° em cada junta.

Local minimum found.

Optimization completed because the size of the gradient is less than the value of the optimality tolerance.

<stopping criteria details>

x =

```
0.5236
0.5236
0.5236
```

res =

```
9.7019e-22
```

Figura 30: Resultado da otimização para o primeiro teste.



Testando para uma nova configuração, colocando as juntas *slew*, *shoulder* e *elbow* em 20°, 45° e 90° respectivamente, e implementando na interface do Peter Corke, o novo ponto obtido foi (252.073, -692.564, 98.913) que posiciona o manipulador na configuração da Figura 31.

Aplicando a otimização para esse ponto fornecido, o resultado dos ângulos das juntas foi de  $\theta_1 = 20.0000^\circ$ ,  $\theta_2 = 45.0001^\circ$ ,  $\theta_3 = 90.0001^\circ$  com um erro de  $4.3092e-22$ , provando então ter sido bem sucedida e com um erro desprezível.

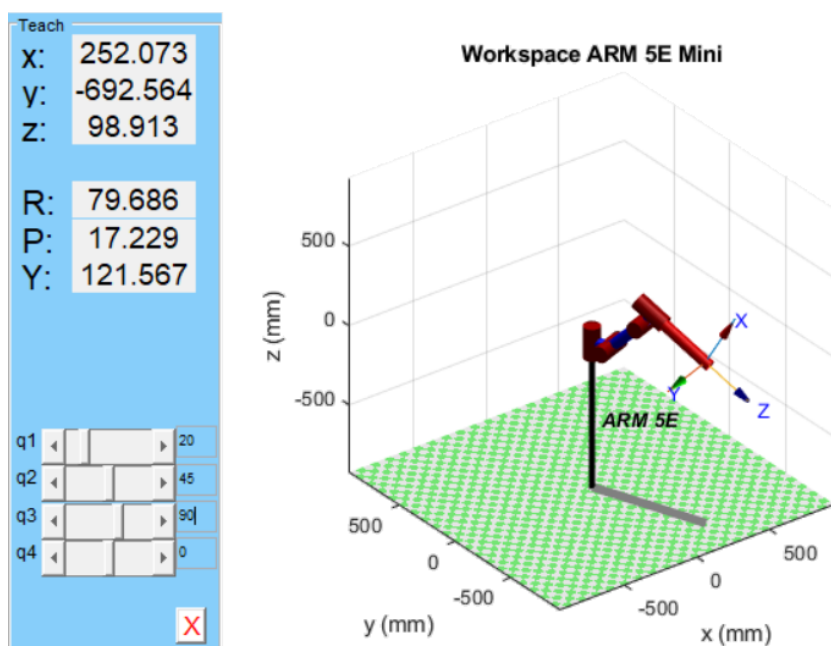


Figura 31: Teste manipulador configurado com juntas em 20°, 45° e 90°.

## 6 Testes Experimentais

### a Interface de comunicação

Uma interface de envio e recebimento de dados do manipulador foi desenvolvida em Python para o controle do braço robótico. O código para leitura dos dados do manipulador está de acordo com o protocolo fornecido pelo fabricante, que pode ser encontrado no apêndice a.

A primeira parte da interface consiste na leitura e exibição dos parâmetros medidos pelos sensores do manipulador, que são enviados para o computador em uma mensagem com 51 bytes. É necessário descompactar essa mensagem e interpretá-la para apresentar ao usuário informações como temperaturas, tensões, consumos de corrente e posição.

A versão inicial é feita no próprio terminal do computador, exibindo a mensagem em hexadecimal, em seguida separada pelos bytes de cada informação e por último exibindo os valores mestres de tensão, corrente e temperatura, além dos valores individuais de cada junta/motor.

Como é possível observar na Figura 32, o algoritmo recebe a mensagem e avalia se está incompleta através da verificação dos bytes iniciais e finais e exibe "Pacote OK". Em seguida, desempacota a mensagem de acordo com a quantidade de bytes referentes a cada informação e exibe esta mensagem desempacotada. Por fim, exibe as informações separadas por motor além das informações "master" referentes ao manipulador como um todo. Como validação em relação a leitura correta dos valores, bastou comparar os dados lidos pela serial com os dados apresentados pelo *software* do fabricante, que provou que o desempacotamento e a exibição da mensagem foram feitos corretamente.

```
e7 0b 8d 06 01 00 00 00 00 80 0b 00 00 01 00 00 00 00 40 0b 00 00 01 00
00 a8 e5
Pacote OK
(231, 11, 141, 6, 1, 0, 0, 128, 11, 0, 1, 0, 0, 64, 11, 0, 1, 0, 0, 192,
Início da mensagem
Master Temperature: 21.5354449522 °C
Master Voltage: 29.9198096886 V
Master Current: 0.362733704652 A
(Shoulder)
Posição: 0.0
Velocidade: 0 RPM
Corrente: 0 AD
Temperatura: 21.5354449522 °C
(Slew)
Posição: 0
Velocidade: 0 RPM
Corrente: 0 AD
Temperatura: 21.5354449522 °C
(Elbow)
Posição: 0.0
Velocidade: 0 RPM
Corrente: 0 AD
Temperatura: 19.5776772293 °C
(Jaw Rotate)
Posição: 0.345113298238
Velocidade: 0 RPM
Corrente: 0 AD
Temperatura: 21.5354449522 °C
(Jaw Open/Close)
Posição: 137.910734722
Velocidade: 0 RPM
Corrente: 0 AD
Temperatura: 21.5354449522 °C
Fim da mensagem
■
```

Figura 32: *Print* das informações recebidas do manipulador exibidas no terminal.

## b Controle de posição das juntas

Além do recebimento de dados, a interface é composta também com a parte de envio de dados, em que através de uma série de perguntas em relação às juntas, elabora-se uma mensagem baseada no protocolo e a envia para o manipulador, movimentando o mesmo. É possível observar uma prévia desse procedimento na Figura 33.

```
Qual motor deseja movimentar? (1-Shoulder, 2-Slew, 3-Elbow, 4-Jaw Rotate, 5-Jaw O/C) 1
Qual sera a demanda? (5-Posição) 5
Qual o angulo desejado? 30
Deseja movimentar algum outro motor? (s ou n)
Qual motor deseja movimentar? (1-Shoulder, 2-Slew, 3-Elbow, 4-Jaw Rotate, 5-Jaw O/C) 2
Qual sera a demanda? (5-Posição) 5
Qual o angulo desejado? 15
Deseja movimentar algum outro motor? (s ou n) n
30 15 0
X: 66.471 Y: -248.075 Z: 144.145

[231, 0, 0, 0, 0, 5, 26, 117, 3, 255, 7, 255, 0, 0, 5, 9, 236, 3, 255, 7, 255, 0, 0, 0, 0
0, 0, 0, 0, 3, 255, 7, 255, 0, 157, 229]

Enviando mensagem ...
Mensagem enviada
```

Figura 33: *Print* da rotina de perguntas no terminal para enviar mensagem para o manipulador.

Para escolher exatamente qual o movimento que o manipulador deverá performar, apresentam-se uma série de perguntas a respeito de qual junta deverá ser movimentada, qual o ângulo desejado e qual demanda o motor deverá performar. No caso deste projeto apenas a demanda de posição está sendo utilizada dado que se trata de uma interface de posicionamento.

Após a configuração, o algoritmo exibe no terminal os ângulos escolhidos e através do cálculo da cinemática direta, exibe qual será o ponto no espaço alcançado pelo efector final. Por fim, o algoritmo monta a mensagem (exibida no terminal) e a envia para o manipulador apresentando uma mensagem de "Enviando mensagem ..." e quando termina exibe "Mensagem enviada". Contudo, estas mensagens são apresentadas diversas vezes dado que o manipulador precisa receber a mensagem a cada 500 ms ou os motores suspendem o funcionamento como medida de segurança.

Para a implementação da cinemática direta do manipulador no Python, são utilizadas as equações (39), (40) e (41) obtidas na Seção 5a. Este algoritmo podem ser encontrado em [18].

Com o software de comando dos ângulos das juntas do manipulador implementado em Python, foram feitos diversos testes comparativos com o simulador do Matlab para validar o algoritmo da interface, considerando o manipulador em configurações diferentes. A seguir são apresentados os testes comparativos entre a simulação do Matlab e a cinemática direta aplicada no manipulador real.

A Figura 34 mostra o primeiro teste com as juntas *slew*, *shoulder* e *elbow* do manipulador configuradas em 90°, 0° e 25° respectivamente. Para o segundo teste, na Figura 35, foi adotada a configuração *slew*, *shoulder* e *elbow* iguais a 90°, 30° e 90°. Por último, a terceira posição de ângulos arbitrada foi *SLEW* em 45°, *SHOULDER* em 90° e *ELBOW* em 135°.

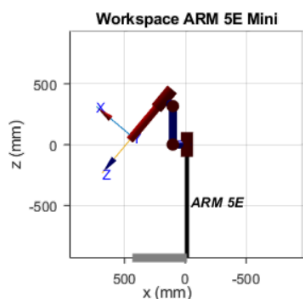
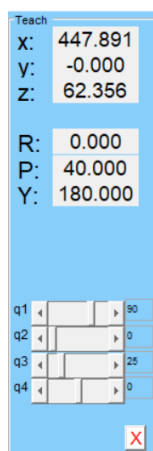


Figura 34: Manipulador posição 1 Matlab e real.

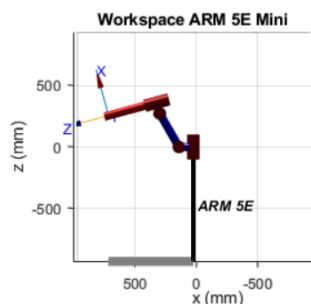
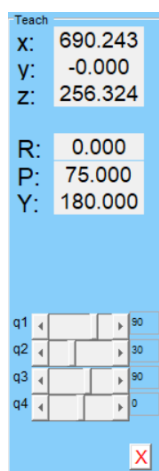


Figura 35: Manipulador posição 2 Matlab e real.

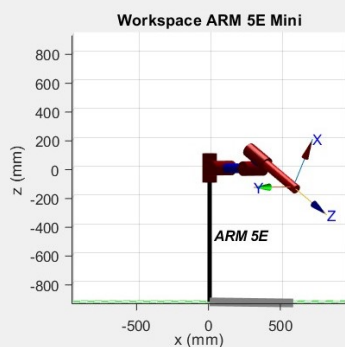
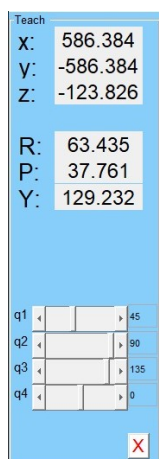


Figura 36: Manipulador posição 3 Matlab e real.

Os testes de movimentação das Figuras 34, 35 e 36, visualmente, parecem estar condizentes com a simulação do Matlab. Para validar a precisão de posicionamento do efector final do manipulador, um segundo experimento foi realizado. Nele, utilizou-se um plano adicional consistente em uma folha de papel e um marcador, de maneira a comparar e mensurar a diferença entre os pontos teóricos e os reais. Esse teste será detalhado nas seções a seguir.

### c Controle de posição do efector final

Para o controle de posicionamento do efector final, é preciso resolver a cinemática inversa do manipulador. O usuário informa a posição desejada para o efector final e um programa em Python computa os ângulos necessários em cada junta. Esse ângulos, por sua vez, são enviados como comandos ao manipulador através da interface de comunicação serial mencionada anteriormente.

Com o objetivo de tornar o projeto o mais modular possível, foi adicionada uma função no algoritmo para a leitura de arquivos do tipo .txt para que o usuário possa escolher o ponto desejado sem ter que interferir diretamente no algoritmo.

Para resolver o problema da cinemática inversa no Python, será utilizada a biblioteca SciPy, por conter módulos para otimização. Nela também existe a função `least_squares` [22] que recebe a função objetivo, um palpite inicial para as variáveis e os seus limites superiores e inferiores, e retorna uma estrutura com todas as informações da otimização realizada. Essa função resolve problemas de mínimos quadrados não-lineares com limites nas variáveis, assim como a função `lsqnonlin` do Matlab. Além disso, o solver *default* da função no Python também é o *Trust Region Reflective*, similar ao Matlab, tornando ainda mais fácil a sua implementação.

Para validar o algoritmo, foi testada a posição desejada do efector final em (233.752, -404.871, -96.522), que pode ser obtido deslocando as três primeiras juntas em 30°. O resultado obtido através da otimização coincide com os ângulos esperados:

$$\theta_1 = 30.000553^\circ, \theta_2 = 30.000581^\circ, \theta_3 = 29.9999^\circ \quad (51)$$

Contudo, nem todos os pontos desejados se encontram dentro da área de trabalho do manipulador. Para averiguar se um ponto é válido, o resultado da otimização é inserido no sistema de equações da cinemática direta apresentando um ponto  $x_d$ ,  $y_d$  e  $z_d$ . Caso a distância entre esse ponto e o ponto desejado seja maior que uma tolerância proposta, o ponto desejado se encontra fora da área de trabalho e portanto é inválido.

Além disso, a estrutura retornada pela função `least_squares` possui como um de seus dados o status da otimização (Sucesso/Falha), indicando se foi possível ou não realizar a otimização. Caso uma dessas condições não seja cumprida, uma mensagem de erro é apresentada ao usuário.

### d Controle de trajetória

Agora que o manipulador consegue alcançar um ponto desejado para o efector final, deve-se pensar na realização de trajetórias pré-estabelecidas, ou seja, o usuário escolhe um conjunto de pontos pelos quais o efector final do manipulador deverá percorrer.

Para o controle de trajetória, computa-se a cinemática inversa para o primeiro ponto desejado, envia-se os respectivos comandos de posição angular das juntas para o manipulador e, quando este ponto é atingido, segue-se para o próximo ponto até que toda a trajetória tenha sido percorrida. A verificação de chegada no ponto desejado é feita através do *feedback* dos parâmetros do manipulador, neste caso a posição. Durante a movimentação do manipulador, é possível monitorar a posição atual das juntas através dos *encoders* presentes em cada motor e, quando os valores lidos coincidirem com os valores obtidos a partir da cinemática inversa, considera-se que o manipulador chegou ao destino, podendo prosseguir então para o próximo ponto.

No entanto, como dito anteriormente, os parâmetros DH e a calibração de movimentação do manipulador baseada na relação ângulo/AD foram feitas empiricamente, o que torna o *feedback* de posição uma medida não muito confiável para verificar se o efector final atingiu o ponto dado.

Além disso, outro grande problema em relação ao *feedback* de posição foi o erro em regime permanente presente no controle posição das juntas fornecido pelo fabricante. Através de diversos testes e da leitura dos dados pela interface de comunicação, foi possível perceber esse erro residual. Por exemplo, quando



o algoritmo requeria uma posição de 1000 em AD em qualquer uma das juntas, o manipulador parava na posição 750 AD.

Uma forma de resolver esse problema poderia ser o ajuste manual dos ganhos do controlador PID de posição. Pelo comportamento observado nos testes, um aumento (ou inserção) de um ganho integrativo seria necessário. No entanto, ao checar o manual do fabricante, observou-se um aviso de que o controlador PID de posição ainda está em desenvolvimento pela empresa. Como os valores *default* dos ganhos do controlador PID de posição não estavam disponíveis no manual, optou-se por deixar o controlador inalterado para o escopo atual do projeto.

Uma segunda maneira de obter a verificação de que o ponto desejado foi atingido é pela análise da velocidade das juntas do manipulador. Nesse caso, verificou-se que quando todas as juntas têm velocidade nula, o manipulador para e a condição de ponto atingido é cumprida. No entanto, isso gerou dois problemas: o primeiro é que deixou a movimentação menos fluida, pois obriga o manipulador a parar completamente antes de seguir para o próximo ponto, e o segundo é que, como o manipulador parte do repouso, a velocidade inicial dele já se encontra nula.

Para resolver este problema foi utilizada a leitura da posição angular das juntas para verificar a saída da condição de repouso e, portanto, garantir que o critério de velocidade nula só passe a ser considerado depois que o manipulador tenha entrado em movimento. Assim, a velocidade só é verificada depois que houver variação na posição angular.

Um dos problemas em se utilizar a posição como critério para verificação da velocidade é a ação da vibração e da gravidade. Esses fatores podem gerar uma pequena variação na posição angular das juntas, criando a ilusão de que ela já entrou em movimento e parou. Isso faz com que o algoritmo eventualmente entenda que o ponto atual já foi atingido, quando não é verdade. Uma solução para isso foi também implementar uma tolerância de deslocamento para indicar o movimento do manipulador, fixada em 10 AD, valor arbitrário obtido através dos testes experimentais. Isso não só resolveu esse problema como melhorou a fluidez do movimento do manipulador, uma vez que quando o ponto é atingido, o manipulador não para completamente e já passa para o próximo ponto.

Na Figura 37, a visualização do progresso da trajetória exibe a condição atual do ponto, se foi alcançado ou não, o ponto desejado, a numeração do ponto atual, o valor em AD desejado das juntas, o *feedback* em AD da posição atual das juntas e a contagem de iterações realizadas em cada ponto.

```
Point reached: True
Trajectory Point: [368.951, -265.576, -334.077]
Point: 1
Target joint position (ad): [16923, 9186, 7283]
Actual joint position (ad): [16899, 8976, 7171]
Count: 100
```

Figura 37: *Print* da interface de trajetória

## e Preparação para testes

Diversas trajetórias de teste foram escolhidas para verificar o funcionamento do algoritmo implementado para controle de trajetória, entre elas, um quadrado, um círculo e um triângulo.

Com algoritmo de trajetória do manipulador estando pronto para os testes práticos, dá-se início a montagem da área de testes e do posicionamento do manipulador. Como visto anteriormente, o manipulador se encontra fixado a um carro de ferramentas do laboratório. Utilizando uma coluna estrutural do laboratório como referência, foi possível posicionar o manipulador de forma que, caso ele fosse movido acidentalmente, haveria sempre uma referência para que os testes sempre se iniciassem da mesma posição.

Além disso, foi fixada numa posição arbitrária dentro do *workspace* do manipulador uma chapa de poli-estireno com um papel quadriculado acoplado, permitindo que vários testes pudessem ser realizados e uma análise comparativa entre todas as trajetórias desenhadas pudesse ser feita. Um lápis foi preso a garra do manipulador para que as trajetórias pudessem ser traçadas e uma espuma foi acoplada à chapa de poliestireno para que o impacto do lápis na chapa pudesse ser absorvido, não danificando sua ponta e nem a folha aonde a trajetória seria desenhada. As Figuras 38 e 39 mostram a estrutura dos testes experimentais.

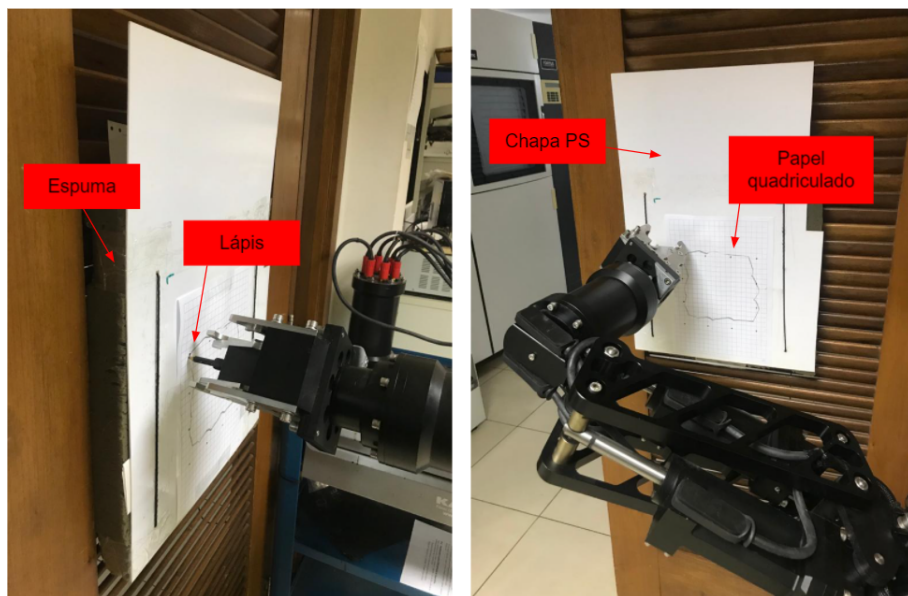


Figura 38: Posicionamento dos componentes do teste.



Figura 39: Posicionamento manipulador.



## f Trajetórias

A primeira trajetória teste escolhida para o manipulador percorrer foi a de um quadrado com quatro pontos equidistantes em cada aresta, fazendo o manipulador percorrer 60 cm de perímetro.

Inicialmente, escolheu-se o primeiro ponto do quadrado da quina superior esquerda, alinhando o manipulador no papel quadriculado através do *software* do fabricante em conjunto com o *joystick* e converteu-se o valor exibido em AD para ângulos através da relação empírica apresentada na Seção 2. A partir desses valores, é possível obter as coordenadas do efector final no espaço.

Em seguida, foram adicionados 50 mm entre cada ponto criando o conjunto de pontos da trajetória desejada, como mostra a Figura 40.

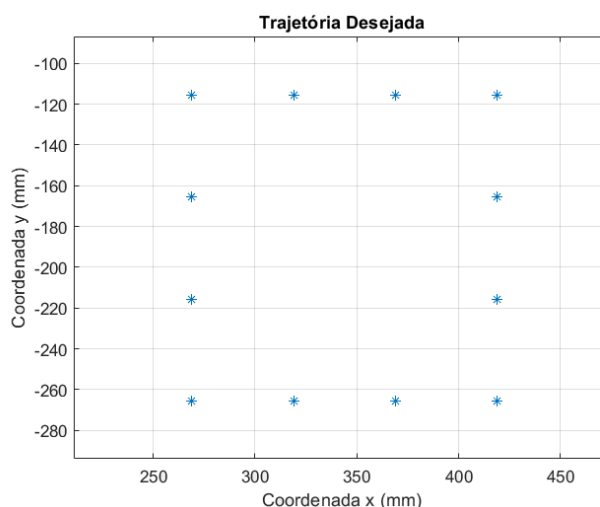


Figura 40: Conjunto de pontos da trajetória quadrado.

A Figura 41 mostra a trajetória real realizada pelo manipulador sobre a mesma folha de teste por 10 vezes, onde foi possível validar a extrema taxa de repetibilidade (precisão) do manipulador. No entanto, é possível perceber a falta de acurácia por parte do manipulador uma vez que a trajetória realizada não passa por vários dos pontos desejados. A Figura 42 mostra a trajetória desejada, representada pelos pontos fornecidos como losangos azuis; a trajetória obtida através dos resultados da cinemática inversa, representada pela linha laranja; e a trajetória real do manipulador (ou atingida), em amarelo, criada através dos dados obtidos durante os testes. A Figura 43, por sua vez, mostra separadamente as coordenadas  $x$  e  $y$  dessas três análises da trajetória para ressaltar a diferença entre o ponto desejado e o ponto obtido.

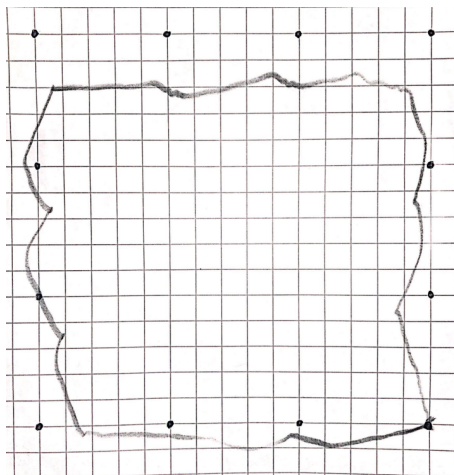


Figura 41: Mesma trajetória desejada percorrida 10 vezes pelo manipulador, em conjunto com a demarcação dos pontos desejados.

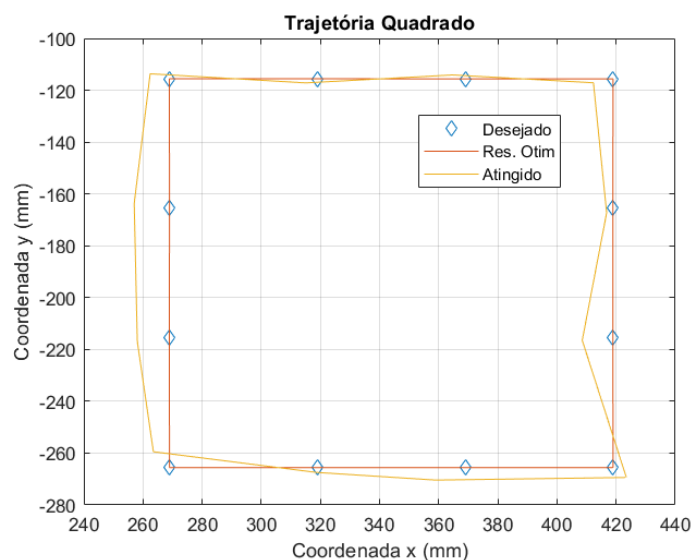


Figura 42: Trajetórias comparadas: Teórica (Pontos em losango azul), Otimizada (trajetória em laranja), Real (trajetória em amarelo).

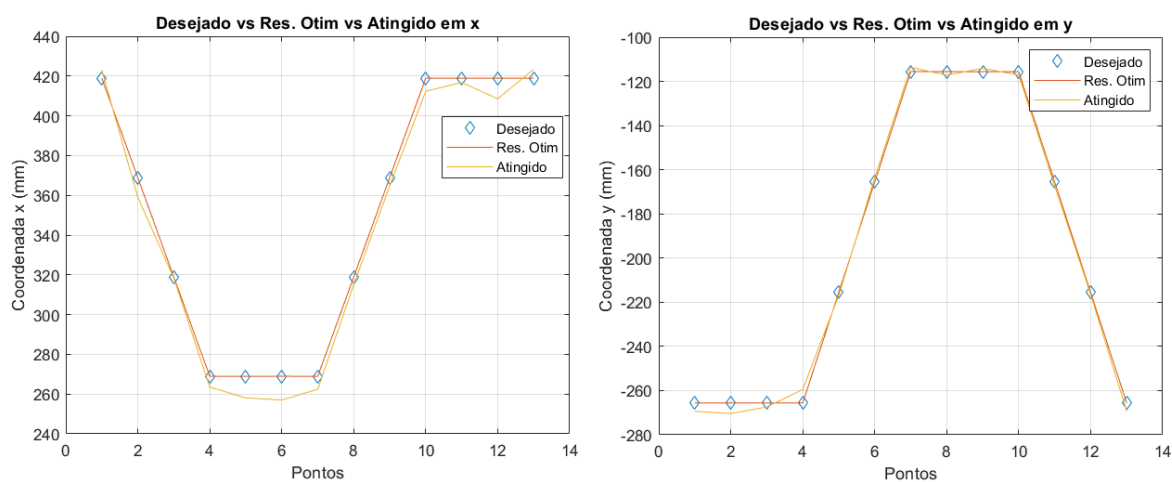


Figura 43: Coordenadas  $x$  e  $y$  comparadas: Teórica, Otimizada, Real.

Pela Figura 42, é possível verificar que a cinemática inversa foi bem sucedida em todos os pontos, já que a trajetória otimizada obtida, em laranja, coincidiu com os pontos desejados, em azul. Para calcular a trajetória otimizada, os ângulos otimizados pela cinemática inversa foram inseridos nas equações da cinemática direta para obter as posições correspondentes do efector final.

Ao comparar a imagem escaneada na Figura 41 com os gráficos do Matlab, é possível perceber que existe uma diferença grande entre as duas representações. O erro entre a trajetória real executada pelo manipulador e os pontos desejados não fica tão visível nos gráficos do Matlab. Isso pode ser justificado por alguns fatores. Primeiramente, a conversão dos valores AD para ângulos e vice-versa não é muito precisa. Por usar apenas o limite máximo das juntas, isso afeta diretamente no *feedback* de posição e na estimativa da posição do efector final. Caso houvessem mais informações no manual do fabricante a respeito do *encoder* responsável por medir a posição das juntas, essa conversão poderia ser mais precisa, diminuindo o erro.

Outro erro que contribuiu para a baixa acurácia da trajetória foi o fato do controlador PID de posição não estar apresentando um resultado satisfatório. Possivelmente o ganho integral está baixo ou até mesmo nulo, o que faz com que não haja um aumento significativo do erro no regime permanente. Isso pode ser observado na Figura 44, em que as posições desejadas em AD das juntas não condizem com as reais enviadas pelo manipulador. Para quantificar esse erro, foi realizada a média entre o valor desejado e o real de cada junta. Para as juntas *shoulder*, *slew* e *elbow*, o erro médio foi de 100.000 AD, 103.3846 AD e 144.1538 AD, respectivamente. Isso representa um erro médio em ângulo de aproximadamente 0,443°, 0,608° e 1,028°.

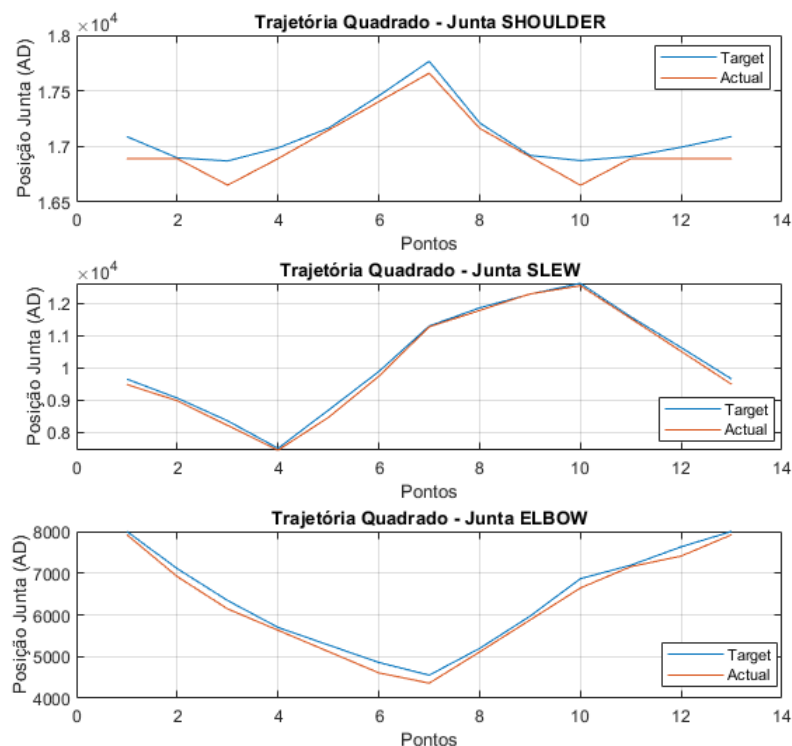


Figura 44: Valor real e desejado em AD das três juntas do manipulador.

Por último, o fato dos parâmetros Denavit-Hartenberg terem sido medidos empiricamente com instrumentos de laboratório, como escalas e paquímetros, fez com que não ficassem muito precisos. Caso houvesse disponibilidade do CAD do manipulador pelo fabricante, seria possível obter a medida exata dos mesmos, fazendo com que as equações de cinemática direta ficassem bem mais precisas, diminuindo o erro da posição do efector final e, portanto, deixando a trajetória percorrida pelo manipulador mais precisa.

Outros tipos de trajetória foram testadas para avaliar o comportamento da movimentação das juntas rotativas do manipulador. Primeiro, um triângulo isósceles com 4 pontos em cada aresta, de base e altura iguais a 15 cm e em seguida um círculo de 17 pontos com raio de 7,5 cm.

Na Figura 45, é possível observar que o manipulador conseguiu realizar trajetórias mais retas, mas ainda muito impreciso em relação aos pontos. Comparando com a Figura 46, o vértice inferior do triângulo é o que está mais similar a trajetória executada de fato pelo manipulador. Todo o resto da trajetória atingida ficou diferente devido aos erros mencionados anteriormente.

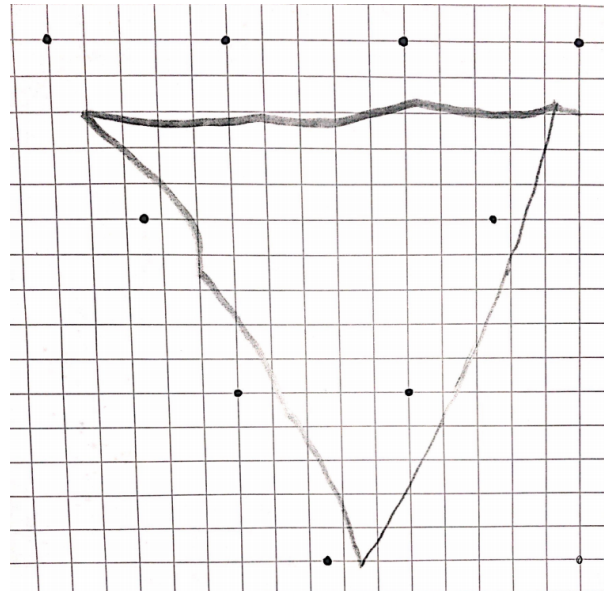


Figura 45: Trajetória triângulo.

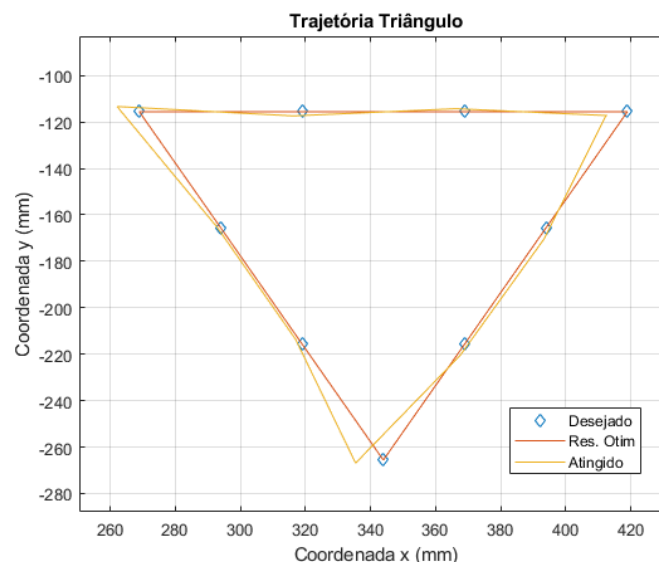


Figura 46: Coordenadas comparadas: Teórica, Otimizada, Real.

No caso do círculo, na Figura 48, foi possível observar os mesmos movimentos circulares que ocorreram trajetória quadrado. Dado que a trajetória se trata de um círculo, esses movimentos auxiliaram em dar uma aparência mais circular no caminho desejado. Na Figura 49, a trajetória real se assemelha com a Figura 48, evidenciando o deslocamento de toda a trajetória para o lado, devido aos mesmos erros mencionados anteriormente.

Em ambas as trajetórias também foram feitas análises das coordenadas  $x$  e  $y$  separadamente, Figuras 47 e 50, para avaliar individualmente os desvios entre a trajetória real e a desejada.

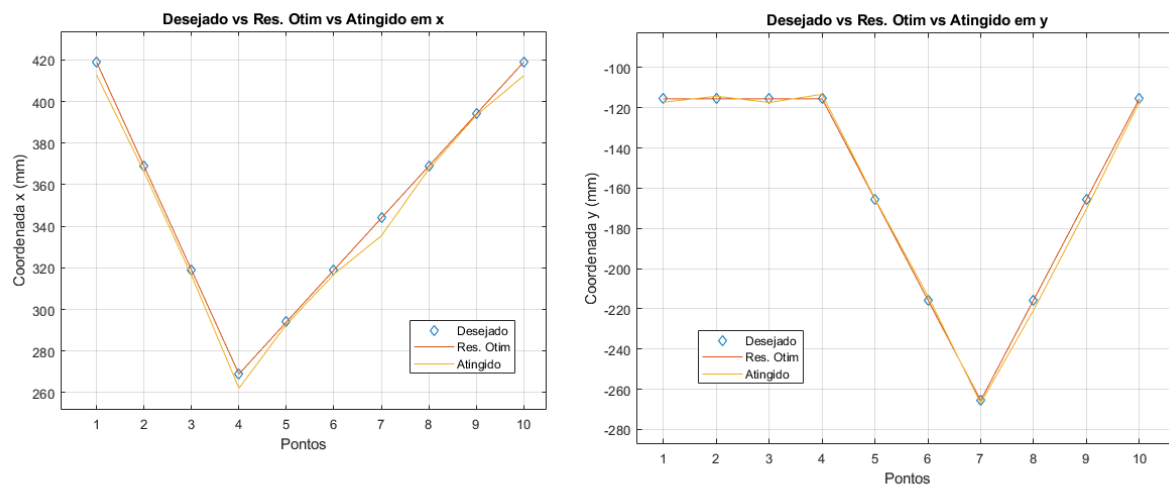


Figura 47: Trajetória triangular: coordenadas x e y comparadas (teórica, otimizada e real).

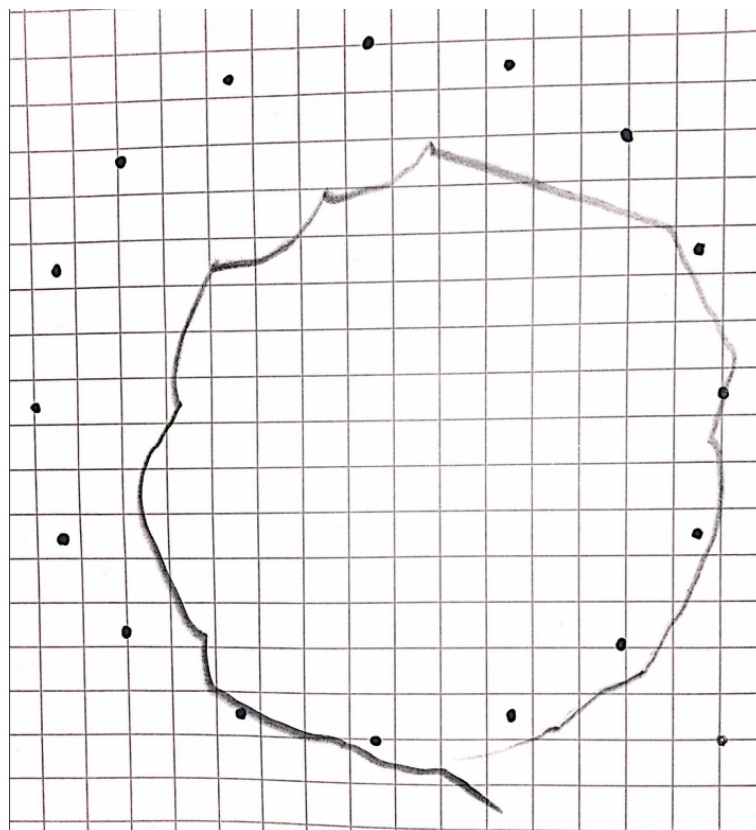


Figura 48: Trajetória círculo.

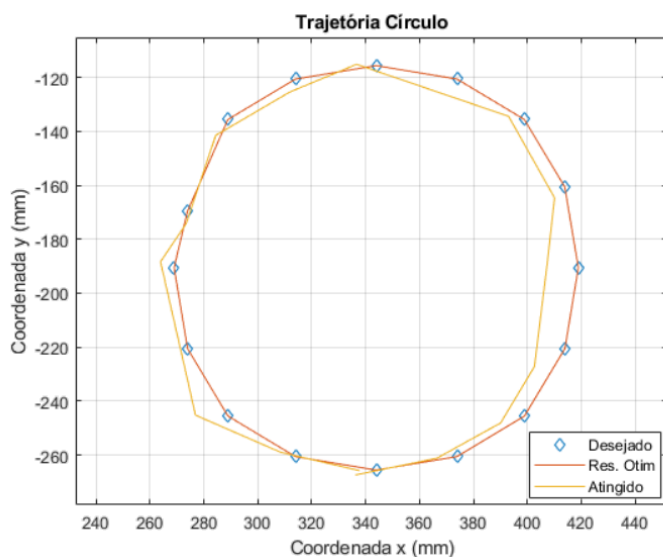


Figura 49: Coordenadas comparadas: Teórica, Otimizada, Real.

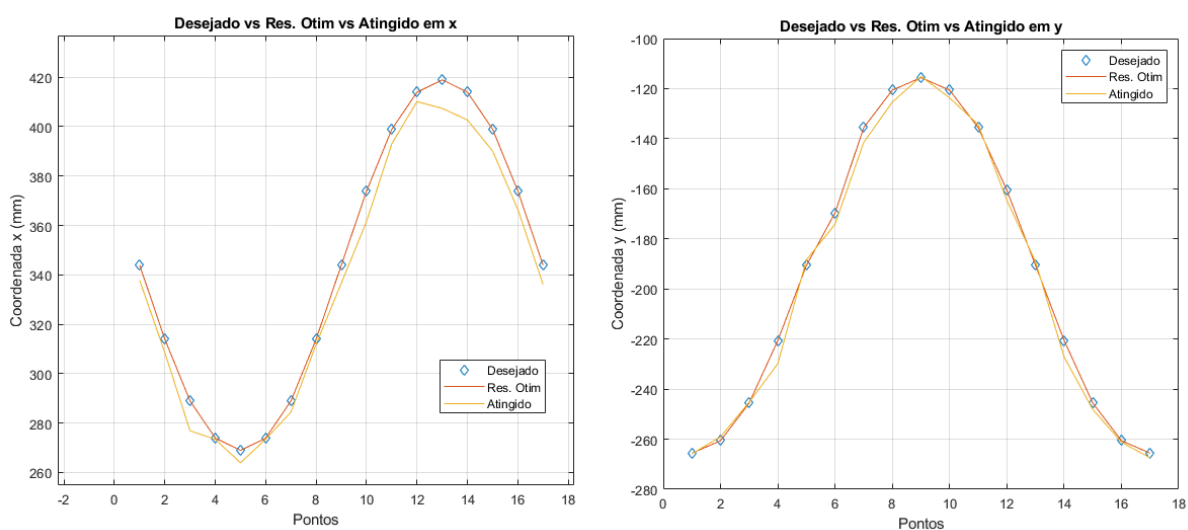


Figura 50: Coordenadas  $x$  e  $y$  comparadas: Teórica, Otimizada, Real.

### g Custo computacional

No caso do algoritmo de solução da cinemática inversa do manipulador, por ter sido implementado em Python 2.7, bastou utilizar a biblioteca `time` [23], que contabiliza o tempo em segundos desde a época, chamando a função antes de a otimização ser realizada e depois de ela terminar. A partir disso, foi feita a checagem da validação dos pontos e calculada a diferença do tempo, de modo a obter, então, o custo computacional da otimização.

Inicialmente, calculou-se o custo computacional para a otimização de um único ponto diversas vezes, obtendo uma média de 8 milissegundos. Realizando o mesmo procedimento por *loop* iterativo para todos os 13 pontos da trajetória quadrado, o custo computacional foi de 95 milissegundos.

Considerando que a otimização é feita a cada novo deslocamento do manipulador, 8 milissegundos constituem perda considerável de tempo, o que pode comprometer a fluidez do movimento. Uma possível melhoria para o projeto seria a implementação da otimização de todos os pontos antes mesmo da execução do comando pelo manipulador, o que demandaria maior intervalo para começar, mas pouparia tempo durante a movimentação. Contudo, isso dependeria de reavaliar o método de otimização *least squares* para receber a trajetória por completo ou realizar o próprio *loop* iterativo.

Para realizar as otimizações em Python foi utilizada a seguinte máquina:

PC Montado  
OS: openSUSE Leap 15.0  
CPU: Intel(R) Core(TM) i3-2100  
Memória RAM: 8,0 GB

Para as otimizações e simulações em Matlab:

Lenovo Ideapad S145  
OS: Windows 10 Home Single Language  
CPU: Intel(R) Core(TM) i7-8565U  
Memória RAM: 12,0 GB  
Versão Matlab R2019b



## 7 Conclusões e Trabalhos Futuros

Neste projeto, foram abordados conceitos de cinemática direta e inversa a fim de realizar o controle de trajetória de um manipulador com 5 graus de liberdade. Estes conceitos permitiram estimar a posição do efector final através de dimensões físicas do manipulador, os parâmetros de Denavit-Hartenberg.

O uso do método dos mínimos quadrados utilizando o algoritmo *Trust Region Reflective* se provou eficaz como otimização do sistema não-linear para encontrar a angulação necessária das juntas para que o efector final pudesse atingir cada um dos pontos propostos da trajetória.

Apesar da trajetória ter sido completada, ficou clara a falta de acurácia do manipulador e quais as possíveis causas. Mas levando em consideração que grande parte dos parâmetros utilizados foram obtidos empiricamente, o resultado final deste projeto foi bastante satisfatório e abre margem para trabalhos futuros.

Além disso, ficaram visíveis as possíveis aplicações de se poder pré-programar a trajetória a ser realizada por um manipulador em operações submarinas, sem precisar depender da precisão humana. Alguns dos pontos a serem melhorados no projeto seriam mitigar os erros mencionados anteriormente como, por exemplo, implementar o controle integral de posição, criar um modelo CAD do manipulador para poder obter os parâmetros de Denavit-Hartenberg de forma mais precisa, e fabricar uma bancada de testes mais estável.

Um link para o vídeo com os testes realizados com manipulador no Centro de Pesquisa em Tecnologia de Inspeção na PUC-Rio está disponível em <https://youtu.be/P7ZdWd60ChI>. E o link para o repositório com algoritmos utilizados neste projeto está disponível em <https://github.com/BLacklight19/Manipulador>

As melhorias mencionadas anteriormente são referentes aos erros e dificuldades encontrados durante a elaboração deste projeto. Além disso, como trabalhos futuros ao projeto sugere-se o rastreamento também da orientação do efector final e o desenvolvimento de uma interface com o usuário que permita a visualização da trajetória desejada do manipulador.

### a Orientação e ação da ferramenta

O objetivo deste trabalho foi fazer o controle de trajetória posicionando o manipulador em pontos desejados. Para isso, apenas as equações de posição foram utilizadas nos problemas de cinemática inversa e direta.

Uma melhoria para o projeto seria levar em consideração também a orientação do efector final em conjunto com a ação de abrir/fechar da garra. Dessa forma, a interface de posicionamento permitiria a interação do manipulador com o meio ambiente, sendo possível, por exemplo, girar alavancas e segurar objetos.

### b Interface

Uma outra melhoria idealizada para este projeto foi a elaboração de uma interface mais intuitiva para o usuário. Apesar de ser possível visualizar todas as informações no terminal, nem sempre os dados ficam alinhados com a página, fazendo com que fique difícil de analisar as leituras e os dados que estão sendo enviados.

Como todos os algoritmos de controle do manipulador foram escritos em Python, optou-se por escolher a biblioteca de interface Tkinter [24] para exibir as informações dos sensores, poder enviar a posição ou ângulos de junta desejados, bem como visualizar uma prévia da trajetória a ser realizada pelo manipulador.

Na Figura 51, é possível observar um exemplo de interface gráfica para este projeto. No canto superior esquerdo está um *dropdown menu* para que o usuário possa escolher entre cinemática direta e inversa, tendo como campo de entrada os ângulos das juntas ou o ponto no espaço, respectivamente. Escolhendo as entradas, basta clicar em "Test" para surgir à direita uma pré-visualização do ponto final ou da trajetória do efector final. Clicando em "Apply", surge uma pergunta de segurança, como na Figura 52, perguntando se o usuário deseja prosseguir para enviar os dados para o manipulador.

Abaixo do campo da cinemática estariam as informações de monitoramento de cada motor, enviadas pelo manipulador. No canto superior direito estão os valores *Master* de temperatura, corrente e tensão do manipulador e, por último, o campo que apresenta uma pré-visualização da trajetória a ser percorrida pelo manipulador.

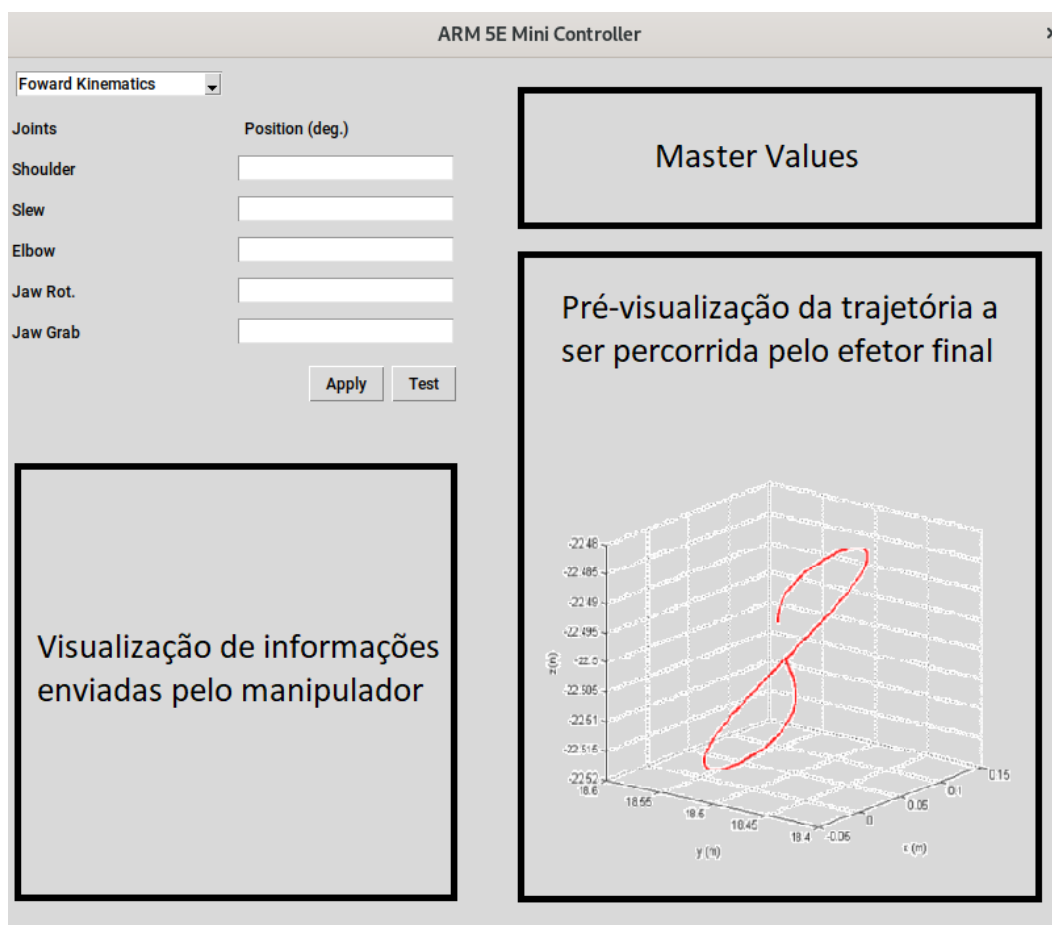


Figura 51: Conceito da interface de controle de trajetória do manipulador.

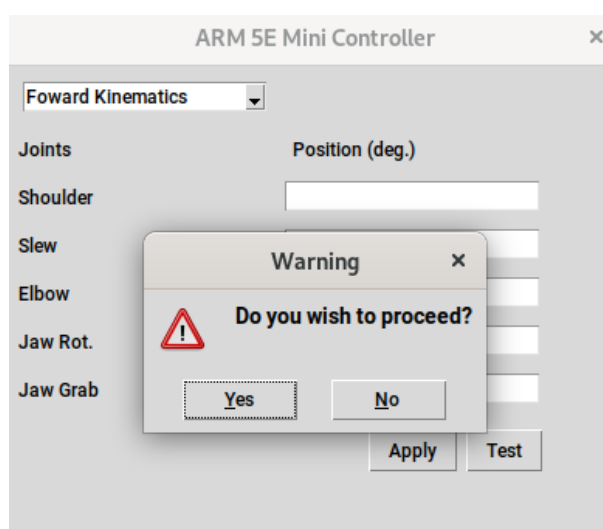
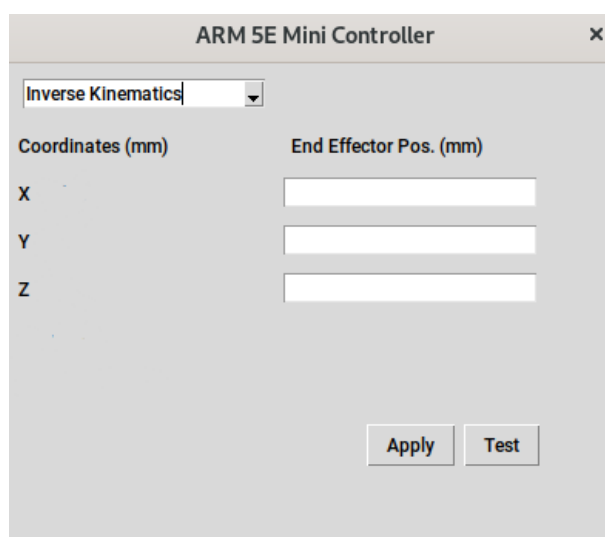


Figura 52: Conceito da interface de controle - seção de aviso antes do teste.



The image shows a software window titled "ARM 5E Mini Controller" with a close button (x) in the top right corner. Inside the window, there is a dropdown menu currently set to "Inverse Kinematics". Below this, there are two columns of input fields. The first column is labeled "Coordinates (mm)" and contains three rows labeled "X", "Y", and "Z". The second column is labeled "End Effector Pos. (mm)" and contains three corresponding empty input boxes. At the bottom right of the window, there are two buttons labeled "Apply" and "Test".

Figura 53: Conceito da interface de controle - seção de cinemática inversa.

## 8 Referências

- [1] E. Robotics. Eca website. [Online]. Available: <https://www.ecagroup.com/en/business/shephard-manipulator-arms-beachcombers-h800-rov>
- [2] S. Sivčev, J. Coleman, E. Omerdić, G. Dooly, and D. Toal, "Underwater manipulators: A review," *Ocean Engineering*, vol. 163, pp. 431–450, 2018. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0029801818310308>
- [3] J. Forster. Watches, automatons, 'soul,' and the digesting duck of jacques de vaucanson. [Online]. Available: <https://www.hodinkee.com/articles/watches-automatons-soul-and-the-digesting-duck-of-jacques-de-vaucanson>
- [4] I. Asimov, *I, Robot*, 1950.
- [5] S. Keating, "Mit m.sc. thesis - renaissance robotics : novel applications of multipurpose robotic arms spanning design fabrication, utility, and art," Ph.D. dissertation, 03 2013.
- [6] cottonbro. House robot. [Online]. Available: <https://www.pexels.com/photo/person-holding-white-round-device-4107252/>
- [7] M. A. Meggiolaro, "Notas de aula da disciplina mec2402 - controle de sistemas robóticos," vol. 1, 2021.
- [8] M. W. Spong, S. Hutchinson, M. Vidyasagar *et al.*, *Robot modeling and control*. Wiley New York, 2006.
- [9] P. Flores, "Análise cinemática e dinâmica de mecanismos com recurso a meios computacionais," 09 2015.
- [10] K. TeleRobotics. Viper kraft telerobotics webpage. [Online]. Available: <http://krafttelerobotics.com/products/viper.htm>
- [11] Manipulador puma. [Online]. Available: <http://automacaoerobotica.blogspot.com/2012/07/11-robos-manipuladores-industriais.html>
- [12] E. Robotics. Eca website. [Online]. Available: <https://www.ecagroup.com/en/solutions/arm-5e-mini>
- [13] T. G. Fcamidu, "Modelagem cinemática de um robô antropomórfico," *PUC-RIO - Maxwell*, vol. 1, pp. 14–21, 12 2017.
- [14] J. Batista, "Notas de aula da disciplina de robótica - isr-coimbra," vol. 4-5, 2010.
- [15] R. Nilsson, "Inverse kinematics," 2009.
- [16] J. C. García, J. Javier Fernández, R. M. P. J. Sanz, and M. Prats, "Towards specification, planning and sensor-based control of autonomous underwater intervention\*," *IFAC Proceedings Volumes*, vol. 44, no. 1, pp. 10 361–10 366, 2011, 18th IFAC World Congress. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S1474667016452778>
- [17] J. J. Fernandez, M. Prats, P. J. Sanz, J. C. Garcia, R. Marin, M. Robinson, D. Ribas, and P. Ridao, "Grasping for the seabed: Developing a new underwater robot arm for shallow-water intervention," *IEEE Robotics Automation Magazine*, vol. 20, no. 4, pp. 121–130, 2013.
- [18] L. O. Vivian. Repositório manipulador. [Online]. Available: <https://github.com/BBlacklight19/Manipulador>
- [19] P. Corke, "A robotics toolbox for matlab," *IEEE Robotics Automation Magazine*, vol. 3, no. 1, pp. 24–32, March 1996.
- [20] MathWorks. lsqnonlin. [Online]. Available: <https://www.mathworks.com/help/optim/ug/lsgnonlin.html>
- [21] ——. Choosing the algorithm. [Online]. Available: <https://www.mathworks.com/help/optim/ug/choosing-the-algorithm.html>
- [22] SciPy. Scipy optimize - least squares. [Online]. Available: [https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.least\\_squares.html](https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.least_squares.html)
- [23] T. P. S. Foundation. time — time access and conversions. [Online]. Available: <https://docs.python.org/3/library/time.html>

- [24] ——. tkinter — python interface to tcl/tk. [Online]. Available: <https://docs.python.org/3/library/tkinter.html>

## A Apêndice

### a Protocolo Manipulador [12]

#### 1 Envia dados para manipulador

Index	Example Value (hex)	Description	Checksum calculation example
0 (first transmitted)	0xE7	Start of message 0xE7	E7
1	0x00 Reserved for future use	Master Temperature	E7+0= E7
2	0x00 Reserved for future use	Master Voltage	E7+0= E7
3	0x00 Reserved for future use	Master Current	E7+0= E7
4	0x00	Motor 1 demand message prefix	E7+0= E7
5	0x01	Motor 1 Voltage demand clockwise	E7+1= E8
6	0xFF	Motor1 Demand MSB 100% PWM	E7+FF= E6 (note roll over)
7	0xFF	Motor 1 Demand LSB 100% PWM	E6+FF= E5
8	0x0F	Motor 1 Speed MSB Limit = 4095 RPM	E5+0F=F4
9	0xFF	Motor 1 Speed LSB Limit = 4095 RPM	F4+FF=F3
10	0x0F	Motor 1 Current MSB Limit = full current	F3+0F= 02
11	0xFF	Motor 1 Current Limit LSB = full current	02+FF= 01
12	0x00 Reserved for future use	Motor 1 Data 7	01+00= 01
13	0x00	Motor 2 demand message prefix	01+00= 01
14	0x03	Motor 2 Speed demand clockwise	01 + 03 = 04
15	0x03	Motor 2 Demand MSB 1000RPM	04 + 03 = 07
16	0xE8	Motor 2 Demand LSB 1000RPM	07 + E8 = EF
17	0x0F	Motor 2 Speed MSB Limit = 4095 RPM	EF + 0F = FE
18	0xFF	Motor 2 Speed LSB Limit = 4095 RPM	FE + FF = FD
19	0x0F	Motor 2 Current MSB Limit = full current	FD + 0F = 0C

20	0xFF	Motor 2 Current LSB Limit = full current	0C + FF = 0B
21	0x00 Reserved for future use	Motor2 Data 7	0B + 0 = 0B
22	0x00	Motor 3 demand message prefix	0B + 0 = 0B
23	0x05	Motor 3 Position demand	0B + 05 = 10
24	0x1F	Motor 3 Demand MSB Pos = 8177	10 + 1F = 2F
25	0xF1	Motor 3 Demand LSB Pos = 8177	2F + F1 = 20
26	0x0F	Motor 3 Speed MSB Limit = 4095 RPM	20 + 0F = 2F
27	0xFF	Motor 3 Speed LSB Limit = 4095 RPM	2F + FF = 2E
28	0x0F	Motor 3 Current MSB Limit = full current	2E + 0F = 3D
29	0xFF	Motor 3 Current LSB Limit = full current	3D + FF = 3C
30	0x00 Reserved for future use	Motor 3 Data 7	3C + 00 = 3C
31	0x01	Motor 4 PID Message prefix	3C + 01 = 3D
32	0xFF	Motor 4 P Position	3D + FF = 3C
33	0x0F	Motor 4 I Position	3C + 0F = 4B
34	0xF0	Motor 4 D Position	4B + F0 = 3B
35	0xFF	Motor 4 P Speed	3B + FF = 3A
36	0x01	Motor 4 I Speed	3A + 01 = 3B
37	0x77	Motor 4 D Speed	3B + 77 = B2
38	0x00 Reserved for future use	Motor3 Data 6	B2 + 00 = B2
39	0x00 Reserved for future use	Motor3 Data 7	B2 + 00 = B2
40	0x01	Motor 5 PID Message prefix	B2 + 01 = B3
41	0xFF	Motor 5 P Position	B2 + FF = B1
42	0x0F	Motor 5 I Position	B1 + 0F = C0
43	0xF0	Motor 5 D Position	C0 + F0 = B0
44	0xFF	Motor 5 P Speed	B0 + FF = AF
45	0x01	Motor 5 I Speed	AF + 01 = B0
46	0x77	Motor 5 D Speed	B0 + 77 = 27
47	0x00 Reserved for future use	Motor5 Data 6	27 + 0 = 27
48	0x00 Reserved for future use	Motor5 Data 7	27 + 0 = 27
49	0x27	Checksum	=27
50	0xE5	EOM 0xE5	



## 2 Recebe dados do manipulador

Index	Example Value (hex)	Description	Checksum calculation example
0 (first transmitted)	0xE7	Start of message 0xE7	E7
1	0x14	Master Temperature = 39.1°C	E7+0x14 = 0xFB
2	0x76	Master Voltage =25V (see conversions section)	FB + 0x76 = 0x71
3	0x0D	Master Current =1A (see conversions section)	0x71+0x0D= 0x7E
4	0x01	Motor 1 Sensors prefix	7E + 0x01 =7F
5	0x0F	Motor 1 position MSB =4095	7F + 0x0F = 0x8E
6	0xFF	Motor1 position LSB =4095	8E + 0xFF = 0x8D
7	0x03	Motor 1 speed MSB =1023 RPM	8D + 0x03 = 0x90
8	0xFF	Motor 1 speed LSB = 1023 RPM	0x90+0xFF= 0x8F
9	0x00	Motor 1 Current MSB = 0	0x8F+0x00=0x8F
10	0x00	Motor 1 Current LSB =0	0x8F+0x00=0x8F
11	0x14	Motor 1 temperature = 39.1 °C	0x8F + 0x14 =0xA3
12	0x00 Reserved for future use	Motor 1 Data 7	0xA3 + 0x00 = 0xA3
13	0x01	Motor 2 Sensors prefix	0xA3+0x01=0xA4
14	0x03	Motor 2 position MSB = 1023	0xA4 + 0x03 = 0xA7
15	0xFF	Motor 2 position LSB = 1023	0xA7+0xFF=0xA6
16	0x0F	Motor 2 speed MSB =4095 RPM	0xA6+0x0F= 0xB5
17	0xFF	Motor 2 speed LSB =4095 RPM	0xB5 + 0xFF =0xB4
18	0x00	Motor 2 Current MSB =0	0xB4 + 0x00= 0xB4
19	0x00	Motor 2 Current LSB =0	0xB4+0x00=0xB4

20	0x14	Motor 2 temperature = 39.1 °C	0xB4+0x14=0xC8
21	0x00 Reserved for future use	Motor2 Data 7	0xC8+0x00=0xC8
22	0x01	Motor 3 Sensors prefix	0xC8+0x01 = 0xC9
23	0x0F	Motor 3 position MSB =4095	0xC9+0x0F = 0xD8
24	0xFF	Motor 3 position LSB =4095	0xD8+0xFF=0xD7
25	0x03	Motor 3 speed MSB =1023 RPM	0xD7+0x03=0xDA
26	0xFF	Motor 3 speed LSB = 1023 RPM	0xDA+0xFF=0xD9
27	0x00	Motor 3 Current MSB = 0	0xD9+0x00 = 0xD9
28	0x00	Motor 3 Current LSB =0	0xD9+0x00 = 0xD9
29	0x14	Motor 3 temperature = 39.1 °C	0xD9+0x14= 0xED
30	0x00 Reserved for future use	Motor 3 Data 7	ED+0x00= 0xED
31	0x01	Motor 4 Sensors prefix	0xED+0x01 = 0xEE
32	0x03	Motor 4 position MSB = 1023	0xEE+0x03= 0xF1
33	0xFF	Motor 4 position LSB = 1023	0xF1 + 0xFF= 0xF0
34	0x0F	Motor 4 speed MSB =4095 RPM	0xF0+0x0F=0xFF
35	0xFF	Motor 4 speed LSB =4095 RPM	0xFF+0xFF=0xFE
36	0x00	Motor 4 Current MSB =0	0xFE+0x00=0xFE
37	0x00	Motor 4 Current LSB =0	0xFE+0x00=0xFE
38	0x14	Motor 4 temperature = 39.1 °C	0xFE+0x14= 0x12
39	0x00 Reserved for future use	Motor4 Data 7	0x12+0x00=0x12

40	0x01	Motor 5 Sensors prefix	0x12+0x01 = 0x13
41	0xFF	Motor 5 position MSB = 65280	0x13+0xFF=0x12
42	0x00	Motor 5 position LSB = 65280	0x12+0x00=0x12
43	0x03	Motor 5 speed MSB = 1023 RPM	0x12+0x03=0x15
44	0xFF	Motor 5 speed LSB = 1023 RPM	0x15+0xFF=0x14
45	0x00	Motor 5 Current MSB = 0	0x14+0x00=0x14
46	0x00	Motor 5 Current LSB = 0	0x14+0x00=0x14
47	0x14	Motor 5 temperature = 39.1 °C	0x14+0x14=0x28
48	0x00 Reserved for future use	Motor5 Data 7	0x28+0x00=0x28
49	0x28	Checksum	0x28
50	0xE5	EOM 0xE5	