

**INSTITUTO FEDERAL
DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA**
Goiano

**DESENVOLVIMENTO DO ROBÔ SCARA PARA FINS
DIDÁTICOS E APLICAÇÕES EDUCACIONAIS NA
ENGENHARIA ELÉTRICA**

Vitor Ramos Machado
Rafael de Jesus
Bruno Vieira Alves

Instituto Federal Goiano
Campus Trindade
16 de fevereiro de 2024

TERMO DE CIÊNCIA E DE AUTORIZAÇÃO PARA DISPONIBILIZAR PRODUÇÕES TÉCNICO-CIENTÍFICAS NO REPOSITÓRIO INSTITUCIONAL DO IF GOIANO

Com base no disposto na Lei Federal nº 9.610, de 19 de fevereiro de 1998, AUTORIZO o Instituto Federal de Educação, Ciência e Tecnologia Goiano a disponibilizar gratuitamente o documento em formato digital no Repositório Institucional do IF Goiano (RIIF Goiano), sem ressarcimento de direitos autorais, conforme permissão assinada abaixo, para fins de leitura, download e impressão, a título de divulgação da produção técnico-científica no IF Goiano.

IDENTIFICAÇÃO DA PRODUÇÃO TÉCNICO-CIENTÍFICA

- | | |
|--|---|
| <input type="checkbox"/> Tese (doutorado) | <input type="checkbox"/> Artigo científico |
| <input type="checkbox"/> Dissertação (mestrado) | <input type="checkbox"/> Capítulo de livro |
| <input type="checkbox"/> Monografia (especialização) | <input type="checkbox"/> Livro |
| <input checked="" type="checkbox"/> TCC (graduação) | <input type="checkbox"/> Trabalho apresentado em evento |

Produto técnico e educacional - Tipo:

Nome completo do autor:

Vitor Ramos Machado, Rafael de Jesus, Bruno Vieira Alves

Matrícula:

[2019108202640] 206, 273 e 117

Título do trabalho:

Desenvolvimento do Robô SCARA para Fins Didáticos e Aplicações Educacionais na Engenharia Elétrica.

RESTRIÇÕES DE ACESSO AO DOCUMENTO

Documento confidencial: Não Sim, justifique:

Informe a data que poderá ser disponibilizado no RIIF Goiano: 16 /02 /2024

O documento está sujeito a registro de patente? Sim Não

O documento pode vir a ser publicado como livro? Sim Não

DECLARAÇÃO DE DISTRIBUIÇÃO NÃO-EXCLUSIVA

O(a) referido(a) autor(a) declara:

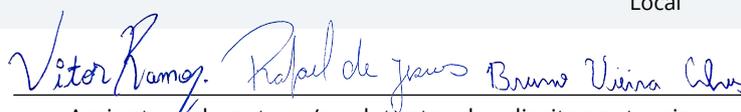
- Que o documento é seu trabalho original, detém os direitos autorais da produção técnico-científica e não infringe os direitos de qualquer outra pessoa ou entidade;
- Que obteve autorização de quaisquer materiais inclusos no documento do qual não detém os direitos de autoria, para conceder ao Instituto Federal de Educação, Ciência e Tecnologia Goiano os direitos requeridos e que este material cujos direitos autorais são de terceiros, estão claramente identificados e reconhecidos no texto ou conteúdo do documento entregue;
- Que cumpriu quaisquer obrigações exigidas por contrato ou acordo, caso o documento entregue seja baseado em trabalho financiado ou apoiado por outra instituição que não o Instituto Federal de Educação, Ciência e Tecnologia Goiano.

Trindade

Local

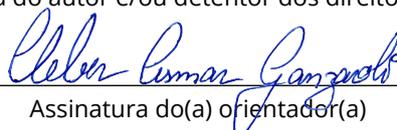
16 /02 /2024

Data

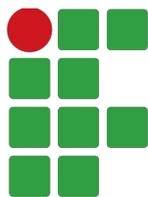


Assinatura do autor e/ou detentor dos direitos autorais

Ciente e de acordo:



Assinatura do(a) orientador(a)



**INSTITUTO FEDERAL
DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA**
Goiano

**DESENVOLVIMENTO DO ROBÔ SCARA PARA FINS
DIDÁTICOS E APLICAÇÕES EDUCACIONAIS NA
ENGENHARIA ELÉTRICA**

Vitor Ramos Machado
Rafael de Jesus
Bruno Vieira Alves

Trabalho de Conclusão do Curso
de Graduação em Engenharia Elé-
trica, orientado pelo Prof. Dr. Cle-
ber Asmar Ganzaroli, aprovado em
08 de fevereiro de 2024.

Instituto Federal Goiano
Campus Trindade
16 de fevereiro de 2024

Sistema desenvolvido pelo ICMC/USP
Dados Internacionais de Catalogação na Publicação (CIP)
Sistema Integrado de Bibliotecas - Instituto Federal Goiano

MM149d Machado, Vitor Ramos; Jesus, Rafael de; Alves, Bruno
Vieira;

Desenvolvimento do Robô SCARA para Fins Didáticos
e Aplicações Educacionais na Engenharia Elétrica /
Vitor Ramos Machado; Rafael de Jesus; Bruno Vieira
Alves; orientador Cleber Asmar Ganzaroli; co-
orientador Geovanne Pereira Furriel. -- Trindade,
2024.

274 p.

TCC (Graduação em Engenharia Elétrica) --
Instituto Federal Goiano, Campus Trindade, 2024.

1. Robô SCARA. 2. Engenharia elétrica. 3. Educação.
4. Construção robótica. 5. Controle de robôs. I.
Ganzaroli, Cleber Asmar, orient. II. Furriel, Geovanne
Pereira, co-orient. III. Título.



SERVIÇO PÚBLICO FEDERAL
MINISTÉRIO DA EDUCAÇÃO
SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA GOIANO

Ata nº 9/2024 - CE-TRI/GE-TRI/CMPTRI/IFGOIANO

ATA DE DEFESA DE TRABALHO DE CURSO

Aos oito (08) dias do mês de fevereiro de 2024, às 19 horas e 30 minutos, reuniu-se a banca examinadora composta pelos docentes: Prof. Me. Robert de Souza Bonuti, Prof. Esp. Roberto Bessa de Araújo, Prof. Dr. Cleber Asmar Ganzaroli, para examinar o Trabalho de Curso intitulado “**DESENVOLVIMENTO DO ROBÔ SCARA PARA FINS DIDÁTICOS E APLICAÇÕES EDUCACIONAIS NA ENGENHARIA ELÉTRICA**” dos estudantes **Vitor Ramos Machado**, Matrícula nº 2019108202640206 , **Rafael de Jesus**, Matrícula nº 2019108202640273 e **Bruno Vieira Alves** , Matrícula nº 2019108202640117 , todos alunos do Curso de Engenharia Elétrica do IF Goiano – Campus Trindade - Goiás. A palavra foi concedida aos estudantes para a apresentação oral do TCC, houve arguição dos candidatos pelos membros da banca examinadora. Após esta etapa, a banca examinadora decidiu pela **APROVAÇÃO** dos estudantes. Ao final da sessão pública de defesa foi lavrada a presente ata que segue assinada pelos membros da Banca Examinadora.

(Assinado Eletronicamente)

Prof. Dr. Cleber Asmar Ganzaroli

Orientador

(Assinado Eletronicamente)

Prof Me. Robert de Souza Bonuti

Membro

(Assinado Eletronicamente)

Prof Esp. Roberto Bessa de Araújo

Membro

Observação: Fazer as devidas correções indicadas pela banca no trabalho enviado por e-mail pelos avaliadores.

() O(a) estudante não compareceu à defesa do TC.

Documento assinado eletronicamente por:

- Robert de Souza Bonuti, PROFESSOR ENS BASICO TECN TECNOLOGICO, em 15/02/2024 16:56:05.
- Cleber Asmar Ganzaroli, PROFESSOR ENS BASICO TECN TECNOLOGICO, em 15/02/2024 16:52:08.

Este documento foi emitido pelo SUAP em 15/02/2024. Para comprovar sua autenticidade, faça a leitura do QRCode ao lado ou acesse <https://suap.ifgoiano.edu.br/autenticar-documento/> e forneça os dados abaixo:

Código Verificador: 572076

Código de Autenticação: fa24207f86



INSTITUTO FEDERAL GOIANO

Campus Trindade

Av. Wilton Monteiro da Rocha, S/N, Setor Cristina II, TRINDADE / GO, CEP 75380-000

(62) 3506-8000



Documento assinado digitalmente

ROBERTO BESSA DE ARAUJO

Data: 15/02/2024 17:46:48-0300

Verifique em <https://validar.iti.gov.br>

“Pense em seus muitos anos de procrastinação; como os deuses repetidamente concederam a você mais períodos de graça, dos quais você não aproveitou. Agora é hora de perceber a natureza do universo a que você pertence, e desse Poder de controle de quem você é descendente; e para entender que seu tempo tem um limite definido. Use-o, então, para avançar sua iluminação; ou ele irá embora e nunca mais estará em seu poder novamente.”

MARCO AURÉLIO
em “Meditações - Livro II”, 180 d.C..

“Por mim se vai ao círculo dolente; por mim se vai ao sofrimento eterno; por mim se vai à perdida gente. Justiça moveu o meu alto fator; criou-me a Suprema Potestade, Suma Sapiência, Primeiro Amor. Antes, foram criadas apenas coisas eternas; eu, eternamente existo. Renunciai as esperanças, vós que entraís.”

DANTE ALIGHIERI
em “A Divina Comédia, Inferno - Canto III”, 1265 – 1321.

“Shout loud moving ahead, ride the horses of justice, virtues of men, yawns. Down and out, losing my head, like a dream you’re returning back from the dead-awake. Shadows will fade someday.”¹

EDU FALASCHI
em “Heroes of Sand - Rebirth”, 2001.

¹Grite alto seguindo em frente, monte nos cavalos da justiça, virtudes dos homens, bocejos. Abatido e exausto, perdendo a cabeça, como um sonho você está voltando dos mortos - acordado. As sombras desaparecerão algum dia.

*A Deus, nossas famílias e amigos que acompanharam o esforço e dificuldades
ultrapassadas. A eles dedicamos este trabalho.*

AGRADECIMENTOS

Vitor Ramos

Agradeço primeiramente a Deus, fonte de toda sabedoria e amor, por Ele e para Ele são todas as coisas. Sua presença foi a luz que guiou cada passo, cada decisão e cada superação neste percurso.

À minha família, meu porto seguro e minha base inabalável, estendo minha mais profunda gratidão. Vocês foram essenciais em cada fase desta caminhada, oferecendo amor, suporte e compreensão incondicionais.

Minha mãe, Luciana Campos, merece o agradecimento especial por sua dedicação incansável e por acreditar em mim incondicionalmente. Sua força, apoio e incentivo foram fundamentais para que eu não desistisse diante dos desafios.

Ao meu pai, Vilmar Ramos, sou eternamente grato pelo suporte, inspiração e pela estrutura que me proporcionou, permitindo que eu me dedicasse inteiramente aos meus estudos e objetivos.

Minha companheira de vida, Larissa Natalia, tem o lugar especial neste agradecimento. Sua presença amorosa, seu incentivo constante e sua capacidade de me motivar nos momentos mais difíceis foram cruciais para a conclusão deste trabalho.

Ao Instituto Federal Goiano, expresso minha gratidão por ter sido acolhido e formado, desde os primeiros passos como aluno de automação industrial até a consolidação da minha formação em engenharia elétrica. Esta instituição foi o cenário de inúmeras aprendizagens e descobertas.

Aos meus amigos e colegas de curso, pela rica troca de conhecimento e experiências compartilhadas. Cada discussão, cada projeto e cada desafio superado juntos contribuíram imensamente para o meu desenvolvimento e para a conclusão deste trabalho.

Ao meu orientador, Professor Cleber Asmar Ganzaroli, pela paciência, sabedoria e orientação precisa. Sua experiência e conhecimento foram pilares que me guiaram nesta pesquisa, incentivando-me a buscar a excelência e a superar os desafios.

E, por fim, expresso minha profunda gratidão a todos que, de diversas formas, contribuíram e me apoiaram ao longo desta jornada.

Rafael de Jesus

É com imensa gratidão que expresso meus agradecimentos aos meus colegas, Vitor Ramos e Bruno Vieira, que compartilharam comigo a jornada desafiadora e recompensadora deste trabalho de conclusão de curso. Cada membro contribuiu de maneira significativa para o sucesso deste projeto, tornando-o uma experiência verdadeiramente colaborativa e enriquecedora.

Primeiramente, dedico meu agradecimento ao nosso orientador, Cleber Asmar Ganzaroli, pela orientação perspicaz, paciência incansável e pela partilha generosa de conhecimentos. Sua contribuição foi essencial para o desenvolvimento deste trabalho, e sou profundamente grato por sua orientação.

À minha esposa Jordana, expresso minha gratidão pela compreensão, apoio incondicional e encorajamento constante. Seu amor e incentivo foram meu alicerce nos momentos desafiadores, e esta conquista é nossa.

Agradecimento especial à minha querida avó, Arlinda, que foi mais do que uma figura familiar. Foi minha fonte constante de apoio e inspiração. Suas histórias, sua sabedoria e seu carinho foram alicerces que sustentaram não apenas minha infância, mas toda a trajetória que me trouxe até aqui.

Agradeço também a minha sogra, Maria da Guia, amigos e família que estiveram ao meu lado, compartilhando conhecimentos, experiências e oferecendo suporte. Suas contribuições foram valiosas, tornando essa jornada mais rica e significativa. Cada um de vocês desempenhou um papel crucial, e sou grato por compartilhar esta conquista com uma rede tão inspiradora. Muito obrigado por fazerem parte desta jornada.

Bruno Vieira

Agradeço primeiramente a Deus, que me guardou, protegeu e me concedeu forças durante esta significativa fase da minha vida, me permitindo concluí-la da melhor forma possível.

Expresso também minha gratidão à minha mãe, Lucicleide Vieira, e ao meu pai, Cleber Alves, pelo amor, carinho e suporte essenciais que forneceram ao longo dessa jornada, desempenhando um papel fundamental na realização deste sonho. Um agradecimento especial é dedicado ao meu irmão, Daniel Vieira, que se destacou como meu ponto de apoio nos momentos críticos em que necessitei de ajuda, mostrando-se

sempre presente.

Por último, estendo minha gratidão a todos os amigos e familiares que, de maneira direta ou indireta, contribuíram para o sucesso desta etapa em minha vida. Cada gesto de apoio e encorajamento foi valioso, e a presença de cada um de vocês fez toda a diferença. Muito obrigado por fazerem parte dessa conquista.

RESUMO

Este trabalho propõe a metodologia para o desenvolvimento do braço robótico SCARA cuja operação esteja adequada para propósitos didáticos e aplicações educacionais na engenharia elétrica. O objetivo deste estudo é oferecer a ferramenta prática para o ensino de robótica, permitindo a integração de conceitos teóricos e práticos. A metodologia proposta abrange desde o projeto mecânico, incluindo a seleção da modelagem e fabricação de peças em 3D, análises estáticas e dinâmicas, além da implementação eletroeletrônica, destacando a seleção de atuadores, sensores e módulos de controle. A simulação e o *software* de controle com interface gráfica são desenvolvidos, possibilitando a execução e programação de tarefas específicas, realçando a funcionalidade e aplicação prática do sistema. Os resultados comprovam a viabilidade da construção do robô SCARA, destacando sua eficácia como recurso didático na engenharia elétrica. A montagem eficaz e a integração do sistema de controle, que emprega conceitos de cinemática direta e inversa, demonstram a utilidade prática do robô, enriquecendo a experiência educativa.

Palavras-Chave: Robô SCARA. Engenharia elétrica. Educação. Construção robótica. Controle de robôs.

DEVELOPMENT OF THE SCARA ROBOT FOR DIDACTIC PURPOSES AND EDUCATIONAL APPLICATIONS IN ELECTRICAL ENGINEERING

ABSTRACT

This work proposes a methodology for the development of the SCARA robotic arm, which is suitable for educational purposes and applications in electrical engineering. The objective of this study is to offer a practical tool for teaching robotics, enabling the integration of theoretical and practical concepts. The proposed methodology encompasses everything from mechanical design, including the selection of modeling and 3D manufacturing of parts, static and dynamic analyses, to the electro-electronic implementation, highlighting the selection of actuators, sensors, and control modules. The simulation and control software with a graphical interface are developed, allowing for the execution and programming of specific tasks, thereby enhancing the functionality and practical application of the system. The results prove the viability of constructing the SCARA robot, emphasizing its effectiveness as a teaching resource in electrical engineering. The efficient assembly and integration of the control system, which employs concepts of forward and inverse kinematics, demonstrate the practical utility of the robot, enriching the educational experience.

Keywords: SCARA robot. Electrical engineering. Education. Robotic Construction. Robot Control.

LISTA DE FIGURAS

	<u>Pág.</u>
2.1 Elementos formadores básicos dos robôs.	11
2.2 Braços com um (à esquerda) e dois graus de liberdade (à direita).	12
2.3 Principais modelos de juntas robóticas.	13
2.4 Morfologia dos manipuladores robóticos.	14
2.5 Vista do plano $x-z$ do primeiro protótipo do robô SCARA.	15
2.6 Volume de trabalho do manipulador SCARA.	17
2.7 Tensão normal máxima no elo de seção transversal retangular.	21
2.8 Tensão de cisalhamento máxima no elo de seção transversal retangular.	22
2.9 Deflexão do elo em balanço com carregamento distribuído.	23
2.10 Transformações entre variáveis de junta e variáveis cartesianas.	27
2.11 Diagrama esquemático da vista superior do robô SCARA.	29
3.1 Fluxograma das etapas de construção do robô.	39
3.2 Fluxograma da etapa de projeto mecânico.	40
3.3 Modelo estático do primeiro e segundo elo no plano $x-z$	43
3.4 Modelo estático do primeiro e segundo elo no plano $y-z$	44
3.5 Diagrama de corpo livre do primeiro elo.	45
3.6 Localização da tensão máxima normal e de cisalhamento no elo engastado.	49
3.7 Diagrama esquemático do robô SCARA.	52
3.8 Fluxograma da etapa de projeto eletroeletrônico.	65
3.9 Fluxograma da etapa de simulação do robô.	69
3.10 Fluxograma da etapa de implementação do <i>software</i> de controle.	71
4.1 Modelo 3D de robô SCARA escolhido para fabricação.	78
4.2 Impressoras utilizadas: (a) Creality Ender 3 e (b) ZMorph VX.	80
4.3 Área de trabalho do fatiador Creality Slicer.	82
4.4 Área de trabalho do fatiador Voxelizer.	82
4.5 Filamentos de PLA: (a) cor amarela, (b) cor preta e (c) cor branca	84
4.6 Fabricação de peças: (a) Acoplador J2 e (b) Braço 01.	86
4.7 Peças 3D finalizadas: (a) Braço 02 e (b) Base do robô.	87
4.8 Hastes de seção circular: (a) haste de $10 \times 400mm$ e (b) haste de $6 \times 150mm$	88
4.9 Sistema de movimentação linear: (a) fuso de avanço de $8 \times 400mm$, (b) acoplador de eixo de $5 \times 8mm$ e (c) polia GT2 com 20 dentes.	89
4.10 Rolamentos utilizados: (a) rolamento linear, (b) rolamento axial e (c) rolamento radial.	89

4.11	Correias GT2: (a) correia $6 \times 400mm$, (b) correia $6 \times 300mm$ e (c) correia $6 \times 200mm$	90
4.12	Acabamento do robô: (a) malha náutica e (b) abraçadeira de <i>nylon</i>	91
4.13	Fluxograma da etapa de montagem mecânica do robô.	92
4.14	Montagem da base: (a) conjunto base e polia e (b) base completa.	93
4.15	Montagem dos elos: (a) vista inferior e (b) vista superior.	94
4.16	Montagem do eixo z : (a) montagem parcial e (b) montagem com o elo.	95
4.17	Montagem do efetuador final: (a) peças utilizadas e (b) efetuador finalizado.	96
4.18	Sistema de transmissão do robô SCARA.	98
4.19	Vista isométrica explodida do robô SCARA.	102
4.20	Curvas de torque das juntas do robô.	106
4.21	Motores utilizados: (a) motores de passo NEMA 17 e (b) servomotor MG996R.	108
4.22	<i>Driver</i> e fim de curso utilizados: (a) vista frontal do <i>driver</i> A4988 e (b) sensor fim de curso.	109
4.23	Fonte de alimentação utilizada: (a) vista frontal da fonte chaveada e (b) fonte com tomada.	111
4.24	Módulos de processamento e controle utilizados: (a) Arduino UNO e (b) CNC <i>Shield</i>	113
4.25	Montagem parcial do robô SCARA: (a) vista com o segundo elo fechado e (b) vista com o segundo elo semiaberto.	114
4.26	Diagrama elétrico de montagem do robô SCARA.	117
4.27	Montagem final do robô SCARA: (a) vista lateral e (b) vista isométrica.	119
4.28	Área de trabalho da simulação do CoppeliaSim.	122
4.29	Diagrama de blocos do sistema de controle do robô SCARA.	126
4.30	Representação da vista superior $x-y$ do robô SCARA.	129
4.31	Fluxograma da programação do sistema de controle.	132
4.32	Programa de interface e controle do robô SCARA.	134
4.33	Inserção de ângulos para uso da cinemática direta.	135
4.34	Inserção de coordenadas cartesianas para uso da cinemática inversa.	137
4.35	Atualização da velocidade e aceleração dos motores.	139
4.36	Salvamento de posições no <i>software</i> de controle do robô.	141
A.1	Prancheta da caixa do Arduino.	150
A.2	Prancheta da tampa da caixa do Arduino.	151
A.3	Prancheta da capa do braço 01.	152
A.4	Prancheta do braço 01.	153
A.5	Prancheta da capa do braço 02.	154
A.6	Prancheta do braço 02.	155
A.7	Prancheta da tampa da base do robô.	156

A.8 Prancheta da base do robô.	157
A.9 Prancheta do acoplador J1.	158
A.10 Prancheta do acoplador J2.	159
A.11 Prancheta do acoplador J3.	160
A.12 Prancheta da tampa superior.	161
A.13 Prancheta da placa inferior do eixo z.	162
A.14 Prancheta da placa superior do eixo z.	163
A.15 Prancheta da plataforma de montagem do eixo z.	164
A.16 Prancheta da abraçadeira de haste lisa.	165
A.17 Prancheta da polia GT2 com 110 dentes.	167
A.18 Prancheta da polia GT2 com 92 dentes.	168
A.19 Prancheta da polia GT2 com 90 dentes.	169
A.20 Prancheta da polia GT2 com 80 dentes.	170
A.21 Prancheta da capa da garra do robô.	172
A.22 Prancheta da extremidade da garra.	173
A.23 Prancheta da garra esquerda.	174
A.24 Prancheta da garra direita.	175
A.25 Prancheta da ligação da garra 01.	176
A.26 Prancheta da ligação da garra 02.	177
A.27 Prancheta do suporte do servomotor.	178
A.28 Prancheta do conector da garra em J3.	179
A.29 Prancheta do deslizador do mecanismo da garra.	180
A.30 Prancheta da extremidade maior da garra esquerda.	181
A.31 Prancheta da extremidade maior da garra direita.	182
A.32 Prancheta da base estrutural do robô.	184
A.33 Prancheta do conjunto base e eixo prismático.	185
A.34 Prancheta do conjunto primeiro e segundo elo.	186
A.35 Prancheta do conjunto efetuator final.	187
A.36 Prancheta do robô SCARA completo.	188

LISTA DE TABELAS

	<u>Pág.</u>
4.1 Lista de materiais mecânicos.	92
4.2 Lista de parafusos para montagem.	96
4.3 Estimativa de vida dos rolamentos - Parte I.	100
4.4 Estimativa de vida dos rolamentos - Parte II.	101
4.5 Substituição de variáveis estáticas.	103
4.6 Tensões normal e de cisalhamento máximos.	103
4.7 Propriedades de massa atribuídas.	104
4.8 Variáveis de velocidade e aceleração atribuídas.	104
4.9 Resultados da análise dinâmica do robô.	105
4.10 Gasto de corrente em cada componente.	110
4.11 Lista de componentes eletroeletrônicos.	114
4.12 Modos de passo do <i>driver</i> A4988.	115
B.1 Lista de peças 3D impressas - Parte I.	191
B.2 Lista de peças 3D impressas - Parte II.	192

LISTA DE ABREVIATURAS E SIGLAS

3D	–	Tridimensional
a.C.	–	Antes de Cristo
ABS	–	Acrilonitrila Butadieno Estireno
ADC	–	<i>Analog-to-Digital Converter</i>
API	–	<i>Application Programming Interface</i>
CA	–	Corrente alternada
CAD	–	<i>Computer Aided Design</i>
CAE	–	<i>Computer Aided Engineering</i>
CC	–	Corrente contínua
CI	–	Circuito integrado
CNC	–	<i>Computer Numeric Control</i>
d.C.	–	Depois de Cristo
DAC	–	<i>Digital-to-Analog Converter</i>
DARPA	–	<i>Defense Advanced Research Projects Agency</i>
EN	–	<i>Enable</i>
ESP	–	Específica
FDM	–	<i>Fused Deposition Modeling</i>
FETs	–	<i>Field-Effect Transistors</i>
<i>G-code</i>	–	Linguagem de programação CNC
GDL	–	Graus de liberdade
GND	–	<i>Ground</i>
GUI	–	<i>Graphical User Interface</i>
I2C	–	<i>Inter-Integrated Circuit</i>
ICSP	–	<i>In-Circuit Serial Programming</i>
IDE	–	<i>Integrated Development Environment</i>
IMP	–	Impressora
ISO	–	<i>International Organization for Standardization</i>
PID	–	Proporcional, Integral e Derivativo
PLA	–	Ácido polilático
PPP	–	Prismático-Prismático-Prismático
PWM	–	<i>Pulse Width Modulation</i>
QTD	–	Quantidade
RIA	–	<i>Robotics Industries Association</i>
RPP	–	Rotativo-Prismático-Prismático
RRP	–	Rotativo-Rotativo-Prismático
RRR	–	Rotativo-Rotativo-Rotativo
SCARA	–	<i>Selective Compliance Assembly Robot Arm</i>
SLA	–	<i>Stereolithography</i>
SLS	–	<i>Selective Laser Sintering</i>
SPI	–	<i>Serial Peripheral Interface</i>
STL	–	<i>Standard Tessellation Language</i>
UART	–	<i>Universal Asynchronous Receiver/Transmitter</i>

UN – Unidade
USB – *Universal Serial Bus*
V-REP – *Virtual Robot Experimentation Platform*
Wi-Fi – *Wireless Fidelity*

LISTA DE SÍMBOLOS

$A_{\ddot{\theta}_1}$	–	Aceleração angular da primeira junta
$A_{\ddot{\theta}_2}$	–	Aceleração angular da segunda junta
A_{r3}	–	Aceleração da junta prismática
a_t	–	Aceleração do motor de passo
\arctan	–	Arco tangente
b_i	–	Largura da secção transversal do elo em balanço
C_{des}	–	Valor unitário aplicado no <i>slider</i>
c	–	Distância entre o eixo neutro à fibra mais afastada deste eixo
C_d	–	Capacidade de carga dinâmica
\cos	–	Cosseno
D_1	–	Distância entre o ponto inicial do primeiro elo e o centro de massa
D_2	–	Distância entre o ponto inicial do segundo elo e o centro de massa
d_m	–	Distância do centro de massa
E	–	Módulo de elasticidade
E_{c1}	–	Energia cinética do primeiro elo
E_{c2}	–	Energia cinética do segundo elo
E_i	–	Módulo de elasticidade do elo
F	–	Função dependente de y
F_{ax}	–	Carga axial necessária ao rolamento
F_e	–	Força necessária para levantar os elos
F_{ra}	–	Carga radial necessária ao rolamento
F_t	–	Força necessária no motor de passo
G	–	Vínculos do sistema
h_i	–	Altura da secção transversal do elo em balanço
h_z	–	Altura do eixo z
I	–	Momento de inércia da secção transversal em relação à linha neutra
I_2	–	Momento de inércia no centro de massa do segundo elo
I_{max}	–	Corrente de operação utilizada pelo motor de passo
I_{yi}	–	Momento de área em relação ao eixo y
L_1	–	Comprimento do primeiro elo
L_2	–	Comprimento do segundo elo
L_{10}	–	Vida em milhões de rotações
L_g	–	Lagrangiano
l_i	–	Comprimento do elo
L_p	–	Laplaciano
$M(x)$	–	Momento fletor ao longo do eixo x
M_a	–	Momento fletor
$M_i(x_i)$	–	Momento fletor dependente do comprimento específico
$ M_i _{max}$	–	Momento fletor máximo em módulo
m_1	–	Centro de massa primeiro elo
m_2	–	Centro de massa segundo elo
m_e	–	Massa dos elos
P_d	–	Carga dinâmica equivalente rolamento
P_i	–	Peso dos motores de passo

q_i	– Carregamento distribuído
Q	– Momento estático de área
R_{cs}	– Resistência de detecção de corrente
$S\theta_1$	– Seno de θ_1
$S\theta_{12}$	– Seno da soma dos ângulos θ_1 e θ_2
\sin	– Seno
t	– Largura da seção transversal na linha neutra
T_i	– Somatórios de todos os conjugados
v	– Velocidade de movimento do elo
V_a	– Tensão de cisalhamento
$V_{\dot{\theta}_1}$	– Velocidade angular da primeira junta
$V_{\dot{\theta}_2}$	– Velocidade angular da segunda junta
$V_i(x_i)$	– Tensão de cisalhamento dependente do comprimento específico
$ V_i _{max}$	– Tensão de cisalhamento máximo em módulo
V_{r1}	– Velocidade da primeira junta do robô
V_{r2}	– Velocidade da segunda junta do robô
V_{r3}	– Velocidade da junta prismática
V_{ref}	– Tensão de referência para ajuste da corrente no <i>driver</i>
V_{cm2}	– Resultante de velocidade para o centro de massa do segundo elo
x_2	– Coordenada de posição da extremidade do segundo elo
x_i	– Seção de comprimento do eixo x referente ao elo do robô
X_{cm2}	– Posição no eixo x do centro de massa do segundo elo
y_0-x_0	– Planos da primeira junta
y_1-x_1	– Planos da segunda junta
y_2	– Coordenada de posição da extremidade do segundo elo
y_2-x_2	– Planos da terceira junta
Y_{cm2}	– Posição no eixo y do centro de massa do segundo elo
z_i	– Seção de comprimento do eixo z referente ao elo do robô
$\ddot{\theta}_1$	– Segunda derivada do ângulo entre o primeiro elo e o eixo x
$\ddot{\theta}_2$	– Segunda derivada do ângulo entre o segundo elo e o eixo x
$\ddot{\theta}_{12}$	– Segunda derivada da soma dos ângulos θ_1 e θ_2
δS	– Variação do funcional S no cálculo variacional
$\dot{y}(x)$	– Primeira derivada
$\dot{\theta}$	– Primeira derivada do ângulo em relação ao tempo
$\dot{\theta}_1$	– Primeira derivada do ângulo entre o primeiro elo e o eixo x
$\dot{\theta}_2$	– Primeira derivada do ângulo entre o segundo elo e o eixo x
$\dot{\theta}_{12}$	– Primeira derivada da soma dos ângulos θ_1 e θ_2
\dot{X}_{cm2}	– Derivada da posição no eixo x do centro de massa do segundo elo
\dot{Y}_{cm2}	– Derivada da posição no eixo y do centro de massa do segundo elo
λ	– Multiplicador de Lagrange
σ_m	– Tensão normal máxima
σ_{mi}	– Tensão normal máxima no elo
τ_m	– Tensão de cisalhamento máxima
τ_{mi}	– Tensão de cisalhamento máxima no elo
τ_n	– Torque necessário ao acionamento

- τ_{nz} – Torques nas juntas
- τ_t – Torque necessário para levantar os elos

SUMÁRIO

Pág.

LISTA DE FIGURAS

LISTA DE TABELAS

CAPÍTULO 1 INTRODUÇÃO	1
CAPÍTULO 2 FUNDAMENTAÇÃO TEÓRICA	9
2.1 Definição de robôs manipuladores	9
2.1.1 Estrutura básica dos robôs	9
2.1.1.1 Graus de liberdade	11
2.1.1.2 Classificação de manipuladores	12
2.1.2 Robô SCARA	14
2.1.2.1 Volume de trabalho	16
2.1.3 Especificações do robô	17
2.1.4 Efetuador final	18
2.1.5 Simuladores robóticos	18
2.1.6 Modelagem e fabricação de peças em 3D	19
2.2 Análise estática	19
2.2.1 Tensões normais e de cisalhamentos máximos	20
2.2.2 Equação de linha elástica	23
2.3 Análise dinâmica	24
2.3.1 Mecânica lagrangiana	25
2.4 Cinemática de manipuladores	27
2.4.1 Cinemática direta	28
2.4.2 Cinemática inversa	30
2.5 Atuadores e sensores	31
2.5.1 Motores de passo	31
2.5.2 Servomotor	32
2.5.3 Sensores de fim de curso	33
2.6 Força e módulos de processamento e controle	33
2.6.1 Fonte de tensão	34
2.6.2 Microcontrolador	34
2.7 Sistemas de transmissões e reduções em esquemas de acionamento	35

2.7.1	Conjunto polia e correia	35
2.8	Rolamentos	36
2.9	Considerações finais	37
CAPÍTULO 3 METODOLOGIA		39
3.1	Estrutura da metodologia	39
3.2	Projeto mecânico	40
3.2.1	Projeto do robô SCARA	41
3.2.2	Cálculos estáticos	43
3.2.3	Cálculos dinâmicos	51
3.2.4	Desenho e processo de fabricação	62
3.2.4.1	Modelagem das peças em 3D	62
3.2.4.2	Impressão das peças em 3D	62
3.3	Projeto eletrônico	64
3.3.1	Atuadores e sensores	65
3.3.2	Módulos de processamento e controle	67
3.3.3	Alimentação do sistema	68
3.4	<i>Software</i> de simulação do robô	69
3.4.1	Simulador do robô SCARA	69
3.5	Implementação do <i>software</i> de controle do robô	70
3.5.1	Cinemática direta e cinemática inversa	71
3.5.2	Controle dinâmico do robô	71
3.6	Considerações finais	72
CAPÍTULO 4 RESULTADOS E DISCUSSÕES		75
4.1	Construção mecânica	75
4.1.1	Seleção e modelagem do manipulador	75
4.1.2	Fabricação da estrutura	79
4.1.2.1	Fatiamento das modelagens	80
4.1.2.2	Impressão 3D das peças	85
4.1.3	Seleção dos componentes mecânicos	87
4.1.4	Processo de montagem do robô	92
4.1.5	Sistema de transmissão	97
4.1.6	Seleção dos rolamentos	98
4.1.6.1	Cálculo de duração dos rolamentos	99
4.1.7	Resolução dos cálculos estáticos	102
4.1.8	Resolução dos cálculos dinâmicos	103
4.1.8.1	Resultados de torques para as juntas	104

4.2	Implementação eletroeletrônica	106
4.2.1	Seleção de componentes	106
4.2.1.1	Seleção de atuadores e sensores	107
4.2.1.2	Seleção de fonte e módulos de processamento	110
4.2.2	Configuração dos <i>drivers</i> de acionamento	115
4.2.2.1	Limitação de corrente do <i>driver</i> A4988	117
4.3	Simulação do robô SCARA	119
4.3.1	Seleção do <i>software</i> de simulação	120
4.3.2	Funcionamento da simulação	121
4.4	Controle robô SCARA	124
4.4.1	Seleção dos <i>softwares</i> de programação	125
4.4.2	Desenvolvimento do sistema de controle	126
4.4.2.1	Modelagem cinemática do robô SCARA	128
4.4.3	Comunicação entre Arduino e Processing	130
4.4.4	Programação de movimentação do robô	131
4.4.5	Interface gráfica de controle	133
4.4.5.1	Modo de gravação do manipulador	140
4.5	Manutenção do robô SCARA	142
4.6	Discussões	143
CAPÍTULO 5 CONCLUSÃO		147
5.1	Trabalhos Futuros	148
APÊNDICE A Vistas 2D e 3D das modelagens utilizadas		149
A.1	Estrutura do robô	150
A.2	Polias e engrenagens do robô	167
A.3	Efetuator final do robô	172
A.4	Estrutura completa do robô	184
APÊNDICE B Dados de impressão 3D		191
APÊNDICE C Códigos de simulação do robô SCARA		195
C.1	Simulação da cinemática direta do robô SCARA	195
C.2	Simulação da cinemática inversa do robô SCARA	197
APÊNDICE D Códigos de controle do robô SCARA		205
D.1	Movimentação e dinâmica do robô SCARA	205
D.2	Interface gráfica de controle do robô SCARA	212

REFERÊNCIAS BIBLIOGRÁFICAS 233

CAPÍTULO 1

INTRODUÇÃO

O constante progresso tecnológico, aliado à crescente demanda por soluções inovadoras, o aumento da produtividade, a melhoria na qualidade do produto e a redução de custos operacionais impulsionam a busca por soluções eficientes no campo da robótica. Os robôs são fundamentais no cumprimento destas tarefas. São empregados para tarefas que demandam precisão, substituindo, assemelhando ou estendendo operações laborais, como montagem de peças, manipulação de objetos, atividades extensivas, perigosas e repetitivas para os humanos (MARTINS, 1993).

O histórico evolutivo da robótica está intrinsecamente vinculado ao desenvolvimento contínuo na capacidade de interação e adaptação dos sistemas robóticos ao ambiente. As primeiras implementações sobre dispositivos autômatos, mencionadas na literatura ocorreram no período da Grécia Antiga por volta do século 350 a.C. Desenvolvido pelo matemático Arquitas de Tarento, traz o pássaro de madeira batizado de O Pombo, propulsionado por vapor e jatos de ar comprimido (SANCHEZ-MARTÍN et al., 2007). Na sequência, destaca-se a construção do triciclo de movimento independente e programável, acionado pela força peso em queda. Este mecanismo criado por Heron de Alexandria, realiza a abertura automática de templos e distribuição automática de vinho, introduzido por volta de 100 d.C. (MERLET, 2000).

Jacques de Vaucanson constrói os primeiros autômatos funcionais, trazendo o boneco flautista capaz de tocar melodias musicais, e a criação do *Digesting Duck*, o pato mecânico com mais de 400 peças móveis com funções gastrointestinais, capaz de grasnar e comer na mão do público, ambos no ano de 1738 (FRYER; MARSHALL, 1979). No ano de 1921, surge pela primeira vez o termo robô, mencionado nas peças teatrais do dramaturgo checo Karel Čapek, se derivando da palavra *robota*, que significa trabalho forçado (BERRY; MARTÍNEZ, 2005). O primeiro robô industrial denominado *Unimate*, é criado por George Devol no ano de 1954 (MORTIMER; ROOKS, 2013). A era da robótica móvel teve início em 1972 com o robô *Shakey*, pioneiro no controle por inteligência artificial, estendendo-se até os dias de atuais com robôs desempenhando funções cada vez mais sofisticadas, inclusive na superfície de outros planetas (TZAFESTAS, 2014).

A fundamentação da teoria de robótica é definida principalmente nos princípios da: i) mecânica, ii) eletrônica, iii) computação e iv) inteligência artificial (D'ABREU; CHELLA, 1999). Inovadores como Engelberger e Devol, com o primeiro robô indus-

trial em 1954, trouxeram impacto significativo para o campo da industrialização, automação e manipulação robótica (KURFESS et al., 2005). Contribuições significativas na inteligência artificial vieram de Turing (1950) e Minsky (1961), estabelecendo a base para robôs autônomos e inteligentes. Avanços na mecânica e na eletrônica contribuíram para o desenvolvimento de robôs mais avançados, enquanto os desafios DARPA¹, a partir de 2003, promoveram robôs adaptáveis a diversos ambientes (THRUN et al., 2006).

Segundo Usategui e León (1986) a robótica engloba áreas de estudo e aplicações distribuídas por quatro setores principais: i) militar, ii) industrial, iii) no campo da saúde e iv) no setor educacional. Cada setor apresenta desafios e características únicas, demonstrando a ampla aplicabilidade da robótica e sua adaptabilidade a diferentes ambientes e necessidades (YANG et al., 2018).

Kotov et al. (2023) desenvolvem o cão robô de combate para aplicações militares, focando na melhoria da segurança e eficácia de operações de reconhecimento. O projeto inicia com o *design* do robô, seguido pela implementação de sistemas de controle e navegação autônoma. Após testes e ajustes, o robô mostra capacidade de operação sobre diversos terrenos, transmitindo dados visuais em tempo real. Os resultados demonstram notável redução de riscos enfrentados pelas forças armadas e aumento da eficiência operacional, alinhando-se às expectativas iniciais do projeto.

Lepri (2013) projeta o robô serpente para operações dentro de ambientes confinados, enfatizando aplicações militares. O projeto começa com a concepção do *design* mecânico e avança para o desenvolvimento do *software* de controle, visando aprimorar a manobrabilidade do robô. Após extensas simulações para ajustar os parâmetros de controle, o sistema é implementado e testado sob condições reais. Os resultados das simulações e testes demonstram a eficácia do robô nos ambientes restritos, atestando sua funcionalidade e adaptabilidade para operações militares específicas.

A demanda por desenvolvimento de robôs avançados é impulsionada pela necessidade de automação e eficiência. Com a crescente complexidade das cadeias de produção, a robótica industrial surge como solução vital para aumentar a produtividade e manter a competitividade (OLIER et al., 1999). A motivação para tal desenvolvimento reside na capacidade dos robôs de executar tarefas com precisão, velocidade e adaptabilidade. Este campo tem sido objeto de extensa pesquisa, como

¹Os Desafios DARPA são competições organizadas pela *Defense Advanced Research Projects Agency* para promover o desenvolvimento de tecnologias inovadoras, com foco particular na robótica e nos sistemas autônomos, enfrentando tarefas complexas e cenários desafiadores.

demonstrado nos estudos de [Gupta et al. \(2021\)](#), que enfoca avanços significativos na robótica e automação para a indústria de manufatura. A aplicação de robôs industriais vai além da manufatura, abrangendo logística, procedimentos cirúrgicos precisos, exploração espacial e operações sob ambientes perigosos, desempenhando funções críticas em cenários complexos e de risco ([GROOVER et al., 1986](#)).

[Muzan et al. \(2012\)](#) implementam o robô industrial para aplicações de pintura, focando no aprimoramento da qualidade e segurança, além da redução no consumo de tinta. O projeto envolve a programação do robô para pintar letras do alfabeto, utilizando *softwares* de *design* e simulação. Os resultados obtidos demonstram aplicação de pintura mais eficiente e segura, atingindo objetivos de economia de tinta e melhoria na qualidade do trabalho executado.

[Summers \(2005\)](#) investiga a precisão e rigidez de robôs industriais para a indústria aeroespacial, desenvolvendo o sistema de posicionamento robótico. O estudo começa com testes de capacidade dos robôs, seguido pelo desenvolvimento e simulação do sistema de controle adaptativo. Embora a meta de precisão absoluta não tenha sido atingida, houve melhorias significativas na precisão com ajustes no envelope de trabalho e calibração. O controle adaptativo demonstrou aumentar a precisão e reduzir a variação posicional, validando a eficácia do sistema desenvolvido.

A robótica médica está transformando a prática clínica, principalmente em cirurgias. De acordo com [Dario et al. \(1996\)](#), a complexidade crescente dos procedimentos cirúrgicos exige robôs cirúrgicos avançados para maior precisão. Estes sistemas permitem cirurgias menos invasivas, melhorando a precisão e acelerando a recuperação dos pacientes ([MAVROIDIS et al., 1998](#)). O objetivo é fornecer aos profissionais de saúde ferramentas que aprimorem suas habilidades e resultados clínicos. Este campo tem sido objeto de extensa pesquisa, como demonstrado no trabalho de [Barria et al. \(2023\)](#), que desenvolveram o *RobHand*, o exoesqueleto robótico para reabilitação neuromotora.

[Christiane \(2009\)](#) projeta o manipulador robótico para cirurgias minimamente invasivas. Este trabalho iniciou-se com revisão detalhada de sistemas existentes, seguida pelo desenvolvimento e teste do novo protótipo. As avaliações e experimentos realizados com o protótipo demonstraram avanços significativos na precisão e na segurança de cirurgias, confirmando a eficácia do *design* proposto.

A robótica na educação está impulsionando mudanças significativas no ensino. [César \(2005\)](#) define que está em expansão e é considerada multidisciplinar, pois é aplicado

o conhecimento de microeletrônica (peças eletrônicas do robô), engenharia mecânica (projeto de peças mecânicas do robô), física cinemática (movimento do robô), matemática (operações quantitativas), inteligência artificial e outras ciências. À medida que cresce a complexidade dos currículos educacionais, a utilização de robôs educacionais avançados torna-se fundamental para aprimorar o aprendizado (ZILLI et al., 2004). Estes sistemas robóticos facilitam o ensino interativo e prático, aumentando o engajamento e a compreensão dos alunos (CÉSAR, 2005). Este campo tem sido objeto de extensa pesquisa, evidenciado por estudos como os de Veiga et al. (2011) sobre robôs de baixo custo para escolas públicas e Junior et al. (2013), focando no acesso à robótica educacional.

Filho et al. (2011) projetam o robô móvel multitarefa de baixo custo para construção de *kits* de robótica educacional. Utilizando materiais recicláveis e acessíveis, visando tornar a robótica mais acessível nas escolas. O robô, baseado no microcontrolador PIC16F628A, integra sensores e atuadores para interagir com o ambiente. O projeto foi validado com sucesso, demonstrando que o robô pode ser a plataforma de ensino eficaz para robótica de baixo custo, adequada para diferentes níveis educacionais.

Chu e Sung (2013) desenvolvem a plataforma robótica educacional, empregando rodas *Mecanum*² para mobilidade omnidirecional. Inicialmente, o modelo do robô foi projetado, seguido pelo desenvolvimento do controlador com base nas simulações do sistema. Este controlador foi posteriormente validado através de implementações práticas no robô. Os resultados das simulações e testes experimentais demonstraram desempenho eficaz, com resposta rápida e alta precisão, confirmando a utilidade da plataforma no contexto educacional.

Os manipuladores robóticos desempenham papel importante nas áreas de estudo e têm aplicações nos diversos setores industriais (EDWARDS, 1984). Conforme apontam os estudos de Gasparetto et al. (2019) e Grau et al. (2020), estes robôs, ao longo das décadas, têm sido utilizados de várias formas na indústria, desde configurações que imitam a mecânica do braço humano até estruturas mais distintas, como o *Selective Compliance Assembly Robot Arm* (SCARA).

Os robôs SCARA, representando inovação significativa na robótica industrial, estão revolucionando a construção e o *design* de robôs (MAKINO, 2014). De acordo com Milutinović e Potkonjak (1990), eles oferecem maior precisão e eficiência nos

²Rodas *Mecanum*, usadas na robótica, permitem movimento omnidirecional ao veículo. Com roletes angulados circundando a roda, elas possibilitam deslocamento em todas as direções, incluindo lateralmente, sem a necessidade de reorientar o robô.

processos de montagem. Este tipo de robô, caracterizado por sua conformidade seletiva e braço de montagem, foi desenvolvido na Universidade de *Yamanashi*, no Japão (GASPARETTO et al., 2019). Seu arranjo é caracterizado por estrutura única que inclui duas juntas rotativas e uma prismática, permitindo operações nos planos paralelos ao eixo principal. Esta configuração otimiza tarefas nas linhas de montagem, pela sua complacência no plano de deslocamento angular e rigidez no deslocamento vertical, facilitando a montagem de componentes e a organização de produtos. (CASTRO, 2019).

Makino et al. (1982) criam o primeiro braço robótico SCARA, marco na robótica industrial, cujo primeiro protótipo foi construído no ano de 1978 e as versões industriais lançadas em 1981. O trabalho abrange desde a concepção inicial do robô até sua aplicação prática na indústria. As inovações introduzidas pelo SCARA, como a conformidade seletiva e eficiência de movimento, demonstraram sua relevância e eficácia, solidificando sua posição na história da robótica industrial e educacional. Seus resultados evidenciaram a importância do SCARA na robótica, marcando avanço significativo na automação industrial.

Shariatee et al. (2014) propõem a concepção do FUM SCARA, o robô SCARA econômico para aplicações industriais, ele é desenvolvido com foco na utilização de componentes locais para reduzir custos. O projeto abrange o *design* de juntas, *links*, controladores e a seleção de componentes mecânicos e elétricos. Com o controlador Proporcional, Integral e Derivativo (PID), testaram a trajetória crítica no espaço de trabalho do robô, alcançando precisão nos movimentos rápidos. Os resultados indicam que o FUM SCARA cumpre as especificações de robôs industriais, destacando-se pelo controle flexível e custo acessível.

Li et al. (2015) desenvolvem o robô SCARA de quatro graus de liberdade com três juntas rotativas e uma prismática para tarefas *pick-and-place*. O trabalho inclui a apresentação da estrutura do robô, modelagem cinemática, planejamento de trajetória e desenvolvimento da interface de controle. A validação experimental do robô demonstra sua eficácia na calibração das peças de trabalho com precisão, destacando sua aplicabilidade prática na automação industrial.

Leal (2022) projeta o manipulador SCARA para auxiliar pessoas com tetraplegia. O projeto visa adaptar o manipulador a cadeiras de rodas, melhorando a autonomia e a qualidade de vida dos usuários. O estudo abrange o *design* do robô, a implementação e testes, com especial atenção à acessibilidade e facilidade. Os resultados revelam progressos em tecnologia assistiva, elevando a autonomia de pessoas com deficiência.

César (2005) afirma que os robôs SCARA são fundamentais na educação e formação na área da robótica, especialmente nos projetos e construções educacionais. Estes robôs proporcionam experiência prática essencial, permitindo aos alunos compreender e aplicar conceitos de engenharia, automação e programação (NKOMO; COLLIER, 2012). Estes conceitos são evidenciados nos trabalhos de Ananias (2022) que constrói o braço robótico colaborativo de baixo custo para aplicações laboratoriais de ensino-aprendizagem no âmbito dos cursos de engenharia, e Tártari et al. (2011) que constroem o braço robótico SCARA de baixo custo para o ensino nas escolas. Tais estudos contribuem para demonstrar a viabilidade e a eficácia dos robôs nos ambientes educacionais.

Mello (2016) desenvolve o robô manipulador SCARA para uso didático. O trabalho incluiu a concepção mecânica, eletrônica e computacional do robô, com ênfase na sua aplicação educacional. O processo de desenvolvimento envolveu análises estáticas e dinâmicas, seleção de componentes e criação de *software* de controle. Os resultados demonstraram a eficácia do robô como ferramenta educacional, enriquecendo a experiência de aprendizado dos alunos na mecatrônica e engenharia robótica.

Castro (2019) projeta o manipulador robótico SCARA para fins educacionais, com foco na sua precisão e repetibilidade. O projeto começou com o *design* e construção do robô, priorizando a análise de erros e controle de desvios. Os resultados confirmaram a eficácia do robô como ferramenta didática valiosa, oferecendo aos estudantes de engenharia experiência prática relevante para a automação industrial.

Pereira (2023) desenvolve o braço robótico SCARA com fins didáticos, utilizando impressão 3D para fabricação de componentes e integrando controladores e motores específicos. O objetivo é criar modelo educativo de robô, acessível e eficiente, para aprimorar o aprendizado em robótica. Os resultados obtidos confirmam a viabilidade do projeto, demonstrando a aplicabilidade prática do manipulador no ambiente educacional, principalmente na facilitação do entendimento de conceitos robóticos e mecânicos.

A literatura apresenta vários trabalhos referentes à importância e aplicação da robótica em contextos militares, industriais, médicos e educacionais. No entanto, observa-se lacunas na literatura quanto à trabalhos que integrem a construção e aplicação prática de robôs, como o SCARA, para ambientes didáticos, visando aprimorar o processo de ensino e aprendizagem na engenharia elétrica. O desenvolvimento do braço robótico SCARA cuja operação esteja adequada para propósitos didáticos, a integração teórica e prática no ensino de conceitos de robótica, eletrônica, controle

e programação, e o foco na preparação de estudantes para as demandas do mercado de trabalho, justifica este trabalho.

Desta forma, é possível construir a hipótese primária deste trabalho: **se** é possível a implementação de *design* eficiente e acessível para o robô SCARA, integrando conceitos de robótica, eletrônica, controle e programação, **então** pode-se contribuir para o ensino prático e interativo na engenharia elétrica. O sucesso desta implementação permitiria não apenas reforçar a compreensão dos estudantes sobre princípios teóricos, mas também proporcionar experiências práticas.

O objetivo geral deste trabalho consiste em desenvolver o braço robótico SCARA cuja operação esteja adequada para propósitos didáticos, atuando como ferramenta prática e acessível para o ensino de conceitos teóricos e práticos de diversos temas, como robótica, eletrônica, controle e programação. Os objetivos específicos são: i) construir o braço robótico, levando em consideração aspectos como o tamanho, peso, resistência, movimentação, graus de liberdade e utilização, ii) desenvolver a montagem mecânica do braço robótico, utilizando ferramentas de *softwares* específicos para modelagem, fabricação e simulação, iii) projetar, especificar e integrar os dispositivos eletrônicos destinados ao acionamento e controle do braço robótico e iv) desenvolver o *software* com interface gráfica de controle do robô, aplicando técnicas de programação e análise cinemática para o controle de movimentação.

A motivação para este trabalho é fundamentada na necessidade de integrar conhecimento teórico e prático no processo de ensino-aprendizagem na área da engenharia elétrica. A construção e utilização prática do braço robótico no ambiente educacional representam estratégia para integrar conhecimentos interdisciplinares, tornando o ensino mais engajador e aplicável. Desta forma, este trabalho visa atender a demanda crescente por métodos de ensino que preparem os estudantes de maneira eficaz para os desafios práticos da engenharia elétrica.

Este trabalho está dividido em cinco capítulos. O Capítulo 2 apresenta a fundamentação teórica dos manipuladores robóticos, incluindo análise estática e dinâmica, cinemática de robôs, e a definição dos componentes mecânicos e eletroeletrônicos. O Capítulo 3 apresenta a metodologia utilizada no trabalho. O Capítulo 4 apresenta os resultados obtidos. O Capítulo 5 finaliza com as conclusões e sugestões para trabalhos futuros.

CAPÍTULO 2

FUNDAMENTAÇÃO TEÓRICA

Neste capítulo, a temática central é a fundamentação teórica sobre robôs manipuladores, elemento-chave para a compreensão dos conceitos e aplicações no campo da robótica. O capítulo é estruturado em várias seções abrangentes, focando em aspectos distintos dos robôs manipuladores, como definições, classificações, especificações técnicas, e aplicações práticas. Esta estruturação permite a exploração aprofundada e sistemática dos temas, contribuindo significativamente para o entendimento holístico do campo da robótica e suas implicações práticas. A abordagem adotada visa não apenas informar, mas também fundamentar teoricamente as aplicações práticas dos robôs manipuladores em diversos contextos, incluindo educacionais e industriais.

2.1 Definição de robôs manipuladores

Segundo Barrientos et al. (2007), a precursora *Robotics Industries Association* (RIA) no ano de 1979 define o robô como o manipulador multifuncional reprogramável, capaz de mover materiais, peças, ferramentas e dispositivos especiais, de acordo com trajetórias variáveis, programado para realizar tarefas diversas. Mataric (2007) entende robôs manipuladores como sistemas robóticos projetados especificamente para a manipulação de objetos e a execução de tarefas em ambientes variados. Groover et al. (1986) oferecem a definição mais técnica, descrevendo robôs manipuladores como sistemas mecânicos com movimentos articulados, controlados automaticamente, reprogramáveis, polivalentes e projetados para mover materiais, peças, ferramentas ou dispositivos especializados.

A primeira tentativa proposta pela RIA de se estabelecer a definição formal curta e concisa deu origem a atual definição da *International Organisation for Standardisation* (ISO) que, segundo a norma ISO 8373 (2012), define o robô industrial como o manipulador controlado automaticamente, reprogramável, multifuncional com três ou mais eixos e que pode ser utilizado em aplicações industriais de automação em posição fixa ou em movimento.

2.1.1 Estrutura básica dos robôs

Segundo a literatura, as definições de manipulador robótico variam conforme os autores e seus países, porém, hodiernamente algumas convenções encontram-se bem disseminadas, como é o caso das morfologias dos manipuladores e de seus componentes. Niku (2010), descreve que o robô manipulador é formado não apenas por sua

estrutura mecânica, mas também pelos seus sistemas de acionamento, transmissão, sensoriamento, controle, entre outros. Estes componentes são caracterizados como: i) elos, ii) juntas, iii) efetuator final, iv) atuadores, v) sensores, vi) controladores, vii) processador e viii) *software*.

De acordo com [Barrientos et al. \(2007\)](#), os elos são elementos rígidos que distanciam e conectam as juntas, podendo apresentar diferentes formatos e configurações construtivas. As juntas correspondem aos elementos que interligam os elos, permitindo o movimento e a transmissão de movimento através de forças e torques. [Groover et al. \(1986\)](#), descreve que o braço é o conjunto de elos unidos por juntas de movimento relativo e punho consiste em várias juntas próximas entre si unidas por elos compactos, que permitem a orientação do órgão terminal nas posições que correspondem à tarefa a ser realizada. O efetuator final é o elemento conectado à última junta ou articulação final do manipulador, consistindo na ferramenta utilizada para realizar trabalho específico, podendo variar de maçaricos de solda, pistolas de pintura, garras ou até mesmo mãos antropomórficas. Este elemento é ligado à articulação final do robô ([NIKU, 2010](#)).

Partindo para o âmbito eletroeletrônico, [Siciliano et al. \(2009\)](#), descreve os atuadores como elementos responsáveis por executar o movimento das articulações. Estes funcionam como músculos para o robô e podem ser do tipo elétrico, hidráulico ou pneumático. O controlador envia os sinais aos atuadores para que eles possam movimentar uma ou mais articulações. Os sensores possibilitam ao robô a interação com o ambiente ou mesmo para obtenção de dados do próprio manipulador, como a posição das juntas, detecção de obstáculos e posicionamento ([NIKU, 2010](#)). Os controladores são utilizados para a implementação do controle de movimentação do robô. Ele recebe informações dos sensores de forma a definir os parâmetros de saída, comandando assim, os atuadores e coordenando os movimentos ([CRAIG, 2013](#)).

Em quesitos de *hardware* e *software*, [Craig \(2013\)](#) caracteriza o processador como o dispositivo utilizado no cálculo dos movimentos, supervisão dos dados dos sensores e das ações do controlador. Este é o elemento integrado ao controlador e ao *software* do robô. Por fim, o *software* fornece as informações ao controlador, determinando como serão executados os cálculos sobre movimentos necessários para cada junta. Ele é o sistema que calcula a movimentação com base na cinemática do robô. Conforme [Mataric \(2007\)](#), o *software* condensa rotinas de programação das funções que podem ser desempenhadas pelo robô em sua operação específica. A [Figura 2.1](#), adaptada de [Craig \(2005\)](#), ilustra alguns dos elementos formadores básicos que compõe o

manipulador robótico.

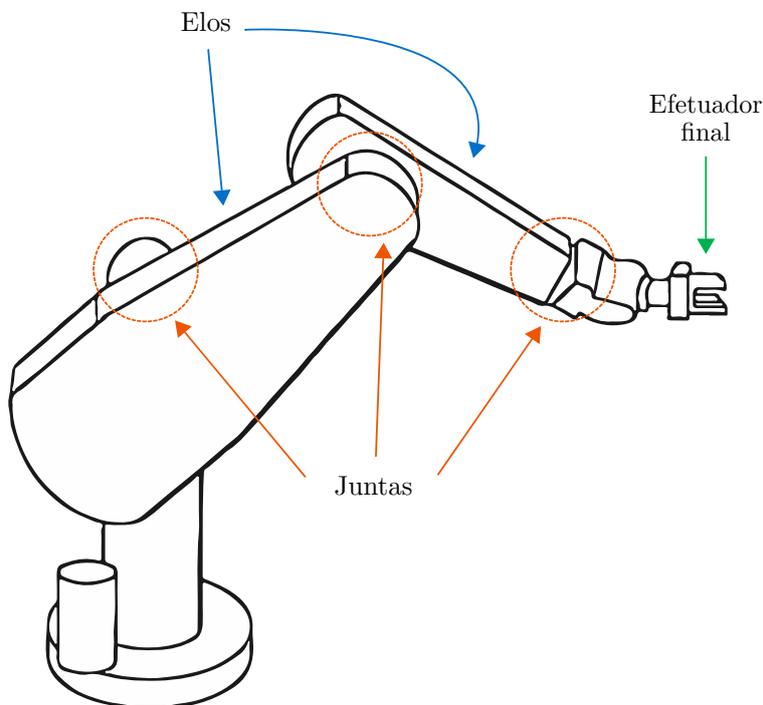


Figura 2.1 - Elementos formadores básicos dos robôs.

A quantidade de elementos presentes no robô como, elos, juntas e o tipo de efetuador varia de acordo com sua aplicação, de forma à alterar o modelo e sua estrutura (SILIANO et al., 2009).

2.1.1.1 Graus de liberdade

Os graus de liberdade (GDL) determinam os movimentos do braço robótico no espaço bidimensional ou tridimensional. Cada junta define um ou dois graus de liberdade, e, assim, o número de GDL do robô é igual à somatória dos graus de liberdade de suas juntas (GROOVER et al., 1986). Segundo Craig (2013), quando o movimento relativo ocorre em um único eixo, a junta tem um grau de liberdade; caso o movimento se dê em mais de um eixo, a junta tem dois graus de liberdade. A Figura 2.2, adaptada de Groover et al. (1986), ilustra estas duas formas de graus de liberdade.

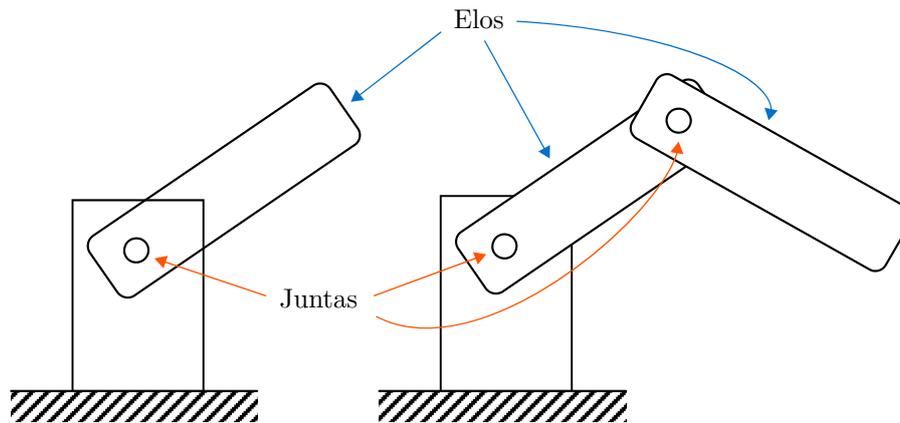


Figura 2.2 - Braços com um (à esquerda) e dois graus de liberdade (à direita).

Segundo Barrientos et al. (2007), quanto maior a quantidade de graus de liberdade, mais complicadas são a cinemática, a dinâmica e o controle do manipulador. O número de GDL do manipulador está associado ao número de variáveis posicionais independentes que permitem definir a posição de todas as partes de forma unívoca.

2.1.1.2 Classificação de manipuladores

De acordo com Barrientos et al. (2007), a classificação de manipuladores pode ser feita por diversas características da estrutura do manipulador, por exemplo, pelo número de graus de liberdade da estrutura cinemática, pela geometria do espaço de trabalho e características do movimento, pelo tipo de controle e acionamento, porém a mais usual é classificar os robôs de acordo com a composição de suas juntas, que dentre as várias formas destacam-se: i) rotativa, ii) prismática e iii) esférica.

A conexão de juntas rotativas permite movimentos de rotação entre dois vínculos. O movimento relativo entre os corpos é restringido de forma unidimensional, possibilitando que os membros da articulação possam rotacionar em relação ao eixo comum aos corpos. Além disso, a junta rotativa gira em torno de linha imaginária fixa, chamada de eixo de rotação. Neste tipo de junta é comum a utilização de acionamentos elétricos por motores de passo ou servomotores (NIKU, 2010). As juntas prismáticas são compostas por mecanismos que deslizam entre si, proporcionando o movimento linear entre dois vínculos. Por fim, a conexão esférica de acordo com Mataric (2007), trata-se da combinação de três juntas rotativas, permitindo a rotação em torno de três eixos simultaneamente. A Figura 2.3, adaptada de Barrientos

et al. (2007), ilustra as principais formas de composições das juntas robóticas.

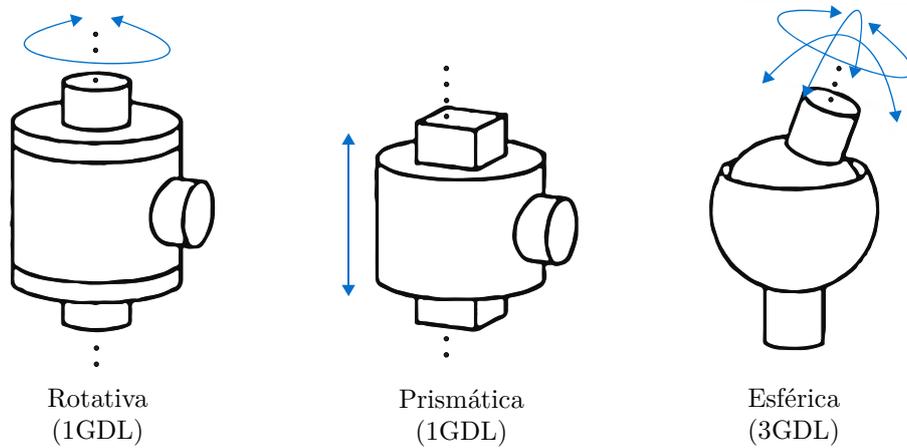


Figura 2.3 - Principais modelos de juntas robóticas.

Conforme Barrientos et al. (2007), a classificação baseada nas juntas é frequentemente utilizada e oferece a visão clara das capacidades e limitações do manipulador. Esta abordagem categoriza os robôs de acordo com os tipos de juntas presentes em sua estrutura, surgindo então o O GDL do robô. O GDL do robô é dado pela soma dos graus de liberdade das articulações que o compõem (GROOVER et al., 1986). Os graus de liberdade do robô são caracterizados como: i) robôs cartesianos, ii) robôs cilíndricos, iii) robôs esféricos, iv) robôs articulados e v) robôs SCARA.

Os robôs cartesianos (PPP), possuem três juntas prismáticas em série, possibilitando movimentos lineares em cada junta. Os robôs cilíndricos (RPP), apresentam uma junta de rotação seguida por duas juntas prismáticas, permitindo movimentos de rotação e translação. Além disso, os robôs esféricos (RRP), combinam juntas de rotação e revolução, proporcionando movimentos em esfera (SICILIANO et al., 2009). Os robôs articulados (RRR), são compostos por juntas de revolução em série, permitindo movimentos de rotação em cada junta, estes também chamados de robôs angulares. Por fim, os robôs SCARA (RRP), esta morfologia possui duas juntas de rotação e uma junta prismática, adequados para aplicações de montagem Makino et al. (1982). A Figura 2.4, adaptada de Barrientos et al. (2007), ilustra os arranjos dos graus de liberdade e das juntas que se configuram como manipuladores robóticos.

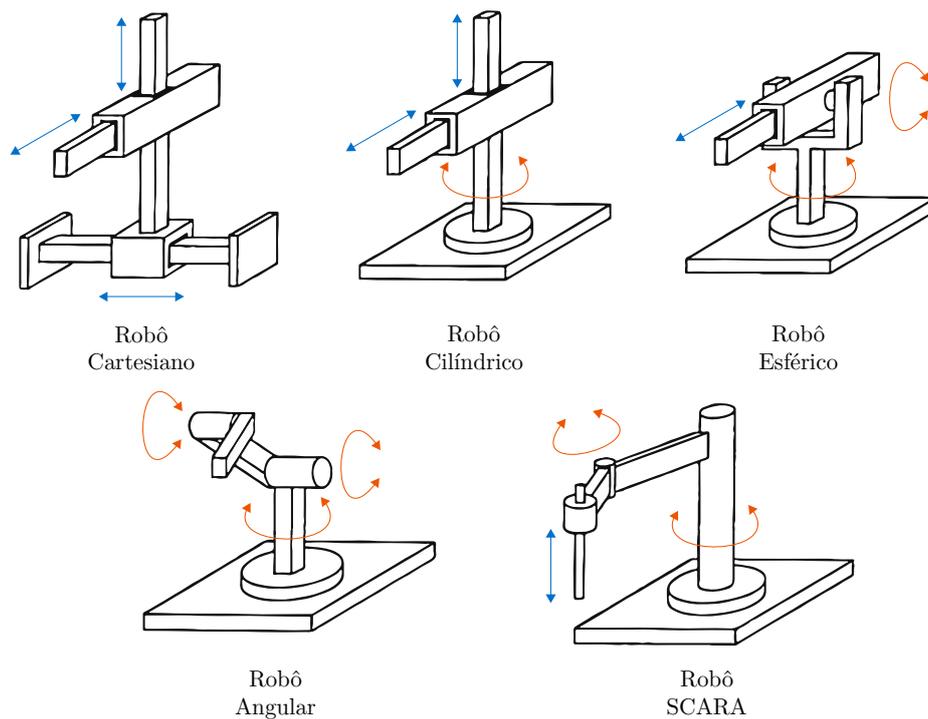


Figura 2.4 - Morfologia dos manipuladores robóticos.

2.1.2 Robô SCARA

Segundo Li et al. (2015), a configuração *Selective Compliance Assembly Robot Arm* (SCARA) foi concebida pelo professor Hiroshi Makino juntamente com sua equipe da Universidade de *Yamanashi* no Japão, sendo seu primeiro modelo construído no ano de 1978. Os robôs SCARA possuem duas ou três articulações de revolução que são paralelas, permitindo que o robô mova-se no plano horizontal, além de articulações prismáticas, permitindo o movimento vertical (NIKU, 2010). Este tipo de configuração torna os adequados em processos de montagem. Eles são mais compatíveis no plano x e y , mas são muito rígidos no plano z (SHARIATEE et al., 2014). A Figura 2.5, adaptada de Makino (2014), ilustra o primeiro protótipo desenvolvido e montado do robô SCARA.

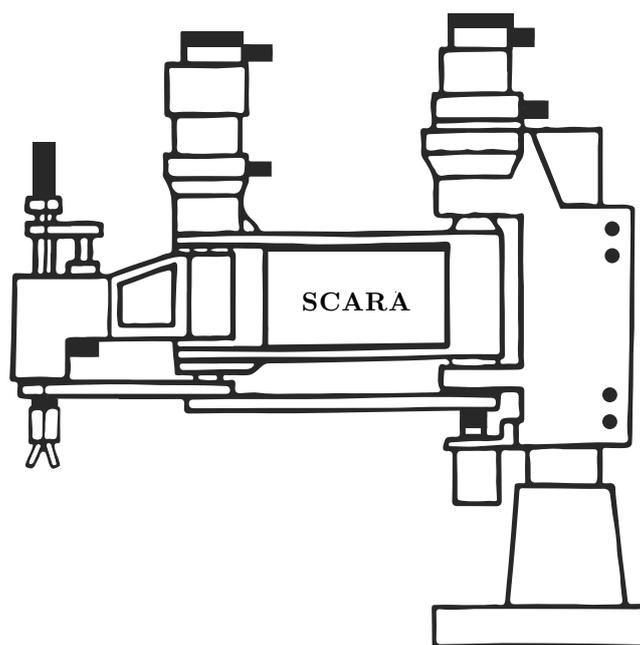


Figura 2.5 - Vista do plano $x-z$ do primeiro protótipo do robô SCARA.

Este arranjo específico apresenta quatro graus de liberdade, sendo que as duas primeiras juntas são articuladas rotativas, cada uma contribuindo com um grau de liberdade. Além disso, há mais dois graus de liberdade atribuídos ao efetuador terminal, que realiza movimentos verticais para cima e para baixo, além da rotação (MAKINO et al., 1982).

De acordo com Yang et al. (2018), o desenvolvimento do SCARA foi impulsionado pela demanda industrial de inserção precisa de componentes, especialmente em tarefas de *peg in hole*¹. Esta configuração é reconhecida por sua alta precisão neste tipo de função. Atualmente, estão sendo exploradas novas aplicações para esta configuração, visando expandir seu uso além das vantagens já conhecidas (GASPARETTO et al., 2019).

O nome SCARA, como explica Makino et al. (1982), deriva de sua complacência seletiva, conceito em robótica que contrasta com a rigidez. A complacência oferece certa flexibilidade ao braço do robô, permitindo que ele se mova sob a aplicação de força e retorne à posição original após a remoção da força (MAKINO, 2014). Esta característica proporciona movimentações suaves e adaptáveis. O SCARA demonstra alta

¹Termo utilizado na robótica e automação industrial para descrever tarefas no qual o objeto (*peg*), geralmente cilíndrico, deve ser inserido precisamente em orifícios ou encaixes (*hole*).

complacência para forças horizontais e rotações no eixo vertical, enquanto apresenta menor complacência para forças verticais e rotações no eixo horizontal (GUPTA et al., 2021).

No que tange à sua aplicação, o robô SCARA apresenta relativa simplicidade em termos de modelagem e programação. Esta configuração contribui para a redução do esforço exigido dos motores, visto que não é requerido o enfrentamento da força gravitacional. Ao invés disso, o robô necessita superar apenas o atrito e a inércia rotacional do manipulador, o que resulta em maior velocidade de operação. De forma complementar, a dispensa de motores de alto torque para o funcionamento do SCARA leva a diminuição nos custos associados à seleção dos atuadores.

2.1.2.1 Volume de trabalho

O volume de trabalho, conforme Groover et al. (1986), é o termo que se refere ao espaço que determinado braço consegue posicionar seu conjunto de elos e juntas. Este volume, em geral, é estabelecido conforme os limites impostos pelo projeto estrutural do braço, ou seja, a configuração física do manipulador robótico, os limites dos movimentos das juntas e o tamanho dos componentes do corpo, braço e punho. Braços robóticos possuem volumes que dependem da geometria e dos limites impostos ao movimento por motivos estruturais ou de controle. Na maior parte deles, o volume é altamente dependente de detalhes construtivos e raramente aparenta ou aproxima-se do volume teórico (KURFESS et al., 2005).

Os volumes de trabalho são medidos em unidades volumétricas, porém isto pouco ou nada contribui na seleção do robô para determinada aplicação. Muito mais importante do que conhecer seu volume é saber se ele consegue ou não atingir pontos à dada distância do seu eixo vertical, por exemplo (GROOVER et al., 1986). Em virtude deste aspecto, os fabricantes de manipuladores robóticos fornecem o volume de trabalho em termos do alcance do braço em um ou mais planos. Por fim, a estrutura geral do robô SCARA também possui facilidade muito grande em se movimentar e alcançar posições e objetos, com precisão (MAKINO et al., 1982). A Figura 2.6, adaptada de Tzafestas (2014), ilustra a representação do volume de trabalho do robô SCARA.

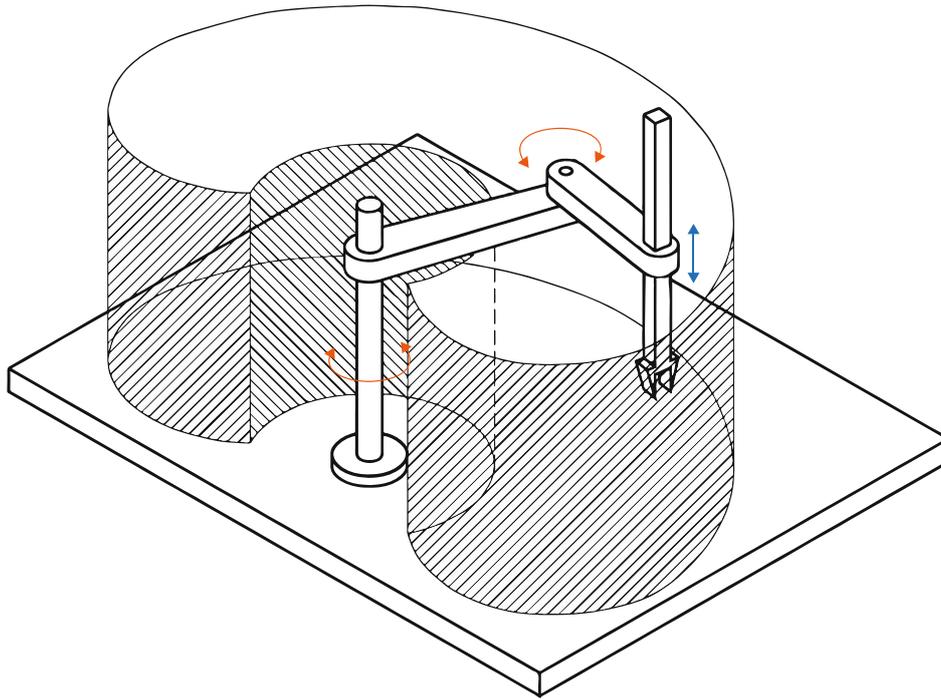


Figura 2.6 - Volume de trabalho do manipulador SCARA.

O volume de trabalho do robô SCARA, descrito por [Craig \(2013\)](#), é definido pela área que o robô pode alcançar com seu efetuador final. Este volume é caracterizado principalmente pela extensão dos braços do robô e pela limitação de seus movimentos verticais. A configuração do SCARA, com seu *design* de braço paralelo, permite grande área de trabalho horizontal, mantendo alta precisão. Estas características tornam os robôs SCARA especialmente adequados para operações que requerem movimentos precisos e repetitivos em planos horizontais, como montagem de circuitos e manipulação de pequenos componentes ([NKOMO; COLLIER, 2012](#)).

2.1.3 Especificações do robô

De acordo com [Niku \(2010\)](#), os robôs são caracterizados por cinco especificações únicas: i) capacidade de carga, ii) precisão, iii) repetibilidade, iv) alcance de trabalho e v) velocidade.

A capacidade de carga, ou carga útil, refere-se ao peso máximo que o robô pode manipular com eficácia, sem comprometer sua utilização. A precisão é a habilidade do robô de alcançar ou posicionar o efetuador final em seu ponto final desejado ([GROOVER et al., 1986](#)). A repetibilidade diz respeito à sua capacidade de atingir à

mesma posição de forma precisa durante diversas execuções. O alcance de trabalho representa a distância máxima que o robô pode atingir dentro de seu volume de trabalho. Por fim, Niku (2010) descreve que a velocidade é a rapidez com que o robô pode mover seu efetuador final entre dois pontos. Craig (2013) destaca a importância destas especificações na determinação da adequação do robô para aplicação específica. Por exemplo, em aplicações industriais, a carga útil e a precisão são fundamentais, enquanto em aplicações de serviço, a velocidade e a capacidade de navegação são mais críticas.

2.1.4 Efetuador final

O efetuador final é a parte do robô que interage diretamente com o ambiente, executando tarefas específicas. No contexto de garras robóticas, estes efetadores são projetados para manipular objetos com precisão e eficácia (CRAIG, 2005).

Existem diversos tipos de garras robóticas, estas são adequadas para diferentes aplicações. Garras paralelas, como descrevem Siciliano et al. (2009), são comuns e consistem em dois dedos que se movem simultaneamente para segurar o objeto, sendo eficazes para pegar e segurar itens com formas regulares. Garras angulares, por outro lado, têm dedos que se fecham em ângulo, úteis para agarrar objetos de formas irregulares. Além disso, garras com dedos adaptativos, que podem mudar sua forma para segurar objetos de várias formas e tamanhos, estão se tornando cada vez mais populares em aplicações avançadas. Estas garras utilizam materiais e mecanismos inovadores para oferecer maior versatilidade e capacidade de manipulação (MATA-RIC, 2007).

2.1.5 Simuladores robóticos

Os simuladores robóticos são *softwares* avançados que proporcionam ambientes virtuais para o teste e a validação de algoritmos, *design* de robôs e sistemas de controle. Estas ferramentas permitem a simulação de cenários complexos e a interação com modelos virtuais de robôs sem os custos e riscos associados aos testes em robôs físicos (SICILIANO et al., 2009).

Corke (2017) salienta a importância dos simuladores robóticos na educação e na pesquisa, proporcionando plataformas acessíveis para estudantes e pesquisadores explorarem conceitos de robótica. Simuladores como o Gazebo oferecem suporte a modelos físicos e ambientais detalhados, possibilitando experiências próximas à realidade, o que é vital para o avanço da pesquisa em robótica.

2.1.6 Modelagem e fabricação de peças em 3D

Conforme definido por Gibson et al. (2015), a modelagem tridimensional (3D) é o processo de criar representações digitais tridimensionais de objetos usando softwares especializados. Esta tecnologia permite aos *designers* e engenheiros visualizar, analisar e refinar suas criações em ambientes virtuais, facilitando iterações² rápidas e eficientes antes da fabricação física. A modelagem 3D não apenas economiza tempo e recursos, mas também oferece flexibilidade sem precedentes no *design* de produtos complexos e personalizados (SACHYANI et al., 2021).

Paralelamente, a fabricação de peças em 3D, principalmente através da impressão 3D, descrita por Lipson e Kurman (2013), é o processo de manufatura aditiva. Este processo constrói objetos físicos camada por camada a partir de modelos digitais. Segundo Barnatt (2016), a impressão 3D permite a produção de geometrias complexas, que muitas vezes são desafiadoras ou impossíveis de serem realizadas por métodos de fabricação tradicionais. As técnicas de impressão 3D incluem *Stereolithography* (SLA), *Fused Deposition Modeling* (FDM) e *Selective Laser Sintering* (SLS), a escolha adequada é embasada na utilização de diferentes materiais e aplicações (EVANS, 2012).

A combinação destas tecnologias tem implicações significativas em setores como a medicina, em que a impressão 3D é usada para criar implantes personalizados e modelos anatômicos para cirurgias, e na aeroespacial, na qual permite a fabricação de componentes leves e resistentes. Além disso, a capacidade de personalização oferecida pela impressão 3D abre novas possibilidades em *design* de produtos, arte e educação (BARNATT, 2016).

2.2 Análise estática

A análise estática do robô é fundamental para a seleção apropriada de materiais e geometria no sistema (BEER et al., 2011). Nesta análise, são calculados diversos parâmetros, incluindo: i) tensões normais, ii) tensões de cisalhamento, iii) deflexão do elo e iv) fadiga. Conforme definido por Makino et al. (1982), em termos de desenvolvimento do robô SCARA, devem ser consideradas as tensões normais e de cisalhamento máximas, juntamente com a deflexão máxima, como critérios aceitáveis. Ao calcular estas variáveis críticas, torna-se possível escolher o material, em con-

²Refere-se ao processo de repetir seqüências de operações ou atividades para aprimorar produtos ou soluções. Na engenharia e *design*, envolve avaliar e modificar protótipos ou conceitos, visando melhorias contínuas até atingir o resultado desejado.

junto com a geometria que satisfaça as exigências específicas do projeto (MAKINO, 2014). Segundo Shariatee et al. (2014), a abordagem de análise estática permite garantir a integridade estrutural do robô SCARA, assegurando que ele possa operar de maneira confiável e segura sob as condições e cargas de trabalho previstas.

2.2.1 Tensões normais e de cisalhamentos máximos

Em elos prismáticos submetidas a cargas transversais, sejam elas pontuais, distribuídas, ou sujeitas a momentos de flexão, ocorrem variações nas intensidades das tensões normais e de cisalhamento ao longo da estrutura (SCHÖN, 2013). Durante o *design* de elos, a avaliação das tensões máximas é utilizada para definir adequadamente o material e a configuração geométrica a serem empregados. Estas tensões máximas devem ser mantidas dentro dos limites de tensão admissíveis estabelecidos pelo projeto, para assegurar a integridade estrutural e a segurança dos elos. (JONES, 2018).

A tensão normal ao longo do eixo x , representada por σ_x , em sua seção varia linearmente com a distância y do eixo neutro³, além disso o maior valor de tensão ocorre nas fibras mais afastadas deste eixo neutro (SCHÖN, 2013). A tensão normal máxima σ_m , conforme definida por Beer et al. (2011) é expressa pela equação (2.1):

$$\sigma_m = \frac{|M|_{max} \cdot c}{I} \quad (2.1)$$

na qual σ_m é a tensão normal máxima, medida em pascal $[Pa]$, $|M|$ é o momento fletor máximo em módulo, medido em newton-metro $[N \cdot m]$, I é o momento de inércia da seção transversal em relação à linha neutra, medido em metros à quarta potência $[m^4]$ e c representa a distância entre o eixo neutro à fibra mais afastada deste eixo, medida em metros $[m]$.

A Figura 2.7, adaptada de Mello (2016), ilustra a localização da tensão normal máxima σ_m para o elo prismático engastado de seção transversal retangular com carregamento distribuído e pontual, ilustrando a tração e a compressão:

³O eixo neutro é o ponto em que a tensão normal τ_x é nula e é conhecido como a linha neutra da seção, ver também em Schön (2013).

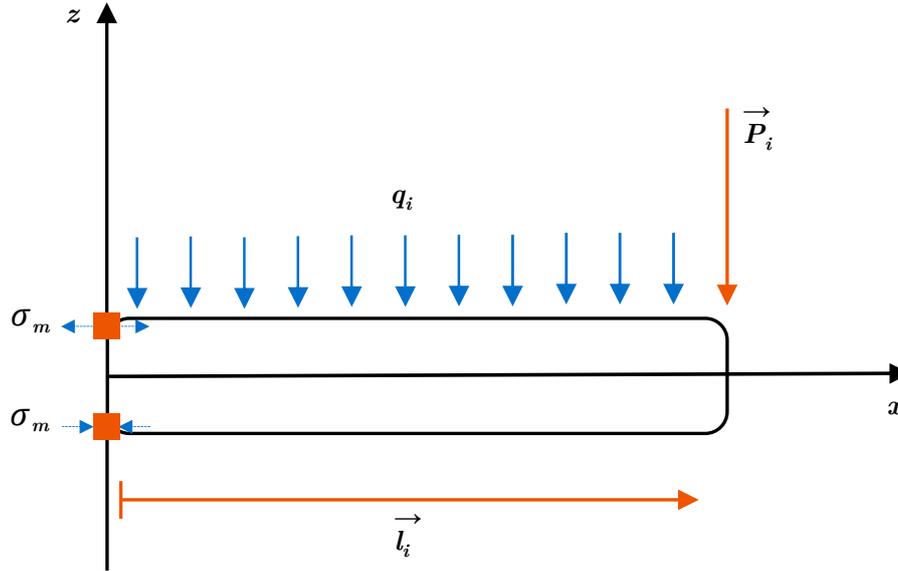


Figura 2.7 - Tensão normal máxima no elo de seção transversal retangular.

no qual σ_m é a tensão normal máxima, medida em pascal $[Pa]$, q_i é o carregamento distribuído, medido em newtons por metro $[N/m]$, P_i é o peso dos motores de passo, medido em newtons $[N]$ e l_i é o comprimento do elo, medido em metros $[m]$.

A tensão de cisalhamento, para a seção retangular, é máxima na linha neutra, conforme observado na equação (2.2) (BEER et al., 2011):

$$\tau_m = \frac{|V|_{max} \cdot Q}{I \cdot t} \quad (2.2)$$

na qual τ_m é a tensão de cisalhamento máxima, medida em pascal $[Pa]$, $|V|_{max}$ corresponde ao módulo da força de cisalhamento máximo, medido em pascal $[Pa]$, Q é o momento estático de área localizada acima ou abaixo da linha neutra, medido em metros cúbicos $[m^3]$, I é o momento de inércia da seção transversal em relação ao eixo neutro, medido em metros à quarta potência $[m^4]$ e t é a largura da seção transversal na linha neutra, medida em metros $[m]$.

De acordo Jones (2018), o momento estático de área Q , pode ser representado pela equação (2.3):

$$Q = \int_0^c y dA \quad (2.3)$$

no qual a integral da função y em relação a área possui o limite inferior de integração no valor de zero e limite superior de integração no valor de c , que representa a distância entre o eixo neutro a fibra mais afastada deste eixo, medido em metros $[m]$.

A Figura 2.8, adaptada de Mello (2016), ilustra a localização da tensão de cisalhamento máxima no elo prismática engastado de seção transversal retangular com carregamento distribuído e pontual:

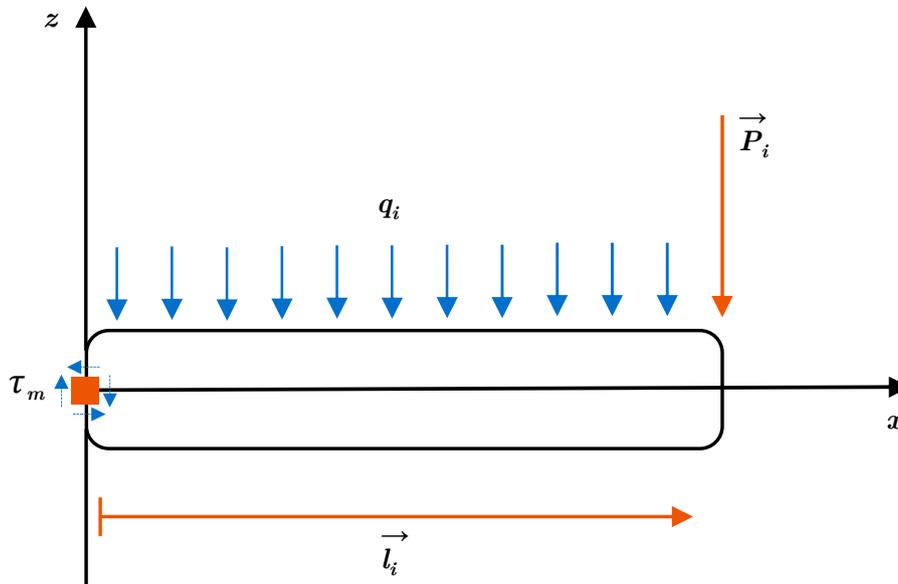


Figura 2.8 - Tensão de cisalhamento máxima no elo de seção transversal retangular.

no qual τ_m é a tensão de cisalhamento máxima, medida em pascal $[Pa]$, q_i é o carregamento distribuído, medido em newtons por metro $[N/m]$, P_i é o peso dos motores de passo, medido em newtons $[N]$ e l_i é o comprimento do elo, medido em metros $[m]$.

Segundo Schön (2013), o momento de inércia I da seção transversal é influenciado pela geometria desta seção, especialmente no caso de seção retangular, em que depende da base e da altura. A escolha adequada dos valores da base e da altura é

crucial para garantir que as tensões normais máximas e as tensões de cisalhamento máximas permaneçam dentro dos limites admissíveis. No entanto, é importante ressaltar que, ao adotar esta abordagem, não se está considerando a deflexão devido aos esforços aplicados à viga. Necessita-se a formulação adicional que se leve em conta esta deflexão. A equação da linha elástica aborda exatamente esta temática.

2.2.2 Equação de linha elástica

Em diversos projetos, a avaliação da deflexão em estruturas mecânicas é essencial para garantir o comportamento desejado do sistema. A deflexão máxima representa a maior distância entre a linha neutra antes e depois da aplicação de cargas na estrutura (JONES, 2018). A Figura 2.9, adaptada de Mello (2016), ilustra a variação desta distância ao longo do comprimento do elo.

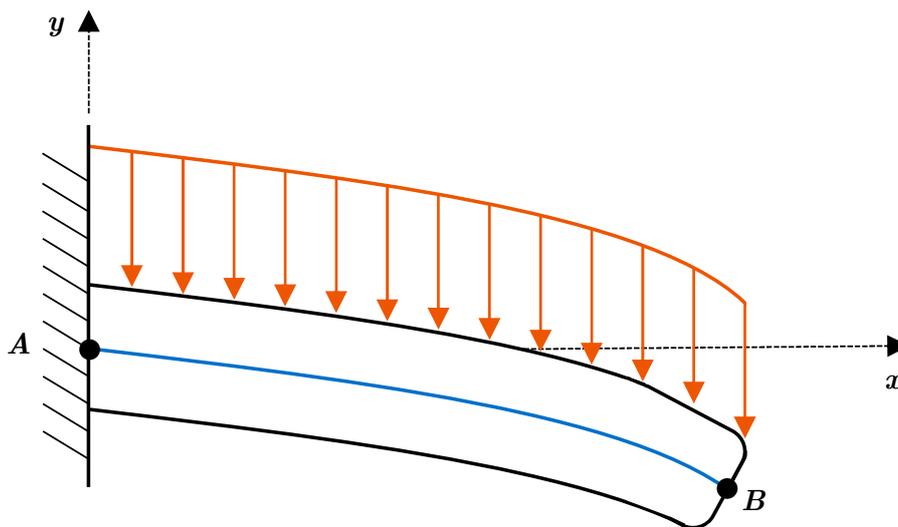


Figura 2.9 - Deflexão do elo em balanço com carregamento distribuído.

A equação que define a deflexão do elo em seu determinado ponto, é denominada como equação diferencial da linha elástica, dentro do regime elástico pode ser expressa conforme definido na expressão (2.4) (BEER et al., 2011):

$$\frac{d^2y}{dx^2} = \frac{M(x)}{E \cdot I} \quad (2.4)$$

na qual d^2y é a segunda derivada da função y , dx^2 indica que a expressão trata-se da segunda derivada de y em relação a x , M é o momento fletor ao longo do eixo x , medido em newton-metro [$N \cdot m$], E é o módulo de elasticidade, medido em pascal [Pa] e I corresponde ao momento de inércia da seção transversal em relação a linha neutra, medido em metros à quarta potência [m^4].

Ao se integrar duas vezes a equação da linha elástica, determina-se a deflexão y em qualquer ponto do elo. As duas constantes de integração, oriundas das integrações sucessivas, são determinadas pelas condições de contorno (SCHÖN, 2013). Para o elo em balanço ilustrado na Figura 2.9, obtêm-se as condições de contorno, conforme representadas nas equações (2.5) e (2.6):

$$\left. \frac{dy}{dx} \right|_A = 0 \quad (2.5)$$

$$y|_A = 0 \quad (2.6)$$

no qual $\frac{dy}{dx}$ representa a derivada da função y em relação à variável x .

Ao examinar a máxima variação de y , é possível calcular a máxima deflexão do elo. Isto possibilita a determinação do material e da geometria a serem utilizados, garantindo que o requisito de máximo valor admissível de deflexão do projeto seja atendido (BEER et al., 2011).

2.3 Análise dinâmica

Conforme definido por Niku (2010), a análise dinâmica do robô se faz necessária para que se possa selecionar os atuadores mais adequados para o sistema. Nesta análise, são calculados os esforços para que o sistema se comporte como esperado, levando em consideração velocidade e aceleração exigidas no projeto.

A aplicação da mecânica newtoniana é comumente utilizada nesta análise, partindo da segunda lei de Newton para calcular os esforços necessários, considerando as restrições do projeto, como acelerações e velocidades (BARRIENTOS et al., 2007). Contudo, à medida que a complexidade do robô aumenta, a mecânica newtoniana pode se tornar dispendiosa, tornando mais vantajoso recorrer a outras ferramentas para modelar o sistema. Desta forma, em literaturas na área de robótica, optam pela modelagem através da mecânica de Lagrange, proporcionando abordagem mais

simplificada para sistemas complexos. Esta técnica oferece a perspectiva eficiente na análise dinâmica de robôs, considerando a interação complexa entre diferentes partes do sistema, o que é particularmente valioso à medida que a complexidade do robô aumenta (NIKU, 2010).

2.3.1 Mecânica lagrangiana

A mecânica lagrangiana emprega as ferramentas do cálculo variacional⁴ em conjunto com o princípio de Hamilton para desenvolver o método de cálculo dos esforços que depende das energias cinéticas e potenciais presentes no sistema (NIKU, 2010). Conforme Neto (2013), o princípio de Hamilton, também conhecido como princípio da mínima ação, é altamente abrangente dentre todos os princípios da física. Ele estabelece que, para cada teoria clássica, há o funcional S , denominado ação, este que satisfaz a equação (2.7):

$$\delta S = 0 \tag{2.7}$$

no qual δS representa a variação do funcional S no cálculo variacional.

No entanto, Leonhard Euler utilizando o princípio de Hamilton, obteve no ano de 1744 a relação expressa pela equação (2.8) (NIKU, 2010):

$$\frac{\partial F}{\partial y} - \frac{d}{dx} \cdot \frac{\partial F}{\partial \dot{y}} = 0 \tag{2.8}$$

no qual F é a função dependente de y , $\frac{\partial F}{\partial y}$ representa a derivada parcial de F em relação a y , $\frac{\partial F}{\partial \dot{y}}$ é a primeira derivada⁵ parcial de F em relação a Y e $\frac{d}{dx}$ indica a derivada em relação a x .

Esta relação apresentada é conhecida como equação de Euler-Lagrange que representa a condição de extremo para o funcional F , em que $F = F(y(x), \dot{y}(x), x)$ e a notação $\dot{y}(x)$ é definida conforme a equação (2.9):

⁴Ramo da matemática focado na determinação de funções que otimizam valores de integrais definidos. Utilizada em física e engenharia para resolver problemas de otimização e formular leis de conservação.

⁵Conforme a notação de Newton, um ponto sobre a variável (\dot{x}) indica a primeira derivada da variável em relação ao tempo.

$$\dot{y}(x) = \frac{d}{dx} \quad (2.9)$$

na qual $\dot{y}(x)$ é a primeira derivada de y em relação a x .

De acordo com Neto (2013), ao se adicionar vínculos ao sistema, a equação de Euler-Lagrange pode ser expressa através da equação (2.10):

$$\frac{\partial}{\partial y} \cdot (F + \lambda G) - \frac{d}{dx} \cdot \frac{\partial}{\partial \dot{y}} \cdot (F + \lambda G) = 0 \quad (2.10)$$

no qual F é a função dependente de y , λ é a quantidade livre chamada de multiplicador de Lagrange e $G = G(y(x), \dot{y}(x), x)$ está associado aos vínculos do sistema.

Segundo Barrientos et al. (2007), os robôs manipuladores apresentam vínculos, que são elos que ligam as juntas. Com isso, utiliza-se a equação de Euler-Lagrange com vínculos para determinar os conjugados, também chamadas de juntas rotacionais e forças, chamadas de juntas prismáticas, isto através do lagrangiano para o sistema expresso pela equação (2.11):

$$L_g = E_c - E_p \quad (2.11)$$

na qual L_g corresponde ao lagrangiano, E_c é a energia cinética, medida em joules [J], E_p é a energia potencial, medida em joules [J].

As equações que definem os conjugados e as forças, deduzidas através da equação de Euler-Lagrange com vínculos, podem ser definidas pelas equações (2.12) e (2.13):

$$T_i = \frac{\partial}{\partial t} \cdot \left(\frac{\partial L_g}{\partial \dot{\theta}_n} \right) - \frac{\partial L_g}{\partial \theta_n} \quad (2.12)$$

$$F_i = \frac{\partial}{\partial t} \cdot \left(\frac{\partial L_g}{\partial \dot{x}_n} \right) - \frac{\partial L_g}{\partial x_n} \quad (2.13)$$

no qual L_g corresponde ao lagrangiano, T_i e F_i são os somatórios de todos os conjugados e forças externas para o movimento de rotação e translação, respectivamente e θ_n e x_n representam o ângulo entre os elos do sistema e o comprimento de cada elo,

respectivamente, em que são medidas em graus $[\circ]$ e em metros $[m]$, respectivamente.

2.4 Cinemática de manipuladores

A cinemática do robô compreende o estudo do movimento do manipulador em relação ao sistema de referência fixo, focando exclusivamente nas relações de posição e suas derivadas temporais, sem considerar os esforços que originam estes movimentos (BARRIENTOS et al., 2007). Conforme descrito por Craig (2005), a cinemática de robôs tem como objetivo descrever o movimento espacial do manipulador como função do tempo, especialmente buscando as relações analíticas de posição e orientação entre a extremidade da cadeia cinemática do manipulador e o sistema de referência estabelecido. A cinemática de manipuladores, portanto, concentra-se em estabelecer as relações entre as variáveis das juntas: i) posição, ii) velocidade e iii) aceleração, além das variáveis do espaço de trabalho do manipulador.

A cinemática do manipulador consiste no estudo do seu movimento, desconsiderando os efeitos dinâmicos das forças e torques que o originam (NIKU, 2010). O problema cinemático se divide em dois na análise de robôs, sendo eles, o problema cinemático direto que busca determinar a posição e orientação da extremidade do manipulador em função das coordenadas de juntas definidas, e no problema cinemático inverso que busca conhecer as variáveis articulares em função da posição e orientação definidas no espaço de trabalho (CRAIG, 2013). A Figura 2.10, adaptada de Groover et al. (1986), ilustra o processo de conversão de coordenadas.

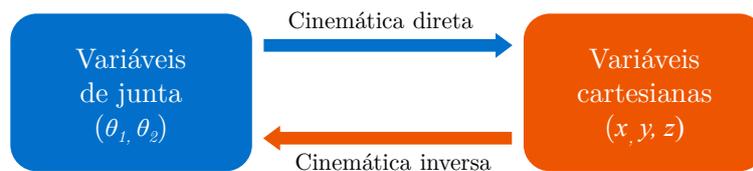


Figura 2.10 - Transformações entre variáveis de junta e variáveis cartesianas.

O cálculo da cinemática, tanto direta quanto inversa, requer o conhecimento do comprimento dos elos com precisão adequada. Fabricantes de manipuladores fornecem não apenas estes comprimentos, como também quaisquer deslocamentos existentes entre juntas, de forma a se poder calcular completamente a posição cartesiana (GROOVER et al., 1986).

2.4.1 Cinemática direta

Segundo Spong et al. (2006), a cinemática direta visa determinar a posição e orientação do efetuador final do robô dentro de seu espaço de trabalho, em relação ao sistema de coordenadas de referência. Este campo de estudo concentra-se em mapear como as variações nos ângulos ou posições das juntas impactam o deslocamento do efetuador final (CRAIG, 2005).

Utilizando as posições das juntas, a localização do efetuador final do robô pode ser determinada por métodos como o geométrico, que é amplamente utilizado devido à sua simplicidade ou através de técnicas de transformações lineares (NIKU, 2010). O método geométrico estabelece a relação entre o sistema de referência da base e a posição e orientação do efetuador final com base em relações geométricas, empregando funções trigonométricas como seno e cosseno em relação aos elos e às coordenadas das juntas do manipulador (CRAIG, 2013). Conforme descrito por Barrientos et al. (2007), embora o método geométrico seja eficaz e de aplicação direta para sistemas robóticos com número restrito de graus de liberdade, sua precisão pode ser limitada por variações dimensionais decorrentes do processo de fabricação. A Figura 2.11, ilustra a vista superior do plano $x-y$ referente ao modelo do robô SCARA com dois graus de liberdade, desconsiderando a junta prismática:

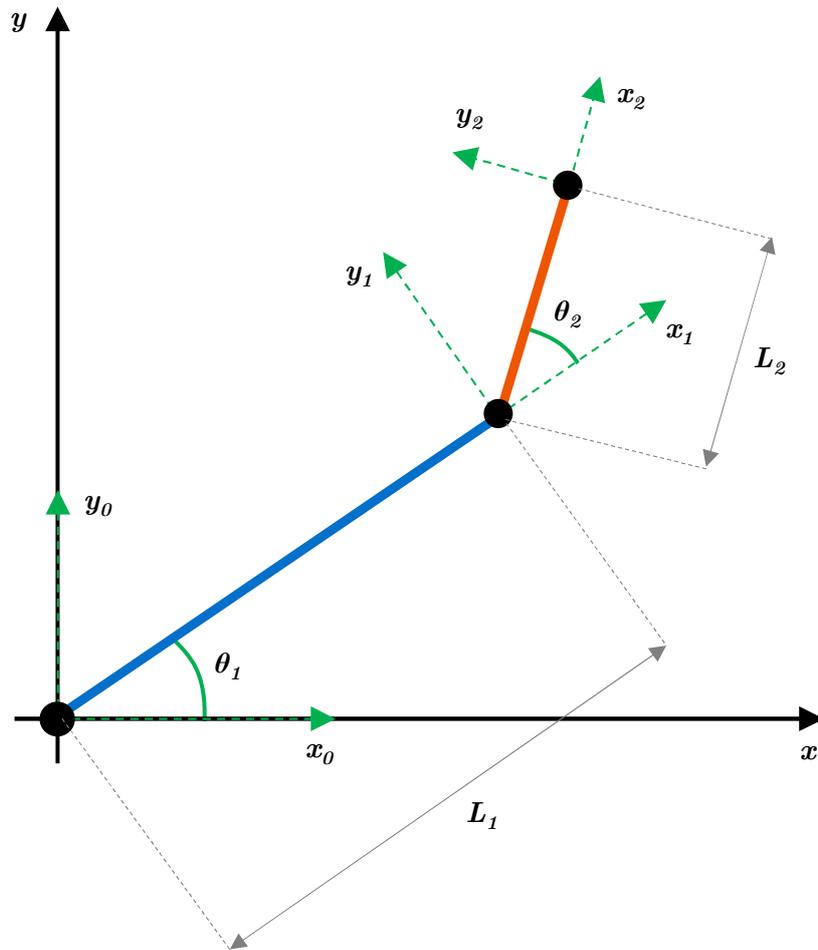


Figura 2.11 - Diagrama esquemático da vista superior do robô SCARA.

no qual as linhas azul e laranja representam o primeiro e segundo elo do robô, respectivamente, y_0-x_0 , y_1-x_1 e y_2-x_2 são os planos da primeira, segunda e terceira junta do manipulador, respectivamente, θ_1 e θ_2 são os ângulos referentes à primeira e segunda junta do robô, respectivamente, ambos medidos em graus $[\circ]$ e L_1 e L_2 correspondem ao comprimento do primeiro e segundo elo, respectivamente, ambos medidos em metros $[m]$.

Desta forma, pode-se calcular facilmente sua cinemática direta utilizando a geometria. O posicionamento da extremidade do braço manipulador (x_2, y_2) conforme a Figura 2.11 em relação ao referencial da base (x_0, y_0) pode ser obtido pelas equações (2.14) e (2.15), deduzidas a partir de relações trigonométricas:

$$x_2 = L_1 \cdot \cos(\theta_1) + L_2 \cdot \cos(\theta_1 + \theta_2) \quad (2.14)$$

$$y_2 = L_1 \cdot \sin(\theta_1) + L_2 \cdot \sin(\theta_1 + \theta_2) \quad (2.15)$$

no qual x_2 e y_2 representam as coordenadas de posição da extremidade do segundo elo, L_1 e L_2 são os comprimentos do primeiro e segundo elo, respectivamente, ambos medidos em metros [m] e θ_1 e θ_2 são os ângulos referentes à primeira e segunda junta do robô, respectivamente, ambos medidos em graus [°]. L_1 e representa o primeiro elo com inclinação θ_1 referente ao plano y_0-x_0 e L_2 representa o segundo elo com inclinação θ_2 referente ao plano y_1-x_1 .

2.4.2 Cinemática inversa

A cinemática inversa envolve a obtenção das coordenadas das juntas a partir da posição e orientação desejadas para o efetuador final do robô (CRAIG, 2005). Este campo de estudo é mais complexo em comparação com a cinemática direta, pois na maioria das vezes implica na resolução de sistemas de equações não-lineares (SPONG et al., 2006). Conforme definido por Niku (2010), na cinemática inversa, deseja-se obter a posição das juntas através da posição do efetuador terminal. Novamente, pode-se utilizar o método geométrico ou por transformações lineares.

Nas equações de cinemática inversa, deve-se observar a possibilidade de estabelecer as coordenadas (x_2, y_2) e, a partir delas, calcular quais ângulos θ_1 e θ_2 que levariam o manipulador até esta condição cartesiana. A partir da Figura 2.11, é possível realizar outra análise trigonométrica que resulta nestas equações (2.16) e (2.17) (CRAIG, 2013):

$$\theta_2 = \pm \arccos \left(\frac{x_2^2 + y_2^2 - L_1^2 - L_2^2}{2 \cdot L_1 \cdot L_2} \right) \quad (2.16)$$

$$\theta_1 = \arctan \left(\frac{y_2}{x_2} \right) - \arctan \left(\frac{L_2 \cdot \sin(\theta_2)}{L_1 + L_2 \cdot \cos(\theta_2)} \right) \quad (2.17)$$

na qual θ_1 e θ_2 são os ângulos referentes à primeira e segunda junta do robô, respectivamente, ambos medidos em graus [°], x_2 e y_2 representam as coordenadas de posição da extremidade do segundo elo, L_1 e L_2 são os comprimentos do primeiro

e segundo elo, respectivamente, ambos medidos em metros [m]. Novamente, a coordenada referente ao eixo z foi omitida da discussão, dada sua relação direta com a junta prismática do robô (BARRIENTOS et al., 2007).

2.5 Atuadores e sensores

Diversas opções de atuadores estão disponíveis no mercado, categorizados pelo tipo específico de acionamento, sendo eles: i) hidráulicos, ii) pneumáticos e iii) elétricos. Estes diferentes métodos de acionamento são amplamente empregados na indústria e em produtos comerciais que requerem atuadores, sendo que cada tipo possui suas próprias vantagens (LEAL, 2022).

Atuadores, como motores de passo e servomotores, descritos por Kenjō e Sugawara (1994) e Bishop (2002), respectivamente, são fundamentais para a execução de movimentos precisos em diversas aplicações, desde a robótica até a automação industrial. Motores de passo são conhecidos por sua capacidade de realizar movimentos angulares precisos em passos fixos, enquanto servomotores, frequentemente combinados com sensores de *feedback*, como *encoders*⁶, permitem o controle refinado de posição e velocidade. Sensores como os de fim de curso, atuam na detecção da posição de componentes mecânicos, contribuindo para a segurança e precisão dos sistemas. Estes sensores podem ser mecânicos, magnéticos ou indutivos (BOLTON, 2021).

2.5.1 Motores de passo

Os motores de passo, descritos por Kenjō e Sugawara (1994), funcionam convertendo pulsos elétricos em movimentos angulares discretos. Esta capacidade de avançar em passos fixos, geralmente medidos em graus, permite o controle extremamente preciso do movimento angular. Esta característica os torna ideais para aplicações como robótica, sistemas de automação e impressoras 3D, na qual o posicionamento exato é fundamental. Além da precisão no movimento, os motores de passo também são valorizados pela sua confiabilidade e repetibilidade. Conforme Hughes e Drury (2013), mesmo sem *feedback* posicional complexo, os motores de passo são capazes de manter sua posição com precisão quando parados, graças ao seu *design* que naturalmente permanece em posição quando energizado.

Barrientos et al. (2007) classificam os motores de passo em três categorias: i) ímã permanente, ii) relutância variável e iii) híbrido. O motor de ímã permanente funciona

⁶Dispositivo sensorial que converte o movimento, seja angular ou linear, em sinais elétricos, geralmente usado para determinar a posição ou velocidade de componentes mecânicos.

através do alinhamento do rotor magnetizado com o campo magnético das bobinas do estator. Já o motor de relutância variável utiliza o rotor ferromagnético que se alinha conforme o campo magnético do estator. O motor híbrido combina aspectos dos dois anteriores, oferecendo precisão e torque superiores. Todos estes motores são caracterizados pela precisão de posicionamento, sendo úteis em manipuladores de pequeno porte.

Os *drivers* de motor de passo atuam como intermediários entre o motor e o controlador, fornecendo a corrente necessária para os motores. Eles controlam a sequência de energização das bobinas do motor, o que é crítico para determinar a direção e o tamanho do passo do motor (HUGHES; DRURY, 2013). A tecnologia de *microstepping* é inovação significativa nos *drivers* de motores de passo. Ela permite que os passos do motor sejam divididos em frações menores, proporcionando o movimento mais suave e maior resolução posicional (KENJŌ; SUGAWARA, 1994).

Niku (2010) destaca que o controle de motores de passo envolve o chaveamento coordenado de suas bobinas, que pode ser feito diretamente por microcontroladores ou através de *drivers*, utilizando circuito integrado (CI) especializado. O microcontrolador, ao gerir diretamente o motor, usa transistores de potência para lidar com a alta corrente das bobinas, ativando-as sequencialmente para controlar a direção e o movimento do motor. Os *drivers* de motor de passo, quando empregados, facilitam este processo. Eles agem como intermediários entre o microcontrolador e o motor, permitindo que o microcontrolador envie pulsos e sinais de direção ao CI, que então controla o motor. Com o CI como *driver*, apenas duas portas de saída do microcontrolador são necessárias para gerenciar eficientemente o deslocamento, a velocidade e a direção do eixo do motor, simplificando o sistema de controle e melhorando a eficácia na operação dos motores de passo (KENJŌ; SUGAWARA, 1994).

2.5.2 Servomotor

Conforme Bishop (2002), os servomotores são tipicamente combinados com sensores de *feedback*, como *encoders*, para formar o sistema de controle em malha fechada⁷. Isto permite que o servomotor execute comandos de posicionamento e velocidade com alta precisão e repetibilidade, tornando-os ideais para aplicações que exigem movimentos precisos e controle dinâmico. Esta precisão e eficiência fazem dos servomotores a escolha preferencial em muitas aplicações industriais e comerciais (HUGHES; DRURY, 2013).

⁷Método de controle no qual a saída é constantemente monitorada e ajustada com base em *feedback* para manter a precisão e estabilidade do sistema.

Os servomotores são dispositivos que podem ser construídos através de diferentes morfologias: i) motores de corrente contínua (CC), ii) corrente alternada (CA), iii) *brushless*⁸ e iv) motores de passo (ROCHA, 2016). Segundo Niku (2010), estes motores são projetados para serem controlados de maneira a atingir a velocidade, torque e ângulo de rotação específicos. Para alcançar estas características, os servomotores são equipados com circuitos de retroalimentação de posição, proporcionando a eles elevada precisão no controle de movimento.

2.5.3 Sensores de fim de curso

De acordo com Bolton (2021), sensores de fim de curso são dispositivos eletromecânicos que operam baseados na atuação física, seja por contato direto ou por proximidade, com o objeto alvo. Estes sensores são amplamente utilizados para fornecer sinais que indicam a posição de componentes de máquinas, como atuadores, válvulas ou portas, garantindo operações seguras e precisas.

Existem vários tipos de sensores de fim de curso, incluindo modelos mecânicos, magnéticos, e indutivos. Os sensores mecânicos, conforme descritos por Butterfield e Szymanski (2018), utilizam atuadores físicos que, ao serem pressionados ou deslocados, fecham ou abrem o circuito elétrico. Estes são simples e confiáveis, mas podem estar sujeitos a desgaste devido ao contato físico.

Sensores magnéticos e indutivos, por outro lado, não requerem contato físico. Os sensores indutivos operam detectando a presença de metais, enquanto sensores magnéticos respondem à presença do campo magnético. Ambos oferecem a vantagem de menor desgaste e maior vida útil comparados aos sensores mecânicos (BOLTON, 2021).

2.6 Força e módulos de processamento e controle

Segundo Sedra e Smith (2014), força, em termos de engenharia elétrica e eletrônica, geralmente se refere à capacidade do sistema ou componente para realizar trabalho ou fornecer energia. Esta energia é utilizada para o funcionamento de diversos sistemas eletrônicos e é fornecida por fontes de tensão, como detalhado por Rashid (2017). Além disso, os módulos de processamento e controle, representados por dispositivos como microcontroladores, são o núcleo de muitos sistemas eletrônicos modernos. Os microcontroladores integram várias funções, como processamento de dados, controle

⁸Refere-se a motores elétricos que operam sem escovas para a comutação de corrente. Requerem menos manutenção do que motores com escovas, devido à ausência do atrito e desgaste associado.

de dispositivos e comunicação, em *chip* único. Esta integração permite o desenvolvimento de sistemas eletrônicos complexos e eficientes (HUGHES, 2016).

2.6.1 Fonte de tensão

As fontes de tensão são dispositivos fundamentais em eletrônica e engenharia elétrica, fornecendo a energia necessária para o funcionamento de circuitos e sistemas eletrônicos. As fontes de tensão podem ser classificadas em dois tipos principais: i) lineares e ii) comutadas (RASHID, 2017).

As fontes lineares, conforme descritas por Sedra e Smith (2014), utilizam transformadores para reduzir a tensão da rede elétrica, seguido pelo retificador para converter a corrente alternada (CA) em corrente contínua (CC). Este tipo de fonte é notável por sua simplicidade e baixo nível de ruído elétrico, sendo ideal para aplicações sensíveis a interferências, como sistemas de áudio e instrumentação de precisão.

Por outro lado, as fontes comutadas, destacadas por Rashid (2017), operam através da conversão da tensão de entrada em corrente de alta frequência antes de transformá-la na tensão desejada. Este processo permite que as fontes comutadas sejam mais eficientes e compactas em comparação com as fontes lineares. Elas são amplamente utilizadas em equipamentos eletrônicos como computadores e televisores, devido à sua eficiência energética e tamanho reduzido (HOROWITZ; HILL, 2015).

2.6.2 Microcontrolador

O microcontrolador, conforme definido por Monk (2016), é o computador compacto em CI único, contendo processador, memória e periféricos de entrada e saída. Estes dispositivos são amplamente utilizados em produtos eletrônicos automatizados, variando de automóveis a aparelhos domésticos. A flexibilidade e a capacidade de controle oferecidas pelos microcontroladores os tornam ideais para inúmeras aplicações em sistemas embarcados. A principal vantagem de consolidar várias funcionalidades no único CI, reside na capacidade de desenvolver sistemas eletrônicos de forma rápida, utilizando o número reduzido de componentes (LIMA; VILLAÇA, 2012).

Os microcontroladores abrigam diversas funcionalidades, como gerador interno de clock, *Analog-to-Digital Converter* (ADC), *Digital-to-Analog Converter* (DAC), temporizador, contador, comparador, saídas *Pulse Width Modulation* (PWM), entre outras (LIMA; VILLAÇA, 2012). Esta diversidade de recursos confere aos microcontroladores versatilidade significativa para atender às exigências de variedades de

aplicações em sistemas embarcados (MONK, 2016).

Em relação ao CNC *Shield*, ele é o exemplo específico de como os microcontroladores podem ser aplicados. Segundo Hughes (2016), o CNC *Shield* é a placa de expansão projetada para ser usada com microcontroladores, especialmente o Arduino, para controlar máquinas de Controle Numérico Computadorizado (CNC). Este *shield* permite ao microcontrolador gerenciar os motores de passo que movem a máquina CNC, proporcionando a precisão necessária para tarefas de corte, gravação ou impressão 3D.

2.7 Sistemas de transmissões e reduções em esquemas de acionamento

Mott et al. (2017), afirma que diversos mecanismos de transmissão de movimento são empregados na robótica em diversas áreas de aplicações, abrangendo desde elementos flexíveis, como polias e correias, até elementos rígidos, como barras e redutores. Mecanismos de redução são amplamente adotados devido à alta velocidade e baixo torque típicos dos motores comerciais, excluindo os motores de passo. Estas reduções desempenham grande papel ao equilibrar as variáveis de velocidade e torque, possibilitando o acionamento e controle eficientes, com rapidez e precisão (CASTRO, 2019).

Conforme definido por Smith e Hashemi (2006), as transmissões, por sua vez, têm como principal objetivo permitir a desvinculação do atuador da posição da junta. Dado que a maior parte da potência dos manipuladores é consumida para movimentar a inércia do braço, a capacidade de deslocar estes elementos para posições estratégicas, distribuindo melhor a massa do manipulador, torna-se importante durante a definição de parâmetros do projeto. Isto não apenas otimiza a eficiência energética, mas também contribui para a redução da potência exigida para a movimentação, proporcionando abordagem fundamental na concepção destes sistemas (CASTRO, 2019).

2.7.1 Conjunto polia e correia

O sistema composto por polias e correias constitui o método eficaz para a transmissão de potência entre dois eixos, especialmente quando há distâncias consideráveis entre eles. As correias, que podem ser planas, trapezoidais, dentadas ou redondas, encaixam-se sobre as polias e transferem energia por meio do movimento rotativo, permitindo assim que o eixo motor distribua força ao eixo movido (PFEIFFER, 2008).

Segundo [Smith e Hashemi \(2006\)](#), a transmissão por correias oferece vantagens significativas devido à sua capacidade de amortecer choques e vibrações e à manutenção relativamente simples que requer. Além disso, este tipo de transmissão é notável pela operação silenciosa e pela eficiência no custo, visto que as correias são componentes de baixo custo em comparação com outras opções de transmissão mecânica, como correntes ou engrenagens ([MOTT et al., 2017](#)).

[Pfeiffer \(2008\)](#) descreve que a eficácia da transmissão por correias é dependente de vários fatores, incluindo o alinhamento correto das polias e a tensão adequada da correia. A tensão incorreta pode levar à desgastes prematuros e a diminuição da eficiência da transmissão, enquanto o alinhamento inadequado pode causar desvios na correia e potencial falha do sistema. Conforme proposto por [Generoso \(2009\)](#), as correias comerciais mais comuns são fabricadas externamente em borracha, garantindo a aderência eficaz entre as superfícies. Internamente, estas correias são reforçadas por cabos de aço, conferindo-lhes maior rigidez e possibilitando a eficiente transmissão de potência entre os elementos. Este arranjo estrutural visa não apenas garantir a confiabilidade da transmissão, mas também assegurar a durabilidade e eficiência do sistema em diferentes aplicações ([CASTRO, 2019](#)).

2.8 Rolamentos

Os rolamentos são componentes mecânicos que limitam o movimento relativo aos desejados, geralmente rotação ou movimento linear. Eles também podem reduzir o atrito entre as partes móveis de máquinas, o que os torna essenciais para a eficiência e a longevidade de muitos sistemas mecânicos ([SHIGLEY; MISCHKE, 1996](#)).

Existem diversos tipos de rolamentos, eles são desenhados para suportar diferentes tipos de cargas e movimentos. Os rolamentos radiais são projetados para suportar cargas perpendiculares ao eixo de rotação. Estes rolamentos, como explicado por [Budynas e Nisbett \(2011\)](#), são utilizados quando a principal carga é radial. Eles podem acomodar desalinhamentos leves e estão disponíveis em variedades de *designs*, como os rolamentos de esferas profundos e os rolamentos de rolos cilíndricos.

Por outro lado, os rolamentos axiais, destacados por [Pfeiffer \(2008\)](#), são otimizados para suportar cargas paralelas ao eixo. Eles são comumente utilizados em aplicações em que as cargas axiais predominam, como em eixos de hélices ou em veículos de carga pesada. Os rolamentos axiais podem ser de esferas ou de rolos, sendo adequados para diferentes níveis de carga e velocidade. Os rolamentos lineares, conforme definido por [Shigley e Mischke \(1996\)](#), são distintos dos rolamentos radiais e axiais,

pois são projetados para movimentos lineares em vez de rotativos. Estes rolamentos são empregados em sistemas que requerem movimentação linear precisa, como em guias de máquinas e sistemas de transporte.

2.9 Considerações finais

O Capítulo 2 aborda de maneira abrangente os robôs manipuladores, enfatizando particularmente no robô SCARA. Este capítulo discutiu conceitos essenciais, incluindo definições, tipos, especificações técnicas e aplicações práticas dos robôs manipuladores. A discussão aprofundada sobre os aspectos técnicos e práticos destes robôs ressalta sua relevância na automação industrial e no contexto educacional. Por fim, o capítulo estabelece a base teórica sólida para os temas subsequentes, preparando o terreno para discussões mais específicas e aplicações práticas dos robôs manipuladores em diversos cenários. O próximo capítulo abordará a metodologia utilizada para o projeto e construção do robô manipulador SCARA.

CAPÍTULO 3

METODOLOGIA

Este capítulo descreve a metodologia proposta na construção do robô SCARA cuja operação esteja adequada para propósitos didáticos. Ela é fundamentada em quatro etapas principais: i) projeto mecânico, ii) projeto eletroeletrônico, iii) simulação do robô e iv) implementação do *software* de controle. Cada etapa é organizada de forma sequencial, incorporando informações e análises de diversas bibliografias da área de engenharia e robótica. Apresenta-se a estrutura detalhada da construção do robô SCARA, oferecendo visão holística das etapas e subetapas envolvidas, realçando sua relevância e aplicabilidade em contextos didáticos.

3.1 Estrutura da metodologia

A metodologia proposta neste estudo é baseada na escolha dos parâmetros para o desenvolvimento do robô. Além disso, incorpora *insights* de literaturas especializadas e artigos sobre o tema para sistematizar todo o processo de construção. A Figura 3.1 representa o fluxograma de desenvolvimento da metodologia aplicada na construção do robô SCARA.

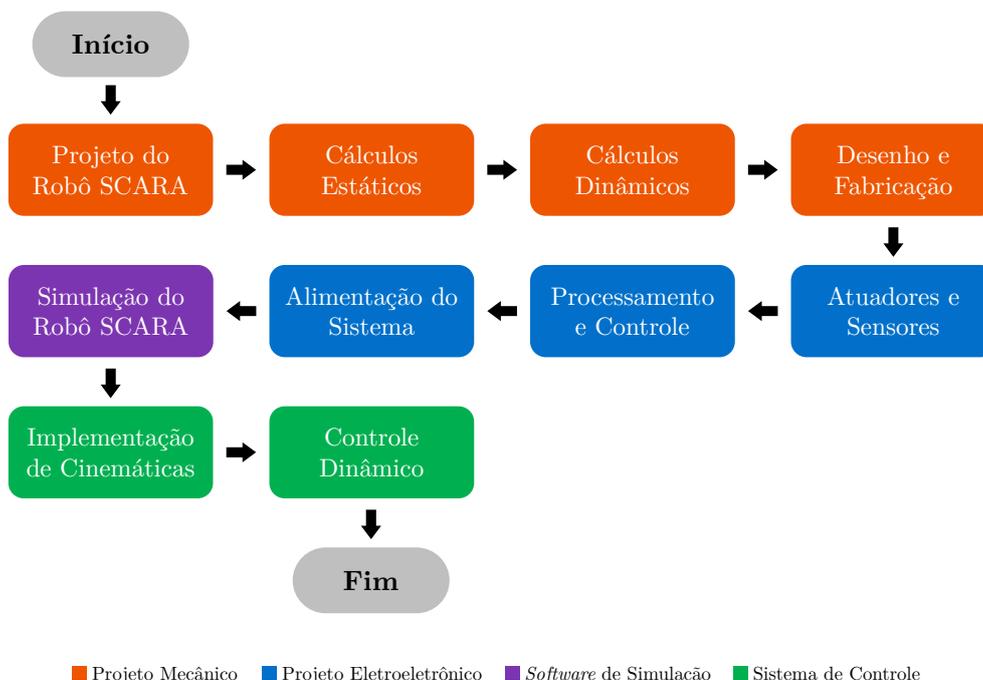


Figura 3.1 - Fluxograma das etapas de construção do robô.

O método aplicado é utilizado para elucidar os intrincados conceitos referentes a cada etapa de todo o processo de construção do robô. Cada uma destas etapas são caracterizadas por cores específicas. Segundo Heller (2022), as cores podem ter impactos significativos no humor, nas decisões humanas, nos sentimentos e até mesmo nas funções cognitivas. Desta forma, foi proposta a distinção destas cores para cada etapa da metodologia, contribuindo para facilitar o entendimento.

A psicologia das cores é aplicada de maneira estratégica. No projeto mecânico, a cor laranja é empregada, simbolizando criatividade, dinamismo e energia, característica vital para a fase inicial de qualquer projeto de engenharia. Para o projeto eletroeletrônico, a cor azul é utilizada, representando calma, tranquilidade e concentração, aspectos aplicados para o trabalho minucioso com componentes eletrônicos. Na etapa do *software* de simulação do robô, a cor roxa, associada à sabedoria e imaginação, é escolhida para incentivar a inovação no desenvolvimento de simulações. Por fim, na implementação do *software* de controle, o verde é selecionado, indicando crescimento, inovação e harmonia, alinhado com a finalização e integração dos sistemas do robô. A seleção desta paleta de cores serve não somente para diferenciar visualmente cada etapa do processo, mas também para sublinhar a natureza intrínseca e os propósitos específicos de cada fase do desenvolvimento.

3.2 Projeto mecânico

A concepção inicial deste estudo visa estabelecer a metodologia de projeto mecânico com quatro subetapas: i) projeto do robô SCARA, ii) cálculos estáticos, iii) cálculos dinâmicos e iv) desenho e processo de fabricação. A Figura 3.2 ilustra o fluxograma de desenvolvimento da etapa de projeto mecânico.



Figura 3.2 - Fluxograma da etapa de projeto mecânico.

A primeira subetapa abrange a análise estrutural, restrições geométricas, definições de parâmetros do robô e graus de liberdade, estabelecendo a base para sua funcionalidade. A segunda subetapa trata dos cálculos estáticos, abordando a deflexão máxima e assegurando que as tensões permaneçam abaixo da tensão de escoamento do material, viabilizando a aplicação da equação da linha elástica. Na terceira subetapa, os cálculos dinâmicos serão desenvolvidos através da mecânica de Lagrange, assumindo acelerações e velocidades máximas para deduzir os torques necessários nas juntas. Por fim, a quarta subetapa, enfoca no desenho e processo de fabricação, ressaltando a necessidade da metodologia integrada que alinhe o *design* e técnicas de fabricação, assegurando que a construção do robô esteja em conformidade com os parâmetros de projeto.

3.2.1 Projeto do robô SCARA

A avaliação detalhada da integridade estrutural é utilizada para garantir que todos os componentes sejam capazes de suportar as cargas e tensões esperadas durante a operação (BEER et al., 2011). Isto envolve verificar a resistência dos materiais utilizados, assegurando que eles não só suportem o estresse mecânico, mas também mantenham a precisão dos movimentos do robô. Além disso, é importante considerar as limitações de espaço e movimento no *design* do robô. Esta atenção ao espaço necessário para a operação e aos limites de movimento dos componentes serve para assegurar que o robô funcione eficientemente e sem interferências. De acordo com Barrientos et al. (2007), a integridade e operação adequada do robô é explorada a partir de dois parâmetros: i) análise estrutural e ii) restrições geométricas.

Conforme destacado por Beer et al. (2011), a análise estrutural possibilita a garantia da resistência e durabilidade das partes constituintes do robô, permitindo assim avaliar a capacidade de cada componente de suportar as cargas operacionais sem sofrer deformações. Este aspecto promove a precisão e a integridade do robô ao longo do tempo. A escolha desta abordagem é motivada pela necessidade de assegurar que todos os elementos do robô sejam robustos e confiáveis, especialmente considerando seu uso em ambientes educativos. Quanto às restrições geométricas, a consideração cuidadosa das limitações de espaço e movimento garantirá que o robô opere de forma eficiente dentro dos parâmetros estabelecidos. Esta abordagem incluirá a definição do alcance máximo do braço do robô, os limites de rotação das juntas e a área necessária para operação segura, garantindo que o robô se adapte bem ao ambiente educativo para o qual foi projetado.

A decisão de escolha da morfologia SCARA é embasada por três razões majoritárias:

i) estrutura simplificada, ii) necessidade de compreensão e iii) precisão e repetibilidade.

Segundo Makino (2014), a estrutura do SCARA é notavelmente simplificada em comparação com outros *designs* robóticos mais complexos. Esta simplicidade torna o modelo ideal para introduzir conceitos fundamentais de robótica. A clareza dos seus movimentos, com articulações bem definidas e movimentos mecânicos observáveis, permite que os estudantes visualizem e compreendam as bases da cinemática robótica. A morfologia SCARA, com seus movimentos distintos e controlados, exemplifica eficientemente os conceitos de controle e automação. Para os alunos, isto significa a oportunidade de ver na prática como os comandos de *software* se traduzem em ações físicas. Em ambientes educacionais, é imperativo que os conceitos sejam demonstrados de forma consistente e repetitiva (TZAFESTAS, 2014). A precisão e a repetibilidade dos movimentos do SCARA garantem que os experimentos e demonstrações sejam confiáveis e que os resultados sejam consistentes.

Foi deliberadamente escolhida a implementação de cinco graus de liberdade. Seus cinco graus de liberdade consistem em: i) movimento de rotação da base, ii) movimentação de elos do robô, iii) rotação da garra, iv) movimento vertical e v) efetuação da garra. A decisão dos cinco graus de liberdade é fundamentada em três necessidades: i) simulação de movimentos complexos, ii) compreensão no controle do robô e iii) versatilidade do robô.

Com cinco graus de liberdade, o robô é capaz de realizar a gama mais ampla e complexa de movimentos (GRAU et al., 2020). Esta capacidade ampliada permite a representação realista das operações robóticas que os alunos podem encontrar em aplicações industriais. Além disso, aumentar os graus de liberdade introduz a camada adicional de complexidade na programação e controle do robô. Isto desafia os alunos a desenvolverem habilidades avançadas, como a coordenação de múltiplos eixos e a resolução de problemas de cinemática inversa. Por fim, o robô torna-se capaz de executar variedades amplas de tarefas. Esta versatilidade demonstra de forma prática a aplicabilidade da robótica em diferentes cenários.

A decisão de incorporar dois elos está intrinsecamente ligada à necessidade de proporcionar movimentos eficientes e versáteis, essencial para realizar tarefas variadas de manipulação e montagem. Primeiramente, a presença de dois elos no robô permite considerável flexibilidade e alcance no plano horizontal referente aos eixos x e y . Estes elos serão projetados para funcionar em harmonia, garantindo que o efetador final do robô possa alcançar posições específicas de forma precisa e controlada.

Por fim, a movimentação no eixo z , permite que o robô SCARA realize movimentos verticais.

3.2.2 Cálculos estáticos

Na seção de cálculos estáticos, apresenta-se as equações necessárias para cálculos de deflexão máxima para os esforços sofridos pelo robô, considerando a exigência de que as tensões máximas sejam inferiores a tensão de escoamento do material que será escolhido, para que a equação da linha elástica possa ser aplicada. Além disso, serão realizadas manipulações em todas as equações, a fim de viabilizar a aplicação prática que esteja adequada à este projeto. As Figuras 3.3 e 3.4, adaptadas de Mello (2016), ilustram o modelo do primeiro e do segundo elo do robô em seu plano $x-z$ e o plano $y-z$ do robô, respectivamente.

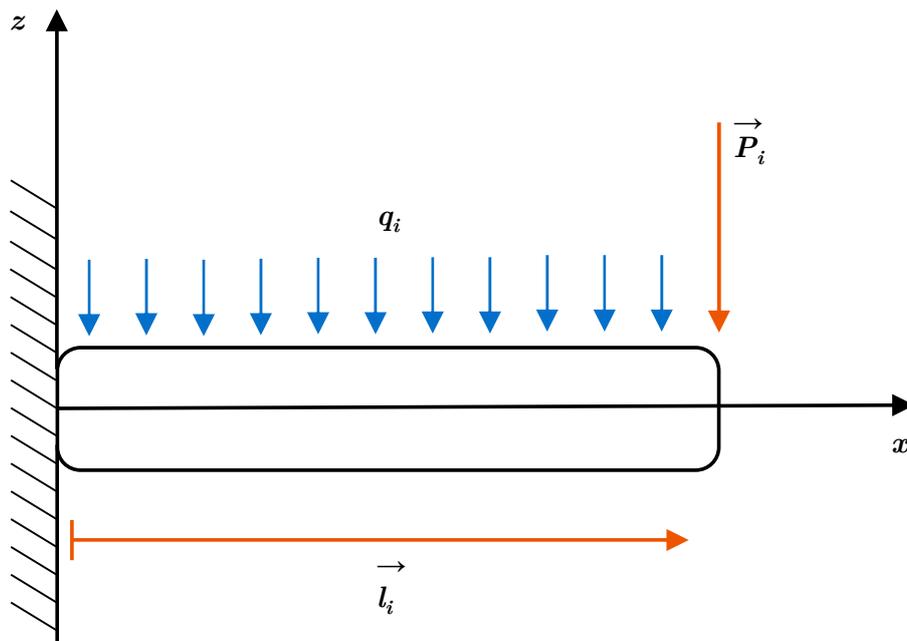


Figura 3.3 - Modelo estático do primeiro e segundo elo no plano $x-z$.

no qual q_i é o carregamento distribuído, medido em newtons por metro $[N/m]$, \vec{P}_i é o peso dos motores de passo ou do órgão terminal, medido em newtons $[N]$ e \vec{l}_i é o comprimento do elo, medido em metros $[m]$.

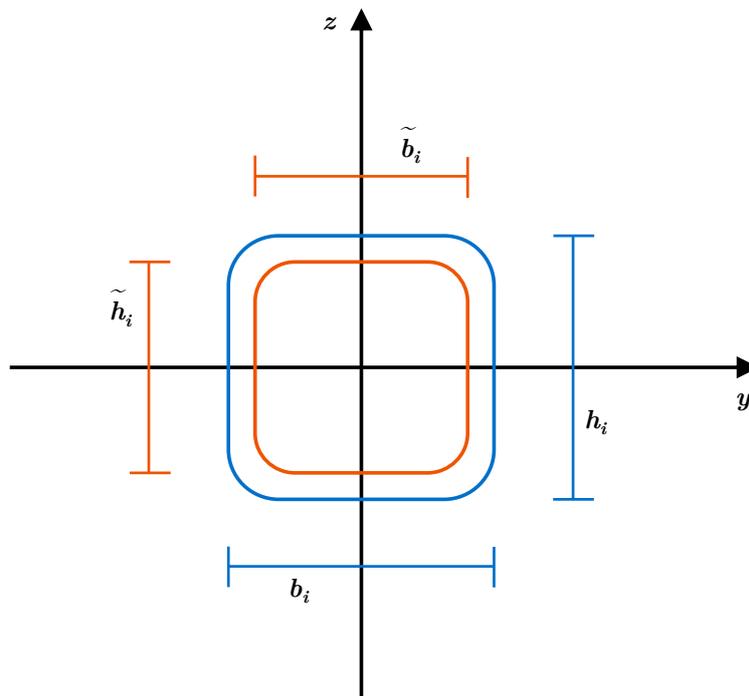


Figura 3.4 - Modelo estático do primeiro e segundo elo no plano y - z .

na qual h_i e b_i respectivamente, são a altura e a largura da secção transversal do elo em balanço, ambos medidos em metros [m]. A variável \tilde{h}_i representa a altura da secção transversal do elo em balanço do lado oposto de h_i e \tilde{b}_i representa a largura da secção transversal do elo em balanço do lado oposto de b_i . O índice i varia de acordo com o elo em que se refere. A Figura 3.5, adaptada de Mello (2016), ilustra o diagrama de corpo livre do elo no plano x - z .

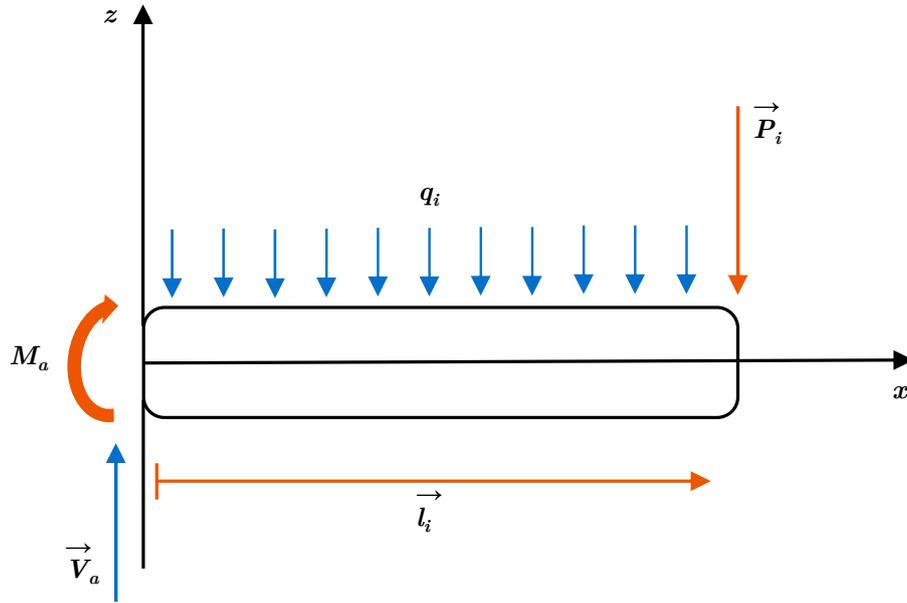


Figura 3.5 - Diagrama de corpo livre do primeiro elo.

no qual q_i é o carregamento distribuído, medido em newtons por metro $[N/m]$, P_i é o peso dos motores de passo ou do órgão terminal, medido em newtons $[N]$, l_i é o comprimento do elo, medido em metros $[m]$, M_a é o momento fletor, medido em newton-metro $[N \cdot m]$ e V_a é a tensão de cisalhamento, medida em pascal $[Pa]$.

Conforme Beer et al. (2011), utiliza-se a segunda lei de Newton para a estática, aplicada à tensão de cisalhamento V_{ai} , conforme a equação (3.1):

$$V_{ai} = P_i + q_i \cdot l_i \quad (3.1)$$

na qual P_i corresponde ao peso dos motores de passo ou do órgão terminal, medido em newtons $[N]$, q_i é o carregamento distribuído, medido em newtons por metro $[N/m]$ e l_i corresponde ao comprimento do elo, medido em metros $[m]$.

Da mesma forma, utiliza-se a segunda lei de Newton para a estática, neste contexto aplicada ao momento fletor, conforme a equação (3.2):

$$M_{ai} = -\frac{q_i \cdot l_i^2}{2} - P_i \cdot l_i \quad (3.2)$$

no qual M_{ai} é o momento fletor, medido em newton-metro $[N \cdot m]$, q_i é o carregamento distribuído, medido em newtons por metro $[N/m]$, l_i corresponde ao comprimento do elo, medido em metros $[m]$ e P_i é o peso dos motores de passo, medido em newtons $[N]$. A adição de i aos valores de tensão de cisalhamento V_{ai} e do momento fletor M_{ai} deve-se ao fato de ambos os valores estarem aplicados à objetos, neste caso, aos elos do robô.

Ao examinar a seção com comprimento x_i , são derivados o momento fletor e a tensão de cisalhamento, através das expressões (3.3) e (3.4), respectivamente. Estes valores são elucidados como funções dependentes deste comprimento específico x_i (BEER et al., 2011):

$$V_i(x_i) = -q_i \cdot x_i + P_i + q_i \cdot l_i \quad (3.3)$$

$$M_i(x_i) = -\frac{q_i}{2} \cdot x_i^2 + (P_i + q_i \cdot l_i) \cdot x_i + \left(-\frac{q_i \cdot l_i^2}{2} - P_i \cdot l_i\right) \quad (3.4)$$

na qual $V_i(x_i)$ e $M_i(x_i)$ correspondem respectivamente, a tensão de cisalhamento e ao momento fletor dependentes do comprimento específico, cuja suas unidades de medida são, pascal $[Pa]$ e newton-metro $[N \cdot m]$, respectivamente, q_i é o carregamento distribuído, medido em newtons por metro $[N/m]$, x_i é a seção de comprimento, medida em metros $[m]$, neste caso $0 \leq x_i \leq l_i$, P_i é o peso dos motores de passo, medido em newtons $[N]$ e l_i corresponde ao comprimento do elo, medido em metros $[m]$.

A tensão de cisalhamento máxima ocorre geralmente na seção transversal mais distante do ponto de apoio e o momento fletor ocorre tipicamente na mesma seção transversal em que a tensão de cisalhamento é máxima (SCHÖN, 2013). Quando o elo está em balanço, este apresenta a tensão de cisalhamento e o momento fletor máximos em $x_i = 0$. Os valores da tensão de cisalhamento e momento fletor podem ser representados conforme as equações (3.5) e (3.6), respectivamente:

$$|V_i|_{max} = P_i + q_i \cdot l_i \quad (3.5)$$

$$|M_i|_{max} = \frac{q_i \cdot l_i^2}{2} + P_i \cdot l_i \quad (3.6)$$

no qual $|V_i|_{max}$ e $|M_i|_{max}$ correspondem respectivamente, a tensão de cisalhamento e ao momento fletor máximos em módulo, cuja suas unidades de medida são, pascal $[Pa]$ e newton-metro $[N \cdot m]$, q_i é o carregamento distribuído, medido em newtons por metro $[N/m]$, l_i é o comprimento do elo, medido em metros $[m]$ e P_i é o peso dos motores de passo, medido em newtons $[N]$.

A equação (3.7) expressa a tensão normal máxima para a flexão (JONES, 2018):

$$\sigma_{mi} = \frac{|M_i|_{max} \cdot c_i}{I_{yi}} \quad (3.7)$$

na qual σ_{mi} é a tensão normal máxima, medida em pascal $[Pa]$, $|M_i|_{max}$ é o momento fletor máximo em módulo, medido em newton-metro $[N \cdot m]$, I_{yi} é o momento de área em relação ao eixo y , medido em metros à quarta potência $[m^4]$ e c_i representa a distância entre a base e a linha neutra do elo, metade da altura da secção transversal do elo em balanço h_i , medida em metros $[m]$.

O momento de área em relação ao eixo y é expresso pela equação (3.8):

$$I_{yi} = \frac{b_i \cdot h_i^3 - \tilde{b}_i \cdot \tilde{h}_i^3}{12} \quad (3.8)$$

no qual I_{yi} é o momento de área em relação ao eixo y , medido em metros à quarta potência $[m^4]$, b_i e h_i respectivamente, são a largura e a altura da secção transversal do elo em balanço, ambos medidos em metros $[m]$, \tilde{b}_i é a largura da secção transversal do lado oposto de b_i e \tilde{h}_i é a altura da secção transversal do lado oposto de h_i , \tilde{b}_i e \tilde{h}_i ambos também são medidos em metros $[m]$.

A partir destas relações, a tensão normal máxima e a tensão de cisalhamento máxima passam a ser calculadas através das equações (3.9) e (3.10):

$$\sigma_{mi} = \frac{3 \cdot l_i \cdot h_i}{b_i \cdot h_i^3 - \tilde{b}_i \cdot \tilde{h}_i^3} \cdot (q_i \cdot l_i + 2 \cdot P_i) \quad (3.9)$$

$$\tau_{mi} = \frac{|V_i|_{max} \cdot Q}{I_y \cdot t_i} \quad (3.10)$$

na qual σ_{mi} é a tensão normal máxima, medida em pascal $[Pa]$, l_i é o comprimento

do elo, medido em metros $[m]$, b_i e h_i respectivamente, são a largura e a altura da secção transversal do elo em balanço, ambos medidos em metros $[m]$, \tilde{b}_i é a largura da secção transversal do lado oposto de b_i e \tilde{h}_i é a altura da secção transversal do lado oposto de h_i , ambos medidos em metros $[m]$, q_i é o carregamento distribuído, medido em newtons por metro $[N/m]$, P_i é o peso dos motores de passo, medido em newton $[N]$, τ_{mi} é a tensão de cisalhamento máxima na estrutura, medida em pascal $[Pa]$, $|V_i|_{max}$ corresponde a tensão de cisalhamento máxima em módulo, medida em pascal $[Pa]$, Q é o momento estático de área, medido em metros cúbicos $[m^3]$, I_{yi} é o momento de área em relação ao eixo y , medido em metros à quarta potência $[m^4]$ e t_i é o comprimento da base da secção transversal do elo b_i , medido em metros $[m]$.

Segundo Jones (2018), o momento estático de área Q , pode ser representado pela equação (3.11):

$$Q = \int_0^{c_i} y dA \quad (3.11)$$

no qual a integral da função y em relação a área do elo possui o limite inferior de integração no valor de zero e limite superior de integração no valor de c_i , que representa a distância entre a base e a linha neutra do elo, medido em metros $[m]$. Para esta integral, y representa o eixo y referente ao elo do robô.

Realizando as manipulações necessárias, obtêm-se a nova expressão do momento estático de área Q , expresso pela equação (3.12):

$$Q = \frac{b_i \cdot h_i^2 - \tilde{b}_i \cdot \tilde{h}_i^2}{8} \quad (3.12)$$

no qual b_i e h_i respectivamente, são a largura e a altura da secção transversal do elo em balanço, ambos medidos em metros $[m]$, \tilde{b}_i é a largura da secção transversal do lado oposto de b_i e \tilde{h}_i é a altura da secção transversal do lado oposto de h_i , ambos também medidos em metros $[m]$.

A partir da equação (3.12), obtêm-se a expressão para a tensão de cisalhamento máxima τ_{mi} , conforme a equação (3.13):

$$\tau_{mi} = \frac{Q \cdot (q_i \cdot l_i + P_i)}{I_{yi} \cdot b_i} \quad (3.13)$$

na qual Q é o momento estático de área, medido em metros cúbicos $[m^3]$, q_i é o carregamento distribuído, medido em newtons por metro $[N/m]$, l_i é o comprimento do elo, medido em metros $[m]$, P_i é o peso dos motores de passo, medido em newtons $[N]$, I_{yi} é o momento de área em relação ao eixo y , medido em metros à quarta potência $[m^4]$ e b_i corresponde a largura da secção transversal do elo em balanço, medido em metros $[m]$.

A representação gráfica em que estão situados os valores de tensão normal máxima σ_{mi} e de tensão de cisalhamento máxima τ_{mi} é ilustrada na Figura 3.6, adaptada de Mello (2016).

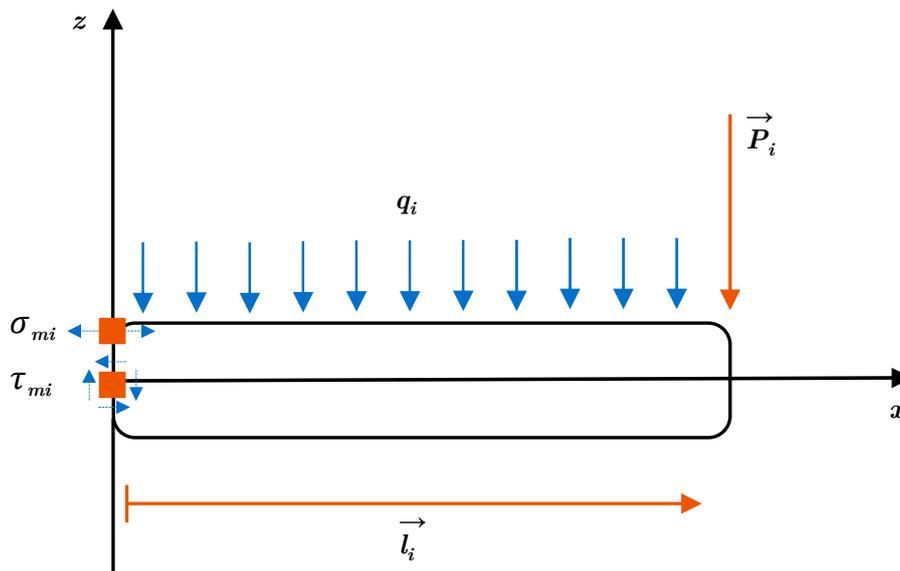


Figura 3.6 - Localização da tensão máxima normal e de cisalhamento no elo engastado.

no qual σ_{mi} é a tensão normal máxima, medida em pascal $[Pa]$, τ_{mi} é a de tensão de cisalhamento máxima na estrutura, medida em pascal $[Pa]$, q_i é o carregamento distribuído, medido em newtons por metro $[N/m]$, P_i é o peso dos motores de passo, medido em newtons $[N]$ e l_i é o comprimento do elo, medido em metros $[m]$.

De acordo com Schön (2013), calcula-se a equação da linha elástica para o elo utilizando a expressão (3.14). Esta equação é aplicada para o elo da Figura 3.3:

$$\frac{d^2 z}{dx^2} = \frac{M_i(x_i)}{E_i \cdot I_{yi}} \quad (3.14)$$

na qual $d^2 z$ é a segunda derivada da função z , dx^2 indica que a expressão trata-se da segunda derivada de z em relação a x , M_i é o momento fletor dependente do comprimento específico x_i , medido em newton-metro [$N \cdot m$], E_i é o módulo de elasticidade do elo, medido em pascal [Pa] e I_{yi} corresponde ao momento de área em relação ao eixo y , medido em metros à quarta potência [m^4]. Para esta expressão, z e x representam o eixo z e o eixo x , respectivamente, ambos referentes ao elo do robô.

Deve-se obter o valor de $z_i = z_i(x_i)$, para isso integra-se duas vezes a equação (3.14), utilizando as condições de contorno para determinar as constantes da equação. A primeira e segunda integração podem ser representadas pelas expressões (3.15) e (3.16), respectivamente:

$$\left. \frac{dz_i}{dx_i} \right|_{x_i=0} = 0 \quad (3.15)$$

$$z_i(x_i)|_{x_i=0} = 0 \quad (3.16)$$

na qual $\frac{dz_i}{dx_i}$ representa a derivada da função z_i em relação à variável x_i . Expressão que define como a função z_i muda à medida que x_i altera. O termo à esquerda da igualdade, $\left. \frac{dz_i}{dx_i} \right|_{x_i=0}$, indica que esta derivada está sendo avaliada no ponto no qual $x_i = 0$. Na expressão $z_i(x_i)|_{x_i=0} = 0$, temos a função z_i avaliada em termos da variável x_i . O segmento $|_{x_i=0}$ indica que estamos interessados no valor da função z_i quando x_i é igual a zero. Para ambas expressões, z_i e x_i representam seções de comprimento do eixo z e o eixo x , respectivamente, ambos referentes ao elo do robô.

Diante de duas integrações e manipulações de equações, resulta-se assim na equação da linha elástica $z_i(x_i)$, que pode ser representada pela equação (3.17):

$$z_i(x_i) = \frac{1}{24 \cdot E_i \cdot I_{yi}} \cdot [-q_i \cdot x_i^4 + 4 \cdot x_i^3 \cdot (P_i + q_i \cdot l_i) - 6x_i^2 \cdot (q_i \cdot l_i^2 + 2 \cdot P_i \cdot l_i)] \quad (3.17)$$

no qual z_i e x_i representam seções de comprimento do eixo z e o eixo x , respecti-

vamente, ambos referentes ao elo do robô e os dois medidos em metros $[m]$, E_i é o módulo de elasticidade do elo, medido em pascal $[Pa]$, I_{yi} é o momento de área em relação ao eixo y , medido em metros à quarta potência $[m^4]$, q_i é o carregamento distribuído, medido em newtons por metro $[N/m]$, P_i é o peso dos motores de passo ou do órgão terminal, medido em newtons $[N]$ e l_i é o comprimento do elo, medido em metros $[m]$. A deflexão máxima, no elo engastado, é obtida se, e somente se, $x_i = l_i$.

3.2.3 Cálculos dinâmicos

A análise dinâmica possibilita a determinação dos torques exigidos em cada junta para promover o movimento do robô SCARA em velocidades e acelerações específicas. Sendo ela responsável pela seleção otimizada dos atuadores necessários para cada junta. No presente projeto optou-se por utilizar a abordagem de energias para a obtenção do modelo dinâmico do manipulador, para isso, foram assumidas as seguintes hipóteses e aproximações: i) centro de massa de cada elo alinhado com seu comprimento, ii) energia potencial nula para o movimento planar do SCARA e iii) atrito nulo nas juntas de rotação, sendo compensado no fator de segurança. A Figura 3.7, ilustra o diagrama esquemático do robô utilizada para os cálculos dinâmicos, de modo a representar e identificar suas variáveis.

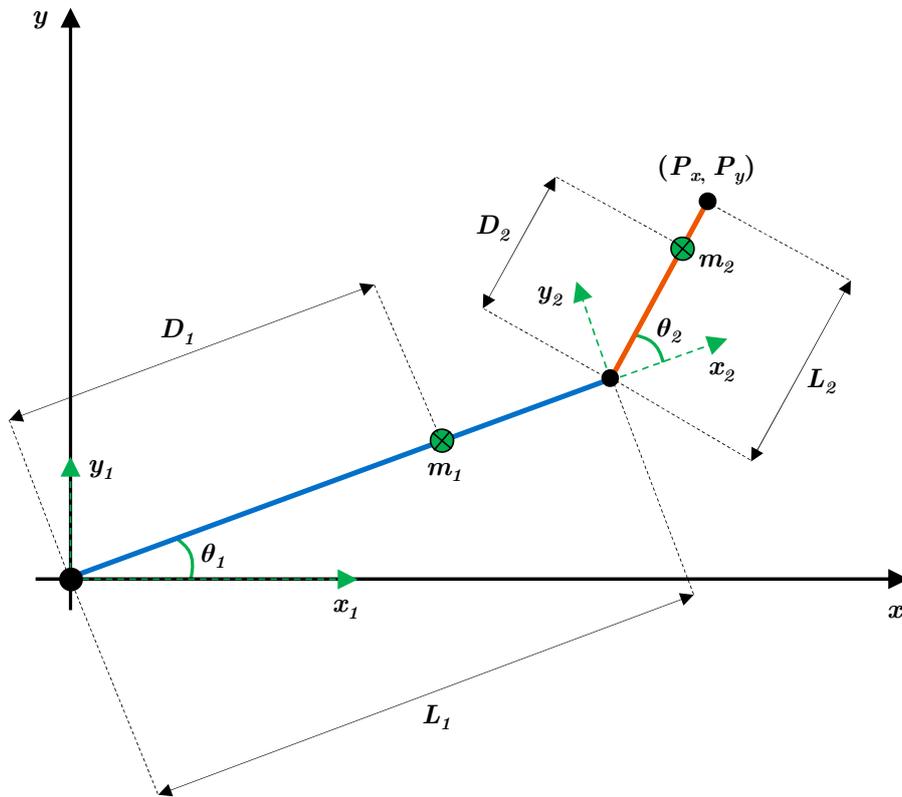


Figura 3.7 - Diagrama esquemático do robô SCARA.

na qual θ_1 é o ângulo entre o primeiro elo e o eixo x , medido em graus $[\circ]$, D_1 é a distância entre o ponto inicial do primeiro elo e o seu centro de massa m_1 , medido em metros $[m]$, L_1 é o comprimento total do primeiro elo, medido em metros $[m]$, θ_2 é o ângulo entre o segundo elo e o eixo x , medido em graus $[\circ]$, D_2 é a distância entre o ponto inicial do segundo elo e o seu centro de massa m_2 , medido em metros $[m]$, L_2 é o comprimento total do segundo elo, medido em metros $[m]$ e (P_x, P_y) corresponde ao ponto de intersecção dos eixos x e y em seu plano.

Conforme definido por Barrientos et al. (2007), a obtenção do modelo dinâmico do manipulador é de extrema importância para a identificação dos esforços atuantes na estrutura, e a obtenção dos torques de acionamento das juntas. A maneira de chegar-se às relações dinâmicas do manipulador é pela abordagem de Lagrange, no qual utiliza-se a análise de energias envolvidas no sistema (BEER et al., 2011). O robô SCARA contará com cinco graus de liberdade, desta forma a abordagem de Lagrange será utilizada, pois a mesma é indicada para a análise das cadeias cinemáticas com grande número de graus de liberdade.

Para a aplicação da abordagem de Lagrange o parâmetro L_g , denominado lagrangiano, é definido pela equação (3.18), e ao tomar a derivada deste parâmetro em relação as variáveis de junta pode-se obter o torque necessário ao acionamento τ_n , conforme a equação (3.19):

$$L_g = E_c - E_p \quad (3.18)$$

$$\tau_n = \frac{d}{dt} \cdot \frac{\partial L_g}{\partial \dot{q}_i} - \frac{\partial L_g}{\partial q_i} \quad (3.19)$$

no qual L_g corresponde ao lagrangiano, E_c é a energia cinética, medida em joules [J], E_p é a energia potencial, medida em joules [J], τ_n é o torque necessário de acionamento, medido em newton-metro [$N \cdot m$], q_i é o carregamento distribuído, medido em newton por metro [N/m] e \dot{q}_i é a primeira derivada¹ de q_i em relação ao tempo.

Com base na representação ilustrada na Figura 3.7, desenvolve-se a manipulação das equações (3.18) e (3.19). Para a equação (3.18), existe a ausência da componente de energia potencial E_p , resultando na manipulação expressa pela equação (3.20) (BARRIENTOS et al., 2007):

$$L_g = E_c = \frac{1}{2} \cdot m_i \cdot v^2 + \frac{1}{2} \cdot I_i \cdot \dot{\theta}^2 \quad (3.20)$$

na qual L_g corresponde ao lagrangiano, E_c é a energia cinética, medida em joules [J], m_i é o centro de massa do elo, medido em quilogramas [kg], v é a velocidade de movimento do elo, medida em metros por segundo [m/s], I_i é o momento de inércia no centro de massa, medido em quilograma-metro quadrado [$kg \cdot m^2$] e $\dot{\theta}$ é a primeira derivada do ângulo em relação ao tempo, medida em radianos por segundo [rad/s].

Conforme a equação (3.20), resolvendo para o primeiro elo do centro de massa m_1 e o momento de inércia no centro de massa I_1 , obtêm-se a expressão (3.21):

$$E_{c1} = \frac{1}{2} \cdot m_1 \cdot (CM_1 \cdot L_1 \cdot \dot{\theta}_1^2) + \frac{1}{2} \cdot I_1 \cdot \dot{\theta}^2 \quad (3.21)$$

¹Conforme a notação de Newton, um ponto sobre a variável (\dot{x}) indica a primeira derivada em relação ao tempo, representando a velocidade se x for posição. Dois pontos (\ddot{x}) simbolizam a segunda derivada em relação ao tempo, geralmente associada à aceleração em análises cinemáticas.

no qual E_{c1} é a energia cinética do primeiro elo, medida em joules $[J]$, m_1 é o centro de massa do primeiro elo, medido em quilogramas $[kg]$, L_1 é o comprimento total do primeiro elo, medido em metros $[m]$, $\dot{\theta}_1$ é a primeira derivada do ângulo entre o primeiro elo e o eixo x em relação ao tempo, medida em radianos por segundo $[rad/s]$, I_1 é o momento de inércia no centro de massa do primeiro elo, medido em quilograma-metro quadrado $[kg \cdot m^2]$, $\dot{\theta}$ é a primeira derivada do ângulo em relação ao tempo, medida em radianos por segundo $[rad/s]$ e CM_1 representa a razão entre a distância do centro de massa ao eixo de rotação da junta e o comprimento do primeiro elo.

Para esta análise, $CM_1 = 1$ indica centro de massa na extremidade do elo e $CM_1 = 0$ indica centro de massa coincidente com o de rotação. Para a energia cinética do segundo elo, verifica-se que a posição de seu centro de massa CM_2 é obtida para o eixo x e o eixo y , expressas pelas equações (3.22) e (3.23), respectivamente:

$$X_{cm2} = L_1 \cdot C\theta_1 + CM_2 \cdot L_2 \cdot C\theta_{12} \quad (3.22)$$

$$Y_{cm2} = L_1 \cdot S\theta_1 + CM_2 \cdot L_2 \cdot S\theta_{12} \quad (3.23)$$

no qual X_{cm2} e Y_{cm2} são as posições nos eixos x e y do centro de massa do segundo elo, respectivamente, ambas medidas em metros $[m]$, L_1 é o comprimento total do primeiro elo, medido em metros $[m]$, CM_2 representa a razão entre a distância do centro de massa ao eixo de rotação da junta e o comprimento do segundo elo, L_2 é o comprimento total do segundo elo, medido em metros $[m]$, $C\theta_1$ corresponde ao cosseno de θ_1 , logo $C\theta_1 = \cos(\theta_1)$, $S\theta_1$ corresponde ao seno de θ_1 , logo $S\theta_1 = \sin(\theta_1)$, θ_{12} é a soma dos ângulos θ_1 e θ_2 , logo $\theta_{12} = \theta_1 + \theta_2$, θ_{12} é medido em graus $[^\circ]$, por fim, $C\theta_{12}$ e $S\theta_{12}$ representam respectivamente o cosseno e o seno da soma dos ângulos θ_1 e θ_2 .

Ao realizar a primeira derivada das equações (3.22) e (3.23), obtêm-se as velocidades para o centro de massa no eixo x e no eixo y , expressas pelas equações (3.24) e (3.25), respectivamente:

$$\dot{X}_{cm2} = -L_1 \cdot S\theta_1 \cdot \dot{\theta}_1 - CM_2 \cdot L_2 \cdot S\theta_{12} \cdot \dot{\theta}_{12} \quad (3.24)$$

$$\dot{Y}_{cm2} = L_1 \cdot C\theta_1 \cdot \dot{\theta}_1 - CM_2 \cdot L_2 \cdot C\theta_{12} \cdot \dot{\theta}_{12} \quad (3.25)$$

na qual \dot{X}_{cm2} e \dot{Y}_{cm2} são as primeiras derivadas das posições nos eixos x e y do centro de massa do segundo elo, respectivamente, ambas medidas em metros por segundo $[m/s]$, L_1 é o comprimento total do primeiro elo, medido em metros $[m]$, $\dot{\theta}_1$ é a primeira derivada do ângulo entre o primeiro elo e o eixo x em relação ao tempo, medida em radianos por segundo $[rad/s]$, CM_2 representa a razão entre a distância do centro de massa ao eixo de rotação da junta e o comprimento do segundo elo, L_2 é o comprimento total do segundo elo, medido em metros $[m]$, $S\theta_{12}$ e $C\theta_{12}$ correspondem ao seno e cosseno da soma de θ_1 e θ_2 , respectivamente e $\dot{\theta}_{12}$ é a primeira derivada da soma dos ângulos θ_1 e θ_2 , $\dot{\theta}_{12}$ é medida em radianos por segundo $[rad/s]$.

A partir das equações (3.24) e (3.25), pode-se encontrar assim a resultante de velocidade, conforme a equação (3.26):

$$V_{cm2}^2 = \dot{X}_{cm2}^2 + \dot{Y}_{cm2}^2 \quad (3.26)$$

no qual V_{cm2} é a resultante de velocidade para o centro de massa do segundo elo, medida em metros por segundo $[m/s]$ e \dot{X}_{cm2} e \dot{Y}_{cm2} correspondem as primeiras derivadas das posições nos eixos x e y do centro de massa do segundo elo, respectivamente, ambas medidas em metros por segundo $[m/s]$.

Ao utilizar a equação (3.26), aplicando multiplicações e simplificações, obtêm-se a expressão (3.27):

$$V_{cm2}^2 = L_1^2 \cdot \dot{\theta}_1^2 + CM_2^2 \cdot L_2^2 \cdot \dot{\theta}_{12}^2 + 2 \cdot L_1 \cdot CM_2 \cdot L_2 \cdot \dot{\theta}_1 \cdot \dot{\theta}_{12} \cdot C\theta_2 \quad (3.27)$$

na qual V_{cm2} é a resultante de velocidade para o centro de massa do segundo elo, medida em metros por segundo $[m/s]$, L_1 é o comprimento total do primeiro elo, medido em metros $[m]$, $\dot{\theta}_1$ é a primeira derivada do ângulo entre o primeiro elo e o eixo x em relação ao tempo, medida em radianos por segundo $[rad/s]$, CM_2 representa a razão entre a distância do centro de massa ao eixo de rotação da junta e o comprimento do segundo elo, L_2 é o comprimento total do segundo elo, medido em metros $[m]$, $\dot{\theta}_{12}$ é a primeira derivada da soma dos ângulos θ_1 e θ_2 , medida em

radianos por segundo $[rad/s]$, $C\theta_{12}$ corresponde ao cosseno da soma de θ_1 e θ_2 , por fim, $C\theta_2$ corresponde ao cosseno de θ_2 .

Resolvendo para o segundo elo do centro de massa m_2 e o momento de inércia no centro de massa I_2 , obtêm-se a expressão (3.28):

$$E_{c2} = \frac{1}{2} \cdot m_2 \cdot (CM_2 \cdot L_2 \cdot \dot{\theta}_1^2) + \frac{1}{2} \cdot I_2 \cdot \dot{\theta}_{12}^2 \quad (3.28)$$

no qual E_{c2} é a energia cinética do segundo elo, medida em joules $[J]$, m_2 é o centro de massa do segundo elo, medido em quilogramas $[kg]$, CM_2 representa a razão entre a distância do centro de massa ao eixo de rotação da junta e o comprimento do segundo elo, L_2 é o comprimento total do segundo elo, medido em metros $[m]$, $\dot{\theta}_1$ é a primeira derivada do ângulo entre o primeiro elo e o eixo x em relação ao tempo, medida em radianos por segundo $[rad/s]$, I_2 é o momento de inércia no centro de massa do segundo elo, medido em quilograma-metro quadrado $[kg \cdot m^2]$ e $\dot{\theta}_{12}$ é a primeira derivada da soma dos ângulos θ_1 e θ_2 , medida em radianos por segundo $[rad/s]$.

A partir da equação (3.28), aplica-se na mesma a expressão (3.27), conforme apresenta-se na equação (3.29):

$$E_{c2} = \frac{1}{2} \cdot m_2 \cdot (L_1^2 \cdot \dot{\theta}_1^2 + CM_2^2 \cdot L_2^2 \cdot \dot{\theta}_{12}^2 + 2 \cdot L_1 \cdot CM_2 \cdot L_2 \cdot \dot{\theta}_1 \cdot \dot{\theta}_{12} \cdot C\theta_2) + \frac{1}{2} \cdot I_2 \dot{\theta}_{12}^2 \quad (3.29)$$

na qual E_{c2} é a energia cinética do segundo elo, medida em joules $[J]$, m_2 é o centro de massa do segundo elo, medido em quilogramas $[kg]$, L_1 é o comprimento total do primeiro elo, medido em metros $[m]$, $\dot{\theta}_1$ é a primeira derivada do ângulo entre o primeiro elo e o eixo x em relação ao tempo, medida em radianos por segundo $[rad/s]$, CM_2 é a razão entre a distância do centro de massa ao eixo de rotação da junta e o comprimento do segundo elo, L_2 é o comprimento total do segundo elo, medido em metros $[m]$, $\dot{\theta}_{12}$ é a primeira derivada da soma dos ângulos θ_1 e θ_2 , medida em radianos por segundo $[rad/s]$, $C\theta_2$ corresponde ao cosseno de θ_2 e I_2 é o momento de inércia no centro de massa do segundo elo, medido em quilograma-metro

quadrado $[kg \cdot m^2]$.

Segundo Beer et al. (2011), referindo-se à primeira junta, a expressão para o laplaciano deve ser dada pela equação (3.30):

$$L_p = E_{c1} + E_{c2} = \frac{1}{2} \cdot m_1 \cdot (CM_1 \cdot L_1 \cdot \dot{\theta}_1^2) + \frac{1}{2} \cdot I_i \cdot \dot{\theta}^2 \cdot \frac{1}{2} \cdot m_2 \cdot (L_1^2 \cdot \dot{\theta}_1^2) + (CM_2^2 \cdot L_2^2 \cdot \dot{\theta}_{12}^2 + 2 \cdot L_1 \cdot CM_2 \cdot L_2 \cdot \dot{\theta}_1 \cdot \dot{\theta}_{12} \cdot C\theta_2) + \frac{1}{2} \cdot I_2 \cdot \dot{\theta}_{12}^2 \quad (3.30)$$

no qual L_p é o laplaciano, E_{c1} e E_{c2} representam a energia cinética do primeiro e segundo elo, respectivamente, ambos medidos em joules $[J]$, m_1 e m_2 correspondem ao centro de massa do primeiro e segundo elo, respectivamente, ambos medidos em quilogramas $[kg]$, CM_1 e CM_2 representam a razão entre a distância do centro de massa ao eixo de rotação da junta e o comprimento do primeiro e segundo elo, respectivamente, L_1 e L_2 correspondem ao comprimento total do primeiro e segundo elo, respectivamente, ambos medidos em metros $[m]$, $\dot{\theta}_1$ é a primeira derivada do ângulo entre o primeiro elo e o eixo x em relação ao tempo, medida em radianos por segundo $[rad/s]$, $\dot{\theta}$ é a primeira derivada do ângulo em relação ao tempo, medida em radianos por segundo $[rad/s]$, $\dot{\theta}_{12}$ é a primeira derivada da soma dos ângulos θ_1 e θ_2 , $\dot{\theta}_{12}$ é medida em radianos por segundo $[rad/s]$, I_i é o momento de inércia no centro de massa, medido em quilograma-metro quadrado $[kg \cdot m^2]$, $C\theta_2$ é o cosseno de θ_2 e I_2 é o momento de inércia no centro de massa do segundo elo, medido em quilograma-metro quadrado $[kg \cdot m^2]$.

Em que pode-se obter os elementos para a aplicação na equação (3.19), aplicando-se a derivada parcial, conforme exposto na equação (3.31):

$$\frac{\partial L_p}{\partial \theta_1} = 0 \quad (3.31)$$

no qual L_p é o laplaciano, θ_1 é o ângulo entre o primeiro elo e o eixo x , medido em graus $[^\circ]$ e a expressão $\frac{\partial L_p}{\partial \theta_1}$ representa a derivada parcial do laplaciano L_p em relação ao ângulo θ_1 .

Realizando a primeira derivada da equação (3.31) de acordo com a aplicação da equação (3.19), obtêm-se a expressão (3.32):

$$\begin{aligned} \frac{\partial L_p}{\partial \dot{\theta}_1} = & m_1 \cdot CM_1^2 \cdot L_1^2 \cdot \dot{\theta}_1 + I_1 \cdot \dot{\theta}_1 + \frac{1}{2} \cdot m_2 \cdot [2 \cdot L_1^2 \cdot \dot{\theta}_1 + 2 \cdot CM_2^2 \cdot L_2^2 \cdot \dot{\theta}_{12} \\ & + 2 \cdot L_1 \cdot L_2 \cdot CM_2 \cdot C\theta_2 \cdot (\dot{\theta}_1 \cdot \dot{\theta}_{12})] + I_2 \cdot \dot{\theta}_{12} \end{aligned} \quad (3.32)$$

na qual L_p é o laplaciano, θ_1 é o ângulo entre o primeiro elo e o eixo x , medido em graus [°], a expressão $\frac{\partial L_p}{\partial \dot{\theta}_1}$ representa a derivada parcial do laplaciano L_p em relação a primeira derivada do ângulo θ_1 , m_1 e m_2 correspondem ao centro de massa do primeiro e segundo elo, respectivamente, ambos medidos em quilogramas [kg], CM_1 e CM_2 representam a razão entre a distância do centro de massa ao eixo de rotação da junta e o comprimento do primeiro e segundo elo, respectivamente, L_1 e L_2 correspondem ao comprimento total do primeiro e segundo elo, respectivamente, ambos medidos em metros [m], $\dot{\theta}_1$ é a primeira derivada do ângulo entre o primeiro elo e o eixo x em relação ao tempo, medida em radianos por segundo [rad/s], $C\theta_2$ é o cosseno de θ_2 , $\dot{\theta}_{12}$ é a primeira derivada da soma dos ângulos θ_1 e θ_2 , medida em radianos por segundo [rad/s] e I_1 e I_2 representam o momento de inércia no centro de massa do primeiro e segundo elo, respectivamente, ambos medidos em quilograma-metro quadrado [kg · m²].

A equação (3.33) corresponde à segunda derivada da equação (3.32) em relação ao tempo:

$$\begin{aligned} \frac{d}{dt} \left(\frac{\partial L_p}{\partial \dot{\theta}_1} \right) = & m_1 \cdot CM_1^2 \cdot L_1^2 \cdot \ddot{\theta}_1 + I_1 \cdot \ddot{\theta}_1 + \frac{1}{2} \cdot m_2 \cdot \{2 \cdot L_1^2 \cdot \ddot{\theta}_1 \\ & + 2 \cdot CM_2^2 \cdot L_2^2 \cdot \ddot{\theta}_{12} + 2 \cdot L_1 \cdot L_2 \cdot CM_2 \cdot [-S\theta_2 \cdot \dot{\theta}_2 \cdot (\dot{\theta}_1 + \dot{\theta}_{12}) \\ & + C\theta_2 \cdot (\ddot{\theta}_1 + \ddot{\theta}_{12})]\} + I_2 \cdot \ddot{\theta}_{12} \end{aligned} \quad (3.33)$$

no qual L_p é o laplaciano, θ_1 é o ângulo entre o primeiro elo e o eixo x , medido em graus [°], a expressão $\frac{\partial L_p}{\partial \dot{\theta}_1}$ representa a derivada parcial do laplaciano L_p em relação a primeira derivada do ângulo θ_1 , a expressão $\frac{d}{dt}$ corresponde à segunda derivada da equação em relação ao tempo, m_1 e m_2 correspondem ao centro de massa do primeiro e segundo elo, respectivamente, ambos medidos em quilogramas [kg], CM_1 e CM_2 representam a razão entre a distância do centro de massa ao eixo de rotação da junta e o comprimento do primeiro e segundo elo, respectivamente, L_1 e L_2 correspondem ao comprimento total do primeiro e segundo elo, respectivamente, ambos medidos

em metros $[m]$, $\dot{\theta}_1$ e $\dot{\theta}_2$ correspondem a primeira derivada do ângulo entre o primeiro e segundo elo e o eixo x em relação ao tempo, respectivamente, ambas medidas em radianos por segundo $[rad/s]$, $\dot{\theta}_{12}$ é a primeira derivada da soma dos ângulos θ_1 e θ_2 , medida em radianos por segundo $[rad/s]$, I_1 e I_2 representam o momento de inércia no centro de massa do primeiro e segundo elo, respectivamente, ambos medidos em quilograma-metro quadrado $[kg \cdot m^2]$, $\ddot{\theta}_1$ e $\ddot{\theta}_2$ correspondem a segunda derivada do ângulo entre o primeiro e segundo elo e o eixo x em relação ao tempo, respectivamente, ambas medidas em radianos por segundo ao quadrado $[rad/s^2]$, $C\theta_2$ e $S\theta_2$ correspondem respectivamente ao cosseno e seno de θ_2 e $\ddot{\theta}_{12}$ é a segunda derivada da soma dos ângulos θ_1 e θ_2 , medida em radianos por segundo ao quadrado $[rad/s^2]$.

Assim substituindo as equações (3.31) e (3.33) em (3.19), obtêm-se o resultado para o torque na primeira junta, conforme a equação (3.34):

$$\begin{aligned} \tau_{n1} = & m_1 \cdot CM_1^2 \cdot L_1^2 \cdot \ddot{\theta}_1 + I_1 \cdot \ddot{\theta}_1 + \frac{1}{2} \cdot m_2 \cdot \{2 \cdot L_1^2 \cdot \ddot{\theta}_1 \\ & + 2 \cdot CM_2^2 \cdot L_2^2 \cdot \ddot{\theta}_{12} + 2 \cdot L_1 \cdot L_2 \cdot CM_2 \cdot [-S\theta_2 \cdot \dot{\theta}_2 \cdot (\dot{\theta}_1 + \dot{\theta}_{12}) \\ & + C\theta_2 \cdot (\ddot{\theta}_1 + \ddot{\theta}_{12})]\} + I_2 \cdot \ddot{\theta}_{12} \end{aligned} \quad (3.34)$$

na qual τ_{n1} é o torque necessário para o acionamento da primeira junta, medido em newton-metro $[N \cdot m]$, m_1 e m_2 correspondem ao centro de massa do primeiro e segundo elo, respectivamente, ambos medidos em quilogramas $[kg]$, CM_1 e CM_2 representam a razão entre a distância do centro de massa ao eixo de rotação da junta e o comprimento do primeiro e segundo elo, respectivamente, L_1 e L_2 correspondem ao comprimento total do primeiro e segundo elo, respectivamente, ambos medidos em metros $[m]$, $\dot{\theta}_1$ e $\dot{\theta}_2$ correspondem a primeira derivada do ângulo entre o primeiro e segundo elo e o eixo x em relação ao tempo, respectivamente, ambas medidas em radianos por segundo $[rads]$, $\dot{\theta}_{12}$ é a primeira derivada da soma dos ângulos θ_1 e θ_2 , medida em radianos por segundo $[rad/s]$, I_1 e I_2 representam o momento de inércia no centro de massa do primeiro e segundo elo, respectivamente, ambos medidos em quilograma-metro quadrado $[kg \cdot m^2]$, $\ddot{\theta}_1$ e $\ddot{\theta}_2$ correspondem a segunda derivada do ângulo entre o primeiro e segundo elo e o eixo x em relação ao tempo, respectivamente, ambas medida em radianos por segundo ao quadrado $[rad/s^2]$, $C\theta_2$ e $S\theta_2$ correspondem respectivamente ao cosseno e seno de θ_2 e $\ddot{\theta}_{12}$ é a segunda derivada da soma dos ângulos θ_1 e θ_2 , medida em radianos por segundo ao quadrado

[rad/s²].

De maneira análoga, será encontrado o valor de torque para a segunda junta. A expressão para o laplaciano é observada conforme a equação (3.35) (BEER et al., 2011):

$$\frac{\partial L_p}{\partial \theta_2} = 0 \quad (3.35)$$

no qual L_p é o laplaciano, θ_2 é o ângulo entre o segundo elo e o eixo x , medido em graus [°], a expressão $\frac{\partial L_p}{\partial \theta_2}$ representa a derivada parcial do laplaciano L_p em relação ao ângulo θ_2 .

Realizando a primeira derivada da equação (3.35) de acordo com a aplicação da equação (3.19), conforme a expressão (3.36):

$$\frac{\partial L_p}{\partial \dot{\theta}_2} = \frac{1}{2} \cdot m_2 \cdot [2 \cdot CM_2^2 \cdot L_2^2 \cdot \dot{\theta}_{12} + 2 \cdot L_1 \cdot CM_2 \cdot L_2 \cdot \dot{\theta}_1 \cdot C\theta_2] + I_2 \cdot \dot{\theta}_{12} \quad (3.36)$$

na qual L_p é o laplaciano, θ_2 é o ângulo entre o segundo elo e o eixo x , medido em graus [°], a expressão $\frac{\partial L_p}{\partial \theta_2}$ representa a derivada parcial do laplaciano L_p em relação a primeira derivada do ângulo θ_2 , m_2 é o centro de massa do segundo elo, medido em quilogramas [kg], CM_2 é a razão entre a distância do centro de massa ao eixo de rotação da junta e o comprimento do segundo elo, L_1 e L_2 correspondem ao comprimento total do primeiro e segundo elo, respectivamente, ambos medidos em metros [m], $\dot{\theta}_1$ é a primeira derivada do ângulo entre o primeiro elo e o eixo x em relação ao tempo, medida em radianos por segundo [rad/s], $C\theta_2$ é o cosseno de θ_2 , $\dot{\theta}_{12}$ é a primeira derivada da soma dos ângulos θ_1 e θ_2 , medida em radianos por segundo [rad/s] e I_2 é o momento de inércia no centro de massa do segundo elo, medido em quilogramas-metro quadrado [kg · m²].

Então, obtêm-se a segunda derivada da equação (3.36) em relação ao tempo, conforme definido na equação (3.37):

$$\frac{d}{dt} \left(\frac{\partial L_p}{\partial \dot{\theta}_2} \right) = \frac{1}{2} \cdot m_2 \cdot [2 \cdot CM_2^2 \cdot L_2^2 \cdot \ddot{\theta}_{12} + 2 \cdot L_1 \cdot CM_2 \cdot L_2 \cdot (C\theta_2 \cdot \ddot{\theta}_1 - S\theta_2 \cdot \dot{\theta}_1 \cdot \dot{\theta}_2)] + I_2 \cdot \ddot{\theta}_{12} \quad (3.37)$$

no qual L_p é o laplaciano, θ_2 é o ângulo entre o primeiro elo e o eixo x , medido em graus $[\circ]$, a expressão $\frac{\partial L_p}{\partial \dot{\theta}_2}$ representa a derivada parcial do laplaciano L_p em relação a primeira derivada do ângulo θ_2 , a expressão $\frac{d}{dt}$ corresponde à segunda derivada da equação em relação ao tempo, m_2 é o centro de massa do segundo elo, medido em quilogramas $[kg]$, CM_2 é a razão entre a distância do centro de massa ao eixo de rotação da junta e o comprimento do segundo elo, L_1 e L_2 correspondem ao comprimento total do primeiro e segundo elo, respectivamente, ambos medidos em metros $[m]$, $\dot{\theta}_1$ e $\dot{\theta}_2$ correspondem a primeira derivada do ângulo entre o primeiro e segundo elo e o eixo x em relação ao tempo, respectivamente, ambas medidas em radianos por segundo $[rad/s]$, I_2 é o momento de inércia no centro de massa do segundo elo, medido em quilograma-metro quadrado $[kg \cdot m^2]$, $\ddot{\theta}_1$ é a segunda derivada do ângulo entre o primeiro elo e o eixo x em relação ao tempo, medida em radianos por segundo ao quadrado $[rad/s^2]$, $C\theta_2$ e $S\theta_2$ correspondem respectivamente ao cosseno e seno de θ_2 e $\ddot{\theta}_{12}$ é a segunda derivada da soma dos ângulos θ_1 e θ_2 , medida em radianos por segundo ao quadrado $[rad/s^2]$.

Assim substituindo as equações (3.35) e (3.37) em (3.19), obtêm-se o resultado para o torque na segunda junta, conforme a equação (3.38):

$$\tau_{n2} = \frac{1}{2} \cdot m_2 \cdot [2 \cdot CM_2^2 \cdot L_2^2 \cdot \ddot{\theta}_{12} + 2 \cdot L_1 \cdot CM_2 \cdot L_2 \cdot (C\theta_2 \cdot \ddot{\theta}_1 - S\theta_2 \cdot \dot{\theta}_1 \cdot \dot{\theta}_2)] + I_2 \cdot \ddot{\theta}_{12} + m_2 \cdot L_1 \cdot CM_2 \cdot L_2 \cdot \dot{\theta}_1 \cdot \dot{\theta}_{12} \cdot S\theta_2 \quad (3.38)$$

na qual τ_{n2} é o torque necessário para o acionamento da segunda junta, medido em newton-metro $[N \cdot m]$, m_2 é o centro de massa do segundo elo, medido em quilograma $[kg]$, CM_2 é a razão entre a distância do centro de massa ao eixo de rotação da junta e o comprimento do segundo elo, L_1 e L_2 correspondem ao comprimento total do primeiro e segundo elo, respectivamente, ambos medidos em metros $[m]$, $\dot{\theta}_1$ e $\dot{\theta}_2$ correspondem a primeira derivada do ângulo entre o primeiro e segundo elo e o eixo x em relação ao tempo, respectivamente, ambas medidas em radianos por segundo $[rad/s]$, $\dot{\theta}_{12}$ é a primeira derivada da soma dos ângulos θ_1 e θ_2 , medida

em radianos por segundo $[rad/s]$, I_2 representa o momento de inércia no centro de massa do segundo elo, medido em quilograma-metro quadrado $[kg \cdot m^2]$, $\ddot{\theta}_1$ é a segunda derivada do ângulo entre o segundo elo e o eixo x em relação ao tempo, medida em radianos por segundo ao quadrado $[rad/s^2]$, $C\theta_2$ e $S\theta_2$ correspondem respectivamente ao cosseno e seno de θ_2 e $\ddot{\theta}_{12}$ é a segunda derivada da soma dos ângulos θ_1 e θ_2 , medida em radianos por segundo ao quadrado $[rad/s^2]$.

3.2.4 Desenho e processo de fabricação

No desenvolvimento do robô SCARA, o desenho detalha cada componente com precisão, enquanto a fabricação, frequentemente utilizando impressão 3D, transforma estes desenhos em peças reais, otimizando a funcionalidade do robô. A seção de desenho e processo de fabricação do robô SCARA é dividida em duas fases: i) modelagem das peças em 3D e ii) impressão das peças em 3D.

3.2.4.1 Modelagem das peças em 3D

Ao considerar a modelagem de peças 3D, é imperativo proceder com análise minuciosa, buscando informações adicionais e precisas que orientem o processo de *design* do robô. [Sachyani et al. \(2021\)](#) descreve que a escolha da modelagem impacta diretamente a funcionalidade, eficiência e eficácia do robô. Esta decisão é guiada por múltiplos fatores: i) precisão e detalhamento, ii) otimização de *design*, iii) facilidade de prototipagem e iteração, iv) integração de componentes e v) compatibilidade com impressão 3D.

O primeiro fator trata da precisão e detalhamento, assegurando que todas as tolerâncias, encaixes e interfaces entre as peças sejam respeitadas, evitando problemas durante a montagem e operação do robô. O segundo aborda a otimização de *design*, sendo aplicada para alcançar objetivos específicos, como a redução de peso e aumento da resistência. O terceiro fala sobre a facilidade de prototipagem e iteração, permitindo testar diferentes versões de peças de forma rápida e econômica. O quarto descreve a integração de componentes dentro do *design* das peças, componentes como sensores, atuadores e sistemas eletrônicos. O quinto e último fator diz sobre a compatibilidade com impressão 3D, considerando aspectos como sobreposições, ângulos e suportes necessários para a impressão eficaz das peças.

3.2.4.2 Impressão das peças em 3D

Em diálogos com professores universitários e engenheiros que pesquisam e atuam na área de robótica e modelagem de peças, foi obtido o consenso pela utilização

da impressão 3D como método primário de fabricação. Esta decisão foi baseada em considerações estratégicas que visam otimizar o processo de *design*, desenvolvimento e funcionamento do robô, estão caracterizadas como: a i) agilidade na prototipagem, ii) capacidade de produção de diversas geometrias, iii) eficiência de custo, iv) diversidade de materiais e v) flexibilidade no processo de *design*.

A primeira consideração aborda a agilidade na prototipagem das impressões 3D, esta característica facilita a experimentação e aprimoramento de variados *designs* em tempo menor. A segunda diz sobre a capacidade de produção de diversas geometrias, algumas podendo ser complexas de fabricar por métodos convencionais, abrindo caminho para o *design* de componentes personalizados e otimizados. A terceira fala sobre a eficiência de custo, já que a impressão 3D pode ser mais econômica se comparada aos métodos tradicionais de fabricação, especialmente para a produção de pequenos lotes ou peças únicas. A quarta aborda a diversidade de materiais, abrangendo ampla seleção com variadas propriedades mecânicas e térmicas. Por fim, a quinta consideração fala sobre a flexibilidade no processo de modelagem, permitindo modificações de forma rápida e eficiente.

Segundo [Barnatt \(2016\)](#), a seleção da impressora 3D depende de fatores como: i) precisão, ii) resolução, iii) volume de impressão, iv) compatibilidade com materiais selecionados e a v) acessibilidade.

A precisão e resolução da impressora garantem a exatidão dos componentes do robô, proporcionando encaixes precisos e tolerâncias apertadas que são essenciais para o seu funcionamento adequado. Além disso, o volume de impressão da máquina permite acomodar as peças maiores do robô, facilitando a fabricação em sessões únicas, reduzindo a necessidade de montar várias peças menores. A compatibilidade da impressora com variados materiais é necessária para processar eficientemente os materiais escolhidos na construção do robô. Paralelamente, a velocidade de impressão influencia diretamente a agilidade do processo de prototipagem e desenvolvimento.

Outro ponto a ser avaliado é a confiabilidade e a facilidade de uso da impressora. Máquinas com histórico de bom desempenho e interfaces de usuário intuitivas pode reduzir significativamente o tempo de inatividade e aumentar a eficiência do processo de fabricação ([EVANS, 2012](#)). O custo da impressora, deve ser avaliado em relação ao orçamento do projeto. É fundamental garantir o equilíbrio entre o custo e as funcionalidades oferecidas pela impressora. Além disso, a capacidade de atualização da impressora permite que o equipamento se adapte às necessidades de evolução do projeto. Por fim, a acessibilidade das impressoras impacta diretamente na logística

e no cronograma do projeto. A garantia de acesso às máquinas necessárias, sem enfrentar problemas de indisponibilidade, evita atrasos no desenvolvimento do robô SCARA.

Além disso, a escolha do material para a impressão 3D é importante já que sua estrutura física irá resistir aos esforços atuantes sobre o manipulador. A escolha do material também tem grande impacto no custo do projeto e no peso final do manipulador, na estética de modo geral e na maneira como será feita a fabricação (PEREIRA et al., 2014). Pela ordem de importância, foram definidos como principais aspectos a serem estudados para os materiais: i) custo, ii) a resistência, iii) durabilidade, iv) o peso, v) resistência a temperaturas, vi) a disponibilidade de fabricação, vii) facilidade de impressão e viii) estética final do manipulador com o material escolhido.

A escolha do material economicamente viável mantém o projeto dentro do orçamento, sem comprometer a qualidade. Segue-se a resistência do material, garantindo que as peças impressas suportem os esforços mecânicos durante a operação do manipulador. A durabilidade assegura vida útil ao manipulador, reduzindo a necessidade de manutenção frequente e substituição de peças. O peso do material afeta diretamente a eficiência do manipulador. Materiais leves podem melhorar a performance e reduzir o consumo de energia. A resistência a temperaturas garante que o manipulador funcione efetivamente em diversos ambientes operacionais, sem sofrer deformações ou danos devido a variações térmicas.

A disponibilidade de fabricação do material diz que materiais facilmente acessíveis simplificam o processo de fabricação e manutenção. A facilidade de impressão é outro aspecto a ser considerado. Materiais que podem ser impressos com facilidade e consistência garantem o processo de fabricação suave e eficiente, reduzindo o risco de falhas e defeitos. Por fim, a estética final do manipulador com o material escolhido não pode ser negligenciada. A aparência do manipulador pode ser importante, especialmente em contextos em que a apresentação visual do robô tem impacto, como nos ambientes educacionais ou de demonstração.

3.3 Projeto eletroeletrônico

O projeto eletroeletrônico envolve o planejamento e a execução dos aspectos elétricos e eletrônicos do robô. Isto inclui a seleção e integração de componentes como sensores, atuadores, microcontroladores e sistemas de alimentação elétrica. O objetivo principal desta fase é garantir que o sistema eletroeletrônico funcione de maneira

eficiente e harmoniosa, permitindo que o robô realize suas funções conforme o previsto (CHU; SUNG, 2013). A seção de projeto eletroeletrônico é dividida em três fases: i) atuadores e sensores, ii) módulos de processamento e controle e iii) alimentação do sistema. A Figura 3.8 ilustra o fluxograma de desenvolvimento da etapa de projeto eletroeletrônico do robô.

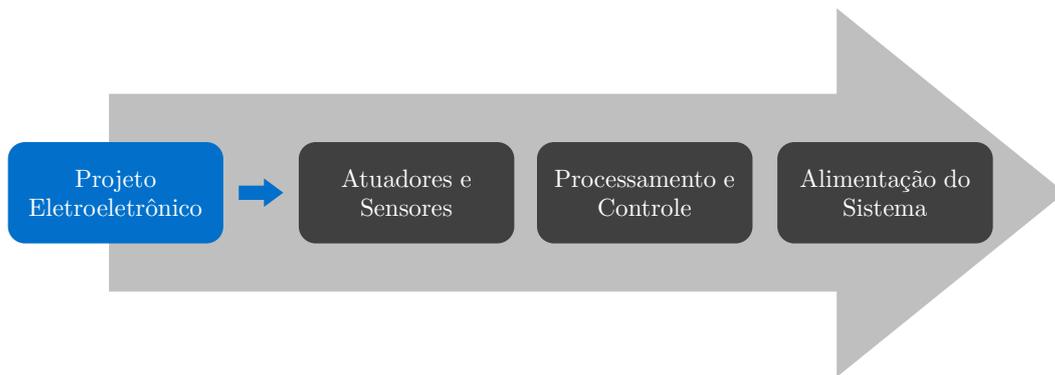


Figura 3.8 - Fluxograma da etapa de projeto eletroeletrônico.

A primeira fase, concentra-se na seleção dos motores que serão utilizados para a movimentação do robô, bem como na escolha dos sensores responsáveis pela identificação da posição do robô. Estas escolhas garantem que o robô tenha a capacidade mecânica necessária para realizar suas funções e que os sensores proporcionem os dados precisos para o controle efetivo. A segunda fase, envolve a definição dos parâmetros selecionados para o microcontrolador e para a placa de controle dos motores. A escolha do microcontrolador e da placa de controle deve alinhar-se com os requisitos de desempenho do sistema, assegurando a operação eficiente e precisa. Por fim, a terceira fase, trata da seleção da fonte de alimentação adequada para o robô. Esta fase é fundamental para garantir que todos os componentes eletroeletrônicos do sistema tenham o fornecimento de energia estável e suficiente.

3.3.1 Atuadores e sensores

Segundo Edwards (1984), a interação entre sensores e atuadores é o que possibilita ao robô SCARA responder dinamicamente ao seu ambiente, realizando operações com precisão e eficiência. A seção de atuadores e sensores envolve a escolha de quatro componentes para o projeto: i) motores de passo, ii) servo motor, iii) *driver* do motor de passo e iv) sensores fim de curso. Para a seleção dos motores de passo,

quatro fatores principais devem ser considerados nesta escolha: i) torque do motor, ii) resolução de passo, iii) velocidade e iv) compatibilidade com o sistema de controle.

Os motores passo, destacam-se pela precisão angular de posicionamento devido ao deslocamento discreto do rotor. Amplamente utilizados em manipuladores, estes motores permitem o controle de posicionamento, mesmo em malha aberta², desde que possuam torque suficiente para a aplicação (ROCHA, 2016). Vale ressaltar que, ao serem acionados em baixas velocidades, podem introduzir vibração ao movimento do manipulador devido ao movimento discretizado do rotor.

O torque adequado é imperativo para assegurar a capacidade do motor em movimentar ou sustentar a carga do robô, permitindo que o robô execute suas tarefas sem falhas ou atrasos nos movimentos (NIKU, 2010). Ademais, a resolução de passo do motor determina o menor ângulo de rotação possível deste. A alta resolução é fundamental para a precisão dos movimentos do robô, permitindo ajustes finos e controlados. Sua capacidade de operar a velocidade adequada mantém a eficiência produtiva do robô. Por fim, a compatibilidade do motor com o sistema de controle é a escolha que visa a coordenação eficaz e a operação suave do robô.

O atuador responsável pelo controle e movimentação da garra será o servo motor, sua seleção é fundamentada nas características específicas que atendem as necessidades do projeto: i) precisão, ii) capacidade de resposta e iii) torque do motor.

Conforme Usategui e León (1986), servomotores proporcionam precisão maior em comparação com motores de passo. No entanto, eles geralmente não são controlados em sistema de malha aberta. Servo motores apresentam baixo torque e alta velocidade de rotação para a maioria dos modelos disponíveis no mercado, exigindo o uso de reduções maiores para alcançar o torque necessário à aplicação.

O servo motor será responsável por proporcionar movimentos precisos e controlados da garra, essenciais para manipular objetos com eficiência. Ademais, a capacidade de resposta rápida é vital para operações dinâmicas, permitindo que a garra execute tarefas de maneira ágil. Por fim, a escolha do servo motor com torque adequado garante que a garra possa lidar com variedades de objetos de diferentes pesos e tamanhos, mantendo a estabilidade durante a manipulação. Por fim, a seleção cuidadosa do servo motor para a garra do robô tem como objetivo assegurar que a

²Sistema de controle no qual a ação de controle é independente da saída. Não utiliza *feedback*, ou seja, não ajusta automaticamente suas operações com base nos resultados, o que pode afetar a precisão em resposta a desvios ou perturbações externas.

garra opere de maneira eficiente, precisa e responsiva.

Ao selecionar o *driver* de motor de passo para o controle e parametrização dos servo motores do robô SCARA, três fatores principais devem ser levados em consideração: i) capacidade de corrente, ii) resolução e iii) compatibilidade.

O primeiro fator é a capacidade de corrente do *driver*. Esta capacidade deve ser compatível com as especificações dos servo motores usados, garantindo que haja energia suficiente para operá-los sem causar sobrecarga ou danos. O segundo aspecto importante é a resolução do *driver*, que determina a precisão com que os movimentos do motor podem ser controlados. A maior resolução permite controle fino dos movimentos. O terceiro fator é a compatibilidade do *driver* com o sistema de controle geral do robô.

A escolha dos sensores de fim de curso, destinados a limitar a posição física do robô, envolve a consideração de três fatores principais: i) precisão, ii) durabilidade e iii) compatibilidade com o sistema de controle.

Estes dispositivos devem ser capazes de detectar de forma precisa e confiável os limites de movimento do robô para evitar ultrapassagens que poderiam causar danos mecânicos ou falhas operacionais. Além disso, a durabilidade dos sensores é fator crítico, visto que eles estarão sujeitos a uso frequente e, por vezes, em condições desafiadoras. Assim, é necessário que sejam robustos e confiáveis para garantir longa vida útil sem necessidade de manutenção frequente. Por fim, os sensores devem ser capazes de se integrar facilmente ao sistema de controle, permitindo a comunicação eficaz e a operação harmoniosa.

3.3.2 Módulos de processamento e controle

De acordo com Perim e Nascimento (2017), os elementos de processamento de sinais devem ser compatíveis com os sistemas de acionamento, como motores de passo e servo motores, além de oferecer capacidades de integração, modularidade e eficiência em termos de custo. A seção de módulos de processamento e controle envolve a escolha de dois componentes para o projeto: i) microcontrolador e ii) *CNC Shield*. Para a seleção do microcontrolador, quatro fatores principais devem ser considerados nesta escolha: i) capacidade de controle de atuadores, ii) interfaces de comunicação e expansão, iii) capacidade de processamento e memória e iv) desempenho em tempo real.

Essencialmente, o microcontrolador deve possuir interfaces adequadas para gera-

ção de sinais PWM e gerenciamento de *feedback* dos *encoders* dos motores, garantindo precisão nos movimentos do robô. Ademais, é importante que o microcontrolador apresente diversas interfaces de comunicação, como *Inter-Integrated Circuit* (I2C), *Serial Peripheral Interface* (SPI), *Universal Asynchronous Receiver/Transmitter* (UART) e portas de expansão para integração com outros sistemas, sensores, módulos e adição de funcionalidades. Sua capacidade de processamento e memória deve ser suficiente para gerir algoritmos de controle, incluindo cálculos de cinemáticas e processamento de dados dos sensores. Por fim, a habilidade de executar tarefas em tempo real, processando rapidamente e lidando com múltiplas tarefas simultâneas, servirá para assegurar a sincronização e precisão dos movimentos do robô.

Em relação ao CNC *Shield*, a seleção desta placa é fundamentada nas características específicas que atendem as necessidades do projeto: i) compatibilidade com motores de passo e servo motores, ii) facilidade de integração com microcontroladores e iii) custo-efetividade.

O CNC *Shield* é altamente compatível com motores de passo e servo motores, característica essencial para garantir movimentos precisos e controlados (NIKU, 2010). Sua habilidade de gerenciar múltiplos motores de passo simultaneamente, juntamente com a capacidade de integrar servo motores, torna-a a escolha estratégica para o projeto, assegurando o controle eficaz dos movimentos do robô. Além disso, a placa é notável pela sua facilidade de integração com microcontroladores populares, simplificando o desenvolvimento do sistema de controle e proporcionando acesso a ampla gama de recursos e suporte comunitário. Economicamente, o CNC *Shield* se destaca como opção custo-efetiva, equilibrando funcionalidade e preço, o que é particularmente benéfico nos contextos educacionais e de pesquisa com orçamentos limitados.

3.3.3 Alimentação do sistema

Cinco fatores principais devem ser cuidadosamente considerados durante a escolha da fonte de alimentação: i) tensão de saída, ii) capacidade de corrente, iii) estabilidade, iv) eficiência energética e v) segurança.

A tensão de saída da fonte deve ser compatível com as necessidades de energia do robô. Isto garante que todos os componentes eletrônicos e motores recebam a tensão adequada para seu funcionamento ótimo. Leal (2022) define que a fonte de alimentação deve ser capaz de fornecer corrente suficiente para operar todos os componentes do robô simultaneamente, sem risco de sobrecarga ou falha. Ademais,

a fonte estável fornece energia constante, evitando flutuações que podem danificar os componentes eletrônicos ou afetar o desempenho do robô. Fontes de alimentação eficientes minimizam a perda de energia, reduzindo o calor gerado e contribuindo para a sustentabilidade operacional do robô. Por fim, a fonte de alimentação deve incluir proteções contra sobretensão, sobrecorrente e curto-circuito, garantindo a segurança do robô e dos usuários.

3.4 *Software* de simulação do robô

Conforme salientado por Nkomo e Collier (2012), a simulação do robô no ambiente tridimensional permite a análise, teste e aprimoramento de algoritmos de controle antes de sua implementação física. A execução da simulação do robô SCARA é imperativa para a representação virtual do manipulador no ambiente tridimensional. A Figura 3.9 ilustra o fluxograma de desenvolvimento da etapa de simulação do robô SCARA.

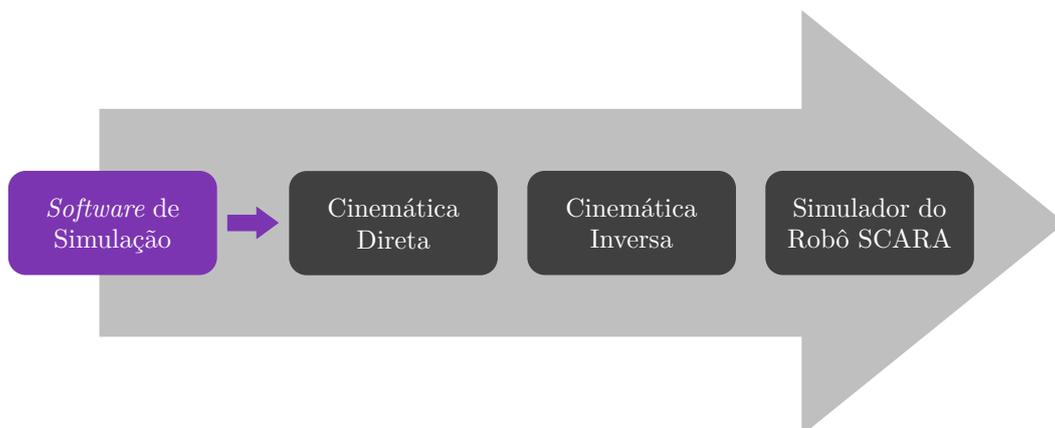


Figura 3.9 - Fluxograma da etapa de simulação do robô.

3.4.1 Simulador do robô SCARA

Na elaboração do *software* de simulação do robô SCARA, é imperativo incorporar duas modalidades fundamentais de operação: i) a cinemática direta e a ii) cinemática inversa.

A implementação da cinemática direta no *software* de simulação do robô oferece o método eficiente para calcular a posição final do efetuador baseando-se na configura-

ção angular dos eixos (SICILIANO et al., 2009). Por outro lado, a cinemática inversa, que será integrada no mesmo *software* de simulação, tem como objetivo calcular os ângulos necessários dos eixos do robô para atingir a posição específica do efetuador. Neste contexto, será desenvolvido e testado, algoritmos de controle, otimização de trajetórias e realização de planejamento de movimentos, além de conduzir análises de desempenho para validar o *design* do robô antes de sua implementação física. Será buscado o simulador capaz de realizar a modelagem detalhada do robô, abrangendo sua cinemática e dinâmica, e permitindo a interação com objetos virtuais.

3.5 Implementação do *software* de controle do robô

Conforme salientado por Pereira (2023), a modelagem cinemática do manipulador representa aspecto fundamental no processo de desenvolvimento, pois a partir da implementação das equações cinemáticas é que se coordena os movimentos das juntas do robô, permitindo o posicionamento do efetuador final, conforme desejado.

O *software* desempenhará o papel fundamental no funcionamento do robô, justificado por múltiplos motivos. Em primeiro lugar, ele será utilizado para interpretar e executar comandos de movimento, utilizando algoritmos de cinemática direta e inversa. Esta funcionalidade permitirá que o robô realize movimentos precisos e complexos, essenciais para demonstrar princípios fundamentais da robótica (MARTINS, 1993). Em segundo lugar, facilitará a interação entre o robô e o usuário, proporcionando interface intuitiva que permitirá aos estudantes programar e controlar o robô de forma eficiente, reforçando o aprendizado prático. Por último, oferecerá flexibilidade na experimentação, permitindo aos alunos testar diferentes abordagens de programação e controle. A Figura 3.10 ilustra o fluxograma de desenvolvimento da etapa de implementação do *software* de controle.

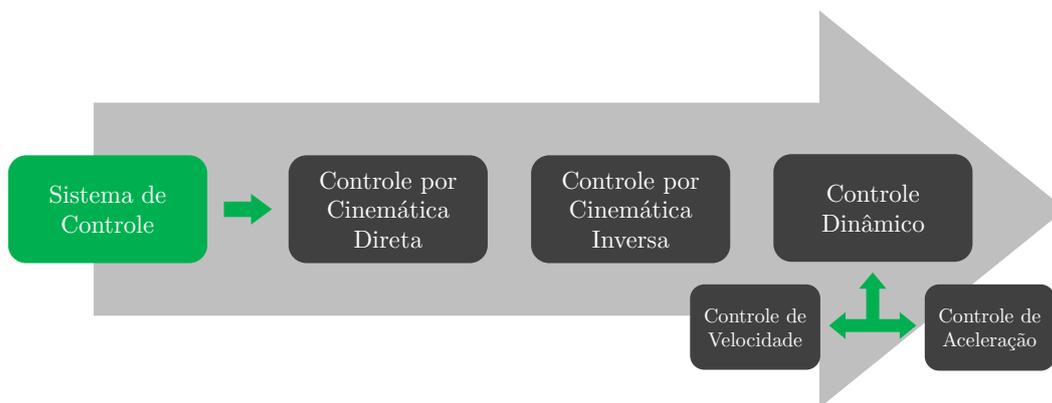


Figura 3.10 - Fluxograma da etapa de implementação do *software* de controle.

3.5.1 Cinemática direta e cinemática inversa

Na elaboração do *software* de controle para o robô SCARA, é imperativo incorporar duas modalidades fundamentais de operação e comando: i) a cinemática direta e a ii) cinemática inversa.

Incluir a cinemática direta no *software* de controle proporcionará o método eficiente para calcular a posição final do efetuador com base na configuração angular dos seus eixos. Esta abordagem é utilizada para tarefas que requerem movimentos precisos e controlados, em que as posições dos eixos são conhecidas e a posição do efetuador precisa ser calculada (KURDILA; BEN-TZVI, 2020).

Em contraste, a cinemática inversa será projetada com o objetivo de calcular os ângulos necessários dos eixos do robô para alcançar a posição desejada do efetuador. Esta abordagem é aplicada nos cenários em que a posição final do efetuador é conhecida, mas os ângulos dos eixos para alcançar esta posição precisam ser determinados. Groover et al. (1986) descreve que a cinemática inversa é especialmente útil para operações dinâmicas e adaptativas, em que o robô precisa se ajustar a diferentes posições ou trajetórias, garantindo a flexibilidade e adaptabilidade do sistema.

3.5.2 Controle dinâmico do robô

A intenção de implementar o controle dinâmico, é melhorar a precisão e a eficiência dos movimentos do robô, assegurando operações exatas e suaves. Além disso, o controle dinâmico permite melhor resposta a variações de carga e a condições ope-

racionais variáveis, muito explorado nos ambientes dinâmicos. Siciliano et al. (2009) sugere a abordagem de duas formas de controle dinâmico nos robôs: i) velocidade e ii) aceleração.

O controle de velocidade é utilizado para equilibrar a rapidez operacional e a exatidão das tarefas executadas (SICILIANO et al., 2009). A escolha deste controle para visa maximizar a produtividade do robô, minimizando o tempo de ciclo, ao mesmo tempo em que se mantém a integridade e a precisão dos movimentos, essenciais em operações de manipulação.

Quanto ao controle da aceleração, a intenção é gerenciar a taxa na qual o robô aumenta ou diminui sua velocidade, o que é fundamental para evitar instabilidades mecânicas e assegurar a segurança operacional. A escolha do controle preciso para a aceleração é motivada pela necessidade de proteger o sistema contra estresses mecânicos e garantir a operação segura e eficiente a longo prazo.

3.6 Considerações finais

Este capítulo descreve a metodologia utilizada para o desenvolvimento do robô SCARA, enfocando em aspectos mecânicos, eletroeletrônicos e de *software*. Detalha-se o processo desde o projeto mecânico, incluindo cálculos estáticos e dinâmicos, além do desenho e fabricação das peças. O projeto eletroeletrônico, abordando atuadores, sensores e alimentação, é explorado. A implementação do *software* de controle do robô, cobrindo cinemática direta e inversa e controle dinâmico, é discutida. O capítulo seguinte focará na apresentação dos resultados e discussões relacionados a esta abordagem.

CAPÍTULO 4

RESULTADOS E DISCUSSÕES

Este capítulo delinea inicialmente a construção e montagem do robô SCARA, destacando a meticulosa seleção e modelagem das peças mecânicas, bem como a utilização de tecnologias de fabricação avançadas como a impressão 3D. Segue-se com a integração dos componentes eletroeletrônicos essenciais para a funcionalidade do robô, incluindo a escolha criteriosa de atuadores, sensores, e sistemas de controle. A implementação do *software* de simulação e os métodos de controle adotados são então examinados, ressaltando a eficácia das soluções de programação e interfaces de usuário desenvolvidas. Ao final, discute-se a manutenção e a robustez do sistema, concluindo com a reflexão sobre as descobertas, desafios enfrentados e as implicações práticas e educacionais do projeto.

4.1 Construção mecânica

Os resultados apresentados estabelecem a base para a compreensão da construção mecânica de robôs, destacando a importância da seleção cuidadosa de *designs* e materiais, assim como a aplicação de princípios de engenharia mecânica e mecatrônica. Esta abordagem assegura a criação de estruturas robustas, precisas e eficientes, capazes de atender às exigências operacionais. Na sequência expõe os detalhes do processo de seleção e modelagem do manipulador, as técnicas de fabricação empregadas, a montagem do robô e a análise dos resultados obtidos nos cálculos de forças e movimentos, fornecendo a visão abrangente das etapas críticas envolvidas na construção do robô SCARA.

4.1.1 Seleção e modelagem do manipulador

Na pesquisa por modelos didáticos de robôs, foram consultados diversos trabalhos acadêmicos através do Google Scholar. A análise focou em estudos que apresentavam a modelagem 3D completa de robôs SCARA. Entre os modelos examinados, o projeto de Nedelkovski (2020), encontrado no site educacional *How To Mechatronics*, destacou-se significativamente. Este site oferece projetos gratuitos em engenharia mecânica, elétrica e de computação. Além disso, o autor Dejan Nedelkovski é técnico em eletrônica e possui formação superior em engenharia mecatrônica pela *Faculty of Mechanical Engineering Skopje*, localizada na cidade de Escópia na Macedônia do Norte.

Este modelo foi escolhido por sua abordagem prática e acessível, oferecendo integra-

ção eficaz entre a tecnologia de impressão 3D e componentes disponíveis no mercado. A escolha do modelo de robô SCARA de Nedelkovski (2020) para o projeto baseia-se em sete motivos principais: i) disponibilidade de recursos, ii) aplicabilidade educacional, iii) adaptabilidade do *design*, iv) viabilidade técnica, v) complexidade adequada, vi) cálculos estáticos e vii) cálculos dinâmicos.

No contexto de disponibilidade, o modelo oferece a combinação de recursos acessíveis e instruções detalhadas, facilitando a montagem e compreensão do projeto. Além disso, o projeto é ideal para fins educacionais, fornecendo a oportunidade prática para estudantes aprenderem sobre robótica e mecatrônica. A estrutura do robô permite modificações e adaptações, tornando-o versátil para diferentes aplicações educacionais. O projeto é tecnicamente viável, empregando técnicas e materiais acessíveis como impressão 3D. O modelo possui complexidade adequada para ser desafiador, mas ao mesmo tempo compreensível para estudantes, equilibrando aprendizado técnico e prático.

A escolha deste projeto permite a realização de análises estáticas detalhadas, essenciais para compreender as forças e os momentos atuantes em cada parte do robô, garantindo a robustez e a estabilidade da estrutura, assegurando o desempenho adequado do robô em diferentes condições operacionais. Por fim, a possibilidade de executar cálculos dinâmicos neste modelo oferece compreensão aprofundada da resposta do robô sob ação de cargas variáveis e movimentos, isto facilita a análise do comportamento do robô durante a operação, permitindo ajustes e melhorias no *design* e na programação para otimizar sua eficiência e precisão.

O projeto detalha a concepção do manipulador com cinco graus de liberdade, empregando motores de passo e servomotores, todos operados pelo microcontrolador e diversos *drivers* de motores de passo. Além disso, a configuração do manipulador SCARA com cinco graus de liberdade incluindo três juntas rotativas, uma junta prismática no eixo z , e movimento de abertura e fechamento da garra oferece versatilidade e complexidade adequadas para o projeto educacional. Seu *design* inclui componentes impressos em 3D, complementados por peças adquiridas em lojas especializadas de máquinas CNC (NEDELKOVSKI, 2020).

O *software* SolidWorks®, foi escolhido para modelar e desenvolver as peças do robô 3D no projeto baseado no modelo de Nedelkovski (2020). O SolidWorks®, é descrito como a ferramenta *Computer Aided Design* (CAD) e *Computer Aided Engineering* (CAE) amplamente utilizada na engenharia mecânica e *design* de produtos. Ele permite a criação de modelos 3D detalhados e realiza análises de elementos fini-

tos, facilitando o desenvolvimento de projetos mecânicos complexos (GIBSON et al., 2015). Seu uso em ambientes educacionais e profissionais é difundido devido à sua capacidade de simplificar processos de *design*, simulação e validação de projetos, tornando-o a escolha robusta e versátil para engenheiros e *designers*. A seleção deste *software* é fundamentada em quatro motivos principais: i) licenciamento institucional, ii) uso pelo autor original, iii) recursos avançados e iv) integração com impressão 3D.

Primeiramente, a disponibilidade da licença institucional do SolidWorks® oferece acesso imediato e sem custos adicionais. Além disso, Nedelkovski utilizou este *software* para o desenvolvimento de sua modelagem 3D, garantindo compatibilidade e continuidade. A interface intuitiva do SolidWorks®, juntamente com seus recursos avançados, facilita o processo de *design*, enquanto a capacidade do *software* de integrar-se com processos de impressão 3D assegura a produção precisa das peças.

Durante a adaptação do modelo, algumas alterações foram feitas para otimizar sua funcionalidade e durabilidade. A mudança de cor foi realizada para melhorar a visibilidade das partes e facilitar o aprendizado. O aumento da densidade do robô visou proporcionar maior estabilidade e resistência durante as operações. Para reforçar a resistência dos elos, a densidade dos materiais foi aumentada, em especial das polias, garantindo que o robô suporte o uso contínuo em ambiente educacional. Por fim, o preenchimento das polias nos elos do robô foi aumentado para melhorar a transmissão de força e a durabilidade das peças.

Na perspectiva da modelagem tridimensional das diversas componentes destinadas à construção do robô SCARA, é possível acessar, por meio do Apêndice A, as pranchetas de desenhos que viabilizam a representação bidimensional e tridimensional de todo o conjunto modelado do referido robô. Estas pranchetas exibem cada componente juntamente com suas respectivas dimensões e medidas, permitindo, assim, a elaboração de cada peça a partir do ponto de partida inicial. O apêndice compreende o total de 36 pranchetas, todas elaboradas utilizando o *software* SolidWorks®. A divisão do apêndice é tripartite, sendo o Apêndice A.1, dedicado à representação da estrutura física do robô, o Apêndice A.2 enfocando as polias e engrenagens internas, que são empregadas no sistema de transmissão de polias e correias, e o Apêndice A.3 destinado à modelagem do efetuador final. Ademais, as peças presentes nas pranchetas de referência A.30 e A.31 são de autoria própria. Estas componentes suplementares apresentam duas extremidades concebidas para serem integradas ao efetuador final, com o propósito de permitir a manipulação de objetos de dimensões

superiores àquelas que as extremidades padrões do manipulador podem acomodar. A Figura 4.1, adaptada de Nedelkovski (2020), ilustra o modelo tridimensional do manipulador selecionado.

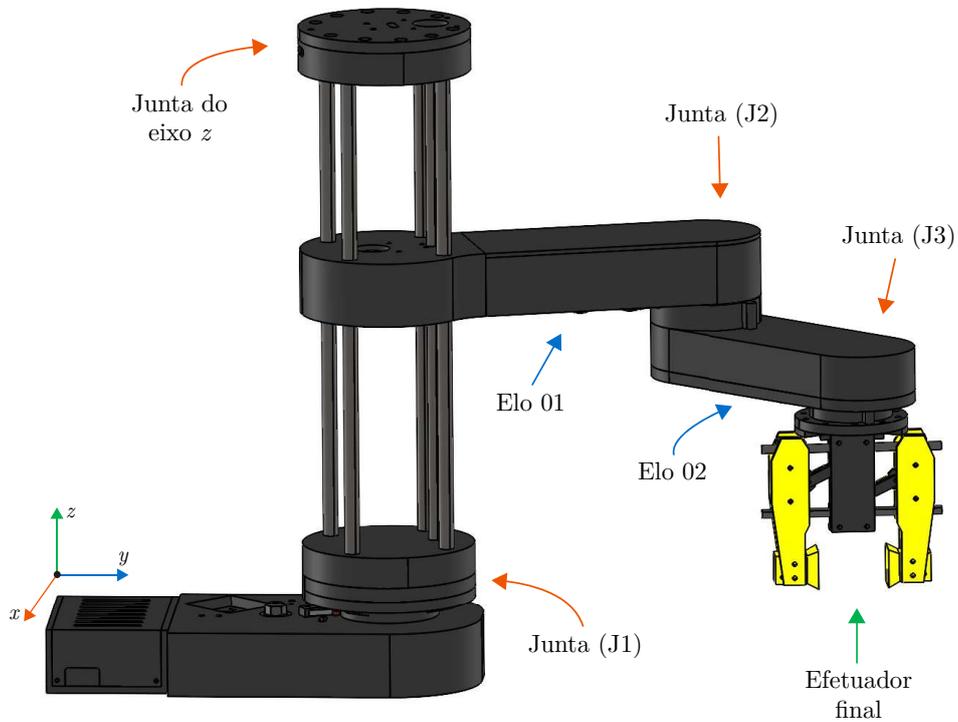


Figura 4.1 - Modelo 3D de robô SCARA escolhido para fabricação.

Nesta representação, a base do movimento é estabelecida pela primeira junta rotativa, conhecida como J1, que define o alcance horizontal do robô, realizando o movimento rotacional em torno do eixo z . O primeiro elo, acoplado à junta J1, é imperativo para o alcance do manipulador e responde aos comandos de rotação. Segue-se a segunda junta, J2, situada na extremidade do primeiro elo, conferindo ao robô um grau adicional de liberdade e contribuindo para a precisão do posicionamento. O segundo elo, ligado à J2, amplia o alcance do robô, sendo crucial para o posicionamento preciso do efetuador final. A junta J3, localizada no término do segundo elo, adiciona dimensão extra ao movimento, realizando a rotação do efetuador final. A junta do eixo z é responsável pelo movimento prismático do robô a partir do eixo z , permitindo que ele desloque-se verticalmente. Por fim, há também a última junta prismática do efetuador final, esta que realiza o movimento linear da garra.

4.1.2 Fabricação da estrutura

A fabricação da estrutura do robô SCARA foi realizada a partir da impressão 3D, incluindo a base do robô, os elos, o eixo de movimentação vertical e as polias dos sistemas de redução. Para a finalidade de produção da estrutura, optou-se pela utilização das impressoras nos modelos Creality Ender 3 e ZMorph VX, ambos os modelos disponíveis na instituição.

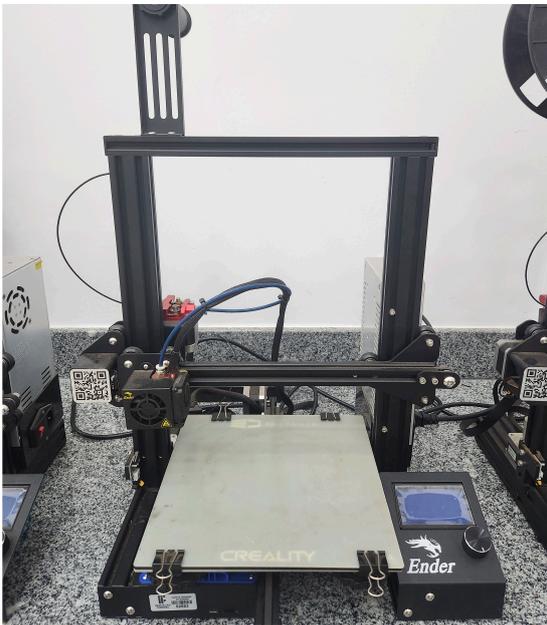
Segundo Evans (2012), o modelo Creality Ender 3 é reconhecido por sua acessibilidade financeira e pela confiabilidade operacional. Com o *design* de estrutura aberta¹, é frequentemente escolhida por usuários iniciantes e projetos de orçamento limitado. Por outro lado, a ZMorph VX se distingue pela sua precisão e versatilidade. O aspecto crítico para a sua precisão é o emprego do sistema de malha fechada, que monitora e controla com maior acurácia a posição dos motores de passo durante o processo de impressão. Resultando assim no controle mais eficaz e na minimização de erros, contribuindo para a produção de objetos com elevada precisão dimensional (LIPSON; KURMAN, 2013). Além da impressão 3D, a ZMorph VX oferece funcionalidades de fresagem CNC e gravação a laser, o que amplia consideravelmente sua aplicabilidade em projetos que requerem operações multiuso.

A seleção destes modelos de impressora 3D é fundamentada em cinco razões majoritárias: i) disponibilidade institucional, ii) expertise dos professores orientadores, iii) competência dos alunos do projeto, iv) precisão e v) versatilidade.

A primeira justificativa reside na disponibilidade destes dois modelos de impressoras 3D nas instalações do Instituto Federal Goiano - *Campus* Trindade, o que assegura o acesso direto e conveniente às tecnologias de impressão 3D para a consecução do projeto. O segundo motivo repousa no profundo conhecimento dos docentes orientadores no tocante ao manuseio das impressoras e à expertise em processos de manufatura aditiva. Considera-se a competência dos alunos envolvidos no projeto, os quais demonstraram conhecimento teórico e prático sólidos em relação à manufatura aditiva. Ambas impressoras destacam-se pela capacidade de produzir peças com elevada qualidade e precisão. Por fim, a versatilidade e a flexibilidade destes modelos de impressoras permitem não apenas a fabricação das peças estruturais do robô, mas também a experimentação com diferentes materiais e configurações de impressão. A Figura 4.2, ilustra os dois modelos de impressoras 3D utilizadas para a impressão da estrutura do robô SCARA. Ambas impressoras encontram-se nos

¹A estrutura aberta em impressoras 3D refere-se ao *design* que não possui a cobertura completa ao redor da área de impressão, permitindo fácil acesso e observação durante o processo de impressão.

laboratórios de engenharia elétrica da instituição.



(a)



(b)

Figura 4.2 - Impressoras utilizadas: (a) Creality Ender 3 e (b) ZMorph VX.

4.1.2.1 Fatiamento das modelagens

No contexto da impressão 3D, é imprescindível realizar a etapa prévia à produção, que consiste na preparação das peças a serem impressas. Esta preparação envolve a conversão do modelo tridimensional digital, geralmente no formato *Standard Tessellation Language* (STL), no conjunto de instruções compreensíveis para a impressora, conhecido como código G, ou *G-code*. Este procedimento é denominado de fatiamento. No processo de fatiamento, o modelo tridimensional é desmembrado em camadas horizontais, de forma que cada camada representa o plano de impressão (GIBSON et al., 2015). O *software* responsável por esta tarefa interpreta o *design* e gera o *G-code*, que instrui a impressora sobre a trajetória, a velocidade, a temperatura e outros parâmetros necessários para a produção camada a camada (EVANS, 2012).

Os *softwares* Creality Slicer e Voxelizer foram selecionados para conduzir o fatiamento das peças a serem produzidas pelas impressoras Creality Ender 3 e ZMorph VX, respectivamente. O Creality Slicer é a ferramenta que se destaca por sua interface intuitiva e facilidade de uso, sendo especialmente direcionado a impressoras da

marca Creality. Este *software* é desenvolvido para realizar o fatiamento de modelos 3D em camadas e a subsequente geração do G-code (LIPSON; KURMAN, 2013). Por outro lado, o Voxelizer é a aplicação que oferece o escopo mais amplo de funcionalidades, abrangendo não apenas o fatiamento de modelos, mas também recursos avançados, como a capacidade de realizar impressão 3D em múltiplos materiais, fresagem CNC e gravação a laser. O Voxelizer é direcionado para impressoras da marca ZMorph. A escolha destes *softwares* é respaldada por quatro razões fundamentais: i) gratuidade, ii) compatibilidade, iii) recursos avançados e iv) precisão.

O primeiro motivo da seleção destes *softwares* é a gratuidade de suas versões, o que os torna acessíveis e economicamente vantajosos para o projeto. Isto elimina a necessidade de despesas adicionais com licenças de *software*. Ambos demonstraram compatibilidade eficaz com as impressoras Creality Ender 3 e ZMorph VX. A capacidade de integração harmoniosa entre o *software* de fatiamento e a plataforma de impressão é imperativa para assegurar a transição eficiente do projeto digital para a manufatura física. Estes *softwares* oferecem recursos avançados, como a capacidade de personalizar configurações de impressão e ajustar parâmetros para atender às necessidades específicas do projeto, o que contribui para a flexibilidade do processo de manufatura aditiva. Por fim, a capacidade destes *softwares* de gerar camadas com elevada qualidade e precisão desempenha o papel crítico na obtenção de peças com acabamento satisfatório e conformidade estrita com as especificações de *design*. As Figuras 4.3 e 4.4, ilustram a área de trabalho dos fatiadores de desenhos Creality Slicer e Voxelizer, respectivamente.

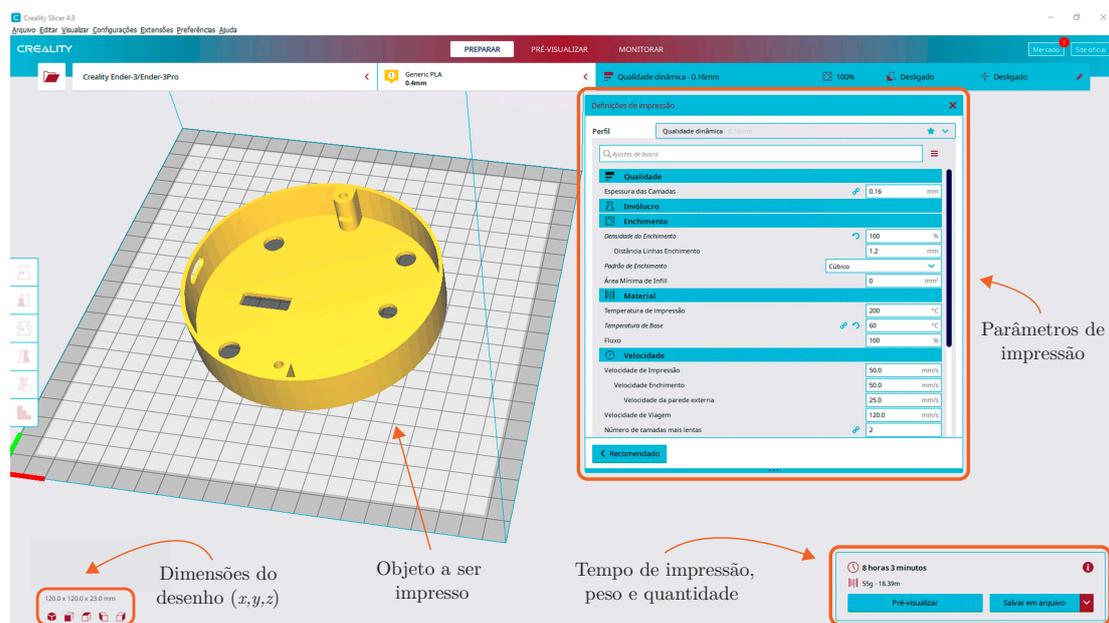


Figura 4.3 - Área de trabalho do fatiador CREALITY Slicer.

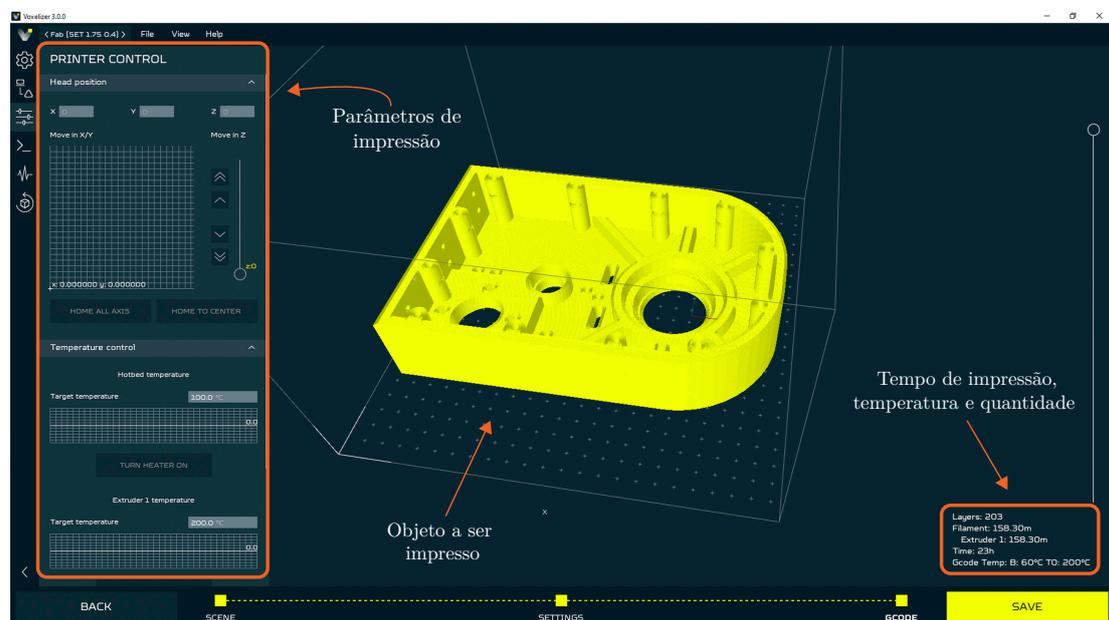


Figura 4.4 - Área de trabalho do fatiador Voxelizer.

Em ambas figuras é perceptível áreas dedicadas para elucidação de dados como

dimensões do desenho, medidas em milímetros [mm], a área de impressão do objeto em cada impressora, o tempo de impressão, medido em horas e minutos, o peso que a peça obterá após sua impressão, medido em gramas [g], a quantidade de filamento que será gasto com a impressão, medido em metros [m] e os parâmetros de impressão 3D, estes parâmetros podem ser modificados conforme o desejo do usuário.

O material selecionado para a impressão 3D na construção do robô foi o ácido polilático (PLA). O PLA é o polímero termoplástico amplamente empregado na manufatura aditiva devido às suas propriedades intrínsecas. Segundo [Barnatt \(2016\)](#), este material é derivado de fontes renováveis, como amido de milho ou cana-de-açúcar, e é reconhecido por sua biocompatibilidade, baixa toxicidade e resistência moderada. Sua natureza biodegradável e a ausência de compostos prejudiciais o tornam ecologicamente viável e seguro para aplicações diversas. A escolha do PLA como material de impressão 3D é respaldada por três motivos relevantes: i) disponibilidade institucional, ii) biocompatibilidade, iii) equilíbrio entre peso e resistência.

A disponibilidade do PLA nas instalações da instituição minimiza a dependência de recursos externos, simplificando o processo de aquisição e garantindo o suprimento consistente, essencial para a conclusão bem-sucedida do projeto. A biocompatibilidade inerente ao PLA, juntamente com sua comprovada segurança para aplicações que envolvem contato com seres humanos ou ambientes sensíveis é relevante quando se trata de construir o robô que pode interagir com seres humanos, garantindo que o material utilizado seja inofensivo para a saúde e atenda às normas regulatórias aplicáveis. O PLA é conhecido por ser leve e, ao mesmo tempo, capaz de proporcionar resistência adequada às peças impressas. Este equilíbrio é utilizado na construção do robô SCARA, de forma que as peças estruturais devem ser suficientemente resistentes para suportar as operações do robô, ao mesmo tempo em que se busca manter o peso adequado para a eficiência do movimento.

A escolha da paleta de cores do PLA para a impressão 3D das partes do robô SCARA, no qual o robô é totalmente preto e o efetuador final é amarelo, foi a decisão que envolve considerações práticas e pedagógicas. As cores possuem impactos significativos no humor, nas decisões humanas, nos sentimentos e até mesmo nas funções cognitivas ([HELLER, 2022](#)).

Primeiramente, a cor preta para o corpo do robô pode ser interpretada sob a ótica da psicologia das cores como representativa de elegância, seriedade e sofisticação. No contexto do robô, a escolha do preto pode transmitir a sensação de confiabilidade e autoridade, o que é desejável ao apresentar conceitos e demonstrações a estudantes

e usuários. Além disso, o preto pode criar o forte contraste visual com outras cores, tornando mais fácil a identificação e a compreensão das partes móveis e dos componentes do robô durante explicações e demonstrações. A cor branca foi escolhida para as polias de forma a facilitar a visualização de sua movimentação juntamente com as correias. Por fim, a cor amarela escolhida para o efetuator final do robô SCARA pode ser associada como representativa de otimismo, criatividade e energia. De acordo com Heller (2022), no contexto didático, o amarelo pode adicionar a atmosfera positiva e estimulante, o que pode ser benéfico para a aprendizagem e a participação dos estudantes. A Figura 4.5, ilustra três rolos de filamento PLA que foram utilizados para a impressão 3D do robô SCARA.

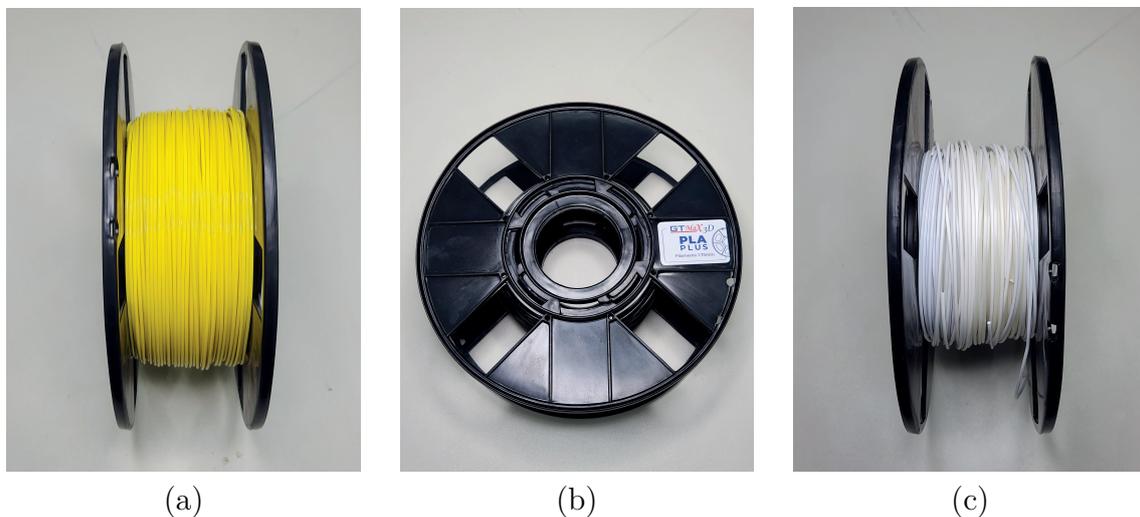


Figura 4.5 - Filamentos de PLA: (a) cor amarela, (b) cor preta e (c) cor branca

Para o PLA, alguns parâmetros devem ser padronizados nos *softwares* de fatiamento de modo a se obter o melhor rendimento e qualidade de impressão. Eles podem ser justificados por considerações técnicas e práticas. A densidade de enchimento de 100% implica que o interior das peças impressas será completamente preenchido com material PLA, sem espaços vazios ou células de estrutura oca. Esta configuração é selecionada quando se busca máxima solidez e robustez na peça final, ideal para aplicações em que a resistência e a integridade estrutural são essenciais (EVANS, 2012).

A temperatura do bico de 200°C é compatível com o PLA e proporciona a temperatura adequada para a fusão e deposição uniforme do material durante a impressão. Manter a temperatura do bico dentro desta faixa é fundamental para evitar pro-

blemas como entupimento ou extrusão irregular do filamento. Esta configuração de temperatura também contribui para a boa adesão entre as camadas, resultando em peças com acabamento mais homogêneo e resistente (GIBSON et al., 2015).

Quanto à temperatura da base de $60^{\circ}C$, esta configuração é selecionada para melhorar a aderência da primeira camada à mesa de impressão. Conforme definido por Lipson e Kurman (2013), o PLA tende a aderir bem a temperaturas mais baixas em comparação com outros materiais, e o aquecimento da base ajuda a evitar que a peça se desprenda durante a impressão. Isto é particularmente relevante quando se imprime peças maiores ou com geometrias complexas, em que a aderência sólida é essencial para evitar defeitos na impressão.

4.1.2.2 Impressão 3D das peças

No processo de impressão 3D das peças que compõem o robô SCARA, adotou-se a abordagem metódica visando a organização e eficiência. Para esta finalidade, foi implementado o sistema de numeração para todas as impressoras disponíveis no instituto, a fim de estabelecer a correlação entre cada impressora e as peças a serem produzidas. A impressora identificada como número um foi designada como a Zmorph VX, enquanto as impressoras numeradas de dois a cinco foram identificadas como as Ender 3. Este arranjo permitiu a distribuição estratégica das peças a serem impressas nas impressoras, otimizando o processo de manufatura.

A numeração das impressoras revelou-se de grande utilidade na criação da tabela que detalhou a organização da impressão de cada peça. Esta tabela disponível no Apêndice B inclui informações essenciais, como os nomes das peças, o número da impressora, no qual IMP é sua abreviatura, o dia e o horário de início e término da impressão, o peso correspondente de cada componente e a cor especificada para as peças. Ela desempenhou o papel fundamental ao longo de todo o processo de impressão 3D, proporcionando controle eficaz e a visão geral precisa do andamento da produção.

Vale ressaltar que a estratégia adotada contemplou a alocação criteriosa das peças, com base em suas características e requisitos de desempenho. Peças que demandavam maior rigidez e resistência devido às elevadas exigências mecânicas durante a operação do robô, tais como polias e elos, foram deliberadamente destinadas à impressora Zmorph VX. Por outro lado, componentes de complexidade mais reduzida, como tampas e o efetuador final, foram direcionados às impressoras Ender 3. Além disso, a alocação estratégica das peças contribuiu para otimizar o tempo de produção

e a utilização dos recursos disponíveis, demonstrando o planejamento cuidadoso e a gestão do processo de impressão 3D. A Figura 4.6, ilustra o processo de impressão 3D de duas peças do robô.

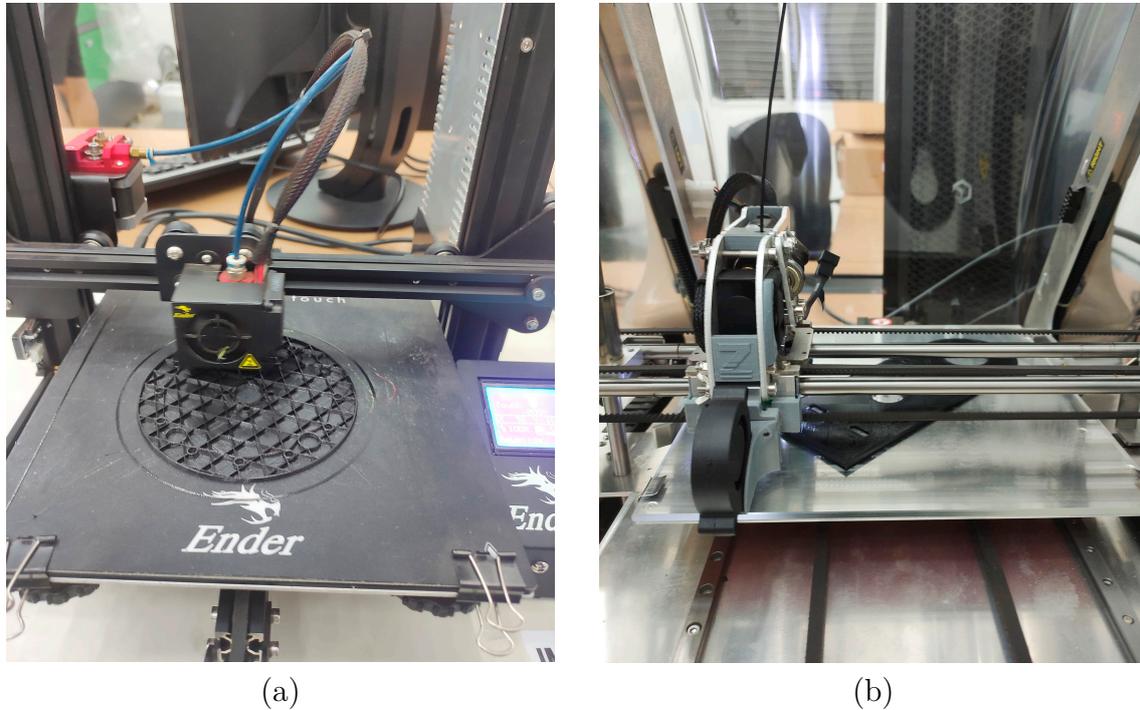


Figura 4.6 - Fabricação de peças: (a) Acoplador J2 e (b) Braço 01.

A representação da esquerda (a) demonstra o procedimento de impressão 3D da peça de acoplamento da junta J2. A mencionada operação de impressão foi executada por meio da utilização da impressora Creality Ender 3. A representação em desenho da peça resultante deste processo de fabricação, encontra-se disponível no Apêndice A, na prancheta A.10. Por outro lado, a ilustração à direita (b) apresenta o processo de impressão 3D da peça do primeiro elo, contudo, neste caso, a impressão foi conduzida utilizando a impressora ZMorph VX. O desenho desta peça resultante deste processo também está disponível para visualização no apêndice, especificamente na prancheta A.4. Ademais, a Figura 4.7, ilustra outras peças após o processo de impressão 3D completamente finalizadas.



Figura 4.7 - Peças 3D finalizadas: (a) Braço 02 e (b) Base do robô.

A Figura (a) exibe a peça do segundo elo do robô após a conclusão de seu processo de impressão 3D. A representação esquemática desta peça encontra-se disponível no Apêndice A, na prancheta A.6. A Figura (b) ilustra a peça correspondente à base do robô, também após a conclusão do processo de impressão tridimensional. O desenho resultante desta peça pode ser visualizado na prancheta A.8 do apêndice. Cabe destacar que o peso total de todas as peças impressas totalizou aproximadamente 1.795g. O peso de cada peça pode ser consultado através da Tabela B.2.

4.1.3 Seleção dos componentes mecânicos

O desenvolvimento do robô SCARA requer cuidadosa seleção e integração de componentes mecânicos, fundamentais para assegurar seu desempenho, precisão e confiabilidade. Cada escolha é pautada em critérios técnicos específicos, visando atender às demandas do projeto e garantir a operação eficiente do robô. Foram utilizados componentes escolhidos para a montagem mecânica do robô: i) hastes lisas de seção circular, ii) fuso de avanço, iii) acoplador de eixo, iv) rolamentos lineares, v) rolamentos de encosto de esferas² são frequentemente utilizados em aplicações industriais, vi) rolamentos radiais de esferas e vii) correias GT2 e viii) malha náutica.

As hastes lisas de seção circular são elementos estruturais utilizados em aplicações mecânicas para transmitir cargas e suportar movimentos lineares. Elas são caracterizadas por sua seção transversal circular uniforme e podem variar em diâmetro (BUDYNAS; NISBETT, 2011). A utilização de hastes lisas de seção circular com dimensões de $10 \times 400\text{mm}$ e $6 \times 150\text{mm}$ foi motivada pela necessidade de componentes estruturais resistentes e rígidos. As hastes de $10 \times 400\text{mm}$ foram utilizadas

²Os rolamentos de encosto de esferas, também são conhecidos como rolamentos axiais de esferas. Eles são projetados para suportar cargas axiais, não radiais.

para suportar toda a estrutura prismática do eixo z , elas proporcionam a base sólida e robusta para o robô, garantindo sua estabilidade e capacidade de suportar cargas substanciais. Por outro lado, as hastes de $6 \times 150mm$ foram selecionadas para a componente estrutural do efetuador final. A Figura 4.8, ilustra as duas hastes selecionadas para a montagem mecânica do robô.

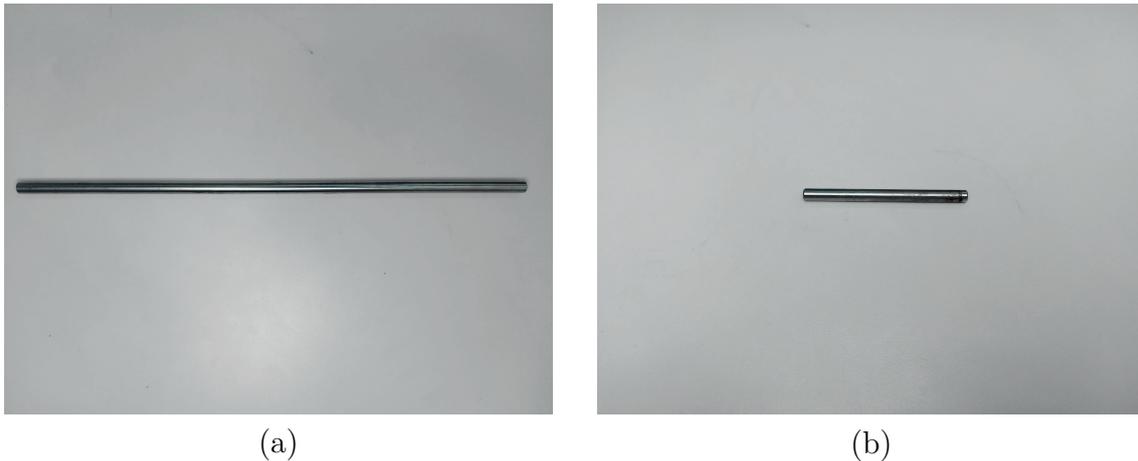


Figura 4.8 - Hastes de seção circular: (a) haste de $10 \times 400mm$ e (b) haste de $6 \times 150mm$.

Segundo Mott et al. (2017), o fuso de avanço é o dispositivo mecânico utilizado para converter o movimento de rotação em movimento linear. Isto é alcançado por meio da rosca presente no fuso, que interage com o componente móvel, permitindo que ele se desloque ao longo do eixo do fuso de maneira precisa e controlada. O acoplador de eixo é o componente utilizado para conectar dois eixos de diferentes tamanhos ou tipos, permitindo a transmissão eficiente de torque e movimento entre eles (PFEIFFER, 2008). A escolha do fuso de avanço com dimensões de $8 \times 400mm$ foi guiada pela necessidade de transmitir o movimento linear de alta precisão do motor de passo utilizado para o movimento vertical prismático do robô. Para garantir a transmissão eficaz da rotação do motor para o fuso, optou-se pelo acoplador de eixo de $5 \times 8mm$, que proporciona a conexão precisa e sem folgas. Por fim, a polia GT2 de 20 dentes foi acoplada ao eixo de $5mm$ do motor de passo. Na extremidade desta polia encontra-se o acoplador de eixo. A Figura 4.9, ilustra o fuso de avanço utilizado, o acoplador de eixo do motor de passo e a polia GT2 de 20 dentes utilizada também no eixo do motor.

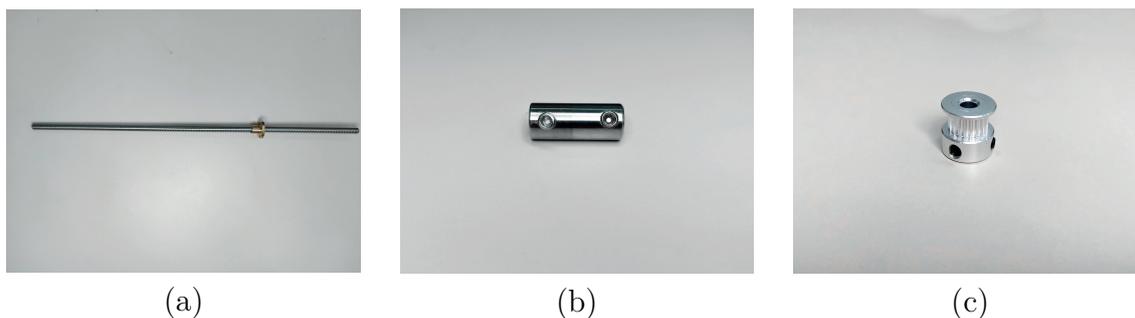


Figura 4.9 - Sistema de movimentação linear: (a) fuso de avanço de $8 \times 400mm$, (b) acoplador de eixo de $5 \times 8mm$ e (c) polia GT2 com 20 dentes.

Os rolamentos são elementos fundamentais em sistemas mecânicos para reduzir o atrito e permitir movimentos suaves. Os rolamentos lineares são projetados para suportar movimentos lineares, enquanto os rolamentos de encosto são utilizados para suportar cargas axiais e os rolamentos radiais suportam cargas radiais (PFEIFFER, 2008). Os rolamentos lineares de $10mm$ foram escolhidos para guiar e suportar movimentos lineares, minimizando o desgaste e a imprecisão. Estes rolamentos serão utilizados como suporte para as hastes de seção circular de $10mm$ de largura. Os rolamentos de encosto de esferas foram empregados para suportar cargas axiais, realizando o movimento de rotação da base e dos elos, enquanto os rolamentos radiais oferecem suporte para cargas radiais, sendo utilizados para o acoplamento da extremidade do fuso de avanço. A Figura 4.10, ilustra os três tipos de rolamentos utilizados na montagem mecânica do robô.

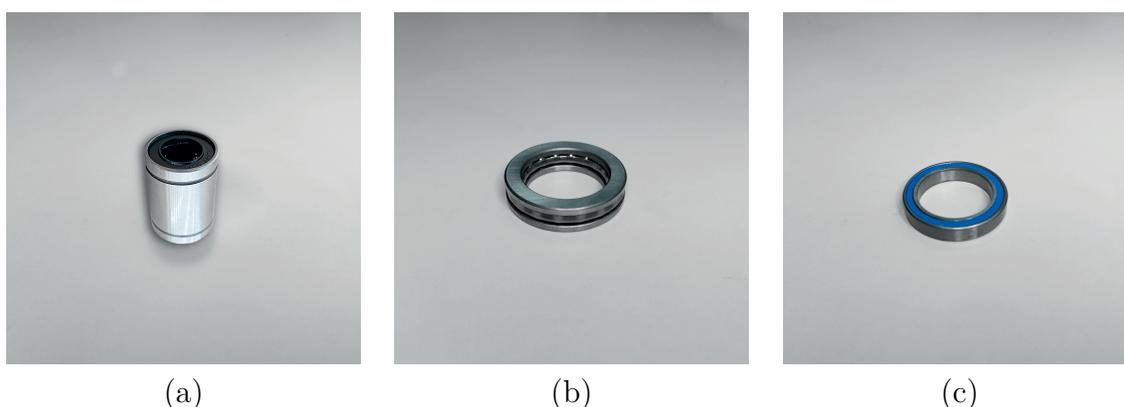


Figura 4.10 - Rolamentos utilizados: (a) rolamento linear, (b) rolamento axial e (c) rolamento radial.

De acordo com Budynas e Nisbett (2011), as correias GT2 são utilizadas em sistemas de transmissão de movimento para transferir rotação de polia para outra. Elas são conhecidas por sua eficiência e capacidade de fornecer movimento preciso e confiável. Os tensores de polias GT2 são utilizados para manter a tensão adequada nas correias, garantindo o funcionamento suave e livre de deslizamento. As correias GT2 foram escolhidas devido à sua eficiência e confiabilidade comprovadas. Os diferentes tamanhos de correias foram empregados para se adequar às necessidades específicas de cada seção do robô, proporcionando flexibilidade de projeto e garantindo a transmissão precisa de movimento. Estas correias serão responsáveis por realizar o movimento de rotação dos elos através da transmissão e redução das polias. Os tensores de polias GT2 de $6mm$ ajudam a manter a tensão adequada nas correias, contribuindo para a precisão dos movimentos. A Figura 4.11, ilustra os três tipos de correias GT2 utilizadas. Estas correias possuem a largura de $6mm$ e diferentes comprimentos, todos medidos em milímetros [mm].



Figura 4.11 - Correias GT2: (a) correia $6 \times 400mm$, (b) correia $6 \times 300mm$ e (c) correia $6 \times 200mm$.

A malha náutica é o material versátil usado para o acabamento estético e a organização de fios em sistemas mecânicos (MOTT et al., 2017). Ela protege os fios contra danos mecânicos, ajuda a evitar interferências indesejadas nos movimentos do sistema e contribui para o aspecto profissional e organizado. A malha náutica foi escolhida para o acabamento estético do robô, proporcionando visual profissional e organizado. Além disso, ela desempenha o papel na organização e proteção dos fios elétricos, evitando interferências indesejadas nos movimentos do robô. A Figura 4.12, ilustra a malha náutica e a abraçadeira de *nylon* utilizadas para o acabamento estético do robô.



(a)



(b)

Figura 4.12 - Acabamento do robô: (a) malha náutica e (b) abraçadeira de *nylon*.

Como forma de organização e registro sistemático dos materiais mecânicos utilizados na montagem do robô SCARA, foi elaborada a lista de todos os componentes selecionados e quantidades correspondentes. Desta forma, a lista de materiais selecionados está disposta na Tabela 4.1, no qual QTD é abreviatura para quantidade e UN é abreviatura para unidade.

Tabela 4.1 - Lista de materiais mecânicos.

ITEM	DESCRIÇÃO	MEDIDAS	QTD	UN
1	Haste de metal liso	10 × 400mm	4	PÇ
2	Haste de metal liso	6 × 150mm	2	PÇ
3	Fuso de avanço	8 × 400mm	1	PÇ
4	Rolamento linear	10mm	4	PÇ
5	Rolamento axial de esferas	40 × 60 × 13mm	2	PÇ
6	Rolamento axial de esferas	35 × 52 × 12mm	4	PÇ
7	Rolamento radial de esferas	8 × 22 × 7mm	5	PÇ
8	Rolamento radial de esferas	35 × 47mm	1	PÇ
9	Rolamento radial de esferas	30 × 42mm	2	PÇ
10	Acoplador de eixo de motor	5 × 8mm	1	PÇ
11	Correia GT2	6 × 400mm	2	PÇ
12	Correia GT2	6 × 300mm	2	PÇ
13	Correia GT2	6 × 200mm	1	PÇ
14	Polia GT2 de alumínio	6mm × 20T	3	PÇ
15	Tensor de polias GT2	6mm - lisa	6	PÇ
16	Malha náutica	10 × 2000mm	1	PÇ

4.1.4 Processo de montagem do robô

O processo de montagem do robô foi dividido em cinco etapas: i) montagem da base, ii) montagem do primeiro elo, iii) montagem do segundo elo, iv) montagem da junta do eixo z e v) montagem do efetuator final. A Figura 4.13, ilustra o fluxograma de desenvolvimento do processo de montagem mecânica do robô SCARA.

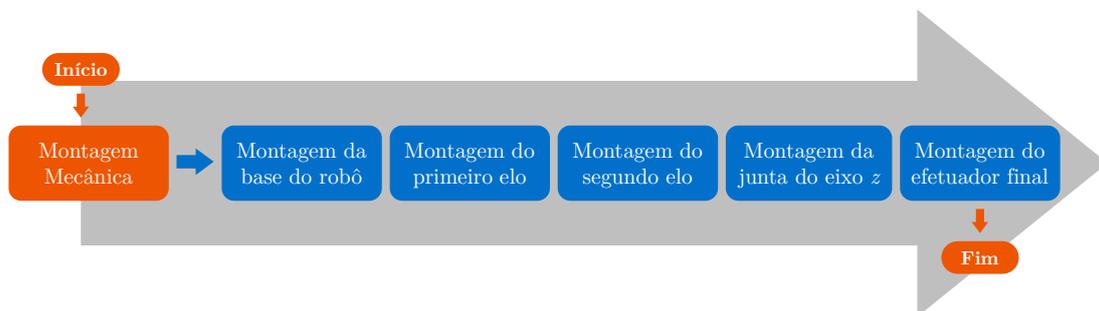
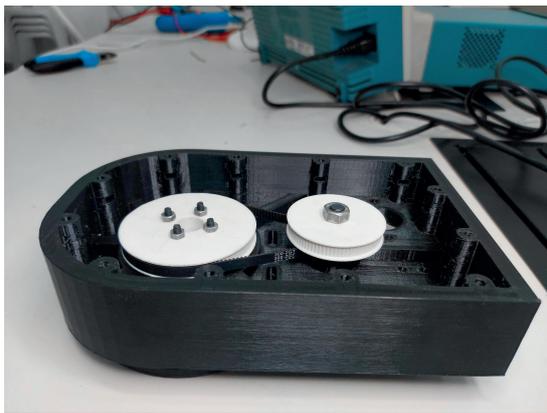


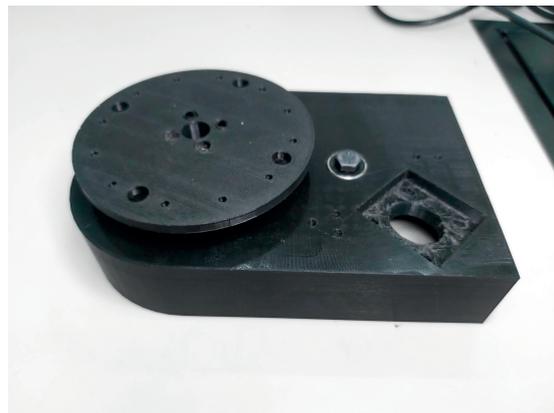
Figura 4.13 - Fluxograma da etapa de montagem mecânica do robô.

A montagem começa com a base do robô, na qual insere-se inicialmente o rolamento

de esferas radiais com dimensões de $35 \times 47mm$ de diâmetro, seguido pela colocação do rolamento de esferas de encosto com dimensões de $40 \times 60mm$ de diâmetro, que se posiciona entre a polia e a base. O segundo rolamento de esferas de encosto de mesmo tamanho é utilizado no lado oposto da base, juntamente com o acoplador da articulação. A montagem da polia e da parte superior é realizada usando quatro parafusos M4 de $50mm$ de comprimento, com a adição de porcas autotravantes para garantir a fixação segura que ainda permita rotação livre. A polia do meio é instalada subsequente, acoplada à polia da junta com uma correia GT2 de $300mm$, utilizando dois rolamentos de esferas com dimensões de $8 \times 22mm$ de diâmetro. A fixação da polia se dá por um parafuso M8 de $45mm$, acompanhado de uma arruela e uma porca autotravante. O motor de passo para esta junta é então instalado, acoplado à polia do meio com a correia de $200mm$, fixado à base com quatro parafusos M3. A Figura 4.14, ilustra o processo de montagem do conjunto da base do robô.



(a)



(b)

Figura 4.14 - Montagem da base: (a) conjunto base e polia e (b) base completa.

O primeiro elo do robô é montado a partir de duas partes, unidas por quatro parafusos M5 de $20mm$. Quatro rolamentos lineares são instalados nestas partes para deslizar pelas hastes de $10mm$. O segundo motor de passo é instalado com a polia GT2 de 20 dentes, seguido pela instalação das correias e polias para a segunda articulação, exigindo correias de $400mm$ e $300mm$ de comprimento. Para as articulações subsequentes, são utilizados rolamentos menores, um radial com dimensões de $30 \times 42mm$ de diâmetro e um de encosto de esferas com dimensões de $35 \times 52mm$. Seis parafusos M4 de $20mm$ são inseridos para instalar o acoplador da articulação, e o segundo braço é fixado ao acoplador com parafusos previamente instalados. A

Figura 4.15, ilustra o processo de montagem do conjunto do primeiro e segundo elo do robô.

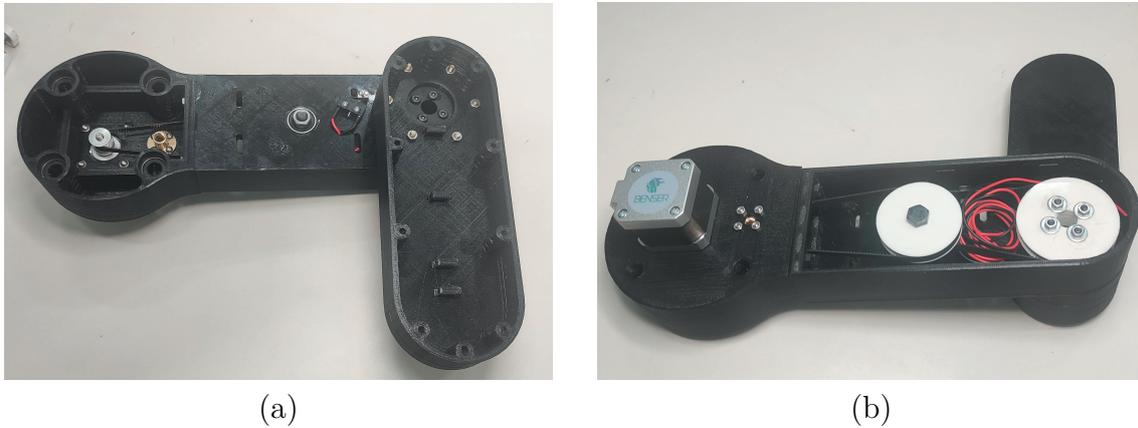
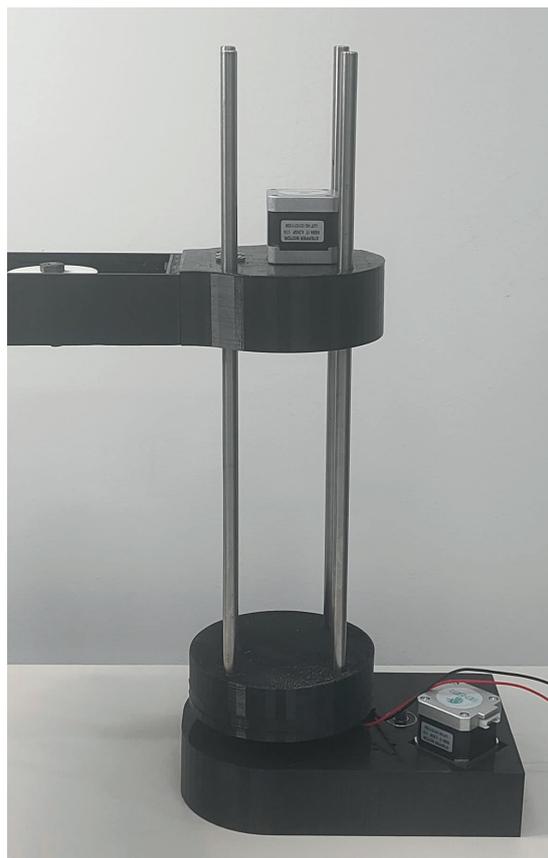


Figura 4.15 - Montagem dos elos: (a) vista inferior e (b) vista superior.

A montagem do eixo z inicia-se com a fixação da parte inferior da placa do eixo z ao acoplador da primeira junta, seguida pela instalação de quatro abraçadeiras das hastes de 10mm , no qual elas são inseridas e firmadas com parafusos M4 e porcas. O rolamento radial com dimensões de $8 \times 22\text{mm}$ é inserido para o fuso de avanço, finalizando com a colocação da tampa para cobrir os componentes e proporcionar estética ao robô. A Figura 4.16, ilustra o processo de montagem do conjunto da base de movimentação linear prismática para o robô SCARA.



(a)



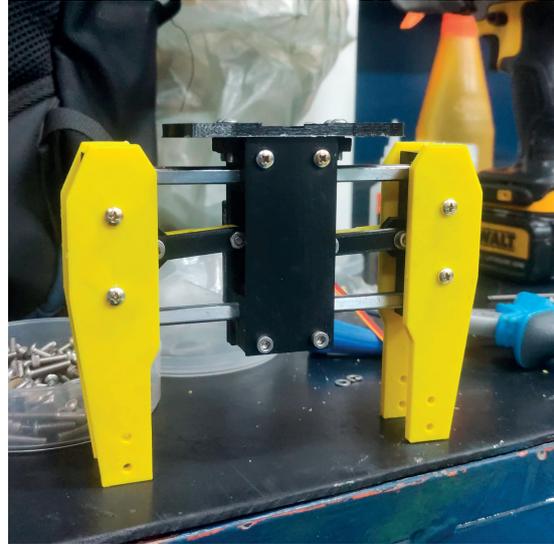
(b)

Figura 4.16 - Montagem do eixo z: (a) montagem parcial e (b) montagem com o elo.

O efetuator final, é acionado pelo servo motor, fixado em duas hastes de metal com dimensões de $6 \times 150mm$. As partes deslizantes da garra são conectadas ao servo com seu braço de articulação, ligações da garra e parafusos e porcas M3. Este *design* permite a troca fácil das extremidades da garra, ajustando-se a necessidades específicas. A garra é fixada ao braço do robô com parafusos e porcas M4, finalizando sua montagem mecânica. A Figura 4.17, ilustra o processo de montagem do efetuator final do robô SCARA.



(a)



(b)

Figura 4.17 - Montagem do efetuator final: (a) peças utilizadas e (b) efetuator finalizado.

A Tabela 4.2 contém a lista com todos os parafusos utilizados para a fixação dos componentes do robô, no qual QTD é abreviatura para quantidade e UN é abreviatura para unidade. Parafusos M3 foram utilizados principalmente no efetuator final, enquanto os M4 foram destinados para a montagem das polias. Os parafusos M5 participaram da união dos elos, e os M8 garantiram a robustez e estabilidade da base do robô.

Tabela 4.2 - Lista de parafusos para montagem.

ITEM	DESCRIÇÃO	MEDIDAS	QTD	UN
1	Parafuso M3	$3 \times 12/10mm$	4	PÇ
2	Parafuso M3	$3 \times 14/12mm$	8	PÇ
3	Parafuso M3	$3 \times 16mm$	2	PÇ
4	Parafuso M3	$3 \times 20mm$	6	PÇ
5	Parafuso M3	$3 \times 30mm$	10	PÇ
6	Parafuso M4	$4 \times 16mm$	4	PÇ
7	Parafuso M4	$4 \times 20mm$	14	PÇ
8	Parafuso M4	$4 \times 25mm$	24	PÇ
9	Parafuso M4	$4 \times 50mm$	8	PÇ
10	Parafuso M5	$5 \times 20mm$	4	PÇ
11	Parafuso M5	$5 \times 35mm$	12	PÇ
12	Parafuso M8	$8 \times 45mm$	2	PÇ

4.1.5 Sistema de transmissão

O mecanismo de transmissão adotado compreende o arranjo de polias e correias, sendo imperativa a implementação de correias dentadas para mitigar possíveis deslizamentos entre a correia e a polia, condição que poderia comprometer a precisão do posicionamento do manipulador. Por razões pragmáticas, a escolha recaiu sobre correias dentadas do tipo GT2, caracterizadas pelo passo de $2mm$ entre os dentes. A decisão fundamentou-se na acessibilidade generalizada deste modelo de correia e na sua eficácia comprovada em dispositivos com requisitos de torque similares, como cortadoras a laser e impressoras 3D. Além disso, as correias da série GT2 são indicadas pelo fabricante para atuar em baixos níveis de potência, alto torque e baixas velocidades (GATES, 2020).

Conforme as especificações do *design*, foram determinadas três variantes de correias sincronizadas GT2, com comprimentos de $400mm$, $300mm$ e $200mm$, para o acionamento das articulações. A correia de $200mm$ juntamente com a de $300mm$ foram designadas para a ativação da primeira articulação, alinhando a relação de redução desejada com a necessidade de posicionar o motor adjacente à articulação sem comprometer o espaço operacional. A primeira articulação é composta pelas polias GT2 com 110 e 80 dentes.

A correia de $400mm$ foi destinada à segunda articulação, mantendo a distância aproximada de $200mm$ entre o fuso de avanço do eixo z e a referida articulação. Além disso, a segunda correia de $300mm$ foi utilizada para a transmissão de movimento da segunda polia GT2 com 80 dentes para a outra polia da segunda articulação que é composta pela polia GT2 com 92 dentes. Finalmente, a segunda correia de $400mm$ foi aplicada para a terceira articulação, visando minimizar a distância entre a segunda e a terceira articulações para cerca de $200mm$. A terceira articulação é composta pela polia GT2 com 90 dentes. Ademais, tensores de correia GT2 foram integrados para assegurar a tensão adequada das correias no local em que foi detectada folga significativa.

No que se refere ao conjunto de polias do robô SCARA, cada junta rotacional incorpora correias sincronizadas e polias. Nas duas primeiras articulações, são empregadas duas etapas de redução para obter relações de 20:1 e 16:1, respectivamente. A terceira articulação opera com a relação de redução de 4:1 em único estágio. A representação visual do sistema de transmissão das juntas do robô SCARA, caracterizada pela perspectiva isométrica e transparente do manipulador, possibilita a inspeção de todos os componentes internos relacionados às correias e polias, conforme ilustrado

na Figura 4.18, adaptada de Nedelkovski (2020).

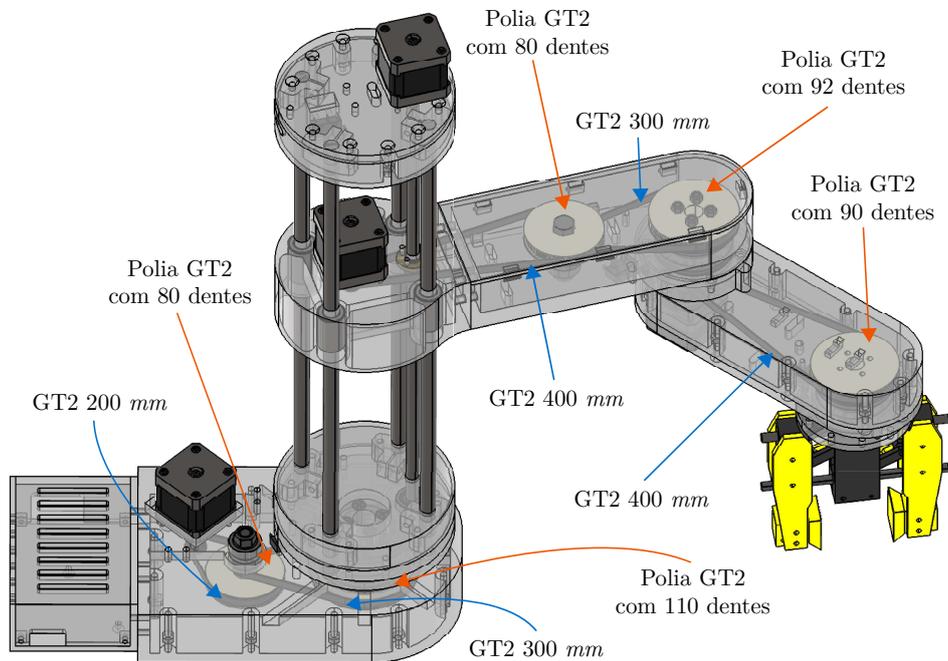


Figura 4.18 - Sistema de transmissão do robô SCARA.

4.1.6 Seleção dos rolamentos

Para facilitar os movimentos articulares do robô, incorporaram-se rolamentos com o intuito de minimizar o atrito nas partes móveis. As articulações do robô estão equipadas tanto com rolamentos axiais quanto radiais, proporcionando assim suporte adequado para cargas axiais e radiais durante a operação. Adicionalmente, o deslocamento no eixo z é facilitado por quatro eixos lineares de 10mm , complementados por quatro rolamentos lineares que contribuem para o movimento estrutural. Cada articulação é composta por dois rolamentos axiais e um radial, enquanto o mecanismo de transmissão no eixo z emprega o fuso de avanço para transformar movimento rotacional em linear.

Para assegurar a fluidez na interação entre os componentes articulados, optou-se por rolamentos de esferas, que mediam o contato entre os elos e outras partes com movimento relativo. Dada a natureza das cargas envolvidas, de magnitude reduzida, a seleção dos rolamentos pautou-se nas especificações dimensionais do projeto e na

consulta a catálogos técnicos, especificamente o referenciado por SKF (2015). Para a base, utilizou-se o rolamento de esferas radiais com o código de fabricante 6807zz, acompanhado de rolamentos de esferas de encosto, identificados pelo código 51108. A estrutura superior da base também integra o rolamento 51108, em conjunto com o acoplador da articulação. A montagem do eixo z envolveu quatro rolamentos lineares, modelo LM10UU, destinados a facilitar o movimento linear. Nas articulações, a montagem requereu rolamentos de esferas radiais, modelo 6806zz, e rolamentos de esferas de encosto, com código 51107. Para a instalação das polias, foram selecionados rolamentos de esferas modelo 608rs.

4.1.6.1 Cálculo de duração dos rolamentos

Segundo Shigley e Mischke (1996), a vida útil dos rolamento representa a duração operacional que ele pode proporcionar, considerando as condições de trabalho previamente definidas. Ela está intimamente relacionada ao número estimado de rotações que o rolamento pode realizar antes de manifestar sinais de desgaste, conhecido como escamamento, e é considerado como o critério principal para determinar a vida útil do rolamento.

No catálogo do fabricante SKF (2015), são apresentados os parâmetros fundamentais para a seleção e cálculo da vida útil dos rolamentos. Na metodologia de seleção de rolamentos de esferas, o fabricante estabelece a correlação entre a vida em milhões de rotações L_{10} , medido em rotações de dez elevado a sexta potência [10^6 rotações], a carga dinâmica equivalente no rolamento P_d , medida em newtons [N] e a capacidade de carga dinâmica C_d , medida em newtons [N], conforme estabelecido na expressão (4.1):

$$L_{10} = \left(\frac{C_d}{P_d} \right)^3 \quad (4.1)$$

De acordo com a norma ISO 281 (2010), a capacidade de carga básica dinâmica C_d , medido em newtons [N], que reflete a habilidade do rolamento em suportar carga, é definida como a carga constante, em direção e intensidade, que resulta na vida nominal de um milhão de rotações (10^6 rotações). Isto ocorre sob a condição em que o anel interno está em movimento e o anel externo está em repouso. Em rolamentos radiais, a carga considerada é radial, central, e de intensidade constante. Já nos rolamentos axiais, a carga avaliada é axial, coincidindo com o eixo central, e também de direção e intensidade constantes (SHIGLEY; MISCHKE, 1996).

A carga dinâmica equivalente no rolamento P_d , medida em newtons $[N]$, afeta os rolamentos que podem ser predominantemente radiais ou axiais em alguns casos (ISO 281, 2010). No entanto, é mais comum lidar com aplicações em que as cargas radiais e axiais agem simultaneamente. Além disso, estas cargas podem variar em intensidade e direção. Em situações complexas como estas, torna-se impraticável usar diretamente a carga aplicada para calcular a vida útil do rolamento. Sob tais condições, torna-se imprescindível estimar a carga hipotética que atue no centro do rolamento, mantendo sua magnitude inalterada. Esta abordagem permite determinar a vida útil teórica do rolamento, que corresponde à sua duração efetiva sob variadas condições de carga e velocidade de rotação, conforme descrito por Budynas e Nisbett (2011). A referida carga hipotética é denominada como carga dinâmica equivalente, esta que pode ser determinada através da equação (4.2):

$$P_d = X_i \cdot F_{ra} + Y_i \cdot F_{ax} \quad (4.2)$$

no qual P_d é a carga dinâmica equivalente, medida em newtons $[N]$, os coeficientes X_i e Y_i ajustam a contribuição relativa das cargas axial e radial, estes são consultados através do catálogo de fabricante SKF (2015), F_{ra} e F_{ax} correspondem as cargas axiais e radiais necessárias ao rolamento, respectivamente, ambas são medidas em newtons $[N]$.

A carga dinâmica equivalente P_d é utilizada na fórmula de vida útil, expressa na equação (4.1), fornecendo a estimativa teórica de quantas rotações o rolamento pode suportar.

Consultando a capacidade de carga dinâmica do rolamento C_d e calculando a carga equivalente pela equação (4.1), pode-se estimar a vida útil do rolamento em milhões de rotações. Os resultados obtidos estão expressos através das Tabelas 4.3 e 4.4.

Tabela 4.3 - Estimativa de vida dos rolamentos - Parte I.

VARIÁVEIS	1º JUNTA - 6807zz	2º JUNTA - 6806zz
F_{ra} (N)	170	100
F_{ax} (N)	210	210
P_d (N)	370	310
C_d (N)	4360	4490
L_{10} (10^6 rotações)	$167,80 \times 10^3$	$298,96 \times 10^3$

Tabela 4.4 - Estimativa de vida dos rolamentos - Parte II.

VARIÁVEIS	AXIAL - 51108	AXIAL - 51107	608rs
F_{ra} (N)	0	0	50
F_{ax} (N)	210	210	100
P_d (N)	210	210	150
C_d (N)	25500	19900	3450
L_{10} (10^6 rotações)	$184,19 \times 10^3$	$871,10 \times 10^3$	$121,67 \times 10^3$

A Figura 4.19, ilustra a vista isométrica explodida³ do robô SCARA, permitindo a análise detalhada de todos os componentes, tanto internos quanto externos, que constituem a estrutura do manipulador. Esta representação inclui elementos como rolamentos, hastes, correias, polias, parafusos, e componentes impressos em 3D necessários para a montagem mecânica do dispositivo. Para o exame mais aprofundado da localização de cada elemento mecânico e do processo de montagem do robô, recomenda-se a consulta às pranchetas de desenho contidas no Apêndice A.4.

³A vista explodida é a representação gráfica de objetos que demonstra a relação espacial entre todas as suas partes, dispostas de maneira a sugerir a sequência de montagem ou desmontagem.

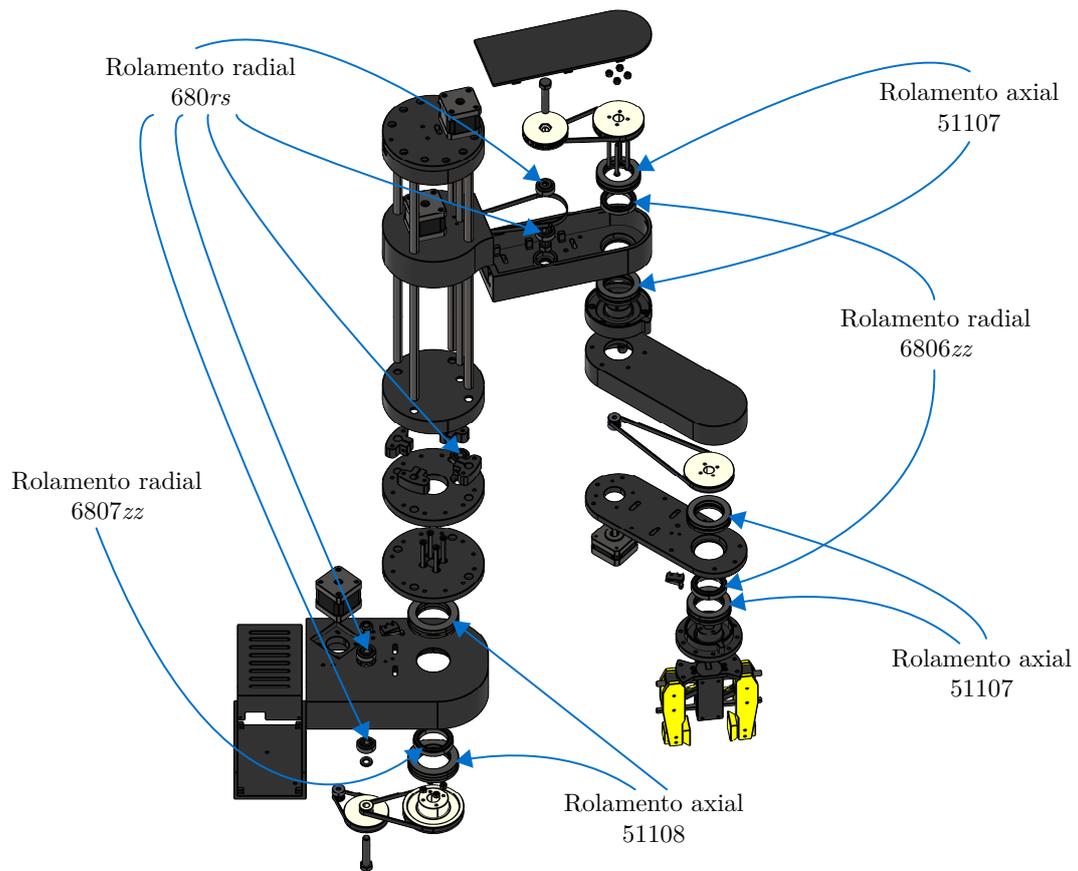


Figura 4.19 - Vista isométrica explodida do robô SCARA.

4.1.7 Resolução dos cálculos estáticos

Nesta seção, é feita a análise e resolução dos cálculos estáticos referentes aos elos do robô SCARA. Foi realizada a avaliação dos primeiros elos do manipulador como elos consolidados do material PLA, caracterizado pela seção transversal de $42,7 \times 31\text{mm}$ e espessura de $4,11\text{mm}$, e considerando os parâmetros estabelecidos na Tabela 4.5 para as variáveis pertinentes, é factível determinar o valor máximo da seção de comprimento do eixo z , intitulado z_i . Nesta tabela, UN é abreviatura para unidade.

Tabela 4.5 - Substituição de variáveis estáticas.

VARIÁVEL	VALOR	UN
l_i	0,427	m
x_i	0,427	m
E_i	$2,4 \times 10^9$	Pa
I_{yi}	$0,72 \times 10^{-7}$	m^4
q_i	39,226	N/m
P_i	15	N
$z_i(l_i)$	$-3,19 \times 10^{-6}$	m

Foram realizados os cálculos das tensões normal e de cisalhamento máximas com o propósito de validar a aplicação da equação da linha elástica. Esta abordagem é condicional à confirmação de que o material permanece na região elástica, ou seja, abaixo da tensão de escoamento. No caso específico do PLA, esta tensão de escoamento é de $2,4 \times 10^9 Pa$ (LIPSON; KURMAN, 2013). Os resultados da análise, conforme apresentado na Tabela 4.6 revela que as tensões calculadas estão significativamente abaixo deste limiar, corroborando a validade da utilização da equação da linha elástica. Na Tabela 4.6, σ_{mi} é a tensão normal máxima, medida em pascal [Pa] e τ_{mi} corresponde a tensão de cisalhamento máxima na estrutura, também medida em pascal [Pa].

Tabela 4.6 - Tensões normal e de cisalhamento máximos.

VARIÁVEL	VALOR	UN
σ_{mi}	$2,14 \times 10^6$	Pa
τ_{mi}	2,85	Pa

4.1.8 Resolução dos cálculos dinâmicos

Nesta seção, é realizada a análise e resolução dos cálculos dinâmicos. Com base nos resultados expostos, foi desenvolvida a rotina para o cálculo dos torques, os quais seriam alimentados com dados provenientes da modelagem do manipulador. Parâmetros como massa, posição do centro de massa e momentos de inércia relativos à rotação de cada elo foram adquiridos mediante a atribuição das propriedades de material específicas a cada componente durante através da modelagem realizada em *software*. A Tabela 4.7, dispõe dos valores das massas específicas empregadas

na modelagem do robô. Nesta tabela, UN é abreviatura para unidade e ESP é a abreviatura para específica, de massa específica.

Tabela 4.7 - Propriedades de massa atribuídas.

MATERIAL	MASSA ESP	UN
Ácido polilático (PLA)	1240	kg/m^3
Motor de passo NEMA 17	0,32	kg

A Tabela 4.8, apresenta os resultados calculados das velocidades e acelerações angulares desejadas para cada junta. A escolha destes parâmetros foi realizada para equilibrar as necessidades operacionais, garantindo simultaneamente a segurança durante a execução das operações do manipulador. Nesta tabela, UN é abreviatura para unidade.

Tabela 4.8 - Variáveis de velocidade e aceleração atribuídas.

VARIÁVEL	VALOR	UN
Velocidade da primeira junta do robô (V_{r1})	0,4	m/s
Velocidade angular da primeira junta ($V_{\dot{\theta}_1}$)	1	rad/s
Aceleração angular da primeira junta ($A_{\ddot{\theta}_1}$)	2	rad/s^2
Velocidade da segunda junta do robô (V_{r2})	0,3	m/s
Velocidade angular da segunda junta ($V_{\dot{\theta}_2}$)	2	rad/s
Aceleração angular da segunda junta ($A_{\ddot{\theta}_2}$)	4	rad/s^2
Velocidade da junta prismática (V_{r3})	0,1	m/s
Aceleração da junta prismática (A_{r3})	0,1	m/s^2

4.1.8.1 Resultados de torques para as juntas

A fim de determinar o torque necessário para os motores que acionam as três juntas, conduziu-se a análise simplificada dos esforços envolvidos. A equação (4.3) apresenta esta análise de força exercida nos motores de passo:

$$F_t = m_e \cdot a_t \quad (4.3)$$

na qual F_t é a força necessária no motor de passo, medida em newtons [N], m_e é a

massa dos elos, medida em newtons $[N]$ e a_t é a aceleração do motor de passo, medida em metros por segundo ao quadrado $[m/s^2]$. A expressão (4.4) é responsável pelo cálculo de torque no motor de passo responsável pelo movimento do eixo prismático.

$$\tau_t = F_e \cdot d_m \quad (4.4)$$

no qual τ_t , é o torque necessário para levantar os elos, medido em newton-metro $[N \cdot m]$, F_e é a força necessária para levantar os elos, medida em newton $[N]$ e d_m distância do centro de massa, medida em metros $[m]$.

Através da análise dinâmica para a primeira junta, segunda junta e junta prismática τ_{nz} , expostas pelas equações (3.34) e (3.38), respectivamente, realizaram-se iterações nos cálculos, buscando harmonizar os resultados de torque com o *design* dos eixos, elos, juntas, e a seleção dos motores e outros componentes. Ao final deste processo foram alcançados os resultados apresentados, refletindo a integração coesa entre os requisitos de torque e o dimensionamento dos elementos estruturais e motores do sistema. A Tabela 4.9, apresenta o resultado dos cálculos de torque para as três juntas disponíveis no robô. Nesta tabela, UN é abreviatura para unidade. Por fim, a Figura 4.20, ilustra o gráfico que contém as curvas de torques de cada junta apresentada.

Tabela 4.9 - Resultados da análise dinâmica do robô.

VARIÁVEL	MÓDULO MÁXIMO	UN
τ_{n1}	0,4	$N \cdot m$
τ_{n2}	0,2	$N \cdot m$
τ_{nz}	0,86	$N \cdot m$

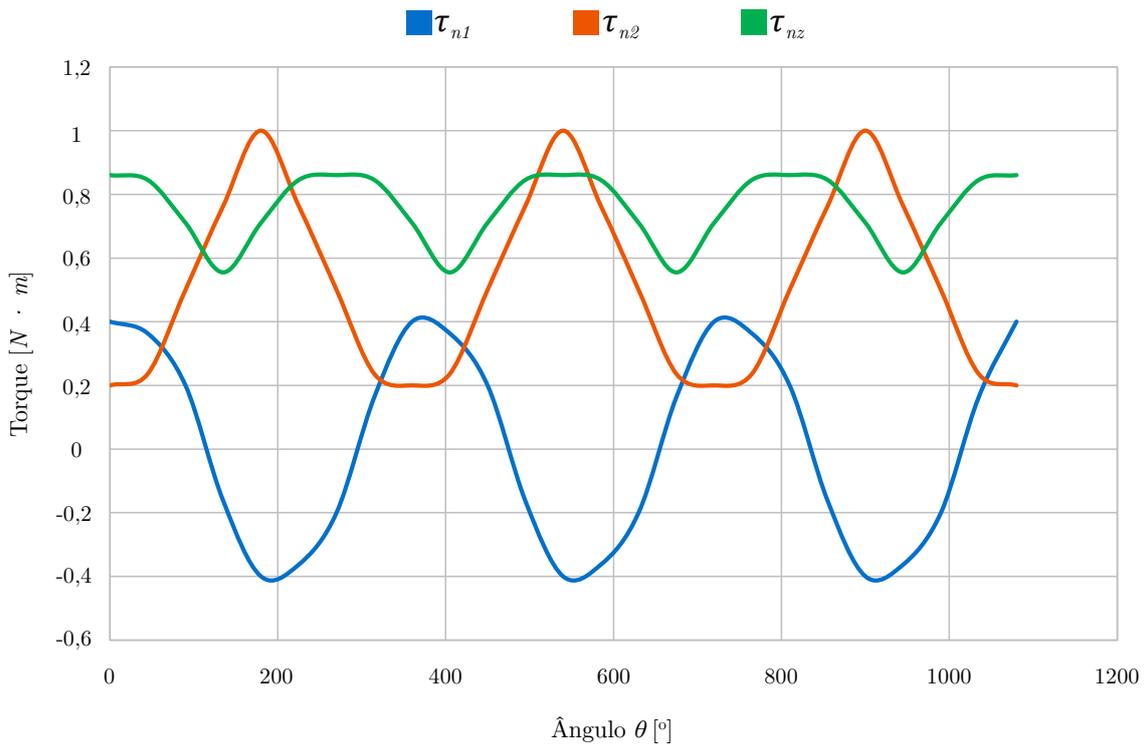


Figura 4.20 - Curvas de torque das juntas do robô.

4.2 Implementação eletroeletrônica

A implementação eletroeletrônica do robô SCARA envolve a seleção meticulosa de componentes e sistemas, incluindo atuadores, sensores e módulos de controle, para assegurar a funcionalidade e eficiência do dispositivo. Este processo incorpora a integração de elementos eletroeletrônicos com a estrutura mecânica do robô, garantindo a precisão dos movimentos e a confiabilidade das operações. A configuração destes componentes, alinhada aos objetivos operacionais do SCARA, é o aspecto significativo para o desempenho do sistema, ressaltando a importância da abordagem sistemática na implementação eletroeletrônica.

4.2.1 Seleção de componentes

A seleção de componentes para o robô SCARA é a etapa que envolve a avaliação e escolha de atuadores, sensores e outros elementos eletrônicos, baseando-se em critérios técnicos específicos como desempenho, compatibilidade e confiabilidade. Este processo é fundamental para garantir que o robô atenda aos requisitos operacionais desejados, mantendo o equilíbrio entre custo e eficiência. A abordagem meticulosa

na seleção de componentes contribui diretamente para a otimização da funcionalidade do robô e sua capacidade de executar tarefas com precisão. A seleção dos componentes para o desenvolvimento da implementação eletroeletrônica é dividida em duas etapas: i) seleção de atuadores e sensores e ii) seleção de fonte e módulos de processamento.

4.2.1.1 Seleção de atuadores e sensores

No âmbito dos atuadores e sensores, os motores de passo são amplamente utilizados em aplicações de robótica devido à sua capacidade de movimentação precisa controlada por pulsos elétricos, sem a necessidade de sistema de realimentação externo. O modelo NEMA 17 é reconhecido por seu tamanho compacto e por fornecer o equilíbrio ideal entre torque e velocidade (HUGHES; DRURY, 2013). A escolha deste modelo de motor é fundamentada por três razões: i) precisão de movimento, ii) torque adequado e iii) compatibilidade elétrica.

O motor NEMA 17 permite controle preciso do ângulo de rotação, essencial para a manipulação e posicionamento exatos no robô SCARA. Além disso, a partir dos torques exigidos em cada junta do robô, conforme a Tabela 4.9, o torque de $4,2\text{kgf}\cdot\text{cm}$ foi selecionado, este motor oferece a força necessária e suficiente para mover os elos e juntas do robô sem sobrecarregar o sistema. Por fim, a corrente de operação de $1,7\text{A}$ é compatível com a maioria dos *drivers* de motor de passo, facilitando a integração com outros componentes do sistema.

Segundo Usategui e León (1986), o servomotor MG996R é o atuador eletromecânico que permite o controle preciso de ângulo, velocidade e posição. Este modelo é amplamente reconhecido por sua construção robusta e alto torque, tornando-o a escolha ideal para aplicações que demandam movimentações confiáveis e precisas. Este motor servirá como o atuador do efetuador final, permitindo o movimento completo da garra. Esta escolha é pautada em três motivos principais: i) torque elevado, ii) confiabilidade e iii) facilidade de integração.

O MG996R oferece torque significativamente alto, cerca de $11\text{kgf}\cdot\text{cm}$, que é essencial para manipular cargas ou executar tarefas que requerem força considerável, sem comprometer a precisão do movimento. Além disso, este servomotor é conhecido por sua construção sólida e confiabilidade operacional. A carcaça metálica e as engrenagens internas reforçadas proporcionam a longevidade superior, assegurando que o robô mantenha sua funcionalidade ao longo de extensos períodos de uso (BISHOP, 2002). Por fim, o MG996R é compatível com ampla gama de controladores como

o Arduino UNO, por exemplo. Sua popularidade e a disponibilidade de bibliotecas e exemplos de código facilitam a integração e a programação no sistema do robô. A Figura 4.21, ilustra os dois tipos de motores selecionados como atuadores para o robô SCARA.

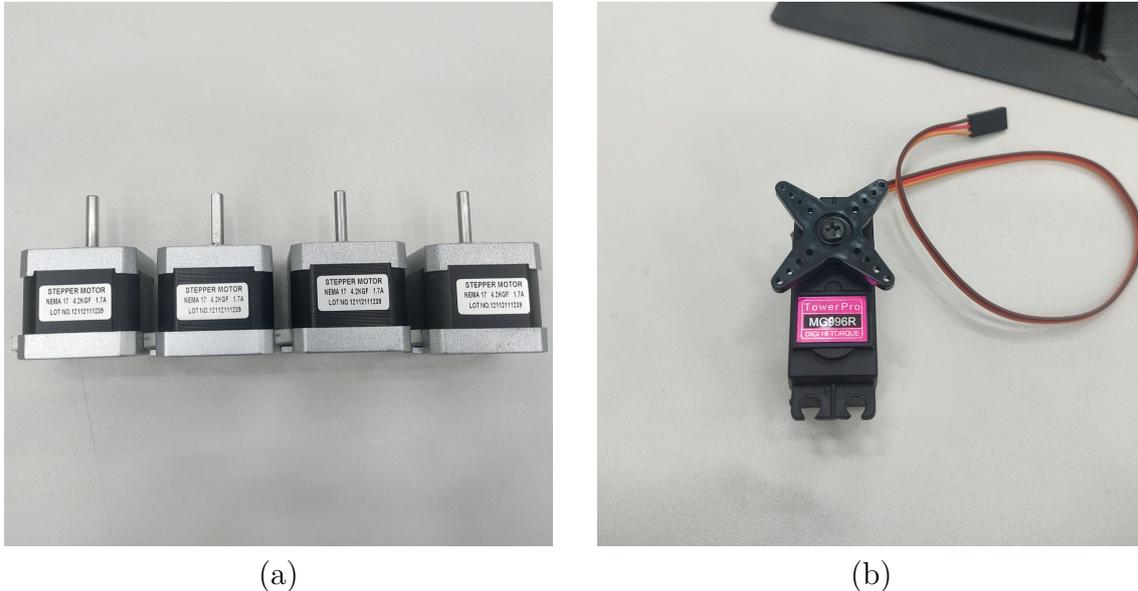


Figura 4.21 - Motores utilizados: (a) motores de passo NEMA 17 e (b) servomotor MG996R.

O A4988 é o *driver* de motor de passo que possui o conversor de corrente digital integrado, capaz de operar motores de passo bipolares com até 2A por bobina. Conforme Perim e Nascimento (2017), este dispositivo inclui o regulador de corrente fixo *off-time*⁴ que opera em modo *slow* ou *mixed decay*. O A4988 é projetado para operar motores de passo em modos de passo inteiro, meio passo, quarto de passo, oitavo de passo e décimo sexto de passo, fornecendo a variação de resolução de passo sem a necessidade de trocar o motor. A escolha do A4988 é fundamentada por três motivos principais: i) controle de movimento, ii) proteções integradas e iii) facilidade de integração.

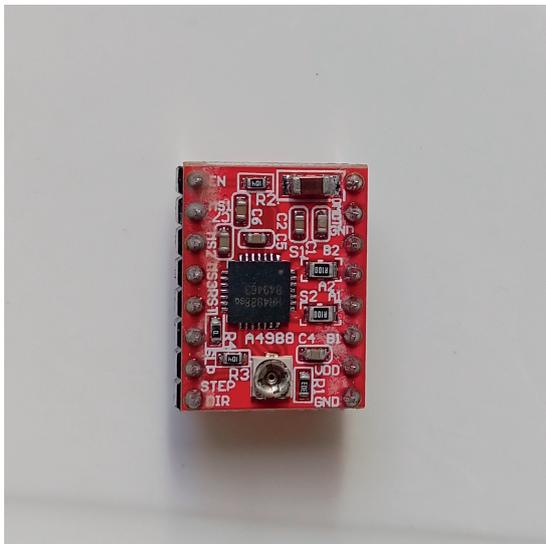
O A4988 permite o ajuste fino da corrente e da resolução de passos⁵ do motor

⁴*Off-time* refere-se ao período em que a corrente não é aplicada às bobinas do motor no *driver* de motor de passo. O modo *slow decay* mantém a corrente mais tempo, enquanto o *mixed decay* utiliza a combinação de *slow* e *fast decay* para melhorar a precisão do controle de corrente.

⁵Passos do motor referem-se aos incrementos discretos de movimento realizados pelo rotor do motor de passo, cada passo corresponde ao ângulo específico de rotação, permitindo movimentos precisos e controlados.

de passo, o que é imperativo para a precisão dos movimentos do robô SCARA. A capacidade de *microstepping* do *driver* assegura movimentos suaves e minimiza as vibrações. Este *driver* inclui várias funcionalidades de proteção, como proteção contra sobrecorrente e sobreaquecimento, assegurando a operação segura do sistema. Estas proteções são fundamentais para prevenir danos aos motores e ao próprio *driver*, garantindo a longevidade e a confiabilidade do robô SCARA. Além disso, o A4988 é compatível com diversos microcontroladores (KENJŌ; SUGAWARA, 1994).

No projeto do robô SCARA, os sensores fim de curso são utilizados para sinalizar os limites máximos de movimentação dos elos e do efetuator final, prevenindo a movimentação além dos pontos estabelecidos. Para estes sensores foi selecionada a especificação de 3A e 250V, isto indica que estes sensores são capazes de lidar com altas cargas, tornando-os adequados para a interrupção ou controle de circuitos que operam dentro destes limites (BUTTERFIELD; SZYMANSKI, 2018). Esta capacidade assegura a confiabilidade do sensor em diversas condições de operação, sem risco de falha devido a sobrecarga. Segundo Bolton (2021), estes sensores são facilmente integráveis ao sistema de controle do robô SCARA, podendo ser conectados diretamente ao microcontrolador ou através de interfaces como o CNC *Shield*. A Figura 4.22, ilustra o *driver* A4988 e o sensor fim de curso selecionados para o âmbito de atuadores e sensores, respectivamente.



(a)



(b)

Figura 4.22 - *Driver* e fim de curso utilizados: (a) vista frontal do *driver* A4988 e (b) sensor fim de curso.

4.2.1.2 Seleção de fonte e módulos de processamento

Segundo Rashid (2017), as fontes de tensão chaveadas diferem das fontes lineares tradicionais em vários aspectos, principalmente na forma como convertem a tensão da rede elétrica em tensões mais baixas e utilizável pelos dispositivos eletrônicos. Utilizando a técnica de comutação de alta frequência, elas transferem energia da entrada para a saída por meio do ciclo de ligar e desligar rapidamente, controlado pelo regulador. Para esta aplicação foi selecionada a fonte de tensão chaveada que opera a 12V e 20A. A escolha desta fonte de tensão é embasada em três razões: i) adequação de tensão e corrente, ii) eficiência energética e iii) compactação.

O robô foi projetado para operar tanto em regimes contínuos quanto intermitentes. Durante a operação, o usuário ativa os motores, articula os elos e aciona o efetuator final. Dependendo do modo selecionado, o manipulador pode manter-se em estado de repouso ou continuar em funcionamento, visto que o robô é equipado com o modo de operação que permite a ativação contínua dos componentes sem interrupções. Para determinar a especificação adequada da fonte de tensão chaveada, adotou-se o critério de operação ininterrupta do manipulador, visando calcular o consumo energético total do sistema. A análise do consumo de corrente de cada componente foi realizada com base nos *datasheets*⁶ e informações técnicas disponíveis, cujos resultados estão compilados na Tabela 4.10. Nesta tabela, COMP é a abreviatura de componente.

Tabela 4.10 - Gasto de corrente em cada componente.

COMP	GASTO POR COMP	QTD	GASTO TOTAL
NEMA 17 (2 fases)	1,7A (por fase)	4	13,6A
Servomotor MG996R	0,9A	1	0,9A
<i>Driver</i> A4988	0,004A	4	0,016A
Arduino UNO	1,11A	1	1,11A
Sensor fim de curso	0,31A	4	1,24A
Total	4,024A	12	16.866A

Com base nestes dados da Tabela 4.10, o consumo total estimado do sistema é de aproximadamente 16.866A. Levando em conta a operação ininterrupta do sistema, estima-se que o manipulador possa funcionar por períodos de até 2 horas contínuas,

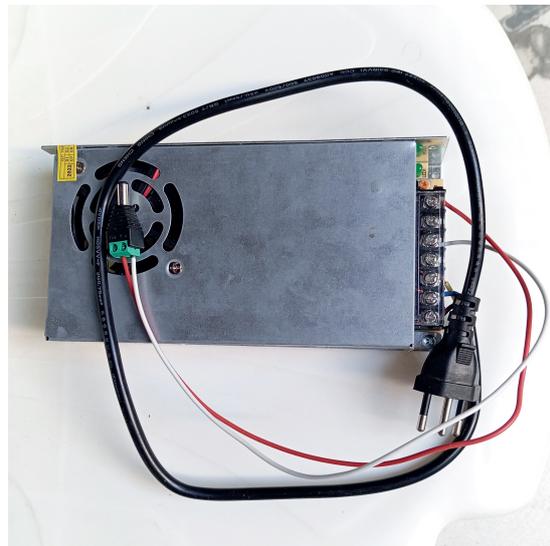
⁶O *datasheet* é o documento técnico que fornece especificações detalhadas e informações técnicas sobre produtos, componentes ou materiais, incluindo características, funcionalidades, padrões de desempenho e diretrizes de aplicação.

mitigando o risco de sobrecarga e desgaste mecânico dos componentes. Para suprir esta demanda, optou-se pela fonte de tensão de 20A, proporcionando a margem de segurança de 3.134A. Esta margem não só previne o aquecimento excessivo dos condutores e a sobrecarga do sistema, como também permite a incorporação de componentes adicionais ou atualizações futuras, incluindo sistemas de resfriamento, como *coolers*, para otimizar a performance do robô SCARA. A tensão de 12V é ideal para a maioria dos componentes eletrônicos e motores utilizados no robô SCARA, enquanto a capacidade de 20A garante a oferta abundante de corrente para suportar todos os componentes simultaneamente, inclusive em situações de carga máxima.

Além disso, as fontes chaveadas são notavelmente mais eficientes do que as alternativas lineares, convertendo a maior porção da energia de entrada em energia útil, ao invés de dissipá-la como calor (SEDRA; SMITH, 2014). Esta eficiência não apenas reduz o consumo de energia mas também diminui a necessidade de dissipação de calor, contribuindo para o sistema mais compacto e sustentável. Por fim, devido à sua alta eficiência e ao método de comutação de alta frequência, as fontes chaveadas são menores e mais leves que as fontes lineares equivalentes. Isto é benéfico em aplicações robóticas em que o espaço e o peso são considerações críticas. A Figura 4.23, ilustra a fonte de tensão selecionada para o robô SCARA, ressaltando que sua operação é referente a 12V de tensão e 20A de corrente.



(a)



(b)

Figura 4.23 - Fonte de alimentação utilizada: (a) vista frontal da fonte chaveada e (b) fonte com tomada.

O Arduino UNO é a placa de microcontrolador baseada no ATmega328P, amplamente reconhecida na comunidade de desenvolvedores, educadores e entusiastas de eletrônica por sua facilidade de uso e flexibilidade (PERIM; NASCIMENTO, 2017). Conforme estabelecido por Lima e Villaça (2012), o Arduino UNO serve como a plataforma introdutória ideal para a eletrônica e programação, devido ao seu *design* intuitivo e ambiente de desenvolvimento integrado que simplifica a escrita de códigos e a transferência para a placa. Ele possui 14 pinos de entrada e saída digital, 6 entradas analógicas, oscilador de cristal de 16MHz, conexão USB, conector de alimentação, conexões ICSP e botão de *reset*. A escolha do Arduino UNO é fundamentada por três motivos principais: i) acessibilidade, ii) compatibilidade e iii) facilidade de programação.

A escolha do Arduino UNO é pautada por sua popularidade e a vasta comunidade de usuários que contribuem com a rica biblioteca de tutoriais, códigos de exemplo e fóruns de suporte. Estes recursos são inestimáveis para resolver problemas, aprender novas técnicas e compartilhar conhecimentos. O Arduino UNO é compatível com ampla gama de *shields* e módulos, permitindo a expansão quase ilimitada de suas capacidades. Isto o torna escolha versátil para muitos projetos, permitindo a adição de funcionalidades como conectividade *Wi-Fi*, controle de motores, sensores adicionais e muito mais, sem a necessidade do redesenho completo do sistema. Por fim, o ambiente de desenvolvimento do Arduino é projetado para ser simples para iniciantes, ao mesmo tempo que oferece profundidade suficiente para usuários avançados. A linguagem de programação baseada em C/C++ é acessível e bem documentada, o que facilita o desenvolvimento rápido de projetos.

O CNC *Shield* é a placa de expansão projetada especificamente para trabalhar com o Arduino UNO, facilitando o controle de máquinas CNC e robôs (HUGHES, 2016). Esta placa é amplamente utilizada em projetos de automação e robótica devido à sua capacidade de controlar até quatro motores de passo simultaneamente, tornando-a escolha ideal para aplicações que requerem movimentos complexos e precisos. Segundo Monk (2016), o CNC *Shield* se encaixa diretamente sobre o Arduino UNO, estendendo suas capacidades de E/S digital⁷ para incluir a funcionalidade específica de controle de motores de passo. Ele é projetado para aceitar *drivers* de motor de passo, como o A4988 ou o DRV8825, que se conectam aos pinos correspondentes na placa do *Shield*. Além disso, ele apresenta *slots* para a conexão de sensores fim de

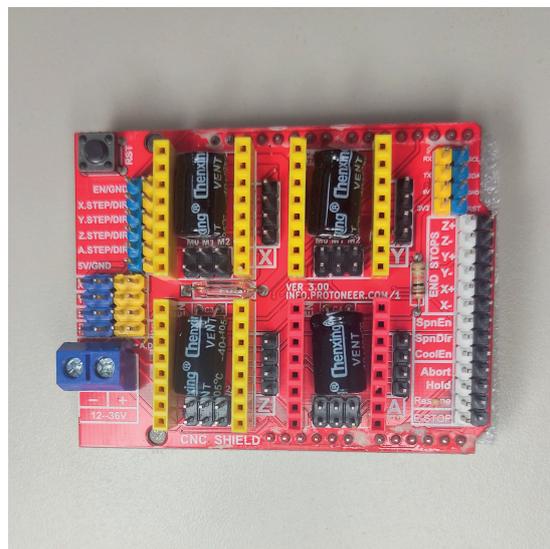
⁷E/S digital refere-se a entrada/saída digital, o mecanismo fundamental em microcontroladores e circuitos eletrônicos para receber e enviar sinais digitais, representados por dois estados: alto ou baixo, correspondendo a níveis de tensão específicos.

curso e outros dispositivos periféricos. A escolha da CNC *Shield* também é formulada através de três razões majoritárias: i) capacidade de controle, ii) facilidade de integração e iii) expansibilidade.

A grande vantagem do CNC *Shield* é sua habilidade de gerenciar até quatro motores de passo independentemente, o que é imperativo para o controle dos múltiplos eixos do robô SCARA. Além disso, a compatibilidade direta do CNC *Shield* com o Arduino UNO simplifica significativamente o processo de montagem e configuração do sistema de controle do robô. A facilidade de instalação dos *drivers* de motor de passo e a disponibilidade de numerosos pinos para conexão dos sensores fim de curso tornam o CNC *Shield* a opção conveniente e eficiente para desenvolvedores. Por fim, o seu *design* modular permite a customização extensa e a adição de funcionalidades conforme necessário. Por exemplo, a capacidade de ajustar a corrente para os motores de passo diretamente na placa do *shield* oferece controle preciso sobre o desempenho dos motores, enquanto os *slots* adicionais para periféricos permitem a expansão das capacidades do robô SCARA sem a necessidade de *hardware* adicional. A Figura 4.24, ilustra os módulos de processamento de sinais utilizados.



(a)



(b)

Figura 4.24 - Módulos de processamento e controle utilizados: (a) Arduino UNO e (b) CNC *Shield*.

Para sistematizar a organização e o registro dos componentes eletroeletrônicos empregados, desenvolveu-se o inventário detalhado que inclui cada item selecionado

juntamente com suas respectivas quantidades. Assim, a lista dos componentes escolhidos encontra-se apresentada na Tabela 4.11, em que QTD representa a abreviação de quantidade e UN denota a de unidade.

Tabela 4.11 - Lista de componentes eletroeletrônicos.

ITEM	DESCRIÇÃO	DADOS	QTD	UN
1	Motor de passo NEMA 17	$4,2\text{kgf} \cdot \text{cm}$ e $1,7\text{A}$	4	PÇ
2	Servomotor MG996R	$11\text{kgf} \cdot \text{cm}$ e 5V	1	PÇ
3	<i>Driver</i> A4988	$8 - 35\text{V}$ e 2A	4	PÇ
4	Arduino UNO	ATmega328P e 5V	1	PÇ
5	<i>CNC Shield</i>	$68 \times 54 \times 12\text{mm}$	1	PÇ
6	Fonte de tensão chaveada	12V e 20A	1	PÇ
7	Sensor fim de curso	250V e 3A	4	PÇ
8	<i>Cabo jumper</i>	$2,54 \times 20\text{mm}$	25	PÇ
9	Conector mini <i>jumper</i>	$2,54 \times 5 \times 6\text{mm}$	7	PÇ
10	Conector P4 macho	36V e 4A	1	PÇ
11	Conector P4 fêmea	36V e 4A	1	PÇ
12	Cabo USB 2.0	500mm	1	PÇ

A Figura 4.25, ilustra a montagem parcial dos componentes eletroeletrônicos e mecânicos do robô SCARA.

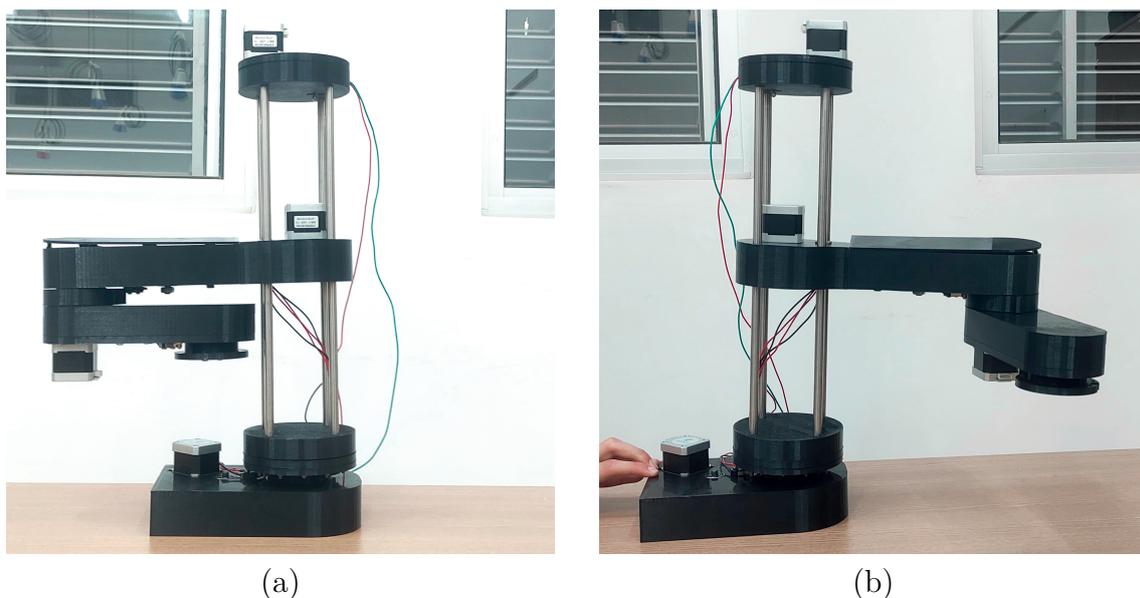


Figura 4.25 - Montagem parcial do robô SCARA: (a) vista com o segundo elo fechado e (b) vista com o segundo elo semiaberto.

4.2.2 Configuração dos *drivers* de acionamento

Segundo Niku (2010), o motor de passo NEMA 17 apresenta a configuração padrão de 200 passos, equivalente a resolução de $1,8^\circ$ por passo. Vale destacar que a designação NEMA 17 refere-se unicamente às dimensões do painel frontal do motor, sendo o 17 a indicação que, ao ser dividida por 10, resulta no painel frontal de 1,7 polegadas. Embora as dimensões do painel frontal sejam padronizadas, o comprimento dos motores NEMA 17 pode variar entre 20mm e 60mm , influenciando diretamente na demanda energética do motor, que varia de $0,3\text{A}$ a $2,5\text{A}$. Para este projeto, selecionou-se o motor que opera com a corrente de $1,7\text{A}$.

O *driver* A4988, capaz de suportar até 2A por bobina, é fundamental para o controle dos motores de passo bipolares. O uso de *Field-Effect Transistors* (FETs) nas pontes H internas do *chip* A4988 reduz a resistência interna, diminuindo a dissipação de potência e, conseqüentemente, o calor gerado (KENJŌ; SUGAWARA, 1994). Os pinos do módulo A4988 facilitam a programação e a customização do controle do motor, incluindo o pino STEP para avançar passos, o pino DIR para alterar a direção de rotação, e o pino ENABLE para ativar os *drivers* do motor. Além disso, pinos como RESET, SLEEP, MS1, MS2 e MS3 permitem o ajuste fino do funcionamento do motor, desde o *reset* do *chip* até a seleção do modo de *microstepping*, aumentando a flexibilidade e eficiência do controle do motor de passo no contexto do robô SCARA. A Tabela 4.12, demonstra a resolução dos modos de passo do *driver* A4988. Estes modos são selecionados através dos pinos MS1, MS2 e MS3.

Tabela 4.12 - Modos de passo do *driver* A4988.

MS1	MS2	MS3	RESOLUÇÃO
<i>Low</i>	<i>Low</i>	<i>Low</i>	Passo completo
<i>High</i>	<i>Low</i>	<i>Low</i>	Meio passo
<i>Low</i>	<i>High</i>	<i>Low</i>	Quarto de passo
<i>Low</i>	<i>High</i>	<i>High</i>	Oitavo de passo
<i>High</i>	<i>High</i>	<i>High</i>	Dezesseis avos de passo

Motores de passo operam de maneira que cada avanço corresponde a $1,8^\circ$. Dada a natureza circular de 360° da rotação completa, estes motores requerem 200 passos para completar seu ciclo. Deste modo, o modo de operação de passo completo é definido por estes 200 passos. A precisão no controle dos motores aumenta à medida que se reduz o tamanho do passo através do *driver* A4988, incrementando o número

total de passos necessários para a rotação completa. Na configuração dos motores de passo do robô SCARA, optou-se pelo modo de quarto de passo. Optar pela configuração de quarto de passo visa aprimorar a precisão e suavidade dos movimentos. Esta escolha eleva a resolução do posicionamento, permitindo o controle mais detalhado, e contribui para transições mais fluidas entre os passos, minimizando vibrações e ruído.

Para esta configuração, foi necessário posicionar conectores mini *jumper* nos pinos correspondentes da CNC *Shield*. Conforme especificado na Tabela 4.12, para esta configuração, os pinos MS1 e MS3 foram configurados em *low*, sem o *jumper*, enquanto o pino MS2 foi ajustado em *high*, com *jumper*. Este procedimento foi aplicado na CNC *Shield* para todos os quatro *drivers* associados aos motores de passo do sistema.

Além disso, é necessária a inserção do conector mini *jumper* entre os pinos *enable* (EN) e *ground* (GND) na CNC *Shield*. O pino EN é responsável por habilitar ou desabilitar os *drivers* dos motores de passo. Na ausência da conexão *jumper* entre EN e GND, os *drivers* permanecem no estado desabilitado, impedindo os motores de passo de receberem sinais de comando e, conseqüentemente, de se moverem. Ao estabelecer esta conexão, os *drivers* são ativados, permitindo que os motores respondam aos sinais de controle enviados pelo microcontrolador.

Por fim, foi necessário a inserção de dois conectores mini *jumper* nos pinos D12 e D13, próximos à conexão de alimentação na CNC *Shield*, está diretamente relacionada à ativação do eixo *z*. Estes pinos são frequentemente utilizados para funções especiais ou para expandir a capacidade de controle do *shield*. No contexto do robô SCARA, os *jumper* em D12 e D13 asseguram que os sinais de controle específicos para o eixo *z* sejam corretamente interpretados e executados pelo sistema. Sem estes *jumper*, o eixo *z* pode não receber os sinais apropriados para a sua ativação, resultando em sua inoperância. A Figura 4.26, ilustra o diagrama elétrico dos elementos utilizados na construção do robô SCARA, exibindo, inclusive, as configurações de *jumper* realizados na placa CNC *Shield*.

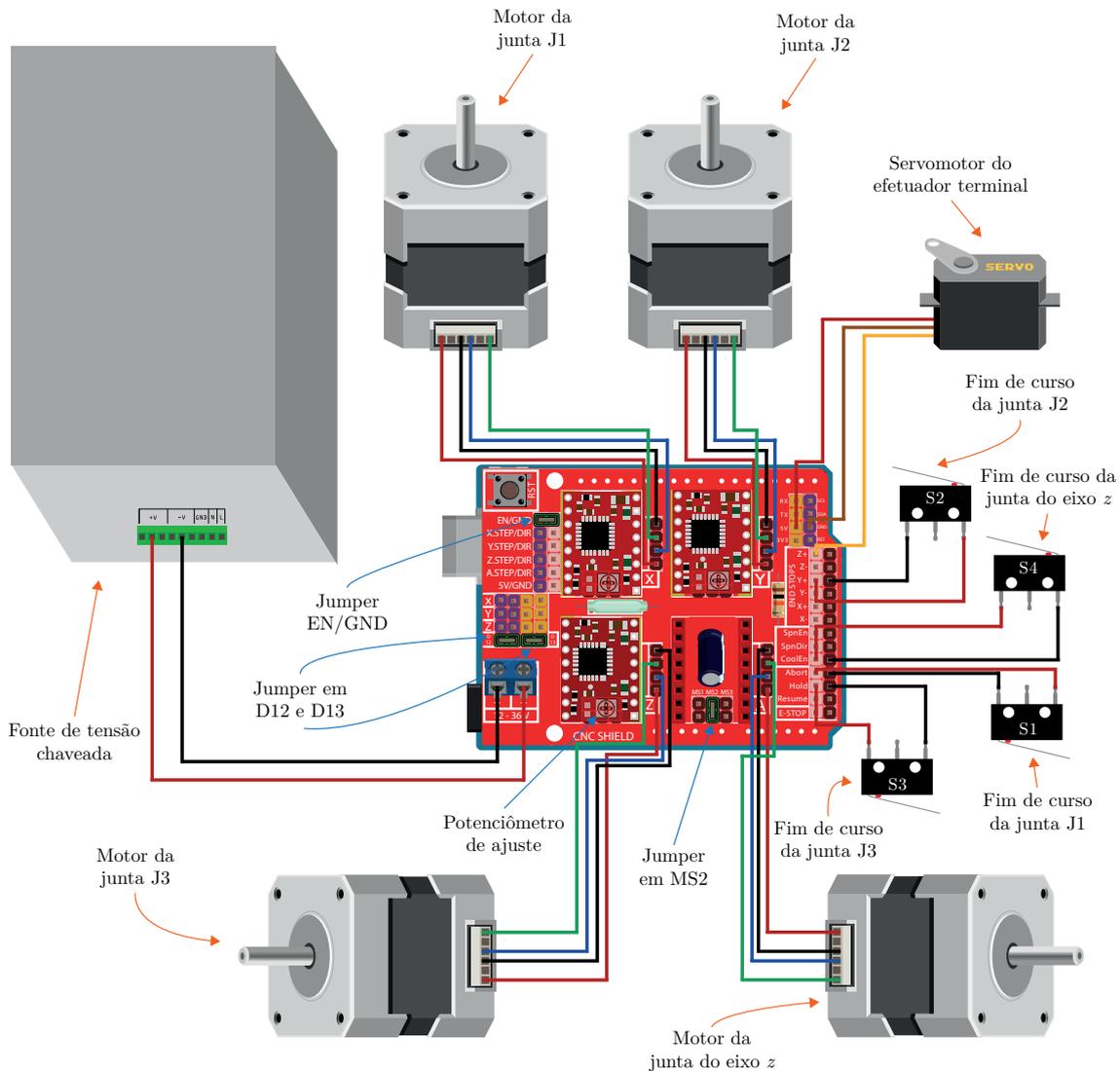


Figura 4.26 - Diagrama elétrico de montagem do robô SCARA.

4.2.2.1 Limitação de corrente do *driver* A4988

A informação relativa à corrente requerida para alimentar os motores de passo pode ser obtida consultando o *datasheet* fornecido pelo fabricante. Cada motor de passo do tipo NEMA 17, empregado no sistema em questão, demanda a corrente de 1,7A por fase. Isto assegura que o *driver* A4988 selecionado atende às exigências dos motores em uso (NEOYAMA, 2023).

O principal atributo do *driver* de passo A4988 é sua capacidade de limitação de corrente. Isto possibilita a determinação precisa da corrente consumida pelo motor,

independentemente de sua classificação nominal. Por exemplo, é viável conectar o motor de passo com a classificação de corrente nominal de $2A$, porém, o *driver* limitará a corrente a $1,5A$. Ainda que o motor não esteja operando em sua capacidade máxima, ele pode ser utilizado com sucesso. Em contrapartida, caso a classificação do motor seja inferior ao limite de corrente estabelecido pelo *driver*, o motor estará sujeito ao superaquecimento. Conforme mencionado por Barrientos et al. (2007), é imprescindível ajustar o limite de corrente do *driver* de forma a ser inferior à corrente nominal do motor, a fim de evitar o superaquecimento.

A Figura 4.26, ilustra a disposição de pequenos potenciômetros *trimmer*⁸ em cada *driver* A4988 incorporado à CNC *Shield*, por meio dos quais é possível ajustar o limite de corrente. Girar o potenciômetro no sentido horário aumenta o limite de corrente, e vice-versa. De acordo com as informações técnicas fornecidas pelo fabricante Allegro (2024), a determinação do valor real do limite de corrente envolve a medição da tensão de referência no próprio potenciômetro e no terminal GND do *driver*. A medição da tensão de referência é realizada utilizando o multímetro, e o valor obtido é posteriormente aplicado à equação (4.5):

$$I_{max} = \frac{V_{ref}}{8 \cdot R_{cs}} \quad (4.5)$$

no qual I_{max} é a corrente de operação utilizada pelo motor de passo, medida em amperes $[A]$, V_{ref} é a tensão de referência para o ajuste da corrente no *driver*, medida em volts $[V]$ e R_{cs} é a resistência de detecção de corrente ou os valores dos resistores de detecção de corrente localizados próximos ao *chip*, esta resistência é medida em ohms $[\Omega]$, o valor de oito localizado no denominador corresponde a utilização máxima de 80% do desempenho máximo de corrente e tensão suportados no motor (ALLEGRO, 2024).

Conforme as especificações do fabricante, os valores das resistências de detecção geralmente se encontram nas faixas de $0,05$, $0,1$ e $0,2\Omega$. Para os *drivers* empregados no sistema do robô SCARA, os resistores incorporados possuem a resistência de $0,1\Omega$. Na aplicação em questão, a corrente máxima de operação para cada motor de passo é estipulada em $1,7A$, no entanto, seguindo a recomendação de NEOYAMA (2023) para aplicar a margem de segurança à corrente do motor de passo, adotou-se o valor de $1,5A$. Através da aplicação da equação (4.5), obteve-se a tensão de

⁸Componente eletrônico ajustável, geralmente o potenciômetro, usado para fazer ajustes precisos em circuitos eletrônicos, como limitar correntes, ajustar tensões ou calibrar dispositivos, girando seu eixo para alcançar os valores desejados.

referência V_{ref} de $1,2V$. Esta tensão foi então ajustada nos quatro *drivers*, a fim de estabelecer a corrente máxima de operação I_{max} em $1,5A$.

Após concluir a montagem de todos os componentes eletroeletrônicos, estes foram acomodados dentro da caixa disponível para o Arduino, e os cabos *jumpers* foram cuidadosamente dispostos internamente na mesma. Adicionalmente, os dissipadores de calor, situados nos *drivers*, foram posicionados de forma a ficarem semi expostos, permitindo a entrada de ar para a refrigeração adequada dos mencionados dispositivos. As representações visuais das caixas utilizadas para o alojamento do Arduino podem ser consultadas nas pranchetas A.1 e A.2, disponibilizadas no Apêndice A.1. Por fim, a base do robô SCARA foi devidamente fixada e aparafusada à chapa de madeira, cujas dimensões são de $400 \times 500mm$. Esta base foi projetada com o propósito de proteger os elementos presentes na parte inferior do robô SCARA, incluindo polias, correias, parafusos e rolamentos. A Figura 4.27, ilustra o robô SCARA em sua totalidade, evidenciando a conclusão da montagem tanto de sua parte mecânica quanto eletroeletrônica.

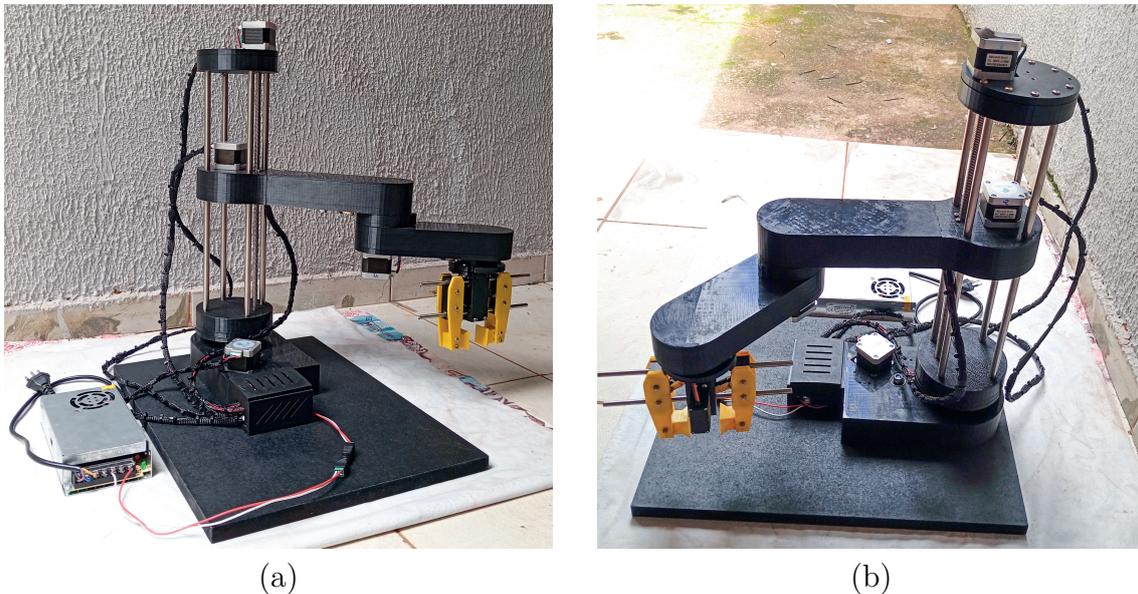


Figura 4.27 - Montagem final do robô SCARA: (a) vista lateral e (b) vista isométrica.

4.3 Simulação do robô SCARA

A simulação do robô SCARA é fundamental para o desenvolvimento e teste de sistemas robóticos, proporcionando a plataforma segura e eficiente para a análise de

algoritmos de controle, desempenho operacional e interação com o ambiente. Esta abordagem permite a identificação e correção de problemas potenciais no ambiente virtual, reduzindo riscos e custos associados a experimentos com *hardware* real, além de acelerar o processo de *design* e otimização do robô SCARA.

4.3.1 Seleção do *software* de simulação

Para a realização da simulação do robô SCARA, optou-se pelo uso do *software* CoppeliaSim em conjunto com o Jupyter Notebook como ambiente de programação. A programação no Jupyter Notebook é efetuada utilizando a linguagem Python, que permite a comunicação eficaz com o CoppeliaSim para a execução da simulação.

O CoppeliaSim, também conhecido como *Virtual Robot Experimentation Platform* (V-REP), é o ambiente de simulação robótica, que oferece a plataforma versátil para o teste e desenvolvimento de algoritmos robóticos em ambiente virtual tridimensional. Com suporte para ampla gama de sensores e atuadores, o CoppeliaSim é projetado para facilitar a simulação de sistemas robóticos complexos, permitindo a análise detalhada de seus comportamentos em diversas condições (SICILIANO et al., 2009). Os usuários podem programar comportamentos complexos, utilizando linguagens como Lua, Python e C++. A simulação física inclui interações realistas entre objetos, como colisões e dinâmica de corpos rígidos. A escolha deste *software* de simulação é fundamentada por três razões principais: i) interatividade, ii) compatibilidade e iii) ferramentas de análise.

O CoppeliaSim oferece o ambiente altamente interativo e visualmente realista, essencial para a simulação precisa de robôs como o SCARA. Isto permite a análise detalhada do comportamento do robô e de sua interação com o ambiente, facilitando a identificação e a solução de potenciais problemas antes da implementação física. Sua ampla compatibilidade com diferentes linguagens de programação e sistemas robóticos torna o CoppeliaSim a escolha flexível, adaptando-se facilmente a diversos projetos e necessidades de pesquisa e desenvolvimento. Por fim, o *software* vem equipado com diversas ferramentas analíticas integradas, que permitem a avaliação de parâmetros críticos durante a simulação, como forças, torques e trajetórias.

Segundo Nkomo e Collier (2012), o Jupyter Notebook é a aplicação de código aberto⁹ que permite a criação e compartilhamento de documentos que contêm código vivo,

⁹Refere-se a *softwares* cujo código-fonte é disponibilizado publicamente, permitindo que qualquer pessoa visualize, modifique e distribua-os conforme necessário, sob a licença que garante estas liberdades.

equações, visualizações e texto narrativo. O Jupyter Notebook é a ferramenta utilizada para prototipagem rápida e iteração de ideias complexas através de código, principalmente em Python. A escolha do Jupyter Notebook para integração com o CoppeliaSim é embasada em três motivos: i) facilidade de uso, ii) integração e iii) documentação.

A interface intuitiva do Jupyter Notebook e a facilidade de uso do Python como linguagem de programação tornam o desenvolvimento e a depuração de algoritmos mais acessíveis, especialmente para aqueles com experiência limitada em programação. Além disso, a capacidade de integração do Jupyter Notebook com o CoppeliaSim, através de bibliotecas específicas em Python, permite a comunicação fluida entre o código e a simulação, facilitando a implementação de algoritmos complexos e o controle preciso dos elementos simulados. O formato de documento do Jupyter Notebook, que suporta a inclusão de código, texto explicativo e visualizações, favorece a documentação detalhada do projeto e o compartilhamento eficiente de informações com outros membros da equipe ou da comunidade científica, promovendo a colaboração efetiva e a disseminação do conhecimento.

4.3.2 Funcionamento da simulação

A Figura 4.28, ilustra a área de trabalho do CoppeliaSim. Este que é o ambiente tridimensional destinado à execução de comandos de cinemática para controlar o modelo do robô SCARA.

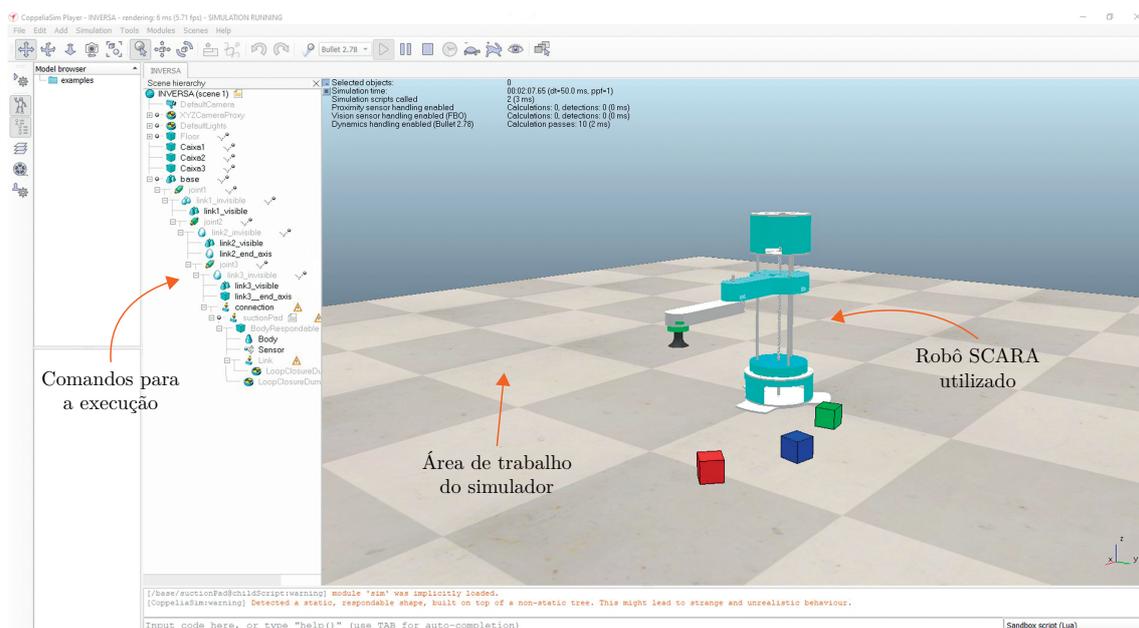


Figura 4.28 - Área de trabalho da simulação do CoppeliaSim.

Para iniciar a simulação, é necessário utilizar a *Application Programming Interface* (API) denominada *ZmqRemoteApi*. Esta API possibilita a comunicação entre o código executado no Jupyter Notebook e o ambiente de simulação CoppeliaSim. Para garantir o funcionamento correto do CoppeliaSim, é imprescindível que algumas bibliotecas sejam previamente instaladas no sistema operacional. Entre estas bibliotecas, destacam-se o *zmq*, utilizado para comunicação *web* entre os *softwares*, o *cbor*, que representa objetos binários de forma concisa, *sympy*, a ferramenta de álgebra computacional, *time*, responsável pela manipulação de tempo, e outras bibliotecas relacionadas. Com as bibliotecas instaladas pode-se executar o Jupyter Notebook. O Código 4.1, desenvolvido em Python, aborda a comunicação entre o Jupyter e o CoppeliaSim, iniciando o processo de simulação.

```

1 #Inicio do codigo:
2 #Deve-se importar algumas bibliotecas, e estabelecer a comunicacao
  entre o Jupyter Notebook e o CoppeliaSim:
3 import zmqRemoteApi #Biblioteca utilizada para estabelecer a
  comunicacao remota com o simulador;
4 import time #Biblioteca utilizada para estabelecer e acessar o
  tempo e a data ao programa (neste programa foi utilizada para
  implementar o 'delay');
5 client = zmqRemoteApi.RemoteAPIClient() #Variavel que recebe a
  comunicacao entre os 'softwares';

```

```

6 sim = client.getObject('sim') #Confirmando a comunicacao entre os
  'softwares';

```

Código 4.1 - *Script* para realizar a comunicação entre o CoppeliaSim e o Jupyter.

Após a etapa de comunicação, foi desenvolvida a cinemática direta, o componente essencial para garantir o desempenho eficaz do sistema, validar o *design* e integrar adequadamente o robô. Isto proporcionou a abordagem prática e eficiente no desenvolvimento e validação do sistema robótico. O modelo 3D do robô SCARA utilizado é nativo do *software* CoppeliaSim. Além disso, foram criadas duas simulações, abordando tanto a cinemática direta quanto a cinemática inversa. O Código 4.2, desenvolvido em Python, foi elaborado para simular o robô SCARA utilizando a cinemática direta.

```

1 #Estabelece a posicao de destino do robo conforme o objeto criado
  (caixas):
2 print('Calculando a Cinematica Direta...')
3 Caixa1 = sim.getObject('/Caixa1')
4 caixa1_pos = sim.getObjectPosition(Caixa1, sim.handle_world)
5 D = MatrixFromPose(caixa1_pos[0], caixa1_pos[1], caixa1_pos[2], 0,
  0, 0)
6 D
7 try:
8     q = sympy.nsolve((T-D), (q1, q2, q3), (1, 1, 2), prec=6)
9 except:
10    print('Nao pode ser executado') #Caso ocorra-se erro ao
  programa, exibe a mensagem que nao e possivel executa-lo;
11    q = [0 , 0.2 , 0]
12 print(q)

```

Código 4.2 - *Script* para a simulação do robô utilizando a cinemática direta.

A segunda fase da simulação envolveu a criação do *script* em Python, conforme demonstrado no Código 4.3, para simular o robô utilizando a cinemática inversa. Nesta etapa, o robô executou a tarefa de pegar caixas e transferi-las para novas posições em três repetições, sendo necessário o deslocamento de três caixas.

```

1 #Realizando os calculos da cinematica inversa de conversao de
  posicoes para angulos:
2 q1_val = 0 * 3.1416/180 #Para posicoes em X;
3 q2_val = 0.022 #Para posicoes em Y;
4 q3_val = 0 * 3.1416/180 #Para posicoes em Z;
5
6 #Enviando os valores das posicoes calculadas para cada junta:

```

```

7 sim.setJointTargetPosition(j1, q1_val) #Para a primeira junta (J1)
;
8 sim.setJointTargetPosition(j2, q2_val) #Para a segunda junta (J2);
9 sim.setJointTargetPosition(j3, q3_val) #Para a terceira junta (J3
);
10 time.sleep(2)
11 setEffector(0) #Neste caso levando seu valor a zero (0) = soltando
o objeto;
12 time.sleep(3)
13 sim.setJointTargetPosition(j1, 0) #Para a primeira junta (J1);
14 sim.setJointTargetPosition(j2, 0.2) #Para a segunda junta (J2);
15 sim.setJointTargetPosition(j3, 0) #Para a terceira junta (J3);
16 print('0 robo esta se movendo!') #Exibe-se a imagem de execucao do
robo;
17 time.sleep(2) #'Delay' aplicado apos mover para as posicoes;
18 print('OK!') #Mensagem emitida apos ser concluida a terceira etapa
de posicionamento do robo;
19 time.sleep(3) #'Delay' aplicado apos exibir mensagem;
20 time.sleep(2)
21 setEffector(0) #Neste caso levando seu valor a zero (0) = soltando
o objeto;
22 time.sleep(3)

```

Código 4.3 - *Script* para simulação do robô utilizando a cinemática inversa.

Após a bem-sucedida conclusão da validação da cinemática direta e inversa no CoppeliaSim, foi evidenciada a robustez e eficácia do modelo simulado do robô. Este processo não apenas assegura a precisão e consistência nos movimentos do manipulador, mas também valida a implementação de algoritmos de controle. Com a confirmação da correspondência correta entre as posições desejadas e as obtidas na simulação, foi fortalecida a confiança na capacidade do robô SCARA de executar suas tarefas com precisão e eficiência. Esta validação, realizada no ambiente virtual, forneceu a base sólida para a subsequente implementação e operação do robô no mundo real. Para maiores estudos, a programação completa para a simulação do robô SCARA pode ser encontrada no Apêndice C.

4.4 Controle robô SCARA

A metodologia empregada no controle do robô SCARA abrange o uso de *softwares* específicos para a programação e desenvolvimento de interfaces gráficas, objetivando aprimorar a interação e a execução de tarefas, além de expandir as capacidades operacionais do robô. A modelagem cinemática, essencial para a definição precisa dos movimentos e posições, juntamente com a comunicação efetiva entre os componentes

de *hardware* e *software*, forma a base para o controle confiável. Este processo integral não apenas facilita a implementação de comandos complexos, mas também contribui significativamente para a eficiência e precisão do SCARA em aplicações práticas.

4.4.1 Seleção dos *softwares* de programação

O Arduino *Integrated Development Environment* (IDE) foi selecionado como a ferramenta primordial para o desenvolvimento dos códigos de programação destinados ao controle de movimentação do robô SCARA. O Arduino IDE é o ambiente de desenvolvimento integrado amplamente utilizado na programação de microcontroladores, especialmente os da família Arduino. Segundo Monk (2016), ele se baseia na linguagem de programação C/C++ e oferece recursos específicos para simplificar o processo de desenvolvimento de código para microcontroladores. A escolha da plataforma Arduino IDE é respaldada em quatro motivos principais: i) suporte à linguagem, ii) abundância de recursos, iii) compatibilidade e iv) facilidade de uso.

O Arduino IDE é compatível com a linguagem de programação C/C++, permitindo que os desenvolvedores aproveitem a precisão e a eficiência desta linguagem ao criar algoritmos de controle para o robô SCARA. Além disso, esta IDE desfruta da vasta comunidade de usuários e desenvolvedores. Isto significa que há abundância de recursos educacionais e documentações acadêmicas disponíveis, o que facilita a resolução de problemas e auxilia no aprimoramento das habilidades de programação (LIMA; VILLAÇA, 2012). O Arduino IDE foi projetado especificamente para funcionar em conjunto com o *hardware* da plataforma Arduino. Isto garante a integração eficiente com os componentes utilizados no robô SCARA, simplificando o processo de desenvolvimento. Por fim, sua interface é projetada de maneira amigável e acessível, tornando-o adequado tanto para programadores iniciantes quanto para experientes.

Como escolha da ferramenta ideal para a criação da *Graphical User Interface* (GUI) destinada ao controle do robô SCARA, optou-se pela seleção do *software* Processing. Conforme definido por Hughes (2016), o Processing é a plataforma de desenvolvimento de *software* de código aberto projetada para artistas, *designers* e programadores. Ele se baseia na linguagem de programação Java e é amplamente conhecido por sua capacidade de criar interfaces gráficas interativas de maneira intuitiva. A plataforma oferece o ambiente de desenvolvimento que facilita a criação de aplicativos visuais e interativos, tornando-o a escolha adequada para a criação da GUI do robô SCARA. A escolha do Processing é fundamentada por quatro razões principais: i) *design* intuitivo, ii) compatibilidade e iii) integração com o Arduino e iv) facilidade de desenvolvimento.

O Processing é especializado na criação de interfaces gráficas interativas, o que é imperativo para proporcionar a experiência de aprendizado intuitiva aos usuários que desejam controlar o robô. De acordo com Junior et al. (2013), este *software* oferece ampla gama de bibliotecas que facilitam a criação de elementos visuais, como botões, *sliders*¹⁰ e gráficos. Isto proporciona flexibilidade na concepção da GUI, tornando-a adaptável às necessidades educacionais. Além disso, a compatibilidade entre o Processing e o Arduino IDE permite a comunicação eficiente entre a GUI e o código de controle do robô SCARA. Isto simplifica a troca de dados e comandos. Por fim, a abordagem visual do Processing simplifica o desenvolvimento da GUI, mesmo para estudantes e educadores que não possuam experiência prévia em *design* de interfaces. Isto acelera o processo de criação e é altamente vantajoso em ambientes educacionais.

A escolha do Arduino IDE e do Processing como ferramentas de desenvolvimento para o robô SCARA se baseia na complementaridade de suas funções. O Arduino IDE se concentra no controle de movimentação e na comunicação direta com os componentes do robô, enquanto o Processing é responsável por criar a interface gráfica de usuário amigável para o controle e monitoramento das operações do robô.

4.4.2 Desenvolvimento do sistema de controle

Para o desenvolvimento do *software* de controle do robô SCARA, foi realizada a concepção do diagrama de blocos que delineasse o sistema de controle. Este diagrama foi projetado para proporcionar a representação visual e sistemática do processo de controle, facilitando a compreensão da estrutura operacional e dos fluxos de dados inerentes ao sistema. A Figura 4.29, ilustra o diagrama de blocos do sistema de controle do robô SCARA.

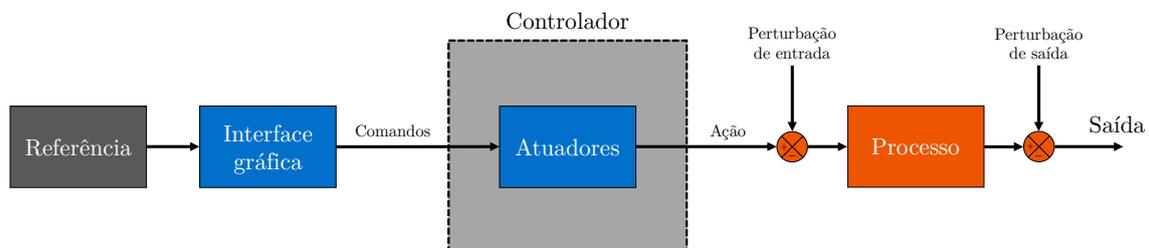


Figura 4.29 - Diagrama de blocos do sistema de controle do robô SCARA.

¹⁰Referem-se a elementos gráficos de interface do usuário que permitem aos usuários ajustar o valor dentro do intervalo predefinido, geralmente representado como barra de controle deslizante

O diagrama de blocos permite identificar e abordar potenciais problemas de integração entre os componentes de *software* e *hardware*. Por exemplo, ele pode destacar como os sinais de controle são gerados e transmitidos, e como as perturbações podem afetar o sistema, possibilitando o desenvolvimento de estratégias para mitigar tais influências (ROCHA, 2016).

Este diagrama, caracteriza-se como sistema de malha aberta devido à ausência do mecanismo de *feedback* que realimenta a saída do sistema de volta ao seu controle. No sistema de malha aberta, os comandos de controle são enviados aos atuadores com base apenas nas entradas ou referências fornecidas pelo usuário, sem qualquer ajuste ou correção baseado na posição ou estado real do sistema após a execução dos comandos. Segundo Ganzaroli (2023), neste arranjo não há garantia da equivalência entre o sinal de saída e o sinal de referência. Isto porque podem haver variações nos parâmetros da planta¹¹ ou o surgimento de perturbações tanto na entrada quanto na saída da mesma.

O componente de referência representa os comandos ou desejos do operador, que são inseridos através da GUI. Ela é projetada de maneira intuitiva para permitir que o usuário especifique parâmetros como a posição, velocidade, aceleração e o estado do efetuador final do robô. Após a entrada de dados pelo usuário, a interface gráfica é o equivalente ao transdutor de entrada, que converte as ações do usuário em sinais eletrônicos que podem ser interpretados pelo sistema de controle. Este processo envolve a digitalização das entradas analógicas fornecidas pelo usuário, como movimentos de controle deslizante ou cliques de botão, em valores numéricos que o controlador possa processar.

O código no sistema Arduino e Processing atuam como o controlador, que processa os sinais de entrada e os utiliza para determinar os comandos de controle adequados para os atuadores do robô. Este bloco ajusta os sinais de controle baseados nas entradas do usuário sem verificar o estado real da planta. Este componente é o cérebro do sistema, no qual a lógica de controle é executada, incluindo cálculos de cinemática direta ou inversa, para traduzir as intenções do usuário em movimentos específicos do robô. Os atuadores são os componentes que realizam o trabalho físico, convertendo os sinais de controle elétrico do controlador em ações mecânicas. No caso do robô SCARA, os atuadores incluem motores de passo que controlam o movimento das juntas do robô e o servomotor que opera o efetuador final.

¹¹No contexto de sistemas de controle, refere-se ao processo físico ou mecanismo que está sendo controlado, ver também em Ganzaroli (2023).

A planta, também chamada de processo, neste contexto, refere-se ao próprio robô SCARA e a todos os seus componentes mecânicos e elétricos que respondem aos comandos dos atuadores para realizar as tarefas desejadas. A planta é o sistema físico que está sendo controlado, e seu comportamento é o resultado direto das ações dos atuadores baseadas nos comandos do controlador. O diagrama também pode incluir perturbações, que representam fatores externos ou internos que podem afetar o desempenho do sistema, como variações na carga ou na alimentação elétrica. Estas perturbações são importantes para considerar no projeto do sistema, pois podem influenciar a precisão e a confiabilidade do robô. Perturbações de entrada, como flutuações de tensão, interferência eletromagnética e erros de comunicação, afetam a precisão e execução dos comandos no sistema de controle do robô SCARA, enquanto as perturbações de saída, incluindo desgaste mecânico, variações de carga e fricção, influenciam o desempenho do robô após a emissão dos comandos.

4.4.2.1 Modelagem cinemática do robô SCARA

O controle do robô SCARA, incluindo seu posicionamento e orientação, foi facilitado pela implementação de cinemática direta e inversa no *software* de controle. Dada a estrutura simplificada do SCARA, que possui o número limitado de graus de liberdade, foi possível derivar as equações cinemáticas utilizando o método geométrico. A disposição e movimento do robô SCARA no plano $x-y$ são ilustrados esquematicamente na Figura 4.30.

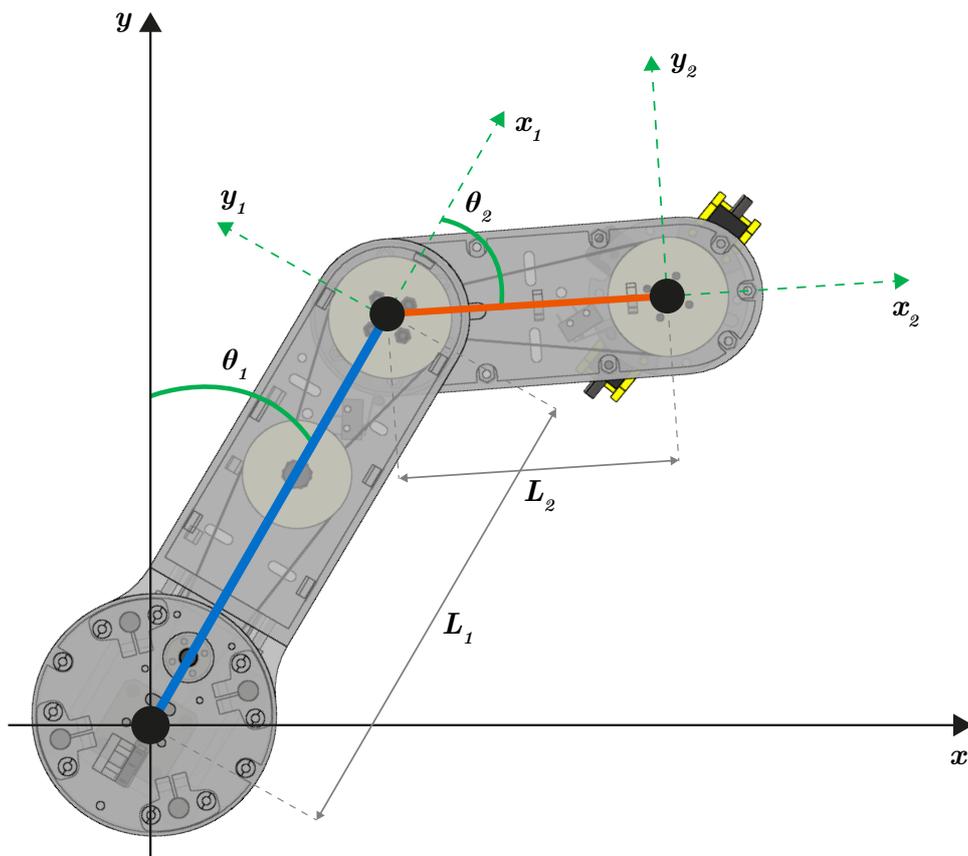


Figura 4.30 - Representação da vista superior x - y do robô SCARA.

na qual y_1 - x_1 e y_2 - x_2 são os planos da segunda e terceira junta do manipulador, respectivamente, θ_1 e θ_2 são os ângulos referentes à primeira e segunda junta do robô, respectivamente, ambos medidos em graus $[\circ]$ e L_1 e L_2 correspondem ao comprimento do primeiro e segundo elo, respectivamente, ambos medidos em metros $[m]$. As coordenadas em y_2 e x_2 representam as coordenadas do final do segundo elo, ou seja, a posição final que o manipulador irá atingir.

É possível realizar a análise da Figura 4.30 através das equações (2.14) e (2.15) anteriormente apresentadas. Com os tamanhos dos elos do robô SCARA definidos como 228mm e $136,5\text{mm}$ para o primeiro e segundo elos, respectivamente, a inserção destes valores nestas equações resulta na cinemática direta específica para o robô construído. Assim, estas equações possibilitam o cálculo das coordenadas y_2 e x_2 a partir dos ângulos θ_1 e θ_2 . Importante ressaltar que a coordenada linear z_p é determinada de maneira independente dos ângulos das juntas rotativas, sendo diretamente influenciada pelo motor que regula este eixo.

A análise trigonométrica baseada na Figura 4.30 mencionada conduz às equações (2.16) e (2.17) que permitem calcular os ângulos θ_1 e θ_2 necessários para alcançar a posição cartesiana específica, definida pelas coordenadas y_2 e x_2 . Esta abordagem caracteriza a cinemática inversa do manipulador, facilitando a determinação dos ângulos das juntas para a localização desejada no plano. A coordenada do eixo z é tratada separadamente devido à sua relação direta com a junta prismática, independente das juntas rotacionais e sua operação linear.

4.4.3 Comunicação entre Arduino e Processing

O sistema de controle para o robô SCARA, pode ser visualizado através dos códigos de programação disponíveis nos Apêndice D. O *script* desenvolvido no Arduino IDE gerencia a comunicação entre o microcontrolador e a parte eletroeletrônica do projeto, possibilitando a movimentação do manipulador, conforme evidenciado no Código D.1 disponível em apêndice.

A comunicação entre o Arduino IDE e o Processing no programa do robô SCARA é estabelecida via comunicação serial, permitindo a troca de dados entre o *hardware*, que é o Arduino UNO e o *software*, que é o Processing. Esta comunicação é bidirecional e essencial para o controle do robô, pois permite que o Processing envie comandos de controle para o Arduino, e que o Arduino envie respostas ou confirmações de volta ao Processing. O Código 4.4, desenvolvido em C++, expõe o trecho específico de como é realizado o início da comunicação serial¹² entre o Arduino e o Processing.

```
1 //Inicio do codigo de configuracao do controle do robo:
2 void setup() {
3   Serial.begin(115200);
```

Código 4.4 - *Script* para transmissão de dados entre o Arduino e o Processing.

No Arduino, a comunicação serial é iniciada no método `setup()` com a chamada `Serial.begin(115200);`, o que configura a taxa de transmissão para 115200 *bits* por segundo. Esta taxa deve ser compatível com a configurada no Processing para garantir a comunicação eficaz. O processo de envio de dados para o Processing é realizado usando comandos como `Serial.print()`, `Serial.write()` e `Serial.read()`. Estes comandos podem ser usados para enviar dados, tais como status de operações ou confirmações de ações completadas, do Arduino para o Processing.

¹²Comunicação serial é o método de transmissão de dados em que a informação é enviada ou recebida *bit a bit* sequencialmente, através de canal único, sendo amplamente utilizada em dispositivos eletrônicos para troca de dados entre microcontroladores e periféricos.

Para estabelecer a comunicação serial no Processing, inicia-se criando e inicializando o objeto da classe serial dentro do método `setup()`. Este processo envolve a seleção da porta serial apropriada, neste caso, a COM3, e a configuração da taxa de transmissão para corresponder à do Arduino. No ambiente do Processing, a GUI permite que os usuários insiram parâmetros de controle, que são então enviados ao Arduino como *strings* serializadas usando o método `write()` do objeto serial. O Arduino, por sua vez, monitora continuamente a porta serial para detectar dados recebidos, utilizando o método `Serial.available()`. Ao receber dados, o Arduino os lê e emprega técnicas de manipulação de *strings* para extrair os comandos individuais, que são então executados para controlar os movimentos e ajustes necessários no robô SCARA.

4.4.4 Programação de movimentação do robô

Conforme delineado nos códigos do Arduino e Processing, estes incorporam o conjunto de algoritmos para determinar as posições do efetuador final e controlar os movimentos do robô. O controle é efetuado por meio da interface gráfica de usuário que permite a entrada de dados e a seleção de parâmetros para as operações do robô. A Figura 4.31, ilustra o fluxograma da programação realizada no Arduino IDE, evidenciado a comunicação entre o Arduino e o Processing.

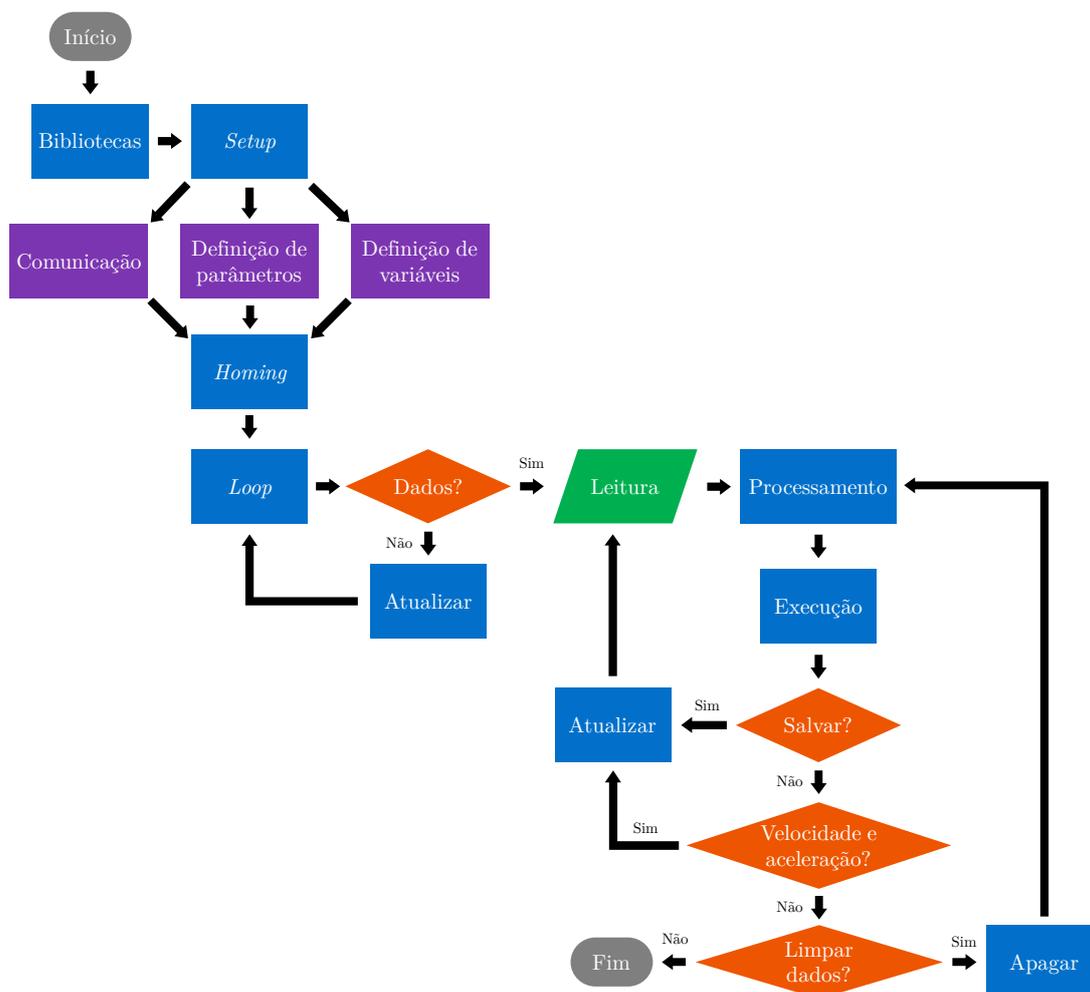


Figura 4.31 - Fluxograma da programação do sistema de controle.

O código começa com a fase de inicialização, na qual bibliotecas específicas são carregadas para facilitar o controle dos motores de passo e da garra através da inclusão das bibliotecas `AccelStepper`, `Servo` e `math`. Estas bibliotecas permitem o ajuste fino dos motores e a execução de cálculos matemáticos complexos necessários para a cinemática do robô. Em seguida, o programa define os pinos de entrada para as chaves de fim de curso, configurando-os como entradas com resistores de `pull-up` internos. A configuração inicial dos motores de passo é realizada, estabelecendo parâmetros como a velocidade máxima e a aceleração para cada motor. Isto é feito utilizando comandos que definem estes valores para os objetos motor criados pela biblioteca `AccelStepper`.

Cada motor de passo passa pela rotina de *homing*, no qual o motor se move até a

direção até que o respectivo sensor de fim de curso seja acionado. Isto indica que o motor alcançou sua posição inicial. Após atingir o fim de curso, o motor pode ser movido ligeiramente para longe do sensor para estabelecer a posição de início consistente. Este processo é realizado sequencialmente para o motor do eixo z , o motor da junta J3, o motor da junta J2 e finalmente o motor da junta J1. No momento em que todos os motores de passo tenham completado a rotina de *homing* e o servomotor da garra esteja na posição inicial, o robô está pronto para iniciar suas operações a partir do estado conhecido e seguro.

O *loop* principal do programa espera pela comunicação via serial. Quando os dados são recebidos, o Arduino lê o *buffer* serial e processa os dados, identificando comandos para movimentar os motores ou operar a garra. Isto envolve converter os comandos em valores de passos para os motores, utilizando constantes de conversão predefinidas. Durante a execução, o robô realiza as ações com base nos comandos processados. Se o comando for salvar a posição atual, ele armazena os valores dos passos em vetores. Se o comando for para ajustar a velocidade ou aceleração, o programa altera os parâmetros dos motores de passo em tempo real. Em caso de comando para limpar os dados, o programa redefine os vetores de posição, preparando o sistema para receber novas instruções.

Se nenhum dado estiver disponível, ou após a execução do comando, o programa pode optar por atualizar o sistema. Isto pode envolver redefinir as variáveis ou preparar o sistema para receber o próximo conjunto de instruções. O programa continuará neste *loop*, alternando entre esperar por novos dados, processá-los e executar comandos, até que seja desligado ou reiniciado, mantendo o robô pronto para responder a qualquer novo comando recebido.

4.4.5 Interface gráfica de controle

Por meio da programação efetuada no ambiente Processing, foi viabilizada a criação da GUI, na qual o utilizador pode efetuar a atualização de todos os parâmetros relacionados ao funcionamento do robô e efetuar seu controle. A Figura 4.32, ilustra a tela de inicialização desta interface gráfica, que serve como plataforma de comando e supervisão completa das operações do manipulador robótico.



Figura 4.32 - Programa de interface e controle do robô SCARA.

A interface gráfica do usuário é desenvolvida utilizando a biblioteca `controlP5` no ambiente Processing IDE, possibilitando a criação eficaz de elementos como botões, *sliders* e campos de texto. O controle da cinemática direta é realizado ao inserir os valores desejados para os ângulos de operação do robô. A Figura 4.33, ilustra a área na GUI destinada à manipulação dos ângulos.

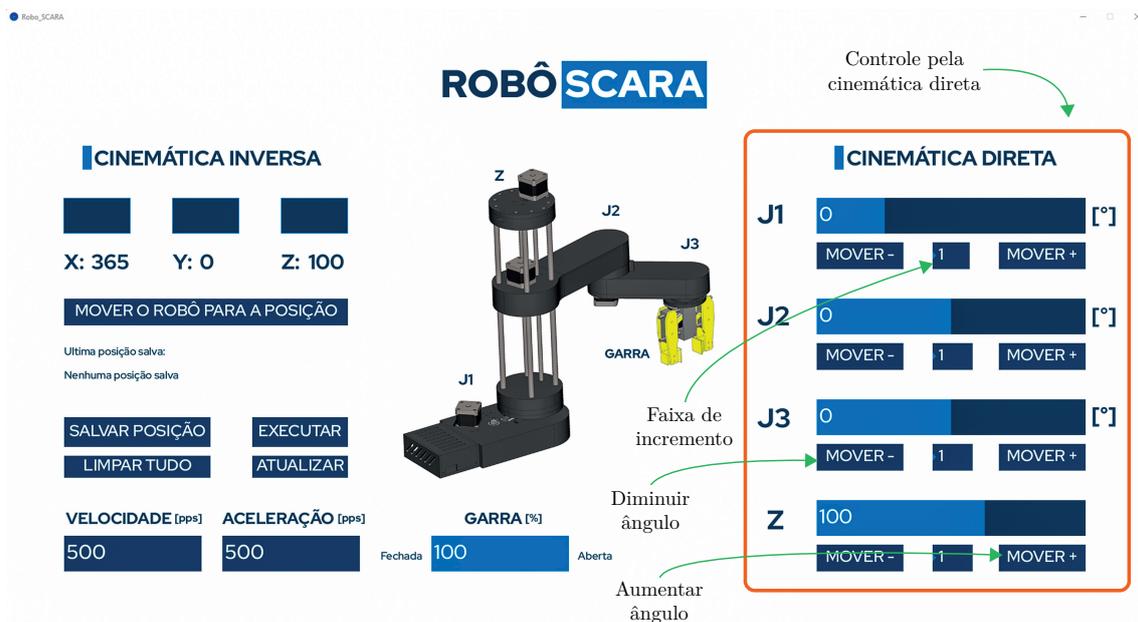


Figura 4.33 - Inserção de ângulos para uso da cinemática direta.

Os ajustes dos ângulos podem ser efetuados por meio das caixas de texto e dos controles deslizantes destacados em laranja na Figura 4.33. Estes valores são determinados pelo usuário e direcionam o manipulador para as posições desejadas. As juntas J1, J2 e J3, medidas em graus $[\circ]$, estão limitadas dentro dos intervalos físicos do robô SCARA. A junta J1 possui o intervalo de -90° a 266° , enquanto a junta J2, responsável pela rotação do segundo elo, varia de -150 a 150° . A junta J3, que controla a rotação do efetuador final, opera no intervalo de -162 a 162° . Estes limites foram estabelecidos com base nas características físicas e de movimentação do robô SCARA.

Adicionalmente, a atualização dos valores dos ângulos pode ser realizada por meio de botões de deslocamento. O botão esquerdo diminui o valor do ângulo, enquanto o botão direito incrementa. A faixa de incremento pode ser ajustada entre 1 e 50, com a preocupação de preservar a integridade mecânica do robô a longo prazo. Atualizações únicas superiores a 50° podem comprometer a estrutura do dispositivo. Os *sliders* também oferecem a funcionalidade de atualização, permitindo que o usuário arraste-os para o valor desejado.

Para o movimento prismático ao longo do eixo z , o intervalo de valores varia de 0 a 160 unidades na GUI. Este intervalo corresponde diretamente ao alcance físico

do eixo z do robô SCARA, que é de $400mm$. A relação proporcional é estabelecida entre os valores do controle deslizante e a altura física do eixo z . A equação (4.6) estabelece o parâmetro para calcular a altura correspondente ao valor do controle deslizante:

$$h_z = \frac{C_{des}}{160} \cdot 400 \quad (4.6)$$

no qual h_z é a altura do eixo z , medida em milímetros [mm] e C_{des} é o valor unitário que será aplicado no *slider*. Esta conversão permite que o usuário ajuste a altura do eixo z intuitivamente na GUI, sem a necessidade de realizar cálculos manuais. O *software* de controle traduz estes valores em comandos de movimento para o motor de passo que controla o eixo z , garantindo que o robô SCARA alcance a altura desejada dentro de seu alcance físico. O Código 4.5, desenvolvido em Java, demonstra a utilização das equações trigonométricas referentes ao cálculo da cinemática direta no *software* de controle.

```

1 //CINEMATICA DIRETA:
2 //Realizando todos os calculos de cinemática direta conforme suas
   formulas pre-estabelecidas:
3 void forwardKinematics() {
4   float theta1F = theta1 * PI / 180; //Graus para radianos;
5   float theta2F = theta2 * PI / 180;
6   xP = round(L1 * cos(theta1F) + L2 * cos(theta1F + theta2F));
7   yP = round(L1 * sin(theta1F) + L2 * sin(theta1F + theta2F));
8
9 //Funcao criada para atualizar os valores e os mesmos serem
   enviados ao Arduino IDE, realizando a comunicacao integrada
   com o 'Processing':
10 public void updateData() {
11   data = str(saveStatus)
12     + "," + str(runStatus)
13     + "," + str(round(cp5.getController("j1Slider").getValue()))
14     + "," + str(round(cp5.getController("j2Slider").getValue()))
15     + "," + str(round(cp5.getController("j3Slider").getValue()))
16     + "," + str(round(cp5.getController("zSlider").getValue()))
17     + "," + str(gripperValue)
18     + "," + str(speedSlider)
19     + "," + str(accelerationSlider);
20 }
21 }

```

Código 4.5 - *Script* para o cálculo da cinemática direta.

Desta forma, por meio da cinemática direta, são determinados os valores de x_2 e y_2 do manipulador, considerando os ângulos de articulação definidos para os dois elos do robô, θ_1 e θ_2 , bem como os comprimentos L_1 e L_2 destes elos. A Figura 4.34, apresenta a seção da interface gráfica de usuário destinada à inserção das coordenadas cartesianas desejadas para o manipulador.

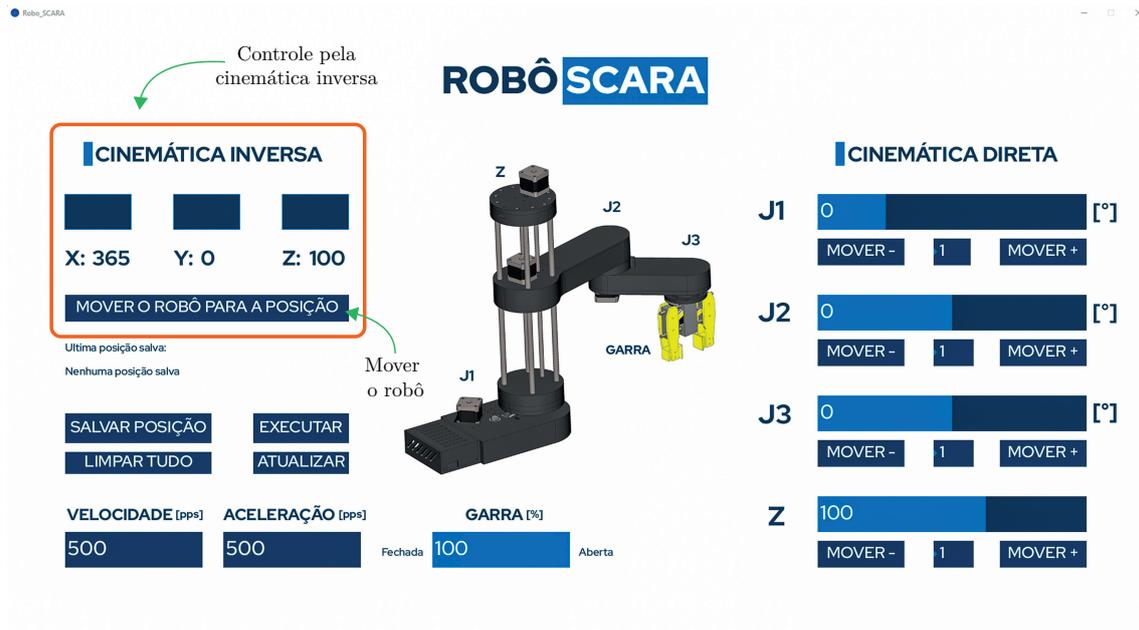


Figura 4.34 - Inserção de coordenadas cartesianas para uso da cinemática inversa.

Na GUI de controle da cinemática inversa, é possível definir as coordenadas cartesianas desejadas para o robô por meio dos campos de texto destacados em laranja na Figura 4.34. No contexto da cinemática inversa, os valores de x , y e z representam as coordenadas cartesianas da posição desejada do efetuador final do robô no espaço de trabalho físico do mesmo, sendo expressas em centímetros [cm]. Ao acionar o botão de deslocamento do robô para a posição desejada, os dados cartesianos são automaticamente convertidos em comandos de motor que orientam o robô a alcançar as posições almeçadas para cada eixo, assegurando, assim, que o robô alcance a posição desejada. O *software* executa esta conversão das coordenadas cartesianas para os ângulos necessários que o robô deve adotar. Através da cinemática inversa, os ângulos articulares, θ_2 e θ_1 , são calculados em conformidade com as coordenadas fornecidas, permitindo ao robô posicionar-se com precisão.

O Código 4.6, elaborado em linguagem de programação Java, exemplifica a aplicação das coordenadas cartesianas para o cálculo da cinemática inversa no *software* de controle.

```
1 //CINEMATICA INVERSA:
2 void inverseKinematics(float x, float y) {
3     theta2 = acos((sq(x) + sq(y) - sq(L1) - sq(L2)) / (2 * L1 * L2))
4     ;
5     if (x < 0 & y < 0) {
6         theta2 = (-1) * theta2;
7     }
8     theta1 = atan(x / y) - atan((L2 * sin(theta2)) / (L1 + L2 * cos(
9         theta2)));
10    theta2 = (-1) * theta2 * 180 / PI;
11    theta1 = theta1 * 180 / PI;
12
13 //Ajuste de angulos dependendo em qual quadrante esta a coordenada
14 final (X,Y):
15 if (x >= 0 & y >= 0) { //1 quadrante;
16     theta1 = 90 - theta1;
17 }
18 if (x < 0 & y > 0) { //2 quadrante;
19     theta1 = 90 - theta1;
20 }
21 if (x < 0 & y < 0) { //3 quadrante;
22     theta1 = 270 - theta1;
23     phi = 270 - theta1 - theta2;
24     phi = (-1) * phi;
25 }
26 if (x > 0 & y < 0) { //4 quadrante;
27     theta1 = -90 - theta1;
28 }
29 if (x < 0 & y == 0) {
30     theta1 = 270 + theta1;
31 }
32
33 //Calcula o angulo "phi" para que a garra fique paralela ao eixo X
34 :
35 phi = 90 + theta1 + theta2;
36 phi = (-1) * phi;
37
38 //Ajuste de angulos dependendo em qual quadrante esta a coordenada
39 final (X,Y):
40 if (x < 0 & y < 0) { //3 quadrante;
41     phi = 270 - theta1 - theta2;
```

```

37 }
38 if (abs(phi) > 165) {
39     phi = 180 + phi;
40 }

```

Código 4.6 - *Script* para o cálculo da cinemática direta.

Conforme o quadrante em que a posição é especificada, determinados ajustes nos ângulos das juntas são efetuados por meio de instruções condicionais *if*. O terceiro ângulo, denominado *phi*, desempenha o papel fundamental ao definir a orientação de rotação do efetuador final.

Na interface gráfica criada no ambiente de desenvolvimento Processing, são disponibilizados dois *sliders* destinados à configuração da velocidade e aceleração. Esta representação visual está ilustrada na Figura 4.35.

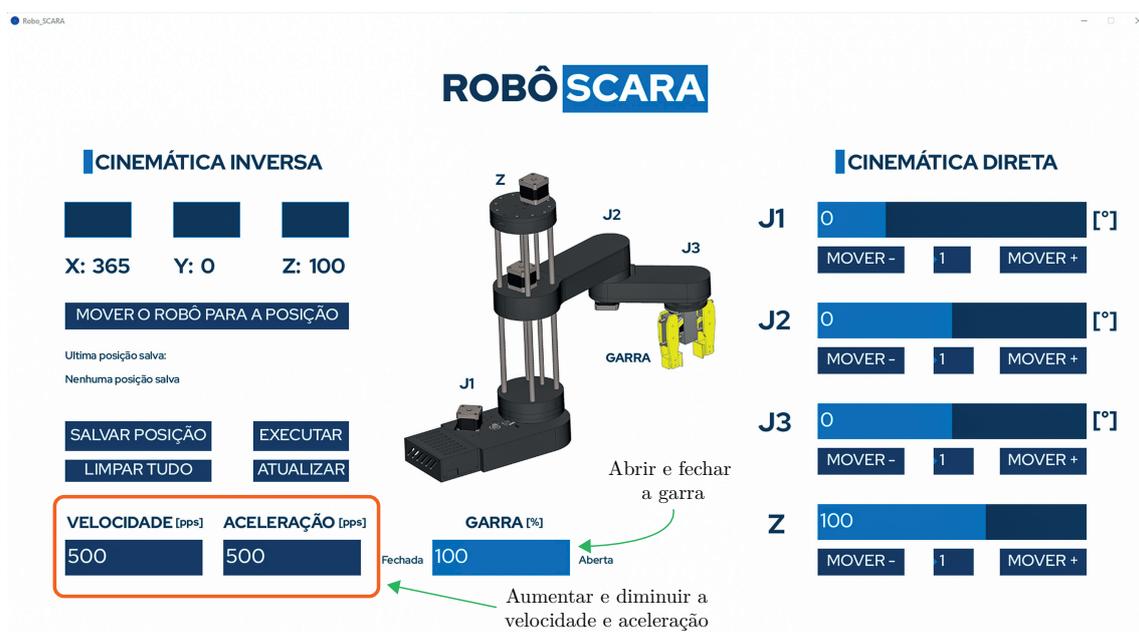


Figura 4.35 - Atualização da velocidade e aceleração dos motores.

Durante a interação com o *slider* de velocidade, o usuário define a rapidez com que deseja que o robô SCARA mova suas juntas. Da mesma forma, o *slider* de aceleração determina a taxa na qual a velocidade de cada junta pode aumentar ou diminuir durante o movimento. Quando o usuário modifica a posição do controle deslizante,

o evento é acionado no ambiente de processamento. Este evento é capturado pela função de *callback*¹³ associada ao *slider*, que, por sua vez, atualiza a variável global correspondente ao valor do controle deslizante. Este valor global é então formatado na *string* de dados junto com outros parâmetros de controle e transmitido ao Arduino através da comunicação serial.

No lado do Arduino, o código responsável por receber esta *string* de dados a desagrega para extrair os valores individuais de velocidade e aceleração. Após a extração, estes valores são aplicados aos motores de passo usando funções disponibilizadas pela biblioteca `AccelStepper`. Esta biblioteca permite o controle preciso sobre os motores de passo, fornecendo métodos para definir a velocidade máxima e a aceleração. Utilizando estes métodos, o Arduino ajusta o comportamento dos motores de passo de acordo com os parâmetros definidos pelo usuário. Os valores de velocidade e aceleração são configurados na GUI, respectivamente com o intervalo que varia de 500 a 4000 passos por segundo, medidos em $[pps]$ e de 500 a 4000 passos por segundo ao quadrado, medidos em $[pps^2]$. Estes valores inseridos na GUI são diretamente utilizados pelas funções da biblioteca `AccelStepper` no Arduino.

No que se refere ao efetuator final, a GUI define o intervalo de 0 a 100%, o qual não se traduz diretamente no ângulo de rotação do servomotor. Em vez disto, este intervalo é usado como a representação percentual da abertura máxima da garra. No código do Arduino, este valor percentual é mapeado para o alcance real do servomotor, que varia de 0 a 90° no caso do robô SCARA. Esta limitação de 90° decorre de restrições físicas na construção do efetuator final, que impedem o movimento de rotação mais amplo do servomotor. O valor de 0 na GUI corresponde à garra totalmente fechada, enquanto o valor de 100 representa a garra totalmente aberta. Todo o código-fonte da programação utilizado para criar a interface de controle pode ser encontrado no Apêndice D, especificamente no Código D.2.

4.4.5.1 Modo de gravação do manipulador

Foi implementado o modo de gravação, em que é possível gravar posições nas quais o robô se encontra, para posterior execução automática. Este modo de gravação cria a interface simples de programação de movimentos que é útil para determinar como serão as atividades de manipulação de objetos durante sua tarefa. O número de posições salvas, é exibido juntamente com os valores de cada junta da última

¹³ *Callback* é a função que é passada como argumento para outra função e é executada após a conclusão de determinada operação ou evento. É comumente usado em programação assíncrona para lidar com eventos e respostas de chamadas de sistema.

posição gravada, conforme ilustrado na Figura 4.36.

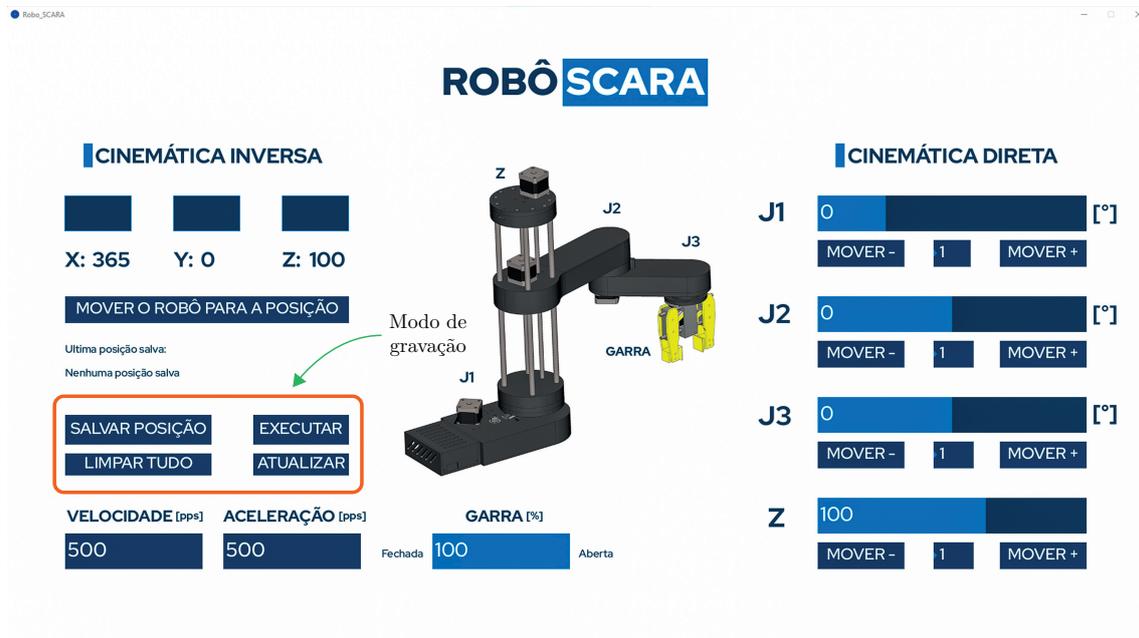


Figura 4.36 - Salvamento de posições no *software* de controle do robô.

Quando o usuário deseja salvar a configuração específica de posição, a ação do botão salvar posição desencadeia a gravação dos valores atuais dos parâmetros de movimento na estrutura de dados, que neste caso é a lista. O processo envolve a leitura sequencial das posições armazenadas e o envio de comandos ao robô para que ele se mova de acordo com estas posições. O processo é iniciado pelo acionamento do botão executar, que coloca o sistema no estado de execução. À medida que o sistema percorre a lista de posições salvas, os comandos correspondentes são transmitidos ao Arduino, que os interpreta e executa as ações físicas dos motores.

Além disso, a função de limpeza permite ao usuário redefinir o sistema, apagando todas as posições previamente armazenadas. Isto é essencial para começar o novo conjunto de movimentos a partir do zero ou corrigir sequências de movimentos que não são mais necessárias ou que foram salvas erroneamente. O sistema também oferece a capacidade de atualizar as configurações de movimento em tempo real, sem interromper o processo em curso. Isto pode incluir ajustes precisos de posição, velocidade, aceleração ou a abertura do efetuator final, permitindo que o operador refine o movimento do robô durante a operação contínua.

4.5 Manutenção do robô SCARA

Para garantir o desempenho eficaz e confiável do robô SCARA, é essencial implementar o regime regular de inspeção e manutenção (GRAU et al., 2020). A finalidade desta abordagem é evitar falhas durante a operação e identificar possíveis defeitos antes que eles afetem adversamente o sistema. Recomenda-se, portanto, a execução de manutenção preventiva periódica para avaliar as condições do robô. Conforme estabelecido por D'Abreu e Chella (1999), o registro meticuloso de inspeções e eventuais reparos é fundamental para manter o histórico completo das intervenções realizadas. As inspeções podem ser conduzidas de maneira visual ou com o auxílio de instrumentos apropriados. É imperativo prestar atenção a certos sinais indicativos que podem sugerir problemas no sistema. Sinais comuns a serem observados incluem ruídos anormais, vibrações anômalas ou superaquecimento de componentes do manipulador.

Os pontos-chave de inspeção são fundamentados como cinco principais: i) tensão das correias, ii) folgas dos fixadores, iii) alinhamento das polias, iv) lubrificação dos rolamentos e v) deformação plástica de peças.

A correta tensão aplicada nas correias é fundamental para evitar problemas durante a operação. A correia muito tensionada contribui para atrito excessivo e perda de potência na transmissão, já a correia frouxa pode causar saltos, imprecisões de posicionamento e intensificar seu próprio desgaste (MELLO, 2016). Conforme o uso do robô, podem surgir folgas nos parafusos responsáveis por manter a junta alinhada e nos parafusos de fixação dos motores, podendo dar origem a baixa tensão nas correias e a danos em componentes mais próximos as uniões. Com o acionamento do manipulador, é natural que surjam folgas com o passar do tempo, podendo surgir o desalinhamento entre as polias o que deve ser corrigido para não intensificar o desgaste da correia pelo atrito excessivo com a borda da polia.

Embora a vedação seja a característica predominante nos rolamentos, verifica-se que os rolamentos axiais e radiais, essenciais para o funcionamento dos elos, requerem lubrificação periódica com o agente apropriado para a sua aplicação específica. A manifestação de ruídos durante operações de movimentação pode ser interpretada como o indicativo de insuficiência na lubrificação. Adicionalmente, considerando que diversas componentes do dispositivo manipulador são confeccionadas em PLA, estas estão sujeitas a deformações plásticas em situações de sobrecarga ou incremento de temperatura. A detecção de alterações plásticas, evidenciadas por modificações na forma ou textura das peças, sinaliza a necessidade de substituição dos componentes

comprometidos (EVANS, 2012). É pertinente ressaltar a susceptibilidade do PLA a deformações notáveis sob exposição a temperaturas elevadas, o que sublinha a relevância da seleção da cor preta para o robô, visando aprimorar a detecção de tais irregularidades. A identificação de deformações nas peças exige a pronta substituição das mesmas.

4.6 Discussões

A implementação de robôs no ambiente educacional têm sido área de crescente interesse e inovação. O projeto de construção do robô SCARA, especificamente, insere-se neste contexto como o exemplo prático da aplicação de conceitos de engenharia mecânica, eletrônica e de controle para resolver problemas complexos de automação e manipulação de objetos. O desenvolvimento do robô SCARA revelou desafios significativos no processo de fabricação, particularmente na rigidez estrutural. A dependência da primeira junta para sustentar o eixo z e os dois elos resultou em tensões consideráveis na base, exacerbadas pelo uso de PLA, material com tendência a flexionar sob carga. Esta condição foi agravada pela escolha de correias como mecanismo de transmissão, cuja folga intrínseca comprometeu adicionalmente a rigidez do sistema.

No que tange ao processo de fabricação 3D, observou-se que a alta densidade e preenchimento das peças resultaram em componentes excessivamente pesados, prejudicando especialmente o movimento de rotação do robô. Esta condição, somada ao peso significativo do motor na junta J3, impôs desafios na manutenção do centro de massa e na integridade estrutural do robô, culminando em repetidas falhas e quebras nos componentes de sustentação do eixo z , o que atrasou todo o processo de montagem do robô. Durante a montagem mecânica, identificou-se a necessidade da revisão no projeto do efetuador final. A limitação física imposta pela modelagem restringiu a amplitude de movimento do servomotor, reduzindo a eficiência da garra. Este aspecto destaca a importância de abordagem holística no *design*, que contemple a funcionalidade plena de todos os componentes.

Os cálculos estáticos focaram na determinação das tensões máximas de cisalhamento e normal, considerando a resistência do material PLA utilizado na fabricação dos elos do robô. Os resultados destes cálculos, indicam que as tensões máximas permaneceram significativamente abaixo do limiar de escoamento do PLA, validando a aplicabilidade da equação da linha elástica e, conseqüentemente, a adequação do material escolhido para as exigências estruturais do robô. Por outro lado, a análise dinâmica abordou o cálculo dos torques necessários nas juntas do robô, conside-

rando a massa dos componentes, a posição do centro de massa e os momentos de inércia. Através de iterações nos cálculos, buscando harmonizar os resultados de torque com o *design* e a seleção dos componentes, alcançou-se a integração coesa entre os requisitos de torque e o dimensionamento dos elementos estruturais e motores. Os resultados desta análise demonstram a capacidade do sistema em atender às demandas dinâmicas impostas pelo movimento dos elos e pela operação das juntas.

A montagem eletromecânica e a simulação do robô SCARA, embora tenham enfrentado desafios, demonstraram resultados promissores. O ponto crítico a ser adicionado na discussão é a importância do ajuste correto de corrente dos *drivers* dos motores de passo. Este ajuste é imperativo para prevenir danos tanto aos motores quanto aos componentes eletrônicos associados. Sem o ajuste adequado, há o risco significativo de superaquecimento, o que pode levar à falha prematura dos motores e à possibilidade de danificar irreversivelmente os circuitos eletrônicos, resultando em atrasos no projeto e custos adicionais com reparos e substituições.

A interface gráfica de controle e os algoritmos de cinemática direta e inversa atenderam às expectativas, proporcionando base sólida para futuras otimizações e desenvolvimentos. No desenvolvimento do *software* de controle, a precisão do manipulador, apesar das folgas inerentes ao sistema de transmissão e à estrutura do manipulador, mostrou-se satisfatória. Este resultado ressalta a capacidade de superação dos desafios mecânicos através de soluções de *software* e algoritmos avançados.

A discussão dos resultados do projeto SCARA destaca a complexidade e os desafios enfrentados na construção de sistemas robóticos, no qual as limitações, como a rigidez estrutural e a precisão do movimento, informam aperfeiçoamentos para futuros projetos. Este trabalho não só provê *insights* valiosos para a comunidade acadêmica e profissionais de engenharia sobre a importância da integração entre *design* mecânico, seleção de materiais e métodos de controle e simulação, mas também atua como estudo de caso significativo na otimização de *design* e escolha de materiais. Ele evidencia como alcançar características importantes como o equilíbrio entre desempenho, custo e fabricação.

Em suma, apesar dos desafios enfrentados, o projeto alcançou seus objetivos principais, demonstrando a viabilidade da construção e operação do robô SCARA cuja operação esteja adequada para propósitos didáticos. As lições aprendidas e as soluções desenvolvidas contribuem significativamente para a base de conhecimento em robótica, oferecendo direções para futuras pesquisas e desenvolvimentos na área.

CAPÍTULO 5

CONCLUSÃO

Este estudo buscou o desenvolvimento do robô SCARA cuja operação esteja adequada para propósitos didáticos em engenharia elétrica, abrangendo robótica, eletrônica, controle e programação. A metodologia incluiu a construção mecânica, integração eletromecânica, simulação e criação do *software* de controle. Os resultados validaram a metodologia, evidenciando o sucesso na montagem e funcionalidade do sistema de controle. Desta forma, o robô SCARA mostrou-se eficaz como ferramenta educacional, realçando seu valor no ensino de conceitos chave em diversas áreas da engenharia.

A metodologia é composta por quatro etapas, iniciando com a construção mecânica que abrangeu a escolha do modelo de robô até a fabricação e montagem das partes, juntamente com análises estática e dinâmica para garantir a viabilidade e eficácia do *design*. Esta fase revelou os desafios relacionados à integração de componentes mecânicos e a importância da abordagem metódica na modelagem e simulação para prever o comportamento do sistema. Na etapa de implementação eletroeletrônica, destacou-se o processo de seleção e montagem de circuitos, sensores e atuadores, essenciais para a funcionalidade do robô. Esta fase foi imperativa para estabelecer a interação entre os sistemas mecânico e eletrônico, enfatizando a necessidade de precisão e compatibilidade entre os componentes. A etapa de simulação do robô, proporcionou a validação do desempenho e a identificação de ajustes necessários antes da operação real. Por fim, o desenvolvimento do *software* de controle, consolida a interface de usuário e a lógica de operação do robô, assegurando a funcionalidade de acordo com os requisitos do projeto.

Nas comparações entre as etapas do projeto, o desempenho e eficácia destacaram-se, realçando a eficiência de cada fase no desenvolvimento do robô SCARA. A construção mecânica e a montagem eletroeletrônica estabeleceram a base sólida para o funcionamento do robô. O desenvolvimento do *software* de controle teve impacto significativo na sua operacionalidade. Quanto ao desempenho, apesar da complexidade do sistema integrado, os resultados esperados foram alcançados em termos de: i) funcionalidade mecânica e eletrônica, ii) resposta ao sistema de controle e iii) interação com o usuário pela interface do *software*.

5.1 Trabalhos Futuros

- Desenvolvimento de sistema de controle de malha fechada para o robô SCARA, visando aprimorar a precisão e a resposta dinâmica do robô em diversas condições operacionais.
- Implementação de algoritmos de *Deep Learning* para aprimorar a visão computacional do robô SCARA, permitindo a identificação e manipulação autônoma de objetos com base em características visuais.
- Exploração do uso de materiais alternativos e técnicas de fabricação para os componentes do robô SCARA, buscando melhorias em termos de peso, custo e sustentabilidade.
- Desenvolvimento de plataforma educacional baseada no robô SCARA, incorporando módulos interativos e recursos de aprendizado *online* para estudantes de engenharia e tecnologia.
- Análise aprofundada dos motores de passo utilizados no robô SCARA, incluindo estudos sobre eficiência energética, otimização de torque e redução de vibrações.
- Integração do robô SCARA com sistemas de *Internet of Things* (IoT) para monitoramento remoto, diagnóstico de falhas e atualizações de *software* em tempo real.

APÊNDICE A

Vistas 2D e 3D das modelagens utilizadas

A.1 Estrutura do robô

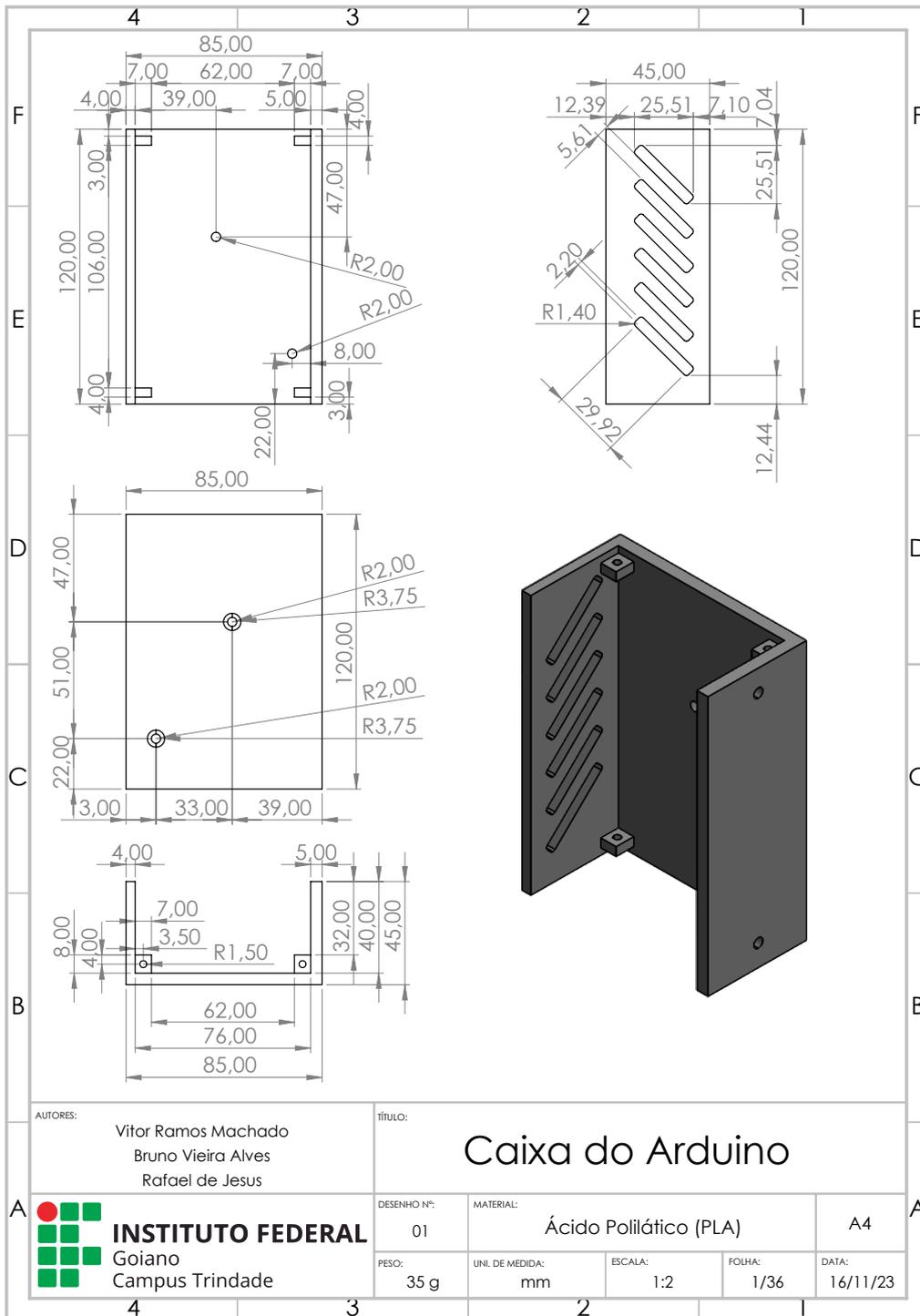


Figura A.1 - Prancheta da caixa do Arduino.

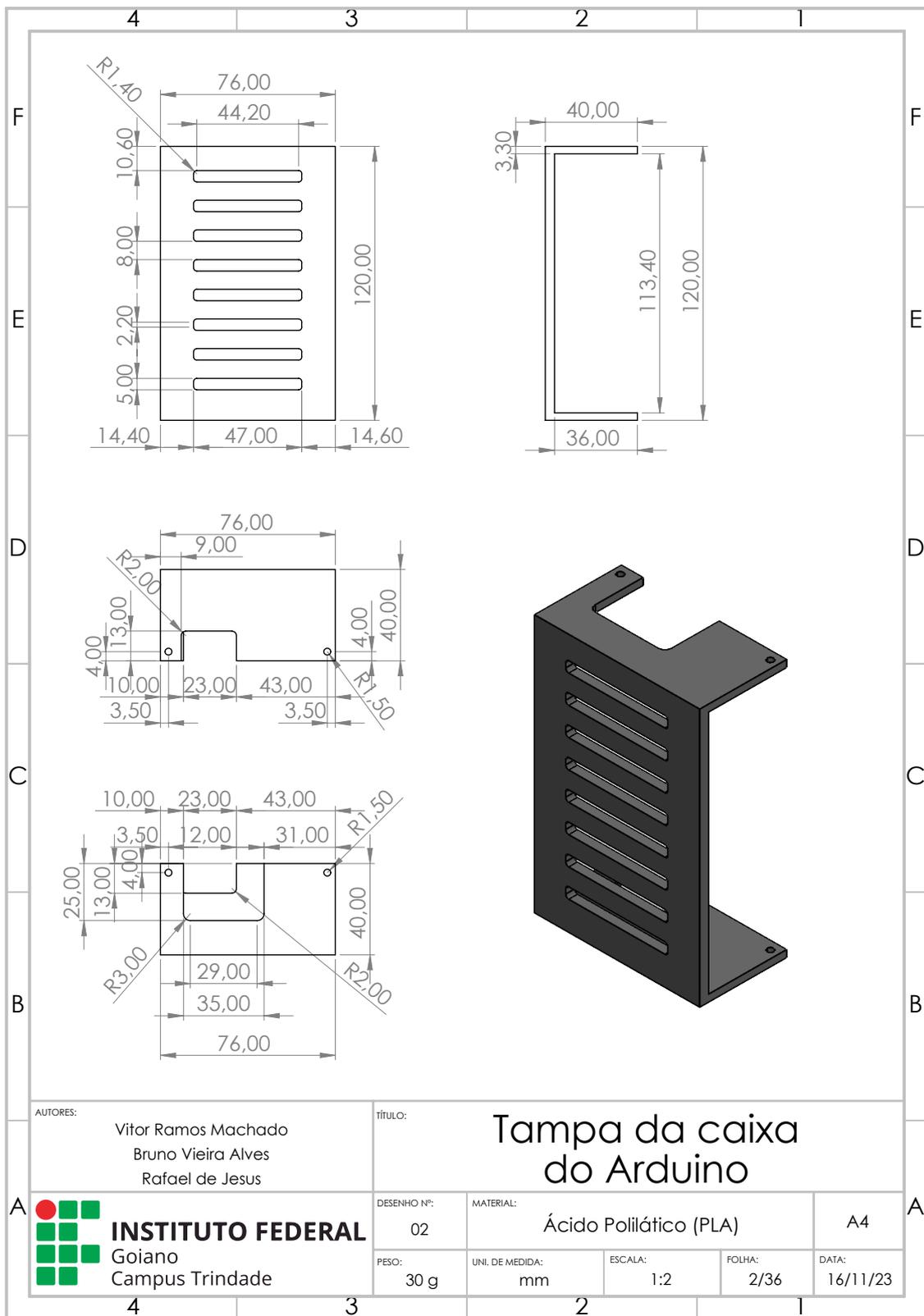


Figura A.2 - Prancheta da tampa da caixa do Arduino.

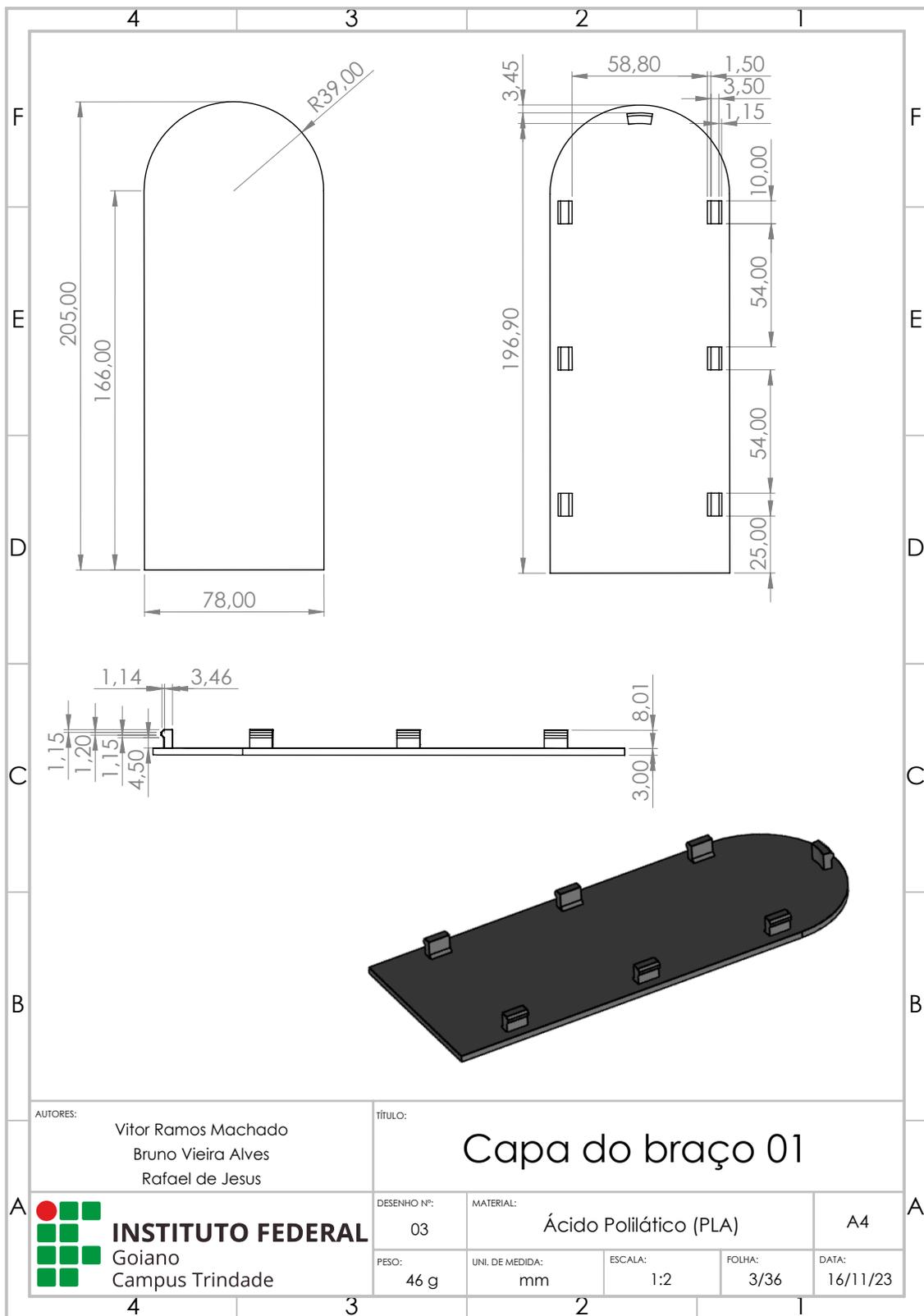


Figura A.3 - Prancheta da capa do braço 01.

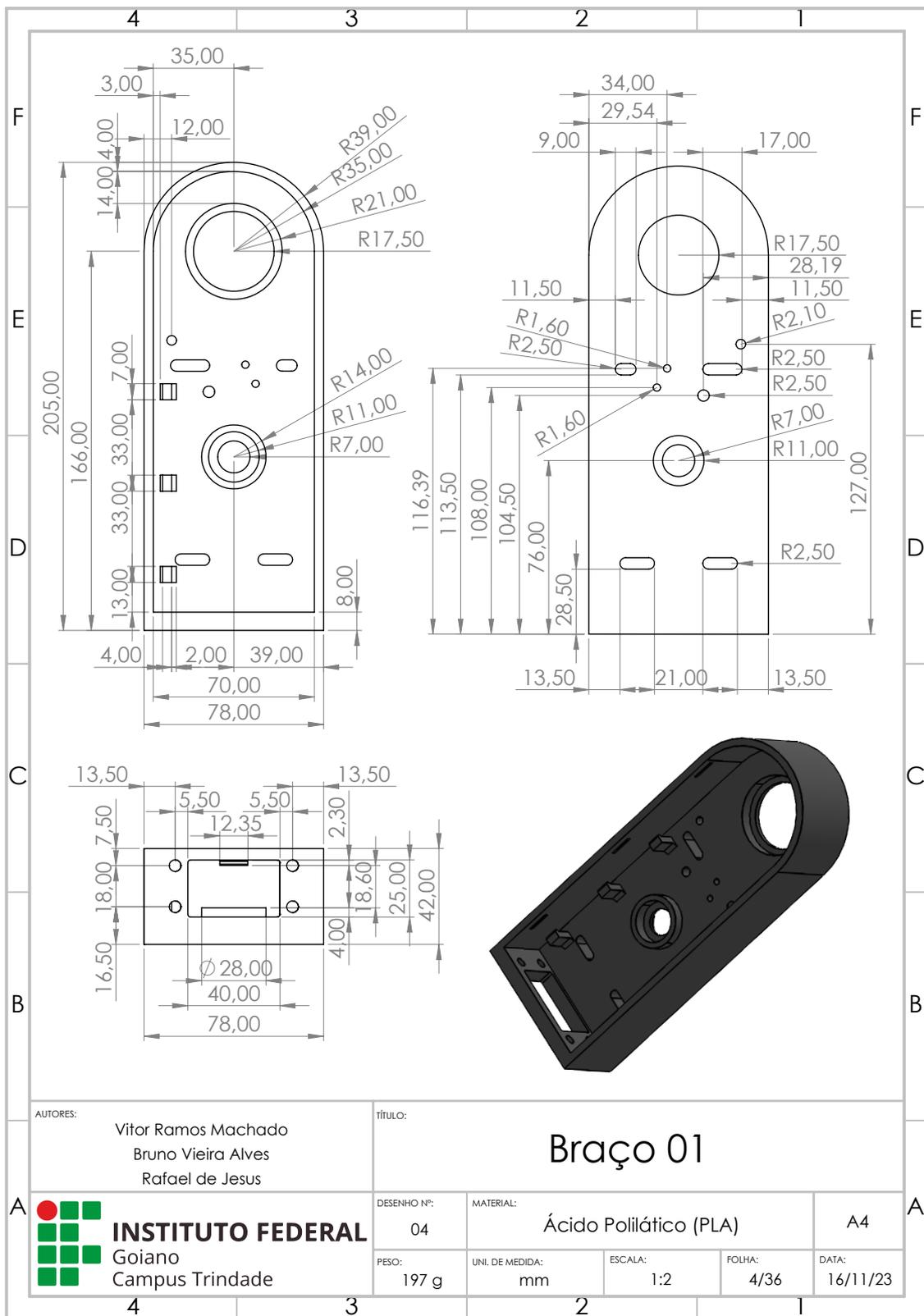


Figura A.4 - Prancheta do braço 01.

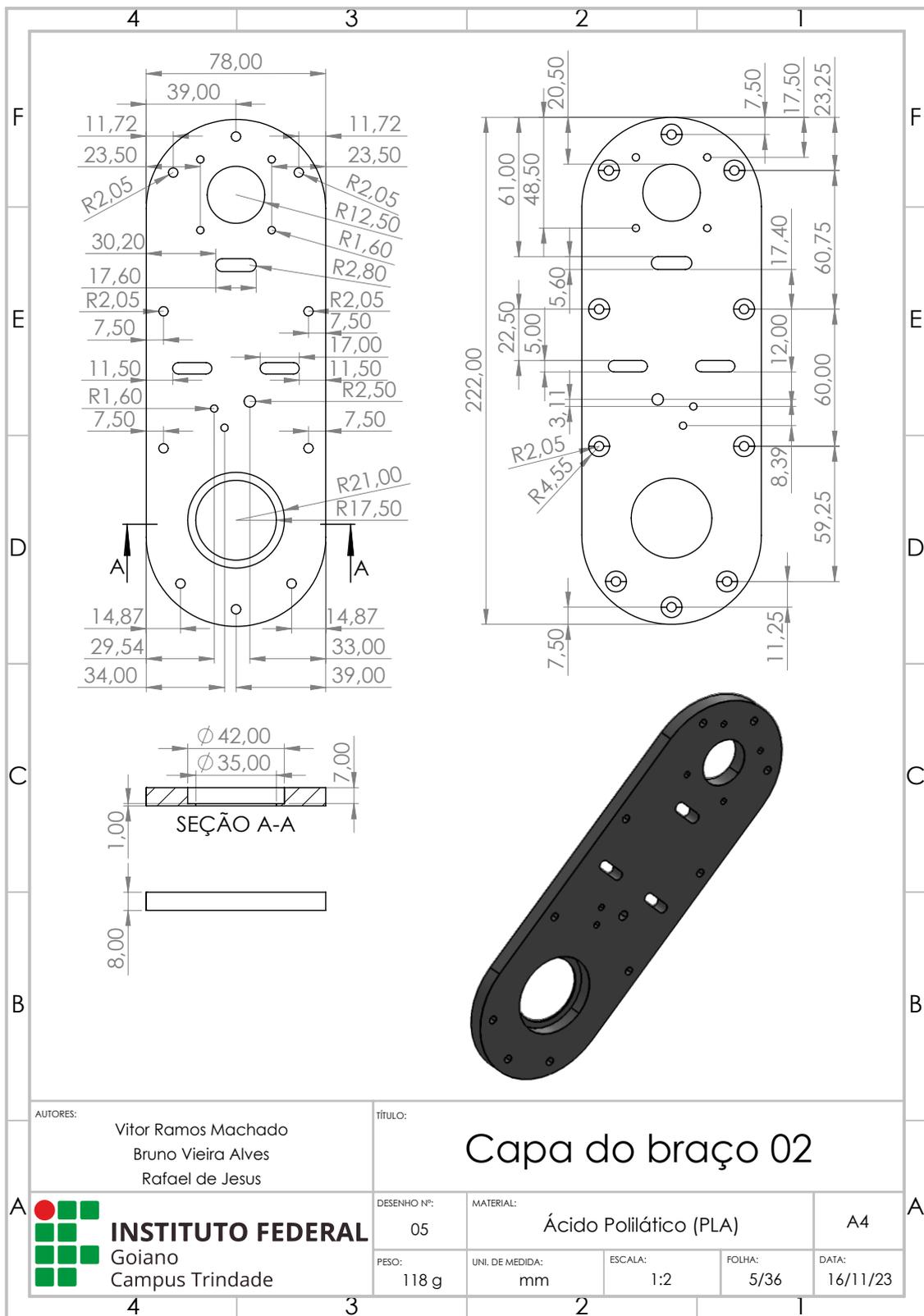


Figura A.5 - Prancheta da capa do braço 02.

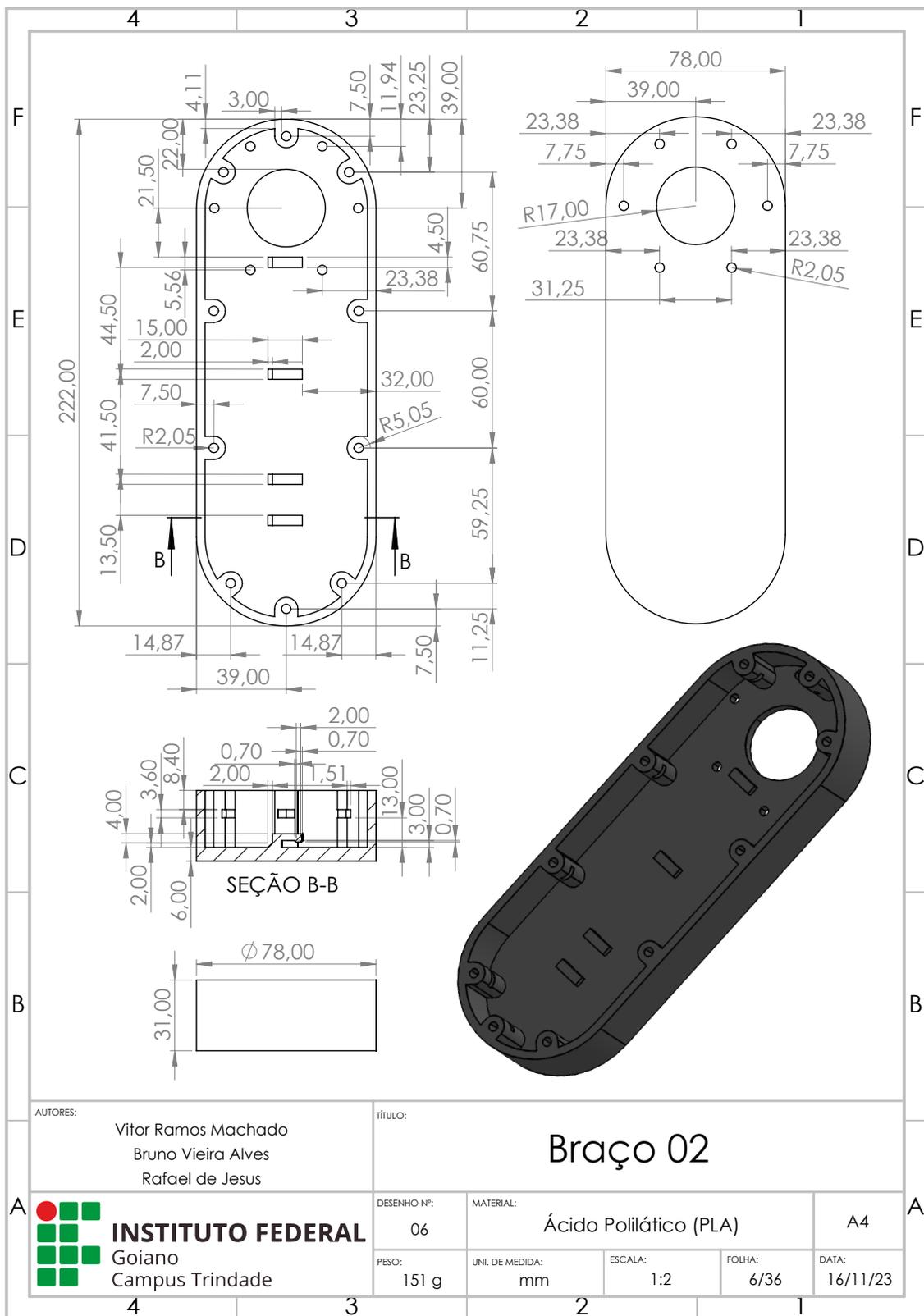


Figura A.6 - Prancheta do braço 02.

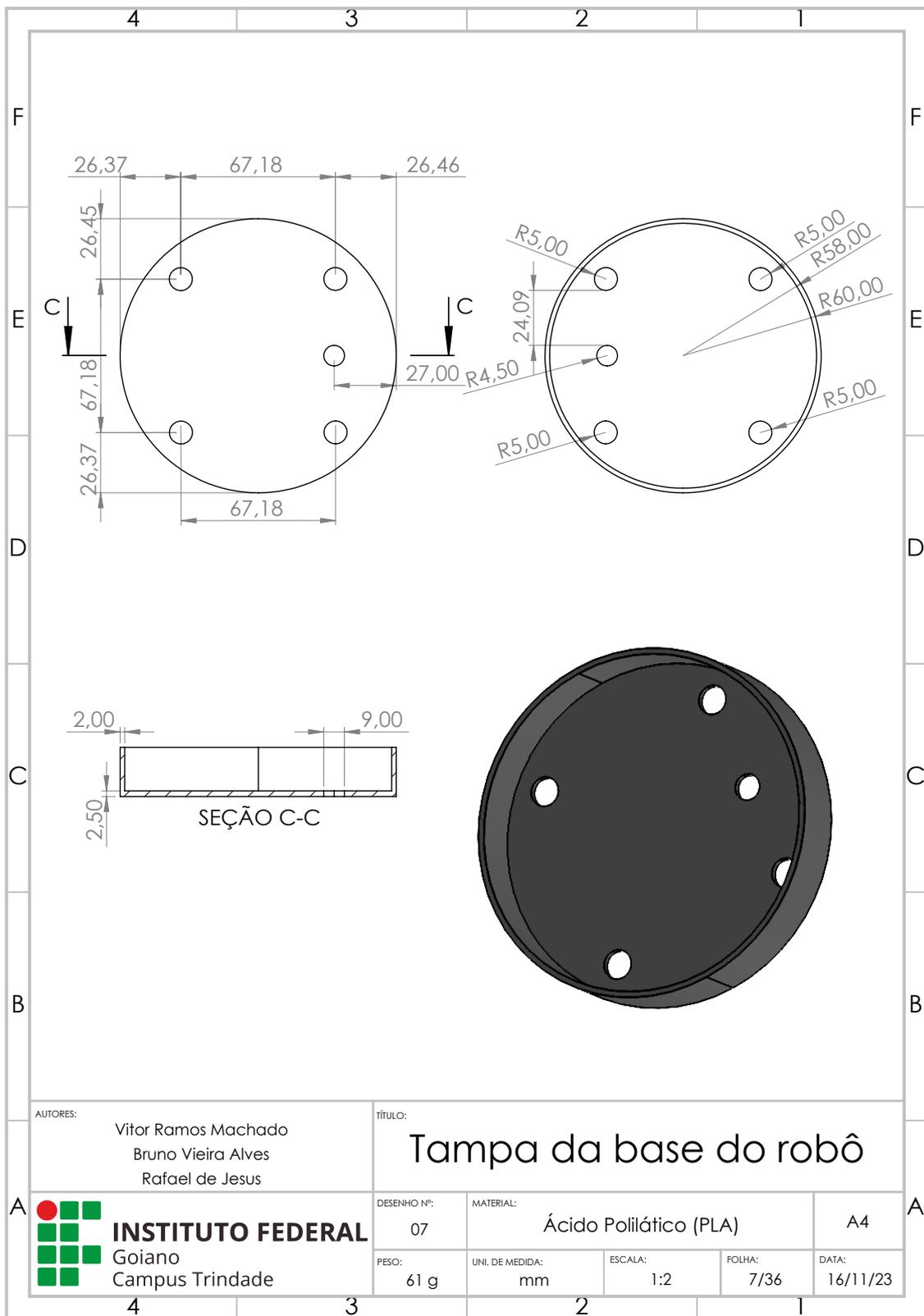


Figura A.7 - Prancheta da tampa da base do robô.

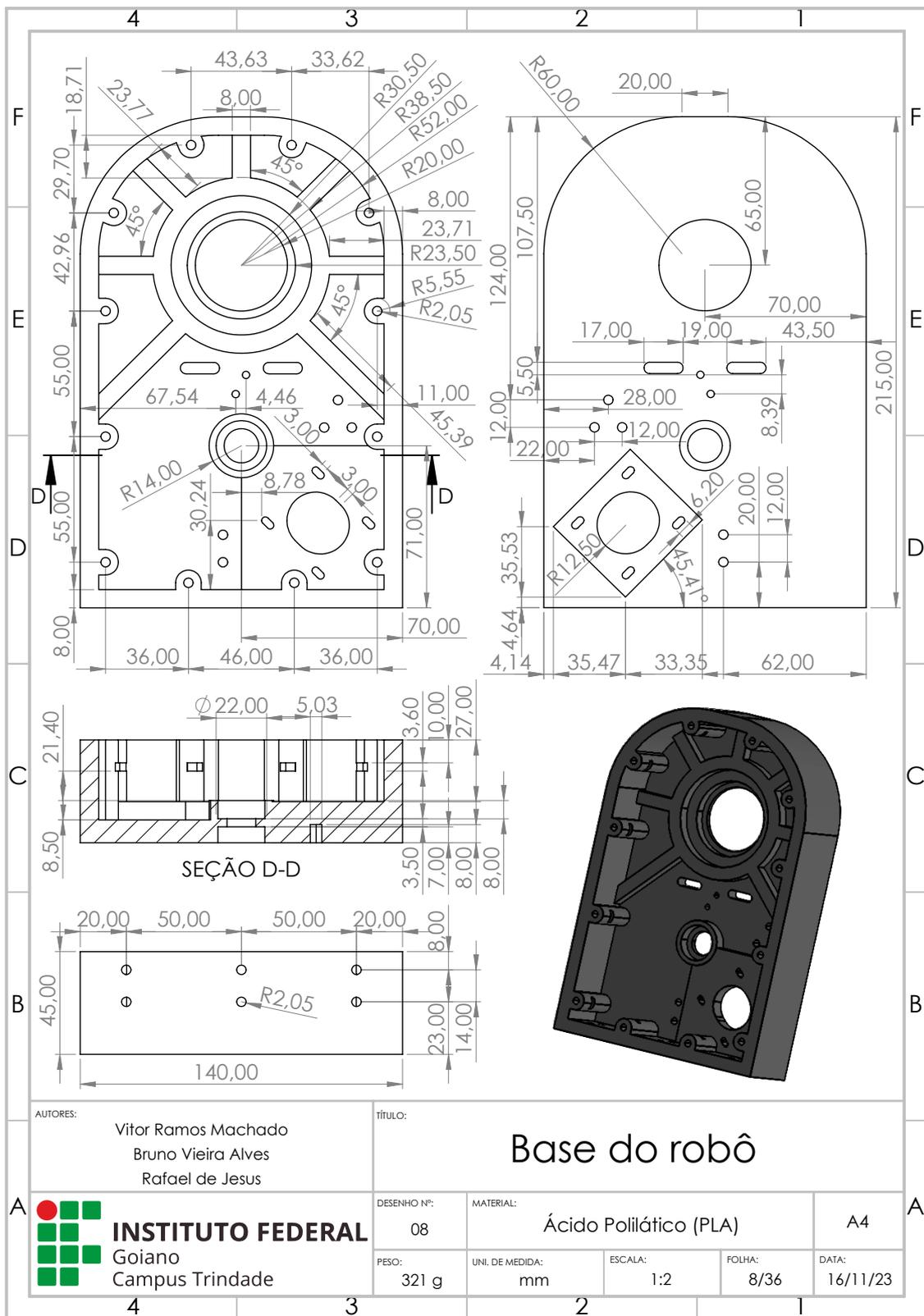


Figura A.8 - Prancheta da base do robô.

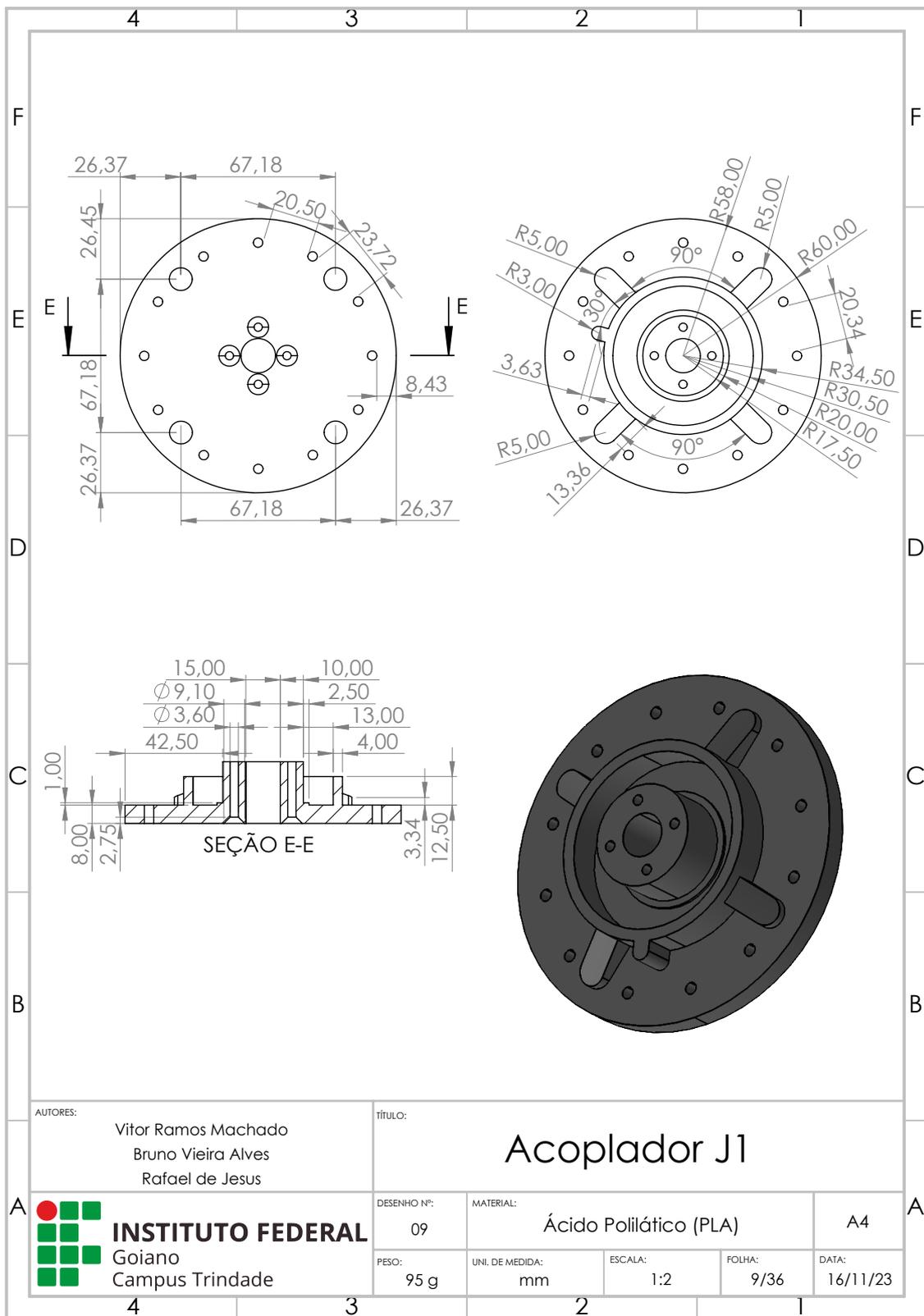


Figura A.9 - Prancheta do acoplador J1.

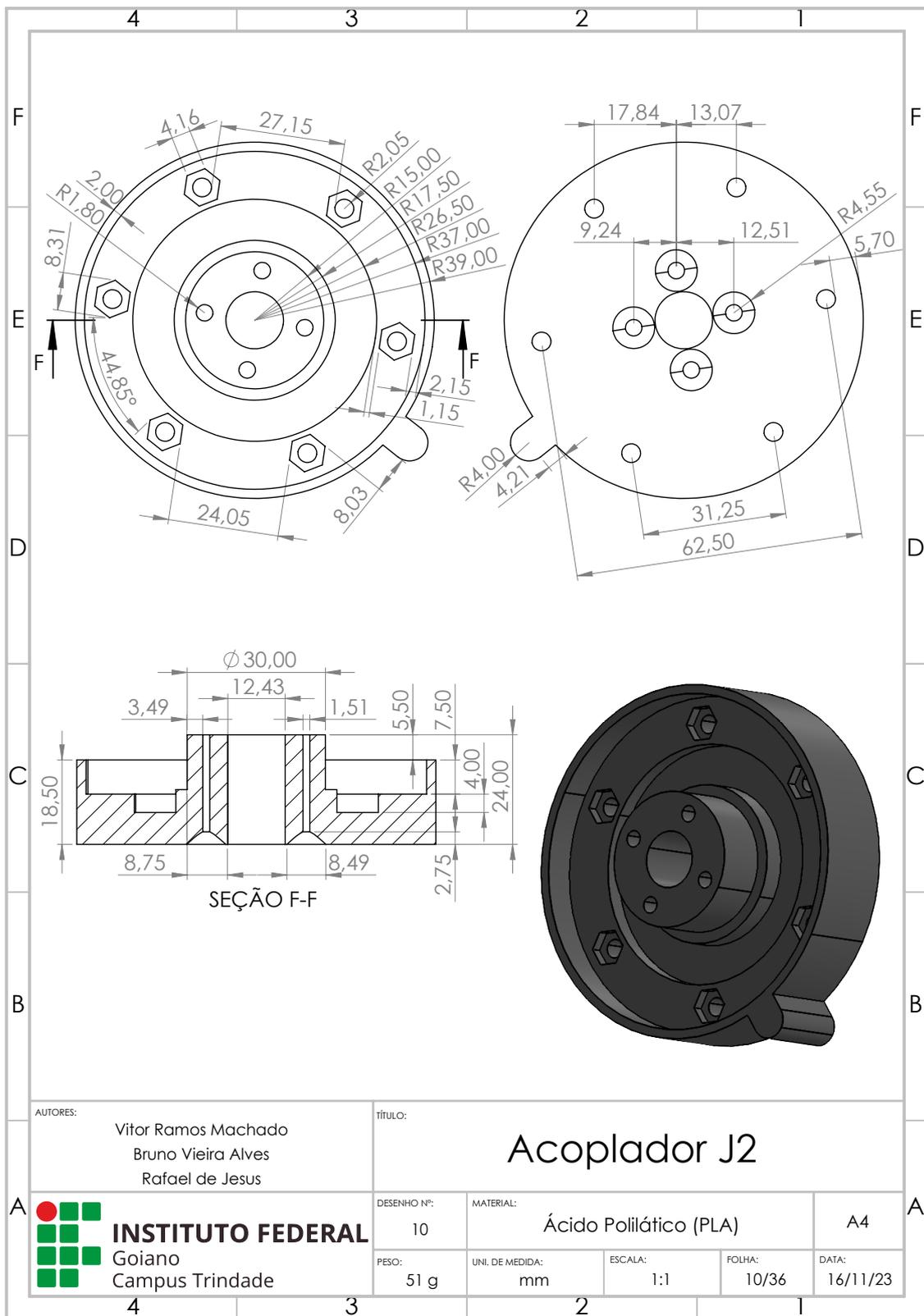


Figura A.10 - Prancheta do acoplador J2.

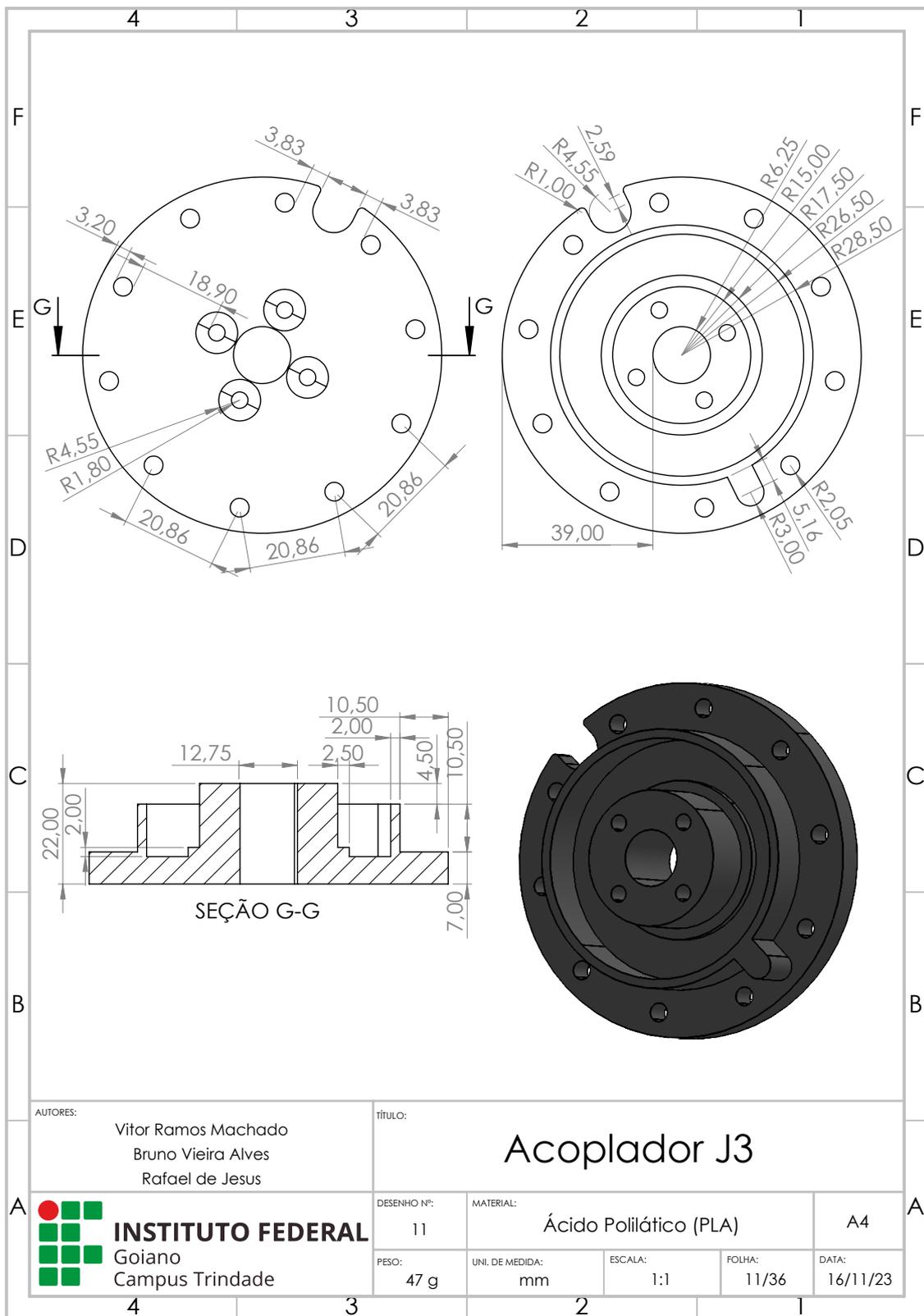


Figura A.11 - Prancheta do acoplador J3.

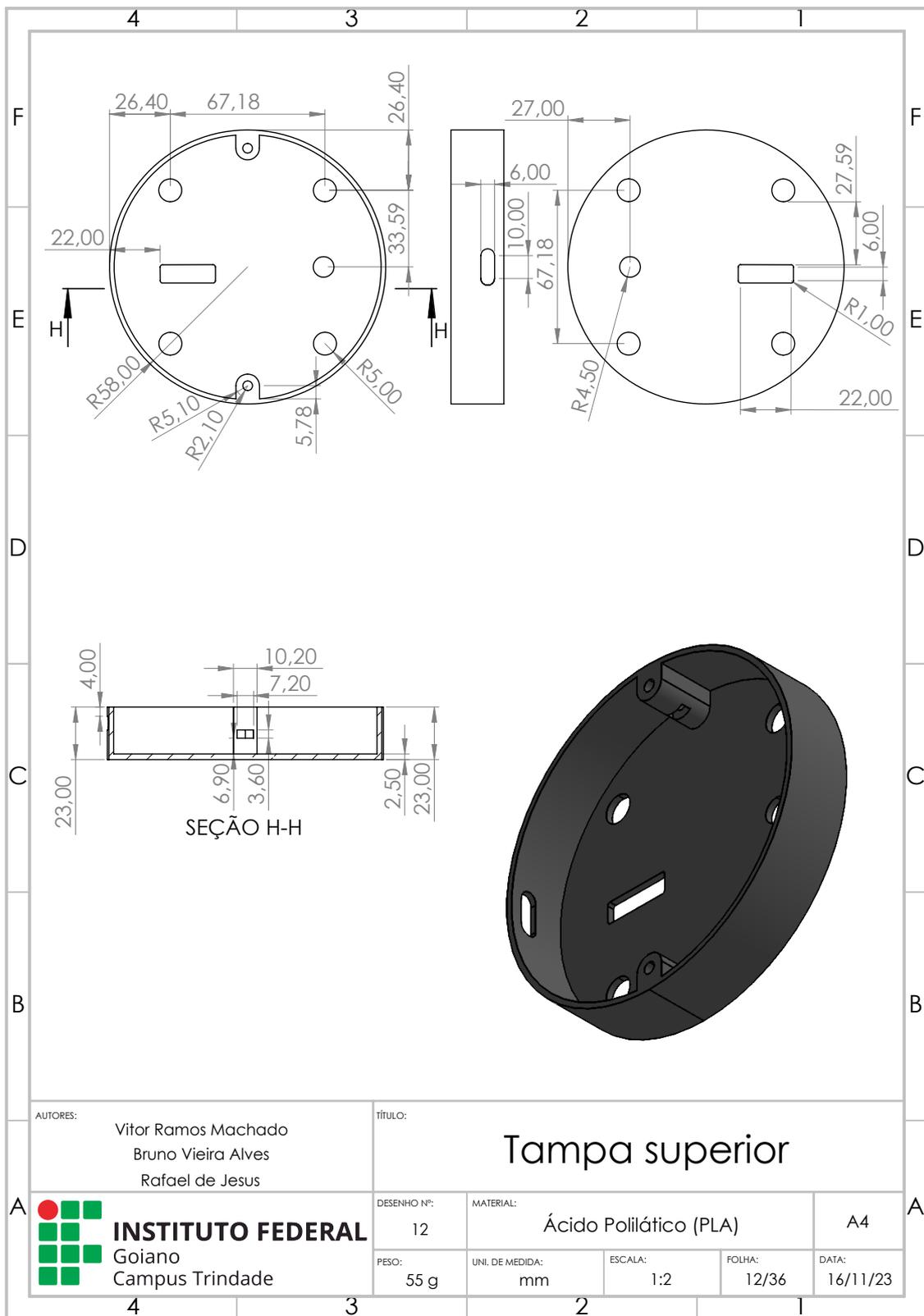


Figura A.12 - Prancheta da tampa superior.

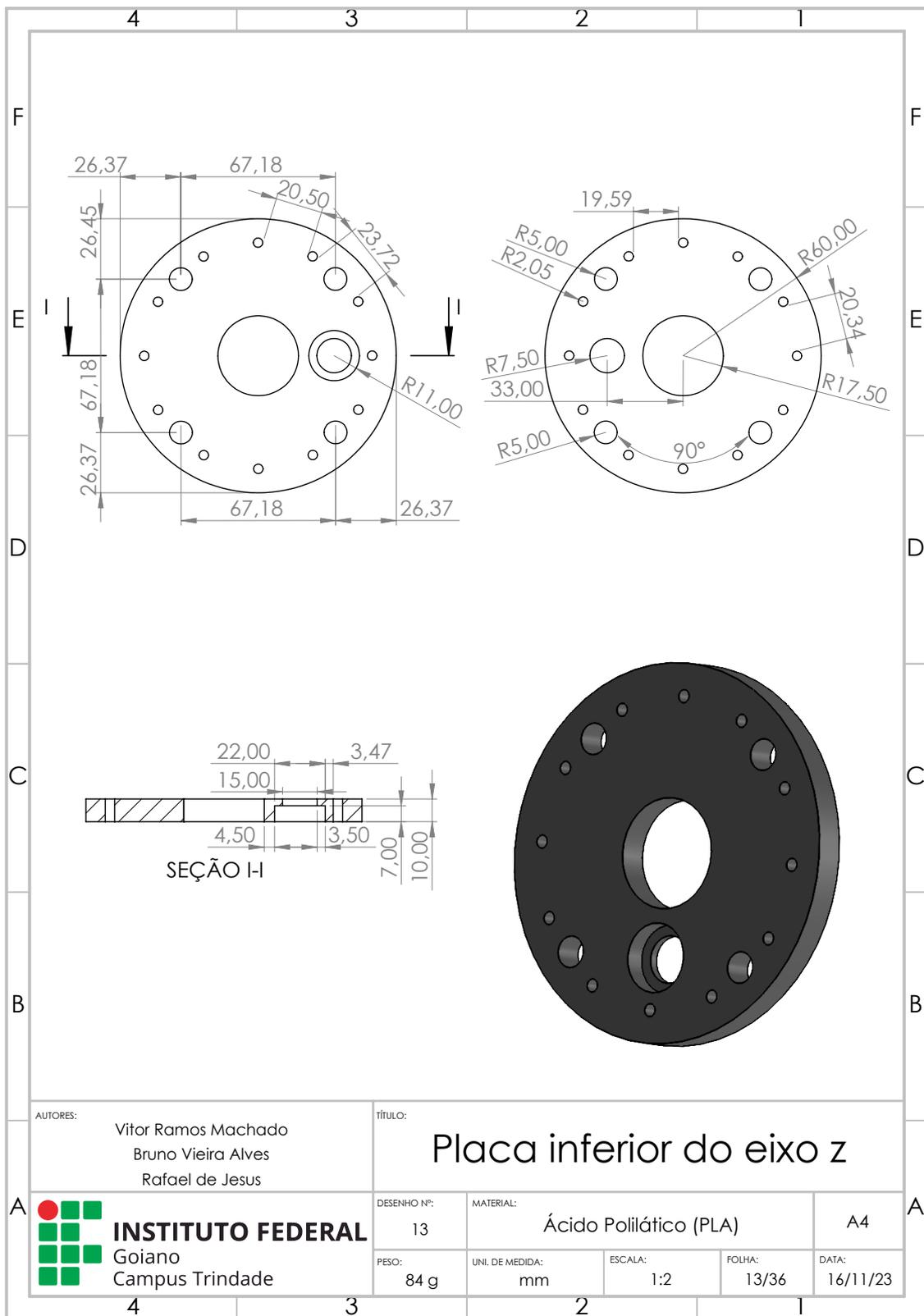


Figura A.13 - Prancheta da placa inferior do eixo z.

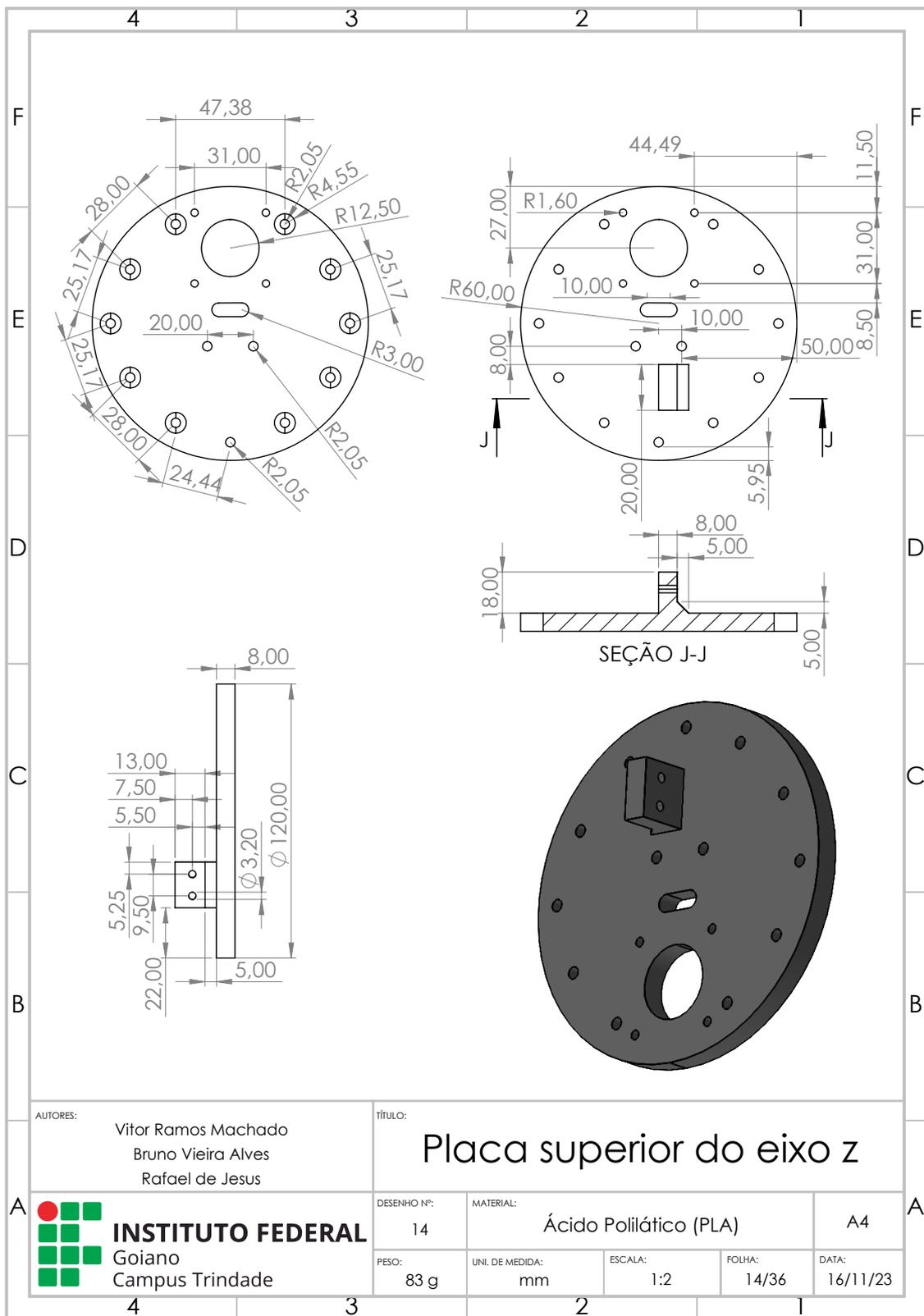


Figura A.14 - Prancheta da placa superior do eixo z.

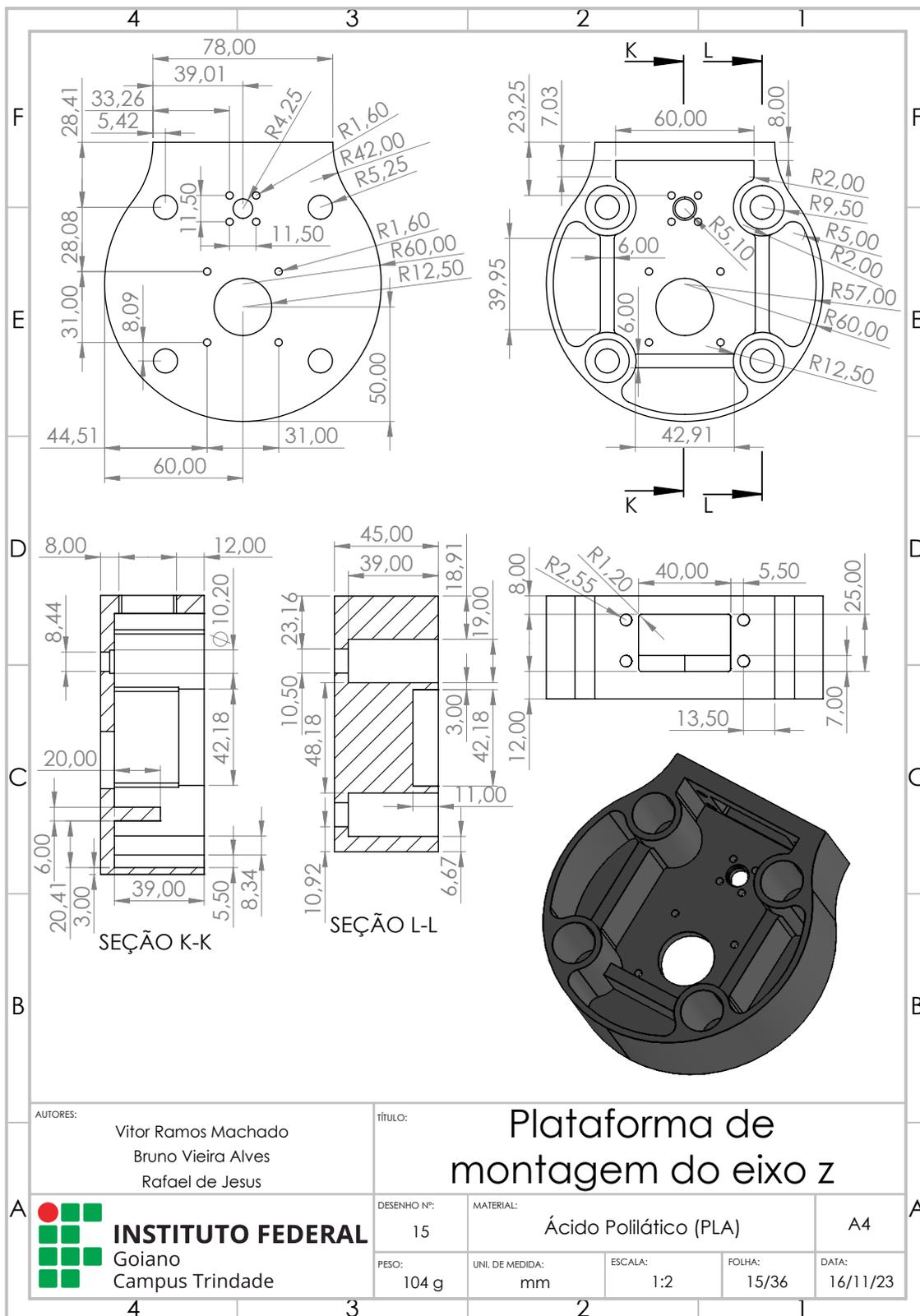


Figura A.15 - Prancheta da plataforma de montagem do eixo z.

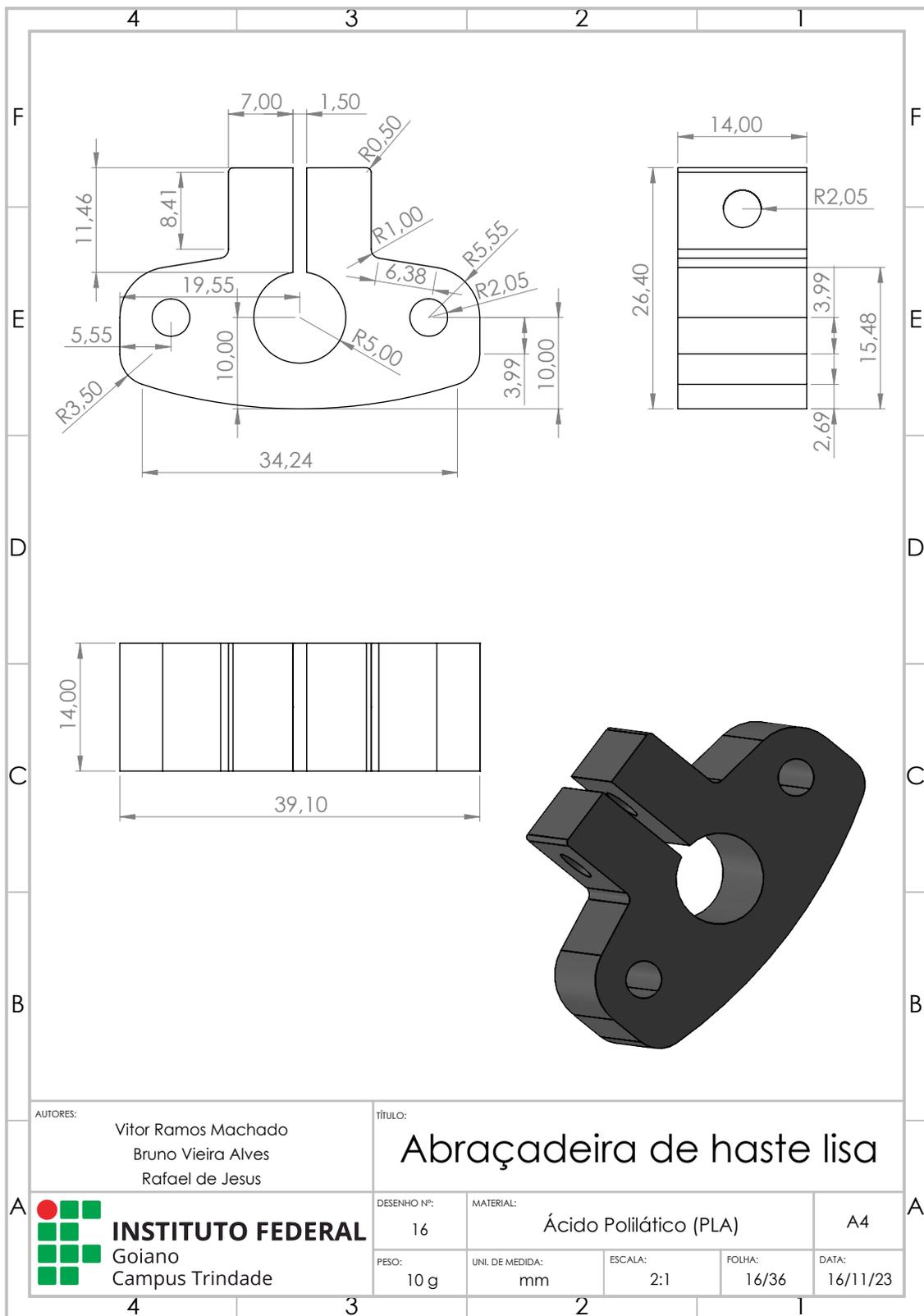


Figura A.16 - Prancheta da abraçadeira de haste lisa.

A.2 Polias e engrenagens do robô

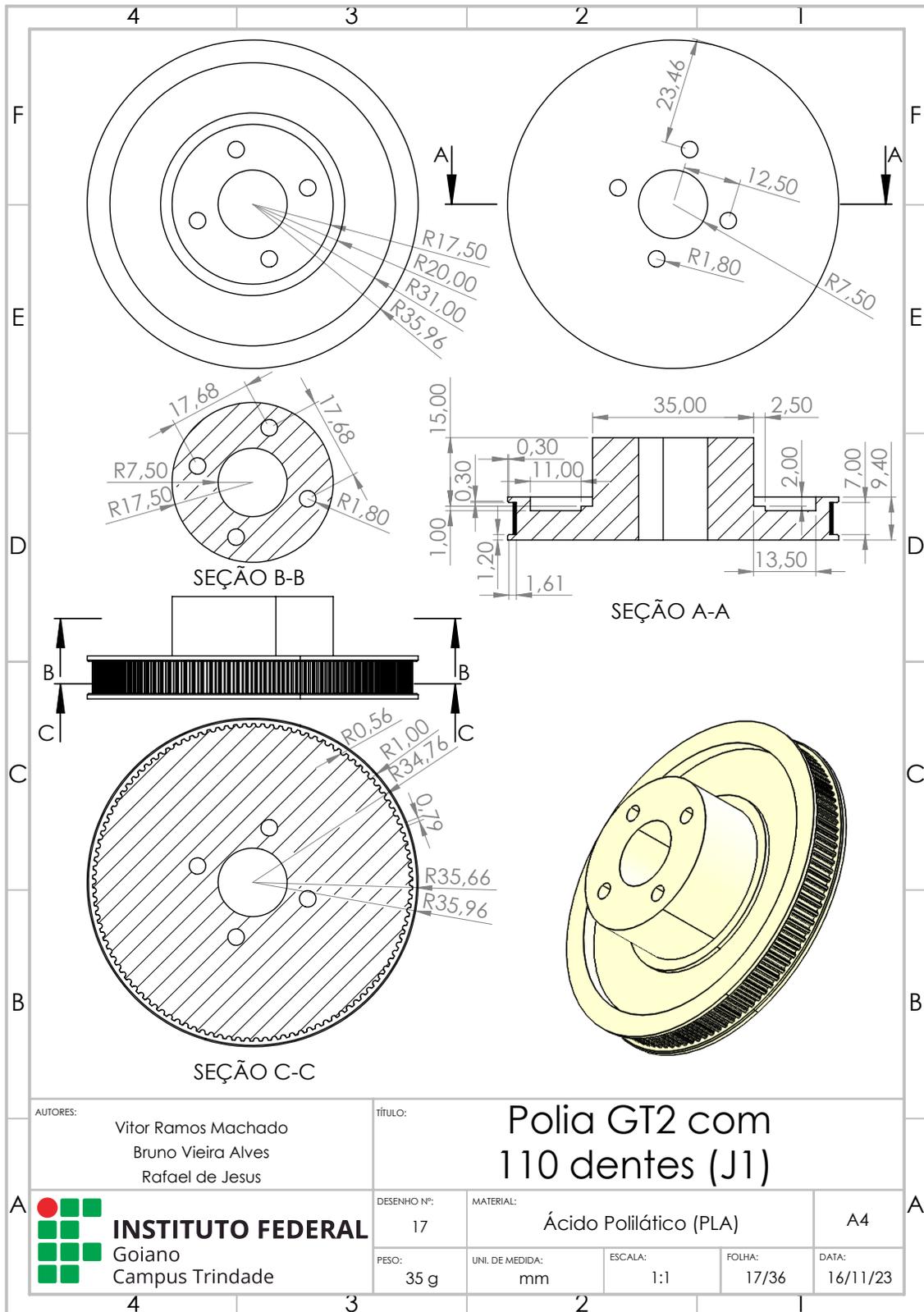


Figura A.17 - Prancheta da polia GT2 com 110 dentes.

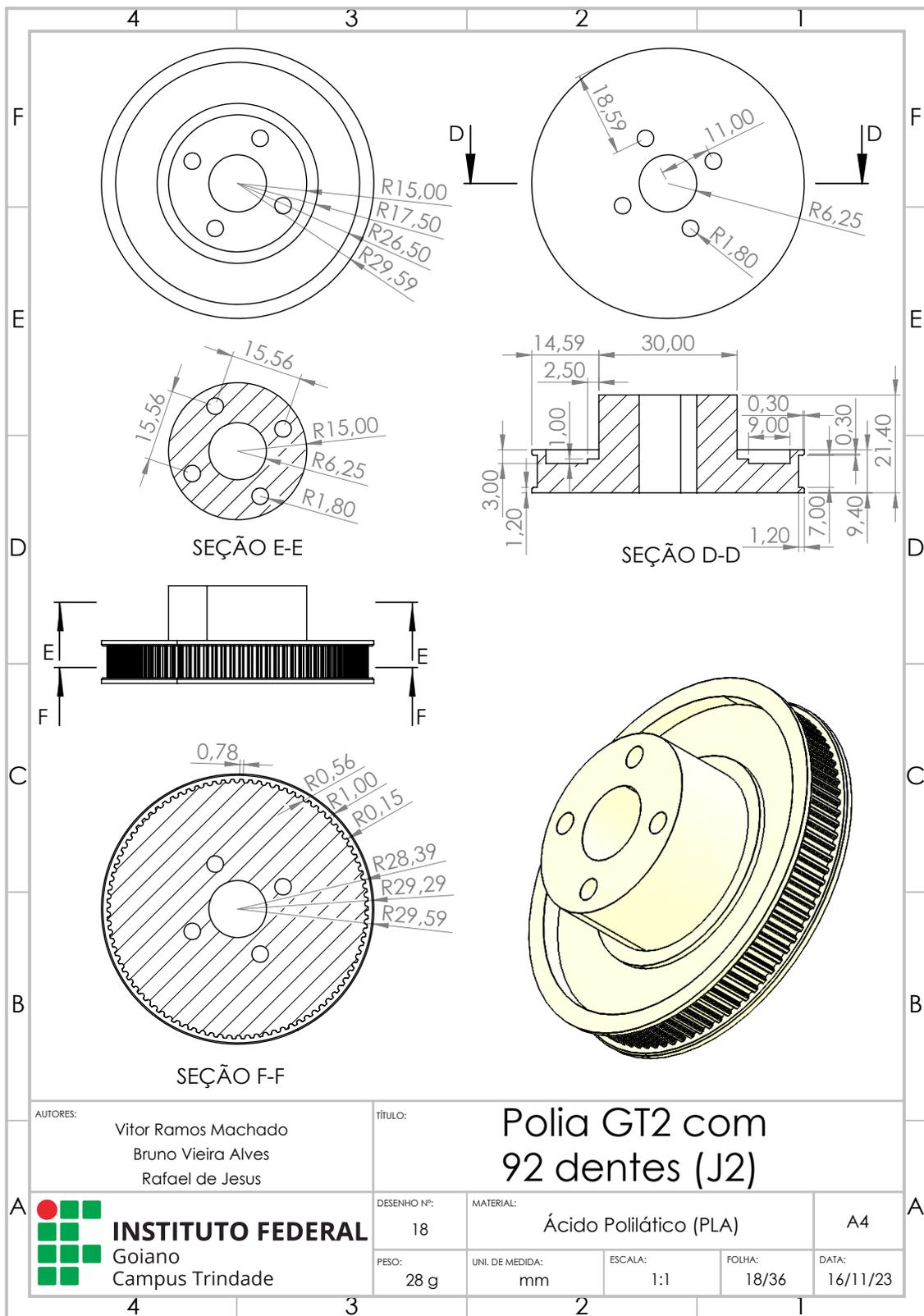


Figura A.18 - Prancheta da polia GT2 com 92 dentes.

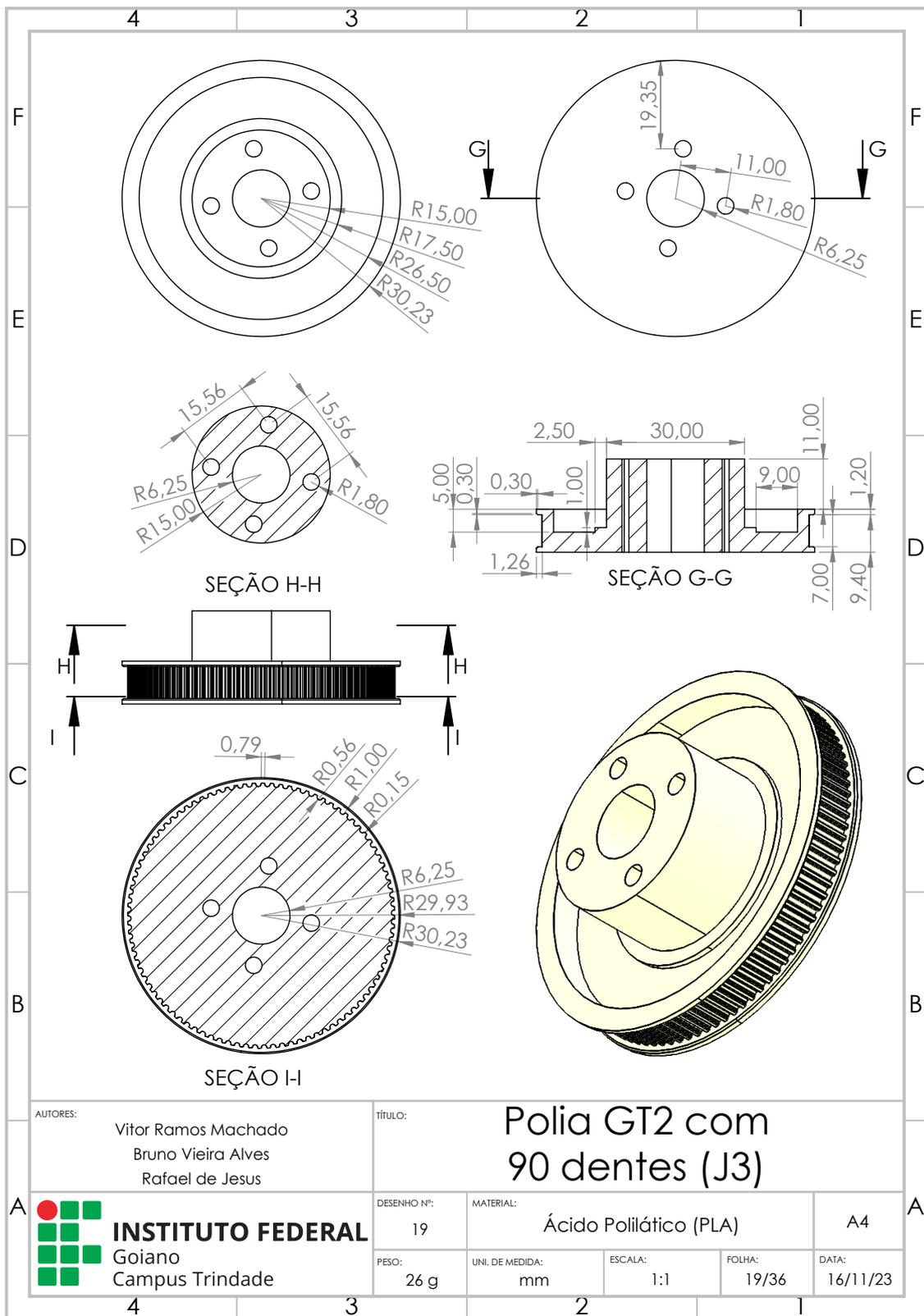


Figura A.19 - Prancheta da polia GT2 com 90 dentes.

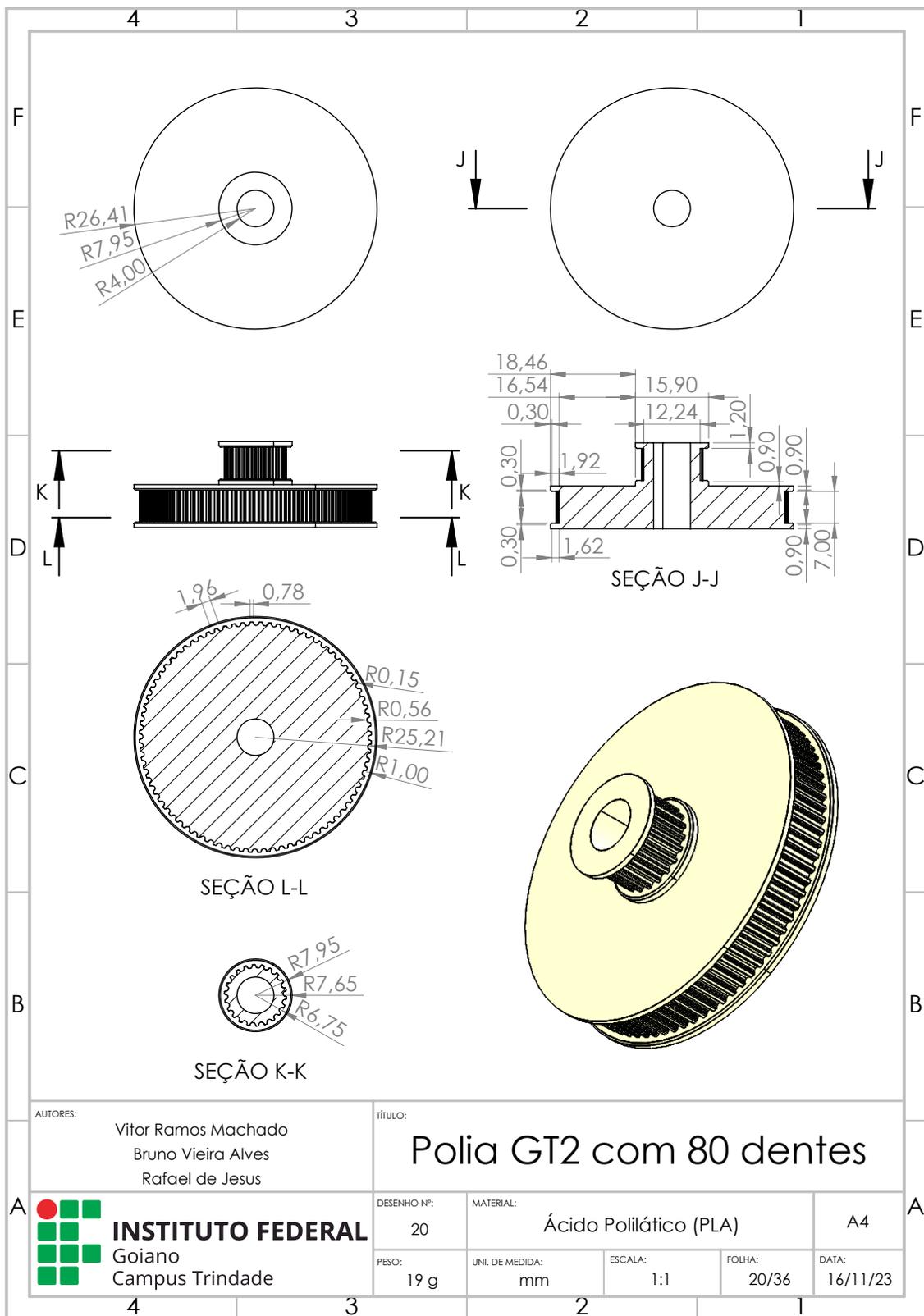


Figura A.20 - Prancheta da polia GT2 com 80 dentes.

A.3 Efetuador final do robô

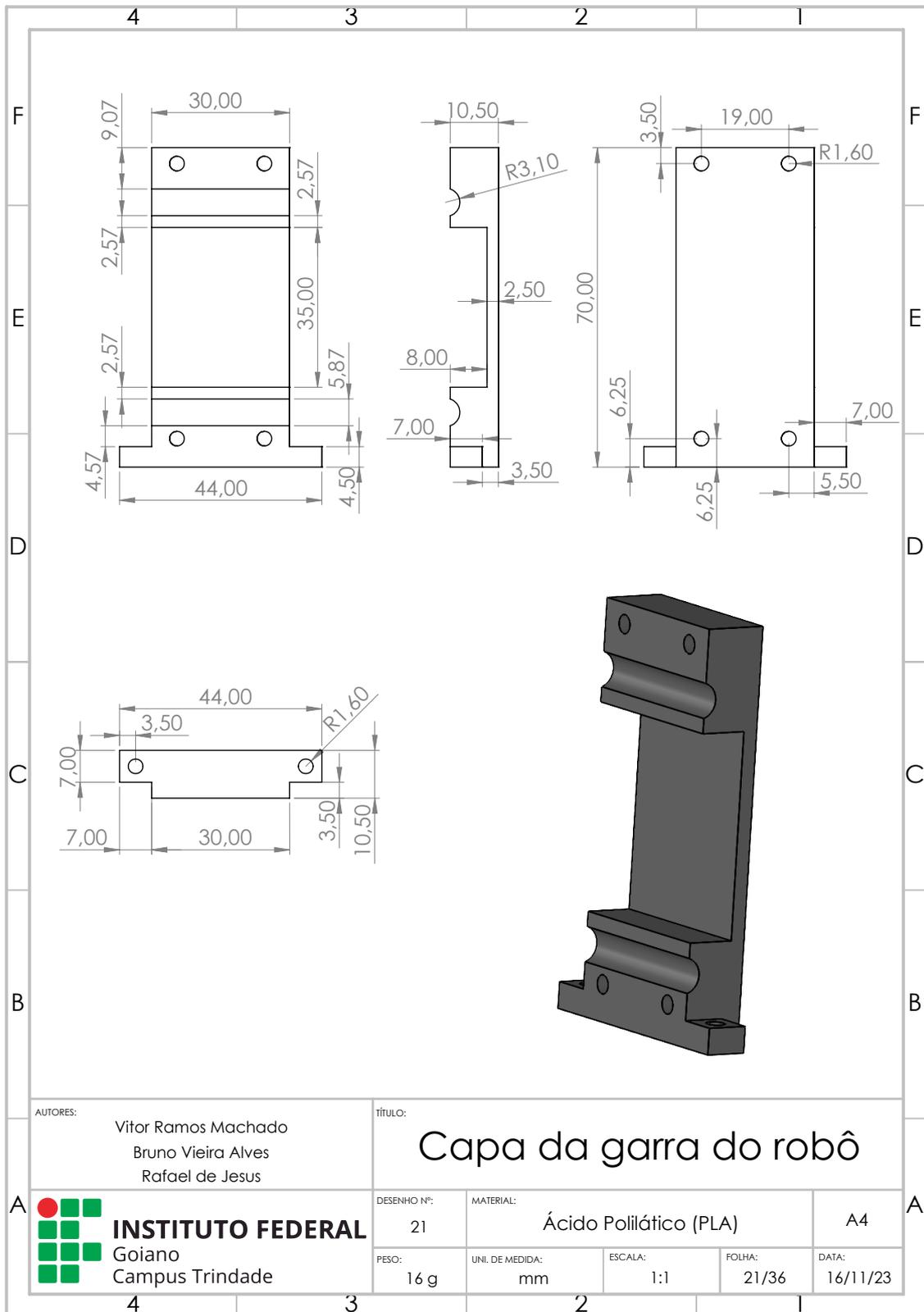


Figura A.21 - Prancheta da capa da garra do robô.

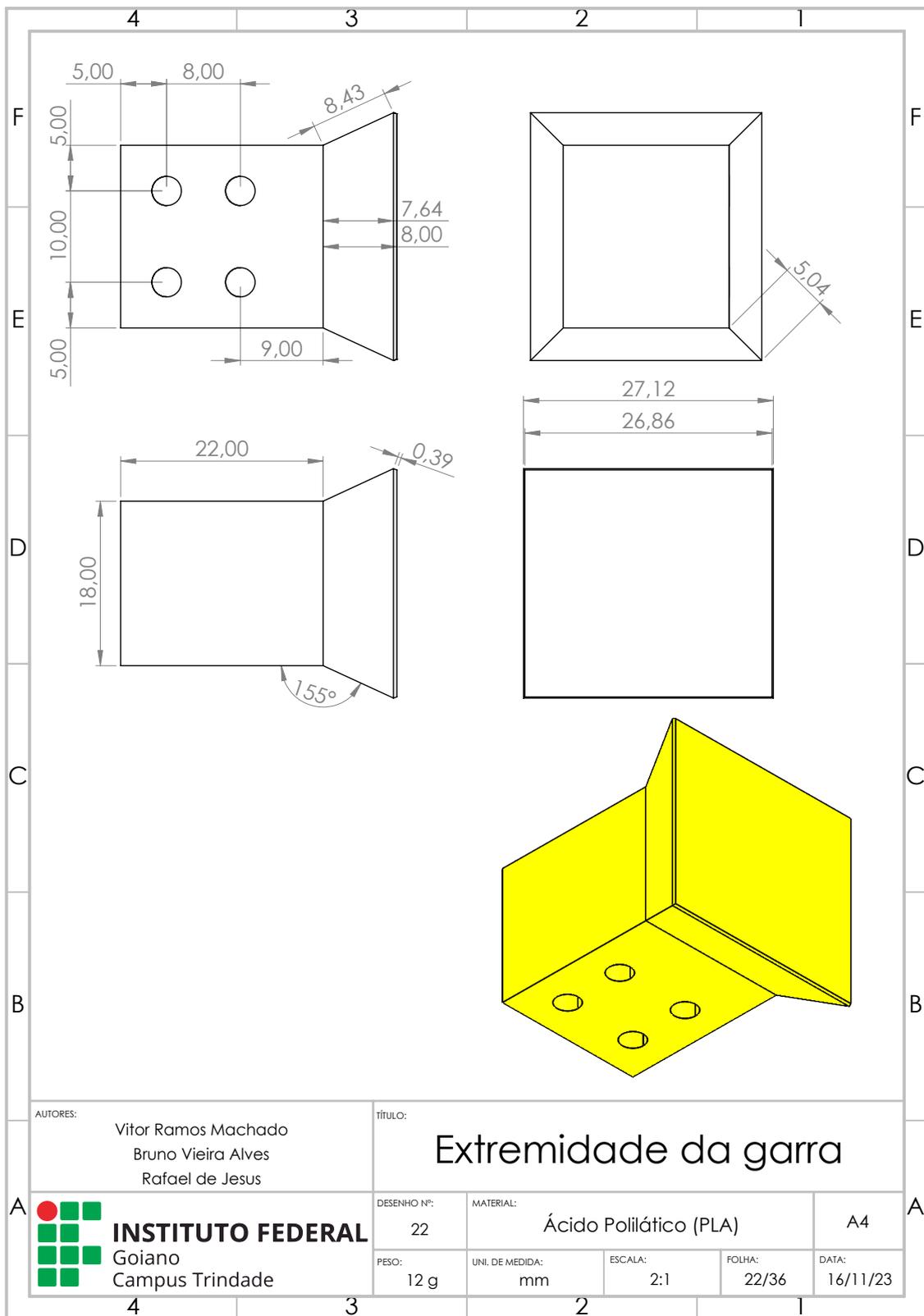


Figura A.22 - Prancheta da extremidade da garra.

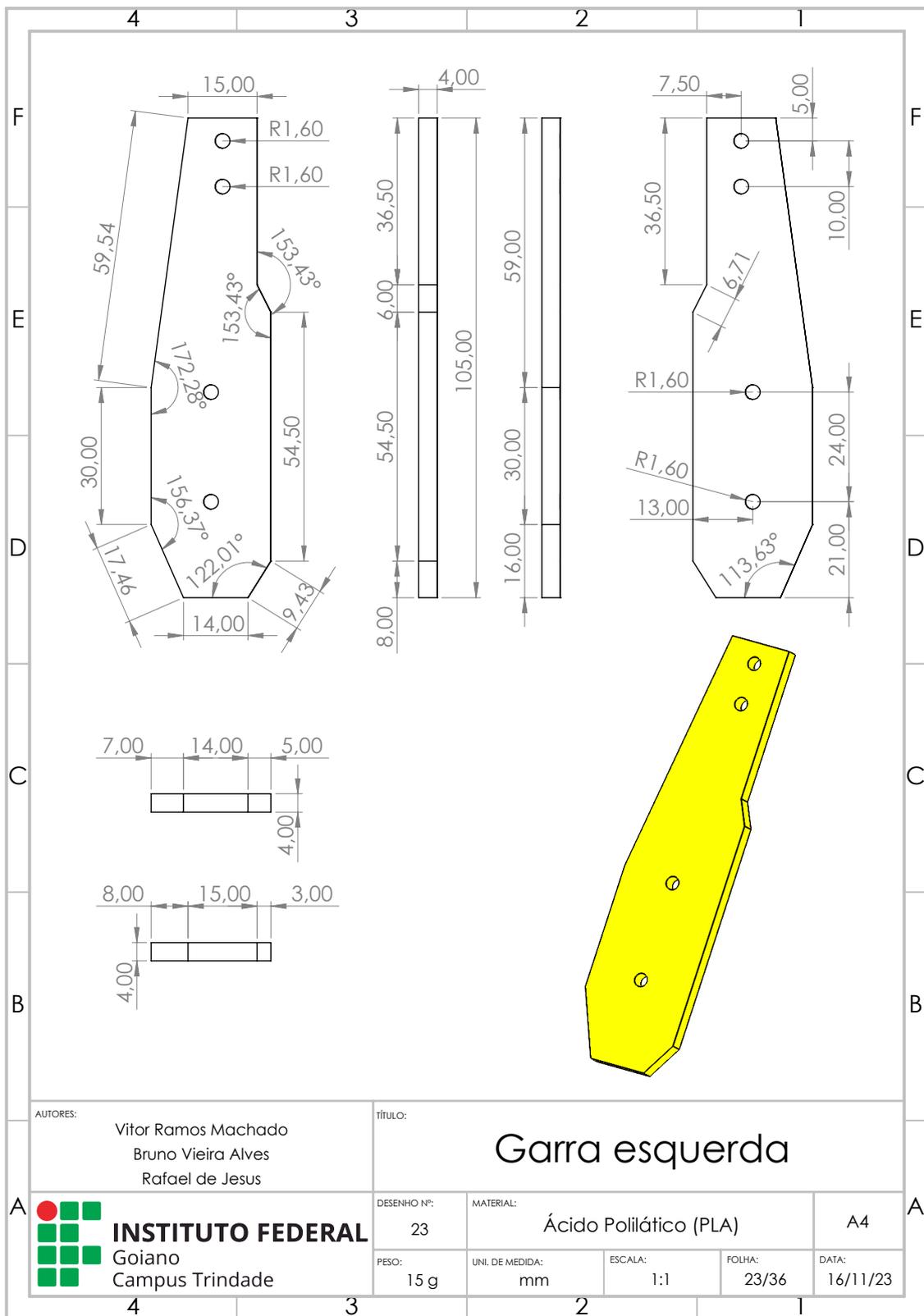


Figura A.23 - Prancheta da garra esquerda.

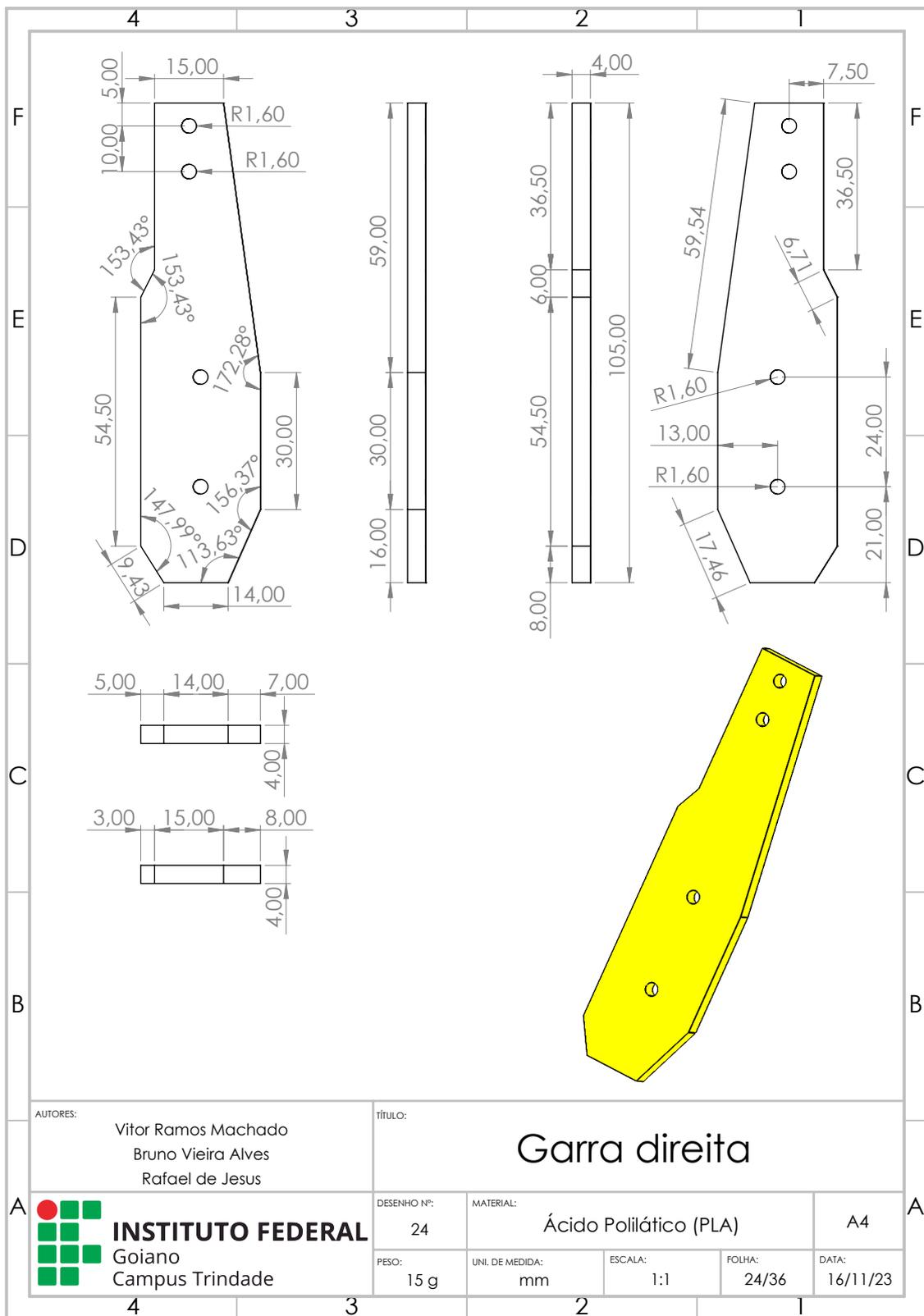


Figura A.24 - Prancheta da garra direita.

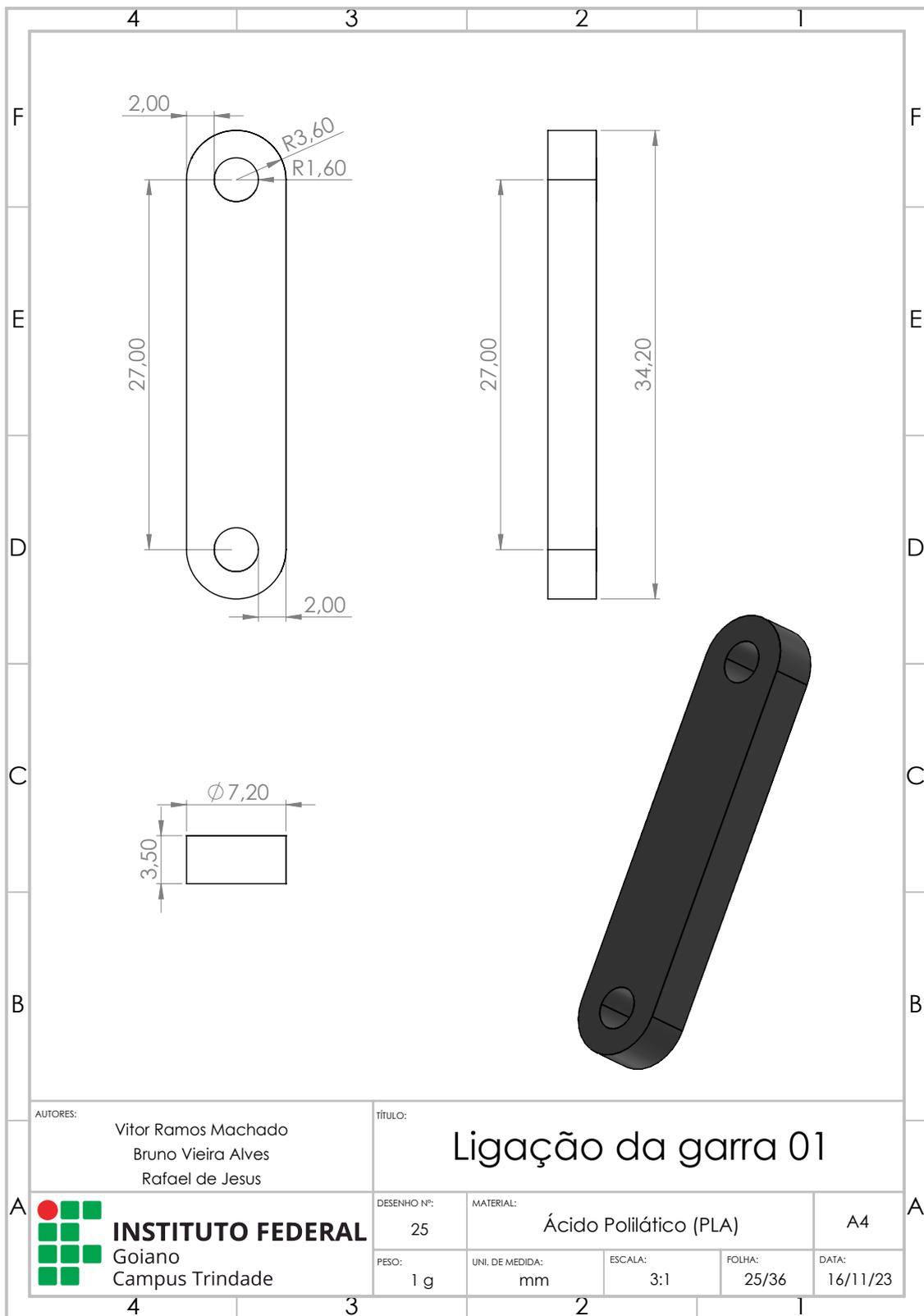


Figura A.25 - Prancheta da ligação da garra 01.

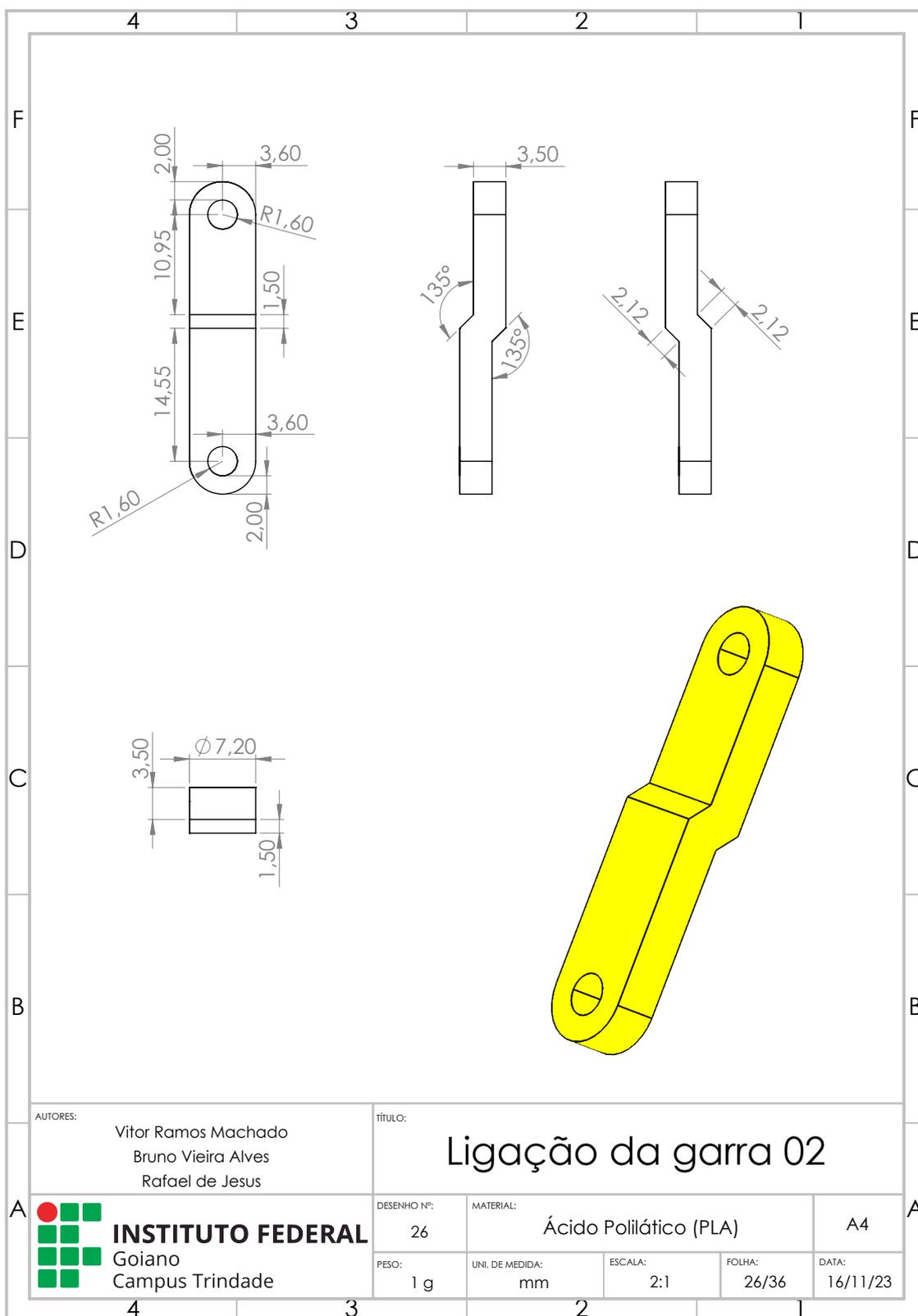


Figura A.26 - Prancheta da ligação da garra 02.

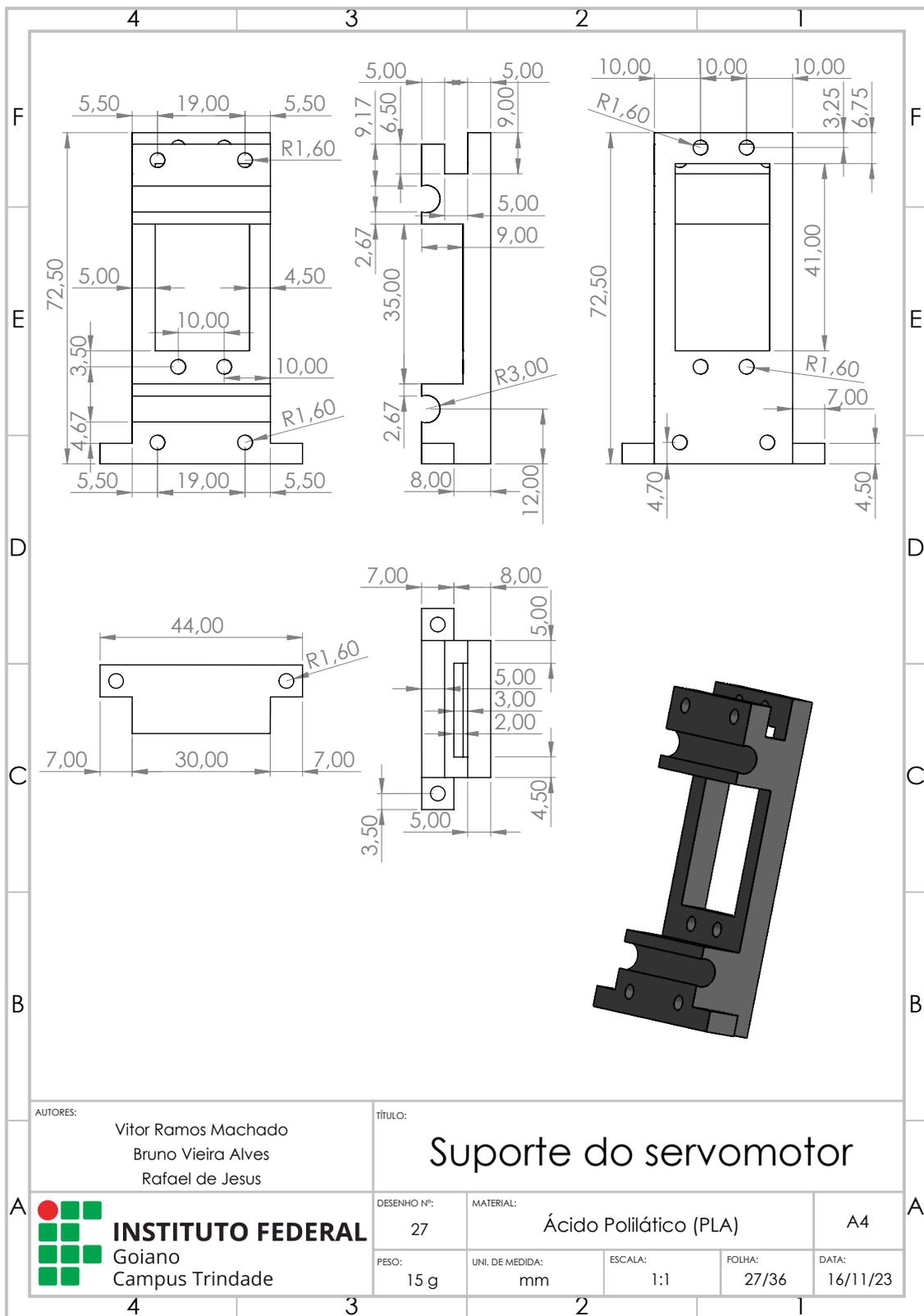


Figura A.27 - Prancheta do suporte do servomotor.

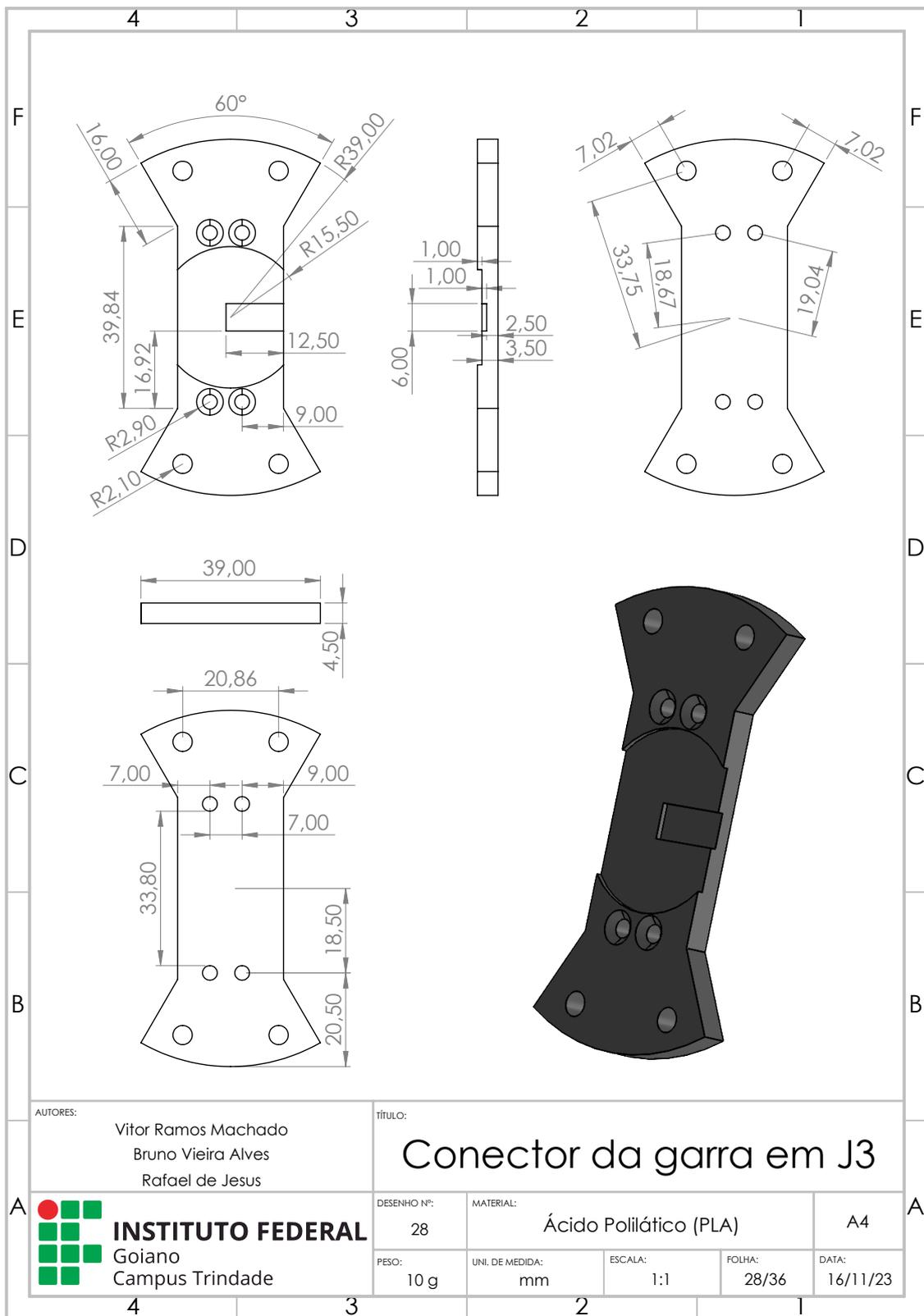


Figura A.28 - Prancheta do conector da garra em J3.

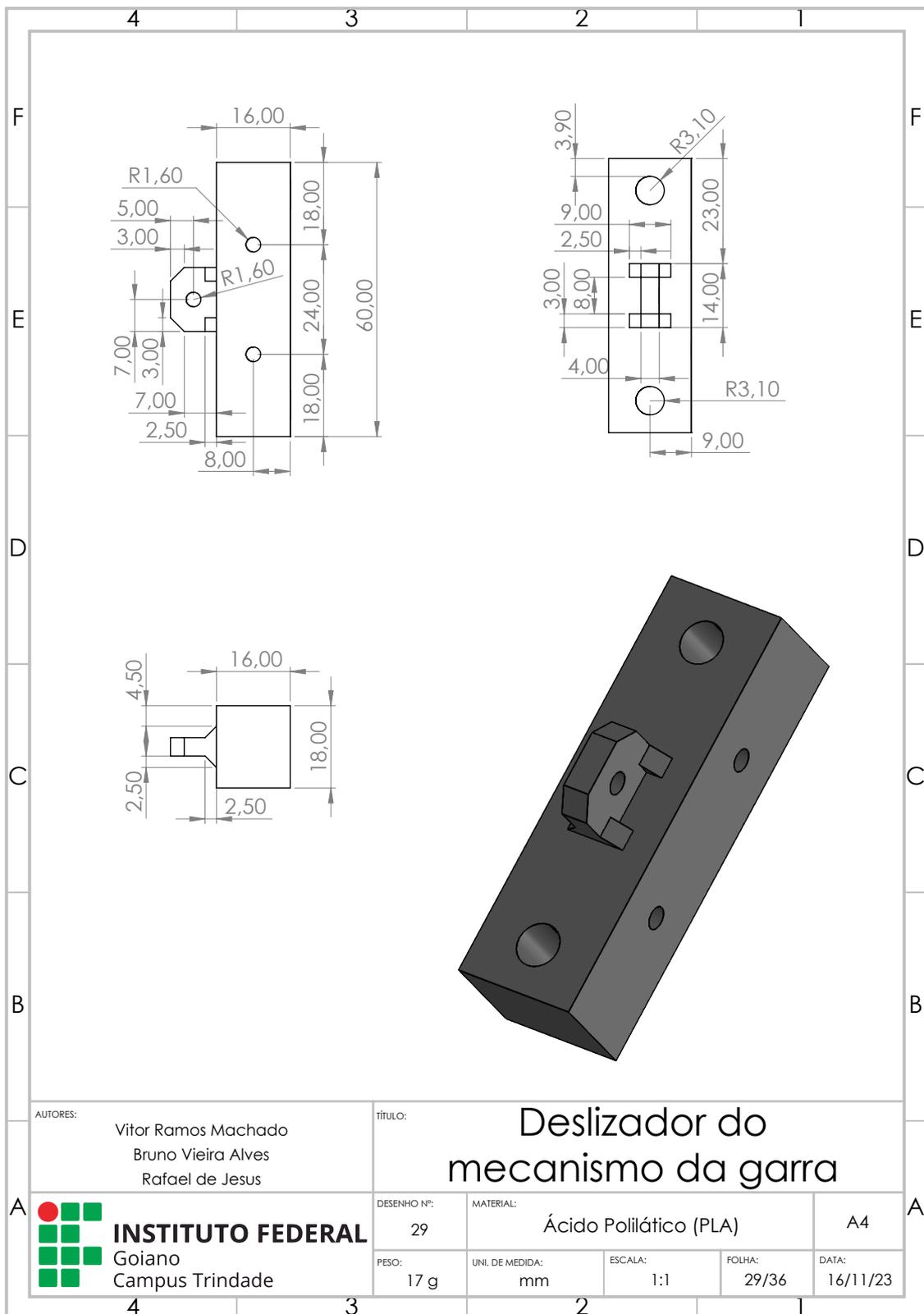


Figura A.29 - Prancheta do deslizador do mecanismo da garra.

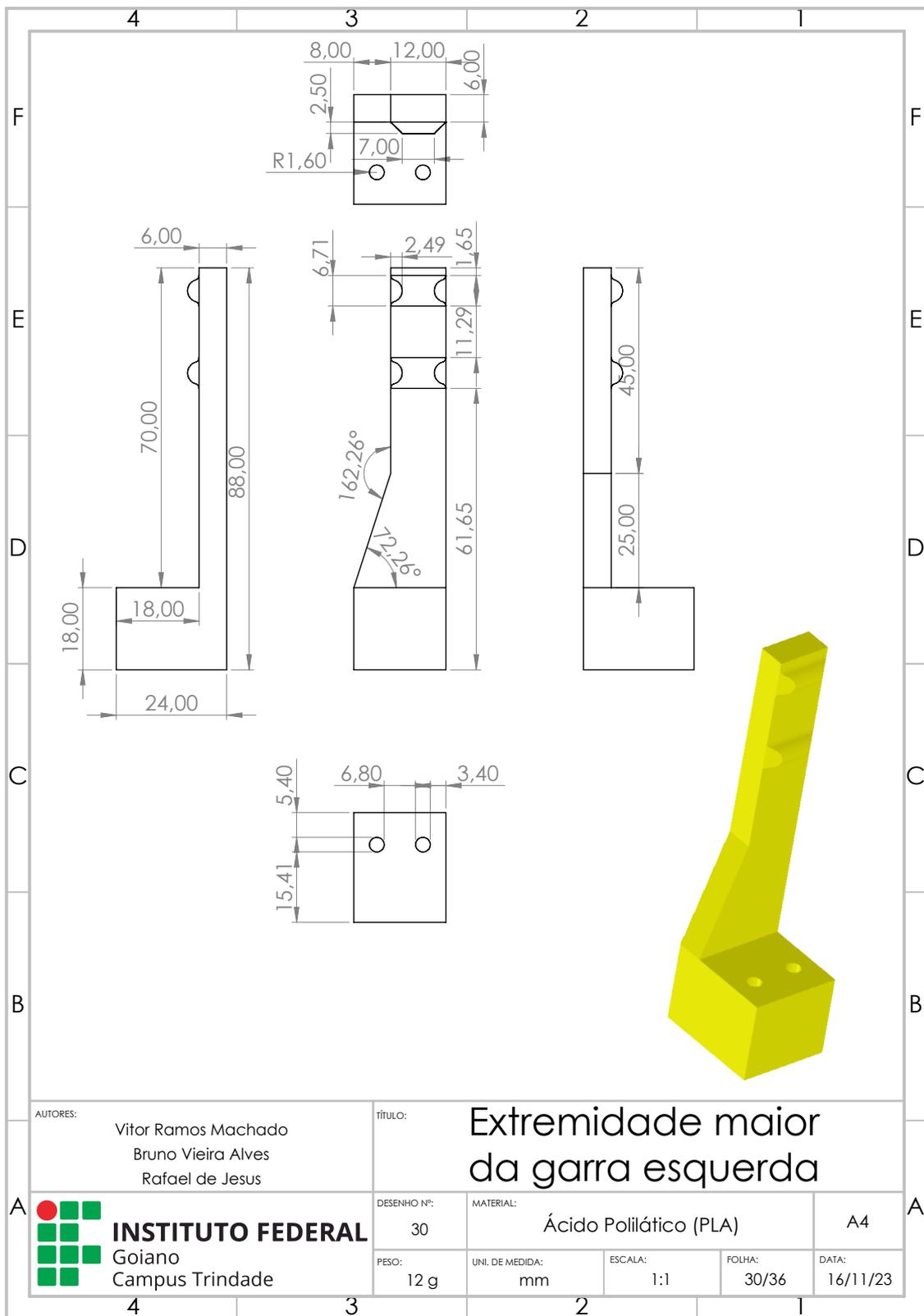


Figura A.30 - Prancheta da extremidade maior da garra esquerda.

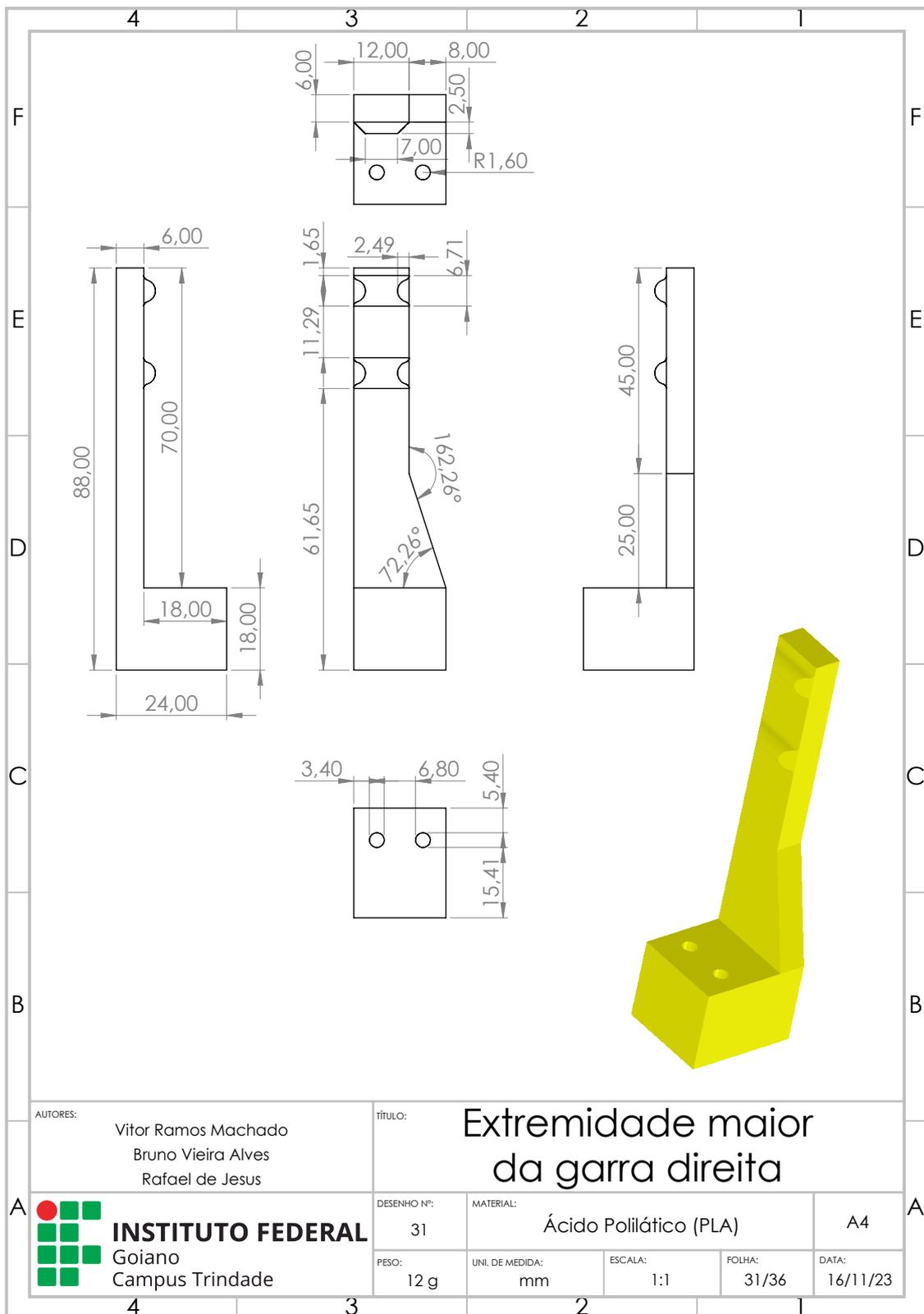


Figura A.31 - Prancheta da extremidade maior da garra direita.

A.4 Estrutura completa do robô

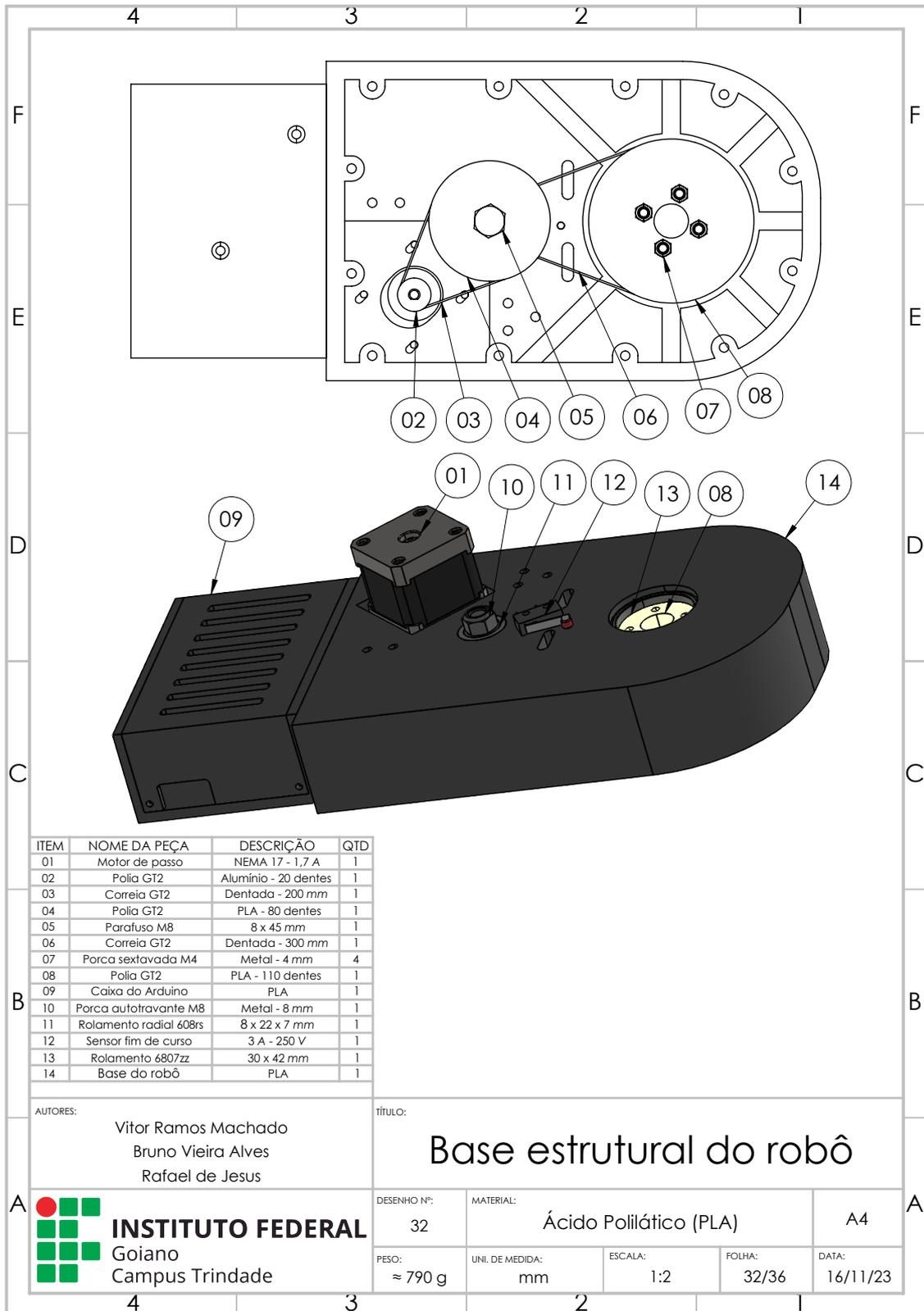


Figura A.32 - Prancheta da base estrutural do robô.

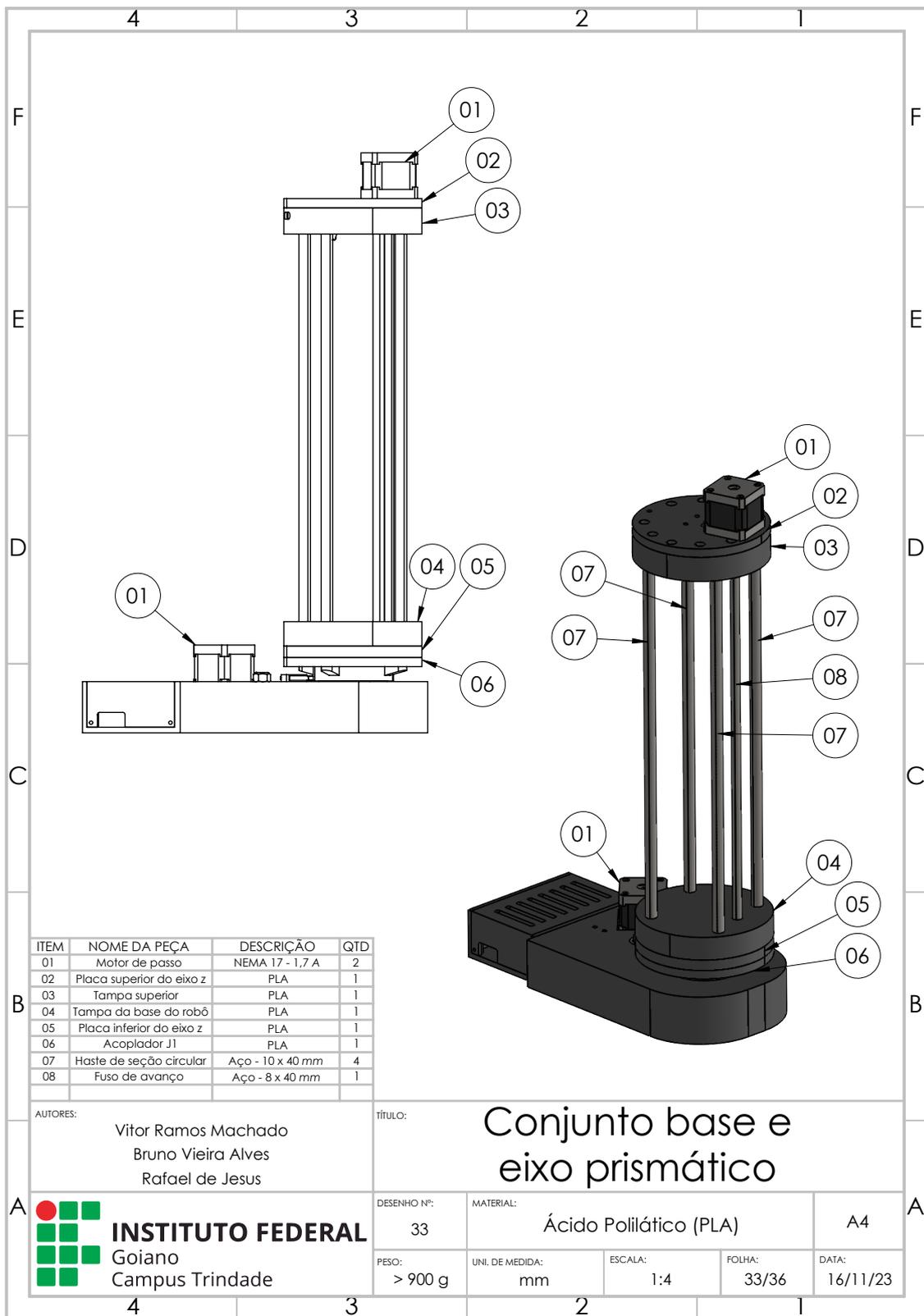


Figura A.33 - Prancheta do conjunto base e eixo prismático.

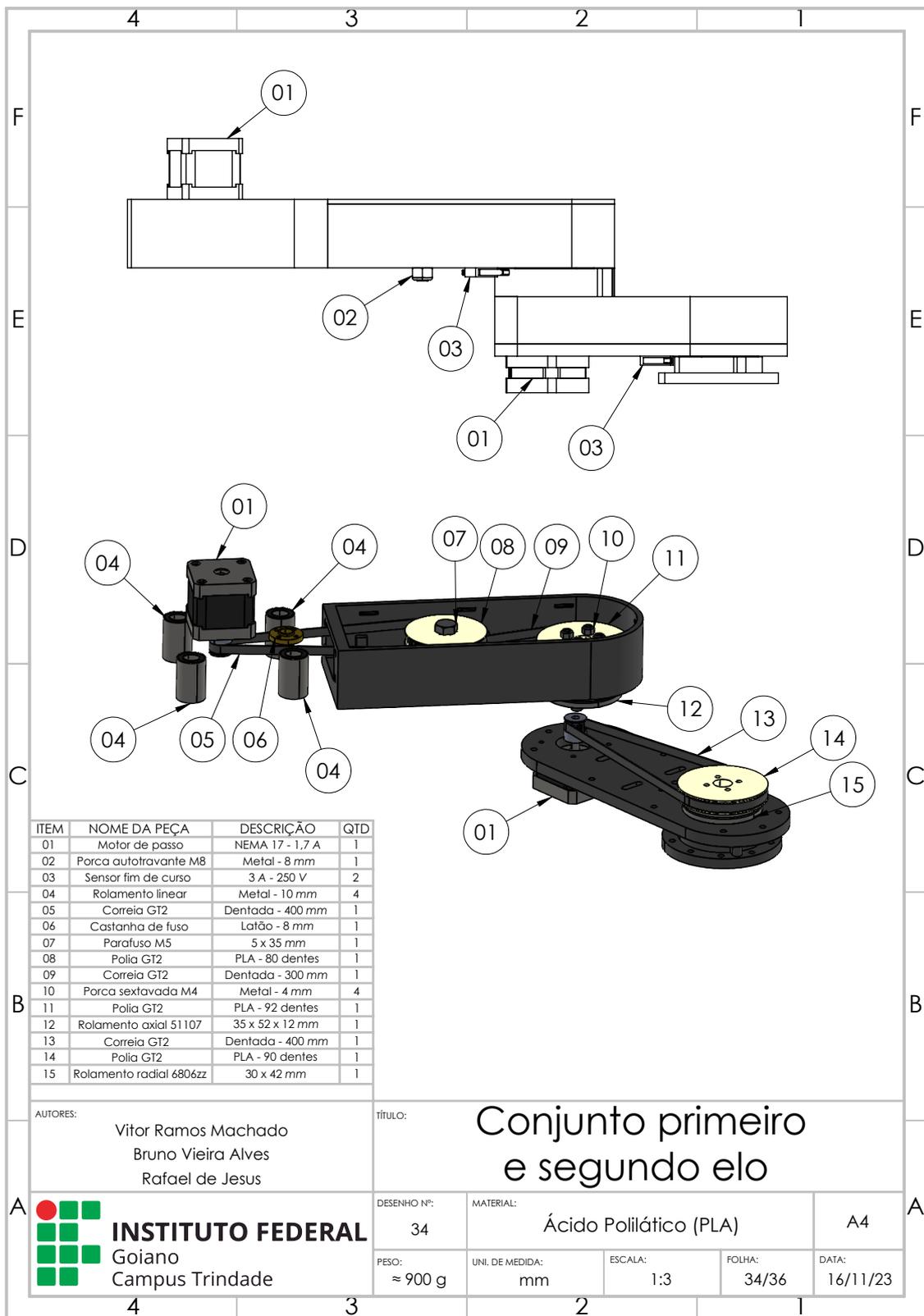


Figura A.34 - Prancheta do conjunto primeiro e segundo elo.

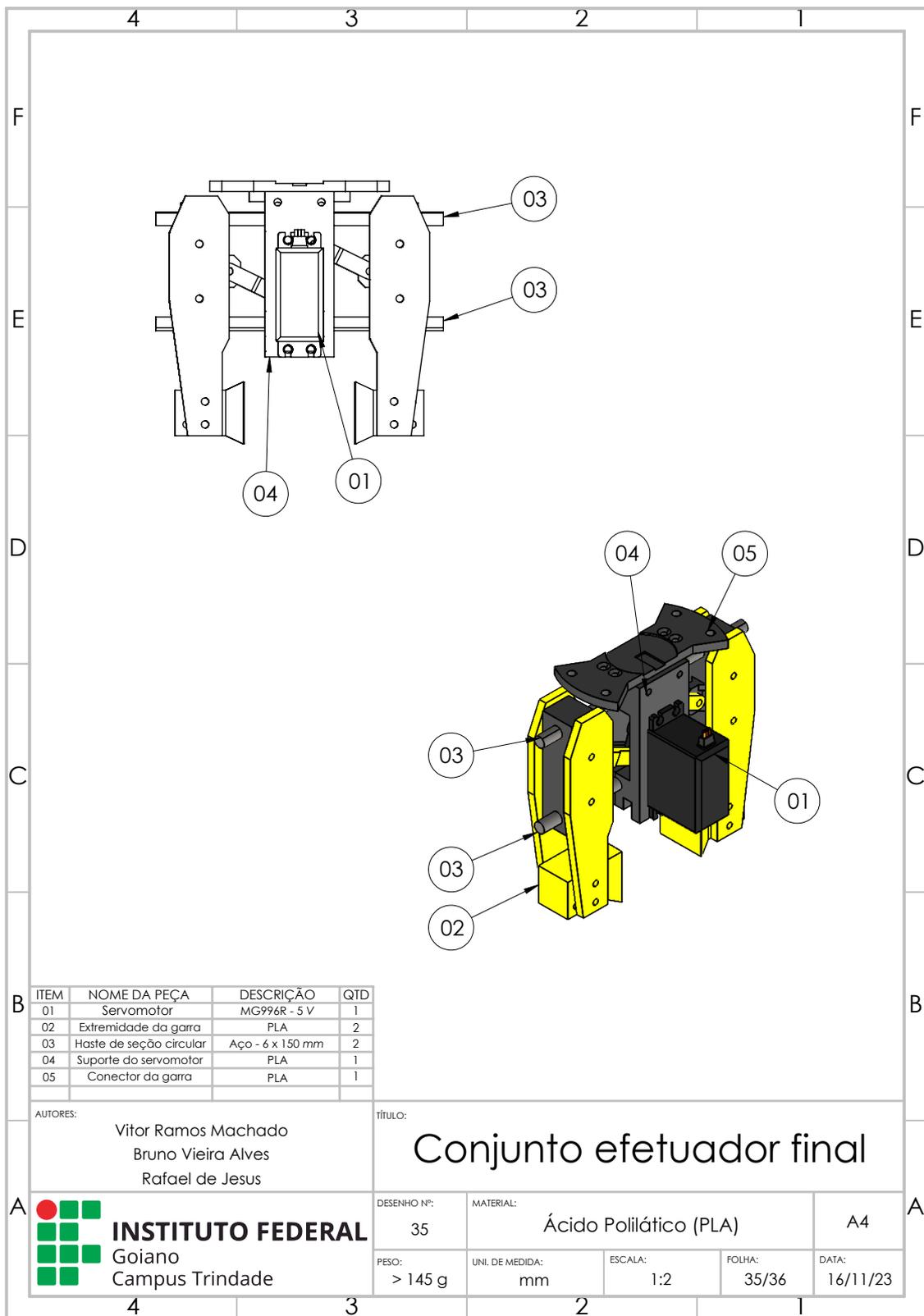


Figura A.35 - Prancheta do conjunto efetuator final.

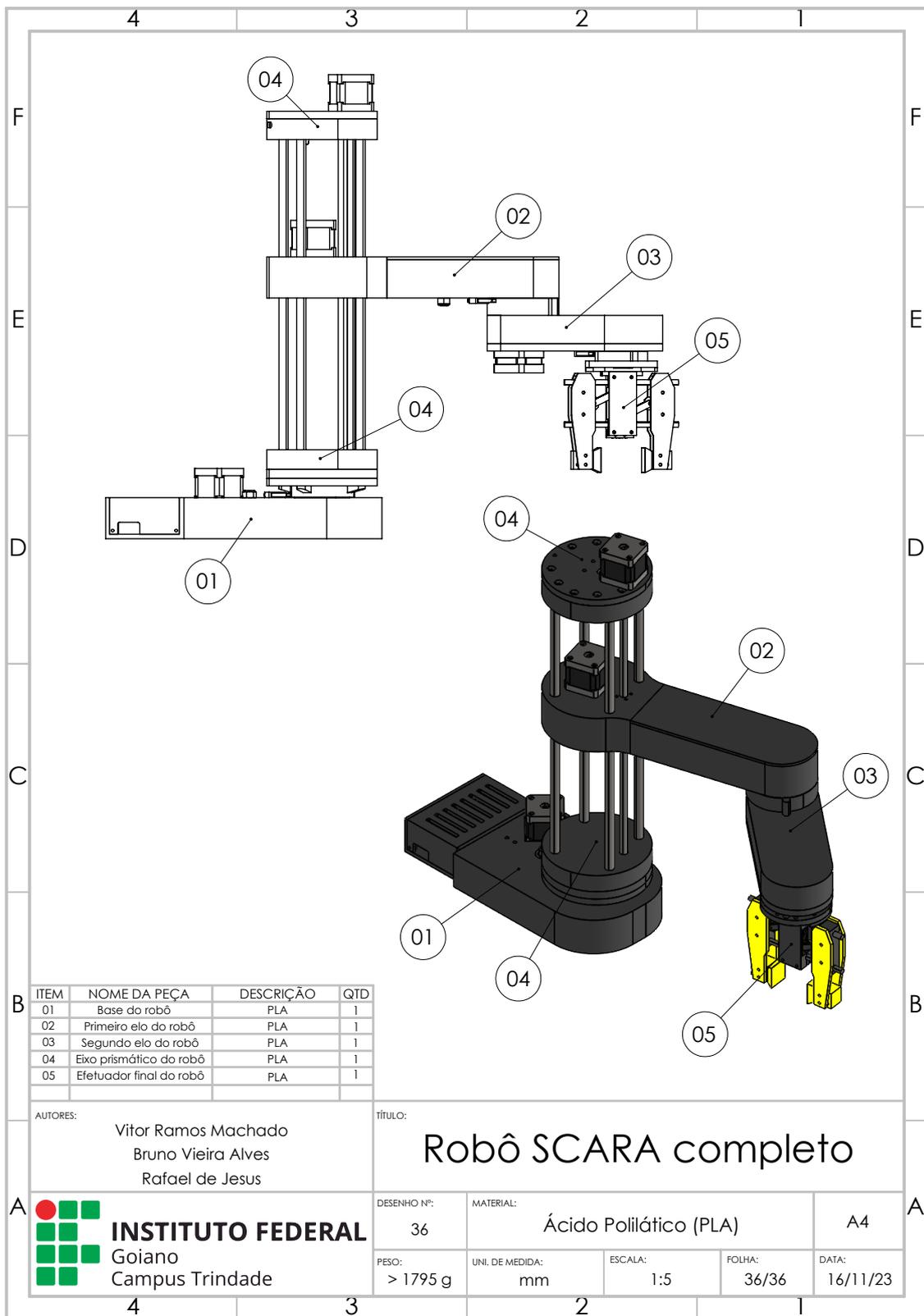


Figura A.36 - Prancheta do robô SCARA completo.

APÊNDICE B

Dados de impressão 3D

Tabela B.1 - Lista de peças 3D impressas - Parte I.

ITEM	NOME	IMP	DIA	HORÁRIO
1	Capa do Braço 01	1	Terça-feira	14:30 - 19:00h
2	Braço 01	1	Terça-feira	20:30 - 23:00h
3	Capa do Braço 02	1	Quarta-feira	14:30 - 21:00h
4	Braço 02	1	Quarta-feira	21:00 - 01:30h
5	Polia GT2 com 80 dentes 1	1	Quinta-feira	14:30 - 16:00h
6	Polia GT2 com 80 dentes 2	1	Quinta-feira	16:00 - 17:30h
7	Polia GT2 com 110 dentes	1	Quinta-feira	17:45 - 20:30h
8	Polia GT2 com 92 dentes	1	Quinta-feira	20:30 - 22:30h
9	Plat. de mont. de eixo z	1	Quinta-feira	22:30 - 07:33h
10	Polia GT2 com 90 dentes	1	Sexta-feira	14:30 - 16:30h
11	Acoplador J1	1	Sexta-feira	18:00 - 00:05h
12	Tampa da caixa do Ard.	2	Terça-feira	14:30 - 17:15h
13	Acoplador J2	2	Terça-feira	18:00 - 21:40h
14	Ligação da garra 01	2	Terça-feira	21:45 - 22:00h
15	Placa inferior do eixo z	2	Quarta-feira	14:30 - 19:00h
16	Des. do mec. da garra	2	Quarta-feira	19:10 - 21:13h
17	Caixa do Arduino	2	Terça-feira	14:30 - 19:20h
18	Acoplador J3	2	Terça-feira	19:30 - 22:30h
19	Placa superior do eixo z	3	Terça-feira	14:30 - 19:00h
20	Ext. maior da garra esq.	3	Terça-feira	19:00 - 20:30h
21	Tampa da base do robô	3	Terça-feira	14:30 - 18:30h
22	Extremidade da garra	4	Terça-feira	20:00 - 21:15h
23	Ligação da garra 02	4	Terça-feira	21:45 - 22:00h
24	Garra esquerda	4	Quarta-feira	14:30 - 16:00h
25	Garra direita	4	Quarta-feira	16:00 - 17:30h
26	Suporte do servomotor	4	Quarta-feira	17:35 - 20:35h
27	Conector da garra em J3	4	Quarta-feira	20:45 - 20:55h
28	Ext. maior da garra dir.	4	Quarta-feira	20:40 - 21:40h
29	Abra. de haste lisa	4	Terça-feira	14:30 - 20:40h
30	Capa da garra do robô	5	Quarta-feira	20:00 - 21:50h
31	Tampa superior	5	Quarta-feira	14:30 - 18:30h
32	Base do robô	5	Quarta-feira	20:00 - 19:30h

Tabela B.2 - Lista de peças 3D impressas - Parte II.

ITEM	NOME	IMP	TEMPO	PESO	COR
1	Capa do Braço 01	1	04h30	46g	Preto
2	Braço 01	1	02h30	117g	Preto
3	Capa do Braço 02	1	06h30	118g	Preto
4	Braço 02	1	04h30	115g	Preto
5	Polia GT2 com 80 dentes 1	1	01h30	19g	Branco
6	Polia GT2 com 80 dentes 2	1	01h30	19g	Branco
7	Polia GT2 com 110 dentes	1	02h45	35g	Branco
8	Polia GT2 com 92 dentes	1	02h00	28g	Branco
9	Plat. de mont. de eixo z	1	09h03	204g	Preto
10	Polia GT2 com 90 dentes	1	02h00	26g	Branco
11	Acoplador J1	1	06h05	95g	Preto
12	Tampa da caixa do Ard.	2	02h45	30g	Preto
13	Acoplador J2	2	03h40	51g	Preto
14	Ligação da garra 01	2	00h15	1g	Preto
15	Placa inferior do eixo z	2	04h30	84g	Preto
16	Des. do mec. da garra	2	02h03	17g	Preto
17	Caixa do Arduino	2	02h50	35g	Preto
18	Acoplador J3	2	03h00	47g	Preto
19	Placa superior do eixo z	3	04h30	83g	Preto
20	Ext. maior da garra esq.	3	01h00	12g	Amarelo
21	Tampa da base do robô	3	04h00	61g	Preto
22	Extremidade da garra	4	01h15	12g	Amarelo
23	Ligação da garra 02	4	00h15	1g	Amarelo
24	Garra esquerda	4	01h30	15g	Amarelo
25	Garra direita	4	01h30	15g	Amarelo
26	Suporte do servomotor	4	03h00	15g	Preto
27	Conector da garra em J3	4	00h10	10g	Preto
28	Ext. maior da garra dir.	4	01h00	12g	Amarelo
29	Abra. de haste lisa	4	05h30	64g	Preto
30	Capa da garra do robô	5	01h20	16g	Preto
31	Tampa superior	5	04h00	55g	Preto
32	Base do robô	5	23h30	325g	Preto

APÊNDICE C

Códigos de simulação do robô SCARA

Nesta seção, apresentam-se dois *scripts* desenvolvidos para a simulação do robô SCARA. Estes códigos foram criados utilizando a linguagem de programação Python e estabelecem uma interface entre os *softwares* de simulação CoppeliaSim e o Jupyter Notebook. O primeiro *script*, referenciado como código C.1, implementa a simulação do robô empregando a abordagem de cinemática direta. Já o segundo, detalhado no código C.2, executa a simulação do robô utilizando a metodologia de cinemática inversa.

C.1 Simulação da cinemática direta do robô SCARA

```
1 #Inicio do codigo:
2 #Deve-se importar algumas bibliotecas, e estabelecer a comunicacao
   entre o Jupyter Notebook e o CoppeliaSim:
3 import zmqRemoteApi #Biblioteca utilizada para estabelecer a
   comunicacao remota com o simulador;
4 import time #Biblioteca utilizada para estabelecer e acessar o
   tempo e a data ao programa (neste programa foi utilizada para
   implementar o 'delay');
5 client = zmqRemoteApi.RemoteAPIClient() #Variavel que recebe a
   comunicacao entre os 'softwares';
6 sim = client.getObject('sim') #Confirmando a comunicacao entre os
   'softwares';
7
8 #Obtem-se os controladores para as articulacoes e o efetuador
   final:
9 j1 = sim.getObject('/base/joint1') #Para a primeira junta (J1);
10 j2 = sim.getObject('/base/joint2') #Para a segunda junta (J2);
11 j3 = sim.getObject('/base/joint3') #Para a terceira junta (J3);
12 suctionPad = sim.getObject('/base/suctionPad') #Para o efetuador
   final (ventosa de succao);
13
14 #Estabelecendo as posicoes das juntas conforme escrito:
15 sim.setJointTargetPosition(j1, 1) #Para a primeira junta (J1);
16 sim.setJointTargetPosition(j2, 0.1) #Para a segunda junta (J2);
17 sim.setJointTargetPosition(j3, 0) #Para a terceira junta (J3);
18
19 #Utiliza-se o temporizador para permitir que o robo execute cada
   passo de forma gradual:
20 print('O robo esta se movendo!') #Exibe-se a imagem de execucao do
   robo;
```

```

21 time.sleep(3) #'Delay' aplicado apos mover para as posicoes;
22
23 #Estabelece a posicao de destino do robo conforme o objeto criado
    (caixas):
24 print('Calculando a Cinematica Direta...')
25 Caixa1 = sim.getObject('/Caixa1')
26 caixa1_pos = sim.getObjectPosition(Caixa1, sim.handle_world)
27 D = MatrixFromPose(caixa1_pos[0], caixa1_pos[1], caixa1_pos[2], 0,
    0, 0)
28 D
29 try:
30     q = sympy.nsolve((T-D), (q1, q2, q3), (1, 1, 2), prec=6)
31 except:
32     print('Nao pode ser executado') #Caso ocorra-se erro ao
    programa, exhibe a mensagem que nao e possivel executa-lo;
33     q = [0 , 0.2 , 0]
34 print(q)
35
36 #Envia-se os angulos para determinada junta:
37 sim.setJointTargetPosition(j1, float(q[0])) #Para a primeira junta
    (J1);
38 sim.setJointTargetPosition(j2, float(q[1])+ 0.026) #Para a segunda
    junta (J2);
39 sim.setJointTargetPosition(j3, float(q[2])) #Para a terceira
    junta (J3);
40 print('O robo esta se movendo!') #Exibe-se a imagem de execucao do
    robo;
41 time.sleep(3) #'Delay' aplicado apos mover para as posicoes;
42 print('OK!') #Mensagem emitida apos ser concluida a primeira etapa
    de posicionamento do robo;
43
44 #Ativando o efetuator final:
45 setEffector(1) #Neste caso levando seu valor a um (1) = pegando o
    objeto;
46 sim.setJointTargetPosition(j2, 0.1) #Para a segunda junta (J2)
    onde encontra-se o efetuator final;
47 time.sleep(2)
48 #Ativando o restante das juntas:
49 sim.setJointTargetPosition(j1, 0) #Para a primeira junta (J1);
50 sim.setJointTargetPosition(j2, 0.2) #Para a segunda junta (J2);
51 sim.setJointTargetPosition(j3, 0) #Para a terceira junta (J3);
52 print('O robo esta se movendo!') #Exibe-se a imagem de execucao do
    robo;
53 time.sleep(3) #'Delay' aplicado apos mover para as posicoes;

```

```

54 print('OK!') #Mensagem emitida apos ser concluida a segunda etapa
    de posicionamento do robo;
55
56 #Realizando os calculos da cinematica direta de conversao de
    angulos para posicoes:
57
58 q1_val = 0 * 3.1416/180 #Para posicoes em X;
59 q2_val = 0.022 #Para posicoes em Y;
60 q3_val = 0 * 3.1416/180 #Para posicoes em Z;
61
62 #Enviando os valores das posicoes calculadas para cada junta:
63 sim.setJointTargetPosition(j1, q1_val) #Para a primeira junta (J1)
    ;
64 sim.setJointTargetPosition(j2, q2_val) #Para a segunda junta (J2);
65 sim.setJointTargetPosition(j3, q3_val) #Para a terceira junta (J3
    );
66 time.sleep(2)
67 setEffector(0) #Neste caso levando seu valor a zero (0) = soltando
    o objeto;
68 time.sleep(3)
69 sim.setJointTargetPosition(j1, 0.5) #Para a primeira junta (J1);
70 sim.setJointTargetPosition(j2, 0.2) #Para a segunda junta (J2);
71 sim.setJointTargetPosition(j3, 0.5) #Para a terceira junta (J3);
72 print('O robo esta se movendo!') #Exibe-se a imagem de execucao do
    robo;
73 time.sleep(2) #'Delay' aplicado apos mover para as posicoes;
74 print('OK!') #Mensagem emitida apos ser concluida a terceira e
    ultima etapa de posicionamento do robo;
75 time.sleep(3) #'Delay' aplicado apos exibir mensagem;
76 #Fim do codigo!

```

Código C.1 - *Script* em Python para simulação do robô utilizando a cinemática direta

C.2 Simulação da cinemática inversa do robô SCARA

```

1 #Inicio do codigo:
2 #Deve-se importar algumas bibliotecas, e estabelecer a comunicacao
    entre o Jupyter Notebook e o CoppeliaSim:
3 import zmqRemoteApi #Biblioteca utilizada para estabelecer a
    comunicacao remota com o simulador;
4 import time #Biblioteca utilizada para estabelecer e acessar o
    tempo e a data ao programa (neste programa foi utilizada para
    implementar o 'delay');
5 client = zmqRemoteApi.RemoteAPIClient() #Variavel que recebe a
    comunicacao entre os 'softwares';
6 sim = client.getObject('sim') #Confirmando a comunicacao entre os

```

```

    'softwares';
7
8 #Obtem-se os controladores para as articulacoes e o efetuator
    final:
9 j1 = sim.getObject('/base/joint1') #Para a primeira junta (J1);
10 j2 = sim.getObject('/base/joint2') #Para a segunda junta (J2);
11 j3 = sim.getObject('/base/joint3') #Para a terceira junta (J3);
12 suctionPad = sim.getObject('/base/suctionPad') #Para o efetuator
    final (ventosa de succao);
13
14 #Estabelecendo as posicoes das juntas conforme escrito e
    selecionado pelo operador:
15 sim.setJointTargetPosition(j1, 0) #Para a primeira junta (J1);
16 sim.setJointTargetPosition(j2, 0.2) #Para a segunda junta (J2);
17 sim.setJointTargetPosition(j3, 0) #Para a terceira junta (J3);
18
19 #Utiliza-se o temporizador para permitir que o robo execute cada
    passo de forma gradual:
20 print('O robo esta se movendo!') #Exibe-se a imagem de execucao do
    robo;
21 time.sleep(3) #'Delay' aplicado apos mover para as posicoes;
22
23 #Estabelece a posicao de destino do robo conforme o objeto criado
    (caixas):
24 print('Calculando a Cinematica Inversa...') #Utilizando atraves da
    cinematica inversa;
25 Caixa1 = sim.getObject('/Caixa1')
26 caixa1_pos = sim.getObjectPosition(Caixa1, sim.handle_world)
27 D = MatrixFromPose(caixa1_pos[0], caixa1_pos[1], caixa1_pos[2], 0,
    0, 0)
28 D
29 try:
30     q = sympy.nsolve((T-D), (q1, q2, q3), (1, 1, 2), prec=6)
31 except:
32     print('Nao pode ser executado') #Caso ocorra-se erro ao
    programa, exibe a mensagem que nao e possivel executa-lo;
33     q = [0 , 0.2 , 0]
34 print(q)
35
36 #Envia-se os angulos para determinada junta:
37 sim.setJointTargetPosition(j1, float(q[0])) #Para a primeira junta
    (J1);
38 sim.setJointTargetPosition(j2, float(q[1])+ 0.026) #Para a segunda
    junta (J2);
39 sim.setJointTargetPosition(j3, float(q[2])) #Para a terceira

```

```

    junta (J3);
40 print('0 robo esta se movendo!') #Exibe-se a imagem de execucao do
    robo;
41 time.sleep(3) #'Delay' aplicado apos mover para as posicoes;
42 print('OK!') #Mensagem emitida apos ser concluida a primeira etapa
    de posicionamento do robo;
43
44 #Ativando o efetuator final:
45 setEffector(1) #Neste caso levando seu valor a um (1) = pegando o
    objeto;
46 sim.setJointTargetPosition(j2, 0.1) #Para a segunda junta (J2)
    onde encontra-se o efetuator final;
47 time.sleep(2)
48 #Ativando o restante das juntas:
49 sim.setJointTargetPosition(j1, 0) #Para a primeira junta (J1);
50 sim.setJointTargetPosition(j2, 0.2) #Para a segunda junta (J2);
51 sim.setJointTargetPosition(j3, 0) #Para a terceira junta (J3);
52 print('0 robo esta se movendo!') #Exibe-se a imagem de execucao do
    robo;
53 time.sleep(3) #'Delay' aplicado apos mover para as posicoes;
54 print('OK!') #Mensagem emitida apos ser concluida a segunda etapa
    de posicionamento do robo;
55
56 #Realizando os calculos da cinematica inversa de conversao de
    posicoes para angulos:
57 q1_val = 0 * 3.1416/180 #Para posicoes em X;
58 q2_val = 0.022 #Para posicoes em Y;
59 q3_val = 0 * 3.1416/180 #Para posicoes em Z;
60
61 #Enviando os valores das posicoes calculadas para cada junta:
62 sim.setJointTargetPosition(j1, q1_val) #Para a primeira junta (J1)
    ;
63 sim.setJointTargetPosition(j2, q2_val) #Para a segunda junta (J2);
64 sim.setJointTargetPosition(j3, q3_val) #Para a terceira junta (J3
    );
65 time.sleep(2)
66 setEffector(0) #Neste caso levando seu valor a zero (0) = soltando
    o objeto;
67 time.sleep(3)
68 sim.setJointTargetPosition(j1, 0) #Para a primeira junta (J1);
69 sim.setJointTargetPosition(j2, 0.2) #Para a segunda junta (J2);
70 sim.setJointTargetPosition(j3, 0) #Para a terceira junta (J3);
71 print('0 robo esta se movendo!') #Exibe-se a imagem de execucao do
    robo;
72 time.sleep(2) #'Delay' aplicado apos mover para as posicoes;

```

```

73 print('OK!') #Mensagem emitida apos ser concluida a terceira etapa
    de posicionamento do robo;
74 time.sleep(3) #'Delay' aplicado apos exibir mensagem;
75
76 #Para cada caixa que sera pegada, usa-se os mesmos comandos:
77 #Obtem-se os controladores para as articulacoes e o atuador final:
78 j1 = sim.getObject('/base/joint1') #Para a primeira junta (J1);
79 j2 = sim.getObject('/base/joint2') #Para a segunda junta (J2);
80 j3 = sim.getObject('/base/joint3') #Para a terceira junta (J3);
81 suctionPad = sim.getObject('/base/suctionPad') #Para o efetuador
    final (ventosa de succao);
82
83 #Move o robo para os valores calculados:
84 sim.setJointTargetPosition(j1, 0) #Para a primeira junta (J1);
85 sim.setJointTargetPosition(j2, 0.2) #Para a segunda junta (J2);
86 sim.setJointTargetPosition(j3, 0) #Para a terceira junta (J3);
87 print('O robo esta se movendo!') #Exibe-se a imagem de execucao do
    robo;
88 time.sleep(3) #'Delay' aplicado apos mover para as posicoes;
89
90 #Recalcula a posicao de destino do robo conforme o objeto criado (
    caixas):
91 print('Calculando a Cinematica Inversa...')
92 Caixa2 = sim.getObject('/Caixa2')
93 caixa2_pos = sim.getObjectPosition(Caixa2, sim.handle_world)
94 D = MatrixFromPose(caixa2_pos[0], caixa2_pos[1], caixa2_pos[2], 0,
    0, 0)
95 D
96 try:
97     q = sympy.nsolve((T-D), (q1, q2, q3), (1, 1, 2), prec=6)
98 except:
99     print('Nao pode ser executado!') #Caso ocorra-se erro ao
    programa, exibe a mensagem que nao e possivel executa-lo;
100     q = [0 , 0.2 , 0]
101 print(q)
102
103 #Novamente, envia-se os angulos para determinada junta,
    atualizando-os:
104 sim.setJointTargetPosition(j1, float(q[0])) #Para a primeira junta
    (J1);
105 sim.setJointTargetPosition(j2, float(q[1])+ 0.026) #Para a segunda
    junta (J2);
106 sim.setJointTargetPosition(j3, float(q[2])) #Para a terceira
    junta (J3);
107 print('O robo esta se movendo!') #Exibe-se a imagem de execucao do

```

```

    robo;
108 time.sleep(3) #'Delay' aplicado apos mover para as posicoes;
109 print('OK!') #Mensagem emitida apos ser concluida a quarta etapa
    de posicionamento do robo;
110
111 #Novamente, ativa-se o efetuator final:
112 setEffector(1) #Neste caso levando seu valor a um (1) = pegando o
    objeto;
113 sim.setJointTargetPosition(j2, 0.1) #Para a segunda junta (J2)
    onde encontra-se o efetuator final;
114 time.sleep(2) #'Delay' aplicado apos mover para as posicoes;
115
116 #Ativando o restante das juntas:
117 sim.setJointTargetPosition(j1, 0) #Para a primeira junta (J1);
118 sim.setJointTargetPosition(j2, 0.2) #Para a segunda junta (J2);
119 sim.setJointTargetPosition(j3, 0) #Para a terceira junta (J3);
120 print('O robo esta se movendo!') #Exibe-se a imagem de execucao do
    robo;
121 time.sleep(3) #'Delay' aplicado apos mover para as posicoes;
122 print('OK!') #Mensagem emitida apos ser concluida a quinta etapa
    de posicionamento do robo;
123 time.sleep(2) #'Delay' aplicado apos exibir mensagem;
124
125 #Move-se o robo para os valores recalculados:
126 sim.setJointTargetPosition(j1, -0.2) #Para a primeira junta (J1);
127 sim.setJointTargetPosition(j2, 0.022) #Para a segunda junta (J2);
128 sim.setJointTargetPosition(j3, 0) #Para a terceira junta (J3);
129 time.sleep(2)
130
131 #Novamente, ativa-se o efetuator final:
132 setEffector(0) #Neste caso levando seu valor a zero (0) = soltando
    o objeto;
133 time.sleep(3)
134 sim.setJointTargetPosition(j1, 0) #Para a primeira junta (J1);
135 sim.setJointTargetPosition(j2, 0.2) #Para a segunda junta (J2);
136 sim.setJointTargetPosition(j3, 0) #Para a terceira junta (J3);
137 print('O robo esta se movendo!') #Exibe-se a imagem de execucao do
    robo;
138 time.sleep(2) #'Delay' aplicado apos mover para as posicoes;
139 print('OK!') #Mensagem emitida apos ser concluida a sexta etapa de
    posicionamento do robo;
140 time.sleep(3) #'Delay' aplicado apos exibir mensagem;
141
142 #Para cada caixa que sera pegada, usa-se os mesmos comandos:
143 #Novamente, obtem-se os controladores para as articulacoes e o

```

```

    atuador final:
144 j1 = sim.getObject('/base/joint1') #Para a primeira junta (J1);
145 j2 = sim.getObject('/base/joint2') #Para a segunda junta (J2);
146 j3 = sim.getObject('/base/joint3') #Para a terceira junta (J3);
147 suctionPad = sim.getObject('/base/suctionPad') #Para o efetuador
    final (ventosa de succao);
148
149 #Move o robo para os valores calculados:
150 sim.setJointTargetPosition(j1, 0) #Para a primeira junta (J1);
151 sim.setJointTargetPosition(j2, 0.2) #Para a segunda junta (J2);
152 sim.setJointTargetPosition(j3, 0) #Para a terceira junta (J3);
153 print('O robo esta se movendo!') #Exibe-se a imagem de execucao do
    robo;
154 time.sleep(3) #'Delay' aplicado apos mover para as posicoes;
155
156 #Posicao de destino final:
157 #Novamente, recalcula a posicao de destino do robo conforme o
    objeto criado (caixas):
158 print('Calculando a Cinematica Inversa...')
159 Caixa3 = sim.getObject('/Caixa3')
160 caixa3_pos = sim.getObjectPosition(Caixa3, sim.handle_world)
161 D = MatrixFromPose(caixa3_pos[0], caixa3_pos[1], caixa3_pos[2], 0,
    0, 0)
162 D
163 try:
164     q = sympy.nsolve((T-D), (q1, q2, q3), (1, 1, 2), prec=6)
165 except:
166     print('Nao pode ser executado!') #Caso ocorra-se erro ao
    programa, exibe a mensagem que nao e possivel executa-lo;
167     q = [0 , 0.2 , 0]
168 print(q)
169
170 #Novamente, envia-se os angulos para determinada junta,
    atualizando-os:
171 sim.setJointTargetPosition(j1, float(q[0])) #Para a primeira junta
    (J1);
172 sim.setJointTargetPosition(j2, float(q[1])+ 0.026) #Para a segunda
    junta (J2);
173 sim.setJointTargetPosition(j3, float(q[2])) #Para a terceira
    junta (J3);
174 print('O robo esta se movendo!') #Exibe-se a imagem de execucao do
    robo;
175 time.sleep(3)
176 print('OK!') #Mensagem emitida apos ser concluida a setima etapa
    de posicionamento do robo;

```

```

177
178 #Novamente, ativa-se o efetuator final:
179 setEffector(1) #Neste caso levando seu valor a um (1) = pegando o
    objeto;
180 sim.setJointTargetPosition(j2, 0.1) #Para a segunda junta (J2)
    onde encontra-se o efetuator final;
181 time.sleep(2) #'Delay' aplicado apos mover para as posicoes;
182
183 #Reativando o restante das juntas:
184 sim.setJointTargetPosition(j1, 0) #Para a primeira junta (J1);
185 sim.setJointTargetPosition(j2, 0.2) #Para a segunda junta (J2);
186 sim.setJointTargetPosition(j3, 0) #Para a terceira junta (J3);
187 print('O robo esta se movendo!') #Exibe-se a imagem de execucao do
    robo;
188 time.sleep(3) #'Delay' aplicado apos mover para as posicoes;
189 print('OK!') #Mensagem emitida apos ser concluida a oitava etapa
    de posicionamento do robo;
190 time.sleep(2) #'Delay' aplicado apos exibir mensagem;
191
192 #Move-se o robo para os valores recalculados e para sua posicao
    final ao executar todas suas tarefas:
193 sim.setJointTargetPosition(j1, 0.2) #Para a primeira junta (J1);
194 sim.setJointTargetPosition(j2, 0.022) #Para a segunda junta (J2);
195 sim.setJointTargetPosition(j3, 0) #Para a terceira junta (J3);
196 time.sleep(2)
197
198 #Novamente, ativa-se o efetuator final pela ultima vez, colocando-
    o em sua posicao final:
199 setEffector(0) #Neste caso levando seu valor a zero (0) = soltando
    o objeto e encerrando o codigo;
200 time.sleep(3)
201 sim.setJointTargetPosition(j1, 0) #Para a primeira junta (J1);
202 sim.setJointTargetPosition(j2, 0.2) #Para a segunda junta (J2);
203 sim.setJointTargetPosition(j3, 0) #Para a terceira junta (J3);
204 print('O robo esta se movendo!') #Exibe-se a imagem de execucao do
    robo;
205 time.sleep(2) #'Delay' aplicado apos mover para as posicoes;
206 print('OK!') #Mensagem emitida apos ser concluida a nona e ultima
    etapa de posicionamento do robo;
207 #Fim do codigo!

```

Código C.2 - *Script* em Python para simulação do robô utilizando a cinemática inversa

APÊNDICE D

Códigos de controle do robô SCARA

Nesta seção, são apresentados dois *scripts* para o controle do robô SCARA. O primeiro, elaborado através do *software* Arduino IDE utilizando C++, gerencia a comunicação entre o microcontrolador e a parte eletroeletrônica do projeto, possibilitando a movimentação do manipulador, conforme evidenciado no código D.1. O segundo *script*, desenvolvido no *software* Processing com a linguagem Java, é dedicado ao controle do robô, incluindo a criação da interface gráfica do usuário (GUI) do projeto, detalhada no código D.2.

D.1 Movimentação e dinâmica do robô SCARA

```
1 //Inicio do codigo:
2 //Insercao das bibliotecas utilizadas no robo:
3 #include <AccelStepper.h> //Biblioteca para realizar o ajuste de
   aceleracao e velocidade dos servo motores;
4 #include <Servo.h> //Biblioteca para realizar o funcionamento e
   controle dos servo motores;
5 #include <math.h> //Biblioteca para realizar as operacoes
   matematicas do codigo;
6
7 //Definicao dos pinos das chaves fim de curso:
8 #define fimdecurso1 11 //Para o motor 01 (J1);
9 #define fimdecurso2 10 //Para o motor 02 (J2);
10 #define fimdecurso3 9 //Para o motor 03 (J3);
11 #define fimdecurso4 A3 //Para o motor 04 (Eixo Z);
12
13 //Definicao dos pinos dos motores de passo que serao utilizados:
14 //Seguindo a ordem temos os seguintes pinos e entradas: (driver,
   STEP, DIR);
15 AccelStepper motor1(1, 2, 5); //(driver, STEP, DIR);
16 AccelStepper motor2(1, 3, 6);
17 AccelStepper motor3(1, 4, 7);
18 AccelStepper motor4(1, 12, 13);
19 Servo efetuador; //Criacao da variavel para controle da garra do
   robo;
20
21 //Declaracao das variaveis para calculos das cinematicas direta e
   inversa:
22 double x = 10.0;
23 double y = 10.0;
24 double L1 = 228; //Comprimento do braco 01 - L1 = 228mm;
```

```

25 double L2 = 136.5; //Comprimento do braco 02 - L2 = 136.5mm;
26 double theta1, theta2, phi, z;
27
28 //Declaracao das variaveis para controle de posicao dos motores de
    passo:
29 int posicaomotor1, posicaomotor2, posicaomotor3, posicaomotor4;
30
31 //Declaracao das variaveis de angulos dos motores para ajuste de
    posicao:
32 const float theta1angulostep = 44.444444;
33 const float theta2angulostep = 35.555555;
34 const float phiangulostep = 10;
35 const float distanciazstep = 100;
36 byte inputValue[5]; //Criacao de vetor para valor de entrada;
37 int k = 0; //Criacao de variavel para controle de posicao;
38 String conteudo = ""; //Criacao de 'string' vazia para receber
    valores posteriores;
39 int data[10]; //Criacao de vetor para receber dados de comando;
40
41 //Declaracao de vetores para angulos e posicoes dos motores:
42 int vetortheta1[100];
43 int vetortheta2[100];
44 int vetorphi[100];
45 int vetorz[100];
46 int vetorgarra[100];
47 int contador = 0; //Criacao de variavel para contar o numero de
    posicoes escolhidas;
48
49 //Inicio do codigo de configuracao do controle do robo:
50 void setup() {
51     Serial.begin(115200);
52
53 //Definicao das chaves fim de curso como entrada ('INPUT'):
54     pinMode(fimdecurso1, INPUT_PULLUP);
55     pinMode(fimdecurso2, INPUT_PULLUP);
56     pinMode(fimdecurso3, INPUT_PULLUP);
57     pinMode(fimdecurso4, INPUT_PULLUP);
58
59 //Definicao da velocidade e aceleracao maxima de cada motor de
    passo:
60     motor1.setMaxSpeed(4000);
61     motor1.setAcceleration(2000);
62     motor2.setMaxSpeed(4000);
63     motor2.setAcceleration(2000);
64     motor3.setMaxSpeed(4000);

```

```

65   motor3.setAcceleration(2000);
66   motor4.setMaxSpeed(4000);
67   motor4.setAcceleration(2000);
68
69   //Definicao da porta do servo motor da garra:
70   efetuator.attach(A0, 600, 2500);
71   data[6] = 180; //Valor inicial do servo motor = garra aberta;
72   efetuator.write(data[6]);
73   delay(1000);
74   data[5] = 100;
75   inicio(); //Chamada para entrar na funcao de inicializacao do
       robo;
76 }
77
78 //Execucao do codigo em 'looping' para comando do robo:
79 void loop() {
80
81   //Configuracao para processamento de dados no 'Processing':
82   if (Serial.available()) {
83       conteudo = Serial.readString(); //Le os dados recebidos do '
           Processing';
84
85   //Extrai os dados da 'string' e coloca em variaveis inteiras
           separadas (data[] array):
86       for (int i = 0; i < 10; i++) {
87           int index = conteudo.indexOf(","); //Localiza o primeiro
           ",";
88           data[i] = atol(conteudo.substring(0, index).c_str()); //
           Extrai o numero do inicio ate ",";
89           conteudo = conteudo.substring(index + 1); //Remove o numero
           da 'string';
90       }
91
92   //Funcionalidade de cada vetor a partir de sua posicao:
93   //OBS: deixe comentado, estas linhas abaixo servem apenas para
           leitura e entendimento do operador!
94   /*
95   data[0] - Status do BOTAO DE SALVAR;
96   data[1] - Status do BOTAO DE REPRODUZIR;
97   data[2] - Angulo da JUNTA 01;
98   data[3] - Angulo da JUNTA 02;
99   data[4] - Angulo da JUNTA 03;
100  data[5] - Posicao em Z;
101  data[6] - Valor da GARRA;
102  data[7] - Valor de VELOCIDADE;

```

```

103     data[8] - Valor de ACELERACAO;
104 */
105
106 //Se o BOTAO DE SALVAR for pressionado, armazena os dados nas
    matrizes apropriadas:
107     if (data[0] == 1) {
108
109 //Aqui armazena-se os valores em passos = angulo * variavel angulo
    de 'step':
110         vetortheta1[contador] = data[2] * theta1angulostep;
111         vetortheta2[contador] = data[3] * theta2angulostep;
112         vetorphi[contador] = data[4] * phiangulostep;
113         vetorz[contador] = data[5] * distanciazstep;
114         vetorgarra[contador] = data[6];
115         contador++;
116     }
117
118 //Apaga-se os dados de comando e posicao:
119     if (data[0] == 2) {
120
121 //Funcoes para limpar os dados da matriz para zero (0):
122         memset(vetortheta1, 0, sizeof(vetortheta1));
123         memset(vetortheta2, 0, sizeof(vetortheta2));
124         memset(vetorphi, 0, sizeof(vetorphi));
125         memset(vetorz, 0, sizeof(vetorz));
126         memset(vetorgarra, 0, sizeof(vetorgarra));
127         contador = 0;
128     }
129 }
130
131 //Se o BOTAO DE REPRODUZIR for pressionado:
132 while (data[1] == 1) {
133     motor1.setSpeed(data[7]);
134     motor2.setSpeed(data[7]);
135     motor3.setSpeed(data[7]);
136     motor4.setSpeed(data[7]);
137     motor1.setAcceleration(data[8]);
138     motor2.setAcceleration(data[8]);
139     motor3.setAcceleration(data[8]);
140     motor4.setAcceleration(data[8]);
141
142 //Execucao de todas as etapas armazenadas nos vetores:
143     for (int i = 0; i <= contador - 1; i++) {
144         if (data[1] == 0) {
145             break;

```

```

146     }
147     motor1.moveTo(vetortheta1[i]);
148     motor2.moveTo(vetortheta2[i]);
149     motor3.moveTo(vetorphi[i]);
150     motor4.moveTo(vetorz[i]);
151     while (motor1.currentPosition() != vetortheta1[i] || motor2.
currentPosition() != vetortheta2[i] || motor3.currentPosition
() != vetorphi[i] || motor4.currentPosition() != vetorz[i]) {
152         motor1.run();
153         motor2.run();
154         motor3.run();
155         motor4.run();
156     }
157     if (i == 0) {
158         efetuador.write(vetorgarra[i]);
159     }
160     else if (vetorgarra[i] != vetorgarra[i - 1]) {
161         efetuador.write(vetorgarra[i]);
162         delay(800); //Espera-se 0.8 segundos (s) para o servo
motor agarrar ou soltar = o servo permanece lento;
163     }
164
165 //Verifica-se mudancas na velocidade e aceleracao ou parada do
programa:
166     if (Serial.available()) {
167         conteudo = Serial.readString(); //Le os dados recebidos do
'Processing';
168
169 //Extrai os dados da 'string' e coloca em variaveis inteiras
separadas (data[] array):
170         for (int i = 0; i < 10; i++) {
171             int index = conteudo.indexOf(","); //Localiza o primeiro
",";
172             data[i] = atol(conteudo.substring(0, index).c_str()); //
Extrai o numero do inicio ate ",";
173             conteudo = conteudo.substring(index + 1); //Remove o
numero da 'string';
174         }
175
176         if (data[1] == 0) {
177             break;
178         }
179 //Define-se a velocidade e aceleracao dos motores de passo:
180         motor1.setSpeed(data[7]);
181         motor2.setSpeed(data[7]);

```

```

182     motor3.setSpeed(data[7]);
183     motor4.setSpeed(data[7]);
184     motor1.setAcceleration(data[8]);
185     motor2.setAcceleration(data[8]);
186     motor3.setAcceleration(data[8]);
187     motor4.setAcceleration(data[8]);
188 }
189 }
190 }
191
192 //Altera-se a posicao de cada um dos motores de passo de acordo
    com o angulo escolhido:
193 posicaomotor1 = data[2] * theta1angulostep;
194 posicaomotor2 = data[3] * theta2angulostep;
195 posicaomotor3 = data[4] * phiangulostep;
196 posicaomotor4 = data[5] * distanciazstep;
197
198 //Altera-se a velocidade enquanto executa o programa:
199 motor1.setSpeed(data[7]);
200 motor2.setSpeed(data[7]);
201 motor3.setSpeed(data[7]);
202 motor4.setSpeed(data[7]);
203
204 //Altera-se a aceleracao enquanto executa o programa:
205 motor1.setAcceleration(data[8]);
206 motor2.setAcceleration(data[8]);
207 motor3.setAcceleration(data[8]);
208 motor4.setAcceleration(data[8]);
209
210 //Movimenta-se os motores de passo de acordo com o angulo
    calculado:
211 motor1.moveTo(posicaomotor1);
212 motor2.moveTo(posicaomotor2);
213 motor3.moveTo(posicaomotor3);
214 motor4.moveTo(posicaomotor4);
215
216 //Conferimos se os motores de passo estao nas posicoes que foram
    definidas pelos calculos:
217 while (motor1.currentPosition() != posicaomotor1 || motor2.
    currentPosition() != posicaomotor2 || motor3.currentPosition()
    != posicaomotor3 || motor4.currentPosition() != posicaomotor4
    ) {
218     motor1.run(); //Enquanto a posicao nao for estabelecida, os
    motores de passo estarao em funcionamento ate atingir as
    respectivas posicoes;

```

```

219     motor2.run();
220     motor3.run();
221     motor4.run();
222 }
223 delay(100);
224 efetuator.write(data[6]);
225 delay(300);
226 }
227
228 //Funcao criada para atualizacoes da comunicacao do 'Processing'
    com o monitor 'serial':
229 void serialFlush() {
230     while (Serial.available() > 0) { //Enquanto houver caracteres no
        'buffer serial', porque Serial.available e > 0;
231         Serial.read(); //Pega um caractere;
232     }
233 }
234
235 //Funcao executada ao ligar o robo;
236 void inicio() {
237
238     //Este e o "MODO INICIAR", todos os motores de passo voltam as
        suas posicoes iniciais para entao o programa funcionar;
239
240     //Iniciando o motor de passo 04 (Eixo Z):
241     while (digitalRead(fimdecurso4) != 1) {
242         motor4.setSpeed(-1500);
243         motor4.runSpeed();
244         motor4.setCurrentPosition(100); //Quando o fim de curso e
            pressionado, definimos a posicao do motor 04 para cem (100)
            passos;
245     }
246     delay(20);
247     motor4.moveTo(10000);
248     while (motor4.currentPosition() != 10000) {
249         motor4.run();
250     }
251
252     //Iniciando o motor de passo 03 (Junta J3):
253     while (digitalRead(fimdecurso3) != 1) {
254         motor3.setSpeed(-1100);
255         motor3.runSpeed();
256         motor3.setCurrentPosition(-1662); //Quando o fim de curso e
            pressionado, definimos a posicao do motor 03 para mil
            seiscentos e sessenta e dois (-1662) passos negativos;

```

```

257 }
258 delay(20);
259
260 motor3.moveTo(0);
261 while (motor3.currentPosition() != 0) {
262     motor3.run();
263 }
264
265 //Iniciando o motor de passo 02 (Junta J2):
266 while (digitalRead(fimdecorso2) != 1) {
267     motor2.setSpeed(-1300);
268     motor2.runSpeed();
269     motor2.setCurrentPosition(-5420); //Quando o fim de curso e
    pressionado, definimos a posicao do motor 02 para cinco mil
    quatrocentos e vinte (-5420) passos negativos;
270 }
271 delay(20);
272
273 motor2.moveTo(0);
274 while (motor2.currentPosition() != 0) {
275     motor2.run();
276 }
277
278 //Iniciando o motor de passo 01 (Junta J1):
279 while (digitalRead(fimdecorso1) != 1) {
280     motor1.setSpeed(-1200);
281     motor1.runSpeed();
282     motor1.setCurrentPosition(-3955); //Quando o fim de curso e
    pressionado, definimos a posicao do motor 01 para tres mil
    novecentos e cinquenta e cinco (-3955) passos negativos;
283 }
284 delay(20);
285 motor1.moveTo(0);
286 while (motor1.currentPosition() != 0) {
287     motor1.run();
288 }
289 }
290 //Fim do codigo!

```

Código D.1 - *Script* de Arduino para obtenção da movimentação do robô

D.2 Interface gráfica de controle do robô SCARA

```

1 //Inicio do codigo:
2 //Insercao das bibliotecas utilizadas na criacao da interface
    grafica de usuario (GUI):

```

```

3 import processing.serial.*; //Biblioteca para realizar a
  comunicacao entre o Arduino IDE e o 'Processing';
4 import controlP5.*; //Biblioteca "ControlP5" para criar botoes,
  deslizadores, potenciometros, caixas para alterar textos e etc
  .;
5 import static processing.core.PApplet.*; //Biblioteca para
  processamento da tela de interacao durante a execucao do
  programa;
6
7 Serial myPort; //Definicao da porta 'serial' para comunicacao com
  o Arduino IDE;
8 ControlP5 cp5; //Definicao do objeto de controle do "ControlP5";
9
10 //Declaracao de variaveis inteiras para as caixas deslizantes e
  caixas de aumento de parametros do robo:
11 int j1Slider = 0;
12 int j2Slider = 0;
13 int j3Slider = 0;
14 int zSlider = 100;
15 int j1JogValue = 0;
16 int j2JogValue = 0;
17 int j3JogValue = 0;
18 int zJogValue = 0;
19 int speedSlider = 500;
20 int accelerationSlider = 500;
21 int gripperValue = 180;
22 int gripperAdd=180;
23 int positionsCounter = 0;
24 int somar = 0;
25
26 //Declaracao de variaveis inteiras para salvamento das posicoes do
  robo:
27 int saveStatus = 0;
28 int runStatus = 0;
29
30 //Declaracao de variaveis inteiras para salvar o status dos
  valores escolhidos e configurados das caixas deslizantes e
  caixas de aumento de parametros:
31 int slider1Previous = 0;
32 int slider2Previous = 0;
33 int slider3Previous = 0;
34 int sliderzPrevious = 100;
35 int speedSliderPrevious = 500;
36 int accelerationSliderPrevious = 500;
37 int gripperValuePrevious = 100;

```

```

38 boolean activeIK = false; //Declaracao de variavel para o controle
    da garra do robo;
39
40 //Declaracao de variaveis inteiras para receber valores de
    comprimento, posicao e angulo para calculos de cinematicas
    direta e inversa:
41 int xP=365;
42 int yP=0;
43 int zP=100;
44 float L1 = 228; //Comprimento do primeiro braco do robo, L1 = 228
    mm;
45 float L2 = 136.5; //Comprimento do primeiro braco do robo, L2 =
    136.5mm;
46 float theta1, theta2, phi, z;
47 PImage img; //Declaracao de variavel para importar imagem para o '
    script';
48
49 //Declaracao de 'string' para receber a posicao e o numero de
    posicoes que o robo foi submetido;
50 String[] positions = new String[100];
51 String data;
52
53 //Inicio do codigo de configuracao da interface grafica de
    controle do robo:
54 void setup() {
55
56     size(1920, 1080); //Define o tamanho da area de trabalho da tela
        de execucao da GUI (1920x1080 'pixels' - Full HD);
57     img = loadImage("EUROBO.png"); //Carregamento da imagem do robo
        disposta na tela de execucao (a imagem inserida deve estar na
        mesma pasta do codigo 'Processing' (.PDE);
58     img.resize(1280, 720); //Redimensionando o tamanho da imagem
        (1280x720 'pixel's - HD: tamanho definido conforme tamanho e
        qualidade da imagem);
59     myPort = new Serial(this, "COM3", 115200); //Quando executar o
        codigo, descomentar esta linha e inserir a "porta COM" onde
        esta conectada a placa Arduino;
60
61     cp5 = new ControlP5(this); //Chamado da biblioteca "ControlP5";
62
63     //Definicao da fonte utilizada, suas variacoes e seus tamanhos:
64     PFont pfont = createFont("Red Hat Display Medium", 25, true); //
        Use verdadeiro/falso para: suave/nao suave;
65     ControlFont font = new ControlFont(pfont, 34);
66     ControlFont font2 = new ControlFont(pfont, 25);

```

```

67 ControlFont font3 = new ControlFont(pfont, 27);
68 textFont(pfont);
69
70 //Inicio da criacao dos botoes deslizantes e caixas de definicao
    de parametros:
71 //Para o controle da junta J1:
72 //Botao de controle deslizante J1:
73 cp5.addSlider("j1Slider")
74     .setColorBackground(#042d4f)
75     .setColorForeground(#0270ca)
76     .setPosition(1360, 290)
77     .setFont(font)
78     .setSize(450, 60)
79     .setRange(-90, 266) //Faixa do controle deslizante,
    corresponde ao angulo da articulacao 1 (J1) ou "theta1" para o
    qual o robo pode se mover;
80     .setColorLabel(#4a5c78)
81     .setFont(font)
82     .setCaptionLabel("");
83
84 //Botao para diminuir valores de J1:
85 cp5.addButton("j1JogMinus")
86     .setPosition(1360, 365)
87     .setSize(145, 45)
88     .setFont(font3)
89     .setCaptionLabel("MOVER -");
90
91 //Botao para aumentar valores de J1:
92 cp5.addButton("j1JogPlus")
93     .setPosition(1665, 365)
94     .setSize(145, 45)
95     .setFont(font3)
96     .setCaptionLabel("MOVER +");
97
98 //Botao para aumentar ou diminuir multiplicador dos valores de J1:
99 cp5.addNumberbox("j1JogValue")
100     .setPosition(1554, 365)
101     .setSize(62, 45)
102     .setRange(0, 50)
103     .setFont(font3)
104     .setMultiplier(0.1)
105     .setValue(1)
106     .setDirection(Controller.HORIZONTAL) //Pode-se mudar a direcao
    do controle para esquerda ou direita;
107     .setCaptionLabel("");

```

```

108
109 //Para o controle da junta J2:
110 //Botao de controle deslizante J2:
111 cp5.addSlider("j2Slider")
112     .setColorBackground(#042d4f)
113     .setColorForeground(#0270ca)
114     .setPosition(1360, 460)
115     .setSize(450, 60)
116     .setRange(-150, 150) //Faixa do controle deslizante,
corresponde ao angulo da articulacao 2 (J2) ou "theta2" para o
qual o robo pode se mover;
117     .setColorLabel(#3269c2)
118     .setFont(font)
119     .setCaptionLabel("");
120
121 //Botao para diminuir valores de J2:
122 cp5.addButton("j2JogMinus")
123     .setPosition(1360, 535)
124     .setSize(145, 45)
125     .setFont(font3)
126     .setCaptionLabel("MOVER -");
127
128 //Botao para aumentar valores de J2:
129 cp5.addButton("j2JogPlus")
130     .setPosition(1665, 535)
131     .setSize(145, 45)
132     .setFont(font3)
133     .setCaptionLabel("MOVER +");
134
135 //Botao para aumentar ou diminuir multiplicador dos valores de J2:
136 cp5.addNumberbox("j2JogValue")
137     .setPosition(1554, 535)
138     .setSize(67, 45)
139     .setRange(0, 50)
140     .setFont(font3)
141     .setMultiplier(0.1)
142     .setValue(1)
143     .setDirection(Controllor.HORIZONTAL) //Pode-se mudar a direcao
do controle para esquerda ou direita;
144     .setCaptionLabel("");
145
146 //Para o controle da junta J3:
147 //Botao de controle deslizante J3:
148 cp5.addSlider("j3Slider")
149     .setColorBackground(#042d4f)

```

```

150     .setColorForeground(#0270ca)
151     .setPosition(1360, 630)
152     .setSize(450, 60)
153     .setRange(-162, 162) //Faixa do controle deslizante,
corresponde ao angulo da articulacao 3 (J3) ou "phi" para o
qual o robo pode se mover;
154     .setColorLabel(#4a5c78)
155     .setFont(font)
156     .setCaptionLabel("");
157
158 //Botao para diminuir valores de J3:
159 cp5.addButton("j3JogMinus")
160     .setPosition(1360, 705)
161     .setSize(145, 45)
162     .setFont(font3)
163     .setCaptionLabel("MOVER -");
164
165 //Botao para aumentar valores de J3:
166 cp5.addButton("j3JogPlus")
167     .setPosition(1665, 705)
168     .setSize(145, 45)
169     .setFont(font3)
170     .setCaptionLabel("MOVER +");
171
172 //Botao para aumentar ou diminuir multiplicador dos valores de J3:
173 cp5.addNumberbox("j3JogValue")
174     .setPosition(1554, 705)
175     .setSize(67, 45)
176     .setRange(0, 50)
177     .setFont(font3)
178     .setMultiplier(0.1)
179     .setValue(1)
180     .setDirection(Controller.HORIZONTAL) //Pode-se mudar a direcao
do controle para esquerda ou direita;
181     .setCaptionLabel("");
182
183
184 //Para o controle do eixo Z:
185 //Botao de controle deslizante Z:
186 cp5.addSlider("zSlider")
187     .setColorBackground(#042d4f)
188     .setColorForeground(#0270ca)
189     .setPosition(1360, 800)
190     .setSize(450, 60)
191     .setRange(0, 160) //Faixa do controle deslizante, corresponde

```

```

    ao angulo do eixo Z (Z) para o qual o robo pode se mover;
192     .setColorLabel(#3269c2)
193     .setFont(font)
194     .setCaptionLabel("");
195
196 //Botao para diminuir valores de Z:
197     cp5.addButton("zJogMinus")
198         .setPosition(1360, 875)
199         .setSize(145, 45)
200         .setFont(font3)
201         .setCaptionLabel("MOVER -");
202
203 //Botao para aumentar valores de Z:
204     cp5.addButton("zJogPlus")
205         .setPosition(1665, 875)
206         .setSize(145, 45)
207         .setFont(font3)
208         .setCaptionLabel("MOVER +");
209
210 //Botao de aumentar ou diminuir multiplicador dos valores de Z:
211     cp5.addNumberbox("zJogValue")
212         .setPosition(1554, 875)
213         .setSize(67, 45)
214         .setRange(0, 50)
215         .setFont(font3)
216         .setMultiplier(0.1)
217         .setValue(1)
218         .setDirection(Controller.HORIZONTAL) //Pode-se mudar a direcao
219         .setCaptionLabel("");
220
221 //Criacao de caixas de texto para alterar parametros de posicoes
222     //Para o eixo X:
223     cp5.addTextfield("xTextfield")
224         .setColorBackground(#042d4f)
225         .setPosition(100, 290)
226         .setSize(110, 60)
227         .setFont(font)
228         .setColor(255)
229         .setCaptionLabel("");
230
231 //Para o eixo Y:
232     cp5.addTextfield("yTextfield")
233         .setColorBackground(#042d4f)

```

```

234     .setPosition(282, 290)
235     .setSize(110, 60)
236     .setFont(font)
237     .setColor(255)
238     .setCaptionLabel("");
239
240 //Para o eixo Z:
241 cp5.addTextfield("zTextfield")
242     .setColorBackground(#042d4f)
243     .setPosition(464, 290)
244     .setSize(110, 60)
245     .setFont(font)
246     .setColor(255)
247     .setCaptionLabel("");
248
249 //Criacao dos botoes para alterar parametros do robo:
250 //Botao de mover o robo para a posicao desejada:
251 cp5.addButton("move")
252     .setPosition(100, 460)
253     .setColorForeground(#0270ca)
254     .setSize(475, 45)
255     .setFont(font3)
256     .setCaptionLabel("MOVER O ROBO PARA A POSICAO");
257
258 //Botao de salvar a posicao desejada do robo:
259 cp5.addButton("savePosition")
260     .setPosition(100, 660)
261     .setColorForeground(#0270ca)
262     .setSize(245, 50)
263     .setFont(font3)
264     .setCaptionLabel("SALVAR POSICAO");
265
266 //Botao para executar a posicao desejada do robo:
267 cp5.addButton("run")
268     .setPosition(415, 660)
269     .setColorForeground(#0270ca)
270     .setSize(160, 50)
271     .setFont(font3)
272     .setCaptionLabel("EXECUTAR");
273
274 //Botao para atualizar a GUI de controle do robo (interessante de
    ser acionado apos realizar alteracoes significativas dos
    valores do robo):
275 cp5.addButton("updateSA")
276     .setPosition(415, 725)

```

```

277     .setColorForeground(#0270ca)
278     .setSize(160, 37)
279     .setFont(font3)
280     .setCaptionLabel("ATUALIZAR");
281
282 //Botao para limpar os parametros de posicao salvos do robo:
283 cp5.addButton("clearSteps")
284     .setPosition(100, 725)
285     .setColorForeground(#0270ca)
286     .setSize(245, 37)
287     .setFont(font3)
288     .setCaptionLabel("LIMPAR TUDO");
289
290 //Criacao de botoes deslizantes para definicao de parametros
291     adicionais do robo:
292 //Botao deslizante para alterar a velocidade dos motores de passo:
293 cp5.addSlider("speedSlider")
294     .setPosition(100, 860)
295     .setColorForeground(#0270ca)
296     .setSize(230, 60)
297     .setRange(500, 4000)
298     .setColorLabel(#3269c2)
299     .setFont(font)
300     .setCaptionLabel("");
301
302 //Botao deslizante para alterar a aceleracao dos motores de passo:
303 cp5.addSlider("accelerationSlider")
304     .setPosition(365, 860)
305     .setColorForeground(#0270ca)
306     .setSize(230, 60)
307     .setRange(500, 4000)
308     .setColorLabel(#3269c2)
309     .setFont(font)
310     .setCaptionLabel("");
311
312 //Botao deslizante para alterar os valores de abertura da garra:
313 cp5.addSlider("gripperValue")
314     .setPosition(715, 860)
315     .setColorForeground(#0270ca)
316     .setSize(230, 60)
317     .setRange(0, 100) //Pode-se mudar a direcao do controle para
318     esquerda (0 - FECHADA) ou direita (100 - ABERTA);
319     .setColorLabel(#3269c2)
320     .setFont(font)
321     .setCaptionLabel("");

```

```

320 }
321
322 //Execucao do codigo em 'looping' para o controle da GUI do robo:
323 void draw() { //Desenha-se a area de trabalho da interface grafica
    ;
324   background(255, 255, 255); //'Background' (BG) de cor sOlida
    branca;
325   image(img, 290, 140);
326
327 /*Palheta de cores escolhidas:
328 - Azul Normal = #0270ca;
329 - Azul Escuro = #042d4f; */
330
331 //Definicao da fonte utilizada, suas variacoes e seus tamanhos:
332   PFont medium = createFont("Red Hat Display Medium", 25, true);
333   PFont bold = createFont("Red Hat Display Bold", 25, true);
334   PFont black = createFont("Red Hat Display Black", 25, true);
335
336 //Criacao de caixa para o nome principal "ROBO SCARA":
337   noStroke(); //Desativa a borda da caixa;
338   fill(unhex("FF0270CA")); //Cor da caixa;
339   int rectX = 933; //Posicao X da caixa;
340   int rectY = 59; //Posicao Y da caixa;
341   int rectWidth = 243; //Largura da caixa;
342   int rectHeight = 80; //Altura da caixa;
343   rect(rectX, rectY, rectWidth, rectHeight);
344
345 //Criacao de caixas para detalhe e 'design' dos nomes "CINEMATICA
    DIRETA" e "CINEMATICA INVERSA":
346   fill(unhex("FF0270CA")); //Cor da caixa
347   int AX = 131; //Posicao X da caixa
348   int AY = 202; //Posicao Y da caixa
349   int AWidth = 15; //Largura da caixa
350   int AHeight = 40; //Altura da caixa
351   rect(AX, AY, AWidth, AHeight);
352
353   fill(unhex("FF0270CA")); //Cor da caixa
354   int BX = 1390; //Posicao X da caixa
355   int BY = 202; //Posicao Y da caixa
356   int BWidth = 15; //Largura da caixa
357   int BHeight = 40; //Altura da caixa
358   rect(BX, BY, BWidth, BHeight);
359
360 //Escrevendo o titulo da GUI:
361   textFont(black);

```

```

362 textSize(65);
363 fill(255, 255, 255);
364 text("ROBO SCARA", 730, 120);
365 fill(#042d4f);
366 text("ROBO", 730, 120);
367
368 //Escrevendo os subtítulos "CINEMATICA DIRETA" e "CINEMATICA
    INVERSA":
369 textFont(bold);
370 textSize(35);
371 fill(33);
372 fill(#042d4f);
373 text("CINEMATICA DIRETA", 1410, 235);
374 text("CINEMATICA INVERSA", 150, 235);
375
376 //Escrevendo os nomes "J1", "J2", "J3" e "Z":
377 textFont(bold);
378 textSize(45);
379 text("J1", 1260, 333);
380 text("J2", 1260, 505);
381 text("J3", 1260, 677);
382 text("Z", 1277, 849);
383
384 //Escrevendo o nome "GARRA":
385 textFont(bold);
386 textSize(25);
387 text("J1", 760, 605);
388 text("J2", 1000, 320);
389 text("J3", 1132, 376);
390 text("Z", 821, 261);
391 textSize(21);
392 text("GARRA", 1005, 560);
393
394 //Escrevendo os nomes "VELOCIDADE" e "ACELERACAO":
395 textSize(27);
396 text("VELOCIDADE", 125, 840);
397 text("ACELERACAO", 388, 840);
398
399 //Realizando o preenchimento dos botoes deslizantes (o
    preechimento foi realizado com cor azul claro):
400 //println("PREV: "+accelerationSlider);
401 fill(speedSlider);
402 fill(accelerationSlider);
403 fill(j1Slider);
404 fill(j2Slider);

```

```

405 fill(j3Slider);
406 fill(zSlider);
407 fill(j1JogValue);
408 fill(j2JogValue);
409 fill(j3JogValue);
410 fill(zJogValue);
411 fill(gripperValue);
412
413 //Atualizando os valores do 'Processing' com o Arduino IDE:
414 updateData();
415 //println(data);
416
417 saveStatus=0; //Mantem a variavel "savePosition" 0 ou falsa.
    Quando o botao SALVAR for pressionado ele torna o valor 1, que
    indica armazenar o valor no codigo do Arduino IDE;
418
419 //Se o controle deslizante for movido, calcula-se a nova posicao
    de X, Y e Z com a cinematica direta:
420 if (slider1Previous != j1Slider) {
421
422 //Verifica se o modo cinematica inversa esta ativo, executa a
    cinematica direta somente se o modo cinematica inversa estiver
    desativado:
423     if (activeIK == false) {
424         theta1 = round(cp5.getController("j1Slider").getValue()); //
        Obtem o valor do "j1Slider";
425         theta2 = round(cp5.getController("j2Slider").getValue()); //
        Obtem o valor do "j2Slider";
426         forwardKinematics();
427         myPort.write(data); //Realiza a comunicacao entre o '
        Processing' e o Arduino IDE (atualizando conforme alterado nos
        botoes);
428     }
429 }
430 slider1Previous = j1Slider;
431
432 if (slider2Previous != j2Slider) {
433
434 //Verifica se o modo cinematica inversa esta ativo, executa a
    cinematica direta somente se o modo cinematica inversa estiver
    desativado:
435     if (activeIK == false) {
436         theta1 = round(cp5.getController("j1Slider").getValue()); //
        Obtem o valor do "j1Slider";
437         theta2 = round(cp5.getController("j2Slider").getValue()); //

```

```

438     forwardKinematics();
439     myPort.write(data); //Realiza a comunicacao entre o '
Processing' e o Arduino IDE (atualizando conforme alterado nos
botoes);
440 }
441 }
442 slider2Previous = j2Slider;
443
444 if (slider3Previous != j3Slider) {
445
446 //Verifica se o modo cinematica inversa esta ativo, executa a
cinematica direta somente se o modo cinematica inversa estiver
desativado:
447     if (activeIK == false) {
448         theta1 = round(cp5.getController("j1Slider").getValue()); //
Obtem o valor do "j1Slider";
449         theta2 = round(cp5.getController("j2Slider").getValue()); //
Obtem o valor do "j2Slider";
450         forwardKinematics();
451         myPort.write(data); //Realiza a comunicacao entre o '
Processing' e o Arduino IDE (atualizando conforme alterado nos
botoes);
452     }
453 }
454 slider3Previous = j3Slider;
455
456 //Verifica se o modo cinematica inversa esta ativo, executa a
cinematica direta somente se o modo cinematica inversa estiver
desativado:
457 if (sliderzPrevious != zSlider) {
458     if (activeIK == false) {
459         zP = round(cp5.getController("zSlider").getValue()); //Obtem
o valor do "zslider";
460         myPort.write(data); //Realiza a comunicacao entre o '
Processing' e o Arduino IDE (atualizando conforme alterado nos
botoes);
461     }
462 }
463 sliderzPrevious = zSlider;
464
465 if (gripperValuePrevious != gripperValue) {
466     if (activeIK == false) {
467         gripperAdd = round(cp5.getController("gripperValue").
getValue()); //Obtem o valor do "gripperValue";

```

```

468     gripperValue=grripperAdd+50;
469     updateData();
470     println(data);
471     myPort.write(data); //Realiza a comunicacao entre o '
Processing' e o Arduino IDE (atualizando conforme alterado nos
botoes);
472 }
473 }
474 gripperValuePrevious = gripperValue;
475 activeIK = false; //Desativa a cinematica inversa para que as
instrucoes "if" acima possam ser executadas na proxima
interacao;
476
477 //Criacao de textos presentes na GUI do robo:
478     textFont(bold);
479     textSize(35);
480     fill(33);
481     fill(#042d4f);
482     text("X: ", 100, 410);
483     text(xP, 145, 410);
484     text("Y: ", 282, 410);
485     text(yP, 327, 410);
486     text("Z: ", 464, 410);
487     text(zP, 509, 410);
488     textSize(27);
489     text("GARRA", 782, 840);
490     textFont(medium);
491     textSize(18);
492     fill(33);
493     fill(#042d4f);
494     text("Fechada", 630, 900);
495     text("Aberta", 960, 900);
496     textSize(18);
497
498 //Condicao realizada para realizar a contagem da quantidade de
posicoes que foram salvas:
499 if (positionsCounter >0 ) {
500     textFont(medium);
501     textSize(18);
502     fill(33);
503     fill(#042d4f);
504     text(positions[positionsCounter-1], 100, 595);
505     text("Ultima posicao salva:", 100, 555);
506     somar = +(positionsCounter-1)+1;
507     text(somar, 100, 635);

```

```

508     text("posicoes salvas", 120, 635);
509 } else {
510     textFont(medium);
511     textSize(18);
512     fill(33);
513     fill(#042d4f);
514     text("Ultima posicao salva:", 100, 555);
515     text("Nenhuma posicao salva", 100, 595);
516 }
517 }
518
519 //CINEMATICA DIRETA:
520 //Realizando todos os calculos de cinematica direta conforme suas
521 //formulas pre-estabelecidas:
522 void forwardKinematics() {
523     float theta1F = theta1 * PI / 180; //Graus para radianos;
524     float theta2F = theta2 * PI / 180;
525     xP = round(L1 * cos(theta1F) + L2 * cos(theta1F + theta2F));
526     yP = round(L1 * sin(theta1F) + L2 * sin(theta1F + theta2F));
527 }
528
529 //CINEMATICA INVERSA:
530 void inverseKinematics(float x, float y) {
531     theta2 = acos((sq(x) + sq(y) - sq(L1) - sq(L2)) / (2 * L1 * L2))
532     ;
533     if (x < 0 & y < 0) {
534         theta2 = (-1) * theta2;
535     }
536     theta1 = atan(x / y) - atan((L2 * sin(theta2)) / (L1 + L2 * cos(
537         theta2)));
538     theta2 = (-1) * theta2 * 180 / PI;
539     theta1 = theta1 * 180 / PI;
540
541     //Ajuste de angulos dependendo em qual quadrante esta a coordenada
542     //final (X,Y):
543     if (x >= 0 & y >= 0) { //1 quadrante;
544         theta1 = 90 - theta1;
545     }
546     if (x < 0 & y > 0) { //2 quadrante;
547         theta1 = 90 - theta1;
548     }
549     if (x < 0 & y < 0) { //3 quadrante;
550         theta1 = 270 - theta1;
551         phi = 270 - theta1 - theta2;
552         phi = (-1) * phi;

```

```

549 }
550 if (x > 0 & y < 0) { //4 quadrante;
551     theta1 = -90 - theta1;
552 }
553 if (x < 0 & y == 0) {
554     theta1 = 270 + theta1;
555 }
556
557 //Calcula o angulo "phi" para que a garra fique paralela ao eixo X
    :
558 phi = 90 + theta1 + theta2;
559 phi = (-1) * phi;
560
561 //Ajuste de angulos dependendo em qual quadrante esta a coordenada
    final (X,Y):
562 if (x < 0 & y < 0) { //3 quadrante;
563     phi = 270 - theta1 - theta2;
564 }
565 if (abs(phi) > 165) {
566     phi = 180 + phi;
567 }
568
569 theta1=round(theta1);
570 theta2=round(theta2);
571 phi=round(phi);
572
573 //Realizando a associacao entre os botoes e as variaveis criadas
    para cada movimento:
574 cp5.getController("j1Slider").setValue(theta1);
575 cp5.getController("j2Slider").setValue(theta2);
576 cp5.getController("j3Slider").setValue(phi);
577 cp5.getController("zSlider").setValue(zP);
578 }
579
580 //Funcao criada para controle de eventos:
581 void controlEvent(ControlEvent theEvent) {
582
583     if (theEvent.isController()) {
584         println(theEvent.getController().getName());
585     }
586 }
587
588 //Funcao criada para alterar parametros e valores do eixos X, Y e
    Z para a cinematica inversa:
589 public void xTextfield(String theText) {

```

```

590
591 //Se for inserido um valor no campo de texto, o valor e lido,
    convertemos para inteiro, ativa-se o modo CINEMATICA INVERSA:
592 //Para o eixo X:
593 xP=Integer.parseInt(theText);
594 activeIK = true;
595 inverseKinematics(xP, yP); //Usa-se a cinematica inversa para
    calcular as posicoes J1(theta1), J2(theta2) e J3(phi),
    atualizando-as;
596 //activeIK = false;
597 println("Test; theta1: "+theta1+" theta2: "+theta2);
598 }
599 public void yTextfield(String theText) {
600
601 //Para o eixo Y:
602 yP=Integer.parseInt(theText);
603 activeIK = true;
604 inverseKinematics(xP, yP); //Usa-se a cinematica inversa para
    calcular as posicoes J1(theta1), J2(theta2) e J3(phi),
    atualizando-as;
605 //activeIK = false;
606 }
607 public void zTextfield(String theText) {
608
609 //Para o eixo Z:
610 zP=Integer.parseInt(theText);
611 activeIK = true;
612 inverseKinematics(xP, yP); //Usa-se a cinematica inversa para
    calcular as posicoes J1(theta1), J2(theta2) e J3(phi),
    atualizando-as;
613 }
614
615 //Controle em J1:
616 //Se for diminuido o valor de "j1Slider":
617 public void j1JogMinus() {
618     int a = round(cp5.getController("j1Slider").getValue());
619     a=a-j1JogValue;
620     cp5.getController("j1Slider").setValue(a);
621 }
622
623 //Se for aumentado o valor de "j1Slider":
624 public void j1JogPlus() {
625     int a = round(cp5.getController("j1Slider").getValue());
626     a=a+j1JogValue;
627     cp5.getController("j1Slider").setValue(a);

```

```

628 }
629
630 //Controle em J2:
631 //Se for diminuido o valor de "j2Slider":
632 public void j2JogMinus() {
633     int a = round(cp5.getController("j2Slider").getValue());
634     a=a-j2JogValue;
635     cp5.getController("j2Slider").setValue(a);
636 }
637
638 //Se for aumentado o valor de "j2Slider":
639 public void j2JogPlus() {
640     int a = round(cp5.getController("j2Slider").getValue());
641     a=a+j2JogValue;
642     cp5.getController("j2Slider").setValue(a);
643 }
644
645 //Controle em J3:
646 //Se for diminuido o valor de "j3Slider":
647 public void j3JogMinus() {
648     int a = round(cp5.getController("j3Slider").getValue());
649     a=a-j3JogValue;
650     cp5.getController("j3Slider").setValue(a);
651 }
652
653 //Se for aumentado o valor de "j3Slider":
654 public void j3JogPlus() {
655     int a = round(cp5.getController("j3Slider").getValue());
656     a=a+j3JogValue;
657     cp5.getController("j3Slider").setValue(a);
658 }
659
660 //Controle em Z:
661 //Se for diminuido o valor de "zSlider":
662 public void zJogMinus() {
663     int a = round(cp5.getController("zSlider").getValue());
664     a=a-zJogValue;
665     cp5.getController("zSlider").setValue(a);
666 }
667
668 //Se for aumentado o valor de "zSlider":
669 public void zJogPlus() {
670     int a = round(cp5.getController("zSlider").getValue());
671     a=a+zJogValue;
672 ;

```

```

673 cp5.getController("zSlider").setValue(a);
674 }
675
676 public void move() { //Chama-se a funcao para mover o robo para a
        posicao desejada;
677     myPort.write(data);
678     println(data);
679 }
680
681 //Funcao criada para salvar todas as posicoes desejadas para o
        robo:
682 public void savePosition() {
683
684 //Salve a posicao J1, J2, J3 e Z na matriz:
685     positions[positionsCounter]="J1="+str(round(cp5.getController("
        j1Slider").getValue()))
686     +"; J2=" + str(round(cp5.getController("j2Slider").getValue()
        )
687     +"; J3="+str(round(cp5.getController("j3Slider").getValue()))
688     +"; Z="+str(round(cp5.getController("zSlider").getValue()));
689     positionsCounter++;
690     saveStatus = 1;
691     updateData();
692     myPort.write(data);
693     saveStatus=0;
694 }
695
696 //Funcao criada para executar os parametros salvos do robo:
697 public void run() {
698     if (runStatus == 0) {
699         cp5.getController("run").setCaptionLabel("PARAR"); //Durante a
            execucao aparece o botao de "PARE" que pode ser pressionada
            para encerrar;
700         cp5.getController("run").setColorLabel(#e74c3c); //
            Estabelecida a cor vermelha;
701
702         runStatus = 1;
703     } else if (runStatus == 1) {
704         runStatus = 0;
705         cp5.getController("run").setCaptionLabel("EXECUTAR");//Retorna
            o botao de "EXECUTAR" apos pressionar o botao "PARE";
706         cp5.getController("run").setColorLabel(255);
707     }
708     updateData();
709     myPort.write(data);

```

```

710 }
711
712 //Funcao criada para atualizar os valores da GUI do robo:
713 public void updateSA() {
714     myPort.write(data);
715 }
716
717 //Funcao criada para limpar os valores salvos de movimentacao do
718     robo:
719 public void clearSteps() {
720     saveStatus = 2; //Limpar todas as posicoes e etapas do programa;
721     updateData();
722     myPort.write(data);
723     println("Clear: "+data);
724     positionsCounter=0;
725     saveStatus = 0;
726 }
727 //Funcao criada para atualizar os valores e os mesmos serem
728     enviados ao Arduino IDE, realizando a comunicacao integrada
729     com o 'Processing':
730 public void updateData() {
731     data = str(saveStatus)
732     +"," +str(runStatus)
733     +"," +str(round(cp5.getController("j1Slider").getValue()))
734     +"," +str(round(cp5.getController("j2Slider").getValue()))
735     +"," +str(round(cp5.getController("j3Slider").getValue()))
736     +"," +str(round(cp5.getController("zSlider").getValue()))
737     +"," +str(gripperValue)
738     +"," +str(speedSlider)
739     +"," +str(accelerationSlider);
740 }
741 //Fim do codigo!

```

Código D.2 - *Script* de Processing para interface gráfica de controle do robô

REFERÊNCIAS BIBLIOGRÁFICAS

- ALLEGRO MICROSYSTEMS®. **A4988 DMOS Microstepping Driver with Translator and Overcurrent Protection**. [S.l.], 2024. 20 p. 118
- ANANIAS, E. d. P. **Braço robótico colaborativo de baixo custo**. 155 p. Dissertação (Mestrado em Engenharia Eletrotécnica de Computadores) — Universidade Beira Interior (UBI), Portugal, 2022. 6
- BARNATT, C. **3D Printing**. 3. ed. Wroclaw, Poland: ExplainingTheFuture, 2016. 19, 63, 83
- BARRIA, P.; RIQUELME, M.; REPPICH, H.; CISNAL, A.; FRAILE, J.-C.; PÉREZ-TURIEL, J.; SIERRA, D.; AGUILAR, R.; ANDRADE, A.; ESPINOSA, C. Nuñez. Hand rehabilitation based on the robhand exoskeleton in stroke patients: A case series study. **Frontiers in Robotics and AI**, Frontiers, v. 10, p. 1146018, 2023. 3
- BARRIENTOS, A.; PEÑÍN, L.; BALAGUER, C.; ARACIL, R. **Fundamentos de robótica**. 2. ed. Madrid, ES: McGraw Hill, 2007. 9, 10, 12, 13, 24, 26, 27, 28, 31, 41, 52, 53, 118
- BEER, F. P.; JOHNSTON, E. R.; DEWOLF, J. T.; MAZUREK, D. F. **Mecânica dos Materiais - 8.ed.** Porto Alegre, RS: McGraw Hill Brasil, 2011. 19, 20, 21, 23, 24, 41, 45, 46, 52, 57, 60
- BERRY, M. S.; MARTÍNEZ, B. C. Medicina y robótica. **Revista Médica Clínica las Condes**, Santiago, v. 16, n. 3, p. 157–167, jul 2005. 1
- BISHOP, R. H. **The Mechatronics Handbook**. [S.l.]: CRC Press, 2002. 31, 32, 107
- BOLTON, W. **Instrumentation and Control Systems**. 3. ed. [S.l.]: Newnes, 2021. 31, 33, 109
- BUDYNAS, R. G.; NISBETT, J. K. **Shigley's Mechanical Engineering Design**. [S.l.]: McGraw-Hill Education, 2011. 36, 87, 90, 100
- BUTTERFIELD, A.; SZYMANSKI, J. **A Dictionary of Electronics and Electrical Engineering**. 5. ed. [S.l.]: Oxford University Press, 2018. 33, 109
- CASTRO, A. L. A. d. **Projeto e construção de um robô manipulador SCARA**. Universidade de Brasília (UNB): [s.n.], 2019. Trabalho de Conclusão de Curso (TCC). 5, 6, 35, 36
- CÉSAR, D. R. Robótica livre: Robótica educacional com tecnologias livres. **Fórum Internacional de Software Livre**, v. 1, p. 1–6, 2005. 3, 4, 6

CHRISTIANE, P.-J. **Development of a minimally invasive robotic surgical manipulator**. Dissertação (Masters of Science in Engineering) — Stellenbosch University, Stellenbosch, 2009. 3

CHU, B.; SUNG, Y. W. Development of educational robot platform based on omnidirectional mobile mechanism. **Journal of the Korean Society for Precision Engineering**, Korean Society for Precision Engineering, v. 30, n. 11, p. 1161–1169, 2013. 4, 65

CORKE, P. **Robotics, Vision and Control: Fundamental Algorithms in MATLAB®**. [S.l.]: Springer International Publishing, 2017. 18

CRAIG, J. J. **Introduction to Robotics: Mechanics and Control**. 3. ed. [S.l.]: Prentice Hall, 2005. 10, 18, 27, 28, 30

_____. **Introdução á Robótica**. 3. ed. [S.l.]: Pearson, 2013. 10, 11, 17, 18, 27, 28, 30

D'ABREU, J. a. V. V.; CHELLA, M. Desenvolvimento de ambientes de aprendizagem baseados no uso de dispositivos robóticos. In: **Anais do X Simpósio Brasileiro de Informática na Educação—SBIE99**. Curitiba, PR: [s.n.], 1999. p. 379–386. 1, 142

DARIO, P.; GUGLIELMELLI, E.; ALLOTTA, B.; CARROZZA, M. C. Robotics for medical applications. **IEEE Robotics & Automation Magazine**, IEEE, v. 3, n. 3, p. 44–56, 1996. 3

EDWARDS, M. Robots in industry: An overview. **Applied ergonomics**, Elsevier, v. 15, n. 1, p. 45–53, 1984. 4, 65

EVANS, B. **Practical 3D printers: The science and art of 3D printing**. [S.l.]: Apress, 2012. 19, 63, 79, 80, 84, 143

FILHO, J. A. B. L.; ALMEIDA, W. R. M.; MARTINS, S. G. Development of a multitasking mobile robot for the construction of educational robotics kits. In: IEEE. **2011 International Conference on Electronic Devices, Systems and Applications (ICEDSA)**. [S.l.], 2011. p. 213–21. 4

FRYER, D. M.; MARSHALL, J. C. The motives of jacques de vaucanson. **Technology and Culture**, v. 20, n. 2, p. 257–269, 1979. 1

GANZAROLI, C. A. **Estudo, implementação e otimização de técnicas de controle avançado**. Tese (Doutorado em Engenharia Elétrica) — Universidade Federal de Goiás (UFG), Escola de Engenharia Elétrica, Mecânica e de Computação, Goiânia, Goiás, Brasil, 2023. 127

GASPARETTO, A. et al. A brief history of industrial robotics in the 20th century. **Advances in Historical Studies**, v. 8, p. 24–35, 2019. 4, 5, 15

GATES® SOUTH AMERICA. **Catalogo de correias industriais - PowerGrip™ GT™2**: Gates poly chain. [S.l.], 2020. 122 p. 97

GENEROSO, D. J. Elementos de máquinas. **Instituto Federal de Educação Ciência e Tecnologia de Santa Catarina - Campus de Araranguá**, 2009. 36

GIBSON, I.; ROSEN, D. W.; STUCKER, B. **Additive Manufacturing Technologies: 3D Printing, Rapid Prototyping, and Direct Digital Manufacturing**. 2. ed. [S.l.]: Springer New York, 2015. 19, 77, 80, 85

GRAU, A.; INDRI, M.; BELLO, L. L.; SAUTER, T. Robots in industry: The past, present, and future of a growing collaboration with humans. **IEEE Industrial Electronics Magazine**, IEEE, v. 15, n. 1, p. 50–61, 2020. 4, 42, 142

GROOVER, M. P.; WEISS, M.; NAGEL, R. N. **Industrial Robotics: Technology, Programming, and Application**. 1. ed. [S.l.]: McGraw-Hill Higher Education, 1986. 3, 9, 10, 11, 13, 16, 17, 27, 71

GRUPO SKF®. **Rolamentos de axiais, radiais e lineares de esferas**: Catalogo geral de rolamentos. [S.l.], 2015. 104 p. 99, 100

GUPTA, S. K.; KROVI, V.; SCHLENOFF, C. I. **Recent Advances in Industrial Robotics - Introduction**. Hackensack, NJ: World Scientific Publishing Co., Inc. New Jersey Office, 2021. 3, 16

HELLER, E. **A psicologia das cores: como as cores afetam a emoção e a razão**. [S.l.]: Editora Olhares, 2022. 40, 83, 84

HOROWITZ, P.; HILL, W. **The Art of Electronics**. 3. ed. [S.l.]: Cambridge University Press, 2015. 34

HUGHES, A.; DRURY, B. **Electric Motors and Drives: Fundamentals, Types, and Applications**. 4. ed. [S.l.]: Elsevier Science, 2013. 31, 32, 107

HUGHES, J. M. **Arduino: A Technical Reference: a Handbook for Technicians, Engineers, and Makers**. [S.l.]: O'Reilly Media, 2016. 34, 35, 112, 125

INTERNATIONAL ORGANIZATION FOR STANDARDIZATION. **ISO 281: Mancais de rolamentos — capacidade de carga dinâmica e vida útil estimada**: Vocabulary. [S.l.], 2010. 60 p. 99, 100

_____. **ISO 8373: Robots and Robotic Devices**: Vocabulary. England, 2012. 38 p. 9

JONES, R. M. **Mechanics of composite materials**. [S.l.]: CRC press, 2018. 20, 21, 23, 47, 48

JUNIOR, L.; GUERRA, F.; BRAVO, L.; HERNANDEZ, M.; NETO, O.; MARTINS, P. A low-cost and simple arduino-based educational robotics kit. **Cyber Journals: Multidisciplinary Journals in Science and Technology, Journal of Selected Areas in Robotics and Control (JSRC)**, v. 3, n. 12, p. 1–7, dec 2013. 4, 126

KENJŌ, T.; SUGAWARA, A. **Stepping Motors and Their Microprocessor Controls**. 2. ed. [S.l.]: Oxford University Press, 1994. 31, 32, 109, 115

KOTOV, N. K.; ORLOV, S. V.; TKACH, V. V. Primenenie robototekhnicheskikh sistem v voennoy sfere. boyevoy robot-sobaka dlya okhrany i razvedki. **Vestnik voennogo obrazovaniya**, v. 43, n. 4, p. 127–127, 2023. 2

KURDILA, A. J.; BEN-TZVI, P. Dynamics and control of robotic systems. **Blucher Design Proceedings**, John Wiley & Sons, 2020. 71

KURFESS, T. R. et al. **Robotics and Automation Handbook**. 1. ed. Boca Raton, FL: CRC Press, 2005. 2, 16

LEAL, M. S. **Desenvolvimento de um manipulador SCARA para auxílio à pessoa com deficiência motora**. Universidade de Brasília (UNB): [s.n.], 2022. Trabalho de Conclusão de Curso (TCC). 5, 31, 68

LEPRI, L. J. **Projeto e fabricação de snake robot para operações em ambientes confinados**. 96 p. Dissertação (Mestrado Profissional em Engenharia Mecânica Aeronáutica) — Instituto Tecnológico de Aeronáutica (ITA), São José dos Campos, 2013. 2

LI, W.-B.; CAO, G.-Z.; GUO, X.-Q.; HUANG, S.-D. Development of a 4-dof scara robot with 3r1p for pick-and-place tasks. In: IEEE. **2015 6th International Conference on Power Electronics Systems and Applications (PESA)**. [S.l.], 2015. p. 1–5. 5, 14

LIMA, C. B. d.; VILLAÇA, M. V. M. **Avr E Arduino: Técnicas De Projeto**. 3. ed. [S.l.]: Clube de Autores, 2012. 34, 112, 125

LIPSON, H.; KURMAN, M. **Fabricated: The New World of 3D Printing**. [S.l.]: John Wiley & Sons, 2013. 19, 79, 81, 85, 103

MAKINO, H. Development of the scara. **Journal of Robotics and Mechatronics**, Fuji Technology Press Ltd., v. 26, n. 1, p. 5–8, 2014. 4, 14, 15, 20, 42

MAKINO, H.; MURATA, M.; FURUYA, N. Development of the scara robot. **Seimitsu-Kikai (Journal of the Japan Society for Precision Engineering)**, v. 48, n. 3, p. 378–383, 1982. 5, 13, 15, 16, 19

MARTINS, A. **O que é Robótica**. 1. ed. São Paulo, SP: Editora Brasiliense, 1993. 1, 70

MATARIC, M. J. **The Robotics Primer**. [S.l.]: MIT Press, 2007. 9, 10, 12, 18

MAVROIDIS, C.; FLANZ, J.; DUBOWSKY, S.; DROUET, P.; GOITEIN, M. High performance medical robot requirements and accuracy analysis. **Robotics and Computer-Integrated Manufacturing**, Elsevier, v. 14, n. 5-6, p. 329–338, 1998. 3

MELLO, O. P. M. d. **Desenvolvimento de um robô manipulador Scara**. Universidade de Brasília (UNB): [s.n.], 2016. Trabalho de Conclusão de Curso (TCC). 6, 20, 22, 23, 43, 44, 49, 142

- MERLET, J.-P. A historical perspective of robotics. In: **International Symposium on History of Machines and Mechanisms Proceedings HMM 2000**. Netherlands: Springer, 2000. p. 379–386. 1
- MILUTINOVIĆ, D.; POTKONJAK, V. A new concept of the scara robot. **Robotics and Computer-Integrated Manufacturing**, Elsevier, v. 7, n. 3-4, p. 337–343, 1990. 4
- MINSKY, M. Steps toward artificial intelligence. **Proceedings of the IRE**, IEEE, v. 49, n. 1, p. 8–30, 1961. 2
- MONK, S. **Programming Arduino: Getting Started with Sketches**. [S.l.]: McGraw-Hill Education, 2016. 34, 35, 112, 125
- MORTIMER, J.; ROOKS, B. **The international robot industry report**. [S.l.]: Springer Science & Business Media, 2013. 1
- MOTT, R. L.; VAVREK, E. M.; WANG, J. **Machine Elements in Mechanical Design**. 5. ed. [S.l.]: Pearson Education, 2017. 35, 36, 88, 90
- MUZAN, I. W.; FAISAL, T.; AL-ASSADI, H. M. A. A.; IWAN, M. Implementation of industrial robot for painting applications. **Procedia Engineering**, Elsevier, v. 41, p. 1329–1335, 2012. 3
- NEDELKOVSKI, D. **SCARA Robot 3D Printed**. How To Mechatronics: [s.n.], 2020. Disponível em: <<<https://howtomechatronics.com/projects/scara-robot-how-to-build-your-own-arduino-based-robot/>>>. Acesso em: 15 de março de 2023. 75, 76, 78, 98
- NEOYAMA®. **Datasheet de Produto: Motores de Passo**: Funcionamento, tipos e especificações de motores de passo. [S.l.], 2023. 13 p. 117, 118
- NETO, J. B. **Mecânica: Newtoniana, Lagrangiana e Hamiltoniana, 2ª Edição**. 2. ed. São Paulo, SP: Editora Livraria da Física, 2013. 25, 26
- NIKU, S. B. **Introduction to Robotics: Analysis, Control, Applications**. [S.l.]: Wiley, 2010. 9, 10, 12, 14, 17, 18, 24, 25, 27, 28, 30, 32, 33, 66, 68, 115
- NKOMO, M.; COLLIER, M. A color-sorting scara robotic arm. In: IEEE. **2012 2nd International Conference on Consumer Electronics, Communications and Networks (CECNet)**. [S.l.], 2012. p. 763–768. 6, 17, 69, 120
- OLIER, I. A.; SÁNCHEZ, O. F. A.; BELLO, J. H. Una introducción a la robótica industrial. **Ciencia e Ingeniería Neogranadina**, Universidad Militar Nueva Granada, v. 8, n. 1, p. 10, 1999. 2
- PEREIRA, E. C. **Desenvolvimento de um Robô Manipulador SCARA**. Instituto Federal do Espírito Santo (IFES): [s.n.], 2023. Trabalho de Conclusão de Curso (TCC). 6, 70

- PEREIRA, P. Z.; SCHERER, F. d. V.; TEIXEIRA, F. G. c.; SILVA, R. P. d.; SILVA, T. L. K. d.; CATTANI, A. Possibilidades de uso da matriz morfológica no processo de geração de alternativas em design. **Blucher Design Proceedings**, Congresso Brasileiro de Pesquisa e Desenvolvimento em Design, p. 1–12, out 2014. 64
- PERIM, V. G. d. C. F.; NASCIMENTO, J. N. R. d. **Microcontroladores e Microprocessadores**. Londrina, PR: Editora e Distribuidora Educacional S.A, 2017. 67, 108, 112
- PFEIFFER, F. **Mechanical System Design**. [S.l.]: Springer Berlin Heidelberg, 2008. 35, 36, 88, 89
- RASHID, M. H. **Power Electronics: Devices, Circuits, and Applications**. 4. ed. [S.l.]: Pearson, 2017. 33, 34, 110
- ROCHA, F. A. S. **Comparação entre dinâmicas de controles em malha aberta e em malha fechada de um manipulador prismático bidimensional**. Universidade Federal de Ouro Preto (UFOP): [s.n.], 2016. Trabalho de Conclusão de Curso (TCC). 33, 66, 127
- SACHYANI, E. K.; KAMYSHNY, A.; TOTARO, M.; BECCAI, L.; MAGDASSI, S. 3d printing materials for soft robotics. **Advanced Materials**, Wiley Online Library, v. 33, n. 19, p. 2003387, 2021. 19, 62
- SANCHEZ-MARTÍN, F. M.; MILLÁN, R. F.; SALVADOR, B. J.; PALOU, R. J.; RODRÍGUEZ, E. F.; ESQUENA, F. S.; VILLAVICENCIO, M. H. Historia de la robótica: de arquitas de tarento al robot da vinci (parte i). **Actas urológicas españolas**, v. 31, n. 2, p. 69–76, 2007. 1
- SCHÖN, C. G. Mecânica dos materiais. **Fundamentos e Tecnologia do**, 2013. 20, 22, 24, 46, 49
- SEDRA, A. S.; SMITH, K. C. **Microelectronic Circuits**. 7. ed. [S.l.]: Oxford University Press, 2014. 33, 34, 111
- SHARIATEE, M.; AKBARZADEH, A.; MOUSAVI, A.; ALIMARDANI, S. Design of an economical scara robot for industrial applications. In: IEEE. **2014 Second RSI/ISM International Conference on Robotics and Mechatronics (ICRoM)**. [S.l.], 2014. p. 534–539. 5, 14, 20
- SHIGLEY, J. E.; MISCHKE, C. R. **Mechanical Engineering Design**. 2. ed. [S.l.]: McGraw-Hill, 1996. 36, 99
- SICILIANO, B.; SCIavicCO, L.; VILLANI, L.; ORIOLO, G. **Robotics: Modeling, Planning and Control**. London, UK: Springer, 2009. 10, 11, 13, 18, 70, 72, 120
- SMITH, W. F.; HASHEMI, J. **Foundations of Materials Science and Engineering**. 4. ed. [S.l.]: McGraw-Hill, 2006. 35, 36

SPONG, M. W.; HUTCHINSON, S.; VIDYASAGAR, M. **Robot Modeling and Control**. 1. ed. [S.l.]: John Wiley & Sons, 2006. 28, 30

SUMMERS, M. Robot capability test and development of industrial robot positioning system for the aerospace industry. **SAE Transactions**, JSTOR, p. 1108–1118, 2005. 3

TÁRTARI, A. H.; POSTAL, A.; CASTRO, J. P. Montando um robô de baixo custo. In: **Anais do IV EPAC Encontro Paranaense de Computação**. [S.l.: s.n.], 2011. p. 90–99. 6

THRUN, S. et al. Stanley: The robot that won the darpa grand challenge. **Journal of Field Robotics**, Wiley Online Library, v. 23, n. 9, p. 661–692, 2006. 2

TURING, A. M. Computing machinery and intelligence. **Mind**, v. 59, n. 236, p. 433–460, oct 1950. 2

TZAFESTAS, S. G. **Mobile Robots: General Concepts. Introduction to Mobile Robot Control**. 1. ed. Atenas: Elsevier Inc, 2014. 1, 16, 42

USATEGUI, J. M. A.; LEÓN, J. N. S. de. **Guia Fácil de Robótica**. Madrid: Paraninfo, 1986. 2, 66, 107

VEIGA, E. F.; ARAÚJO, W. M.; JÚNIOR, C. R. S. Projeto de um robô de baixo custo para utilização como ferramenta de robótica educativa para escolas públicas. In: **Mostra Nacional de Robótica MNR**. Inhumas, GO: [s.n.], 2011. 4

YANG, G.-Z.; BELLINGHAM, J.; DUPONT, P.; FISCHER, P.; FLORIDI, L.; FULL, R.; JACOBSTEIN, N.; KUMAR, V.; MCNUTT, M.; MERRIFIELD, R.; NELSON, B.; SCASSELLATI, B.; TADDEO, M.; TAYLOR, R.; VELOSO, M.; WANG, Z.; WOOD, R. The grand challenges of science robotics. **Science Robotics**, v. 3, n. 14, p. eaar7650, jan 2018. 2, 15

ZILLI, S. d. R. et al. A robótica educacional no ensino fundamental: Perspectivas e prática. **Robotics and Computer-Integrated Manufacturing**, Florianópolis, SC, 2004. 4