

Sistema FIEB



PELO FUTURO DA INOVAÇÃO

CENTRO UNIVERSITÁRIO SENAI CIMATEC

Pós-Graduação em Robótica e Sistemas Autônomos

Relatório Final do Programa de Formação em Robótica e Sistemas Autônomos

Apresentada por:

Jéssica Lima Motta

Dezembro de 2020

Jéssica Lima Motta

**Relatório Final do Programa de Formação em
Robótica e Sistemas Autônomos**

Salvador

Centro Universitário SENAI CIMATEC
2020

Agradecimentos

Agradeço ao CCSA (Centro de Competência em Sistemas Autônomos) e a todos os seus pesquisadores, por todo o suporte técnico dado durante o curso. Ao SENAI CIMATEC, por oportunizar e ser o principal fomentador do programa. E ao professores do curso de pós-graduação pela excelente didática no ensino das disciplinas.

Salvador, Brasil

19 de Dezembro de 2020

Jéssica Lima Motta

Resumo

Este relatório tem por finalidade trazer os resultados obtidos, através dos projetos e estudos realizados durante o curso de formação em Robótica e Sistemas Autônomos. E demonstrar o quanto valoroso, desafiador e gratificante foi esse período. Os conhecimentos derivados dessas atividades proporcionaram a formação de uma Especialista em Robótica e Sistemas Autônomos, com entendimento sobre as ferramentas utilizadas para modelagem, simulação e construção real desses sistemas, sobre como os estudos estatísticos são aplicados para fazer análise dos projetos, e saber elaborar o planejamento, direcionar a execução e entregar os resultados aos clientes dos projetos propostos.

Palavras-chave: Robótica, Sistemas Autônomos, Projetos, Inovação, ROS

Abstract

This report aims to bring the results obtained, through the projects and studies carried out during the training course in Robotics and Autonomous Systems. And demonstrate how valuable, challenging and rewarding that period was. The knowledge derived from these activities provided the formation of a Specialist in Robotics and Autonomous Systems, with an understanding of the tools used for modeling, simulation and real construction of these systems, about how statistical studies are applied to analyze projects, and to know how to elaborate the project. planning, direct the execution and deliver the results to the clients of the proposed projects.

Keywords: Robotics, Autonomous System, Projects, Innovation, ROS

Sumário

1	Introdução	1
1.1	Objetivos	1
1.1.1	Objetivos Específicos	1
1.2	Justificativa	1
1.3	Organização do documento	1
2	Metodologia	3
3	Métodos e Resultados	5
3.1	Desafio 1.0	5
3.2	Relatório Parcial do Manipulador TIMON-HM- Desafio 2.0	6
3.3	Resultado do Manipulador Robótico JeRoTIMON- Desafio 2.2	6
3.4	TIMON- HM- Desafio 2.5	7
3.5	UGV SACI: Integrado com Detecção Visual e Manipulador- Desafio 3.0	9
3.6	Analise estatística R&R da simulação do robô Darwin OP	11
3.7	Planejamento de Experimentos (DOE) -Helicóptero de Papel (TIMON-HM)	11
3.8	Resultado do Artigo Manipulador Robótico TIMON-HM- Evento SAPCT 2020	11
3.9	Resultado do Artigo publicado TRIS: Thermal Remote Identification System of Feverish People- Evento SIINTEC 2020	12
4	Conclusão	13
A	Manipulador Robótico TIMON-HM	14
B	Manipulador Robótico JeRoTIMON	101
C	Read me do Desafio 2.5	236
D	UGV SACI: Integrado com Detecção Visual e Manipulador	242
E	Analise estatística R&R da simulação do robô Darwin OP	318
F	Planejamento de Experimentos (DOE) -Helicóptero de Papel (TIMON-HM)	328
G	Artigo Manipulador Robótico TIMON-HM- Evento SAPCT 2020	347
H	Certificado de Participação do Evento SAPCT 2020	351
I	Artigo publicado TRIS: Thermal Remote Identification System of Feverish People- Evento SIINTEC 2020	353
J	Certificado de Participação do Evento SIINTEC 2020	362

Lista de Figuras

2.1	Metodologia do Programa de Formação em Robótica e Sistemas Autônomos	4
3.1	Área externa do CIMATEC 3 e 4, ambiente de simulação do <i>Gazebo</i>	5
3.2	Realização do desafio no ambiente de simulação do <i>Gazebo</i>	6
3.3	Realização do desafio no ambiente real	7
3.4	Simulação do Desafio 2.5- Marcha	8
3.5	Simulação do Desafio 2.5- Revezamento	8
3.6	Ambiente entre os prédios do CIMATEC 3 e 4 na simulação do <i>Gazebo</i> . .	9
3.7	Saci modelado para simulação no <i>Gazebo</i>	10
3.8	Modelo Real do Saci, com a identificação dos equipamentos integrados que foram utilizados	10

Introdução

Neste relatório serão expostos os objetivos, a justificativa, os conhecimentos absorvidos durante o período do curso de formação bem como citados os trabalhos realizados, quais materiais e métodos empregados para realização de cada projeto, e os resultados alcançados.

1.1 *Objetivos*

O relatório tem como objetivo agrupar todos os trabalhos desenvolvidos durante o período do programa de formação em Robótica e Sistemas Autônomos, mostrar os conhecimentos adquiridos e como foi estruturado o curso. Além de ressaltar os resultados gerados de cada etapa deste.

1.1.1 *Objetivos Específicos*

Tem também como objetivo demonstrar o valor do curso na formação profissional. Assim como a estrutura deste permitiu a formação de uma especialista em Robótica e Sistemas Autônomos, no desenvolvimento de competências e habilidades, fundamentais nos projetos realizados nestas áreas em questão. Aliando durante o programa a teoria à prática.

1.2 *Justificativa*

Esse relatório tem por finalidade reunir todos os trabalhos desenvolvidos, e certificados recebidos, para mostrar como a partir deles os conhecimentos puderam ser adquiridos e aprimorados durante o curso.

1.3 *Organização do documento*

Este documento apresenta 4 capítulos e está estruturado da seguinte forma:

- **Capítulo 1 - Introdução:** Neste capítulo estão descritos os objetivos gerais e

específicos, a justificativa e como está organizado este relatório;

- **Capítulo 2 - Materiais e Métodos:** Estão descritos os materiais e métodos utilizados em cada projeto;
- **Capítulo 3 - Resultados:** Foram apresentados os resultados obtidos em cada projeto realizado durante o curso de formação;
- **Capítulo 4 - Conclusão:** Apresenta as conclusões em relação ao programa de formação.

Metodologia

Um dos pontos característico do programa de Formação em Robótica é a sua metodologia, onde buscará a aprendizagem ativa do estudante, com a construção dos seus conhecimentos, complementando as aulas expositivas com atividades e dinâmicas de grupo, elaboração e apresentação de trabalhos e pesquisas, emprego de meios audiovisuais, estudos individualizados, pesquisa de artigos técnicos e científicos, entre outros condicionantes ao programa. A metodologia em si é um caminho para o sucesso na formação dos estudantes, pois esta convergência baseia-se em 5 pontos principais:

1. **Criatividade:** espaço que estimula a criatividade, possibilitando interação e acesso à diferentes tecnologias.
2. **Engajamento:** atividades práticas de aprendizado aumentam os níveis de concentração
3. **Programação:** a inteligência artificial se torna cada vez mais presente nas escolas e escritórios
4. **Trabalho em equipe:** robótica incorpora uma gama de habilidades e promove um ambiente de aprendizagem para pessoas com diferentes talentos
5. **Diversão:** aprender sobre robótica deve ser divertido, e a medida que os estudantes continuem melhorando sua interação com eles, isso aumenta mais ainda o nível de diversão

Basicamente o caminho para o sucesso compreende em 4 fases distintas, demonstradas na Figura 2.1:

Assimilação: desenvolver habilidades de codificação e lógica;

Simulação: testar as missões de um robô de forma eficiente;

Integração: garantir informações sobre o ambiente e o robô;

Criação: elaborar um projeto aplicado a tecnologia.

As três primeiras fases são compreendidas em 6 meses, ficando a fase Criação com 6 meses para finalizar o programa. Entre cada fase, desafios são lançados para que o estudante

obtenha maior sucesso na assimilação dos conceitos ministrados, para a última fase um projeto final é lançado para que o estudante possa realizar a demonstração de seu sistema para os avaliadores. Bom salientar que durante as fases, temas serão tratados e discutidos de forma expositiva e prática.

Figura 2.1: Metodologia do Programa de Formação em Robótica e Sistemas Autônomos



Com uma abordagem inovadora, o programa tenta realçar a busca por um aprendizado mais real e excitante para isso o conceito de professor facilitador que estimule a experiência nas várias tecnologias se faz necessário. Isso promoverá vários feedbacks mais intensos aos estudantes. Em resumo o professor será o facilitador de uma experiência de aprendizado, criando recursos e experiência para a formação do aprendiz.

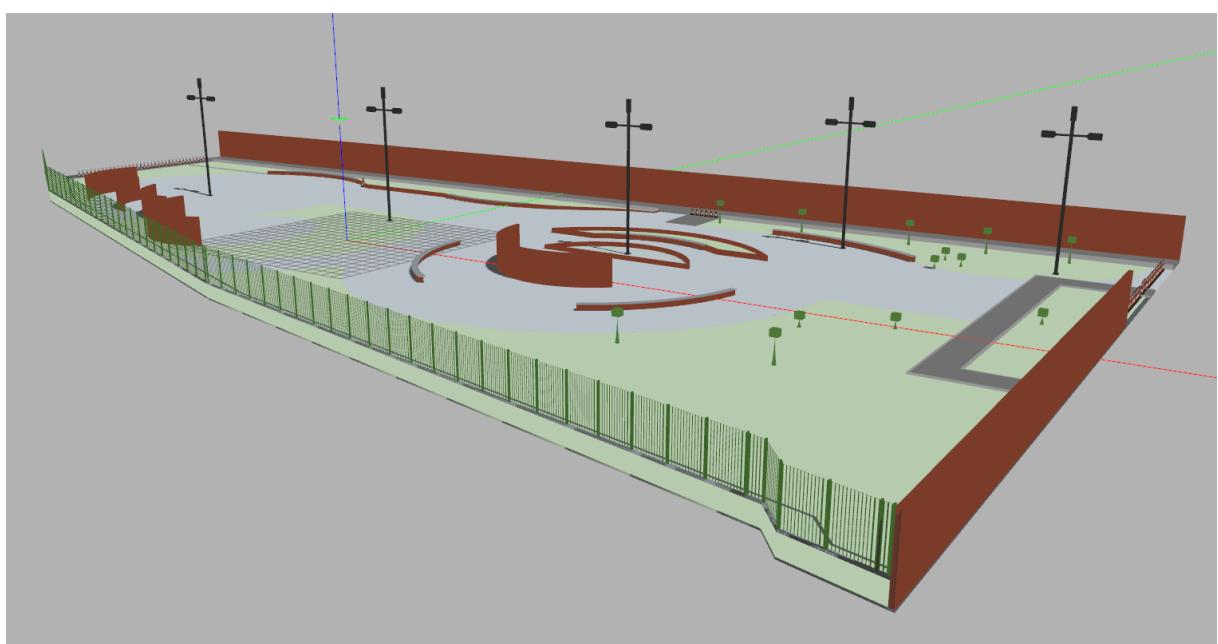
Métodos e Resultados

Nesta seção serão demonstrados os resultados do trabalhos realizados durante o período do curso de formação em Robótica e Sistemas Autônomos, identificando nos Apêndices os relatórios, estudos, artigos publicados e certificados obtidos. Nos textos são indicados os repositórios onde é possível verificar os códigos, os pré-requisitos para funcionar o pacote do projeto e o tutorial para realizar a simulação ou o modelo real.

3.1 Desafio 1.0

Este desafio foi feito individualmente, e ele foi realizado com o objetivo de programar o robô da *Clearpath Robotics- Husky*, no ambiente de simulação do *Gazebo*. A área de operação desse robô foi a área externa entre os prédios do CIMATEC 3 e 4, mostrada na Figura 3.1. O *Husky* tinha como missão explorar esse ambiente externo a procura de uma bola amarela, e ao identificar esta ele deveria ir até ela e parar de frente para mesma informando que a missão foi completada. Como nesse desafio foi realizada apenas a simulação, apenas o computador foi utilizado.

Figura 3.1: Área externa do CIMATEC 3 e 4, ambiente de simulação do *Gazebo*

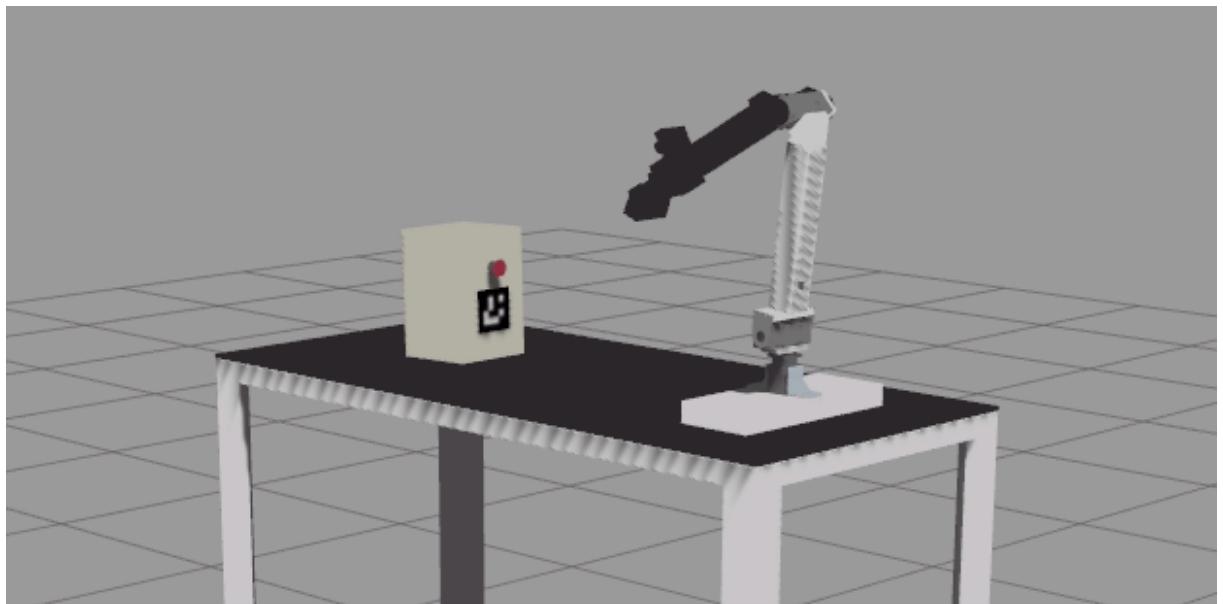


Fonte: Grupo de Formação em Robótica e Sistemas Autônomos

3.2 Relatório Parcial do Manipulador TIMON-HM- Desafio 2.0

O segundo desafio foi realizado em grupo, onde um manipulador deveria ser concebido desde sua fase inicial modelando toda sua estrutura e posteriormente realizada a simulação deste no *Gazebo*, com a missão da câmera integrada ao manipulador identificar o *tag ArUco* na caixa e pressionar o botão. Este desafio também foi realizada apenas a simulação visto na Figura 3.2. No Apêndice A encontra-se o relatório elaborado para esse desafio, com todas as especificações do mesmo, o estado da arte, metodologia e materiais utilizados.

Figura 3.2: Realização do desafio no ambiente de simulação do *Gazebo*



Fonte: Grupo de Formação em Robótica e Sistemas Autônomos

3.3 Resultado do Manipulador Robótico JeRoTIMON- Desafio 2.2

Este desafio foi para construir o modelo real do manipulador JeRoTIMON modelado e simulado no desafio 3.2, realizado em grupo, onde o objetivo era o mesmo, reconhecer a *tag ArUco* na caixa e pressionar o botão, só que dessa vez no ambiente real. Os materiais utilizados nesse desafio foram perfis de alumínio, motores *Dynamixel*, câmera RGB modelo *Teledyne Genie Nano C2590*, peças modeladas no *OnShape* e impressas em *ABS* por uma impressora 3D, conexões para alimentação e para comunicação. Na Figura 3.3 é demonstrado o manipulador JeRoTIMON real e a caixa, e o mesmo executando a missão de pressionar o botão. No Apêndice B está disponível o relatório gerado para esse projeto.

Figura 3.3: Realização do desafio no ambiente real

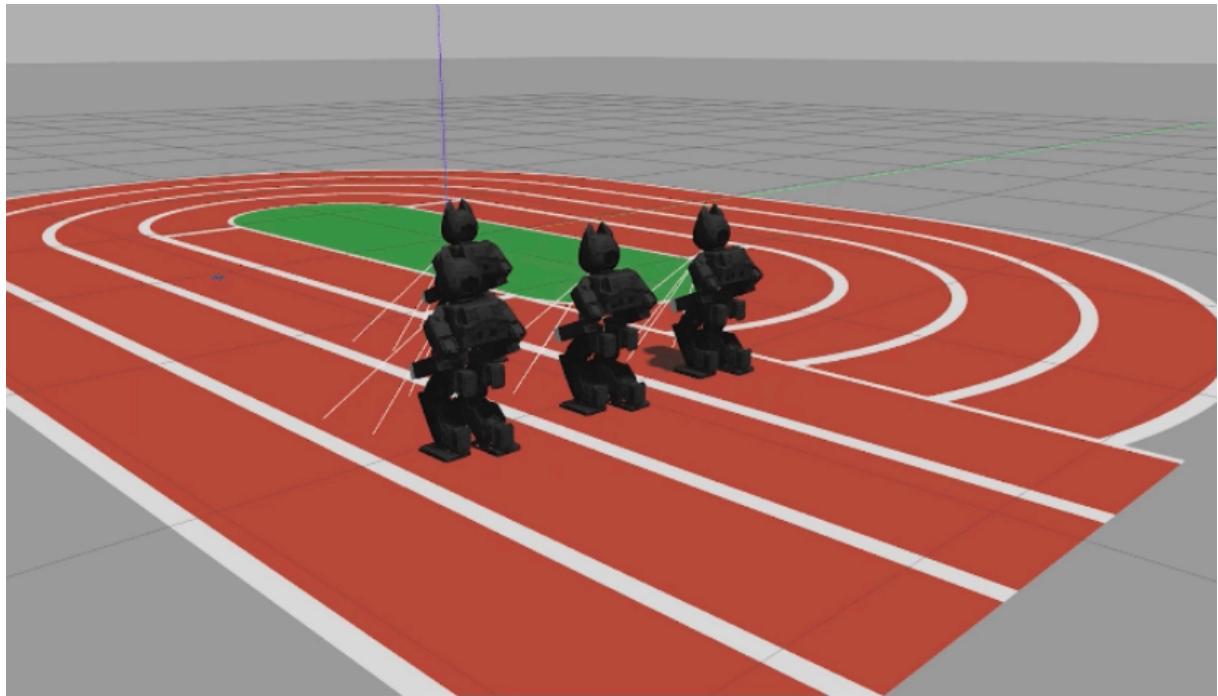


Fonte: Grupo de Formação em Robótica e Sistemas Autônomos

3.4 TIMON- HM- Desafio 2.5

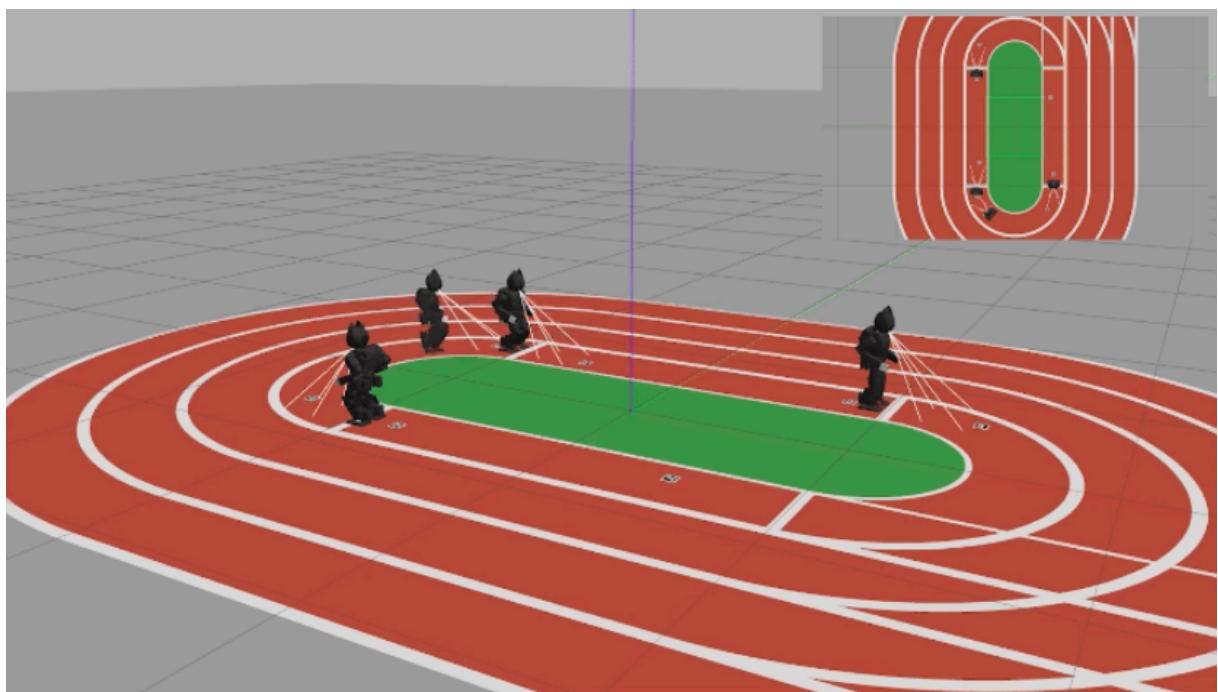
O desafio 2.5 foi realizado em equipe, a mesma do desafio 3.2, onde o robô programado foi o *Darwin-OP* e este deveria realizar duas missões, a primeira, é a marcha, onde quatro robôs *Darwin-OP* deveriam andar de forma sincronizada de um ponto a outro da pista de corrida. E a segunda missão foi realizada a programação para que os quatro robôs realizassem a corrida com revezamento, onde cada robô está posicionado numa parte específica da pista de corrida e ao chegar próximo um do outro eles mantém por um período a movimentação sincronizada depois o anterior para e o outro segue, igualmente a uma corrida com revezamento real. Na Figura 3.4 tem-se os robôs executando a programação da marcha, e na Figura 3.5 tem-se os robôs executando o código do revezamento. No Apêndice C está o *Read me* do repositório.

Figura 3.4: Simulação do Desafio 2.5- Marcha



Fonte: Grupo de Formação em Robótica e Sistemas Autônomos

Figura 3.5: Simulação do Desafio 2.5- Revezamento



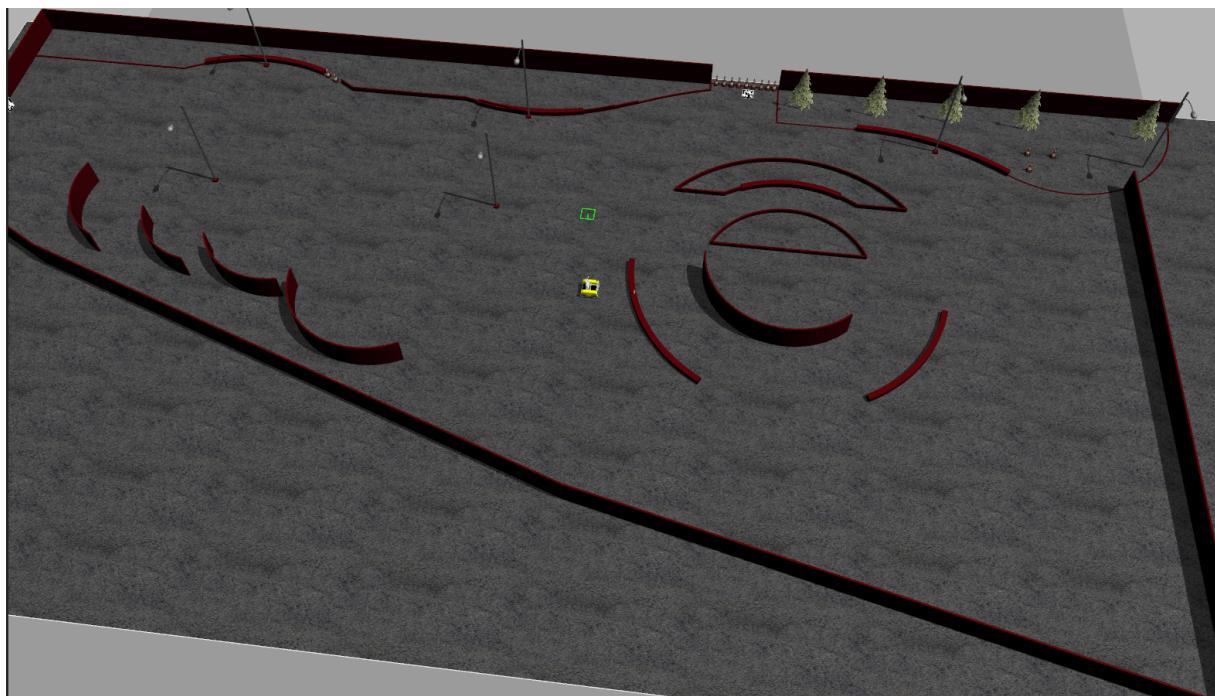
Fonte: Grupo de Formação em Robótica e Sistemas Autônomos

3.5 UGV SACI: Integrado com Detecção Visual e Manipulador- Desafio 3.0

Neste desafio foi desenvolvido o Saci, que integra o veículo autônomo da *Clearpath Robotics Warthog* equipado com sensores (câmeras, LiDAR e GPS) e o manipulador robótico JeRoTIMON, com o propósito de transformá-lo em um robô autônomo. Este foi construído com o intuito de que o mesmo tivesse navegação autônoma para realizar investigação em ambiente externo e construir um mapa deste ambiente, detectasse a "bomba" escondida, e realizasse o desarme da bomba através do manipulador. Esse projeto foi desenvolvido em duas etapas, a de simulação, onde foram utilizados o software *Gazebo* e a ferramenta de visualização *Rviz*, e para configuração do pacote de parâmetros do manipulador foi utilizado *MoveIt*. E em paralelo foi desenvolvido este robô em sua versão real, onde foi possível realizar testes e verificar seu desempenho em campo.

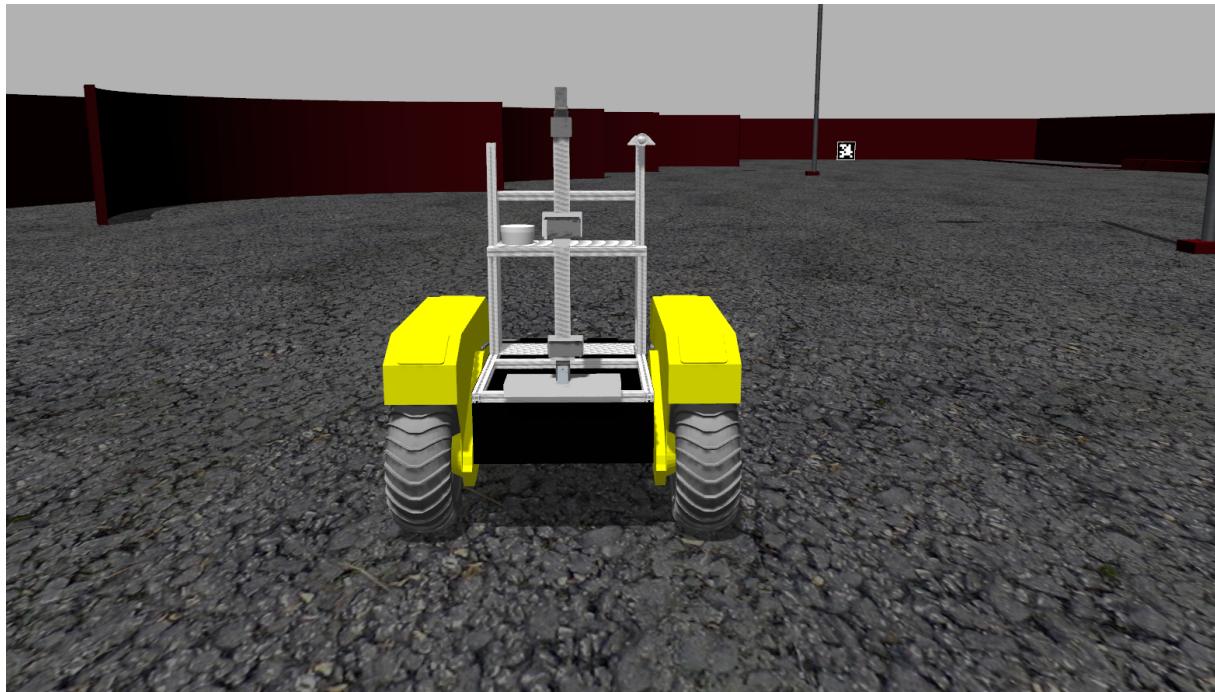
Como resultado obtido desse projeto é possível ver nas Figuras 3.6, 3.7 e 3.8, respectivamente, o ambiente externo modelado na simulação, o robô no ambiente de simulação e este no ambiente real. No Apêndice D está inserido o relatório gerado para esse projeto.

Figura 3.6: Ambiente entre os prédios do CIMATEC 3 e 4 na simulação do *Gazebo*



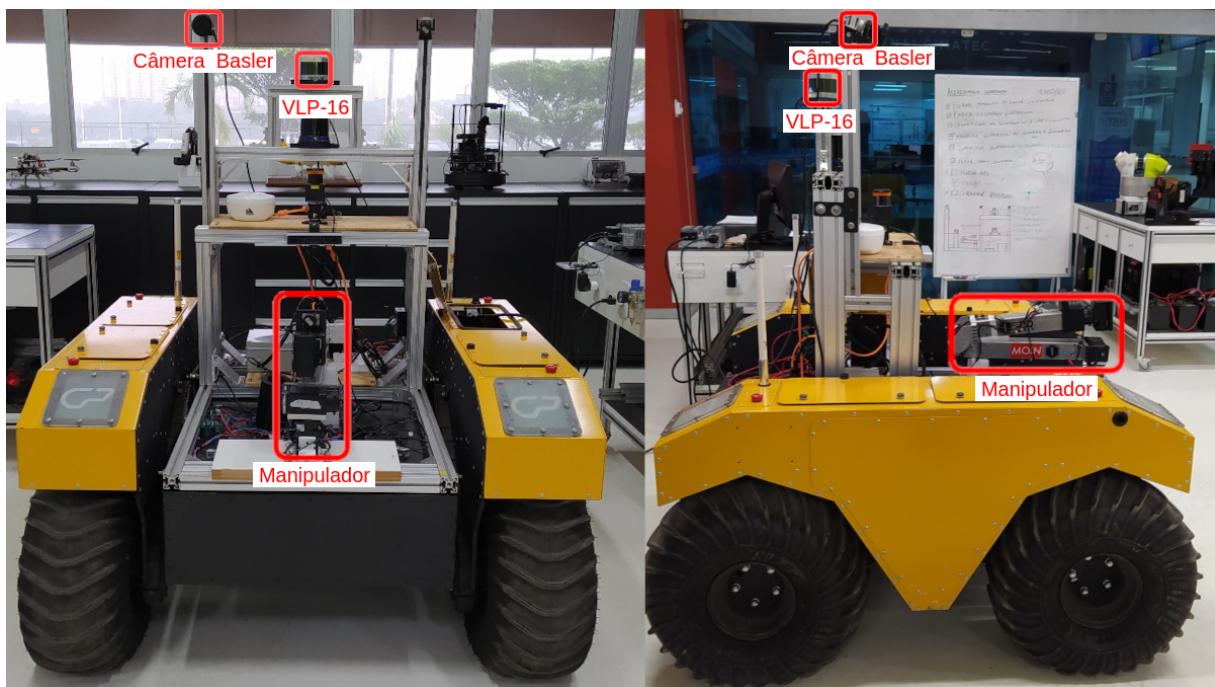
Fonte: Grupo de Formação em Robótica e Sistemas Autônomos

Figura 3.7: Saci modelado para simulação no *Gazebo*



Fonte: Grupo de Formação em Robótica e Sistemas Autônomos

Figura 3.8: Modelo Real do Saci, com a identificação dos equipamentos integrados que foram utilizados



Fonte: Grupo de Formação em Robótica e Sistemas Autônomos

3.6 Análise estatística R&R da simulação do robô Darwin OP

Teve como objetivo analisar o sistema de medição dos dados coletados durante os testes realizados nas etapas: de marcha e de revezamento do Desafio 2.5 (??), utilizando o método de análise de variância (ANOVA). Nessa análise foi possível aplicar os conhecimentos obtidos em estatística, utilizando a ferramenta e linguagem de programação *R* em um projeto realizado durante o curso, a fim de verificar o desempenho desse projeto, exemplo, a análise de precisão produzida através do estudo R&R (Repetibilidade e Reprodutibilidade). Foram utilizadas apenas as ferramentas de simulação e para realização do estudo estatístico. O resultado proveniente deste estudo está descrito no documento Avaliação do Sistema de Medição- TIMON 2.5 no Apêndice E. Onde estão expostos dos dados coletados e a interpretação dos resultados obtidos.

3.7 Planejamento de Experimentos (DOE) -Helicóptero de Papel (TIMON-HM)

Esse experimento teve como objetivo aplicar os conceitos de Planejamento de Experimento- *DOE*, a um modelo de helicóptero de papel. O propósito principal foi identificar quais são os fatores que mais influenciam seu tempo de voo e como estas variáveis podem melhorar o seu desempenho. Durante o processo, foi utilizado um modelo de helicóptero em papel onde foi medido o seu tempo de voo em duas alturas diferentes, além disto, foram adicionados adesivos e um clipe em sua estrutura a fim de verificar a influência da variação destes parâmetros no resultado final. Esse estudo proporcionou a aplicação do aprendizado adquirido ao uso da ferramenta e linguagem R usada para manipulação, análise e visualização de dados, e dos conhecimentos de Estatística. Esse estudo resultou no relatório Planejamento de Experimentos (DOE) -Helicóptero de Papel (TIMON-HM) disponível no Apêndice F, onde neste consta informações sobre o que é o *DOE*, como e qual o motivo da escolha dos fatores de influência e os níveis e os resultados obtidos.

3.8 Resultado do Artigo Manipulador Robótico TIMON-HM- Evento SAPCT 2020

Para o evento V Seminário de Avaliação de Pesquisa Científica e Tecnológica (SAPCT) e IV Workshop de Integração e Capacitação em Processamento de Alto Desempenho (ICPAD), foi realizado o artigo Projeto e Simulação de um Manipulador Robótico com 5 Graus de Liberdade e Sistema de Visão Integrado, com base no projeto do manipulador robótico TIMON-HM, este artigo consta no Apêndice G, bem como o certificado de participação

neste evento no Apêndice [H](#).

3.9 *Resultado do Artigo publicado TRIS: Thermal Remote Identification System of Feverish People- Evento SIINTEC 2020*

Esse artigo, que possui também um sistema real de mesmo nome (*TRIS*), foi modelado a partir da necessidade exposta pela pandemia do COVID-19, para identificar pessoas foram usadas câmeras (RGB e Infravermelho), um computador para utilizar uma rede neural, e que identificasse pessoas com temperatura acima de 37,8 °C, e informasse que aquela pessoa em questão era objeto de interesse pois estaria com febre, ou estado febril, que é um dos sintomas do COVID-19. Esse sistema foi criado com o propósito de realizar o controle da propagação do vírus. Nesse projeto puderam ser desenvolvidos os conhecimentos de rede neural, interface de sistemas, utilização de câmeras RGB e Infravermelho, e a como acontece a evolução de um projeto. O resultado obtido do projeto do *TRIS* foi o artigo publicado no evento VI International Symposium on Innovation and Technology (SIINTEC) 2020, e posteriormente sua premiação em primeiro lugar dentre os trabalhos apresentados. Este artigo e o certificado de participação nestes evento constam, respectivamente, nos Apêndices [I](#) e [J](#).

Conclusão

De forma geral, o programa de formação proporcionou o desenvolvimento de conhecimentos e habilidades requeridas nas áreas de robótica e sistemas autônomos. Os resultados derivados dos projetos, foram expostos no capítulo 3 onde envolveu um enorme aprendizado de planejamento, execução e entrega de projetos. Este documento mostrou o desenvolvimento de um especialista no curso de formação em Robótica e Sistemas Autônomos, que foi formado com base nas ferramentas utilizadas para modelagem, simulação e construção real desses sistemas, e que são usadas no mundo todo nessa área de Robótica e Sistemas Autônomos, nas linguagens de programações fundamentais como *C++*, *Python* e *R*, sobre como os estudos estatísticos são aplicados para fazer análise dos projetos, e saber elaborar o planejamento, direcionar a execução e entregar os resultados aos clientes dos projetos propostos.

Manipulador Robótico TIMON-HM

MANIPULADOR ROBÓTICO TIMON-HM

Relatório Parcial do Projeto Manipuladores Inteligentes

Autores:

Jéssica Lima Motta
Leonardo Mendes de Souza Lima
Miguel Felipe Nery Vieira
Vinicius José Gomes de Araújo Felismino

Facilitadores:

Lucas Cruz da Silva
Tiago Pereira de Souza
Marco Antonio dos Reis

Salvador
Bahia, Brasil

Abril de 2020

Título: Manipulador Robótico Timon-HM	
PROD. TEC. BIR - 001 / 2020	Versão
Classificação: () Confidencial (X) Restrito () Uso Interno () Público	01

Informações Confidenciais - Informações estratégicas para o BIR e Senai Cimatec.

Seu manuseio é restrito a usuários previamente autorizados pelo Gestor da área.

Informações Restritas - Informação cujo conhecimento, manuseio e controle de acesso devem estar limitados a um grupo restrito de pesquisadores que necessitam utilizá-la para exercer suas atividades profissionais.

Informações de Uso Interno - São informações destinadas à utilização interna por pesquisadores e parceiros.

Informações Públicas - Informações que podem ser distribuídas ao público externo, o que, usualmente, é feito através dos canais apropriados.

Dados Internacionais de Catalogação na Publicação (CIP)

000	Jéssica Lima Motta Leonardo Mendes de Souza Lima Miguel Felipe Nery Vieira Vinicius José Gomes de Araújo Felismino
	Lucas Cruz da Silva Tiago Pereira de Souza Marco Antonio dos Reis
	Manipulador Robótico Timon-HM Salvador Bahia, Brasil Abril de 2020
	Keywords: 1. Manipulator. 2. Simulation. 3. Computer vision. 000

SUMÁRIO EXECUTIVO

O projeto de Manipuladores - Desafio 2, também conhecido como **Timon-HM Manipulator** configura-se: sob o Programa de Formação de Novos Talentos do Serviço Nacional de Aprendizagem Industrial, Departamento Regional da Bahia - Senai/DR/BA, sendo este o principal fomentador do programa.

O projeto foi considerado como início técnico no dia 27 de Fevereiro de 2020.

O prazo de execução planejado foi de 50 dias.

RESUMO

Timon-HM é um manipulador projetado, simulado e construído com o intuito de atender às demandas relacionadas ao reconhecimento de marcadores visuais e acionamento de interruptores, chaves ou botões. Posteriormente, este mesmo manipulador robótico será integrado ao robô *Warthog*, desenvolvido pela *Clearpath Robotics*, com o propósito de realizar a atividade de investigação em ambiente externo, desta vez participando de uma simulação de busca e desarme de bombas. O pacote de simulação do manipulador foi construído através do software *Gazebo* aliado ao *MoveIt* e à ferramenta de visualização *Rviz*, possibilitando assim que as atividades realizadas no mundo real tenham sido previamente testadas em ambiente simulado onde é possível analisar os movimentos e limitações do manipulador.

ABSTRACT

Timon-HM is a manipulator designed, simulated and built in order to meet demands related to the recognition of visual markers and activation of interrupters, switches or buttons. Posteriorly, this manipulator robot arm will be attached in *Warthog*, robot designed by *Clearpath Robotics*, with the purpose of carrying out the research activity in an external environment, this time, integrating a simulation for searching and defusing bombs. The manipulator simulation package was built using software *Gazebo*, allied to *Moveit* and *Rviz* visual tool, allowing that activities practiced in the real world have been previously tested in a computer environment where it is possible to analyze manipulator's movements and limitations.

LISTA DE FIGURAS

Figura 1:	Elos e junta de um manipulador robótico.	14
Figura 2:	Arquitetura geral do sistema.	17
Figura 3:	Configuração D-H do manipulador Timon-HM.	19
Figura 4:	<i>Workspace</i> do manipulador no plano X-Y.	20
Figura 5:	<i>Workspace</i> do manipulador no plano X-Z.	21
Figura 6:	<i>Workspace</i> do manipulador no plano Y-Z.	21
Figura 7:	Estrutura analítica do protótipo.	22
Figura 8:	Fluxograma do sistema de escanamento.	23
Figura 9:	Fluxograma do sistema de planejamento e execução de trajetória.	25
Figura 10:	Modelos simulados do manipulador e do painel elétrico.	27
Figura 11:	Imagen capturada pela câmera RGB.	27
Figura 12:	Pose 1 testada para botão.	29
Figura 13:	Pose 4 testada para botão.	29
Figura 14:	Timon-HM pressionando botão localizado em pose 2.	31
Figura 15:	Timon-HM pressionando botão localizado em pose 6.	31
Figura 16:	Árvore de falhas do sistema.	35
Figura 17:	Coordenadas.	51
Figura 18:	Link 0.	51
Figura 19:	Link 1.	52
Figura 20:	Link 2.	53
Figura 21:	Link 3.	53
Figura 22:	Link 4.	54
Figura 23:	Link 5.	55

LISTA DE TABELAS

Tabela 1:	Especificações técnicas do manipulador Timon-HM.	18
Tabela 2:	Parâmetros D-H para o manipulador Timon-HM.	19
Tabela 3:	Poses testadas para o botão.	30
Tabela 4:	Testes realizados para painel elétrico com orientação vertical.	30
Tabela 5:	Testes realizados para painel elétrico com orientação horizontal.	30
Tabela 6:	<i>FMECA</i> do sub-sistema de potência	33
Tabela 7:	<i>FMECA</i> do sub-sistema de aquisição	33
Tabela 8:	<i>FMECA</i> do sub-sistema estrutural	34
Tabela 9:	<i>FMECA</i> do sub-sistema de atuação	34
Tabela 10:	<i>FMECA</i> do sub-sistema de processamento	34
Tabela 11:	Lições aprendidas.	37
Tabela 12:	Especificações do motor <i>Dynamixel MX-106</i>	75
Tabela 13:	Especificações do motor <i>Dynamixel H54-S500-R</i>	79

LISTA DE SÍMBOLOS E ABREVIATURAS

DoF Degrees of Freedom

SOTA Study Of The Art

ROS Robot Operating System

AGV Automated Guided Vehicle

URDF Unified Robot Description Format

CAD Computer Aided Design

FMECA Failure Modes, Effects and Critically Analysis

OpenCV Open Source Computer Vision

OMPL Open Motion Planning Library

SUMÁRIO

1 INTRODUÇÃO	11
1.1 Objetivos	11
1.2 Justificativa	11
1.3 Organização do relatório	12
2 CONCEITO DO SISTEMA	13
2.1 Parâmetros básicos	13
2.1.1 Requisitos do cliente	13
2.1.2 Requisitos técnicos	13
2.1.3 Estudo do estado da arte	13
3 DESENVOLVIMENTO DO SISTEMA	17
3.1 Descrição do sistema	17
3.1.1 Arquitetura geral	17
3.1.2 Especificação técnica	18
3.1.3 Ambiente de operação	19
3.1.4 Estrutura analítica do protótipo	22
3.2 Especificação funcional	22
3.2.1 Escaneamento	22
3.2.1.1 Descrição	23
3.2.1.2 Premissas necessárias	23
3.2.1.3 Dependências	24
3.2.1.4 Saídas	24
3.2.2 Planejamento e Execução de Trajetória	24
3.2.2.1 Descrição	24
3.2.2.2 Premissas necessárias	25
3.2.2.3 Dependências	25
3.2.2.4 Saídas	26
3.3 Arquitetura de software	26
3.4 Simulação do sistema	26

4 RESULTADOS E ANÁLISES	29
4.1 Resultados alcançados	30
5 CONFIABILIDADE DO SISTEMA	33
5.1 Análise dos modos e efeitos de falhas	33
5.2 Análise da árvore de falhas	34
6 GESTÃO DO CONHECIMENTO	37
6.1 Lições aprendidas	37
6.2 Guia de uso	37
7 CONCLUSÃO	41
REFERÊNCIAS	43
APÊNDICE A Árvores de TF desconectadas	45
APÊNDICE B Árvores de TF conectadas	48
APÊNDICE C Diagrama de nós do sistema	50
APÊNDICE D Propriedades de Massa do Timon-HM	51
APÊNDICE E Algoritmo de busca e acionamento do painel	57
APÊNDICE F Diagrama elétrico do Timon-HM	63
APÊNDICE G Diagrama de conexão do Timon-HM	67
ANEXO A Especificações da câmera Dalsa Genie Nano	71
ANEXO B Especificações do motor Dynamixel MX-106	75
ANEXO C Especificações do motor Dynamixel H54-S500-R	79

1 INTRODUÇÃO

Os robôs estão cada vez mais presentes tanto no dia-a-dia das pessoas quanto na indústria. O aumento da produtividade, aliado à qualidade final do produto, maior confiabilidade e repetibilidade, tornam evidentes a importância das pesquisas na área de robótica. Esta, é oriunda de uma fusão disciplinar, em que diversos ramos de conhecimento como cinemática, dinâmica, estática, controle, inteligência artificial, programação, computação gráfica e visão computacional estão sendo estudados para serem implementados nos sistemas robóticos e melhorar a performance dos mesmos (OLIVEIRA et al., 2012).

Este relatório tem por finalidade descrever o processo de construção de um manipulador robótico realizado no Laboratório de Robótica e Sistemas Autônomos do SENAI-CIMATEC-BA para o programa de formação de Novos Talentos. Serão descritas as etapas de concepção, simulação e testes realizados que viabilizarão sua futura implementação física.

1.1 Objetivos

O propósito deste projeto é construir um manipulador robótico capaz de identificar um marcador visual por meio de uma câmera RGB e proceder com a função acionar um painel elétrico. Para isso, os objetivos específicos são:

- Realizar estudo do Estado da Arte (*SOTA*) sobre manipuladores.
- Realizar testes e parametrização dos servomotores.
- Propor modelo do manipulador robótico.
- Parametrizar pacote de reconhecimento de marcadores visuais.
- Desenvolver pacote de configuração do *MoveIt*.
- Desenvolver nó para realizar missão.
- Realizar simulação do protótipo do manipulador em software.

1.2 Justificativa

Apesar da crescente demanda, há uma falta de profissionais habilitados em desenvolver pesquisas e aplicações na área da robótica . O presente trabalho tem como impulsionador principal a capacitação de novos pesquisadores preparados para solucionar os mais diversos problemas relacionados a robótica e sistemas autônomos.

A busca por uma melhor eficiência e precisão na realização de atividades em locais que a presença do ser humano torna-se difícil, arriscado e até mesmo impossível, vem se

tornando cada vez maior no cotidiano dos ambiente industriais. Além disso, é importante a capacidade do robô de interagir com o ambiente a partir da captura e análise de estímulos visuais (LEITE, 2005).

Este projeto traz, dentre os benefícios, a utilização de manipuladores robóticos autônomos que sejam capazes de identificar marcadores visuais e realizar tarefas que possam ser perigosas e/ou repetitivas para o ser humano. Espera-se que este projeto seja continuado e seus resultados sejam compartilhados na comunidade científica, contribuindo para a construção de outros manipuladores com características e/ou objetivos semelhantes .

1.3 Organização do relatório

O presente relatório está organizado em sete capítulos, sendo este de Introdução com vista à descrição da justificativa/motivação, os objetivos e a organização do relatório.

No capítulo 2, intitulado Conceitos do sistema, são descritos parâmetros básicos do projeto, dentre eles os requisitos do cliente, requisitos técnicos e o estudo do estado da arte.

O capítulo 3, intitulado Desenvolvimento , apresenta a descrição do sistema onde serão apresentados a arquitetura geral, especificações técnicas, o ambiente de operação do manipulador e a estrutura analítica do protótipo. Além disso, trará as especificações funcionais que compõe o sistema, sua arquitetura de software e o que foi desenvolvido para simulação.

Já no capítulo 4, intitulado Resultados e análises, são apresentados os resultados alcançados e a análise dos dados amostrados.

O capítulo 5, intitulado Confiabilidade do sistema, detalha a análise dos modos e efeitos de falhas além de mostrar análise da árvore de falhas.

No capítulo 6, intitulado Gestão do conhecimento, é feito um estudo sobre as lições aprendidas e o guia uso.

Por fim, o capítulo 7 apresenta a Conclusão do relatório.

2 CONCEITO DO SISTEMA

Neste capítulo serão tratados os requisitos solicitados pelo cliente, os requisitos técnicos do projeto, o estudo do estado da arte sobre manipuladores e o ambiente de operação em que este manipulador realizará a atividade.

2.1 Parâmetros básicos

Nesta seção encontram-se os requisitos solicitados pelo cliente, ou seja, a tarefa que precisa ser realizada e em qual ambiente o manipulador precisa ser simulado. Além disso, são exibidos os requisitos técnicos que tratam das especificações do sistema e uma breve revisão teórica de conceitos relacionados ao manipulador.

2.1.1 Requisitos do cliente

1. Desenvolver um manipulador robótico.
2. Realizar a tarefa de detecção de um marcador visual e acionamento do painel elétrico.
3. Realizar a simulação do manipulador no ambiente *ROS* utilizando o software *Gazebo*.
4. Gerar pacote de configuração no *MoveIt* de acordo com o manipulador.

2.1.2 Requisitos técnicos

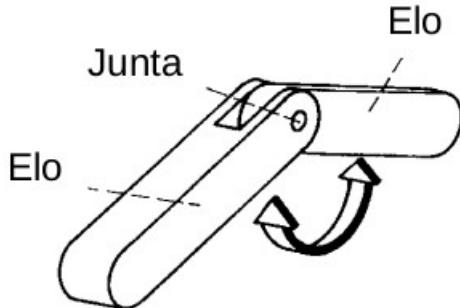
1. A base deve estar a 0,25 m de uma das extremidades da bancada.
2. Deve suportar uma carga máxima de 2 kg.
3. O manipulador deverá acionar um painel elétrico.

2.1.3 Estudo do estado da arte

Com o advento do exponencial crescimento da tecnologia há um foco crescente na pesquisa e comercialização de robôs ([HERNÁNDEZ-ORDOÑEZ et al., 2018](#)). Estes, por sua vez, são classificados em três grupos: manipuladores, veículos auto-guiados ([AGV](#)) e robôs móveis. Neste projeto, o objeto de interesse são os manipuladores robóticos, sistemas que possuem estrutura física similar a um braço humano. Estes robôs são compostos por partes rígidas, denominadas de elos, conectados entre si por juntas ([SANTOS, 2004](#)). Esta estrutura encontra-se descrita na Figura 1.

Muitas pesquisas vêm sido realizadas na área de manipuladores robóticos. Em ([HERNANDEZ-MENDEZ et al., 2017](#)) é descrito o desenvolvimento de um manipulador robótico com 3

Figura 1: Elos e junta de um manipulador robótico.

Fonte: ([SANTOS, 2004](#)).

DoF e dois dedos independentes. Este robô foi desenvolvido com o propósito de manipular objetos, cujas localizações são conhecidas, e transportá-los de uma localidade para outra utilizando *ROS*. Os autores utilizaram o *MoveIt* para tratar do planejamento de trajetória e o *Rviz* como ferramenta de visualização. Além disso, foi desenvolvido um controlador de posição e força para o *endeffecter*¹ durante o processo de "escolher e colocar". Os experimentos realizados trouxeram bons resultados para o que foi proposto, sendo ressaltada a necessidade de adicionar um sistema de visão que permita identificar e localizar o objeto alvo.

O planejamento de trajetória para um manipulador de 5 *DoF* a partir do *MoveIt* é exibido em ([ZHANG; LIN; WU, 2019](#)). A partir de um modelo *CAD* já existente foi construído o modelo *URDF* utilizado para simulação no *ROS*. O processo de planejamento foi visualizado a partir do *Rviz* e o caminho planejado foi transmitido para os servo-motores possibilitando ao manipulador executar a sua rotina. O robô tem como tarefa manipular um objeto alvo de um local para outro, identificado a partir de técnicas de visão computacional que combinam os algoritmos *SIFT* e *RANSAC*. Os resultados experimentais mostraram que o uso do *MoveIt* reduz as dificuldades de operação para manipuladores e oferece vantagens em termos de validação de algoritmos e exploração de funções, sendo possível utilizar o método proposto para o controle em tempo real do robô.

A aplicação de técnicas de visão computacional em um manipulador do tipo *Robai Cyton Gamma 3000* é exibida em ([KHAN; KONDA; RYU, 2018](#)). O robô é conectado com uma câmera externa via *ROS*, possui um *endeffecter* em forma de garra que segura uma estrutura similar a um prato, e tem como tarefa equilibrar uma bola localizada no centro do prato. Para realizar o controle da tarefa de equilíbrio, a bola é identificada a partir de um algoritmo escrito em C++ e que utiliza bibliotecas do *OpenCV*. As juntas do manipulador são atuadas a partir de servo-motores *Dynamixel* que são diretamente controlados por

¹ Na robótica, um *endeffecter* é o dispositivo no final de um braço robótico, projetado para interagir com o meio ambiente.

um algoritmo. O sistema foi desenvolvido de forma que os componentes se comunicassem entre si para receber respostas do sistema de visão e dos motores, computá-las e enviar comandos de controle que permitissem executar a tarefa. Foram realizados testes e os resultados experimentais foram satisfatórios, sendo o manipulador capaz de equilibrar a bola em uma pequena vizinhança do centro do prato.

O uso de marcadores visuais do tipo *ArUco* associados ao planejamento de trajetória para um manipulador robótico é abordado em (JAVEED; PRAKASH; KULKARNI, 2019). O manipulador encontra-se acoplado a uma plataforma móvel e tem como objetivo o transporte de objetos de forma autônoma em um determinado espaço. Os autores também descrevem a integração dos mecanismos de detecção do marcador visual, navegação do robô e planejamento de trajetória, realizados no *ROS*. O robô é capaz de estimar a pose do marcador, aproximar-se e realizar sua tarefa. Os testes realizados mostraram a eficiência do sistema, sendo apontada a necessidade de um estudo futuro para possibilitar o planejamento de trajetória em um ambiente com obstáculos.

3 DESENVOLVIMENTO DO SISTEMA

Nesta seção serão explicitadas as características do manipulador Timon-HM, abordando os sistemas que o compõem em *software* e em *hardware*. Suas funcionalidades principais são abordadas e a conexão entre as mesmas é exibida.

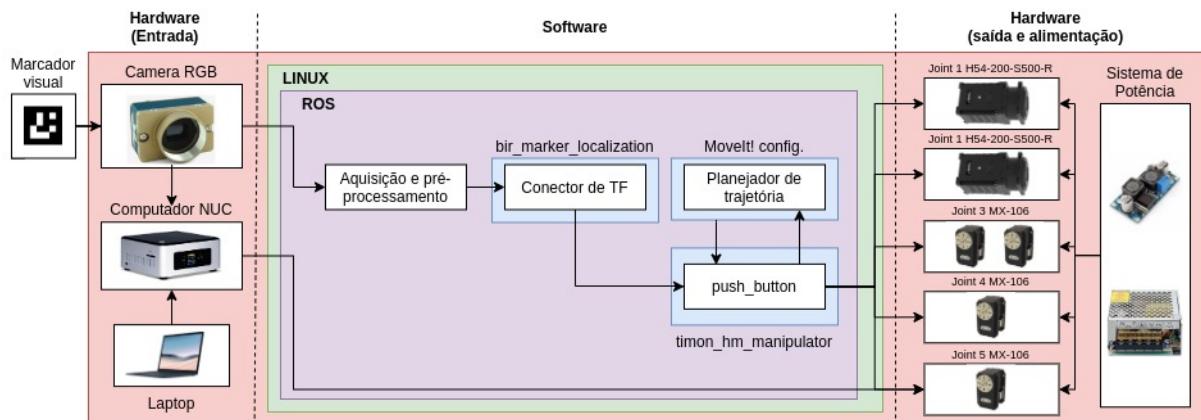
3.1 Descrição do sistema

Timon-HM é um manipulador desenvolvido para atender às demandas relacionadas ao reconhecimento de marcadores visuais e, a partir desta identificação, realizar o acionamento de um painel elétrico. Os pacotes que constituem este robô foram concebidos através do *software* de simulação *Gazebo*, da ferramenta de visualização *Rviz* e do *framework*¹ para planejamento de trajetória *MoveIt*. O uso dessas ferramentas possibilita que uma grande variedade de atividades que venham a ser realizadas no mundo real tenham sido previamente testadas em ambiente computacional.

3.1.1 Arquitetura geral

A Figura 2 ilustra a estruturação do sistema e a relação entre *software* e *hardware*. As cores representam o sistema geral(salmão), sistema operacional(verde), *framework*(salmão) e pacotes(azul).

Figura 2: Arquitetura geral do sistema.



Fonte: Autoria própria.

Um laptop, conectado via acesso remoto, dá início a aplicação no computador *NUC*² que possui instalado o software do protótipo. Com o sistema iniciado, a câmera RGB é

¹ São conjuntos de aplicações dentro de um projeto que interagem entre si e com isso se alcança resultados como uma determinada função de um programa.

² Computador pequeno, completo e altamente eficiente energeticamente.

capaz de obter dados visuais do ambiente e enviá-los para o *ROS*. Ao encontrar o marcador visual, o pacote *bir_marker_localization* é capaz de unir as árvores de *TF*³ do painel elétrico e do manipulador, que antes encontravam-se desconectadas. Esta conexão, garante que sejam conhecidos os dados de posição ao nó *push_button* possibilitando o planejamento de trajetória para o ponto desejado. Com o caminho planejado, *push_button* pode enviar os comandos para cada junta do manipulador, onde encontram-se os motores *Dynamixel*, que são alimentadas pelo sistema de potência.

3.1.2 Especificação técnica

Na Tabela 1 estão elencadas as especificações técnicas do manipulador robótico Timon-HM. O numero de Graus de Liberdade foi definido baseado na capacidade de movimentação necessária para a realização dos desafios propostos. A carga útil, peso e o alcance máximo foram calculados com o auxilio do software *Onshape*, uma alternativa ao cálculo manual. A faixa de operação dos motores foi obtida segundo os limites de segurança observados, o que acrescenta proteção principalmente ao cabeamento do sistema. Informações a respeito de componentes como câmeras e motores seguem o seu padrão original de fabricação.

Tabela 1: Especificações técnicas do manipulador Timon-HM.

Item	Timon-HM
Graus de liberdade	5
Carga útil	1,94 kg
Alcance	979 mm
Peso	7,38 kg
Tensão de operação	24 V
Resolução	Junta 1, Junta 2: 1.003.846 pulsos/rev
	Junta 3, Junta 4, Junta 5: 4096 pulsos/rev
Motores	Junta 1, Junta 2: H54-200-S500-R (200 W)
	Junta 3(2), Junta 4, Junta 5: MX-106 (65 W)
Faixa de operação	Junta 1: $-90^\circ \sim 90^\circ$
	Junta 2: $-90^\circ \sim 90^\circ$
	Junta 3: $-90^\circ \sim 135^\circ$
	Junta 4: $-180^\circ \sim 180^\circ$
	Junta 5: $-90^\circ \sim 90^\circ$
Camera	Teledyne Genie Nano C2590
Tipo do sensor de posição	Pose inicial: Codificador Absoluto
	Controle: Codificador Incremental
Comunicação	RS485
Taxa de transmissão	1 Mbps

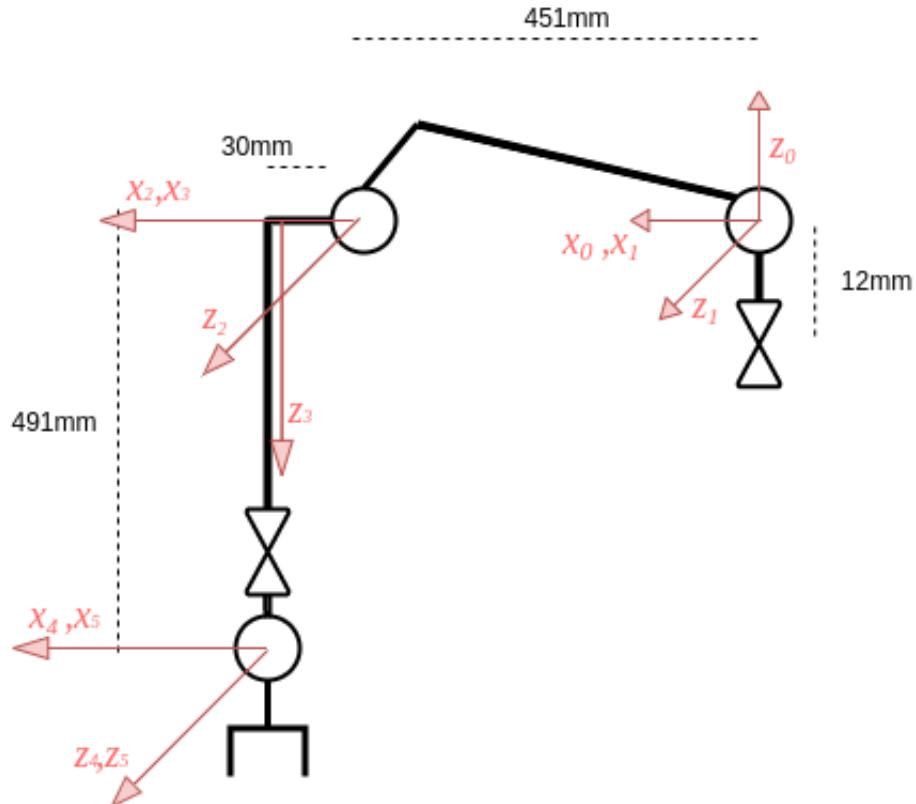
Fonte: Autoria própria.

Para poder relacionar a pose do *endeffecter* com a base do manipulador é necessário

³ Pacote do *ROS* que permite verificar as relações entre os *frames* na estrutura de árvore.

descrever o seu sistema de coordenadas em relação ao sistema de coordenadas de origem. Para isto, é utilizada a notação de *Denavit-Hartenger*(D-H). A partir da configuração D-H exibida na Figura 3 foram retirados os parâmetros exibidos na tabela 2.

Figura 3: Configuração D-H do manipulador Timon-HM.



Fonte: Autoria própria.

Tabela 2: Parâmetros D-H para o manipulador Timon-HM.

Link	a (mm)	$\alpha(^{\circ})$	d (mm)	$\theta(^{\circ})$
1	0	90	12	0
2	452	0	0	$90 - \arctan(30/451)$
3	30	-90	0	$45 + \arctan(30/451)$
4	0	90	491	0
5	0	0	0	0

Fonte: Autoria própria.

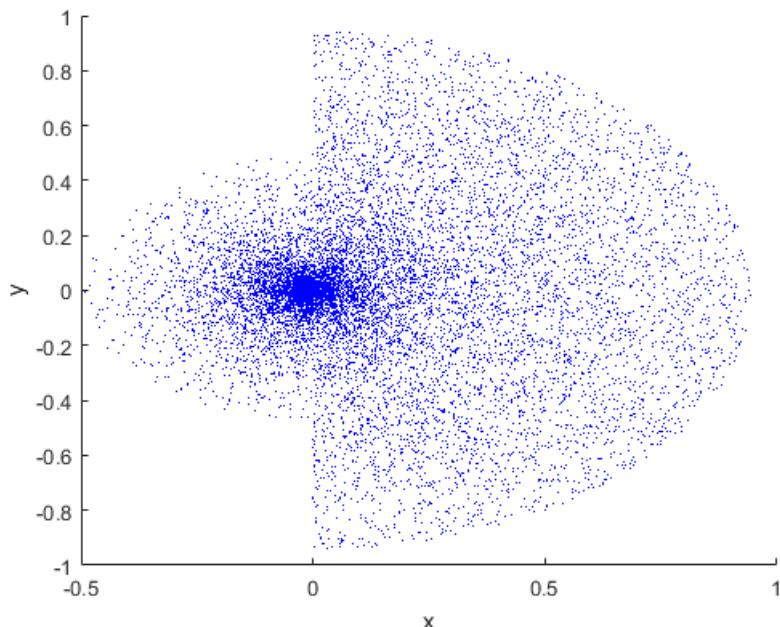
3.1.3 Ambiente de operação

O ambiente físico para a realização do desafio, onde serão incluídos o manipulador e um painel elétrico, é uma mesa com 1.7 m de comprimento por 0.8 m de largura. É

necessário então que verifique-se quais os pontos deste ambiente de trabalho que estão dentro da área de alcance (*workspace*)⁴ do manipulador.

A partir da tabela 2 foi desenvolvido um código utilizando o software *Matlab R2020a* capaz de gerar pontos que populem o *workspace* do manipulador nas projeções dos planos X-Y, X-Z e Y-Z, conforme exibido nas figuras 4, 5 e 6. A região em azul indica o alcance do robô e, a partir destas imagens, é possível observar que o manipulador possui restrições de operação devidas às limitações existentes em cada junta.

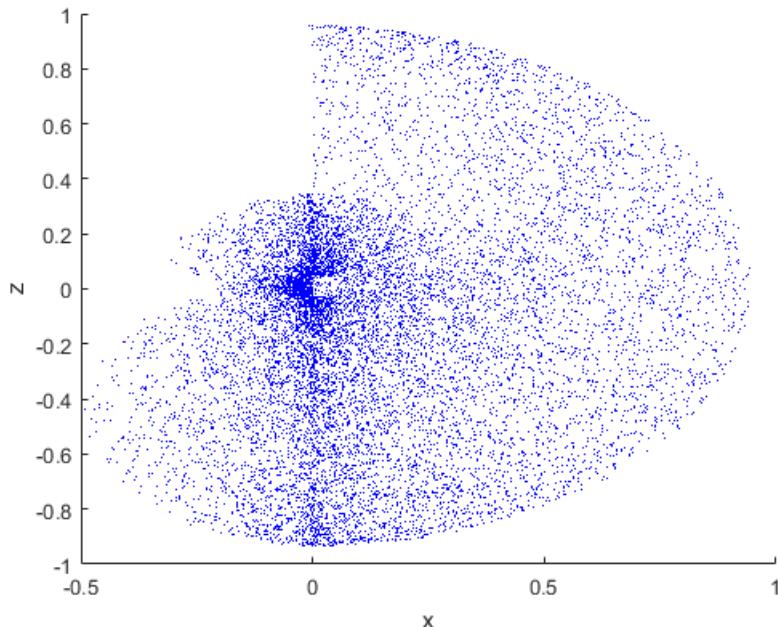
Figura 4: *Workspace* do manipulador no plano X-Y.



Fonte: Autoria própria.

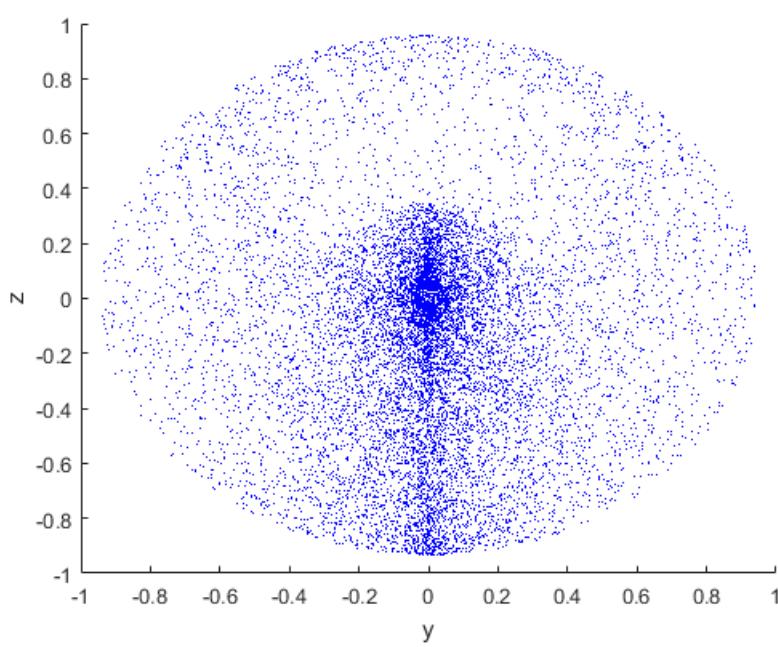
⁴ É a área de alcance do manipulador, região na qual este consegue operar.

Figura 5: *Workspace* do manipulador no plano X-Z.



Fonte: Autoria própria.

Figura 6: *Workspace* do manipulador no plano Y-Z.

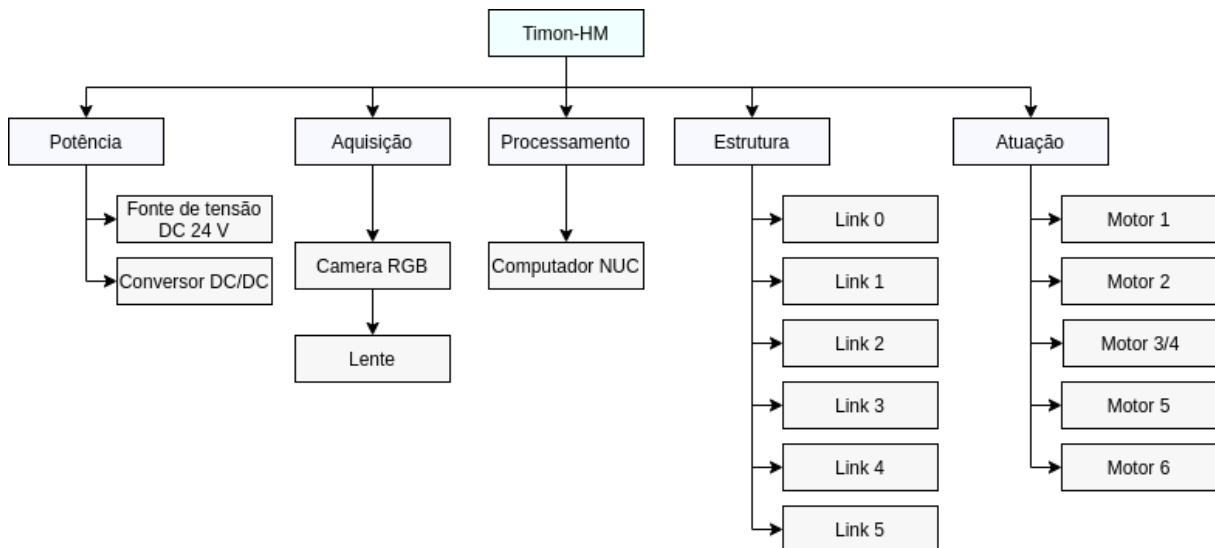


Fonte: Autoria própria.

3.1.4 Estrutura analítica do protótipo

A estrutura analítica do protótipo mostrada na Figura 7 exibe as relações sistemáticas entre as partes que compõem o manipulador. A estrutura hierárquica possui três níveis: o primeiro, referente ao sistema principal Timon-HM; o segundo, que é composto pelos sub-sistemas de potência, aquisição, processamento, estrutura e atuação; e o terceiro, composto pelos itens que fazem parte de cada um destes sub-sistemas.

Figura 7: Estrutura analítica do protótipo.



Fonte: Autoria própria.

3.2 Especificação funcional

O manipulador descrito trabalha acionando os botões encontrados pelo sistema de aquisição. Seu software funciona baseado na troca de mensagens entre duas funcionalidades principais: Escaneamento e Planejamento/Execução de trajetória. O Escaneamento corresponde à detecção de um marcador visual, que uma vez detectado, informa a posição no espaço de um painel elétrico que precisa ser acionado. O planejamento e execução de trajetória utiliza cálculos de cinemática direta e inversa para definir a trajetória de movimentação que permitirá ao manipulador realizar sua tarefa.

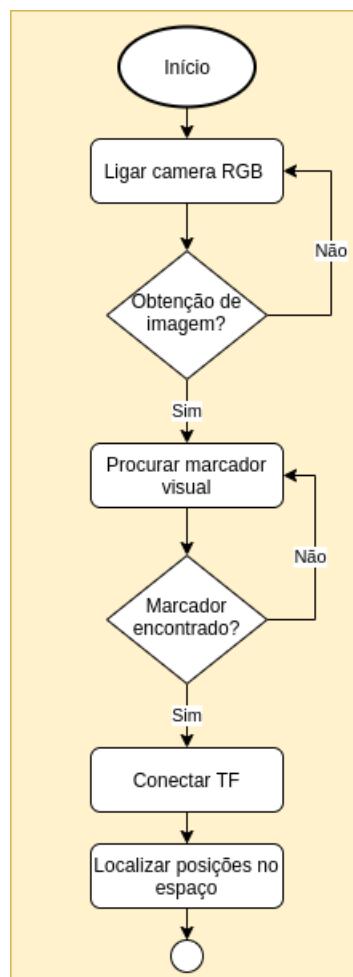
3.2.1 Escaneamento

Uma câmera RGB *Teledyne Genie Nano c2590* equipada com lente *kowa LM8FC* foi acoplada ao manipulador Timon-HM. Através da mesma é realizada a detecção do marcador visual, utilizando a biblioteca *ArUco* e o pacote *Bir Marker Localization*, hospedado no site do Github no perfil do BIR - Brazilian Institute of Robotics ([BIR... , 2020](#)).

3.2.1.1 Descrição

A Figura 8 exibe o fluxograma que descreve o funcionamento do sistema de escaneamento integrado ao manipulador. Após a captura da imagem, a partir da câmera RGB, é feito um processamento dos dados obtidos afim de localização da *tag ArUco*, cujos tamanho e ID foram previamente estabelecidos. Caso o marcador seja encontrado, as árvores de *TF* do painel elétrico onde encontra-se o marcador, e do manipulador robótico são conectadas, possibilitando a localização no espaço da pose alvo. Os apêndices A e B exibem as árvores de *TF* antes e após a conexão realizada.

Figura 8: Fluxograma do sistema de escanamento.



Fonte: Autoria própria.

3.2.1.2 Premissas necessárias

- Deverá haver um marcador visual associado ao painel elétrico.
- A orientação dos elementos envolvidos no escaneamento, câmera RGB e marcador visual, devem ser definidos conforme estabelecido pelo pacote *Bir Marker Localization*.

- Não haver oclusão do marcador visual.
- O marcador visual deverá possuir tamanho adequado para detecção.

3.2.1.3 Dependências

Para realizar a etapa de detecção é necessária a instalação do *OpenCV* versão 3.3.1 e a inserção do pacote *Bir Marker Localization* no *workspace* do manipulador.

3.2.1.4 Saídas

É fornecida ao sistema resposta por meio de uma sequência de imagens publicadas no tópico */timon/camera/image_raw*. Estes dados são analisadas pelo detector *ArUco*, possibilitando seu uso em um pacote desenvolvido na linguagem C++ que determina a posição do painel elétrico associado ao marcador visual.

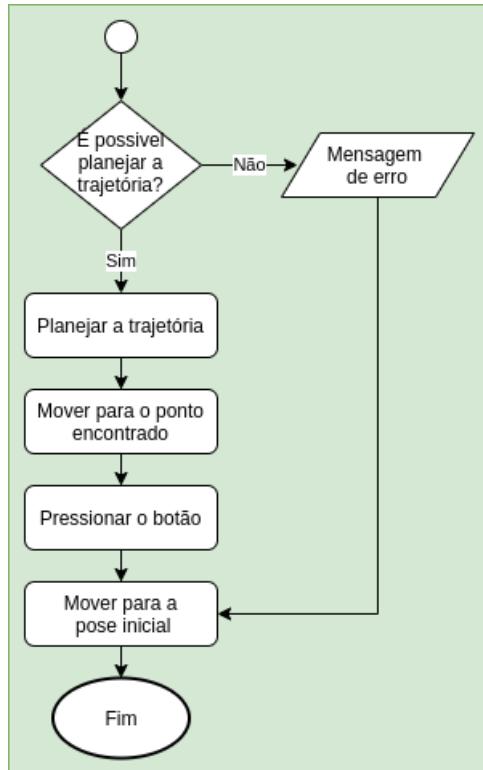
3.2.2 Planejamento e Execução de Trajetória

As equações cinemáticas são a base que possibilitam a pesquisa do movimento dos manipuladores. A cinemática inversa provê um conjunto de valores para as juntas do manipulador com o intuito de alcançar uma determinada pose pré-estabelecida do seu *endeffector*. Para resolver as equações da cinemática inversa do Timon-HM, optou-se por utilizar o plugin TRAC-IK, um método alternativo ao padrão da inversa Jacobiana utilizado pelo *MoveIt*. Este método se adequa bem a manipuladores que possuem limitações em suas juntas, ao contrário de algoritmos baseados no teorema de Newton (BEESON; AMES, 2015). Para o planejamento de trajetória foi utilizada a biblioteca *OMPL*, uma coleção de algoritmos de planejamento utilizada por padrão no *MoveIt* (SUCAN; MOLL; KAVRAKI, 2012).

3.2.2.1 Descrição

A Figura 9 exibe o funcionamento do sistema de planejamento e execução de trajetória aplicado ao manipulador. A pose do painel elétrico, determinada a partir do que foi mostrado em 3.2.1, é enviada como entrada para o *MoveIt*. Caso seja possível, é realizado o planejamento de uma trajetória para cada uma das juntas do manipulador, a fim de movê-lo para a pose desejada. Esta trajetória é então enviada para os atuadores das juntas, que passam a executá-la. Após a realização da rotina para o pressionamento do painel elétrico, o manipulador retorna para sua posição inicial. Caso alguma condição impeça o planejamento de trajetória, como por exemplo o posicionamento do painel elétrico fora da área de trabalho do manipulador ou falhas nas soluções para as equações da cinemática inversa, uma mensagem de erro é exibida e o robô retorna para sua posição inicial.

Figura 9: Fluxograma do sistema de planejamento e execução de trajetória.



Fonte: Autoria própria.

3.2.2.2 Premissas necessárias

- Detecção do marcador visual.
- Viabilidade das soluções para cinemática inversa.
- Painel elétrico estar posicionado na área de trabalho do manipulador.

3.2.2.3 Dependências

O sistema de movimentação é dependente da versão Melodic Morenha do *framework ROS* e da plataforma *MoveIt*. Além destes, uma lista de pacotes deve ser instalada previamente para o funcionamento correto do sistema:

- ros-melodic-ros-control
- ros-melodic-gazebo-ros-control
- ros-melodic-controller-manager
- ros-melodic-joint-trajectory-controller
- ros-melodic-joint-state-controller

- ros-melodic-position-controllers
- ros-melodic-trac-ik-kinematics-plugin

3.2.2.4 Saídas

As respostas fornecidas pelo sistema de movimentação são entregues aos motores *Dynamixel* integrados às juntas do manipulador. A trajetória gerada pelo *MoveIt* pôde ser visualizada a partir do tópico */move_group/display_planned_path*.

3.3 Arquitetura de software

O robô Timon-HM foi desenvolvido para atuar em conjunto com o *ROS*, isto é, segue o propósito de conectar diferentes módulos, como câmeras, motores, sensores e códigos. A proposta é conectar um programa de visão capaz de conectar árvores de *TF*, através da identificação de marcadores visuais, com um sistema de movimentação que posiciona o manipulador para acionar o painel elétrico.

O apêndice C exibe o diagrama do *software* do sistema, obtido via *rqt_graph*. Observa-se a comunicação entre o manipulador (*/timon/robot_state_publisher*) e o painel elétrico (*/box/box_state_publisher*) a partir da conexão das árvores de *TF* realizadas pelo */marker_localization* que recebe e analisa os dados da câmera (*/timon/camera_image_raw*). Pode-se verificar também que o nó */push_button* comunica-se com o nó */tf* a fim de adquirir a pose do painel elétrico e enviá-la ao nó */move_group* que realiza o planejamento e execução da trajetória.

3.4 Simulação do sistema

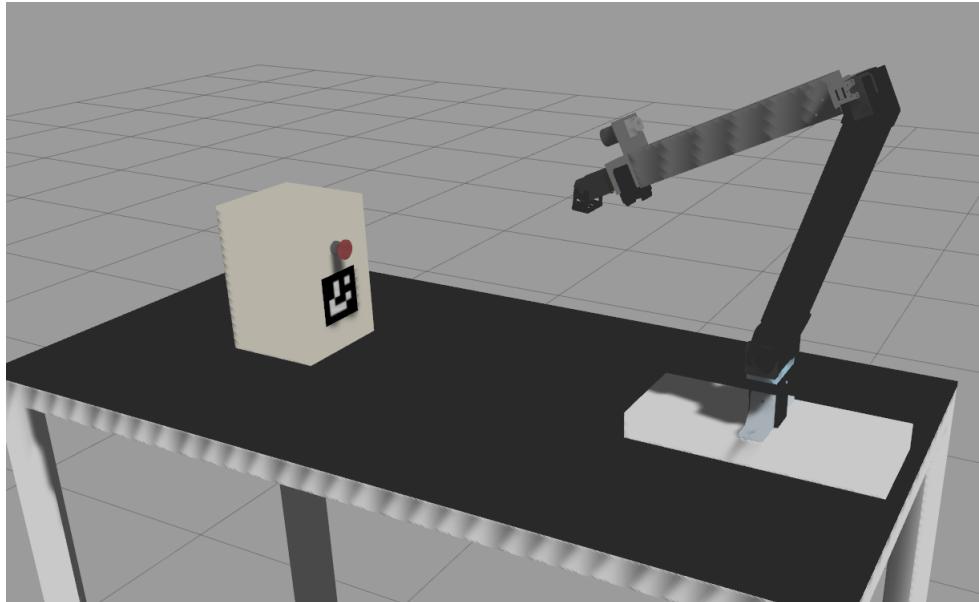
A simulação do robô Timon-HM inserido no seu ambiente de trabalho foi realizada na plataforma *Gazebo*. Para isto, realizou-se o desenvolvimento dos arquivos *timon_arm_hand_base.urdf* e *box.urdf*, modelos *URDF* que descrevem o robô e o painel elétrico a ser acionado. Foram levados em consideração características de massa, inércia e dimensão para cada componente destes modelos, para que houvesse maior fidelidade possível com o que representam fisicamente, de forma a garantir que os testes realizados possam ser validados no mundo real.

O pacote *ros_control* dispõe de uma lista de controladores disponíveis. Para Timon-HM, foram utilizados controladores do tipo *position_controllers/JointTrajectoryController*, e as transmissões para cada junta do manipulador foram definidas em seu modelo *URDF*.

Para simulação dâ câmera RGB, foi desenvolvido o arquivo *camera.xacro*. O plugin padrão do *Gazebo* foi utilizado, *ligazebo_ros_camera.so* e as especificações da câmera RGB *Teledyne Genie Nano c2590* foram levados em consideração, garantindo maiorrealismo à

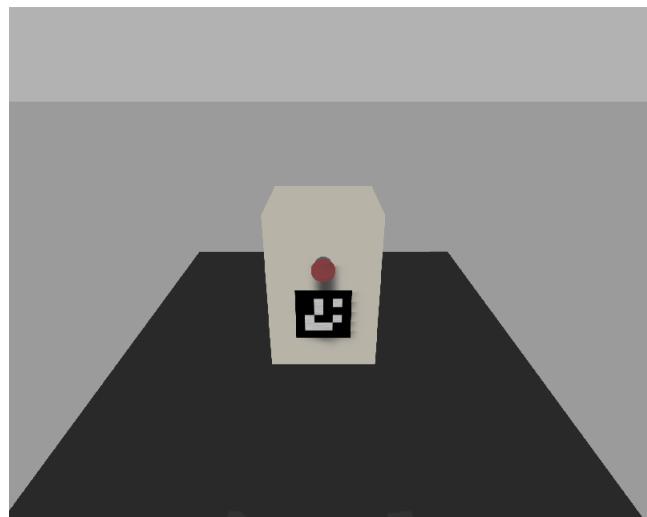
simulação. A figura 10 exibe os modelos simulados do manipulador e do painel elétrico, enquanto a figura 11 exibe imagem capturada pela câmera instalada no manipulador.

Figura 10: Modelos simulados do manipulador e do painel elétrico.



Fonte: Autoria própria.

Figura 11: Imagem capturada pela câmera RGB.



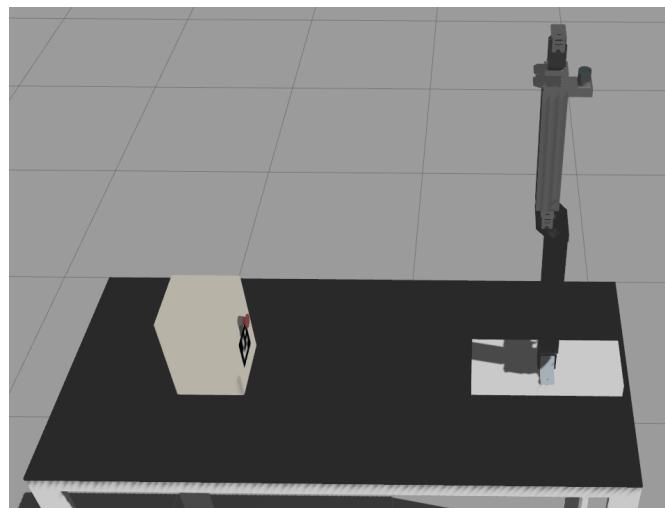
Fonte: Autoria própria.

4 RESULTADOS E ANÁLISES

Os testes para a amostragem dos dados exibidos nesta seção foram realizados em um computador com sistema operacional Ubuntu 18.04, processador Intel Core i7-4790 3.60GHZ, 16GB de memória RAM e placa de vídeo NVIDIA GeForce GT 730.

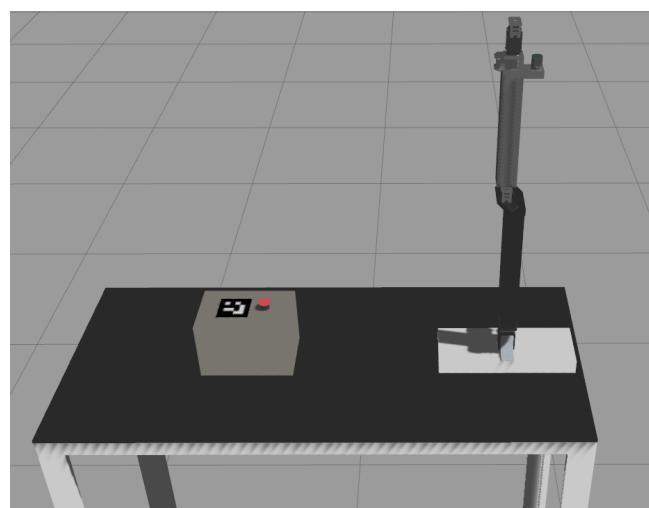
Foram estipuladas 6 diferentes posições para o botão, considerando orientações na vertical e na horizontal para o painel elétrico, conforme Tabela 3. Duas destas posições estão exibidas nas Figuras 12 e 13.

Figura 12: Pose 1 testada para botão.



Fonte: Autoria própria.

Figura 13: Pose 4 testada para botão.



Fonte: Autoria própria.

Tabela 3: Poses testadas para o botão.

Orientação do painel elétrico	Pose	Distância no eixo x (m)	Distância no eixo y (m)
Vertical	1	0.9	0
	2	0.6	0.2
	3	0.95	-0.2
Horizontal	4	0.9	0
	5	0.6	0.2
	6	0.8	-0.2

4.1 Resultados alcançados

Foram realizados 10 ensaios consecutivos para cada posição exibida na Tabela 3. Os dados referentes a estas simulações podem ser visualizados nas Tabelas 4 e 5.

Tabela 4: Testes realizados para painel elétrico com orientação vertical.

Painel elétrico com orientação vertical									
Pose 1			Pose 2			Pose 3			
N	Tempo(s)	Sucesso	N	Tempo(s)	Sucesso	N	Tempo(s)	Sucesso	
1	22	sim	1	25	sim	1	26	sim	
2	24	sim	2	29	sim	2	24	sim	
3	20	sim	3	23	sim	3	19	sim	
4	23	sim	4	32	sim	4	19	sim	
5	22	sim	5	28	sim	5	33	não	
6	24	sim	6	27	sim	6	24	sim	
7	21	sim	7	23	sim	7	26	sim	
8	22	sim	8	23	sim	8	29	não	
9	21	sim	9	24	sim	9	21	sim	
10	22	sim	10	21	sim	10	29	sim	

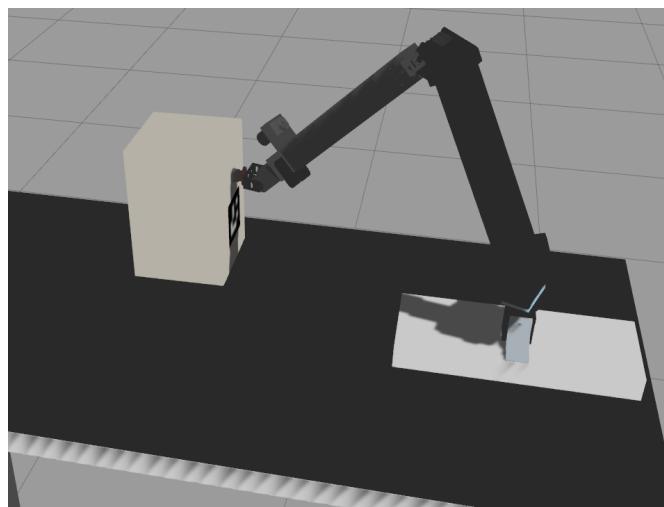
Tabela 5: Testes realizados para painel elétrico com orientação horizontal.

Painel elétrico com orientação horizontal									
Pose 4			Pose 5			Pose 6			
N	Tempo(s)	Sucesso	N	Tempo(s)	Sucesso	N	Tempo(s)	Sucesso	
1	24	sim	1	27	sim	1	26	sim	
2	21	sim	2	17	sim	2	27	não	
3	24	sim	3	25	sim	3	25	não	
4	26	sim	4	21	sim	4	18	sim	
5	26	sim	5	22	sim	5	21	sim	
6	17	sim	6	23	sim	6	25	sim	
7	23	sim	7	22	sim	7	20	sim	
8	25	sim	8	25	sim	8	21	sim	
9	21	sim	9	24	sim	9	23	sim	
10	27	sim	10	23	sim	10	23	sim	

Conforme os dados exibidos verificou-se que o manipulador obteve sucesso em 56 das 60 tentativas realizadas, resultando em uma taxa total de 90% de aproveitamento com tempo médio de execução da rotina de 24 segundos. Nas 4 tentativas em que houve falha, o *endeffector* chegou à posição desejada, porém, devido a erros de planejamento para a sua orientação, não foi capaz de pressionar o botão.

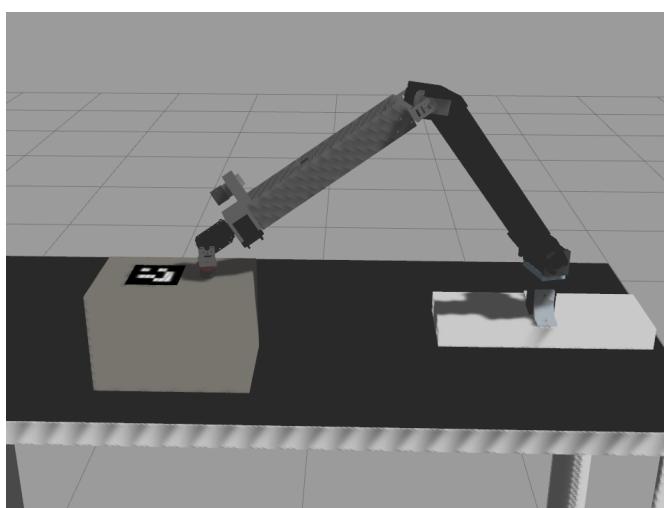
As Figuras 14 e 15 exibem o robô Timon-HM no exato momento em que foi capaz de pressionar o botão para duas das poses desejadas, após localização do mesmo via identificação da *tag ArUco*. Pode se constatar, a partir das mesmas e dos dados exibidos, que os resultados foram satisfatórios no que diz respeito ao reconhecimento da *tag ArUco*, planejamento e execução de trajetória para um determinado ponto.

Figura 14: Timon-HM pressionando botão localizado em pose 2.



Fonte: Autoria própria.

Figura 15: Timon-HM pressionando botão localizado em pose 6.



Fonte: Autoria própria.

5 CONFIABILIDADE DO SISTEMA

No presente capítulo para análise da confiabilidade do sistema, foi realizado levantamento de cada componente que constitui o manipulador robótico e dessa forma possibilitar o estudo detalhado das prováveis falhas. Para isso, foi utilizado o *FMECA* afim de analisar cada sub-sistema, como pode ser visto nas tabelas da seção 5.1.

E na seção 5.2 foi desenvolvida a árvore de falha, que permite através de um processo lógico dedutivo chegar-se às causas-raiz de uma determinada falha.

5.1 Análise dos modos e efeitos de falhas

Nas Tabelas 6, 7, 8, 9 e 10 estão representadas informações referentes ao estudo sistemático e estrutura das falhas potenciais para os sub-sistemas de potência, de aquisição, estrutural, de atuação e de processamento, respectivamente.

Tabela 6: *FMECA* do sub-sistema de potência

Análise do Tipo e Efeito de Falha								
Sub-sistema de potência								
Função(es)	Modo(s) de falha em potencial	Efeito(s) potencial(is) da falha	S	Causa(s) potenciais	O	D	C	Ação(es) recomendada(s)
Transmissão de dados	Incapacidade de transmissão de dados	Comprometimento dos dados transmitidos	6	Fissura dos fios	3	5	90	Manutenções preventivas
		Perda da capacidade de transmissão de dados	8	Desgaste causado pelo tempo	2	1	16	
Transmissão de energia	Incapacidade de transmissão de energia	Perda da capacidade de transmissão de energia	8	Derretimento por temperatura elevada	1	1	8	Manutenções preventivas
		Má qualidade da energia transmitida	6	Torção dos fios	3	5	90	
Alimentação do sistema	Incapacidade em fornecer energia	Não energização do sistema	9	Má conexão com fonte de tensão	3	6	162	Verificar conexão com fonte
				Desgaste nas soldas causado pelo tempo	4	7	252	Verificar continuidade na placa do circuito

Fonte: Autoria própria.

Tabela 7: *FMECA* do sub-sistema de aquisição

Análise do Tipo e Efeito de Falha								
Sub-sistema de aquisição								
Função(es)	Modo(s) de falha em potencial	Efeito(s) potencial(is) da falha	S	Causa(s) potenciais	O	D	C	Ação(es) recomendada(s)
Aquisição de dados visuais	Incapacidade de obter dados visuais	Perda da capacidade de obter dados visuais	7	Obstrução do campo de visão da câmera	2	1	14	Verificar condições do ambiente antes das missões
	Ruptura da estrutura de fixação da câmera			Baixa visibilidade do ambiente	2	5	70	
Aquisição de dados visuais	Coleta de dados inconsistentes ou insuficientes	Comprometimento dos dados	6	Set-up incorreto	2	7	84	Verificar cabeamento do sistema
				Falha no canal de comunicação	4	7	168	

Fonte: Autoria própria.

Tabela 8: *FMECA* do sub-sistema estrutural

Análise do Tipo e Efeito de Falha									
Sub-sistema estrutural									
Função(es)	Modo(s) de falha em potencial	Efeito(s) potencial(is) da falha	S	Causa(s) potenciais	O	D	C	Ação(es) recomendada(s)	
Ligar o frame ao perfil de alumínio	Incapacidade de sustentar o perfil	Ruptura do conector	8	Excesso de peso aplicado	2	1	16	Verificar a fabricação da peça	
				Desgaste pelo tempo	2	5	80		
	Compor estrutura mecânica	Trinca do conector	6	Set-up incorreto	4	6	144	Verificar as condições de peso aplicadas	
				Defeito de fabricação	5	6	180		
Danificação estrutural	Ruptura do perfil	8	Sobrecarga	2	4	64	Cuidado no manuseamento dos perfis		
	Folga entre conexões	6	Colisão	4	2	48	Definir área de trabalho		

Fonte: Autoria própria.

Tabela 9: *FMECA* do sub-sistema de atuação

Análise do Tipo e Efeito de Falha									
Sub-sistema de atuação									
Função(es)	Modo(s) de falha em potencial	Efeito(s) potencial(is) da falha	S	Causa(s) potenciais	O	D	C	Ação(es) recomendada(s)	
Movimentação do manipulador	Rotação inconsistente com o desejado	Perda de desempenho (velocidade; controle)	6	Emperramento parcial	4	6	144	Inspecções periódicas	
Fornecer o torque mínimo de sustentação	Curto circuito	Parada do motor	8	Cabeamento incorreto	2	5	80	Medir corrente nos terminais	
	Perda de comunicação	Perda de desempenho	6	Falha na alimentação	4	5	120		
	Tensão insuficiente								

Fonte: Autoria própria.

Tabela 10: *FMECA* do sub-sistema de processamento

Análise do Tipo e Efeito de Falha								
Sub-sistema de processamento								
Função(es)	Modo(s) de falha em potencial	Efeito(s) potencial(is) da falha	S	Causa(s) potenciais	O	D	C	Ação(es) recomendada(s)
Processamento dos dados	Não processamento dos dados do sistema principal	Não funcionamento	9	Queima/ danificação de componentes	3	8	216	Manutenção preditiva
				Falha no sistema operacional	2	8	144	Executar teste de verificação do software
				Falha na porta de comunicação	2	2	36	Executar teste de entrada e saída de dados com <i>NUC</i>
				Problemas de contato elétrico	2	3	54	Inspecções periódicas em bancada

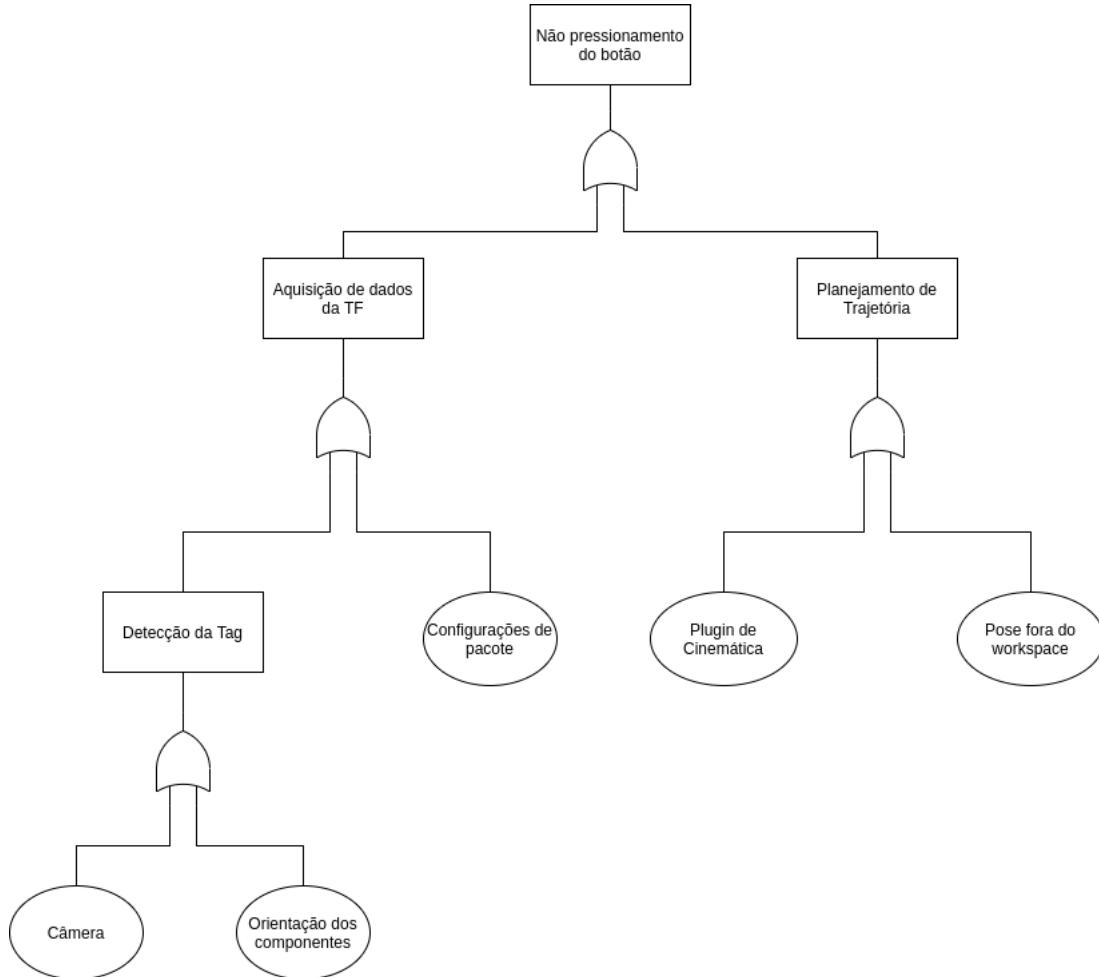
Fonte: Autoria própria.

5.2 Análise da árvore de falhas

Na Figura 16 encontra-se a árvore de falhas do sistema do manipulador Timon-HM. Observa-se que caso o manipulador não consiga realizar a tarefa de pressionar o botão, duas causas principais devem ser investigadas: a conexão entre as TF's, realizada a partir do pacote *Bir Marker Localization*, ou o planejamento de trajetória, realizado pelo *Moveit*.

Para a primeira causa, devem ser verificadas as configurações do pacote e da câmera, bem como a orientação dos componentes envolvidos no processo de detecção. Já para solucionar os problemas correspondentes ao planejamento de trajetória, deve-se observar se a pose desejada está dentro do *workspace* do manipulador e se o *plugin* de cinemática inversa utilizado está sendo capaz de realizar as conversões necessárias para a pose desejada.

Figura 16: Árvore de falhas do sistema.



Fonte: Autoria própria.

6 GESTÃO DO CONHECIMENTO

Neste capítulo estão descritas as lições aprendidas durante o processo de desenvolvimento do protótipo que foram criadas a partir da comparação entre o que era esperado e o que realmente aconteceu em cada etapa do projeto.

Além das lições aprendidas, a seção 6.2 traz o guia de uso com o propósito de auxiliar o usuário na replicação dos experimentos realizados neste relatório.

6.1 Lições aprendidas

A Tabela 11 mostra como foi estruturada cada lição aprendida abordando os seguintes aspectos: Tema, Fase, Impacto, O que ocorreu?, Como resolveu?, Resultados e Recomendações para os próximos projetos. O objetivo deste estudo é a correção dos impactos negativos para os projetos subsequentes.

LIÇÕES APRENDIDAS						
Tema	Fase	Impacto	O que ocorreu?	Como resolveu?	Resultados	Recomendações para os próximos projetos
Gestão	Planejamento	Negativo	Ausência de uma metodologia de trabalho	Reunião com foco em definir metodologia de projeto	Evolução na comunicação dos membros	Antes de começar o projeto realizar reunião para definição de metodologia
Gestão	Planejamento	Negativo	Ausência de uma ferramenta para gestão de projeto	Escolha de uma ferramenta gratuita e de fácil aplicação	Melhoria na organização das atividades	Definição prévia de uma ferramenta de gestão de projeto
Gestão	Execução	Negativo	Montagens e desmontagens do manipulador	Planejamento prévio das atividades	Otimização do tempo	Cronograma definindo as atividades a serem realizadas
Tecnológico	Execução	Negativo	Falha no dimensionamento das peças	Consultoria sobre materiais aplicados na confecção	Obtenção de peças com maior resistência mecânica	Pesquisa e consultoria prévia antes da modelagem das peças
Tecnológico	Execução	Negativo	Dificuldade em planejamento de trajetória para determinadas poses do robô	Utilização do plugin Trac-IK	Maior eficiência de planejamento	Pesquisa e consultoria prévia do pacote mais adequado para o projeto

Tabela 11: Lições aprendidas.

6.2 Guia de uso

Para replicar a simulação do manipulador robótico Timon-HM é necessário seguir os passos descritos nesta seção. Recomenda-se a utilização do Ubuntu 18.04 LTS e o *ROS Melodic Morenia*.

Antes de inserir o pacote do manipulador no *workspace* é fundamental que sejam instalados os pacotes requeridos para este. Primeiramente, no terminal, segue-se os comandos listados:

- Instalar MoveIt:

```
$ sudo apt-get install ros-melodic-moveit
```

- Instalar pacote de ferramentas visuais do MoveIt:

```
$ sudo apt-get install ros-melodic-moveit-visual-tools
```

- Instalar TRAC-IK para resolução da cinemática:

```
$ sudo apt-get install ros-melodic-trac-ik
```

- Instalar pacote de controle no *Gazebo ROS*:

```
$ sudo apt-get install ros-melodic-gazebo-ros-control
```

- Instalar pacote do controlador do *ROS*:

```
$ sudo apt-get install ros-melodic-controller-*
```

- Instalar pacote controlador de posição do *ROS*:

```
$ sudo apt-get install ros-melodic-position-controller
```

- Instalar pacote controlador de esforço do *ROS*:

```
$ sudo apt-get install ros-melodic-effort-controller
```

- Instalar pacote de juntas do *ROS*:

```
$ sudo apt install ros-melodic-joint-*
```

Antes de instalar o pacote *bir_marker_localization* é necessária a instalação do *OpenCV* versão 3.3.1, este possui o guia de instalação próprio disponível em:

<https://www.learnopencv.com/install-opencv3-on-ubuntu/>.

Após a instalação do *OpenCV*, para clonar o repositório do *bir_marker_localization* segue-as as seguintes etapas, no terminal:

- Criar um *workspace* para adicionar dentro deste os pacotes que serão usados na simulação.

```
$ mkdir nomeoworkspace_ws
```

- Entrar no *workspace*:

```
$ cd nomeoworkspace_ws
```

- Criar a pasta *source*¹:

```
$ mkdir src
```

- Entrar no *source*:

```
$ cd src
```

- Clonar o repositório *Bir Marker Localization* para *workspace* (Verificar qual a *branch* estável):

```
$ git clone https://github.com/Brazilian-Institute-of-Robotics/bir_marker_localization.git
```

- Clonar o pacote do manipulador para dentro da pasta *src*:

```
$ git clone https://github.com/Brazilian-Institute-of-Robotics/timon_hm_manipulator.git
```

- Compilar o *workspace*:

```
$ catkin_make
```

- Ativar o ambiente virtual do *workspace*:

```
$ source devel/setup.bash
```

Após realizar os procedimentos citados o *workspace* estará configurado para executar a simulação, com os seguintes comandos:

- Iniciar a simulação no *Gazebo*:

```
$ roslaunch manipulator_gazebo gazebo.launch
```

- Iniciar pacote MoveIt do Timon-HM:

```
$ roslaunch manipulator_gazebo moveit_demo.launch
```

- Iniciar o *bir_marker_localization*:

```
$ roslaunch timon_demo bir_marker_localization.launch
```

- Iniciar o algoritmo de busca do marcador visual e para acionar o painel elétrico:

```
$ roslaunch timon_demo push_button.launch
```

¹ Pasta onde contém os arquivos fonte.

7 CONCLUSÃO

O presente relatório descreveu a idealização e simulação de Timon-HM, um manipulador robótico com 5 *DoF*, integrado ao *ROS*, capaz de acionar um painel elétrico a partir da localização de um marcador visual *ArUco*. A estrutura física do robô foi definida e modelos *URDF* descrevendo a mesma e o ambiente no qual o robô estava inserido foram desenvolvidos.

Para possibilitar o uso da ferramenta *Moveit*, arquivos de configuração foram gerados e sua comunicação com o modelo simulado no *Gazebo* foi estabelecida. Para resolver as equações de cinemática inversa optou-se pelo plugin TRAC-IK e para o planejamento de trajetória foi utilizada a biblioteca *OMPL*.

Um pacote capaz de comunicar o sistema de escaneamento, composto por uma câmera RGB e o pacote *Bir Marker Localization*, e o sistema de planejamento/execução de trajetória foi desenvolvido utilizando a linguagem C++. A partir da identificação da *tag ArUco*, o sistema é capaz de localizar o painel elétrico e planejar uma trajetória que leve o *endeffector* do manipulador até o alvo estabelecido.

Foram realizados 60 testes considerando diferentes posições e orientações para o painel elétrico. Os resultados alcançados mostram que Timon-HM foi capaz de realizar a tarefa em 90% dos casos, com tempo médio de execução de 24 segundos, resultados estes considerados satisfatórios para o prosseguimento do projeto.

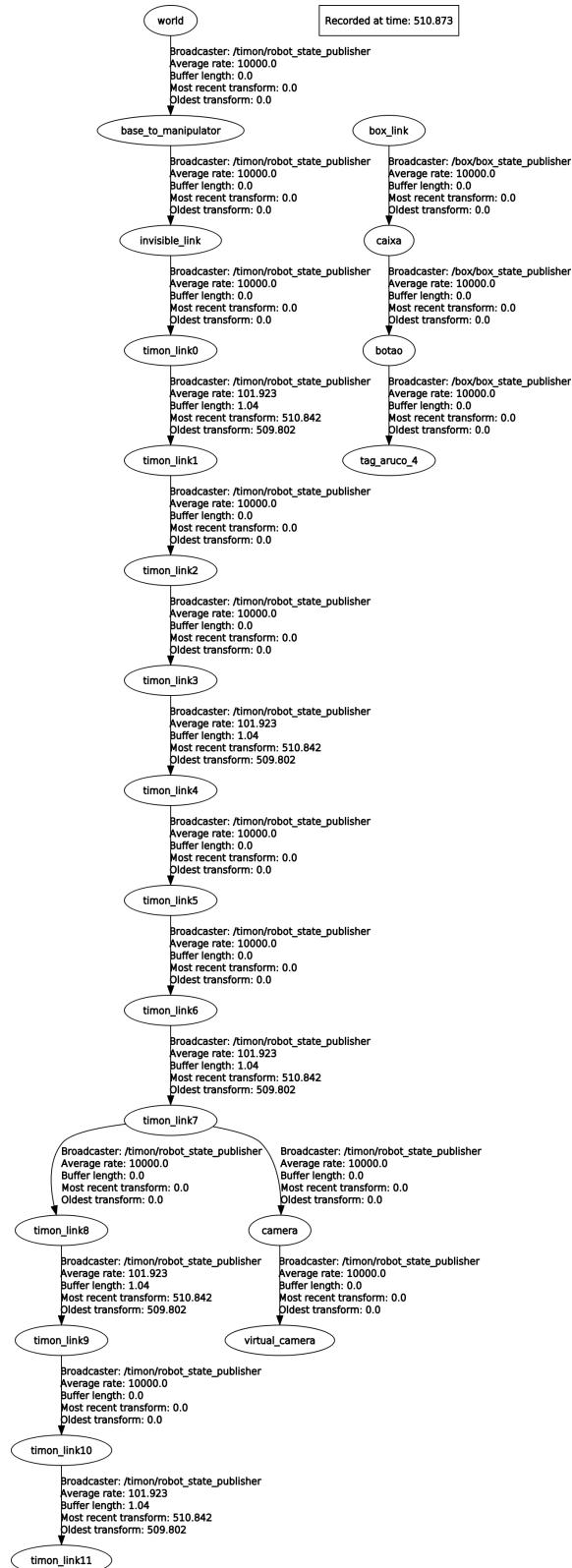
Futuramente a implementação do modelo físico sera realizada em laboratório e novos testes serão executados a fim de comprovar o bom funcionamento do sistema. Por fim, o manipulador será instalado em um robô móvel *Warthog* para autonomamente localizar e desarmar uma bomba hipotética instalada em ambiente aberto.

REFERÊNCIAS

- BEESON, P.; AMES, B. Trac-ik: An open-source library for improved solving of generic inverse kinematics. In: IEEE. *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*. [S.l.], 2015. p. 928–935. Citado na página 24.
- BIR Marker Localization. 2020. <https://github.com/Brazilian-Institute-of-Robotics/bir_marker_localization/>. Accessed: 2020-04-24. Citado na página 22.
- HERNANDEZ-MENDEZ, S. et al. Design and implementation of a robotic arm using ros and moveit! In: IEEE. *2017 IEEE International Autumn Meeting on Power, Electronics and Computing (ROPEC)*. [S.l.], 2017. p. 1–6. Citado na página 13.
- HERNÁNDEZ-ORDOÑEZ, M. et al. An education application for teaching robot arm manipulator concepts using augmented reality. *Mobile Information Systems*, Hindawi, v. 2018, 2018. Citado na página 13.
- JAVEED, A.; PRAKASH, V. G.; KULKARNI, S. P. Autonomous service robot. In: *Proceedings of the Advances in Robotics 2019*. [S.l.: s.n.], 2019. p. 1–6. Citado na página 15.
- KHAN, K. A.; KONDA, R. R.; RYU, J.-C. Ros-based control for a robot manipulator with a demonstration of the ball-on-plate task. *Advances in robotics research*, v. 2, n. 2, p. 113, 2018. Citado na página 14.
- LEITE, A. C. *Controle Híbrido de Força e Visão de um Manipulador Robótico Sobre Superfícies Desconhecidas*. Tese (Doutorado) — Tese de Mestrado, Departamento de Engenharia Elétrica, 2005. Citado na página 12.
- OLIVEIRA, G. T. d. S. et al. Projeto ótimo de robôs manipuladores 3r considerando a topologia do espaço de trabalho. Universidade Federal de Uberlândia, 2012. Citado na página 11.
- SANTOS, V. M. Robótica industrial. *Universidade de Aveiro-Departamento de Engenharia Mecânica*, 2004. Citado 2 vezes nas páginas 13 e 14.
- SUCAN, I. A.; MOLL, M.; KAVRAKI, L. E. The open motion planning library. *IEEE Robotics & Automation Magazine*, IEEE, v. 19, n. 4, p. 72–82, 2012. Citado na página 24.
- ZHANG, S.; LIN, Z.; WU, G. Motion planning of a 5-dof anthropomorphic robotic arm under ros environment. In: SPRINGER. *IFTToMM International Conference on Mechanisms, Transmissions and Applications*. [S.l.], 2019. p. 409–418. Citado na página 14.

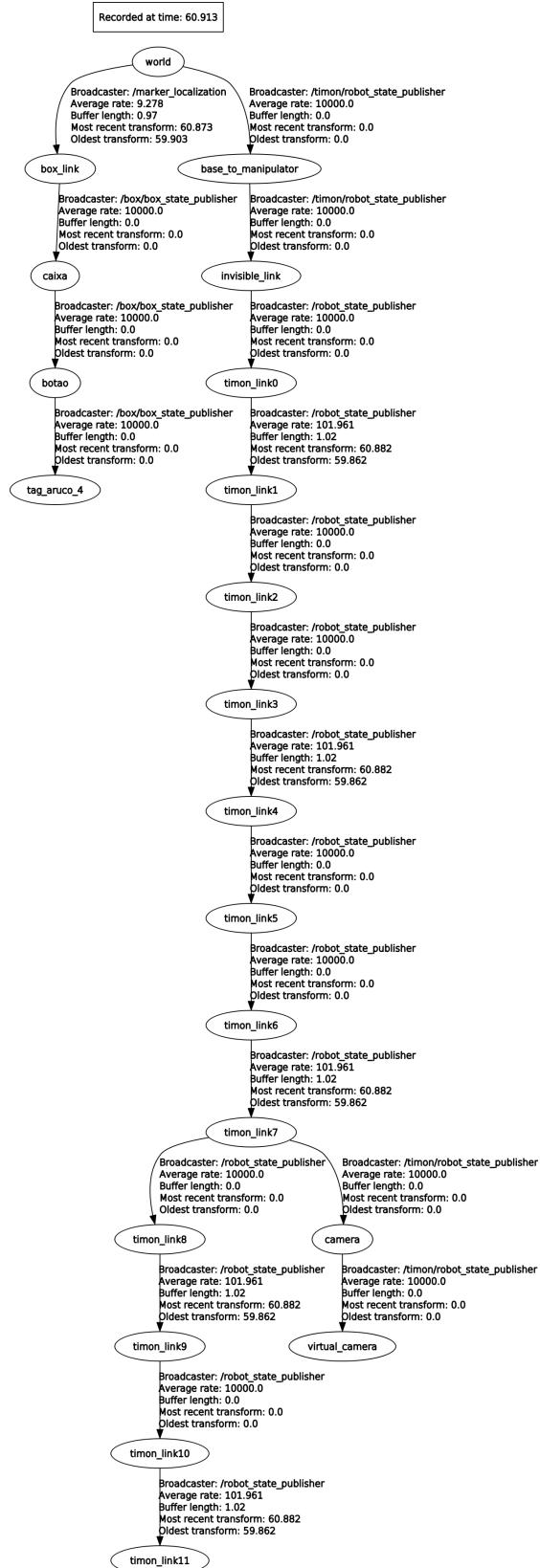
APÊNDICE A

Árvores de TF desconectadas



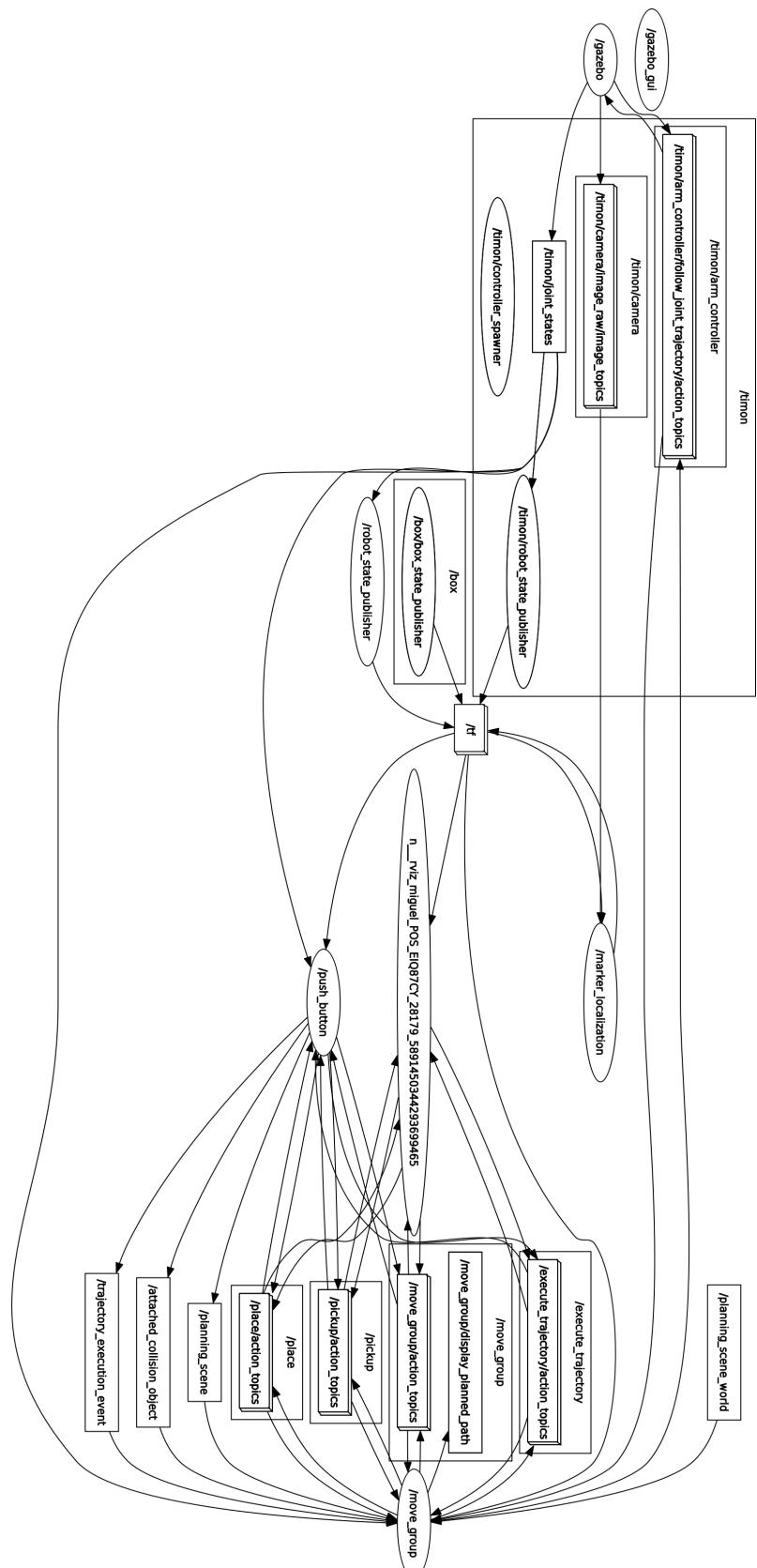
APÊNDICE B

Árvores de TF conectadas



APÊNDICE C

Diagrama de nós do sistema



APÊNDICE D

Propriedades de Massa do Timon-HM

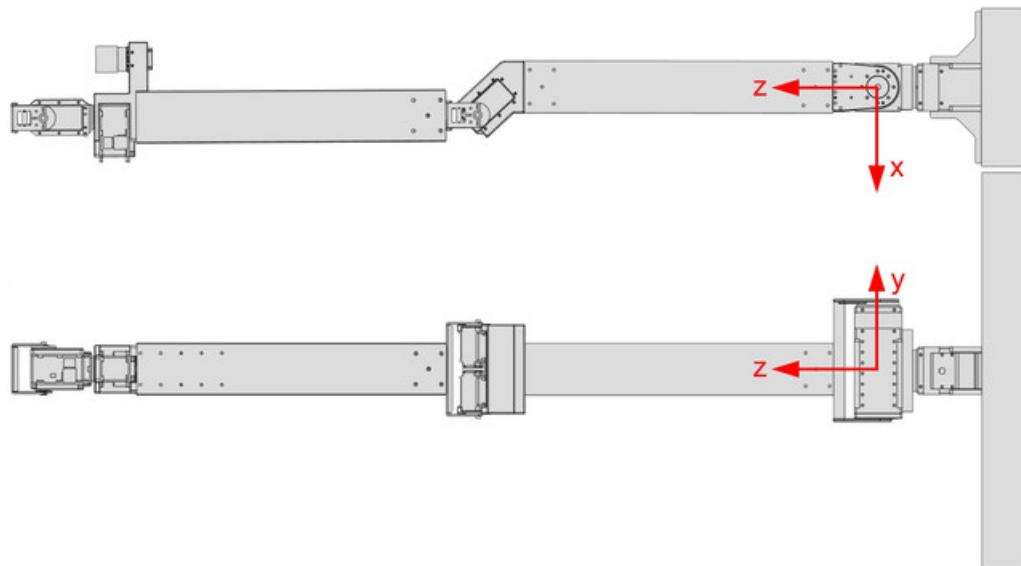


Figura 17: Coordenadas.

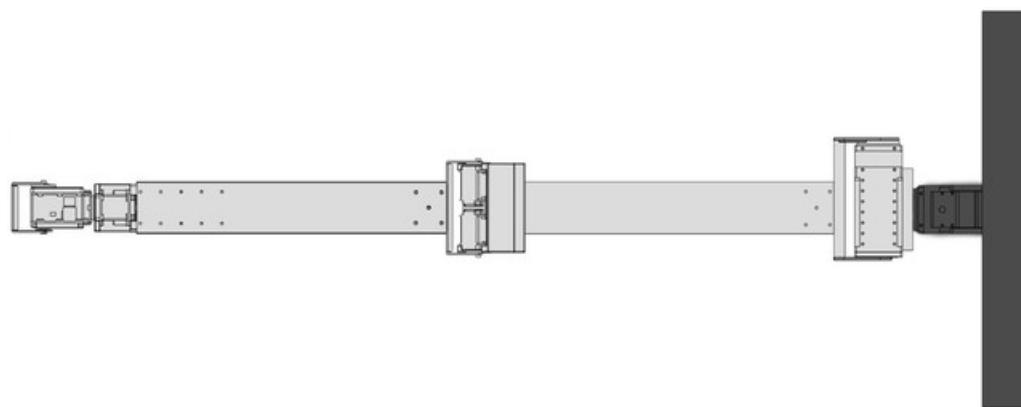


Figura 18: Link 0.

- **Massa:** 3.72264467 kg
- **Volume:** $0.00412764m^3$
- **Área:** $0.28500950m^2$
- **Centro de Massa:**
 - X: -0.00000012 m
 - Y: 0.00000000 m

- Z: 0.03490035 m

- **Momento de inércia:** kg m^2

- **L_{xx}:** 1.06821989e-2 **L_{xy}:** -2.59829620e-6 **L_{xz}:** 1.68828926e-8
- **L_{yx}:** -2.59829620e-6 **L_{yy}:** 4.86112355e-2 **L_{yz}:** 0.00000000e+0
- **L_{zx}:** 1.68828926e-8 **L_{zy}:** 0.00000000e+0 **L_{zz}:** 5.38837109e-2

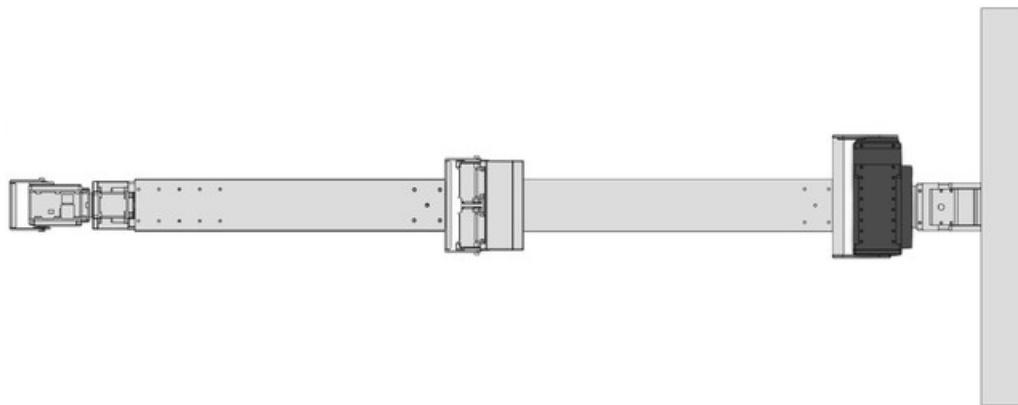


Figura 19: Link 1.

- **Massa:** 1.00643107 kg

- **Volume:** 0.00040209 m^3

- **Área:** 0.05733417 m^2

- **Centro de Massa:**

- X: 0.01039500 m

- Y: 0.00005146 m

- Z: 0.16006794 m

- **Momento de inércia:** kg m^2

- **L_{xx}:** 5.52840983e-4 **L_{xy}:** -4.82304098e-6 **L_{xz}:** -5.15873870e-5
- **L_{yx}:** -4.82304098e-6 **L_{yy}:** 1.52709797e-3 **L_{yz}:** -2.55376135e-7
- **L_{zx}:** -5.15873870e-5 **L_{zy}:** -2.55376135e-7 **L_{zz}:** 1.41817064e-3

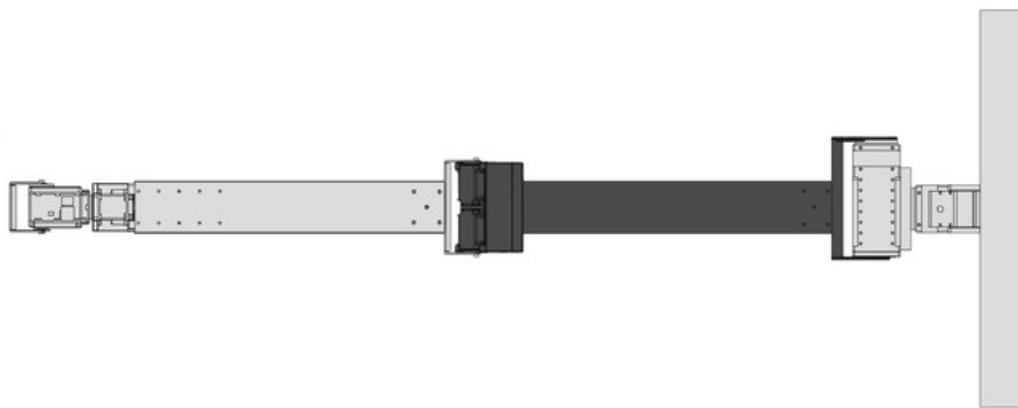


Figura 20: Link 2.

- **Massa:** 1.43140941 kg
- **Volume:** 0.00072217 m^3
- **Área:** 0.30338280 m^2
- **Centro de Massa:**
 - X: 0.00402379 m
 - Y: -0.00489519 m
 - Z: 0.42191352 m
- **Momento de inércia:** kg m^2
 - **Lxx:** 4.01177388e-2 **Lxy:** -1.99033225e-6 **Lxz:** 7.25766892e-4
 - **Lyx:** -1.99033225e-5 **Lyy:** 4.07927622e-2 **Lyz:** 1.35014731e-3
 - **Lzx:** 7.25766892e-4 **Lzy:** 1.35014731e-3 **Lzz:** 1.93910351e-3

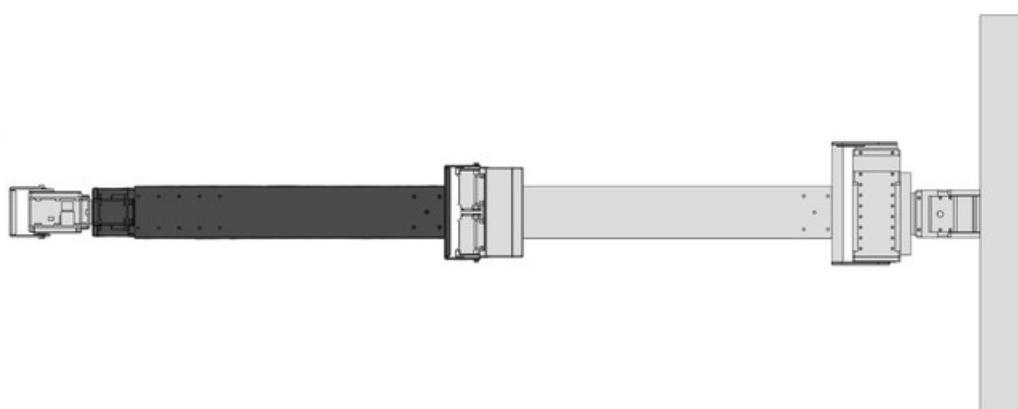


Figura 21: Link 3.

- **Massa:** 1.02430185 kg

- **Volume:** 0.00048651 m^3

- **Área:** 0.27643743 m^2

- **Centro de Massa:**

- X: 0.00177442 m
 - Y: -0.02543863 m
 - Z: 0.88744337 m

- **Momento de inércia:** kg m^2

- **L_{xx}:** 2.15420518e-2 **L_{xy}:** 4.48113214e-6 **L_{xz}:** 7.29467793e-6
 - **L_{yx}:** 4.48113214e-6 **L_{yy}:** 2.09581064e-2 **L_{yz}:** -9.20042318e-4
 - **L_{zx}:** 7.29467793e-6 **L_{zy}:** -9.20042318e-4 **L_{zz}:** 1.37743730e-3

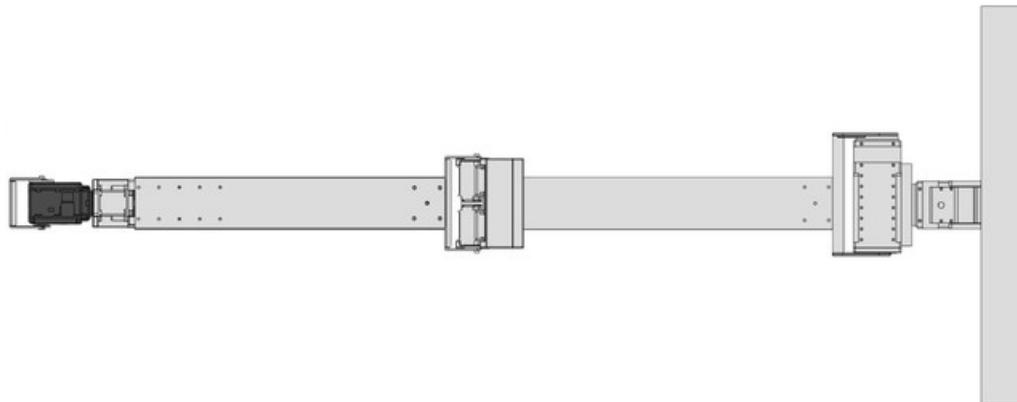


Figura 22: Link 4.

- **Massa:** 0.17387384 kg

- **Volume:** 0.00008979 m^3

- **Área:** 0.02133374 m^2

- **Centro de Massa:**

- X: 0.00204423 m
 - Y: -0.03416008 m
 - Z: 1.09140950 m

- **Momento de inércia:** kg m^2

- **L_{xx}:** 7.27594009e-5 **L_{xy}:** 3.60335785e-7 **L_{xz}:** -1.43417976e-6

- **Lyx:** 3.60335785e-7 **Lyy:** 9.11402947e-5 **Lyz:** 1.03937291e-8
- **Lzx:** -1.43417976e-6 **Lzy:** 1.03937291e-8 **Lzz:** 4.72120463e-5

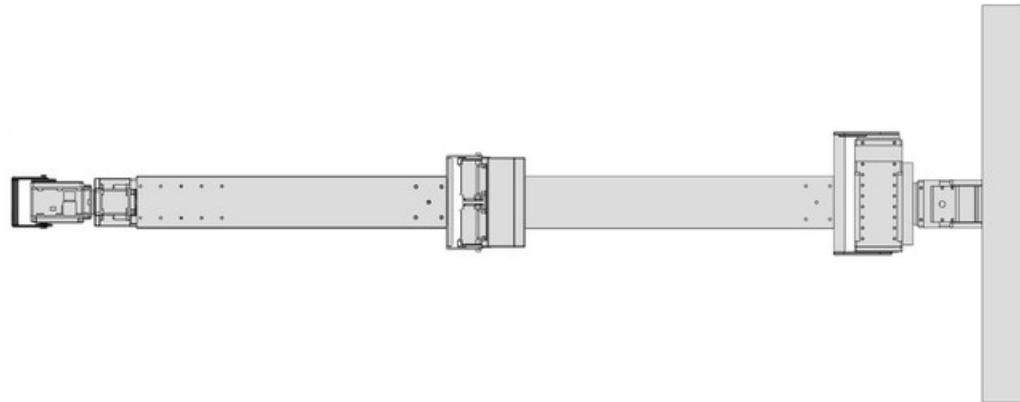


Figura 23: Link 5.

- **Massa:** 0.01873963 kg
- **Volume:** 0.00000694 m³
- **Área:** 0.00883439 m²
- **Centro de Massa:**
 - X: 0.00184227 m
 - Y: -0.03451643 m
 - Z: 1.13452531 m
- **Momento de inércia:** kg m²
 - **Lxx:** 4.49832477e-6 **Lxy:** 1.27964175e-7 **Lxz:** 9.56680130e-10
 - **Lyx:** 1.27964175e-7 **Lyy:** 1.12015451e-5 **Lyz:** 4.61798902e-9
 - **Lzx:** 9.56680130e-10 **Lzy:** 4.61798902e-9 **Lzz:** 1.08114418e-5

APÊNDICE E

Algoritmo de busca e acionamento do painel

```
#include <ros/ros.h>
#include <geometry_msgs/PoseWithCovarianceStamped.h>
#include <moveit/move_group_interface/move_group_interface.h>
#include <moveit/planning_scene_interface/
    planning_scene_interface.h>
#include <moveit_msgs/DisplayRobotState.h>
#include <moveit_msgs/DisplayTrajectory.h>
#include <moveit_msgs/AttachedCollisionObject.h>
#include <moveit_msgs/CollisionObject.h>
#include <moveit_visual_tools/moveit_visual_tools.h>
#include <trajectory_msgs/JointTrajectory.h>
#include <trajectory_msgs/JointTrajectoryPoint.h>
#include <tf/transform_listener.h>
#include <tf/tf.h>

int main(int argc, char** argv)
{
    ros::init(argc, argv, "push_button");
    ros::NodeHandle nh;
    ros::AsyncSpinner spinner(2);
    spinner.start();
    static const std::string PLANNING_GROUP = "timon_arm";
    moveit::planning_interface::MoveGroupInterface move_group(
        PLANNING_GROUP);
    moveit::planning_interface::PlanningSceneInterface
        planning_scene_interface;
    const robot_state::JointModelGroup* joint_model_group =
        move_group.getCurrentState()->getJointModelGroup(
            PLANNING_GROUP);
    namespace rvt = rviz_visual_tools;
    moveit_visual_tools::MoveItVisualTools visual_tools(
        "invisible_link");
    visual_tools.deleteAllMarkers();
    visual_tools.loadRemoteControl();
    Eigen::Isometry3d text_pose = Eigen::Isometry3d::Identity();
```

```
text_pose.translation().z() = 1.75;
visual_tools.publishText(text_pose, "Timon push_button
demonstration", rvt::WHITE, rvt::XLARGE);
visual_tools.trigger();
ROS_INFO_NAMED( "Reference frame: %s", move_group.
getPlanningFrame().c_str());
ROS_INFO_NAMED( "End effector link: %s", move_group.
getEndEffectorLink().c_str());

std::cout << move_group.getCurrentPose();

//ADDING FLOOR
// Define a collision object ROS message.
moveit_msgs::CollisionObject collision_object;
collision_object.header.frame_id = move_group.getPlanningFrame()
();
moveit_msgs::CollisionObject collision_object1;
collision_object1.header.frame_id = move_group.
getPlanningFrame();
// The id of the object is used to identify it.
collision_object.id = "floor";
collision_object1.id = "box";
// Define a box to add to the world.
shape_msgs::SolidPrimitive primitive;
primitive.type = primitive.BOX;
primitive.dimensions.resize(3);
primitive.dimensions[0] = 2.0;
primitive.dimensions[1] = 3.0;
primitive.dimensions[2] = 0.0;
shape_msgs::SolidPrimitive primitive1;
primitive1.type = primitive.BOX;
primitive1.dimensions.resize(3);
primitive1.dimensions[0] = 0.2;
primitive1.dimensions[1] = 0.2;
primitive1.dimensions[2] = 0.3;
// Define a pose for the box (specified relative to frame_id)
geometry_msgs::Pose box_pose;
box_pose.orientation.w = 0.0;
box_pose.position.x = 0.0;
```

```
box_pose.position.y = 0.0;
box_pose.position.z = -0.11;
    geometry_msgs::Pose box_pose1;
box_pose1.orientation.w = 0.0;
box_pose1.position.x = 0.0;
box_pose1.position.y = -1.06;
box_pose1.position.z = 0.0;

collision_object.primitives.push_back(primitive);
collision_object.primitive_poses.push_back(box_pose);
collision_object.operation = collision_object.ADD;
collision_object1.primitives.push_back(primitive1);
collision_object1.primitive_poses.push_back(box_pose1);
collision_object1.operation = collision_object.ADD;

std::vector<moveit_msgs::CollisionObject> collision_objects;
collision_objects.push_back(collision_object);
planning_scene_interface.addCollisionObjects(collision_objects
);

//GET MANIPULATOR STATE
robot_state::RobotState start_state(*move_group.
    getCurrentState());
move_group.setStartState(start_state);
//BEGIN POSE
geometry_msgs::Pose target_pose;
target_pose.orientation.w = 0.474989;
target_pose.orientation.x = -0.522257;
target_pose.orientation.y = 0.47662;
target_pose.orientation.z = -0.523895;
target_pose.position.x = 0.000603714;
target_pose.position.y = -0.335967;
target_pose.position.z = 0.305978;
move_group.setPoseTarget(target_pose);

moveit::planning_interface::MoveGroupInterface::Plan my_plan;
move_group.plan(my_plan);
move_group.execute(my_plan);
```

```
tf::TransformListener listener;
//GETTING BUTTON POSE
tf::StampedTransform transform;
try{
// ros::Time now = ros::Time::now();
listener.waitForTransform("/fake_botao", "/invisible_link",
                         ros::Time(0), ros::Duration(2.0));
listener.lookupTransform("/fake_botao", "/invisible_link",
                         ros::Time(0), transform);
}
catch (tf::TransformException &ex) {
ROS_ERROR("%s", ex.what());
// ros::Duration(1.0).sleep();
}
//CHECK BOX ORIENTATION
//HORIZONTAL
if (transform.getRotation().w() >= 0.99) {
robot_state::RobotState start_state6(*move_group.
getCurrentState());
move_group.setStartState(start_state6);

//POSE
target_pose.position.x = -transform.getOrigin().x();
target_pose.position.y = -transform.getOrigin().y();
target_pose.position.z = -transform.getOrigin().z();
move_group.setGoalPositionTolerance(0.001);
move_group.setGoalOrientationTolerance(0.3);
move_group.setPlanningTime(20);

move_group.setPoseTarget(target_pose);
move_group.plan(my_plan);
move_group.execute(my_plan);

//PRESS BUTTON
robot_state::RobotState start_state7(*move_group.
getCurrentState());
move_group.setStartState(start_state7);
target_pose.position.z -= 0.050;
move_group.setPoseTarget(target_pose);
```

```
move_group.plan(my_plan);
move_group.execute(my_plan);

robot_state::RobotState start_state8(*move_group.
    getCurrentState());
move_group.setStartState(start_state8);
target_pose.position.z +=0.050;
move_group.setPoseTarget(target_pose);
move_group.plan(my_plan);
move_group.execute(my_plan);
}

//VERTICAL
else{
    robot_state::RobotState start_state2(*move_group.
        getCurrentState());
    move_group.setStartState(start_state2);
    //POSE
    target_pose.orientation.w = 0;
    target_pose.orientation.x = 0.707;
    target_pose.orientation.y = 0;
    target_pose.orientation.z = -0.707;
    target_pose.position.x = transform.getOrigin().x();
    target_pose.position.y = -transform.getOrigin().z();
    target_pose.position.z = -transform.getOrigin().y();

    move_group.setGoalPositionTolerance(0.001);
    move_group.setGoalOrientationTolerance(0.3);
    move_group.setPlanningTime(20);
    move_group.setPoseTarget(target_pose);

    move_group.plan(my_plan);
    move_group.execute(my_plan);

    //PRESS BUTTON
    robot_state::RobotState start_state3(*move_group.
        getCurrentState());
    move_group.setStartState(start_state3);
    target_pose.position.y -=0.051;
    move_group.setPoseTarget(target_pose);
```

```
move_group.plan(my_plan);
move_group.execute(my_plan);

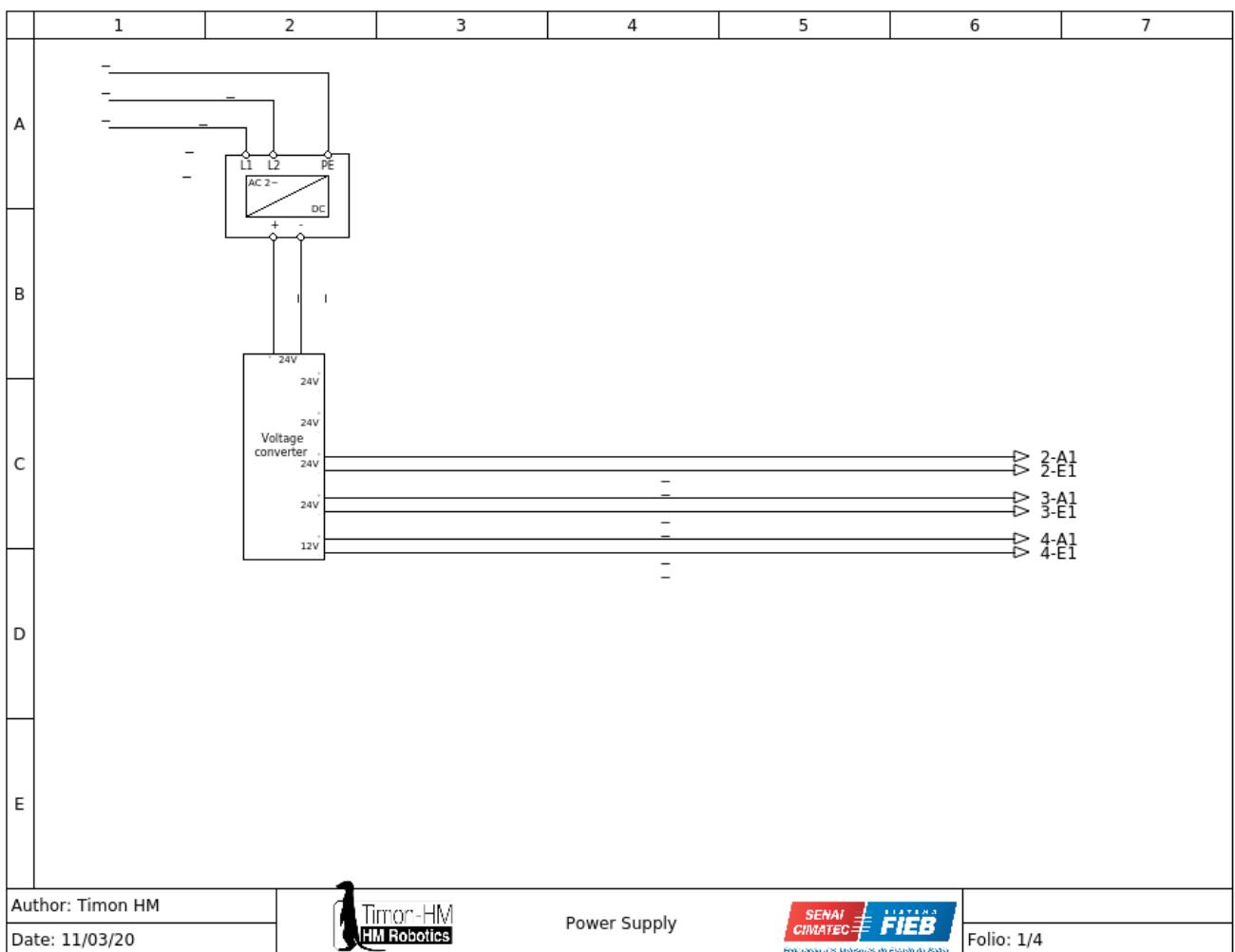
robot_state::RobotState start_state4(*move_group.
    getCurrentState());
move_group.setStartState(start_state4);
target_pose.position.y +=0.051;
move_group.setPoseTarget(target_pose);
move_group.plan(my_plan);
move_group.execute(my_plan);
}

//BACK TO UP POSITION
robot_state::RobotState start_state5(*move_group.
    getCurrentState());
move_group.setStartState(start_state5);
moveit::core::RobotStatePtr current_state3 = move_group.
    getCurrentState();
std::vector<double> joint_group_positions3;
current_state3->copyJointGroupPositions(joint_model_group,
    joint_group_positions3);
joint_group_positions3[0] = 0;
joint_group_positions3[1] = 0;
joint_group_positions3[2] = 0;
joint_group_positions3[3] = 0;
joint_group_positions3[4] = 0;
move_group.setJointValueTarget(joint_group_positions3);
move_group.plan(my_plan);
move_group.execute(my_plan);
//REMOVE OBJECTS
std::vector<std::string> object_ids;
object_ids.push_back(collision_object.id);
object_ids.push_back(collision_object1.id);
planning_scene_interface.removeCollisionObjects(object_ids);

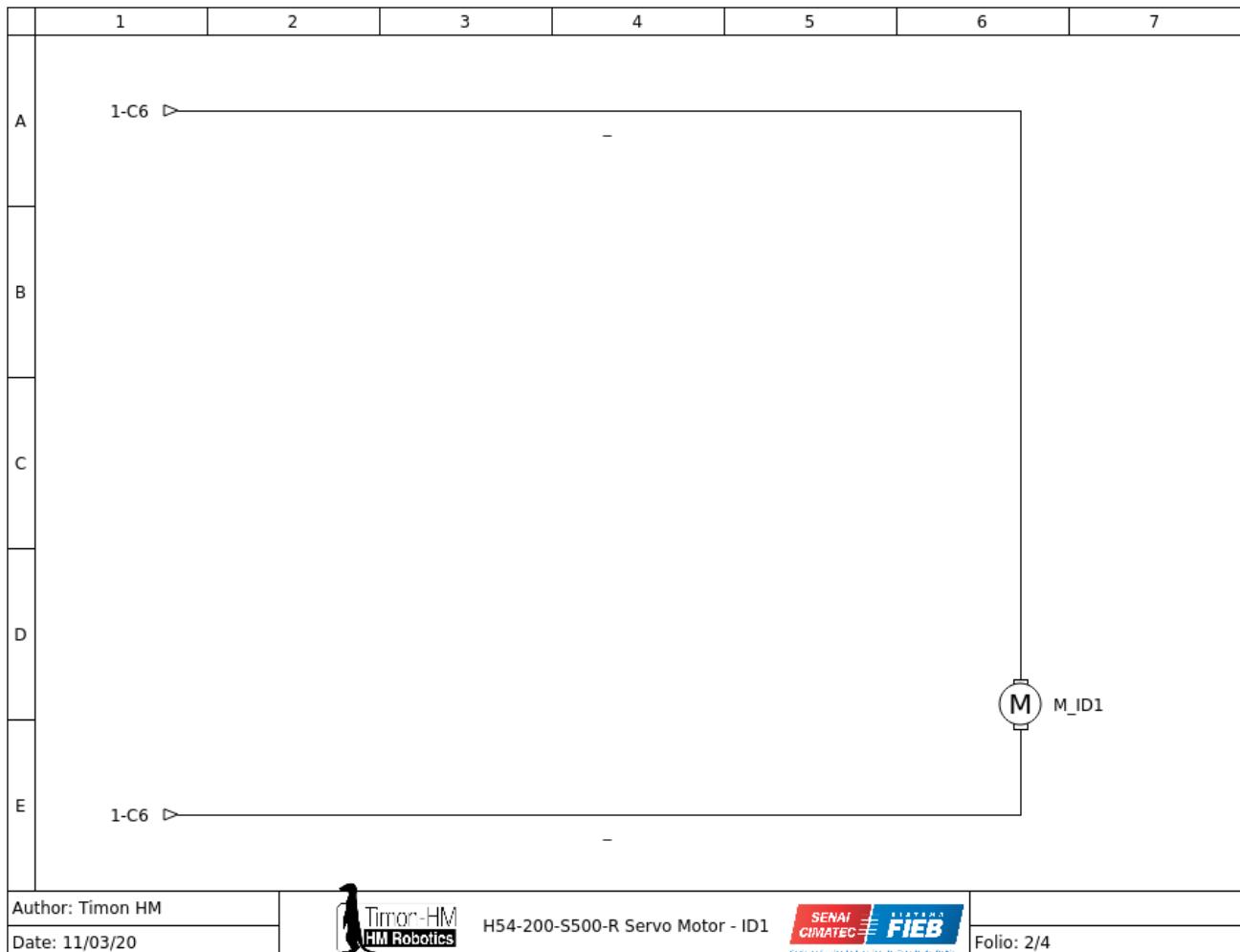
ros::shutdown();
return 0;
}
```

APÊNDICE F

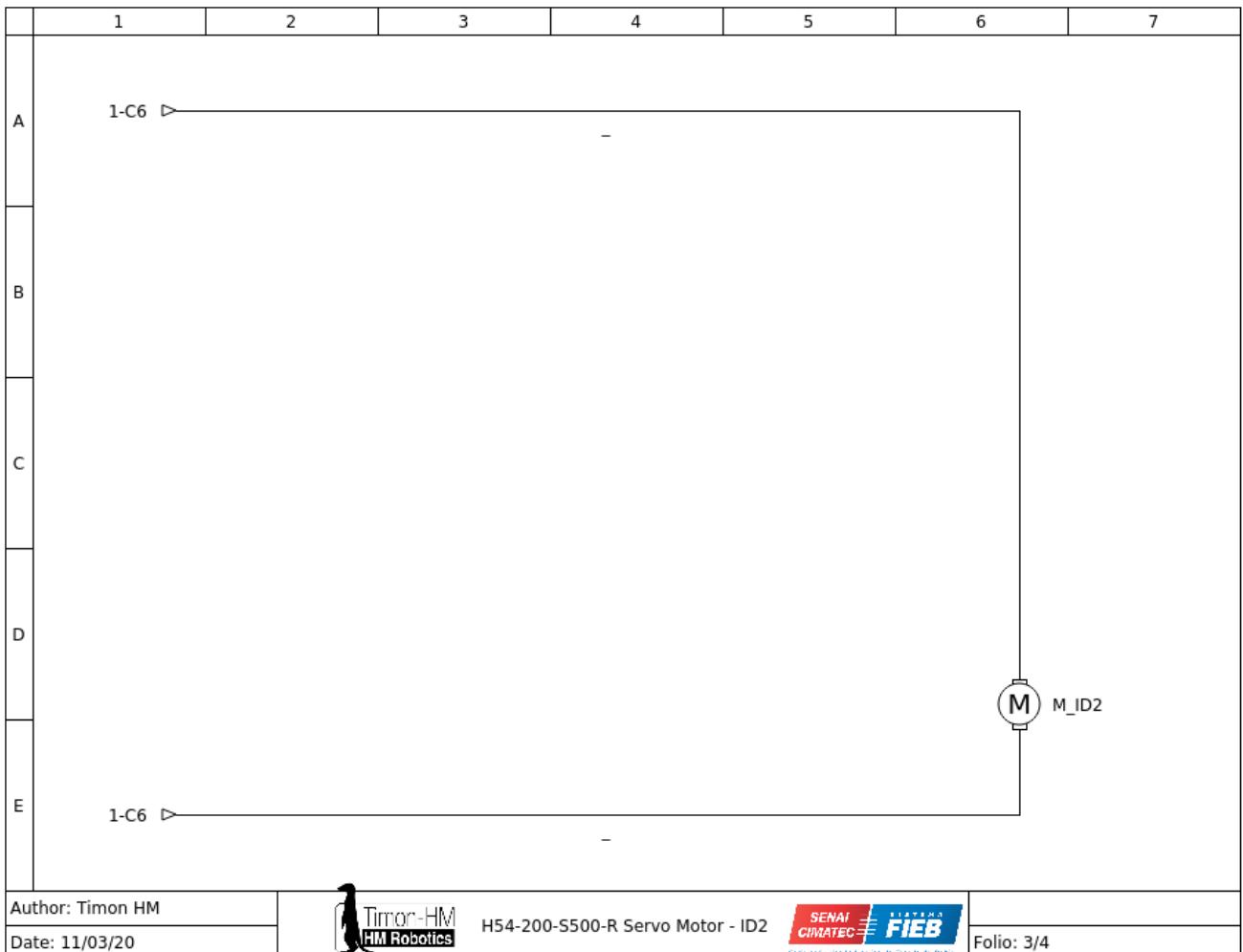
Diagrama elétrico do Timon-HM



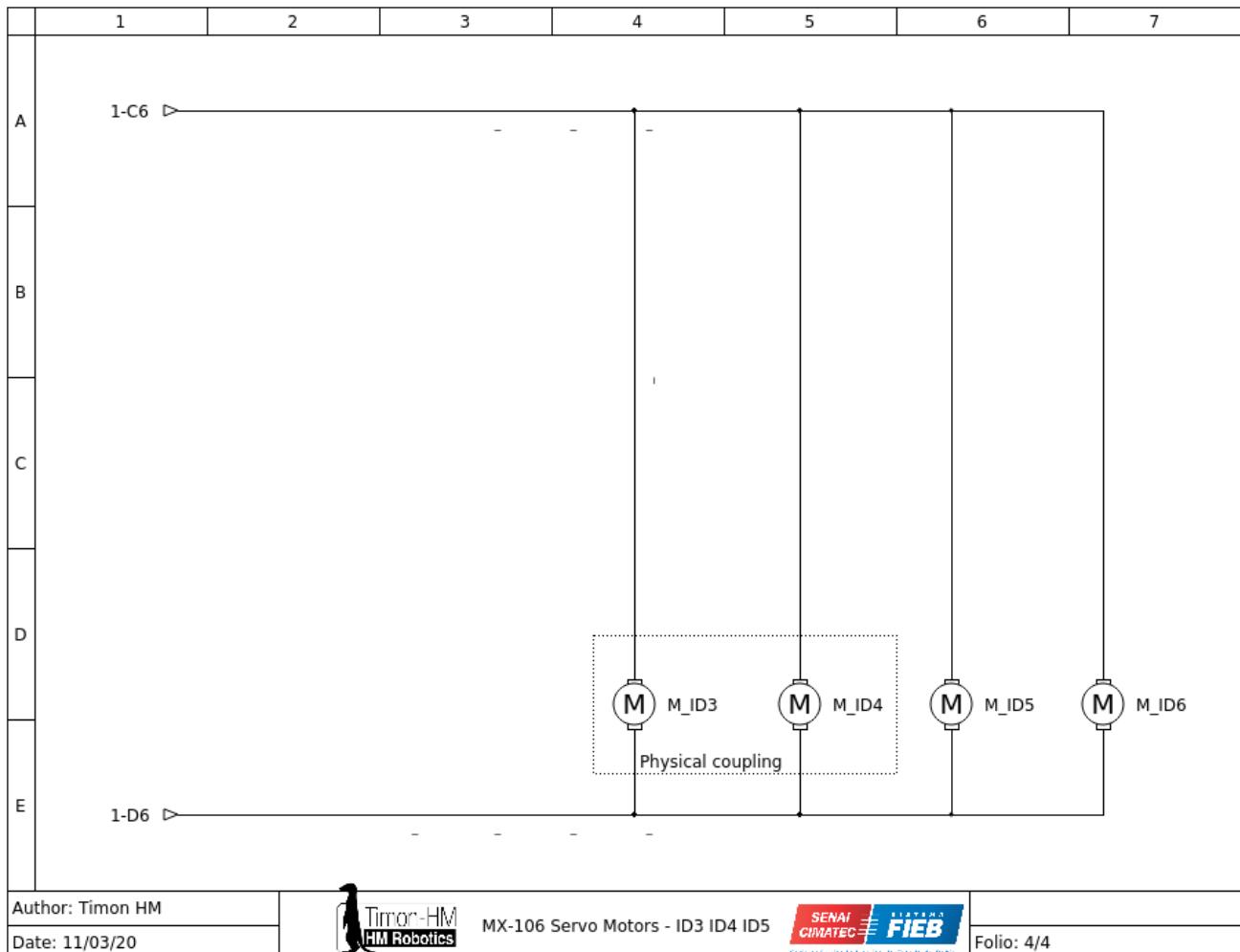
Apêndice A



Apêndice A

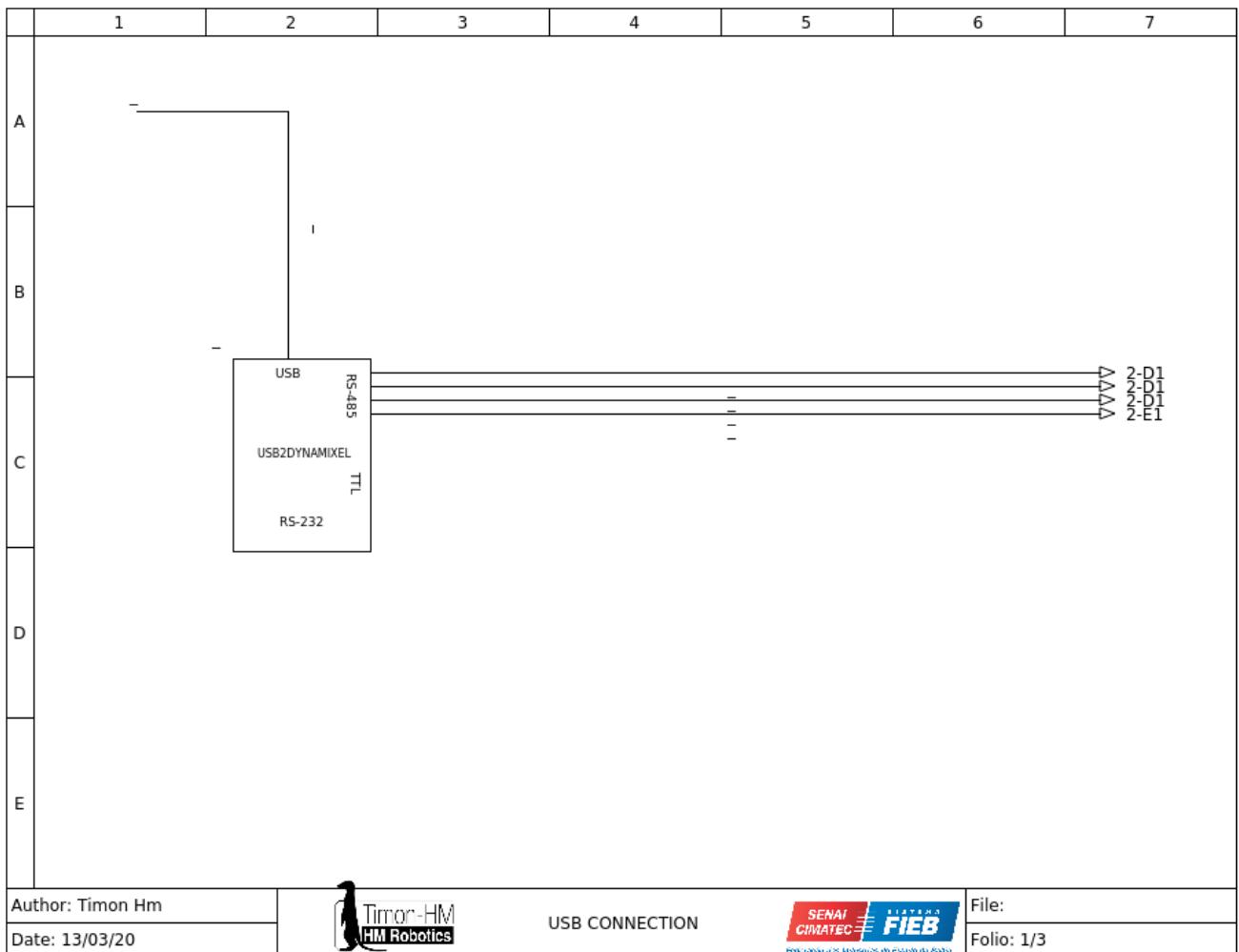


Apêndice A



APÊNDICE G

Diagrama de conexão do Timon-HM



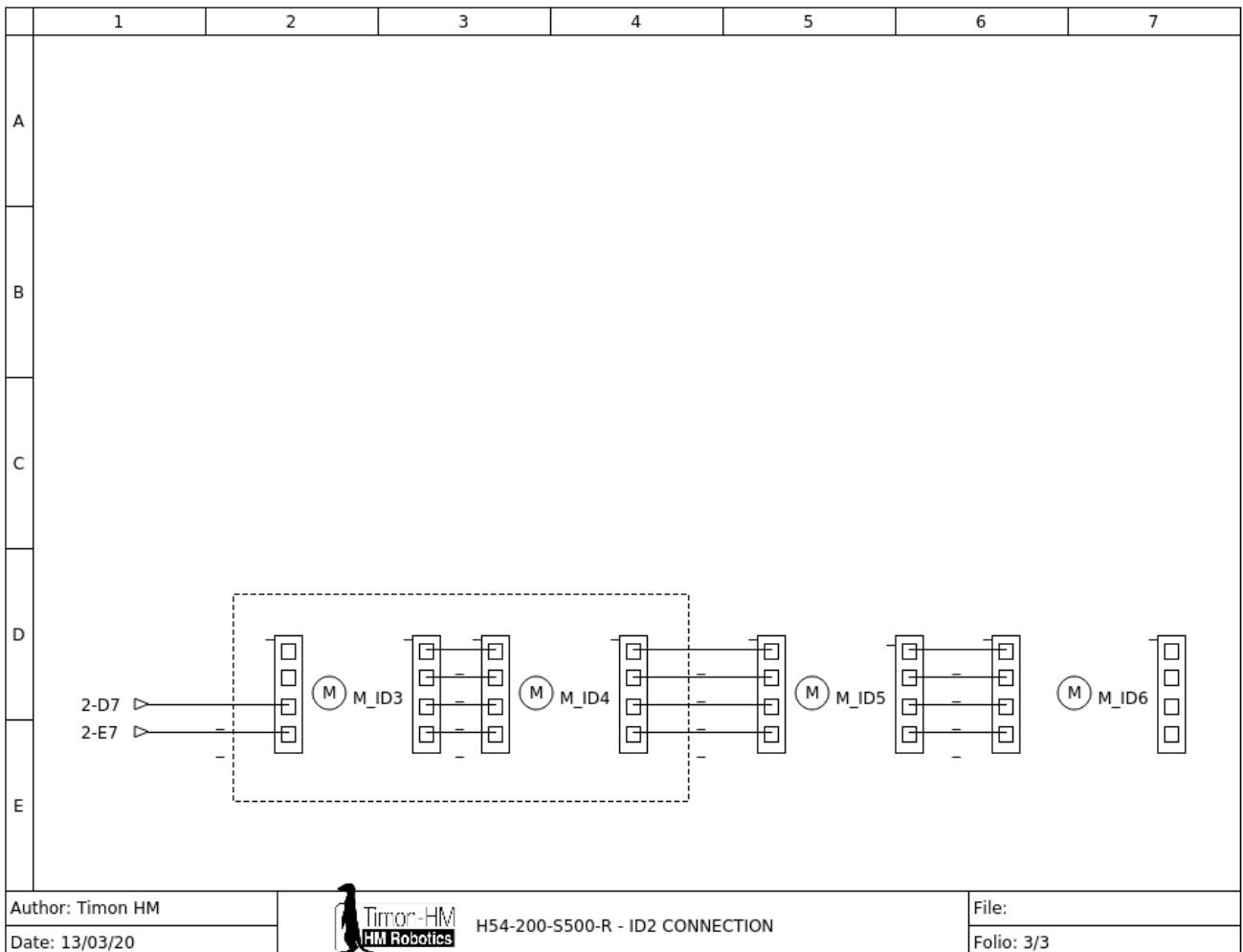
Apêndice A

	1	2	3	4	5	6	7
A							
B							
C							
D	1-B7	▷					
	1-C7	▷	-				
	1-C7	▷	-				
	1-C7	▷	-				
		-					
E							

Diagram showing connections from pins 1-B7, 1-C7, and 1-D7 to terminal blocks M_ID1 and M_ID2. M_ID1 has four outputs, and M_ID2 has three outputs. The outputs from M_ID1 connect to 3-D1 and 3-E1. The outputs from M_ID2 connect to 3-D1 and 3-E1.

Author: Timon HM	Timor-HM HM Robotics	H54-200-S500-R - ID1 CONNECTION	SENAI CIMATEC FIEB	File:
Date: 13/03/20				Folio: 2/3

Apêndice A



ANEXO A

Especificações da câmera Dalsa Genie Nano

Specifications: M2590, M2590-NIR, C2590

Supported Features	M2590, M2590-NIR	Nano-C2590	
Resolution	2592 x 2048		
Sensor	OnSemi Python5000 P1 (5.1M)		
Pixel Size	4.8 µm x 4.8 µm		
Shutter type	Full frame electronic global shutter function		
Full Well charge	10ke (max)		
Firmware option (Field programmable)	Standard Design Monochrome (factory default)	Standard Design Bayer (factory default)	RGB-Output Design
Max. Internal Frame Rate Full Resolution (2592 x 2048)	51.8 fps (Fast Readout Enable) 24.7 fps (Normal Readout Enable)		
Maximum Sustained Frame Rate Output (with TurboDrive v1)*	42.7 fps (8-bit) 24.9 fps (10-bit)		N/A
Maximum Sustained Frame Rate Output (without TurboDrive)	22 fps (8-bit)		5.5 fps (RGBA) 8.7 fps (RGB) 11 fps (Yuv422) 22 fps (8-bit mono)
Pixel Data Formats	Mono 8-bit Mono 10-bit	Bayer 8-Bit Bayer 10-Bit	RGBA 32-bit RGB 24-bit Yuv422 16-bit Mono 8-bit
Trigger to Exposure Minimum delay (Synchronous Exposure Alignment)	8 µs if exposureAlignment = Synchronous With No Overlap between the new exposure and the previous readout 26.2 µs if exposureAlignment = Synchronous With Overlap between the new exposure and the previous readout		
Trigger to Exposure Minimum delay (Reset Exposure Alignment)	3 µs		
Trigger to Exposure Start jitter (best case with Synchronous Exposure Alignment)	Up to 1 line time		
Trigger to Exposure Start jitter (Reset Exposure Alignment) †	0 µs		
Exposure Time Minimum (see "exposureTimeActual" in Sensor Control)	87 µs (increment steps of 1µs)		
Min. Time from End of Exposure to Start of Next Exposure (second frame)	49 µs – Normal Readout 47 µs – Fast Readout		
Horizontal Line Time:	11.33 µs – Normal Readout 9.33 µs – Fast Readout		
Readout Time	23242 µs – Normal Readout for 2592 x 2048 Add 76µs when overlapping Exposure and Readout 19142 µs – Fast Readout for 2592 x 2048 Add 64µs when overlapping Exposure and Readout <i>Specifically: (Horizontal line time at current resolution * number of lines) + (3 * (line time of the 2590 model))</i>		
Auto-Brightness	Yes , with Auto-Exposure and AGC (FPGA Gain)		
Black offset control	Yes (in DN)		

Gain Control	In-sensor Analog Gain (1.0x to 8x) in 11 gain steps (1.0, 1.14, 1.33, 1.6, 2.0, 2.29, 2.67, 3.2, 4.0, 5.33, 8.0) In-sensor Digital Gain (1x to 32x) in 0.01x steps In-FPGA Digital Gain (1x to 4x) in 0.007x steps	
Binning Support	Yes In-FPGA (summing and average, 2x2, 4x4) Yes In- Sensor (averaging 2x2)	No
Color Correction Support	No	Yes
Decimation Support	No	
Defective Pixel Replacement	Yes, up to 512 positions	
Image Correction	No	
Image Flip Support	Yes, In-Sensor, Vertical Only	
Multi-ROI Support	Yes, in Sensor, up to 16 ROI (mutually exclusive with binning)	
On-Board Image Memory	90MB	
Output Dynamic Range (dB)	62.1 dB (in 10-Bit Pixel Format)	
SNR (dB)	39.8 dB (in 10-Bit Pixel Format)	

*TurboDrive internal limitation of 250MB/sec

† Note: The actual internal minimum exposure may be different than what is programmed. Use the feature "exposureTimeActual" from the [Sensor Control](#) category to read back the actual sensor exposure. The exposure start sensor event is delayed 4 µs from the actual start.

Firmware Files for Models 1280, 1930, 2590

The latest firmware files for all Nano models are available on the Teledyne DALSA support web site:
<http://www.teledynedalsa.com/imaging/support/downloads/firmware/>

The firmware files for these models are listed below. The xx denotes the build number.

M1280, M1930, M2590

- Standard
"Genie_Nano_OnSemi_Python_0.3M-0.5M-1.3M-2M-5M_Mono_STD_Firmware_5CA18.xx.cbf"

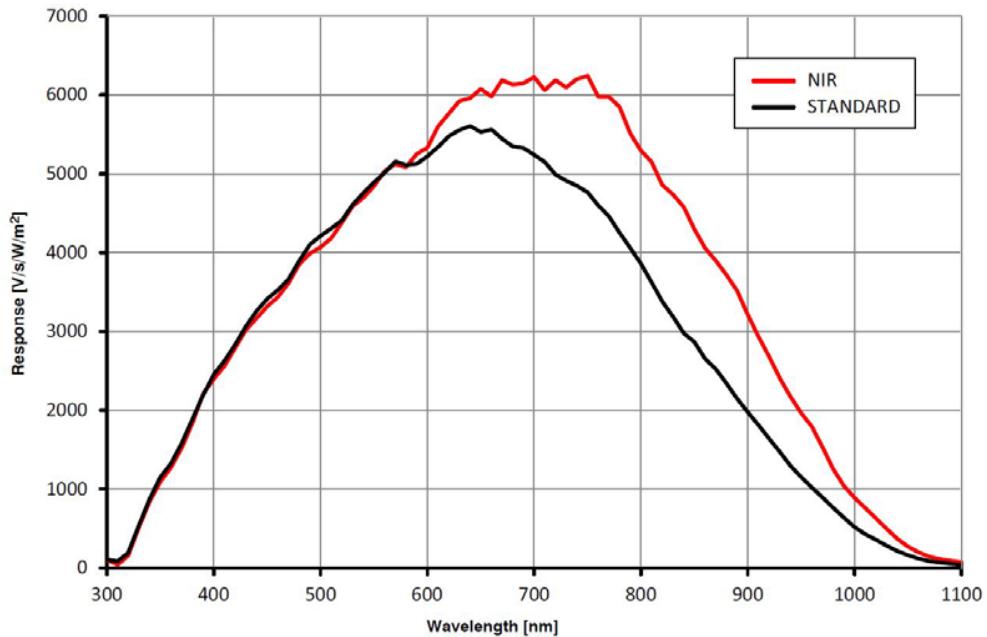
C1280, C1930, C2590

- Bayer Output
"Genie_Nano_OnSemi_Python_0.3M-0.5M-1.3M-2M-5M_Bayer_STD_Firmware_6CA18.xx.cbf"
- RGB Output
"Genie_Nano_OnSemi_Python_0.3M-0.5M-1.3M-2M-5M_RGB_Output_Firmware_6CA18.xx.cbf"

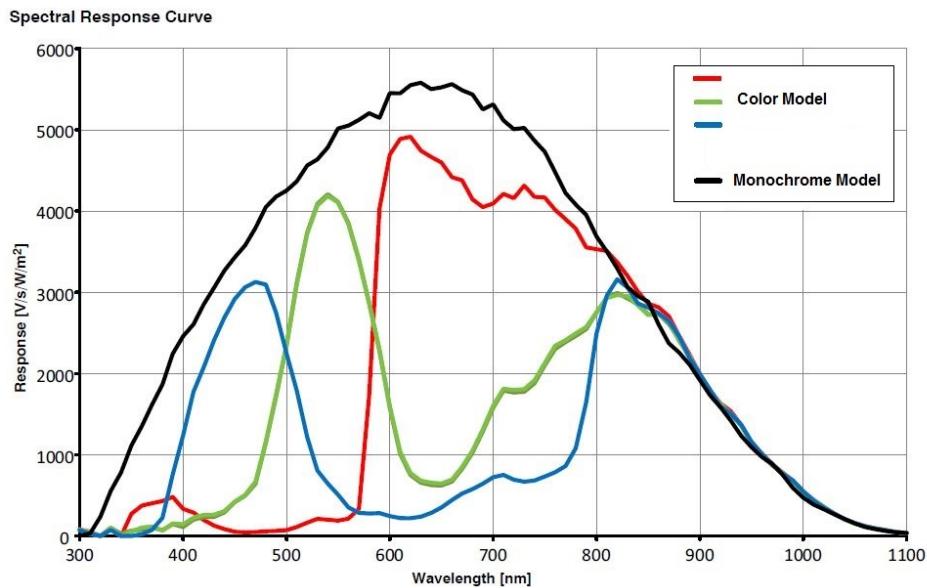
Spectral Response (Python 4.8 µm series)

Model specific specifications and response graphics for the On-Semi Python (VGA to 5M) series are provided here. The response curves describe the sensor, excluding lens and light source characteristics.

On-Semi Python Series (with 4.8 µm pixels) — Monochrome and NIR



On-Semi Python Series (with 4.8 µm pixels) — Monochrome and Color



ANEXO B

Especificações do motor Dynamixel MX-106

Item	Especificações
MCU	ARM CORTEX-M3 (72 [MHz], 32 Bit)
Sensor de posição	Encoder absoluto (sem contato) (12 Bit, 360 [°]) Fabricante: AMS , Numero da peça : AS5045
Motor	Coreless (Maxon)
Taxa de transmissão	8,000 [bps] ~4.5 [Mbps]
Algoritmo de controle	Controle PID
Resolução	4096 [pulse/rev]
Folga	20 [arcmin] (0.33 [°])
Modo operacional	Modo junta (0 ~ 360 [°]) Modo roda (Curso sem fim)
Peso	165 [g]
Dimensões (c x l x h)	40.2 x 65.1 x 46 [mm]
Relação de engrenagem	225 : 1
Torque de Parada	8.0 [Nm] (at 11.1 [V], 4.8 [A]) 8.4 [Nm] (at 12.0 [V], 5.2 [A]) 10.0 [Nm] (at 14.8 [V], 6.3 [A])
Velocidade (sem carga)	41 [rev/min] (at 11.1 [V]) 45 [rev/min] (at 12.0 [V]) 55 [rev/min] (at 14.8 [V])
Carga radial	40 [N] (10 [mm] afastado da face do eixo)
Carga axial	20 [N]
Temperatura de operação	-5 ~+80 [°C]
Tensão de entrada	10.0 ~14.8 [V] (Recomendado : 12.0 [V])
Sinal de Comando	Pacote digital
Tipo de protocolo	RS485 Comunicação serial assíncrona com 8bit, 1stop, sem paridade
Conexão física	RS485 Barramento multiponto
ID	254 ID (0 ~253)
Feedback	Posição, temperatura, carga, tensão de entrada, etc
Material	engrenagem toda de metal Plástico de engenharia (frente, meio e verso) Metal (Frente)
Corrente de repouso	100 [mA]

Tabela 12: Especificações do motor *Dynamixel MX-106*

Enter Search Terms



DYNAMIXEL

PLATFORM

STEAM

SOFTWARE

PARTS

FAQ

Apêndice A

MX-106T/R

1. Specifications
2. Control Table
3. How to Assemble
4. Maintenance
5. Reference

[Edit on GitHub](#)

ROBOTIS



Robot Source



GitHub



MX-106R, MX-106T

NOTE : Compliance has been replaced with PID Gains.**NOTE :** Although the MX-106T (TTL) and MX-106R (RS-485) differ in communications protocols both have the same features and perform equally. (TTL uses 3-pin connectors while RS-485 uses 4)**NOTE :** In order to use Protocol 2.0, please update the firmware to V39 or above. ([Update firmware](#) using R+ Manager 2.0)**Dynamixel MX Series (Protocol v2.0) F/W Recovery****WARNING :** For MX-106(2.0) Protocol, please refer to the [MX-106\(2.0\) Control Table](#) as they are different.**1. Specifications**

Item	Specifications
MCU	ARM CORTEX-M3 (72 [MHz], 32Bit)
Position Sensor	Contactless absolute encoder (12Bit, 360 [°]) Maker : ams(www.ams.com), Part No : AS5045
Motor	Coreless(Maxon)
Baud Rate	8,000 [bps] ~ 4.5 [Mbps]
Control Algorithm	PID control
Resolution	4096 [pulse/rev]
Backlash	20 [arcmin] (0.33 [°])



Apêndice A

MX-106T/R

- 1. Specifications
- 2. Control Table
- 3. How to Assemble
- 4. Maintenance
- 5. Reference

TOP

Item	Specifications
Operating Mode	Joint Mode (0 ~ 360 [°]) Wheel Mode (Endless Turn)
Weight	165 [g]
Dimensions (W x H x D)	40.2 x 65.1 x 46 [mm]
Gear Ratio	225 : 1
Stall Torque	8.0 [Nm] (at 11.1 [V], 4.8 [A]) 8.4 [Nm] (at 12[V], 5.2 [A]) 10.0 [Nm] (at 14.8 [V], 6.3 [A])
No Load Speed	41 [rev/min] (at 11.1 [V]) 45 [rev/min] (at 12 [V]) 55 [rev/min] (at 14.8 [V])
Radial Load	1 40 [N] (10 [mm] away from the horn)
Axial Load	1 20 [N]
Operating Temperature	-5 ~ +80 [°C]
Input Voltage	10.0 ~ 14.8 [V] (Recommended : 12.0 [V])
Command Signal	Digital Packet
Protocol Type	TTL Half Duplex Asynchronous Serial Communication with 8bit, 1stop, No Parity RS485 Asynchronous Serial Communication with 8bit, 1stop, No Parity
Physical Connection	RS485 / TTL Multidrop Bus
ID	254 ID (0 ~ 253)
Feedback	Position, Temperature, Load, Input Voltage, etc
Material	Full Metal Gear Engineering Plastic(Front, Middle, Back) 1 Metal(Front)
Standby Current	100 [mA]

1 Applies to aluminum housing products(MX-28AR/AT, MX-64AR/AT, MX-106R/T).

**DANGER**

(May cause serious injury or death)

- Never place items containing water, flammables, and solvents near product.
- Never place fingers, arms, toes, and other body parts near product during operation.
- Cut power off if product emits strange odors or smoke.
- Keep product out of reach of children.
- Check the power polarity before wiring.

**CAUTION**

(May cause injury or damage to product)

- Do not operate the product at a temperature exceeding -5 ~ +80 [°C] range.
- Do not insert sharp blades nor pins during product operation.

**ATTENTION**

(May cause injury or damage to product)

- Do not disassemble or modify product.
- Do not drop or apply strong shock to product.

Enter Search Terms



Apêndice A

MX-106T/R

1. Specifications

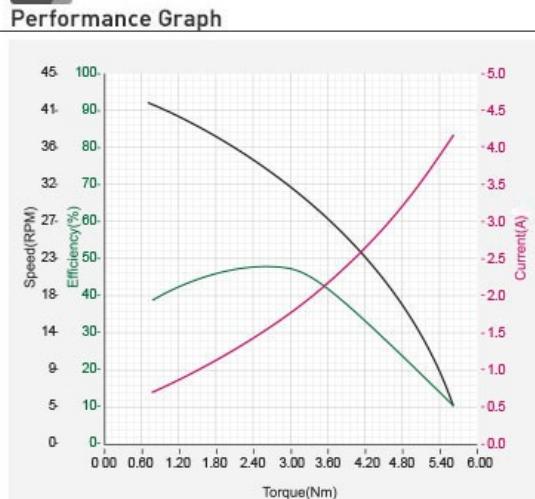
2. Control Table

3. How to Assemble

4. Maintenance

5. Reference

1. 1. Performance Graph



TOP

NOTE : The Max Torque and the Stall Torque of Performance Graph are different in measurement methods. Stall torque is a measured value of the momentary torque that it can reach. This is generally how RC servos are measured. The Performance graph is also called as N-T curves, which is measured with the gradually increasing load. The actual motor operation environment is closer to the performance graph, not stall torque method. For this reason, the performance graph is broadly used in the industrial field. Generally, Max Torque of the Performance Graph is less than the Stall Torque.

CAUTION : When supplying power

- It is recommended using ROBOTIS controller or SMPS2DYNAMIXEL.
- Do not connect or disconnect DYNAMIXEL when power is being supplied.

2. Control Table

The Control Table is a structure of data implemented in the device. Users can read a specific Data to get status of the device with Read Instruction Packets, and modify Data as well to control the device with WRITE Instruction Packets.

2. 1. Control Table, Data, Address

The Control Table is a structure that consists of multiple Data fields to store status or to control the device. Users can check current status of the device by reading a specific Data from the Control Table with Read Instruction Packets. WRITE Instruction Packets enable users to control the device by changing specific Data in the Control Table. The Address is a unique value when accessing a specific Data in the Control Table with Instruction Packets. In order to read or write data, users must designate a specific Address in the Instruction Packet. Please refer to [Protocol 1.0](#) for more details about Instruction Packets.

NOTE : Two's complement is applied for the negative value. For more information, please refer to [Two's complement](#) from Wikipedia.

2. 1. 1. Area (EEPROM, RAM)

The Control Table is divided into 2 Areas. Data in the RAM Area is reset to initial values when the power is reset(Volatile). On the other hand, data in the EEPROM Area is maintained even when the device is powered off(Non-Volatile).

Data in the EEPROM Area can only be written to if Torque Enable(24) is cleared to '0'(Off).

ANEXO C

Especificações do motor Dynamixel H54-S500-R

Item	Especificações
MCU	ARM CORTEX-M4 (168 [MHz], 32Bit)
Motor	BLDC (Maxon)
Taxa de transmissão	9,600 [bps] ~10.5 [Mbps]
Modos de operação	Modo de controle de torque Modo de controle de velocidade Modo de controle de posição Modo de controle de posição estendida Modo de controle PWM (modo de controle de tensão)
Peso	855 [g]
Dimensões (c x l x h)	54 x 126 x 54 [mm]
Resolução	1,003,846 [pulse/rev]
Relação de engrenagem	501.923 : 1
Folga	<6 [arcmin], 0.1 [°]
Carga radial	370 [N] (10 [mm] afastado da face do eixo)
Carga axial	130 [N]
Velocidade (sem carga)	33.1 [rev/min]
Corrente (sem carga)	1.65 [A]
Velocidade contínua	29.0 [rev/min]
Torque contínuo	44.7 [Nm]
Corrente contínua	9.3 [A]
Saída	200 [W]
Temperatura de operação	5 ~55 [°C]
Tensão operacional	24.0 [V]
Sinal de comando	Pacote digital
Tipo de protocolo	RS485 Comunicação Serial Assíncrona (8bit, 1stop, Sem paridade)
Conexão física	RS485 Barramento multiponto
ID	253 ID (0 ~252)
Corrente de repouso	40 [mA]

Tabela 13: Especificações do motor *Dynamixel H54-S500-R*

Enter Search Terms



AM

SOFTWARE

PARTS

FAQ

H54-200-S500-R

1. Specifications

[1. 1. Performance Graph](#)[2. Control Table](#)[3. How to Assemble](#)[4. Maintenance](#)[5. Reference](#)

ROBOTIS



Robot Source

GitHub
TOP[Edit on GitHub](#)**H54-200-S500-R**

1. Specifications

Item	Specifications
MCU	ARM CORTEX-M4 (168 [MHz], 32Bit)
Motor	BLDC(Maxon)
Baud Rate	9,600 [bps] ~ 10.5 [Mbps]
Operating Modes	Torque Control Mode Velocity Control Mode Position Control Mode Extended Position Control Mode
Weight	855 [g]
Dimensions (W x H x D)	54 x 126 x 54 [mm]
Resolution	501,923 [pulse/rev]
Gear Ratio	501.923 : 1
Backlash	< 6 [arcmin], 0.1 [°]
Radial Load	370 [N] (10 [mm] away from the horn)
Axial Load	130 [N]
No Load Speed	33.1 [rev/min]
No Load Current	1.65 [A]
Continuous Speed	29.0 [rev/min]
Continuous Torque	44.7 [Nm]
Continuous Current	9.3 [A]
Output	200 [W]
Operating Temperature	5 ~ 55 [°C]
Input Voltage	24.0 [V]
Command Signal	Digital Packet

Apêndice A

Enter Search Terms

**H54-200-S500-R****1. Specifications****1. 1. Performance Graph****2. Control Table****3. How to Assemble****4. Maintenance****5. Reference**

Item	Specifications
Protocol Type	RS485 Asynchronous Serial Communication (8bit, 1stop, No Parity)
Physical Connection	RS485 Multidrop Bus
ID	253 ID (0 ~ 252)
Standby Current	80 [mA]

TOP

[1] These specifications are calculated based on the specifications of the core motor.

Please consult ROBOTIS for the long term use or special use, or else refer to the Performance Graph for general use.

**DANGER**

(May cause serious injury or death)

- Never place items containing water, flammables, and solvents near product.
- Never place fingers, arms, toes, and other body parts near product during operation.
- Cut power off if product emits strange odors or smoke.
- Keep product out of reach of children.
- Check the power polarity before wiring.

**CAUTION**

(May cause injury or damage to product)

- Do not operate the product at a temperature exceeding 5 ~ 55 [°C] range.
- Do not insert sharp blades nor pins during product operation.

**ATTENTION**

(May cause injury or damage to product)

- Do not disassemble or modify product.
- Do not drop or apply strong shock to product.

1. 1. Performance Graph

Apêndice A

Enter Search Terms

**H54-200-S500-R**

1. Specifications

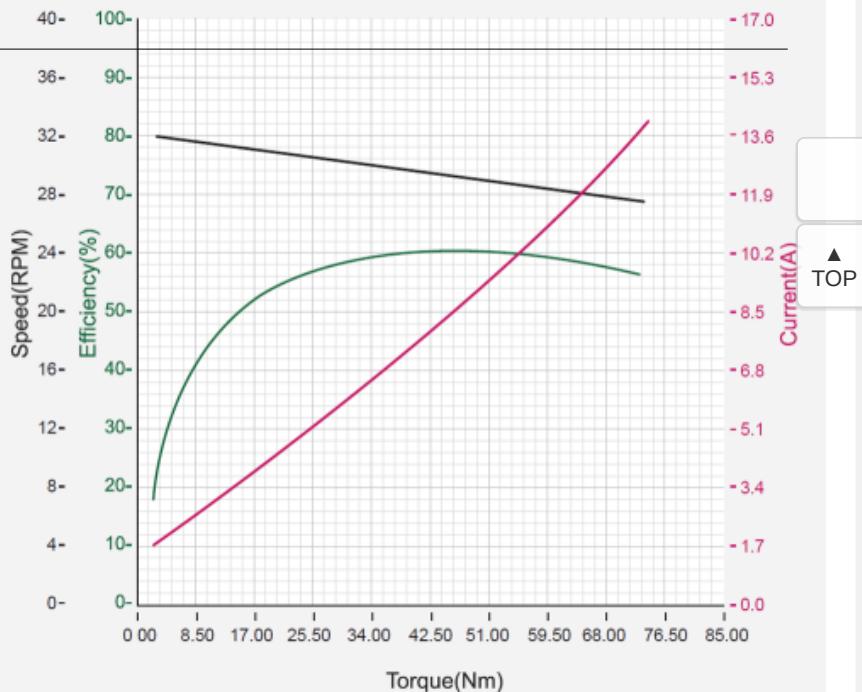
1. 1. Performance Graph

2. Control Table

3. How to Assemble

4. Maintenance

5. Reference

[Show Enlarged Graph](#)

NOTE : The Max Torque and the Stall Torque of Performance Graph are different in measurement methods. Stall torque is a measured value of the momentary torque that it can reach. This is generally how RC servos are measured. The Performance graph is also called as N-T curves, which is measured with the gradually increasing load. The actual motor operation environment is closer to the performance graph, not stall torque method. For this reason, the performance graph is broadly used in the industrial field. Generally, Max Torque of the Performance Graph is less than the Stall Torque.

CAUTION : When supplying power

- It is recommended using ROBOTIS controller or SMPS2DYNAMIXEL.
- Do not connect or disconnect DYNAMIXEL when power is being supplied.
- In case of DYNAMIXEL PRO and DYNAMIXEL-P series, please supply power through 24V power port.

2. Control Table

The Control Table is a structure of data implemented in the device. Users can read a specific Data to get status of the device with Read Instruction Packets, and modify Data as well to control the device with WRITE Instruction Packets.

2. 1. Control Table, Data, Address

The Control Table is a structure that consists of multiple Data fields to store status or to control the device. Users can check current status of the device by reading a specific Data from the Control Table with Read Instruction Packets. WRITE Instruction Packets enable users to

Manipulador Robótico JeRoTIMON

MANIPULADOR ROBÓTICO JEROTIMON

Relatório do Projeto Manipuladores Inteligentes

Autores:

Jean Paulo Silva
Jéssica Lima Motta
Leonardo Mendes de Souza Lima
Miguel Felipe Nery Vieira
Rodrigo Formiga Farias
Vinicius José Gomes de Araújo Felismino

Facilitadores:

Lucas Cruz da Silva
Rebeca Tourinho Lima
Tiago Pereira de Souza
Marco Antonio dos Reis

**Salvador
Bahia, Brasil**

Novembro de 2020

Título: Manipulador Robótico JeRoTIMON	
PROD. TEC. CCRoSA - 001 / 2020	Versão
Classificação: () Confidencial (X) Restrito () Uso Interno () Público	01

Informações Confidenciais - Informações estratégicas para o CCRoSA e Senai Cimatec.

Seu manuseio é restrito a usuários previamente autorizados pelo Gestor da área.

Informações Restritas - Informação cujo conhecimento, manuseio e controle de acesso devem estar limitados a um grupo restrito de pesquisadores que necessitam utilizá-la para exercer suas atividades profissionais.

Informações de Uso Interno - São informações destinadas à utilização interna por pesquisadores e parceiros.

Informações Públicas - Informações que podem ser distribuídas ao público externo, o que, usualmente, é feito através dos canais apropriados.

Dados Internacionais de Catalogação na Publicação (CIP)

Jean Paulo Silva Jéssica Lima Motta Leonardo Mendes de Souza Lima Miguel Felipe Nery Vieira Rodrigo Formiga Farias 000 Vinicius José Gomes de Araújo Felismino	Lucas Cruz da Silva Rebeca Tourinho Lima Tiago Pereira de Souza Marco Antonio dos Reis	Manipulador Robótico JeRoTIMON Salvador Bahia, Brasil Novembro de 2020	Keywords: 1. Manipulator. 2. Simulation. 3. Computer vision. 000
---	---	---	--

SUMÁRIO EXECUTIVO

O projeto de Manipuladores - Desafio 2.2, também conhecido como **JeRoTIMON Manipulator** configura-se: sob o Programa de Formação de Novos Talentos do [Centro de Competência de Robótica e Sistemas Autônomos \(CCRoSA\)](#) do Serviço Nacional de Aprendizagem Industrial, Departamento Regional da Bahia - Senai/DR/BA, sendo este o principal fomentador do programa. O presente trabalho tem como impulsionador principal a capacitação de novos pesquisadores preparados para solucionar os mais diversos problemas relacionados a robótica e sistemas autônomos.

O projeto foi considerado como início técnico no dia 27 de Fevereiro de 2020, ocorrendo um hiato de desenvolvimento entre o período de 17 de março de 2020 a 07 de setembro de 2020 devido às medidas de isolamento social decorrentes da pandemia do novo coronavírus, havendo seu retorno no dia 08 de setembro de 2020.

O prazo de execução planejado foi de 50 dias.

RESUMO

JeRoTIMON é um manipulador projetado, simulado e construído com o intuito de atender às demandas relacionadas ao reconhecimento de marcadores visuais e acionamento de interruptores, chaves ou botões. Posteriormente, este mesmo manipulador robótico será integrado ao robô *Warthog*, desenvolvido pela *Clearpath Robotics*, com o propósito de realizar a atividade de investigação em ambiente externo, desta vez participando de uma busca e desarme de bombas sem potencial destrutivo. O pacote de simulação do manipulador foi construído através do software *Gazebo* aliado ao *MoveIt* e a ferramenta de visualização *Rviz*, possibilitando assim que as atividades realizadas no mundo real tenham sido previamente testadas em ambiente simulado onde é possível analisar os movimentos e limitações do manipulador.

ABSTRACT

JeRoTIMON is a manipulator designed, simulated and built in order to meet demands related to the recognition of visual markers and activation of interrupters, switches or buttons. Posteriorly, this manipulator robot arm will be attached in *Warthog*, robot designed by *Clearpath Robotics*, with the purpose of carrying out the research activity in an external environment, this time, integrating a searching and defusing non-destructive bombs. The manipulator simulation package was built using software *Gazebo*, allied to *Moveit* and *Rviz* visual tool, allowing that activities practiced in the real world have been previously tested in a computer environment where it is possible to analyze manipulator's movements and limitations.

LISTA DE FIGURAS

Figura 1:	Elos e junta de um manipulador robótico.	18
Figura 2:	Arquitetura geral do sistema.	21
Figura 3:	Configuração D-H do manipulador JeRoTIMON.	24
Figura 4:	<i>Workspace</i> do manipulador no plano X-Y.	25
Figura 5:	<i>Workspace</i> do manipulador no plano X-Z.	25
Figura 6:	<i>Workspace</i> do manipulador no plano Y-Z.	26
Figura 7:	Estrutura analítica do protótipo.	27
Figura 8:	Fluxograma do sistema de escanamento.	28
Figura 9:	Fluxograma do sistema de planejamento e execução de trajetória.	30
Figura 10:	Modelos simulados do manipulador e do painel elétrico.	32
Figura 11:	Imagen capturada pela câmera RGB.	32
Figura 12:	Identificação dos motores.	33
Figura 13:	Base do manipulador.	34
Figura 14:	Vista lateral do <i>link</i> 2.	34
Figura 15:	Vista superior do <i>link</i> 2.	35
Figura 16:	Teledyne Genie Nano C2590.	36
Figura 17:	Lente 16mm C Series VIS-NIR.	36
Figura 18:	Suporte para fixação da câmera.	37
Figura 19:	Integração suporte-câmera-lente.	37
Figura 20:	UWE-12/10-Q12PB-C.	38
Figura 21:	U2D2.	39
Figura 22:	Caixa objetivo.	40
Figura 23:	Manipulador na posição home.	41
Figura 24:	Histograma do conjunto de dados sem detecção da <i>tag</i> .	47
Figura 25:	Boxplot do conjunto de dados sem detecção da <i>tag</i> .	47
Figura 26:	<i>Boxplot</i> da variância do tempo de busca entre os pontos A e B.	50
Figura 27:	<i>Boxplot</i> da variância do tempo total da missão entre os pontos A e B.	51
Figura 28:	Modelo completo teste R&R para tempo de busca.	52
Figura 29:	Modelo completo teste R&R para tempo total da missão.	52
Figura 30:	Gráficos do estudo R&R para tempo de busca.	53
Figura 31:	Árvore de falhas do sistema.	57
Figura 32:	Coordenadas.	75
Figura 33:	Link 0.	75
Figura 34:	Link 1.	76
Figura 35:	Link 2.	77

Figura 36: Link 3	77
Figura 37: Link 4	78
Figura 38: Link 5	79
Figura 39: Desenho técnico do JeRoTIMON, vistas: frontal, lateral esquerda e superior	84
Figura 40: Desenho técnico do JeRoTIMON, vista isométrica.	85
Figura 41: Analise estática - condição de maior esforço exigido.	86
Figura 42: Analise estática - condição de maior esforço exigido (cargas no centróide).	86

LISTA DE TABELAS

Tabela 1:	Especificações técnicas do manipulador JeRoTIMON.	23
Tabela 2:	Parâmetros D-H para o manipulador JeRoTIMON.	24
Tabela 3:	Parâmetros dos motores.	33
Tabela 4:	Parâmetros dos motores.	38
Tabela 5:	Torque das juntas.	39
Tabela 6:	Ângulos dos motores na posição home.	41
Tabela 7:	Modelo de experimentos para análise de 3 variáveis.	43
Tabela 8:	Modelo de experimentos ANOVA sem detecção da <i>tag</i> .	44
Tabela 9:	Modelo de experimentos ANOVA com detecção da <i>tag</i>	44
Tabela 10:	Resultados das amostras coletadas.	44
Tabela 11:	Resultados da análise de regressão linear.	45
Tabela 12:	Resultados da análise de regressão linear (variável algoritmo).	45
Tabela 13:	Resultados dos experimentos sem detecção da <i>tag</i> .	46
Tabela 14:	Resultados dos experimentos sem detecção da <i>tag</i> .	48
Tabela 15:	Resumo dos conjuntos de dados das variáveis de saída.	50
Tabela 16:	Tabela dos valores de ρ -valor do teste ANOVA.	50
Tabela 17:	<i>FMEA</i> do sub-sistema de potência	55
Tabela 18:	<i>FMEA</i> do sub-sistema aquisição	55
Tabela 19:	<i>FMEA</i> do sub-sistema estrutural	56
Tabela 20:	<i>FMEA</i> do sub-sistema de atuação	56
Tabela 21:	<i>FMEA</i> do sub-sistema de processamento	56
Tabela 22:	Lições aprendidas	59
Tabela 23:	Tabela de suportes	81
Tabela 24:	Especificações do motor <i>Dynamixel HP42-020-S300-R</i>	119
Tabela 25:	Especificações do motor <i>Dynamixel PH54-100-S500-R</i>	123
Tabela 26:	Especificações do motor <i>Dynamixel PH54-200-S500-R</i>	127

LISTA DE SÍMBOLOS E ABREVIATURAS

AGV Automated Guided Vehicle

CAD Computer Aided Design

CCRoSA Centro de Competência de Robótica e Sistemas Autônomos

DoF Degrees of Freedom

FMECA Failure Modes, Effects and Critically Analysis

OMPL Open Motion Planning Library

OpenCV Open Source Computer Vision

ROS Robot Operating System

SOTA Study Of The Art

URDF Unified Robot Description Format

SUMÁRIO

1 INTRODUÇÃO	13
1.1 Objetivos	13
1.2 Justificativa	14
1.3 Organização do relatório	14
2 CONCEITO DO SISTEMA	17
2.1 Parâmetros básicos	17
2.1.1 Requisitos do cliente	17
2.1.2 Requisitos técnicos	17
2.1.3 Estudo do estado da arte	18
3 DESENVOLVIMENTO DO SISTEMA	21
3.1 Descrição do sistema	21
3.1.1 Arquitetura geral	21
3.1.2 Especificação técnica	22
3.1.3 Ambiente de operação	24
3.1.4 Estrutura analítica do protótipo	26
3.2 Especificação funcional	27
3.2.1 Escaneamento	27
3.2.1.1 Descrição	27
3.2.1.2 Premissas necessárias	28
3.2.1.3 Dependências	29
3.2.1.4 Saídas	29
3.2.2 Planejamento e Execução de Trajetória	29
3.2.2.1 Descrição	29
3.2.2.2 Premissas necessárias	30
3.2.2.3 Dependências	30
3.2.2.4 Saídas	31
3.3 Arquitetura de software	31
3.4 Simulação do sistema	31

4 IMPLEMENTAÇÃO	33
4.1 Parametrização dos motores	33
4.2 Estrutura física	33
4.2.1 Base	34
4.2.2 Elos ou <i>links</i>	34
4.2.3 Suportes	35
4.2.4 Câmera	35
4.2.5 Atuadores	37
4.3 Sistema de Potência	38
4.4 Comunicação	38
4.5 Análise de esforços	39
4.6 Configurações	40
5 RESULTADOS E ANÁLISES	43
5.1 Caracterização do problema e determinação do modelo	43
5.2 Planejamento dos experimentos	43
5.3 Resultados alcançados	44
5.3.1 Análise de regressão linear	44
5.3.2 ANOVA	46
5.3.2.1 Análise de variância sem detecção da <i>tag</i>	46
5.3.2.2 Análise de variância com detecção da <i>tag</i>	48
5.3.3 Análise de repetibilidade e reproduzibilidade	51
5.4 Conclusões dos testes estatísticos	54
6 CONFIABILIDADE DO SISTEMA	55
6.1 Análise dos modos e efeitos de falhas	55
6.2 Análise da árvore de falhas	56
7 GESTÃO DO CONHECIMENTO	59
7.1 Lições aprendidas	59
7.2 Guia de uso para simulação	59
7.3 Guia de uso para o modelo real	62
8 CONCLUSÃO	65

REFERÊNCIAS	67
APÊNDICE A Árvores de TF desconectadas	70
APÊNDICE B Árvores de TF conectadas	71
APÊNDICE C Diagrama de nós do sistema	74
APÊNDICE D Propriedades de Massa do JeRoTIMON	75
APÊNDICE E Lista de suportes.	81
APÊNDICE F Projeto mecânico	83
F.1 Desenho mecânico	83
F.2 Analise estática	86
APÊNDICE G Algoritmo de busca e acionamento do painel	89
APÊNDICE H Diagrama elétrico do JeRoTIMON	101
APÊNDICE I Diagrama de conexão do JeRoTIMON	109
ANEXO A Especificações da câmera Dalsa Genie Nano	115
ANEXO B Especificações do motor <i>Dynamixel PH42-020-S300-R</i>	119
ANEXO C Especificações do motor <i>Dynamixel PH54-100-S500-R</i>	123
ANEXO D Especificações do motor <i>Dynamixel PH54-200-S500-R</i>	127

1 INTRODUÇÃO

A robótica é um campo relativamente jovem da tecnologia moderna que atravessa os limites da engenharia tradicional ([SPONG; HUTCHINSON; VIDYASAGAR, 2005](#)). O estudo da robótica é um ramo da tecnologia que engloba área de mecânica, eletrônica e computação, com graus de teoria de controle, microeletrônica, inteligência artificial, fatores humanos e de produção ([PIMENTA, 2009](#)). Segundo ([ERTHAL, 1992](#)), o crescimento da robótica na indústria é justificado em face das exigências de maior qualidade, produtividade e flexibilidade nos processos fabris. Na área industrial, a robótica evoluiu devido ao aumento de uso de robôs e manipuladores industriais.

O estudo do desenvolvimento de manipuladores robóticos foi iniciado por volta de 1954 com George Devol, quando foi desenvolvido o primeiro robô programável e desde então grandes desenvolvimentos nessa área foram atingidos. Como resultado desse avanço, o investimento de empresas foi em torno de 16,5 bilhões de dólares em 2018, chegando a marca de 420 mil unidades enviadas mundialmente, com perspectiva de crescimento médio de 12% ao ano entre 2020 e 2022 ([INDUSTRIAL..., 2019](#)).

Um manipulador robótico é um dispositivo mecânico composto de elementos rígidos (elos) que proporcionam a sustentação e alcance do braço. A inevitabilidade de apresentar algum grau de flexibilidade faz com que esses elos necessitem ser projetados para apresentar elevada rigidez aos esforços que o manipulador será submetido. Esses elos são conectados entre si através de articulações (juntas), que oferecem graus de liberdade ao manipulador e controle de movimento relativo entre os elos. Essas juntas podem ser basicamente divididas em dois grupos: juntas prismáticas e juntas de rotação. Neste projeto foram utilizadas juntas de rotação. A disposição dessas juntas determina a classificação dos manipuladores como Cartesianos, Esféricos, Cilíndricos, entre outros ([ROMANO, 2002](#)).

Este relatório descreve o processo de construção de um manipulador robótico desenvolvido no Centro de Competência em Robótica e Sistemas Autônomos do SENAI CIMATEC e é destinado ao programa de formação Novos Talentos. São descritas as etapas de concepção, simulação, testes e implementação física.

1.1 Objetivos

O propósito deste projeto é construir um manipulador robótico capaz de identificar um marcador visual por meio de uma câmera RGB e proceder com a função de acionar um painel elétrico. Para isso, os objetivos específicos são:

- Realizar estudo do Estado da Arte([SOTA](#)) sobre manipuladores.
- Realizar testes e parametrização dos servomotores.

- Propor um modelo de manipulador robótico.
- Parametrizar o pacote de reconhecimento de marcadores visuais.
- Desenvolver um pacote de configuração do *MoveIt*.
- Desenvolver o código para realizar a missão.
- Realizar a simulação do protótipo do manipulador em software.
- Implementar a versão física do protótipo.
- Realizar testes e estudos estatísticos.

1.2 Justificativa

Apesar da crescente demanda, há falta de profissionais habilitados à desenvolver pesquisas e aplicações na área da robótica. O presente trabalho tem como impulsionador principal a capacitação de novos pesquisadores preparados para solucionar os mais diversos problemas relacionados a robótica e sistemas autônomos.

A busca por melhor eficiência e precisão na realização de atividades em locais que a presença do ser humano torna-se difícil, arriscado e até mesmo impossível, vem se tornando cada vez maior no cotidiano dos ambiente industriais. Além disso, é importante a capacidade do robô de interagir com o ambiente a partir da captura e análise de estímulos visuais ([LEITE, 2005](#)).

Este projeto traz, dentre os benefícios, a utilização de manipuladores robóticos autônomos que sejam capazes de identificar marcadores visuais e realizar tarefas que possam ser perigosas e/ou repetitivas para o ser humano. Espera-se que este projeto seja continuado e que seus resultados sejam compartilhados na comunidade científica, contribuindo para a construção de outros manipuladores com características e/ou objetivos semelhantes.

1.3 Organização do relatório

O presente relatório está organizado em oito capítulos, sendo este de introdução e descrição da justificativa/motivação dos objetivos e da organização do documento.

No capítulo [2](#), Conceito do Sistema, são descritos parâmetros básicos do projeto, dentre eles os requisitos do cliente, requisitos técnicos e o estudo do estado da arte.

O capítulo [3](#), Desenvolvimento do Sistema, apresenta a descrição do sistema onde serão apresentados a arquitetura geral, especificações técnicas, o ambiente de operação do manipulador e a estrutura analítica do protótipo. Além disso, trará as especificações funcionais que compõem o sistema, sua arquitetura de software e o que foi desenvolvido para simulação.

O capítulo 4, Implementação, explica a construção física do manipulador. São expostos seus parâmetros de configuração e sua estrutura.

O capítulo 5, Resultados e Análises, são apresentados os resultados alcançados e a análise dos dados amostrados através de estudos estatísticos.

O capítulo 6, Confiabilidade do Sistema, detalha a análise dos modos e efeitos de falhas.

No capítulo 7, Gestão do Conhecimento, é feito um estudo sobre as lições aprendidas além de apresentar o guia de uso.

Por fim, o capítulo 8 apresenta a conclusão do relatório.

2 CONCEITO DO SISTEMA

A norma técnica (ISO-8373, 2012) criada para padronizar o vocabulário referente aos robôs e dispositivos robóticos operando em ambientes industriais e não industriais, define o manipulador como uma máquina na qual o mecanismo, geralmente, consiste em uma série de segmentos, articulados ou deslizantes entre si, com o objetivo de empunhar e/ou mover objetos (peças ou ferramentas) em vários graus de liberdade. Em outras palavras, um manipulador é um equipamento programável baseado em atuadores, com um certo grau de liberdade, projetado para realizar uma variedade de atividades, assim como realização de diversos processos industriais (ISO-8373, 2012).

Neste capítulo serão tratados os requisitos solicitados pelo cliente, os requisitos técnicos do projeto, o estudo do estado da arte sobre manipuladores e o ambiente de operação em que este manipulador realizará a atividade.

2.1 Parâmetros básicos

Nesta seção encontram-se os requisitos solicitados pelo cliente, ou seja, a tarefa que precisa ser realizada e em qual ambiente o manipulador será simulado e concebido. Além disso, são exibidos os requisitos técnicos que tratam das especificações do sistema e uma breve revisão teórica de conceitos relacionados ao manipulador.

2.1.1 Requisitos do cliente

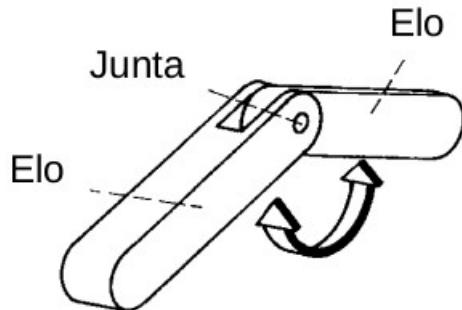
Requisitos predeterminados pelo cliente consistem em exigências de funcionamento que devem ser observadas ao final do projeto, para que se considere um sucesso a concepção deste. Para tal, foram determinadas algumas características desejáveis no projeto:

1. Desenvolver um manipulador robótico.
2. Realizar a tarefa de detecção de um marcador visual e acionamento do painel elétrico na orientação vertical e horizontal.
3. Realizar a simulação do manipulador no ambiente *ROS* utilizando o software *Gazebo*.
4. Utilizar o *framework MoveIt* para o controle do manipulador.
5. Realizar estudo estatístico para verificação da performance do manipulador.

2.1.2 Requisitos técnicos

Os requisitos técnicos de um projeto são especificações necessárias para o funcionamento esperado do projeto. Podem ser sobrepostos aos requisitos do cliente em caso de conflito

Figura 1: Elos e junta de um manipulador robótico.



Fonte: ([SANTOS, 2004](#)).

entre o esperado pelo cliente e o necessário para que o projeto seja bem sucedido, objetivando manter o projeto o mais eficiente dentro do escopo planejado. Os requisitos foram:

1. O manipulador deve estar acoplado em uma base fixa situada a 0,28 m de uma das extremidades da bancada e centralizada com a mesma.
2. Utilizar servo motores Dynamixel PH54-200-S500-R, PH54-100-S500-R e PH42-020-S300-R da Robotis.
3. Conter uma câmera RGB Teledyne Dalsa Genie Nano C2590.
4. Deverá possuir 5 graus de liberdade.
5. Suportar uma carga máxima de 2 kg.
6. Ser capaz de acionar um painel elétrico.

2.1.3 Estudo do estado da arte

Com o advento do exponencial crescimento da tecnologia há um foco crescente na pesquisa e comercialização de robôs ([HERNÁNDEZ-ORDOÑEZ et al., 2018](#)). Estes, por sua vez, são classificados em três grupos: manipuladores, veículos auto-guiados ([AGV](#)) e robôs móveis. Neste projeto, o objeto de interesse são os manipuladores robóticos, sistemas que possuem estrutura física similar a um braço humano. Estes robôs são compostos por partes rígidas, denominado de elos, conectados entre si por juntas ([SANTOS, 2004](#)). Esta estrutura encontra-se descrita na Figura 1.

Muitas pesquisas têm sido realizadas na área de manipuladores robóticos. Em ([HERNANDEZ-MENDEZ et al., 2017](#)) é descrito o desenvolvimento de um manipulador robótico com 3 *DoF* e dois dedos independentes. Este robô foi desenvolvido com o propósito de manipular objetos, cujas localizações são conhecidas, e transportá-los de uma localidade para outra

utilizando *ROS*. Os autores utilizaram o *MoveIt* para tratar do planejamento de trajetória e o *Rviz* como ferramenta de visualização. Além disso, foi desenvolvido um controlador de posição e força para o *endeffector*¹ durante o processo de *pick and place*². Os experimentos realizados trouxeram bons resultados para o que foi proposto, sendo ressaltada a necessidade de adicionar um sistema de visão que permita identificar e localizar o objeto alvo.

O planejamento de trajetória para um manipulador de 5 *Dof* a partir do *MoveIt* é exibido em (ZHANG; LIN; WU, 2019). A partir de um modelo *CAD* já existente foi construído o modelo *URDF* utilizado para simulação no *ROS*. O processo de planejamento foi visualizado a partir do *Rviz* e o caminho planejado foi transmitido para os servo-motores possibilitando ao manipulador executar a sua rotina. O robô tem como tarefa manipular um objeto alvo de um local para outro, identificado a partir de técnicas de visão computacional que combinam os algoritmos *SIFT* e *RANSAC*. Os resultados experimentais mostraram que o uso do *MoveIt* reduz as dificuldades de operação para manipuladores e oferece vantagens em termos de validação de algoritmos e exploração de funções, sendo possível utilizar o método proposto para o controle em tempo real do robô.

A aplicação de técnicas de visão computacional em um manipulador do tipo *Robai Cyton Gamma 3000* é exibida em (KHAN; KONDA; RYU, 2018). O robô é conectado com uma câmera externa via *ROS*, possui um *endeffector* em forma de garra que segura uma estrutura similar a um prato, e tem como tarefa equilibrar uma bola localizada no centro do prato. Para realizar o controle da tarefa de equilíbrio, a bola é identificada a partir de um algoritmo escrito em C++ e que utiliza bibliotecas do *OpenCV*. As juntas do manipulador são atuadas a partir de servo-motores *Dynamixel* que são diretamente controlados por um algoritmo. O sistema foi desenvolvido de forma que os componentes se comunicassem entre si para receber respostas do sistema de visão e dos motores, computá-las e enviar comandos de controle que permitissem executar a tarefa. Foram realizados testes e os resultados experimentais foram satisfatórios, sendo o manipulador capaz de equilibrar a bola em uma pequena vizinhança do centro do prato.

O uso de marcadores visuais do tipo *ArUco* associados ao planejamento de trajetória para um manipulador robótico é abordado em (JAVEED; PRAKASH; KULKARNI, 2019). O manipulador encontra-se acoplado a uma plataforma móvel e tem como objetivo o transporte de objetos de forma autônoma em um determinado espaço. Os autores também descrevem a integração dos mecanismos de detecção do marcador visual, navegação do robô e planejamento de trajetória, realizados no *ROS*. O robô é capaz de estimar a pose do marcador, aproximar-se e realizar sua tarefa. Os testes realizados mostraram a

¹ Na robótica, um *endeffector* é o dispositivo no final de um braço robótico, projetado para interagir com o meio ambiente.

² Sequência de movimentos na qual o manipulador robótico pega determinado objeto e o transfere a uma pose alvo.

eficiência do sistema, sendo apontada a necessidade de um estudo futuro para possibilitar o planejamento de trajetória em um ambiente com obstáculos.

3 DESENVOLVIMENTO DO SISTEMA

Nesta seção serão explicitadas as características do manipulador JeRoTIMON, abordando os sistemas que o compõem em *software* e em *hardware*. Suas funcionalidades principais são abordadas e a conexão entre as mesmas é exibida.

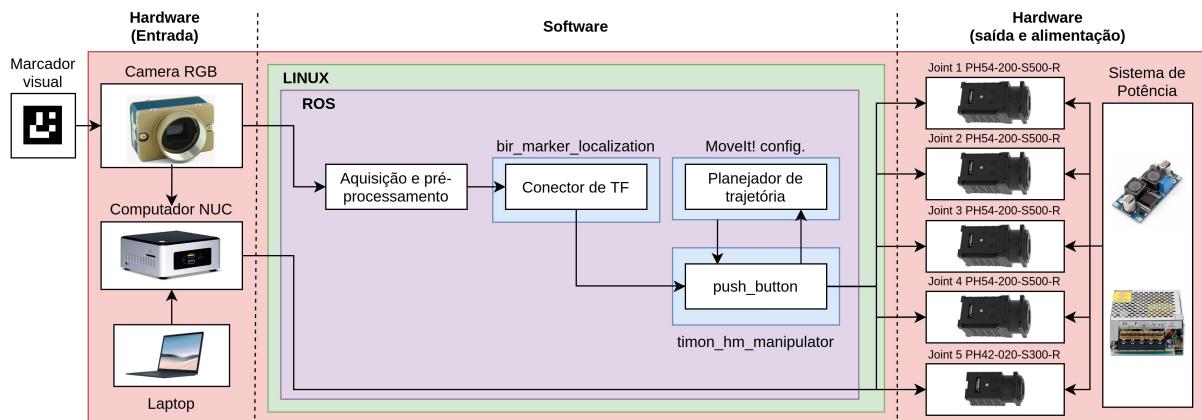
3.1 Descrição do sistema

JeRoTIMON é um manipulador desenvolvido para atender às demandas relacionadas ao reconhecimento de marcadores visuais e, a partir desta identificação, realizar o acionamento de um painel elétrico. Os pacotes que constituem este robô foram concebidos através do *software* de simulação *Gazebo*, da ferramenta de visualização *Rviz* e do *framework*¹ para planejamento de trajetória *MoveIt*. O uso dessas ferramentas possibilita que uma grande variedade de atividades que venham a ser realizadas no mundo real tenham sido previamente testadas em ambiente computacional.

3.1.1 Arquitetura geral

A Figura 2 ilustra a estruturação do sistema e a relação entre *software* e *hardware*. As cores representam o sistema geral(salmão), sistema operacional(verde), *framework*(roxo) e pacotes(azul).

Figura 2: Arquitetura geral do sistema.



Fonte: Autoria própria.

Um laptop, conectado via acesso remoto, dá início a aplicação no computador *NUC*² que possui instalado o software do protótipo. Com o sistema iniciado, a câmera RGB é

¹ São conjuntos de aplicações dentro de um projeto que interagem entre si e com isso se alcança resultados como uma determinada função de um programa.

² Computador pequeno, completo e altamente eficiente energeticamente.

capaz de obter dados visuais do ambiente e enviá-los para o *ROS*. Ao encontrar o marcador visual, o pacote *bir_marker_localization* é capaz de unir as árvores de *TF*³ do painel elétrico e do manipulador, que antes encontravam-se desconectadas. Esta conexão, garante que sejam conhecidos os dados de posição ao nó *push_button* possibilitando o planejamento de trajetória para o ponto desejado. Com o caminho planejado, *push_button* pode enviar os comandos para cada junta do manipulador, onde encontram-se os motores *Dynamixel* que são alimentadas pelo sistema de potência.

3.1.2 Especificação técnica

Na Tabela 1 estão elencadas as especificações técnicas do manipulador robótico JeRoTIMON. O numero de Graus de Liberdade foi definido baseado na capacidade de movimentação necessária para a realização dos desafios propostos. A carga útil, peso e o alcance máximo foram calculados com o auxilio do software *Onshape*, uma alternativa ao cálculo manual. A faixa de operação dos motores foi obtida segundo os limites de segurança observados, o que acrescenta proteção principalmente ao cabeamento do sistema. Informações a respeito de componentes como câmeras e motores seguem o seu padrão original de fabricação.

³ Pacote do *ROS* que permite verificar as relações entre os *frames* na estrutura de árvore.

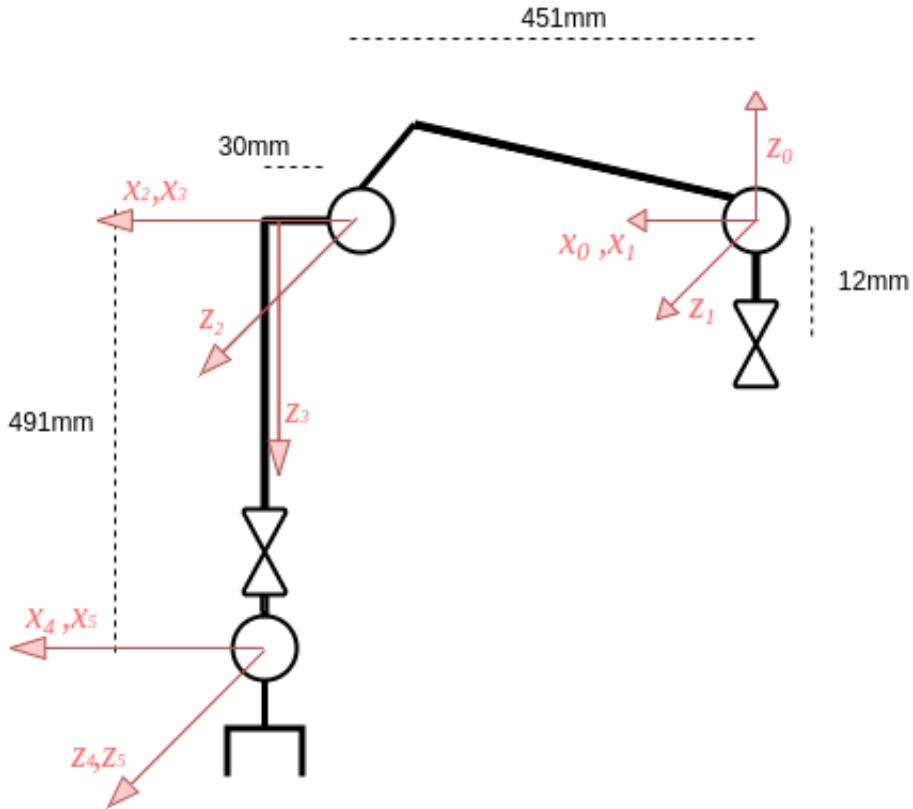
Tabela 1: Especificações técnicas do manipulador JeRoTIMON.

Item	JeRoTIMON
Graus de liberdade	5
Carga útil	2 [kg]
Alcance	981 [mm]
Peso (sem base)	6,4 [kg]
Peso (com base)	10 [kg]
Tensão de operação	24 [V]
Resolução	Junta 1, Junta 2, Junta 3, Junta 4: 1,003,846 [pulsos/rev]
	Junta 5: 607,500 [pulsos/rev]
Motores	Junta 1, Junta 2, Junta 3: PH54-200-S500-R (200 W)
	Junta 4: PH54-100-S500-R (100W)
	Junta 5: PH42-020-S300-R (20 W)
Faixa de operação	Junta 1: $-45^\circ \sim 45^\circ$
	Junta 2: $-90^\circ \sim 90^\circ$
	Junta 3: $-43^\circ \sim 173^\circ$
	Junta 4: $-90^\circ \sim 90^\circ$
	Junta 5: $-90^\circ \sim 90^\circ$
Câmera	Teledyne Genie Nano C2590
Tipo de sensor de posição	Posse inicial: Codificador Absoluto
	Controle: Codificador Incremental
Comunicação	USB
Padrão elétrico	RS485
Taxa de transmissão	57,600 [bps]

Fonte: Autoria própria.

Para estabelecer uma relação entre o *endeffector* e a base do manipulador é necessário descrever o seu sistema de coordenadas em relação ao sistema de coordenadas de origem. Para isto, é utilizada a notação de *Denavit-Hartenger*(D-H). A partir da configuração D-H exibida na Figura 3 foram retirados os parâmetros exibidos na tabela 2.

Figura 3: Configuração D-H do manipulador JeRoTIMON.



Fonte: Autoria própria.

Tabela 2: Parâmetros D-H para o manipulador JeRoTIMON.

Link	a (mm)	$\alpha(^{\circ})$	d (mm)	$\theta(^{\circ})$
1	0	90	12	0
2	452	0	0	$90 - \arctan(30/451)$
3	30	-90	0	$45 + \arctan(30/451)$
4	0	90	491	0
5	0	0	0	0

Fonte: Autoria própria.

3.1.3 Ambiente de operação

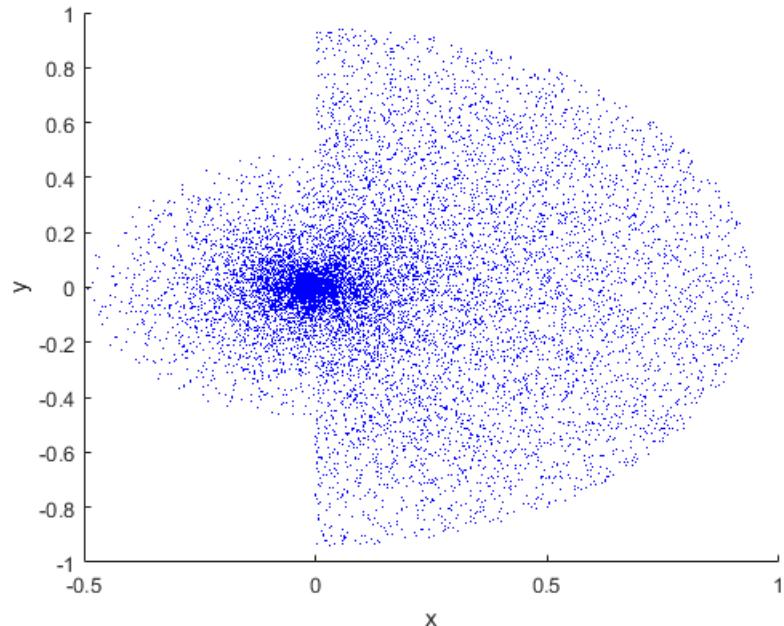
O ambiente físico para a realização do desafio, onde serão incluídos o manipulador e um painel elétrico, é uma mesa com 1.7 m de comprimento por 0.8 m de largura. É necessário então que verifique-se quais os pontos deste ambiente de trabalho que estão dentro da área de alcance (*workspace*)⁴ do manipulador.

A partir da tabela 2 foi desenvolvido um código utilizando o software *Matlab R2020a* capaz de gerar pontos que populem o *workspace* do manipulador nas projeções dos planos

⁴ É a área de alcance do manipulador, região na qual este consegue operar.

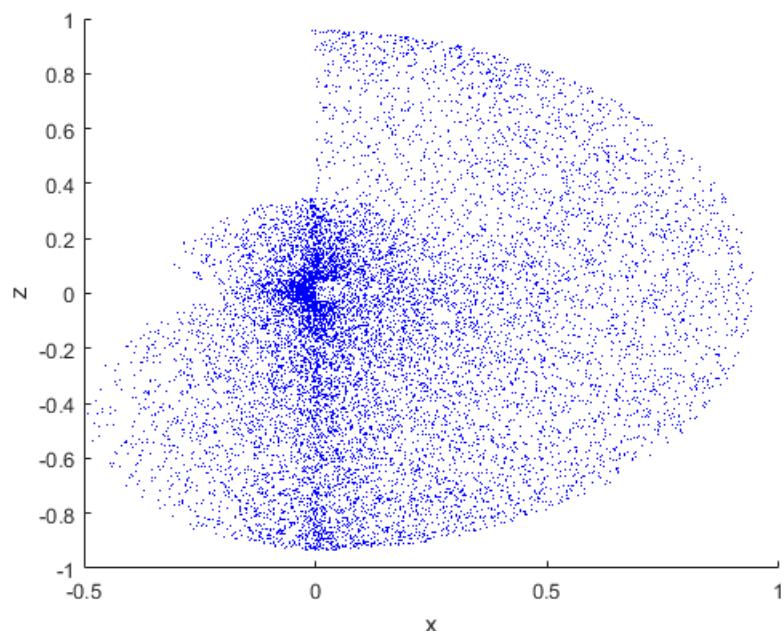
X-Y, X-Z e Y-Z, conforme exibido nas figuras 4, 5 e 6. A região em azul indica o alcance do robô e, a partir destas imagens, é possível observar que o manipulador possui restrições de operação devidas às limitações existentes em cada junta.

Figura 4: *Workspace* do manipulador no plano X-Y.



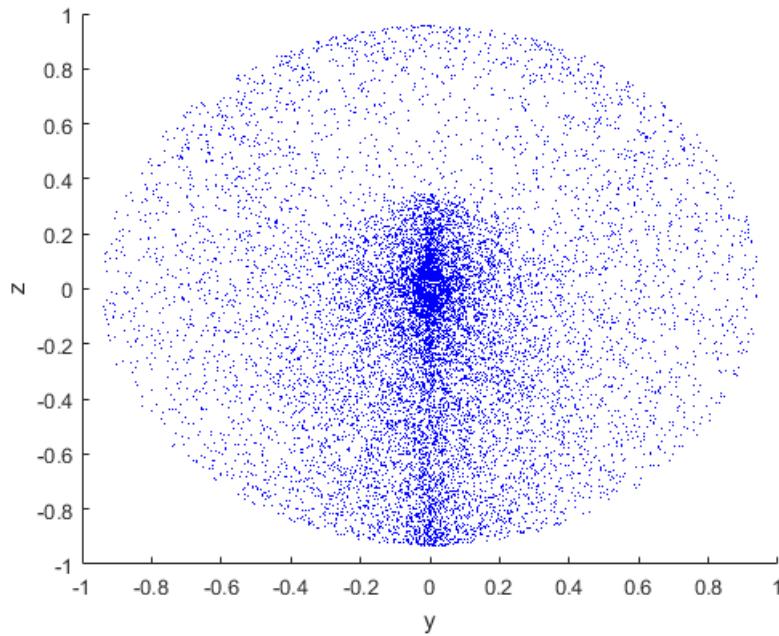
Fonte: Autoria própria.

Figura 5: *Workspace* do manipulador no plano X-Z.



Fonte: Autoria própria.

Figura 6: *Workspace* do manipulador no plano Y-Z.

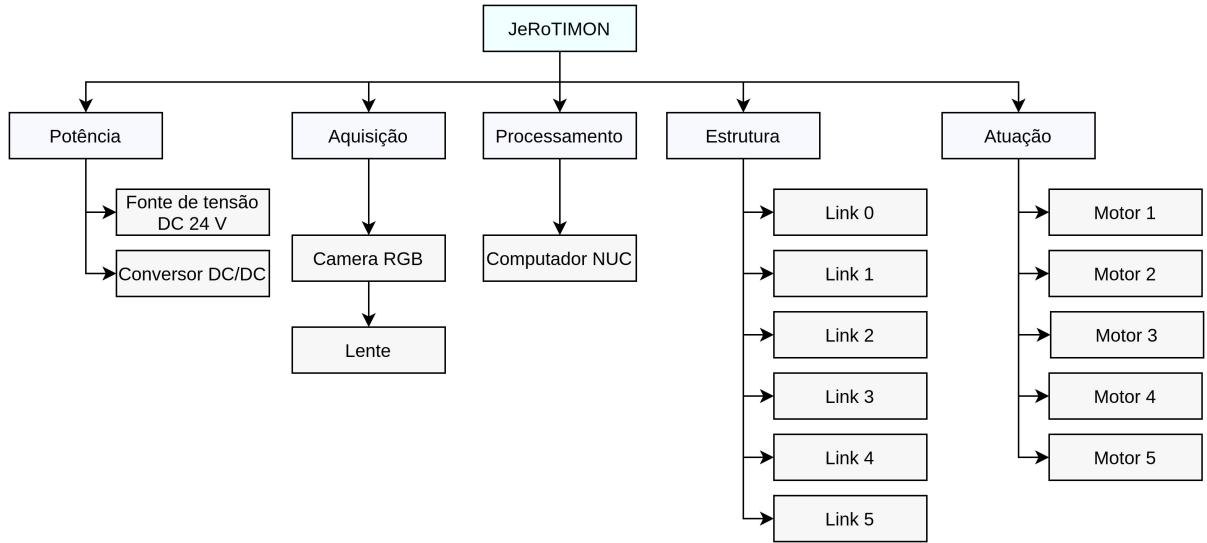


Fonte: Autoria própria.

3.1.4 Estrutura analítica do protótipo

A estrutura analítica do protótipo mostrada na Figura 7 exibe as relações sistemáticas entre as partes que compõem o manipulador. A estrutura hierárquica possui três níveis: o primeiro, referente ao sistema principal JeRoTIMON; o segundo, que é composto pelos sub-sistemas de potência, aquisição, processamento, estrutura e atuação; e o terceiro, composto pelos itens que fazem parte de cada um destes sub-sistemas.

Figura 7: Estrutura analítica do protótipo.



Fonte: Autoria própria.

3.2 Especificação funcional

O manipulador descrito trabalha acionando os botões encontrados pelo sistema de aquisição. Seu software funciona baseado na troca de mensagens entre duas funcionalidades principais: Escaneamento e Planejamento/Execução de trajetória. O Escaneamento corresponde à detecção de um marcador visual, que uma vez detectado, informa a posição no espaço de um painel elétrico que precisa ser acionado. O planejamento e execução de trajetória utiliza cálculos de cinemática direta e inversa para definir a trajetória de movimentação que permitirá ao manipulador realizar sua tarefa.

3.2.1 Escaneamento

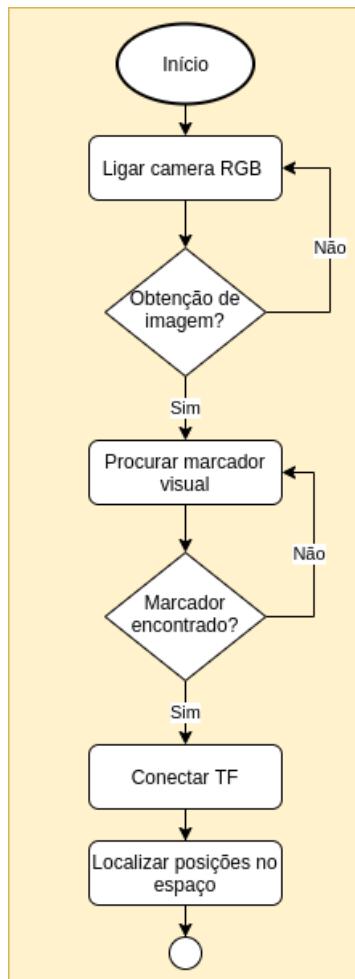
Uma câmera RGB *Teledyne Genie Nano C2590* equipada com lente *kowa LM8FC* foi acoplada ao manipulador JeRoTIMON. Através da mesma é realizada a detecção do marcador visual, utilizando a biblioteca *ArUco* e o pacote *Bir Marker Localization*, hospedado no site do Github no perfil do BIR - Brazilian Institute of Robotics ([BIR... , 2020](#)).

3.2.1.1 Descrição

A Figura 8 exibe o fluxograma que descreve o funcionamento do sistema de escaneamento integrado ao manipulador. Após a captura da imagem, a partir da câmera RGB, é feito um processamento dos dados obtidos afim de localização da *tag ArUco*, cujos tamanho e ID foram previamente estabelecidos. Caso o marcador seja encontrado, as árvores de *TFs* do painel elétrico onde encontra-se o marcador, e do manipulador robótico são conectadas,

possibilitando a localização no espaço da pose alvo. Os apêndices A e B exibem as árvores de *TFs* antes e após a conexão realizada.

Figura 8: Fluxograma do sistema de escanamento.



Fonte: Autoria própria.

3.2.1.2 Premissas necessárias

- Conter um marcador visual anexado ao painel elétrico.
- A orientação dos elementos envolvidos no escaneamento, câmera RGB e marcador visual, devem ser definidos conforme estabelecido pelo pacote *bir_marker_localization*.
- Não haver oclusão do marcador visual.
- Nível de iluminação do ambiente suficiente para a execução da detecção.
- O marcador visual deverá possuir tamanho adequado para detecção.

3.2.1.3 Dependências

Para realizar a etapa de detecção é necessária a instalação do *OpenCV* versão 3.3.1, *driver GigE-V Framework* e a inserção dos pacotes *bir_marker_localization* e *def_cam_tedelyne_nano* no *workspace* do manipulador.

3.2.1.4 Saídas

É fornecida ao sistema resposta por meio de uma sequência de mensagens publicadas no tópico */timon/camera/image_raw*. Estes dados são analisados pelo detector *ArUco*, possibilitando seu uso em um pacote desenvolvido na linguagem C++ que determina a posição do painel elétrico associado ao marcador visual.

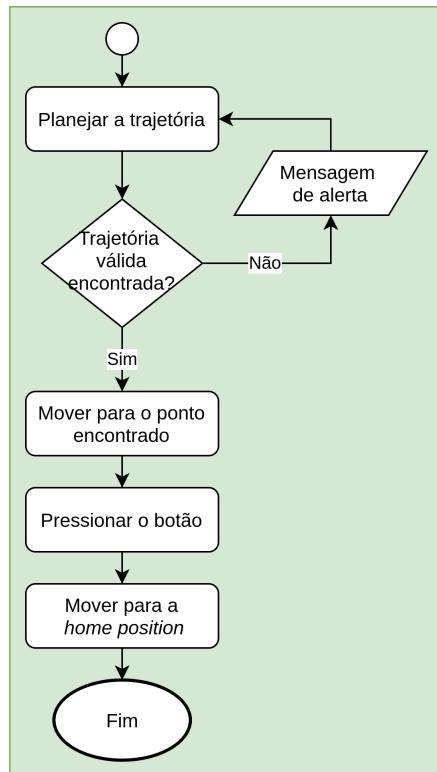
3.2.2 Planejamento e Execução de Trajetória

As equações cinemáticas são a base que possibilitam a pesquisa do movimento dos manipuladores. A cinemática inversa provê um conjunto de valores para as juntas do manipulador com o intuito de alcançar uma determinada pose pré-estabelecida do seu *endeffector*. Para resolver as equações da cinemática inversa do JeRoTIMON, optou-se por utilizar o plugin TRAC-IK, um método alternativo ao padrão da inversa Jacobiana utilizado pelo *MoveIt*. Este método se adequa bem a manipuladores que possuem limitações em suas juntas, ao contrário de algoritmos baseados no teorema de Newton ([BEESON; AMES, 2015](#)). Para o planejamento de trajetória foi utilizada a biblioteca *OMPL*, uma coleção de algoritmos de planejamento utilizada por padrão no *MoveIt* ([SUCAN; MOLL; KAVRAKI, 2012](#)).

3.2.2.1 Descrição

A Figura 9 exibe o funcionamento do sistema de planejamento e execução de trajetória aplicado ao manipulador. A pose do painel elétrico, determinada a partir do que foi mostrado em 3.2.1, é enviada como entrada para o *MoveIt*. Caso seja possível, é realizado o planejamento de uma trajetória para cada uma das juntas do manipulador, a fim de movê-lo para a pose desejada. Esta trajetória é então enviada para os atuadores das juntas, que passam a executá-la. Após a realização da rotina para o pressionamento do painel elétrico, o manipulador retorna para sua posição inicial. Caso alguma condição impeça o planejamento de trajetória, como por exemplo o posicionamento do painel elétrico fora da área de trabalho do manipulador ou falhas nas soluções para as equações da cinemática inversa, uma mensagem de alerta é exibida e o robô realiza uma nova tentativa de planejamento.

Figura 9: Fluxograma do sistema de planejamento e execução de trajetória.



Fonte: Autoria própria.

3.2.2.2 Premissas necessárias

- Viabilidade das soluções para cinemática inversa.
- Painel elétrico estar posicionado na área de trabalho do manipulador.

3.2.2.3 Dependências

O sistema de movimentação é dependente da versão Melodic Morenia do *framework ROS* e da plataforma *MoveIt*. Além destes, uma lista de pacotes deve ser instalada previamente para o funcionamento correto do sistema:

- ros-melodic-ros-control
- ros-melodic-gazebo-ros-control
- ros-melodic-controller-manager
- ros-melodic-joint-trajectory-controller
- ros-melodic-joint-state-controller
- ros-melodic-position-controllers

- ros-melodic-trac-ik-kinematics-plugin

3.2.2.4 Saídas

As respostas fornecidas pelo sistema de planejamento e execução são publicadas nos motores *Dynamixel* integrados às juntas do manipulador. A trajetória gerada pelo *MoveIt* pôde ser visualizada a partir do tópico */move_group/display_planned_path* e é publicada nos motores a partir do tópico */timon_arm_controller/dynamixel_state*.

3.3 Arquitetura de software

O robô JeRoTIMON foi desenvolvido para atuar em conjunto com o *ROS*, isto é, segue o propósito de conectar diferentes módulos, como câmeras, motores, sensores e códigos. A proposta é conectar um programa de visão capaz de conectar árvores de TF, através da identificação de marcadores visuais, com um sistema de movimentação que posiciona o manipulador para acionar o painel elétrico.

O apêndice C exibe o diagrama do *software* do sistema, obtido via *rqt_graph*. Observa-se a comunicação entre o manipulador (*/timon/robot_state_publisher*) e o painel elétrico (*/box/box_state_publisher*) a partir da conexão das árvores de *TFs* realizadas pelo */marker_localization* que recebe e analisa os dados da câmera (*/timon/camera_image_raw*). Pode-se verificar também que o nó */arm* comunica-se com o nó */move_group* e com o nó */timon_arm_controller* a fim de enviar a trajetória planejada pelo *MoveIt* para os motores *dynamixel*.

3.4 Simulação do sistema

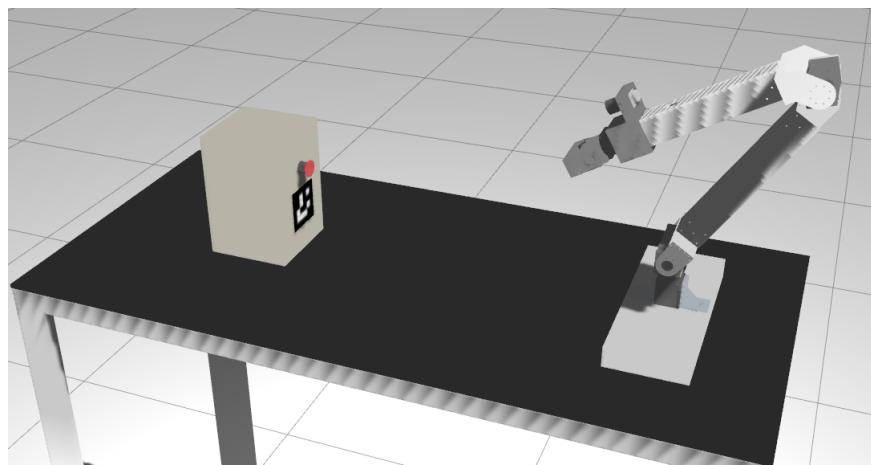
A simulação do robô JeRoTIMON inserido em seu ambiente de trabalho foi realizada na plataforma *Gazebo*. Para isto, realizou-se o desenvolvimento dos arquivos *timon_arm.urdf* e *box.urdf*, modelos *URDF* que descrevem o robô e o painel elétrico a ser acionado. Foram levados em consideração características de massa, inércia e dimensão para cada componente destes modelos, para que houvesse maior fidelidade possível com o que representam fisicamente, de forma a garantir que os testes realizados possam ser validados em ambiente real.

O pacote *ros_control* dispõe de uma lista de controladores disponíveis. Para JeRoTIMON, foram utilizados controladores do tipo *position_controllers/JointTrajectoryController*, e as transmissões para cada junta do manipulador foram definidas em seu modelo *URDF*.

Para simulação dâ câmera RGB, foi desenvolvido o arquivo *camera.xacro*. O plugin padrão do *Gazebo* foi utilizado, *ligazebo_ros_camera.so* e as especificações da câmera RGB *Teledyne Genie Nano C2590* foram levados em consideração, garantindo maior realismo à simulação. A figura 10 exibe os modelos simulados do manipulador e do painel elétrico,

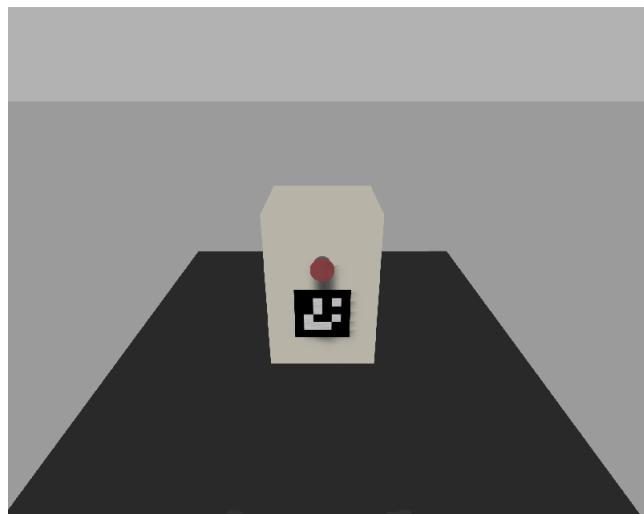
enquanto a figura 11 exibe imagem capturada pela câmera instalada no manipulador.

Figura 10: Modelos simulados do manipulador e do painel elétrico.



Fonte: Autoria própria.

Figura 11: Imagem capturada pela câmera RGB.



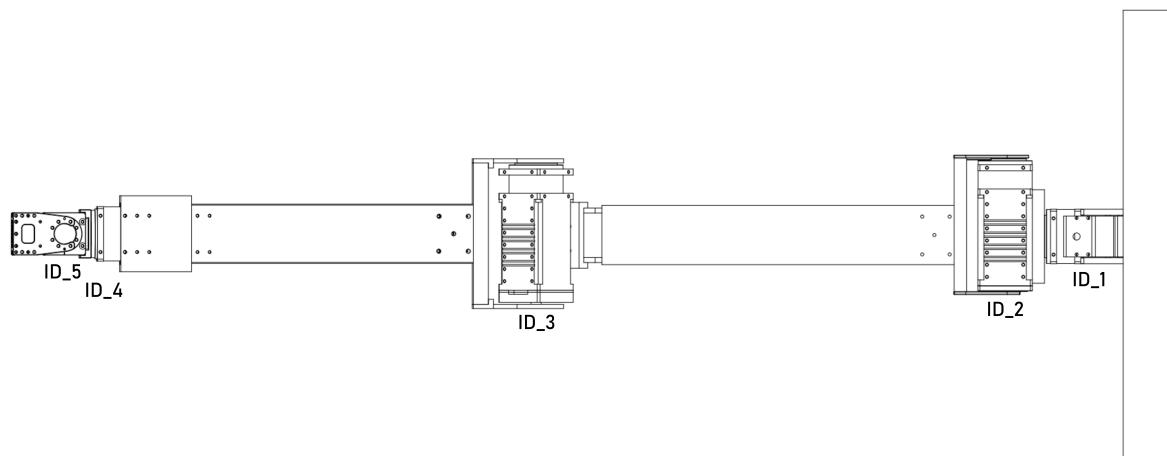
Fonte: Autoria própria.

4 IMPLEMENTAÇÃO

4.1 Parametrização dos motores

Na Tabela 3 encontram-se as configurações para os motores, que estão identificados na Figura 12. Nesta Tabela estão especificados os ângulos e as posições mínimas e máximas definidas para cada motor. Estes parâmetros foram escolhidos após testes e verificações de acordo com a capacidade do manipulador interagir com o ambiente de trabalho, pensando nas possíveis localizações que a caixa poderá estar situada.

Figura 12: Identificação dos motores.



Fonte: Autoria própria.

Tabela 3: Parâmetros dos motores.

Motor	Ângulo (min)	Posição (min)	Ângulo (máx)	Posição (máx)
ID_1	-45°	-125000	45°	125000
ID_2	-90°	-250962	90°	250962
ID_3	-43°	-119095	173°	483855
ID_4	-170°	-475464	170°	475464
ID_5	-90°	-151875	90°	151875

Fonte: Autoria própria.

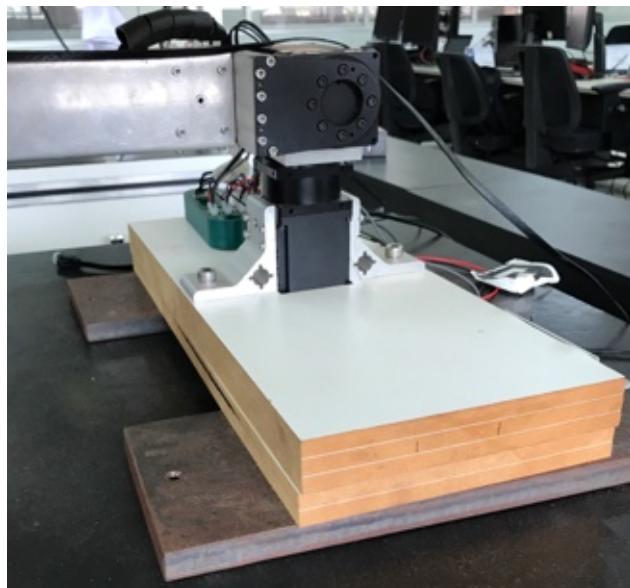
4.2 Estrutura física

Após montagem física, o manipulador obteve um alcance de aproximadamente 980 mm. Levou-se em consideração o mínimo alcance necessário para que a missão possa ser realizada conforme as especificações do cliente. Então foram projetadas cinco juntas rotacionais como mostra a Figura 12.

4.2.1 Base

Foi decidido utilizar uma base de madeira com dimensões de aproximadamente 450×180 mm e espessura de 50 mm (Figura 13). Esta base será utilizada para fixar o manipulador, garantindo estabilidade durante a execução da tarefa.

Figura 13: Base do manipulador.



Fonte: Autoria própria.

4.2.2 Elos ou *links*

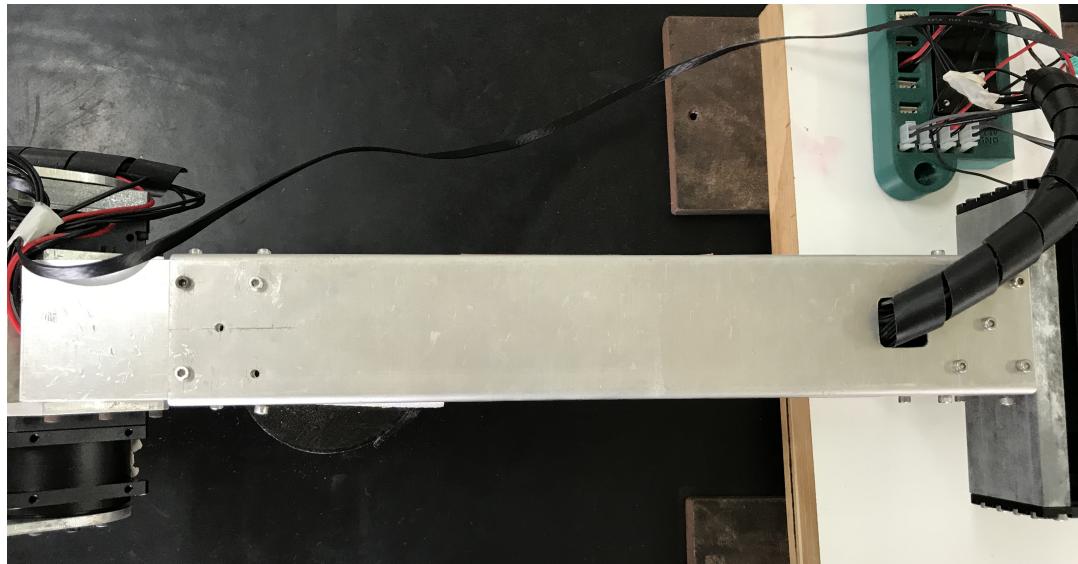
Para estruturação dos elos do robô são utilizados dois perfis de alumínio vazados com comprimento, largura e altura de $350 \times 58.7 \times 58.7$ mm, respectivamente (Figuras 14 e 15). Estas dimensões foram escolhidas levando em consideração a disponibilidade dos materiais, capacidade de alcance do braço para realizar a tarefa e capacidade estrutural do manipulador, para assim suportar esforços de natureza estática e dinâmica.

Figura 14: Vista lateral do *link* 2.



Fonte: Autoria própria

Figura 15: Vista superior do *link 2*.



Fonte: Autoria própria

4.2.3 Suportes

Foram utilizados suportes com finalidade de unir os elos do manipulador e distribuir movimentos rotativos para as juntas seguintes, feitos em alumínio. Alguns suportes foram adquiridos através da ROBOTIS, enquanto outros foram confeccionados pelo laboratório CCRoSA. A Tabela com os suportes utilizados e suas imagens e medidas pode ser vista no apêndice E.

4.2.4 Câmera

Para prover o sistema com capacidade de detecção da *tag* e assim obter dados necessários para aquisição de pose e orientação relativa, utilizou-se uma câmera de vídeo modelo Teledyne Genie Nano C2590 (Figura 16) e foi acoplada a lente 16mm C Series VIS-NIR (Figura 17). A comunicação entre a câmera e o sistema é feita através de um cabo categoria 6 RJ45, e sua ligação pode ser vista no apêndice I.

Figura 16: Teledyne Genie Nano C2590.



Fonte: ([TELEDYNE...](#), s.d.)

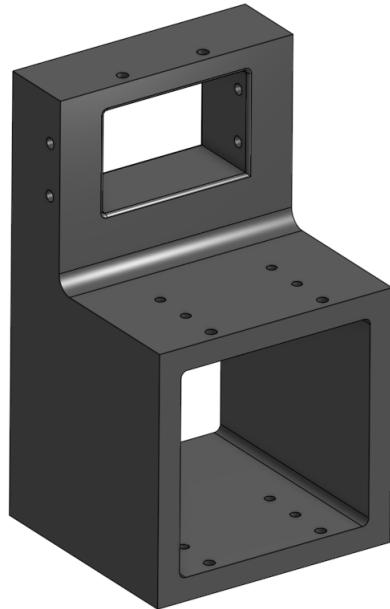
Figura 17: Lente 16mm C Series VIS-NIR.



Fonte: ([16MM...](#), s.d.)

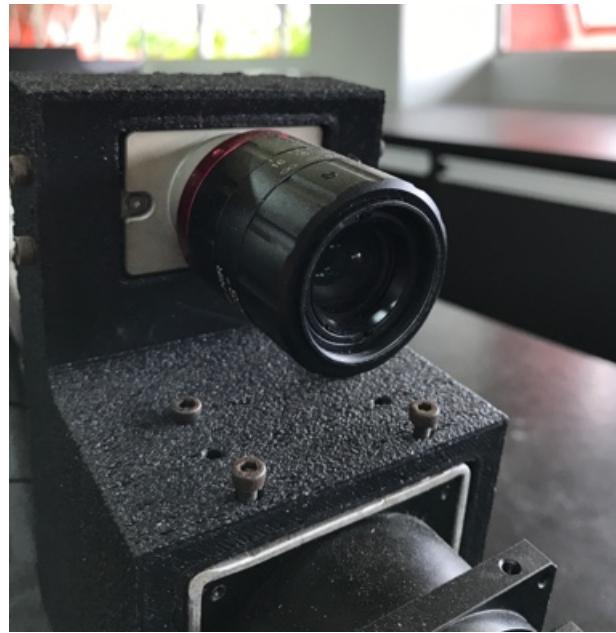
Este conjunto pode ser fixado próxima a extremidade do manipulador robótico utilizando um suporte impresso em ABS (Figura 18), facilitando a detecção da *tag* e não comprometendo a estrutura do manipulador. O suporte é conectado ao final do *link 3*, como mostra o apêndice D. A integração pode ser visualizada na Figura 19.

Figura 18: Suporte para fixação da câmera.



Fonte: Autoria própria

Figura 19: Integração suporte-câmera-lente.



Fonte: Autoria própria

4.2.5 Atuadores

Os atuadores do manipulador são motores de corrente contínua, integrados com redutor de velocidade, controlador e driver. Foram utilizados atuadores *Dynamixel* da fabricante

ROBOTIS. Entre os modelos figuram o PH54-200-S500-R, presente nas juntas 0, 1 e 2, o motor PH54-100-S500-R para a junta 3 e o motor PH42-020-S300-R foi utilizado para a junta 4. As especificações do fabricante mais relevantes no projeto estão apresentadas nas tabelas 4 e 5. As folhas de dados estão disponíveis nos anexos B, C e D para os atuadores PH42-020-S300-R, PH54-100-S500-R, PH54-200-S500-R, respectivamente.

Tabela 4: Parâmetros dos motores.

Tipo	Torque (N.m)	Tensão (V)	Corrente (A)	Velocidade de Rotação (rpm)	Dimensões (mm)
PH54-200-S500-R	44.7	24.0	9.3	29.0	54.0 X 126.0 X 54.0
PH54-100-S500-R	25.3	24.0	5.5	29.2	54.0 X 108.0 X 54.0
PH42-020-S300-R	5.1	24.0	1.5	29.2	42.0 X 84.0 X 42.0

Fonte: ([ROBOTIS, s.d.](#))

4.3 Sistema de Potência

O sistema necessitará ser energizado em dois níveis de tensão, 12 V e 24 V. Durante a realização dos testes, utilizou-se uma fonte com níveis de tensão de 0-30 V e fornecimento de corrente de 0-10 A, para atender os requisitos necessários de potência dos motores. Foi utilizado um conversor DC-DC modelo UWE-12/10-Q12PB-C (Figura 20) na saída da fonte para atingir o nível de tensão de 12 V e energizar a câmera. Um esquema elétrico é fornecido no apêndice H, indicando as conexões dos cabos entre os motores e a câmera.

Figura 20: UWE-12/10-Q12PB-C.



Fonte: ([UWE-12/10-Q12PB-C, s.d.](#))

4.4 Comunicação

O manipulador é conectado via USB através de um dispositivo denominado U2D2 ([U2D2, s.d.](#)). Ele consegue parametrizar e enviar comandos para os motores através dos softwares originais da ROBOTIS chamados Dynamixel Workbench e Dynamixel Wizard. O padrão elétrico utilizado é RS-485, ou seja, utiliza-se de 4 fios de conexão que transmitem níveis de tensão e dados. A conexão de entre o computador e os motores, utilizando o U2D2 de

intermédio, pode ser vista no apêndice I. A velocidade de transmissão (*baud rate*) definida para todos os motores no sistema é de 57600 bps.

Figura 21: U2D2.



Fonte: ([U2D2, s.d.](#))

4.5 Análise de esforços

A análise de esforços mecânicos aos quais o manipulador está exposto foi realizada para embasamento e confirmação da capacidade da estrutura e dos atuadores suportarem os esforços de momento torsor. Para isso foram analisadas individualmente as juntas em seus estados críticos. Na Tabela 5 é apresentada a análise de esforços nas juntas e comparados com os máximos esforços suportados pelos atuadores (conforme vistos nos apêndices B, C e D), para evitar falhas mecânicas.

Levando em consideração os valores obtidos nas análises de esforços, tem-se as juntas 0 e 1 como juntas críticas do sistema, pois elas sofrem a maior força do sistema e tem o maior torque exigido. Na configuração apresentada, as juntas 0 e 1 suportam a carga máxima de 45.22 N sem chegar a falha mecânica, logo o manipulador possui um limite (*payload*) de aproximadamente 2.1 kg na sua extremidade.

Tabela 5: Torque das juntas.

Junta	Torque máximo fornecido pelos motores (Nm)	Força resultante na junta (N)	Torque exigido pela junta (Nm)
0	44.7	45.22	25.8
1	44.7	45.22	25.8
2	44.7	21.44	6.9
3	25.3	4.8	0.12
4	5.1	0.87	0.05

Fonte: Autoria própria.

4.6 Configurações

O conjunto formado pelo manipulador, seu espaço de trabalho e os objetos com os quais ele deverá interagir compõem o sistema abordado neste trabalho. O espaço de trabalho foi representado como uma bancada de $1.70\text{ m} \times 0.80\text{ m} \times 0.028\text{ m}$. Há apenas um objeto com o qual o manipulador deverá interagir, uma caixa de dimensões $0.30\text{ m} \times 0.20\text{ m} \times 0.20\text{ m}$.

Para o desafio foi utilizado um marcador visual *ArUco* com dimensões $58\times 58\text{ mm}$ de id 4, conforme mostrado na Figura 22. Este tem 8 cm do centro do *ArUco* para o botão e 7 cm do centro do *ArUco* para a lâmpada, esta identifica se o botão foi acionado ou não pelo manipulador.

Figura 22: Caixa objetivo.



Fonte: Autoria própria

O manipulador possui a posição home definida como da Figura 23 onde o ângulo de cada motor está descrito na Tabela 6.

Figura 23: Manipulador na posição home.



Fonte: Autoria própria.

Tabela 6: Ângulos dos motores na posição home.

Motor	Ângulo
ID_1	0°
ID_2	-24°
ID_3	141°
ID_4	0°
ID_5	0°

Fonte: Autoria própria.

5 RESULTADOS E ANÁLISES

Os experimentos apresentados nesta seção possuem como objetivo avaliar o desempenho do manipulador robótico desenvolvido.

5.1 Caracterização do problema e determinação do modelo

Para análise da performance do manipulador, deseja-se avaliar inicialmente a interação de algumas variáveis que podem influenciar no desempenho do manipulador. Em um segundo momento, após estabelecidas as melhores configurações de algoritmo e velocidade para o robô, os testes devem seguir uma análise de repetibilidade e de variância do sistema.

No planejamento dos experimentos, foram levantadas quais variáveis seriam de interesse para avaliar a performance do robô, com isso foram selecionadas como variáveis de saída a precisão do manipulador, ou seja, a capacidade do mesmo de chegar a uma posição determinada com o menor erro possível, tempo de busca do marcador visual, e o tempo necessário para o acionamento do botão (missão imposta ao manipulador).

Também foram selecionadas as variáveis independentes, ou seja, as variáveis de entrada do processo, que serão analisadas quanto às suas influências no sistema de forma isolada ou a interação entre as mesmas. As variáveis selecionadas para o estudo foram: velocidade de operação dos motores, algoritmo de planejamento de trajetória utilizado e a posição da caixa no ambiente de trabalho do manipulador.

5.2 Planejamento dos experimentos

Definido o problema e as variáveis a serem estudadas, foi realizado um planejamento dos experimentos, para assim gerar dados e resultados confiáveis acerca da melhor configuração para o manipulador. As tabelas 7, 8 e 9 apresentam os modelos dos experimentos.

Tabela 7: Modelo de experimentos para análise de 3 variáveis.

Variáveis Independentes	Níveis		Variáveis dependentes
Algoritmo de planejamento de trajetória	RRT-CONNECT	Kpiece	Erro de posição
Velocidade dos motores	300	500	Tempo total
Posição da caixa	A	B	

Fonte: Autoria própria.

Tabela 8: Modelo de experimentos ANOVA sem detecção da *tag*.

Ponto	Número de repetições	Variáveis dependentes
A	10	Erro de posição
B	10	Erro de posição

Fonte: Autoria própria.

Tabela 9: Modelo de experimentos ANOVA com detecção da *tag*

Operador	Pose da caixa	Número de regravações	Sucesso	Variáveis dependentes
1	A	10	-1,1	Erro de posição
2	B	10	-1,1	Tempo de busca
1	B	10	-1,1	Tempo total
2	A	10	-1,1	

Fonte: Autoria própria.

5.3 Resultados alcançados

A partir dos testes realizados sob orientação do planejamento, foram coletados os resultados do tempo de busca do marcador visual, tempo da missão do manipulador e o erro de posição em cada experimento. As amostras dos testes coletadas foram analisados da seguinte forma: Análise de regressão linear, análise de variância e teste R&R, para os testes com e sem a detecção do marcador visual .

5.3.1 Análise de regressão linear

A partir do planejamento de experimentos realizados, foram tomados três dados de cada condição estabelecida e utilizada a média entre os três resultados como valor do experimento. As amostras coletadas estão representadas na tabela 10. O algoritmo de planejamento RRT CONNECT é representado pelo sinal de subtração (-) enquanto o sinal de adição (+) representa o algoritmo Kpiece. Já para as velocidades, (+) representa o valor de 500 rpm e (-) representa 300 rpm.

Tabela 10: Resultados das amostras coletadas.

Ordem	Algoritmo	Velocidade	Erro	Tempo
1	-	-	0.0004759353	175.25667
2	+	-	0.0009316073	82.24667
3	-	+	0.0002889380	139.39333
4	+	+	0.0008311057	90.34333

Fonte: Autoria própria.

O resultado obtido a partir da análise linear aplicada aos dados é disposto na tabela 11. O primeiro modelo dessa análise pode ser visualizado na Equação 5.1. Esta equação

mostra um modelo que consegue explicar o valor do erro de posição a partir da relação entre os parâmetros.

Tabela 11: Resultados da análise de regressão linear.

	Estimate	Std.	Error	t value	Pr(> t)
(Intercept)	4.543e-04	3.745e-05	12.130	0.0524	
algoritmo+	4.989e-04	4.325e-05	11.536	0.0550	
velocidade+	-1.437e-04	4.325e-05	-3.324	0.1860	

Fonte: Autoria própria.

$$\text{erro} = 4.543.(10^{-04}) + 4.989.(10^{-04}).\text{algoritmo}(+) - 1.437.(10^{-04}).\text{velocidade}(+) \quad (5.1)$$

Observando os dados apresentados na tabela 11, o ρ -valor da variável velocidade é superior a 0,05, aceitando a hipótese nula, ou seja, a variável velocidade possui pouca ou nenhuma influência na tomada de dados do sistema. Porém, percebe-se que o ρ -valor da variável algoritmo é próxima de 0.05, implicando que a mesma pode ser analisada de forma individual, rejeitando-se a hipótese nula para esta variável. Desta forma refazendo-se a análise linear do erro de posição com relação apenas a variável algoritmo obtém-se o novo modelo representado na equação 5.2. O resultado dos coeficientes pode ser visto na tabela 12.

Tabela 12: Resultados da análise de regressão linear (variável algoritmo).

	Estimate	Std.	Error	t value	Pr(> t)
(Intercept)	3.824e-04	7.506e-05	5.095	0.0364	
algoritmo+	4.989e-04	1.062e-04	4.700	0.0424	

Fonte: Autoria própria.

$$\text{erro} = 3.824.(10^{-04}) + 4.989.(10^{-04}).\text{algoritmo}(+) \quad (5.2)$$

Como apresentado na tabela 12 o algoritmo apresenta ρ -valor menor que 0.05, ou seja, rejeita-se a hipótese nula para esta variável, mostrando que a mesma tem influência no erro de posição do manipulador.

Após esta análise serão realizados novos testes levando em consideração as variáveis examinadas neste estudo, sendo configurado o sistema com o algoritmo RRT CONNECT, uma vez que o algoritmo Kpiece teve influência positiva, ou seja, aumentou o erro de posição do manipulador, o que não é desejável para este sistema. A velocidade dos motores não apresentou influência considerável nos testes, portanto foi configurada com a menor velocidade por motivos de segurança.

5.3.2 ANOVA

Com o algoritmo de planejamento e velocidade estabelecidas para a continuidade dos testes, foram realizados dois tipos de experimentos: análise de erro de posição do *end-effector* sem a detecção do marcador visual, ou seja, o manipulador recebe comandos de posição em x, y e z dentro da sua área de trabalho e então é observado o erro ao executar o comando de posição. O segundo teste foi semelhante, porém, este contou com a detecção da *tag*, recebendo assim o comando de posição a partir da detecção do marcador visual. Com isso foram tomados dados em duas diferentes poses para assim analisar a variância do erro em cada pose e entre elas.

5.3.2.1 Análise de variância sem detecção da *tag*

Para esta análise foram estabelecidas, na área de trabalho do manipulador, duas posições, onde Pose A equilava a $x = 0.314445$ $y = 0.730867$ $z = 0.233471$ e Pose B equilava a $x = -0.153158$ $y = 0.755479$ $z = 0.238906$, ambas referentes ao *endeffecter* do manipulador, e então foram realizados 10 execuções de movimento para cada pose estabelecida. A tabela 13 mostra os erros de posição do *endeffecter* para cada uma das execuções.

Tabela 13: Resultados dos experimentos sem detecção da *tag*.

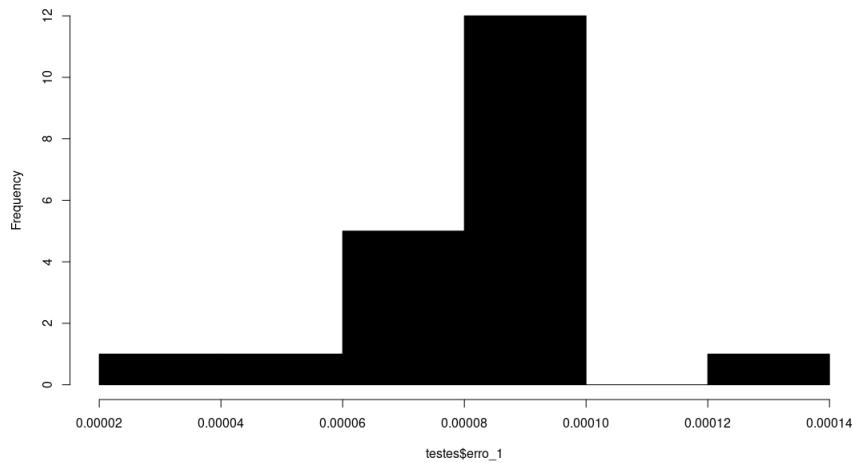
	Pose	Erro
1	A	0.000135296
2	A	0.000095000
3	A	0.000090747
4	A	0.000097944
5	A	0.000081093
6	A	0.000068184
7	A	0.000088978
8	A	0.000074532
9	A	0.000036373
10	A	0.000086499
11	B	0.000095900
12	B	0.000085300
13	B	0.000064900
14	B	0.000089500
15	B	0.000089000
16	B	0.000048300
17	B	0.000096700
18	B	0.000090200
19	B	0.000077800
20	B	0.000079800

Fonte: Autoria própria.

Para avaliar os dados, inicialmente foi realizado um teste de normalidade, como forma

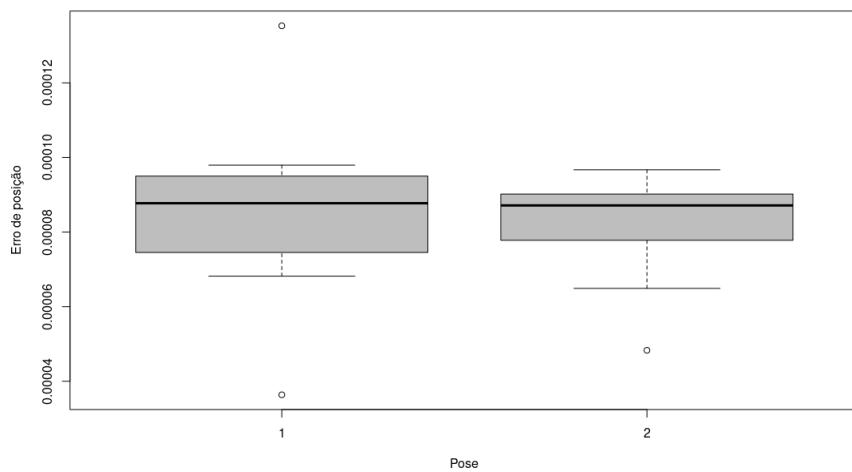
de verificar se os dados são bem modelados por uma distribuição normal ou não. O teste de Shapiro-Wilk foi utilizado para esse conjunto de dados, retornando um ρ -valor de 0.05569, o que indica que a hipótese nula não pode ser rejeitada, logo os dados seguem uma distribuição normal. A distribuição dos mesmos pode ser melhor visualizada através de um histograma, como mostra a figura 24.

Figura 24: Histograma do conjunto de dados sem detecção da *tag*.



Segundo a análise dos testes, foi realizado um teste ANOVA com a amostra de dados coletada. O objetivo foi avaliar a variância dos dados em cada ponto e entre eles, analisando se estes diferem estatisticamente entre si. Com esta análise, obteve-se um ρ -valor de 0.6912, com isso tem-se embasamento estatístico para afirmar que os dados no ponto "A" e no ponto "B" são estatisticamente iguais, os erros de pose apresentados nos dois pontos não diferem entre si. Esta análise pode ser visualizada na figura 25.

Figura 25: Boxplot do conjunto de dados sem detecção da *tag*.



Observando a figura 25 a pose 1 (A) possui maior variação dos seus dados em relação a pose 2 (B), porém as duas não diferem estatisticamente entre si.

5.3.2.2 Análise de variância com detecção da *tag*

Foi tomado um novo conjunto de dados, como mostra a tabela 14. Este experimento foi realizado com variação do operador e posição da caixa. Foram tomados dados de tempo de busca da *tag*, tempo total da missão e o erro de posição do *end-effector*. Também foi observado a porcentagem de sucesso na conclusão da missão, sendo o índice '1' correspondente ao sucesso na missão e o índice '-1' correspondente à falha.

Tabela 14: Resultados dos experimentos sem detecção da *tag*.

	Operador	Pose	Erro de pose	Tempo de busca	Tempo total	Sucesso da missão
1	1	A	0.010451819	27.1022	93.9316	1
2	1	A	0.009745817	29.1912	91.8192	1
3	1	A	0.010017589	27.3904	95.6437	1
4	1	A	0.010819201	28.4745	90.5214	1
5	1	A	0.011133634	27.7250	98.7845	1
6	1	A	0.010562436	28.8388	94.6226	1
7	1	A	0.010132297	29.1519	101.8420	1
8	1	A	0.010693537	30.2186	90.1885	1
9	2	A	0.010905047	30.2317	94.0104	1
10	2	A	0.010469991	28.9793	96.2312	1
11	2	A	0.010355747	29.0720	103.6160	1
12	2	A	0.010268870	28.6952	93.0651	1
13	2	A	0.010230915	28.6420	87.2120	1
14	2	A	0.004079350	26.8585	86.4156	1
15	2	A	0.004427968	26.1808	85.7109	1
16	2	A	0.009985857	30.0858	90.4296	1
17	1	A	0.009754491	28.2042	86.8758	1
18	1	A	0.010334966	28.6983	92.2869	1
19	1	A	0.010058060	29.4320	94.3545	1
20	1	A	0.010153228	28.1812	90.6696	1
21	1	A	0.010120659	29.2442	87.1852	1
22	1	A	0.010105795	28.9948	88.3569	1
23	1	A	0.010105795	28.9948	88.3569	-1
24	1	A	0.011557242	32.9496	102.3640	1
25	2	A	0.011245238	34.7259	98.7636	1
26	2	A	0.011190817	31.1780	113.0500	1
27	2	A	0.010174117	28.7923	88.9735	1
28	2	A	0.010007278	28.8178	91.8702	1
29	2	A	0.009963924	29.0347	86.7183	1
30	2	A	0.010475097	28.2422	99.1881	1
31	2	A	0.010741058	28.2118	91.7459	1
32	2	A	0.004469904	28.4826	88.1583	-1
33	1	A	0.010086673	28.7112	87.9372	1
34	1	A	0.010515376	28.8792	92.9002	1
35	1	A	0.010062717	29.1420	91.5910	1
36	1	A	0.010422717	29.0783	94.3296	1
37	1	A	0.010185682	27.9274	88.6971	1
38	1	A	0.010281946	29.7805	100.1800	1
39	1	A	0.013133281	20.0540	82.3651	-1
40	1	A	0.012344603	19.1706	81.7740	1

	Operador	Pose	Erro de pose	Tempo de busca	Tempo total	Sucesso da missão
41	2	B	0.005245180	20.4070	84.7338	1
42	2	B	0.007158873	20.4408	84.3043	1
43	2	B	0.011482759	20.7824	83.2411	1
44	2	B	0.087866430	23.4152	87.9248	-1
45	2	B	0.008514302	19.9797	86.4344	1
46	2	B	0.005144988	19.8502	84.5138	1
47	2	B	0.013200185	19.9392	85.7667	-1
48	2	B	0.004874412	20.2860	80.6176	1
49	1	B	0.012512453	20.7894	85.9667	1
50	1	B	0.014427715	19.4977	81.7383	-1
51	1	B	0.007453902	20.5577	82.4847	1
52	1	B	0.006078841	20.4143	81.6851	1
53	1	B	0.009550672	19.5227	84.7312	1
54	1	B	0.006123619	16.9920	74.6466	1
55	1	B	0.003705978	18.6949	80.1175	1
56	1	B	0.006896455	19.9972	76.2407	1
57	2	B	0.006869856	18.8696	79.7714	1
58	2	B	0.002917472	18.4679	73.9755	1
59	2	B	0.004802752	19.4849	80.2423	1
60	2	B	0.004557056	19.5606	77.7174	1
61	2	B	0.006353444	18.5702	76.3386	1
62	2	B	0.007120232	19.6487	78.9122	1
63	2	B	0.004832266	17.0441	77.3120	1
64	2	B	0.004535360	18.0856	77.5731	1
65	1	B	0.007897063	17.1585	78.1917	1
66	1	B	0.007330963	20.5918	81.1574	1
67	1	B	0.008284862	20.2515	79.0624	1
68	1	B	0.008096144	19.9652	81.4914	1
69	1	B	0.013425916	19.0627	79.8650	-1
70	1	B	0.007966466	18.6261	75.1340	1
71	1	B	0.008486843	18.1093	72.8817	1
72	1	B	0.007759457	18.7750	77.1361	1
73	2	B	0.012489658	20.1018	77.2422	1
74	2	B	0.008122309	18.6224	77.0045	1
75	2	B	0.012097601	20.3431	79.8706	1
76	2	B	0.087776215	17.5113	77.7186	1
77	2	B	0.006999748	17.3791	74.1548	1
78	2	B	0.008223493	18.3666	76.5340	1
79	2	B	0.008721295	19.4229	73.9818	1
80	2	B	0.008375202	18.6691	73.7671	1

Fonte: Autoria própria.

Observando a tabela 14, os testes apresentaram 91.25% de sucesso, ou seja, dos 80 testes realizados, em 73 obteve-se sucesso no pressionamento do botão. A tabela 15 resume alguns indicadores importantes das três saídas do sistema (tempo de busca, tempo total da missão e erro de pose).

Os testes de normalidade apresentaram valores de ρ -valor inferiores a 0.05, ou seja, a hipótese nula não pode ser rejeitada, logo os dados seguem uma distribuição normal.

Tabela 15: Resumo dos conjuntos de dados das variáveis de saída.

	Média	Desvio padrão	ρ -valor (Shapiro-Wilk)
Tempo de busca	23.95025	5.035711	1.103e-07
Tempo total	86.06149	8.343288	0.02409
Erro de pose	0.01095061	0.0126478	2.2e-16

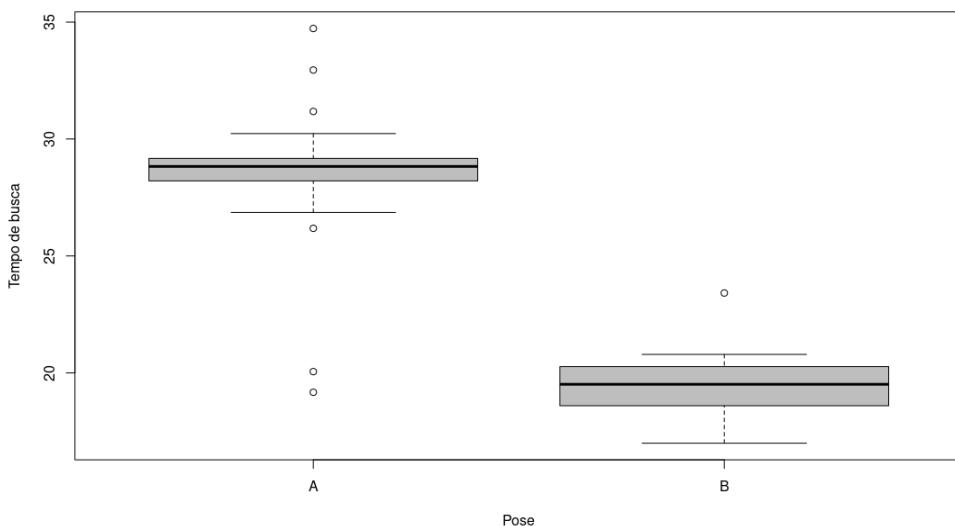
Fonte: Autoria própria.

Tabela 16: Tabela dos valores de ρ -valor do teste ANOVA.

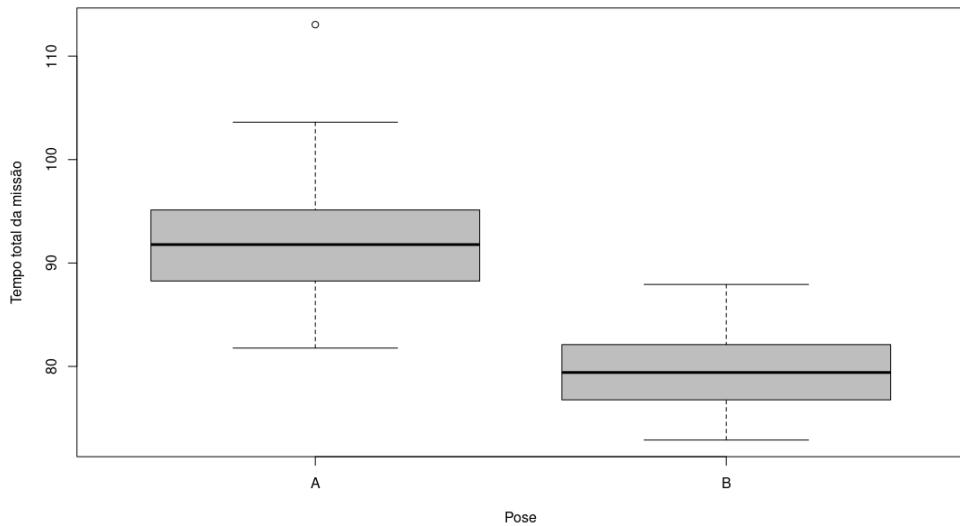
	p-valor
Erro de pose	0.525
Tempo de busca	2.2e-16
Tempo total	2.2e-16

Fonte: Autoria própria.

Realizando o teste do ANOVA, podemos visualizar a variância dos pontos A e B em relação ao erro de pose, tempo de busca e tempo total da missão. A tabela 16 apresenta o ρ -valor para cada conjunto de dados, e a partir dessa análise podemos chegar a conclusão que a hipótese nula não pode ser rejeitada para o erro de pose assim como visto na análise passada, ou seja, os dados obtidos do erro de posição do *end-effector* nos pontos A e B são estatisticamente iguais. Já os dados dos tempo de busca e tempo total da missão apresentam ρ -valor menor que 0.05, logo a hipótese nula é rejeitada e pode-se concluir que os dados diferem entre si, como mostram os *boxplots* das figuras 26 e 27.

Figura 26: *Boxplot* da variância do tempo de busca entre os pontos A e B.

Fonte: Autoria própria.

Figura 27: *Boxplot* da variância do tempo total da missão entre os pontos A e B.

Fonte: Autoria própria.

5.3.3 Análise de repetibilidade e reproduzibilidade

Como forma de avaliação do sistema robótico, também foi adotado um estudo de repetibilidade e reproduzibilidade do sistema. Com isso, foram adotados os dados da tabela 14. Foram tomadas 20 medições de cada operador para cada pose da caixa, totalizando 80 testes e assim foram obtidos os seguintes dados apresentados na figura 28 e 29.

Observando a figura 28, correspondente ao teste de R&R para o tempo busca do marcador visual, neste teste percebe-se o número de categorias igual a 6, número considerável e acima de valores de referências, o que demonstra que o sistema de medição pode ser utilizado para análise dos dados. Outro ponto importante para observado é o indicador "*Total Gage R&R*", que foi calculado em 20.37, inferior a 30% (Valor máximo aceitável adotado em algumas literaturas).

O sistema sendo aceitável para análise, pode-se observar outros indicadores importantes para análise, como exemplo do ρ -valor, que mostra a maior influência na variação devido a pose da caixa, não ocorrendo influência do operador ou da interação entre operador e pose. O indicador "*Part-to-Part*" indica que aproximadamente 98% da variação do sistema deve-se a própria variação de pose da caixa, o que demonstra que o sistema possui repetibilidade e reproduzibilidade, quanto a variável de tempo de busca.

Figura 28: Modelo completo teste R&R para tempo de busca.

```

Complete model (with interaction):
      Df Sum Sq Mean Sq F value Pr(>F)
pose_rep        1 1902.1 1902.1 403.599 0.0317
operador       1     0.1     0.1   0.012 0.9308
pose_rep:operador 1     4.7     4.7   2.450 0.1215
Repeatability   80 153.9    1.9
Total          83 2060.7

alpha for removing interaction: 0.05

Reduced model (without interaction):
      Df Sum Sq Mean Sq F value Pr(>F)
pose_rep        1 1902.1 1902.1 971.443 <2e-16
operador       1     0.1     0.1   0.029 0.866
Repeatability  81 158.6    2.0
Total          83 2060.7

Gage R&R

      VarComp %Contrib
Total Gage R&R  1.957994  4.15
  Repeatability  1.957994  4.15
  Reproducibility 0.000000  0.00
    operador     0.000000  0.00
  Part-To-Part   45.240977 95.85
  Total Variation 47.198971 100.00

      StdDev StudyVar %StudyVar
Total Gage R&R  1.399283  8.39570   20.37
  Repeatability  1.399283  8.39570   20.37
  Reproducibility 0.000000  0.00000   0.00
    operador     0.000000  0.00000   0.00
  Part-To-Part   6.726141 40.35685   97.90
  Total Variation 6.870151 41.22090   100.00

Number of Distinct Categories = 6

```

Fonte: Autoria própria.

Figura 29: Modelo completo teste R&R para tempo total da missão.

```

Complete model (with interaction):
      Df Sum Sq Mean Sq F value Pr(>F)
pose_rep        1   3659    3659 431.235 0.0306
operador       1     2      2   0.272 0.6941
pose_rep:operador 1     8      8   0.351 0.5550
Repeatability   80 1932     24
Total          83 5602

alpha for removing interaction: 0.05

Reduced model (without interaction):
      Df Sum Sq Mean Sq F value Pr(>F)
pose_rep        1   3659    3659 152.736 <2e-16
operador       1     2      2   0.096 0.757
Repeatability  81 1941     24
Total          83 5602

Gage R&R

      VarComp %Contrib
Total Gage R&R  23.95875  21.68
  Repeatability  23.95875  21.68
  Reproducibility 0.00000  0.00
    operador     0.00000  0.00
  Part-To-Part   86.55716  78.32
  Total Variation 110.51591 100.00

      StdDev StudyVar %StudyVar
Total Gage R&R  4.894768 29.36861   46.56
  Repeatability  4.894768 29.36861   46.56
  Reproducibility 0.000000  0.00000   0.00
    operador     0.000000  0.00000   0.00
  Part-To-Part   9.303610 55.82166   88.50
  Total Variation 10.512655 63.07593   100.00

Number of Distinct Categories = 2

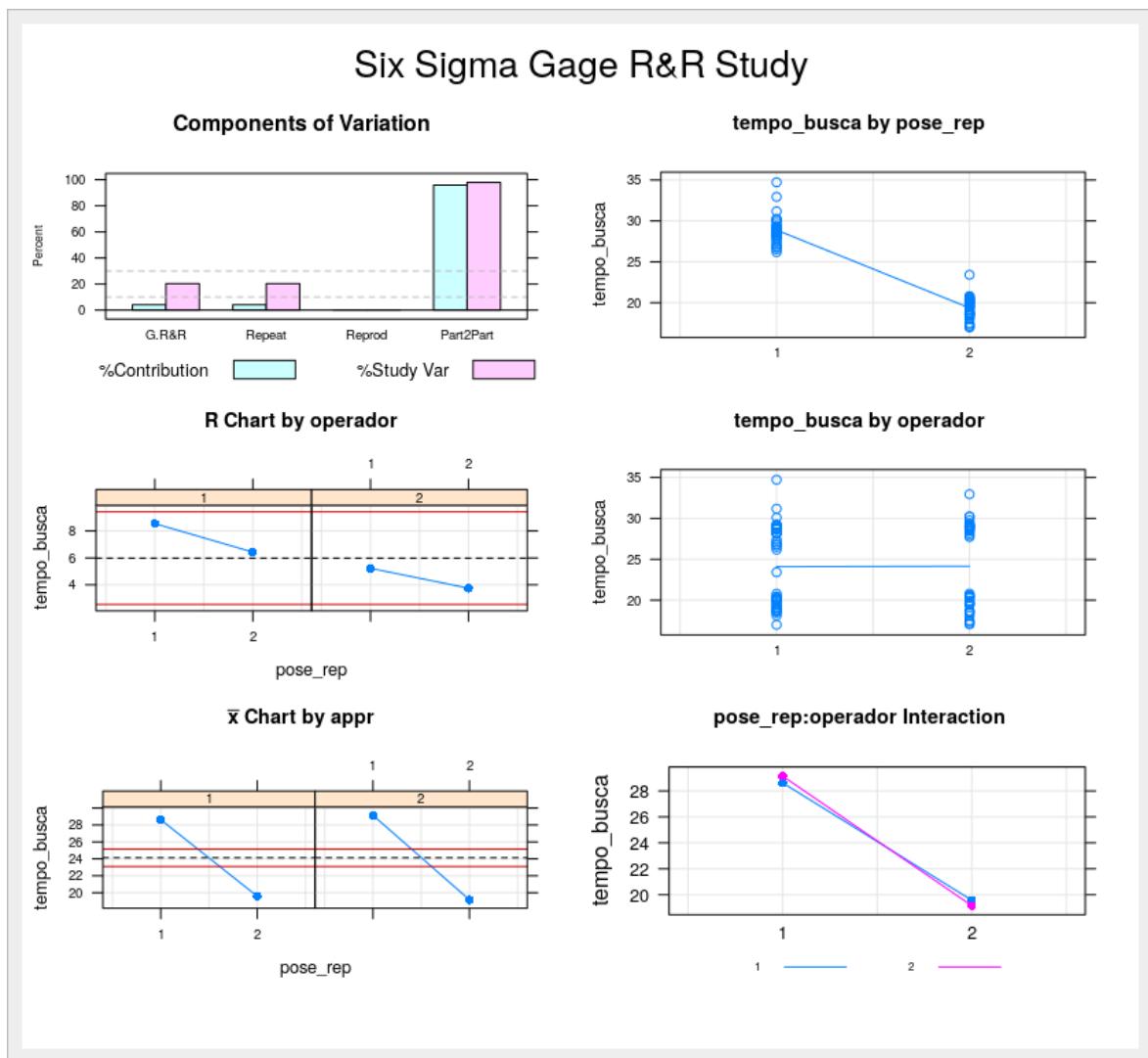
```

Fonte: Autoria própria.

O mesmo não pode ser observado na figura 29, onde o número de categorias apresentados é baixo e o índice "Total Gage R&R" foi calculado em 46.56%, acima do valor máximo de referência (30%), o que mostra que o sistema de medição não é aceitável quando observado a variável de tempo total da missão.

Segundo a análise dos dados obtidos no estudo de repetibilidade e reprodutibilidade do sistema para o tempo de busca, visualizamos cartas na figura 30 que possibilita um melhor entendimento do sistema de medição.

Figura 30: Gráficos do estudo R&R para tempo de busca.



Fonte: Autoria própria.

A carta "Componentes de variação" representa atribuições da variação do sistema, e como observado na figura 30, a variação deve-se principalmente a "peça por peça" ou "parte por parte", o que é desejável, demonstrando que o sistema de medição é aceitável para análise do tempo de busca do manipulador.

A carta R é uma carta de controle de amplitudes que representa graficamente a consistência do operador. Os pontos apresentados na figura 30 estão dentro dos limites máximos e mínimos, ou seja, é sinal de que os operadores estão realizando os testes de forma consistente.

Os pontos apresentados na carta Xbar, na figura 30, os dados estão acima ou abaixo dos limites de controle. Estes resultados indicam que a variação entre as poses são maiores que a variação do dispositivo de medição.

Outro ponto importante de ser destacado é o gráfico de interação entre o operador e o tempo de busca (figura 30), a barra horizontal visualizada, demonstra que os operadores estão realizando medições semelhantes entre si, com variações semelhantes entre as medidas máximas e mínimas do tempo de busca.

5.4 Conclusões dos testes estatísticos

Após estudo dos dados obtidos a partir de testes no manipulador robótico, verifica-se que os dados apresentaram distribuição normal, o que possibilita uma análise confiável dos mesmos, comprovando isso através de histogramas apresentados durante a seção.

Também é possível observar nos testes realizados que o algoritmo de planejamento de trajetória RRT CONNECT apresentou melhor resultado em relação ao Kpiece, bem como a velocidade pouco influenciou nos testes, mostrando que a melhor configuração para o sistema para as missões impostas é utilizando o algoritmo RRT.

Não houve diferença no erro de posicionamento do *end-effector* quando observado as poses A e B e com ou sem detecção do marcador visual, ou seja, pode-se concluir que o manipulador robótico apresenta erro constante nas poses analisadas. É observado também que o tempo de busca e tempo total da missão diferem entre as poses.

Por fim, conclui-se que a análise de repetibilidade e reprodutibilidade foi positiva quando observados os dados da variável tempo de busca do marcador visual, uma vez que o sistema de medição atendeu ao teste; enquanto que para a variável tempo total da missão os resultados não foram de todo satisfatórios, indicando possíveis direções de melhoria no sistema.

6 CONFIABILIDADE DO SISTEMA

A fim de verificar-se a confiabilidade do sistema, foi realizado o levantamento de cada componente que constitui o manipulador robótico e o estudo detalhado das prováveis falhas. Para isso, foi utilizado o *FMECA* com o propósito de analisar cada sub-sistema, como pode ser visto nas tabelas da seção 6.1.

Na seção 6.2 foi desenvolvida a análise da árvore de falhas do sistema, método que permite através de um processo lógico dedutivo chegar-se às causas básicas de um problema de proporções maiores.

6.1 Análise dos modos e efeitos de falhas

Nas Tabelas 17, 18, 19, 20 e 21 estão representadas informações referentes ao estudo sistemático e estrutura das falhas potenciais para os sub-sistemas de potência, de aquisição, estrutural, de atuação e de processamento, respectivamente.

Tabela 17: *FMECA* do sub-sistema de potência

Análise do Tipo e Efeito de Falha								
Sub-sistema de potência								
Função(es)	Modo(s) de falha em potencial	Efeito(s) potencial(is) da falha	S	Causa(s) potenciais	O	D	C	Ação(es) recomendada(s)
Transmissão de dados	Incapacidade de transmissão de dados	Comprometimento dos dados transmitidos	6	Fissura dos fios	3	5	90	Manutenções preventivas
		Perda da capacidade de transmissão de dados	8	Desgaste causado pelo tempo	2	1	16	
Transmissão de energia	Incapacidade de transmissão de energia	Perda da capacidade de transmissão de energia	8	Derretimento por temperatura elevada	1	1	8	Manutenções preventivas
		Má qualidade da energia transmitida	6	Torção dos fios	3	5	90	
Alimentação do sistema	Incapacidade em fornecer energia	Não energização do sistema	9	Má conexão com fonte de tensão	3	6	162	Verificar conexão com fonte
				Desgaste nas soldas causado pelo tempo	4	7	252	

Fonte: Autoria própria.

Tabela 18: *FMECA* do sub-sistema aquisição

Análise do Tipo e Efeito de Falha								
Sub-sistema de aquisição								
Função(es)	Modo(s) de falha em potencial	Efeito(s) potencial(is) da falha	S	Causa(s) potenciais	O	D	C	Ação(es) recomendada(s)
Aquisição de dados visuais	Incapsidade de obter dados visuais	Perda da capacidade de obter dados visuais	8	Obstrução do campo de visão da câmera	2	1	16	Verificar condições do ambiente antes das missões
	Ruptura da estrutura de fixação da câmera			Baixa visibilidade do ambiente	2	5	80	
	Coleta de dados inconsistentes ou insuficientes			Calibração incorreta da câmera	3	1	24	
		Comprometimento dos dados	6	Set-up incorreto	2	7	84	Verificar cabeamento do sistema
				Falha no canal de comunicação	4	7	168	

Fonte: Autoria própria.

Tabela 19: *FMECA* do sub-sistema estrutural

Análise do Tipo e Efeito de Falha									
Sub-sistema estrutural									
Função(es)	Modo(s) de falha em potencial	Efeito(s) potencial(is) da falha	S	Causa(s) potenciais	O	D	C	Ação(es) recomendada(s)	
Ligar o frame ao perfil de alumínio	Incapacidade de sustentar o perfil	Ruptura do conector	8	Excesso de peso aplicado	2	1	16	Verificar a fabricação da peça	
				Desgaste pelo tempo	2	5	80		
	Compor estrutura mecânica	Trinca do conector	6	Set-up incorreto	4	6	144	Verificar as condições de peso aplicadas	
				Defeito de fabricação	5	6	180		
Danificação estrutural	Ruptura do perfil	8	Sobrecarga	2	4	64	Cuidado no manuseamento dos perfis		
	Folga entre conexões	6	Colisão	4	2	48	Definir área de trabalho		

Fonte: Autoria própria.

Tabela 20: *FMECA* do sub-sistema de atuação

Análise do Tipo e Efeito de Falha									
Sub-sistema de atuação									
Função(es)	Modo(s) de falha em potencial	Efeito(s) potencial(is) da falha	S	Causa(s) potenciais	O	D	C	Ação(es) recomendada(s)	
Movimentação do manipulador	Rotação inconsistente com o desejado	Perda de desempenho (velocidade; controle)	6	Emperramento parcial	4	6	144	Inspecções periódicas	
Fornecer o torque mínimo de sustentação	Curto circuito	Parada do motor	8	Cabeamento incorreto	2	5	80	Medir corrente nos terminais	
	Perda de comunicação	Perda de desempenho	6	Falha na alimentação	4	5	120		
	Tensão insuficiente								

Fonte: Autoria própria.

Tabela 21: *FMECA* do sub-sistema de processamento

Análise do Tipo e Efeito de Falha								
Sub-sistema de processamento								
Função(es)	Modo(s) de falha em potencial	Efeito(s) potencial(is) da falha	S	Causa(s) potenciais	O	D	C	Ação(es) recomendada(s)
Processamento dos dados	Não processamento dos dados do sistema principal	Não funcionamento	9	Queima/ danificação de componentes	3	8	216	Manutenção preditiva
				Falha no sistema operacional	2	8	144	Executar teste de verificação do software
				Falha na porta de comunicação	2	2	36	Executar teste de entrada e saída de dados com <i>NUC</i>
				Problemas de contato elétrico	2	3	54	Inspecções periódicas em bancada

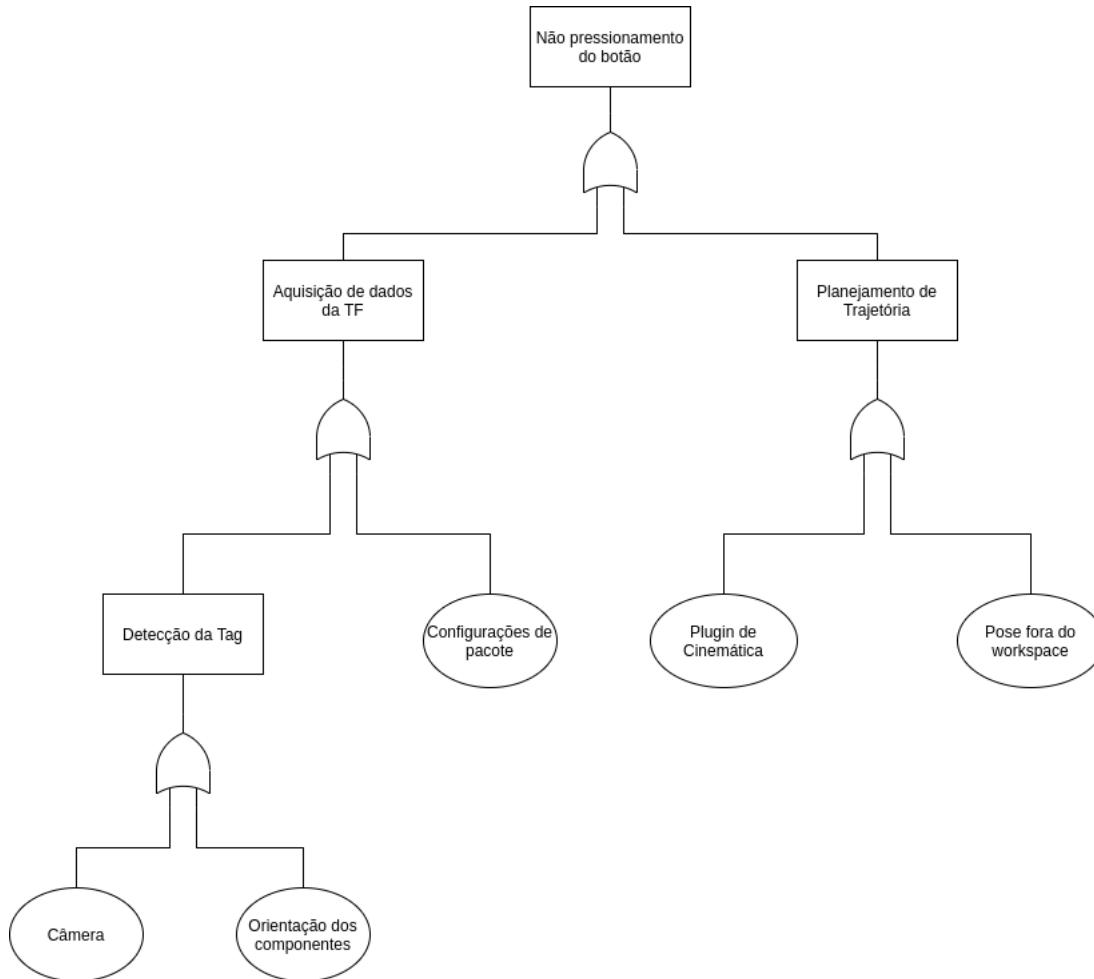
Fonte: Autoria própria.

6.2 Análise da árvore de falhas

Na Figura 31 encontra-se a árvore de falhas do sistema do manipulador JeRoTIMON. Observa-se que caso o manipulador não consiga realizar a tarefa de pressionar o botão, duas causas principais devem ser investigadas: a conexão entre as TF's, realizada a partir do pacote *Bir Marker Localization*, ou o planejamento de trajetória, realizado pelo *Moveit*.

Para a primeira causa, devem ser verificadas as configurações do pacote e da câmera, bem como a orientação dos componentes envolvidos no processo de detecção. Já para solucionar os problemas correspondentes ao planejamento de trajetória, deve-se observar se a pose desejada está dentro do *workspace* do manipulador e se o *plugin* de cinemática inversa utilizado está sendo capaz de realizar as conversões necessárias para a pose desejada.

Figura 31: Árvore de falhas do sistema.



Fonte: Autoria própria.

7 GESTÃO DO CONHECIMENTO

Neste capítulo estão descritas as lições aprendidas durante o processo de desenvolvimento do protótipo que foram criadas a partir da comparação entre o que era esperado e o que realmente aconteceu em cada etapa do projeto. Esta sub-seção engloba desde a fase inicial do planejamento até a construção do modelo real.

Além das lições aprendidas, as seções 7.2 e 7.3 trazem os guias de uso para a simulação e o modelo real, respectivamente, com o propósito de auxiliar o usuário na replicação dos experimentos realizados neste relatório.

7.1 Lições aprendidas

A Tabela 22 mostra como foi estruturada cada lição aprendida abordando os seguintes aspectos: Tema, Fase, Impacto, O que ocorreu?, Como resolveu?, Resultados e Recomendações para os próximos projetos. O objetivo deste estudo é a correção dos impactos negativos para os projetos subsequentes.

Tabela 22: Lições aprendidas

LIÇÕES APRENDIDAS						
Tema	Fase	Impacto	O que ocorreu?	Como resolveu?	Resultados	Recomendações para os próximos projetos
Gestão	Planejamento	Negativo	Ausência de uma metodologia de trabalho	Reunião com foco em definir metodologia de projeto	Evolução na comunicação dos membros	Antes de começar o projeto realizar reunião para definição de metodologia
Gestão	Planejamento	Negativo	Ausência de uma ferramenta para gestão de projeto	Escolha de uma ferramenta gratuita e de fácil aplicação	Melhoria na organização das atividades	Definição prévia de uma ferramenta de gestão de projeto
Gestão	Execução	Negativo	Montagens e desmontagens do manipulador	Planejamento prévio das atividades	Otimização do tempo	Cronograma definido as atividades a serem realizadas
Tecnológico	Execução	Negativo	Falha no dimensionamento das peças	Consultoria sobre materiais aplicados na confecção	Obtenção de peças com maior resistência mecânica	Pesquisa e consultoria prévia antes da modelagem das peças
Tecnológico	Execução	Negativo	Dificuldade em planejamento de trajetória para determinadas poses do robô	Utilização do plugin Track-IK	Maior eficiência de planejamento	Pesquisa e consultoria prévia do pacote mais adequado para o projeto
Tecnológico	Execução	Negativo	Motor dynamixel (H54-200-S500-R PRO) parou de funcionar	Realizado a Análise 8D para fazer investigação do ocorrido	Compreensão de que há procedimentos a serem feitos antes de utilizar um produto	Realizar leitura meticolosa do manual do produto para saber quais são os procedimentos

Fonte: Autoria própria.

7.2 Guia de uso para simulação

Para replicar a simulação do manipulador robótico Timon-HM é necessário seguir os passos descritos nesta seção. Recomenda-se a utilização do Ubuntu 18.04 LTS e o *ROS Melodic Morenia*.

Antes de inserir o pacote do manipulador no *workspace* é fundamental que sejam instalados os pacotes requeridos para este. Primeiramente, no terminal, segue-se os comandos listados:

- Instalar MoveIt:

```
$ sudo apt-get install ros-melodic-moveit
```

- Instalar pacote de ferramentas visuais do MoveIt:

```
$ sudo apt-get install ros-melodic-moveit-visual-tools
```

- Instalar TRAC-IK para resolução da cinemática:

```
$ sudo apt-get install ros-melodic-trac-ik-kinematics-plugin
```

- Instalar pacote de controle no *Gazebo ROS*:

```
$ sudo apt-get install ros-melodic-gazebo-ros-control
```

- Instalar pacote do controlador do *ROS*:

```
$ sudo apt-get install ros-melodic-controller-*
```

- Instalar pacote controlador de posição do *ROS*:

```
$ sudo apt-get install ros-melodic-position-controller
```

- Instalar pacote controlador de esforço do *ROS*:

```
$ sudo apt-get install ros-melodic-effort-controller
```

- Instalar pacote de juntas do *ROS*:

```
$ sudo apt install ros-melodic-joint-*
```

- Instalar pacote de controle do *ROS*:

```
$ sudo apt install ros-melodic-ros-control
```

Antes de instalar o pacote *bir_marker_localization* é necessária a instalação do *OpenCV* versão 3.3.1, este possui o guia de instalação próprio disponível em:

<https://www.learnopencv.com/install-opencv3-on-ubuntu/>.

Após a instalação do *OpenCV*, para clonar o repositório do *bir_marker_localization* segue-as as seguintes etapas, no terminal:

- Criar um *workspace* para adicionar dentro deste os pacotes que serão usados na simulação.

```
$ mkdir nomedoworkspace_ws
```

- Entrar no *workspace*:

```
$ cd nomedoworkspace_ws
```

- Criar a pasta *source*¹:

```
$ mkdir src
```

- Entrar no *source*:

```
$ cd src
```

- Clonar o repositório *Bir Marker Localization* para *workspace* (Verificar qual a *branch* estável):

```
$ git clone https://github.com/Brazilian-Institute-of-Robotics/bir_marker_localization.git
```

- Clonar o pacote do manipulador para dentro da pasta src:

```
$ git clone -b feature/simulation https://github.com/Brazilian-Institute-of-Robotics/timon_hm_manipulator.git
```

- Clonar o pacote do *Open Manipulator* para dentro da pasta src:

```
$ git clone https://github.com/ROBOTIS-GIT/open_manipulator_msgs.git
```

- Retornar para a raiz do *workspace*:

```
$ cd ..
```

- Compilar o *workspace*:

```
$ catkin_make
```

- Ativar o ambiente virtual do *workspace*:

```
$ source devel/setup.bash
```

¹ Pasta onde contém os arquivos fonte.

Após realizar os procedimentos citados o *workspace* estará configurado para executar a simulação, com os seguintes comandos (cada comando terá que ser inserido em uma aba do terminal):

- Iniciar a simulação no *Gazebo*:

```
$ roslaunch manipulator_gazebo gazebo.launch
```

- Iniciar pacote MoveIt do Timon-HM:

```
$ roslaunch manipulator_gazebo moveit_demo.launch
```

- Iniciar o *bir_marker_localization*:

```
$ roslaunch timon_demo bir_marker_localization.launch
```

- Iniciar o algoritmo de busca do marcador visual e para acionar o painel elétrico:

```
$ roslaunch timon_demo push_button_simulation.launch
```

7.3 Guia de uso para o modelo real

Para replicar o modelo real do manipulador a versão do Ubuntu, do *ROS* e os pacotes necessários são os mesmos descritos na seção 7.2. Após a instalação dos pacotes, a seguir estão descritas as etapas subsequentes:

- Criar *workspace* e a pasta *source*:

```
$ mkdir -p catkin_ws/src
```

- Entrar no *workspace* e no *source*:

```
$ cd catkin_ws/src
```

- Clonar para dentro da pasta *source* o repositório do manipulador:

```
$ git clone https://github.com/Brazilian-Institute-of-Robotics/timon_hm_manipulator.git
```

- Clonar para dentro do *source* o repositório do *Bir Marker Localization*:

```
$ git clone -b final_settings https://github.com/Brazilian-Institute-of-Robotics/bir_marker_localization.git
```

- Clonar para dentro do *source* o repositório do *Open Manipulator*:

```
$ git clone https://github.com/ROBOTIS-GIT/  
open_manipulator_msgs.git
```

- Clonar para dentro do *source* o repositório do *Dynamixel workbench*:

```
$ git clone https://github.com/ROBOTIS-GIT/dynamixel-  
workbench.git
```

- Clonar para dentro do *source* o repositório da câmera *Teledyne*:

```
$ git clone -b refactor_code https://github.com/Brazilian  
-Institute-of-Robotics/def_cam_teledyne_nano.git
```

- Retornar para a raiz do *workspace*:

```
$ cd ..
```

- Compilar o *workspace*:

```
$ catkin_make
```

- Ativar o ambiente virtual do *workspace*:

```
$ source devel/setup.bash
```

Para executar a aplicação é necessário realizar os seguintes comandos (cada comando terá que ser inserido em uma aba do terminal):

- Iniciar os controladores do manipulador:

```
$ roslaunch timon_arm_controller dxl_controllers.launch
```

```
$ roslaunch timon_arm_controller moveit.launch
```

```
$ roslaunch timon_arm_controller dxl_moveit_bridge.launch
```

- Iniciar a câmera:

```
$ roslaunch def_cam_teledyne_nano camera_example.launch
```

- Iniciar o *Bir Maker Localization*:

```
$ roslaunch timon_demo bir_marker_localization.launch
```

- Executar a missão:

```
$ roslaunch timon_demo push_button_real.launch
```

8 CONCLUSÃO

O presente relatório descreveu a idealização, simulação e construção do JeRoTIMON, um manipulador robótico com 5 *DoF*, integrado ao *ROS*, capaz de acionar um painel elétrico a partir da localização de um marcador visual *ArUco*. A estrutura física do robô foi concebida de acordo com os modelos *URDF* projetados.

Para possibilitar o uso da ferramenta *Moveit*, arquivos de configuração foram gerados e sua comunicação com o modelo simulado no *Gazebo* foi estabelecida. Para resolver as equações de cinemática inversa optou-se pelo plugin TRAC-IK e para o planejamento de trajetória foi utilizada a biblioteca *OMPL*.

Foi desenvolvido em linguagem C++ um pacote capaz de comunicar o sistema de escaneamento e o sistema de planejamento/execução. A utilização da câmera Teledyne Genie Nano C2590 possibilitou a identificação da *tag ArUco* com precisão, tornando o sistema capaz de localizar o painel elétrico e planejar uma trajetória que leve o *endeffector* do manipulador até o alvo estabelecido.

Foram realizados 80 testes considerando diferentes algoritmos, posições e orientações para o painel elétrico. Os resultados alcançados mostram que JeRoTIMON foi capaz de realizar a tarefa em 91.25% dos casos, com tempo médio de busca de 23.95 segundos e o tempo médio de execução de 86.06 segundos, resultados estes considerados satisfatórios para o prosseguimento do projeto.

Para sua próxima aplicação, o manipulador será instalado em um robô móvel *Warthog* que estará equipado com sensores de localização, mapeamento e detecção de obstáculos. De forma autônoma, este conjunto irá localizar e desarmar uma bomba hipotética instalada em ambiente aberto.

REFERÊNCIAS

- 16MM C Series VIS-NIR Fixed Focal Length Lens. s.d. <<https://bit.ly/35zzQMq>>. Acessado: 26-10-2020. Citado na página 36.
- BEESON, P.; AMES, B. Trac-ik: An open-source library for improved solving of generic inverse kinematics. In: IEEE. *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*. [S.l.], 2015. p. 928–935. Citado na página 29.
- BIR Marker Localization. 2020. <https://github.com/Brazilian-Institute-of-Robotics/bir_marker_localization/>. Accessed: 2020-04-24. Citado na página 27.
- ERTHAL, J. L. Estudo de métodos para a solução da cinemática inversa de robôs industriais para implementação computacional. *UFSC, Programa de Pós-Graduação*, 10 1992. Citado na página 13.
- HERNANDEZ-MENDEZ, S. et al. Design and implementation of a robotic arm using ros and moveit! In: IEEE. *2017 IEEE International Autumn Meeting on Power, Electronics and Computing (ROPEC)*. [S.l.], 2017. p. 1–6. Citado na página 18.
- HERNÁNDEZ-ORDÓÑEZ, M. et al. An education application for teaching robot arm manipulator concepts using augmented reality. *Mobile Information Systems*, Hindawi, v. 2018, 2018. Citado na página 18.
- INDUSTRIAL Robots: Robot Investment Reaches Record 16.5 billion USD. 2019. <<https://ifr.org/ifr-press-releases/news/robot-investment-reaches-record-16.5-billion-usd/>>. Acesso em: 15-04-2020. Citado na página 13.
- ISO-8373. *Robots and robotic devices — Vocabulary*. Geneva, CH, 2012. Citado na página 17.
- JAVEED, A.; PRAKASH, V. G.; KULKARNI, S. P. Autonomous service robot. In: *Proceedings of the Advances in Robotics 2019*. [S.l.: s.n.], 2019. p. 1–6. Citado na página 19.
- KHAN, K. A.; KONDA, R. R.; RYU, J.-C. Ros-based control for a robot manipulator with a demonstration of the ball-on-plate task. *Advances in robotics research*, v. 2, n. 2, p. 113, 2018. Citado na página 19.
- LEITE, A. C. *Controle Híbrido de Força e Visão de um Manipulador Robótico Sobre Superfícies Desconhecidas*. Tese (Doutorado) — Tese de Mestrado, Departamento de Engenharia Elétrica, 2005. Citado na página 14.
- PIMENTA, T. T. Controle de manipuladores robóticos. *PUC-Rio de Janeiro, Departamento de Engenharia de Controle e automação*, 12 2009. Citado na página 13.
- ROBOTIS. s.d. <<http://www.robotis.us/>>. Acessado: 19-10-2020. Citado 2 vezes nas páginas 38 e 81.
- ROMANO, V. F. *Robótica Industrial: Aplicação na indústria de manufatura e processos*. São Paulo: Edgar Blucher, 2002. v. 256. Citado na página 13.

SANTOS, V. M. Robótica industrial. *Universidade de Aveiro-Departamento de Engenharia Mecânica*, 2004. Citado na página 18.

SPONG, M. W.; HUTCHINSON, S.; VIDYASAGAR, M. Robot modeling and control, john wiley & sons. Inc., ISBN-100-471-649, 2005. Citado na página 13.

SUCAN, I. A.; MOLL, M.; KAVRAKI, L. E. The open motion planning library. *IEEE Robotics & Automation Magazine*, IEEE, v. 19, n. 4, p. 72–82, 2012. Citado na página 29.

TELEDYNE Dalsa. s.d. <<https://www.teledynedalsa.com/en/products/imaging/cameras/genie-nano-1gige/>>. Acessado: 19-10-2020. Citado na página 36.

U2D2. s.d. <<https://emanual.robotis.com/docs/en/parts/interface/u2d2/>>. Acessado: 26-10-2020. Citado 2 vezes nas páginas 38 e 39.

UWE-12/10-Q12PB-C. s.d. <<https://bit.ly/31i2SyP>>. Acessado: 19-10-2020. Citado na página 38.

ZHANG, S.; LIN, Z.; WU, G. Motion planning of a 5-dof anthropomorphic robotic arm under ros environment. In: SPRINGER. *IFToMM International Conference on Mechanisms, Transmissions and Applications*. [S.l.], 2019. p. 409–418. Citado na página 19.

APÊNDICE A

Árvores de TF desconectadas



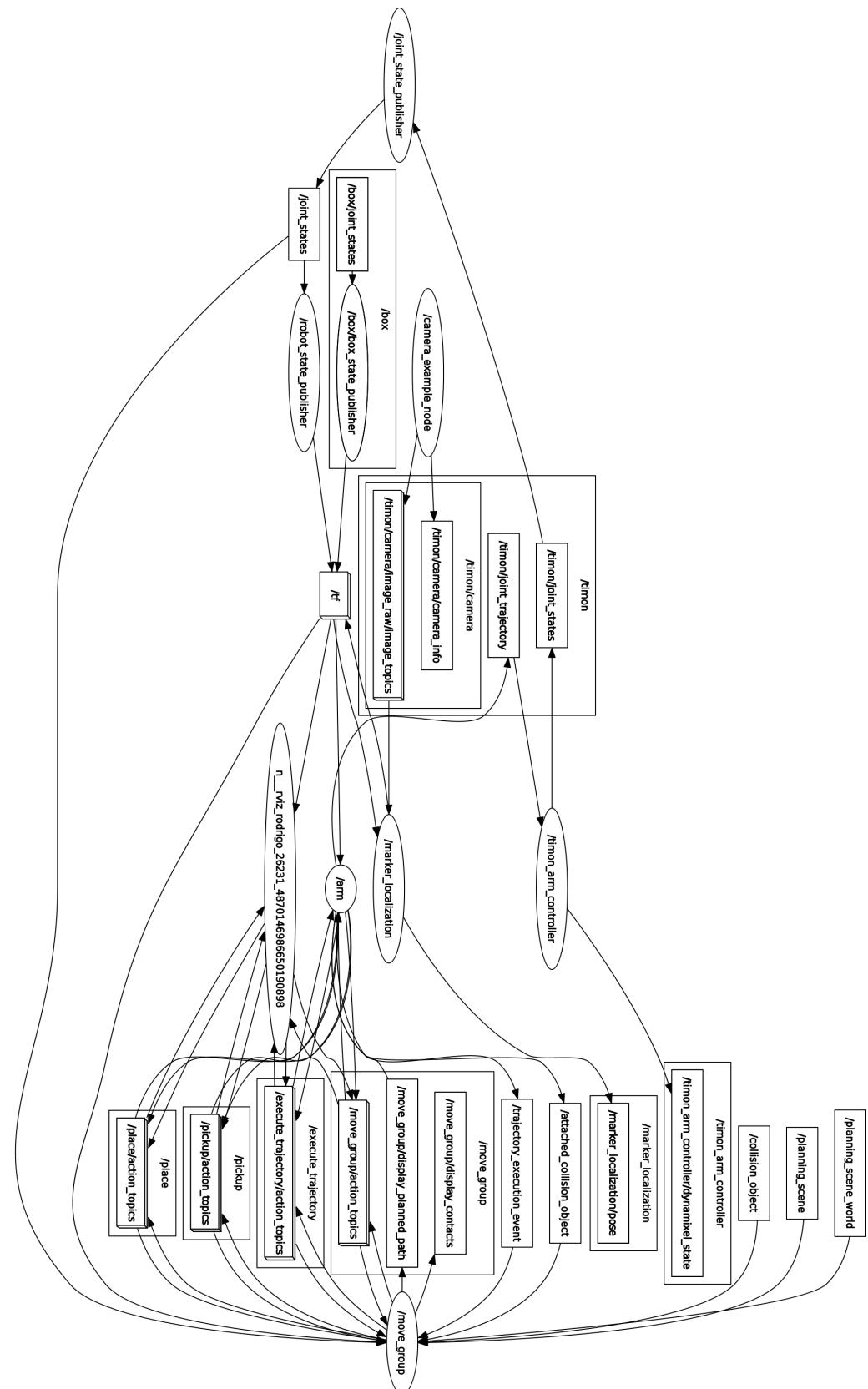
APÊNDICE B

Árvores de TF conectadas



APÊNDICE C

Diagrama de nós do sistema



APÊNDICE D

Propriedades de Massa do JeRoTIMON

Figura 32: Coordenadas.

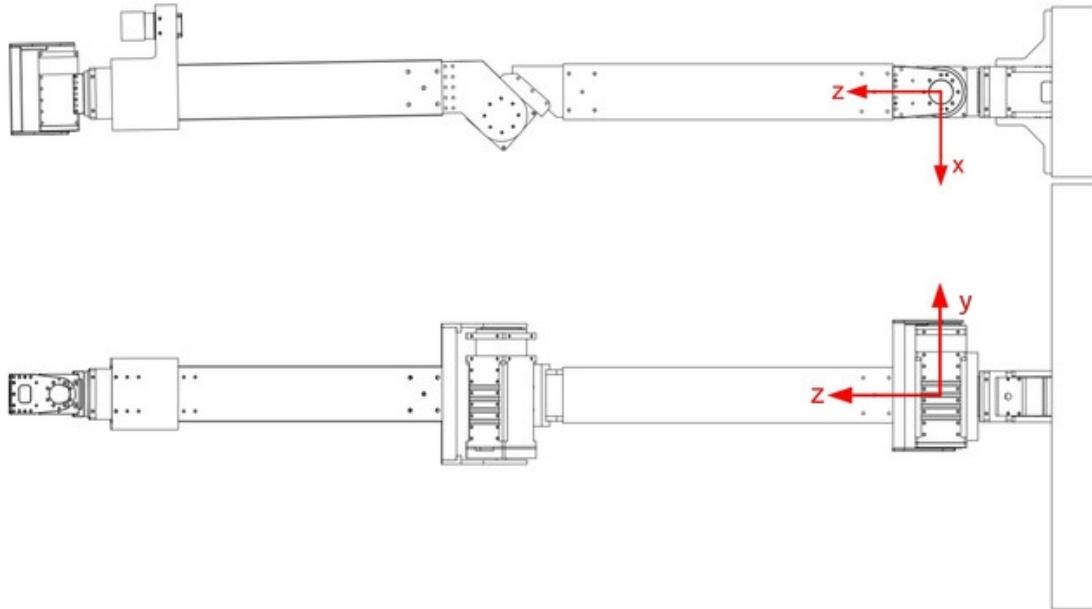
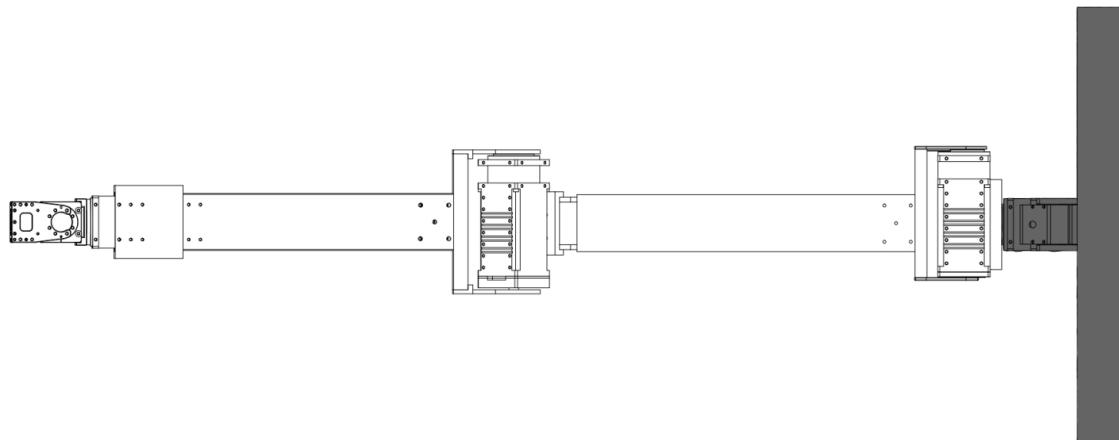


Figura 33: Link 0.



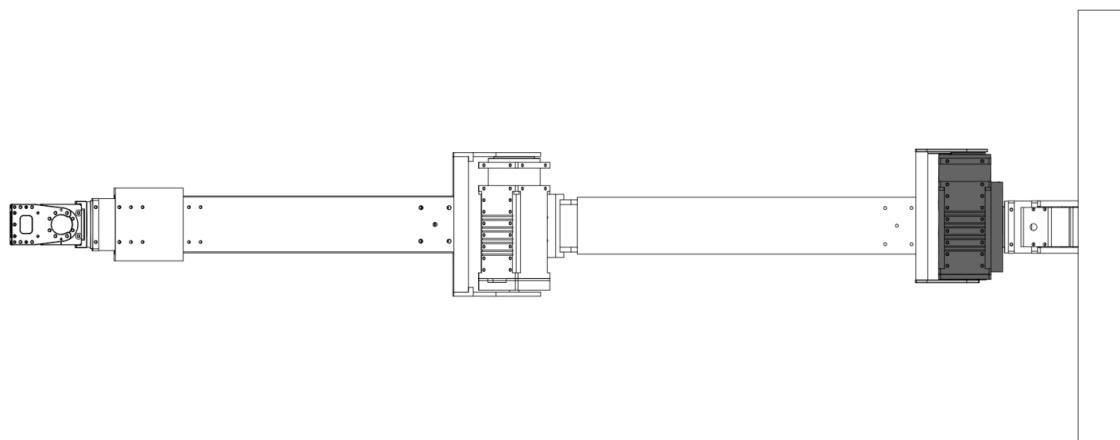
- **Massa:** 3.72384654 kg
- **Volume:** 0.00413037 m^3
- **Área:** 0.28622123 m^2
- **Centro de Massa:**

- X: -0.00000147 m
- Y: 0.00000000 m
- Z: 0.03504069 m

- **Momento de inércia:** kg m^2

- **L_{xx}:** 0.01072644 **L_{xy}:**-9.465e-9 **L_{xz}:**-3.247e-8
- **L_{yx}:** -9.465e-9 **L_{yy}:** 0.04865651 **L_{yz}:** 6.830e-13
- **L_{zx}:** -3.247e-8 **L_{zy}:** 6.830e-13 **L_{zz}:** 0.05388213

Figura 34: Link 1.



- **Massa:** 1.03781084 kg

- **Volume:** 0.00040169 m^3

- **Área:** 0.05853078 m^2

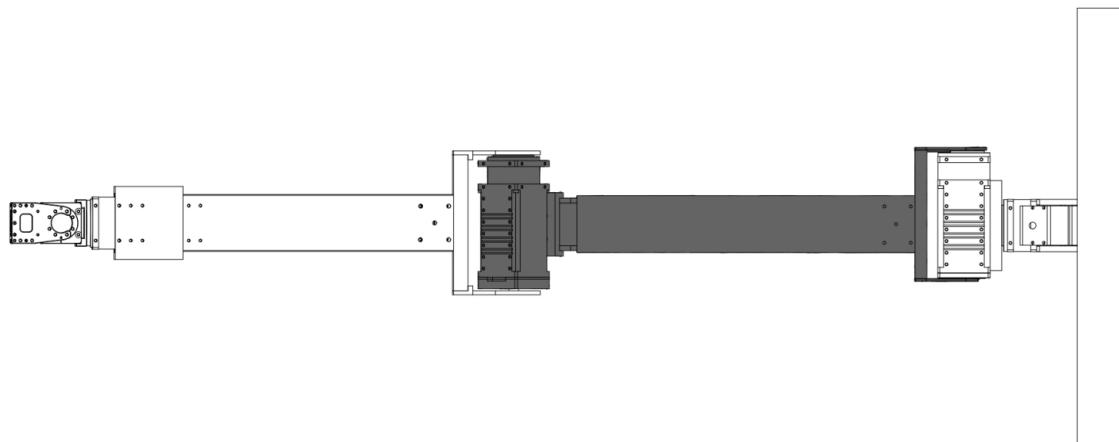
- **Centro de Massa:**

- X: 0.00814457 m
- Y: 4.45597047e - 8 m
- Z: 0.16022275 m

- **Momento de inércia:** kg m^2

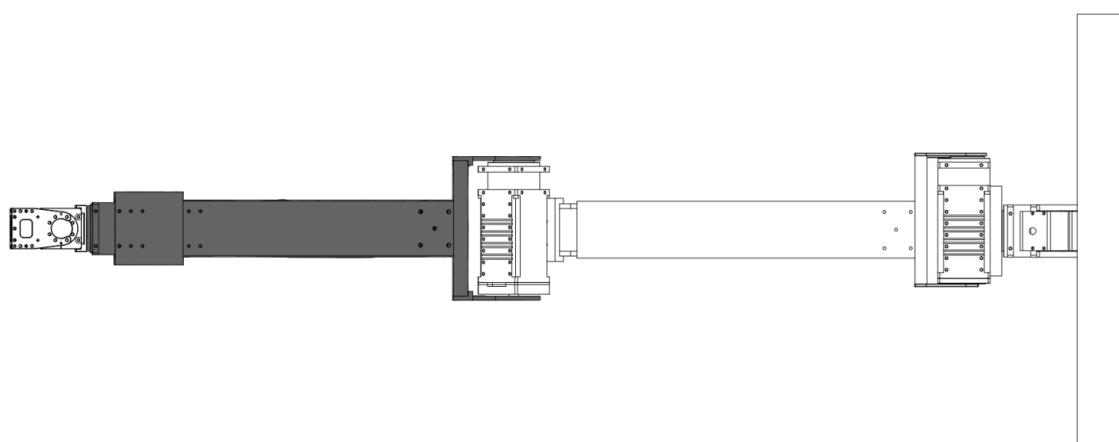
- **L_{xx}:** 0.0005687 **L_{xy}:** 2.602e-10 **L_{xz}:** -0.00004027
- **L_{yx}:** 2.602e-10 **L_{yy}:** 0.00166759 **L_{yz}:** -2.222e-10
- **L_{zx}:** -0.00004027 **L_{zy}:** -2.222e-10 **L_{zz}:** 0.00155695

Figura 35: Link 2.



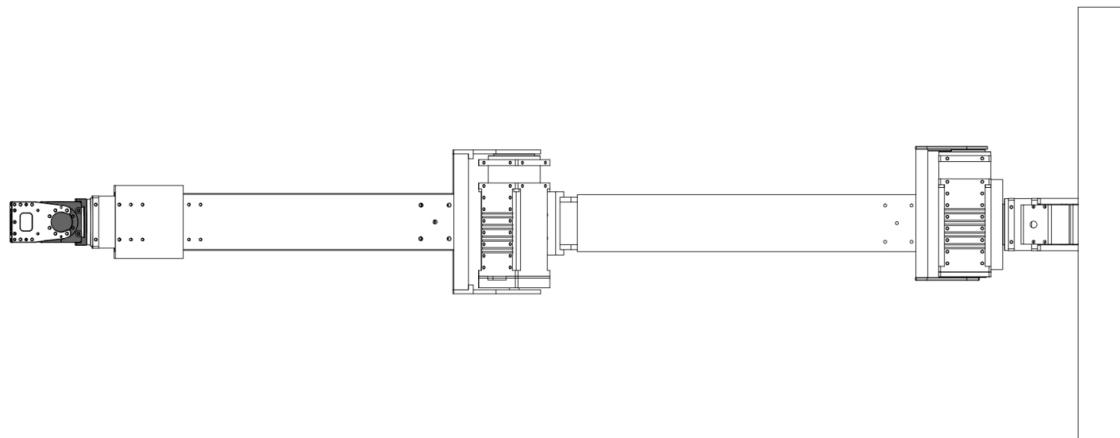
- **Massa:** 2.05026959 kg
- **Volume:** 0.00077667 m^3
- **Área:** 0.29790369 m^2
- **Centro de Massa:**
 - X: 0.00294129 m
 - Y: -0.01058166 m
 - Z: 0.48960322 m
- **Momento de inércia:** kg m^2
 - **L_{xx}:** 0.06174738 **L_{xy}:** -0.00003476 **L_{xz}:** 0.00098626
 - **L_{yx}:** -0.00003476 **L_{yy}:** 0.06319525 **L_{yz}:** 0.00311701
 - **L_{zx}:** 0.00098626 **L_{zy}:** 0.00311701 **L_{zz}:** 0.0034029

Figura 36: Link 3.



- **Massa:** 1.83580505 kg
- **Volume:** 0.00081683 m^3
- **Área:** 0.32055793 m^2
- **Centro de Massa:**
 - X: 0.00257384 m
 - Y: 0.00207951 m
 - Z: 0.91287054 m
- **Momento de inércia:** kg m^2
 - **L_{xx}:** 0.03439469 **L_{xy}:** -0.00000261 **L_{xz}:** -0.00001194
 - **L_{yx}:** -0.00000261 **L_{yy}:** 0.03489631 **L_{yz}:** -0.0001288
 - **L_{zx}:** -0.00001194 **L_{zy}:** -0.0001288 **L_{zz}:** 0.0023687

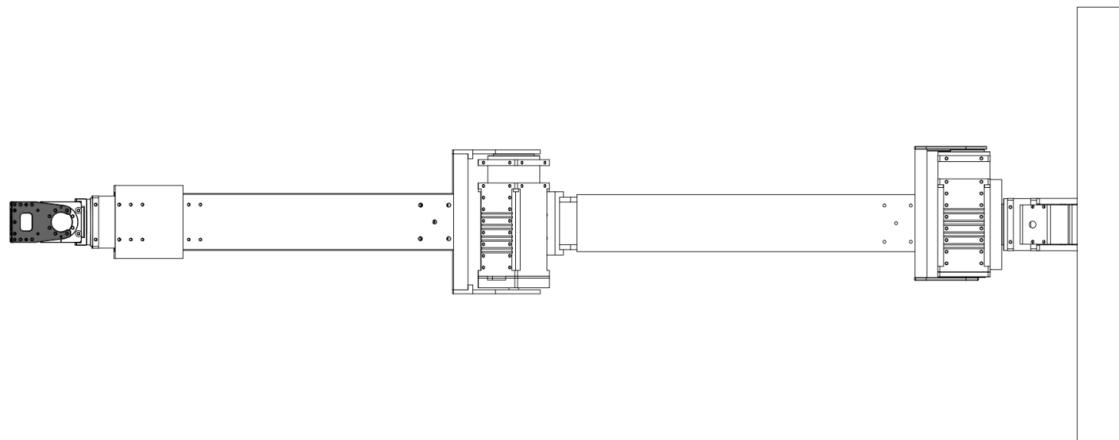
Figura 37: Link 4.



- **Massa:** 0.39425655 kg
- **Volume:** 0.00014602 m^3
- **Área:** 0.03018022 m^2
- **Centro de Massa:**
 - X: 0.00257156 m
 - Y: 0.00133073 m
 - Z: 1.09673426 m
- **Momento de inércia:** kg m^2

- **Lxx:** 0.00028795 **Lxy:** 2.010e-10 **Lxz:** -8.166e-13
- **Lyx:** 2.010e-10 **Lyy:** 0.00011981 **Lyz:** -0.00000446
- **Lzx:** -8.166e-13 **Lzy:** -0.00000446 **Lzz:** 0.0002842

Figura 38: Link 5.



- **Massa:** 0.08694786 kg
- **Volume:** 0.0000322 m^3
- **Área:** 0.02435944 m^2
- **Centro de Massa:**
 - X: 0.00257155 m
 - Y: 0.00684212 m
 - Z: 1.13578562 m
- **Momento de inércia:** kg m^2
 - **Lxx:** 0.00012963 **Lxy:** -5.235e-13 **Lxz:** -9.996e-14
 - **Lyx:** -5.235e-13 **Lyy:** 0.00003703 **Lyz:** 0.00000247
 - **Lzx:** -9.996e-14 **Lzy:** 0.00000247 **Lzz:** 0.00012613

APÊNDICE E

Lista de suportes.

Tabela 23: Tabela de suportes

Imagen	Junta	Fornecedor	Tipo	Modelo	Quantidade	Medidas (mm)
	Id_1	CCRoSA	Cantoneira	Original	2	59 x 59 x 40.5
	Id_2	CCRoSA	Frame Perpendicular	Original	1	92.6 x 51.1 x 12
	Id_2	ROBOTIS	Rolamento	FRP54-I110K	1	54 x 54 x 5
	Id_3	ROBOTIS/CCRoSA	Frame de Rotação	FRP54-H221K/Original	1	105.5 x 138 x 54.3
	Id_3	CCRoSA	Fixador	Original	1	95 x 54 x 54
	Id_3	CCRoSA	Fixador	Original	1	60 x 65.2 x 17
	Id_3	CCRoSA	Calço	Original	1	54 x 54 x 2
	Id_3	CCRoSA	Rolamento	Original	1	54 x 54 x 10
	Id_4	CCRoSA	Frame de Rotação	Original	1	85.03 x 89.46 x 148.2
	Id_5	ROBOTIS	Frame Perpendicular	FRP42-A110K	1	48 x 42 x 11.5
	Id_5	ROBOTIS	Rolamento	FRP42-I110K	1	42 x 42 x 5
	Id_6	ROBOTIS	Frame de Rotação	FRP42-H121K	1	96 x 42 x 56.6

Fonte: Adaptado de ([ROBOTIS, s.d.](#))

APÊNDICE F

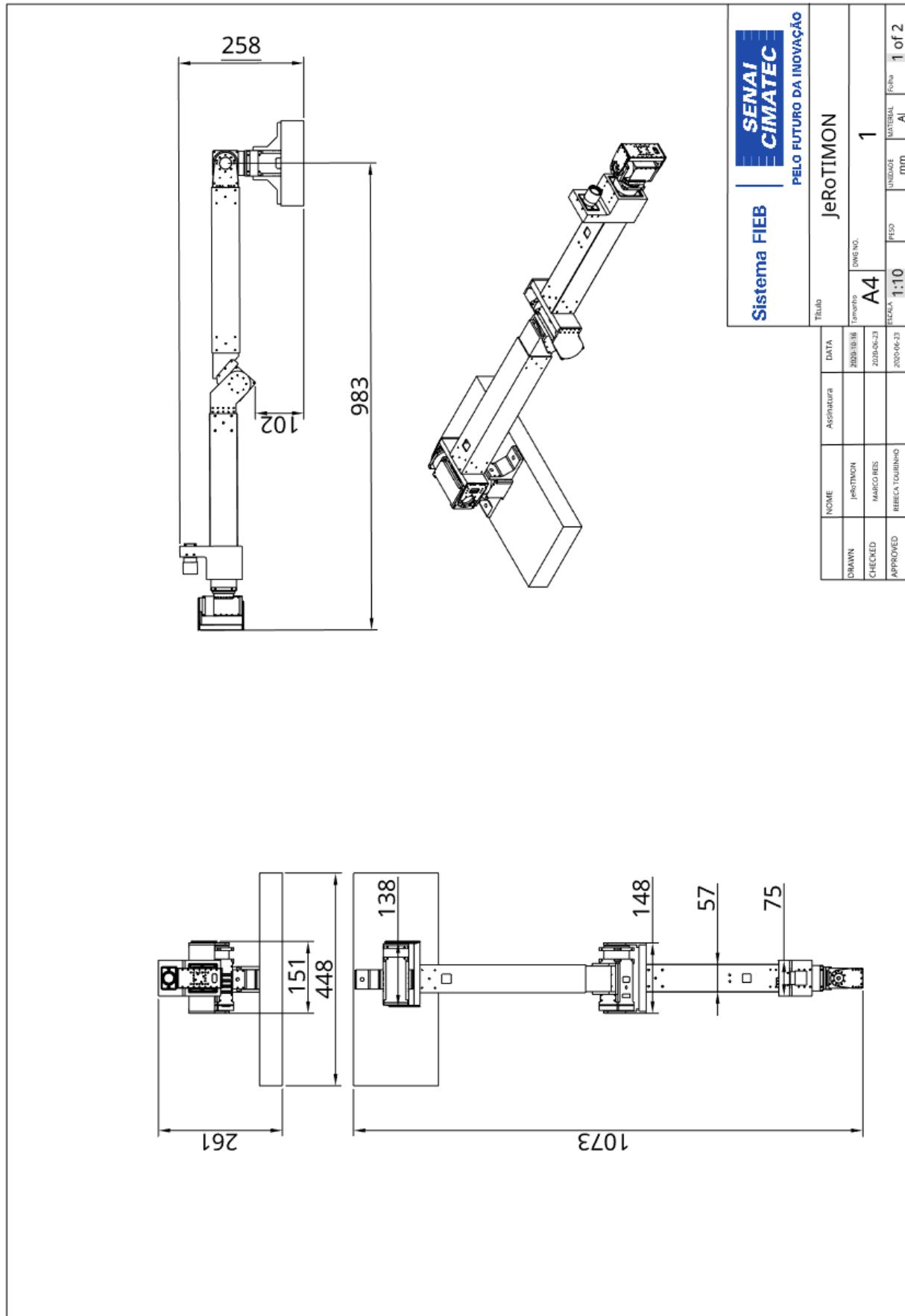
Projeto mecânico

O projeto mecânico do manipulador JeRoTIMON foi desenvolvido em duas etapas: Modelagem 3D e Cálculos Estáticos. A etapa de modelagem levou em consideração, além da análise estrutural e requisitos do cliente, as restrições físicas para que o manipulador pudesse ser anexado no UGV *Warthog*. Por fim, foi realizado os cálculos de esforços estáticos em que analisou-se os esforços sofridos pelo manipulador, na situação em que exige uma maior carga dos atuadores, e com isso garanta a funcionalidade do mesmo. A seguir, será visto o desenho mecânico final do manipulador e como foi realizado a análise estática.

F.1 Desenho mecânico

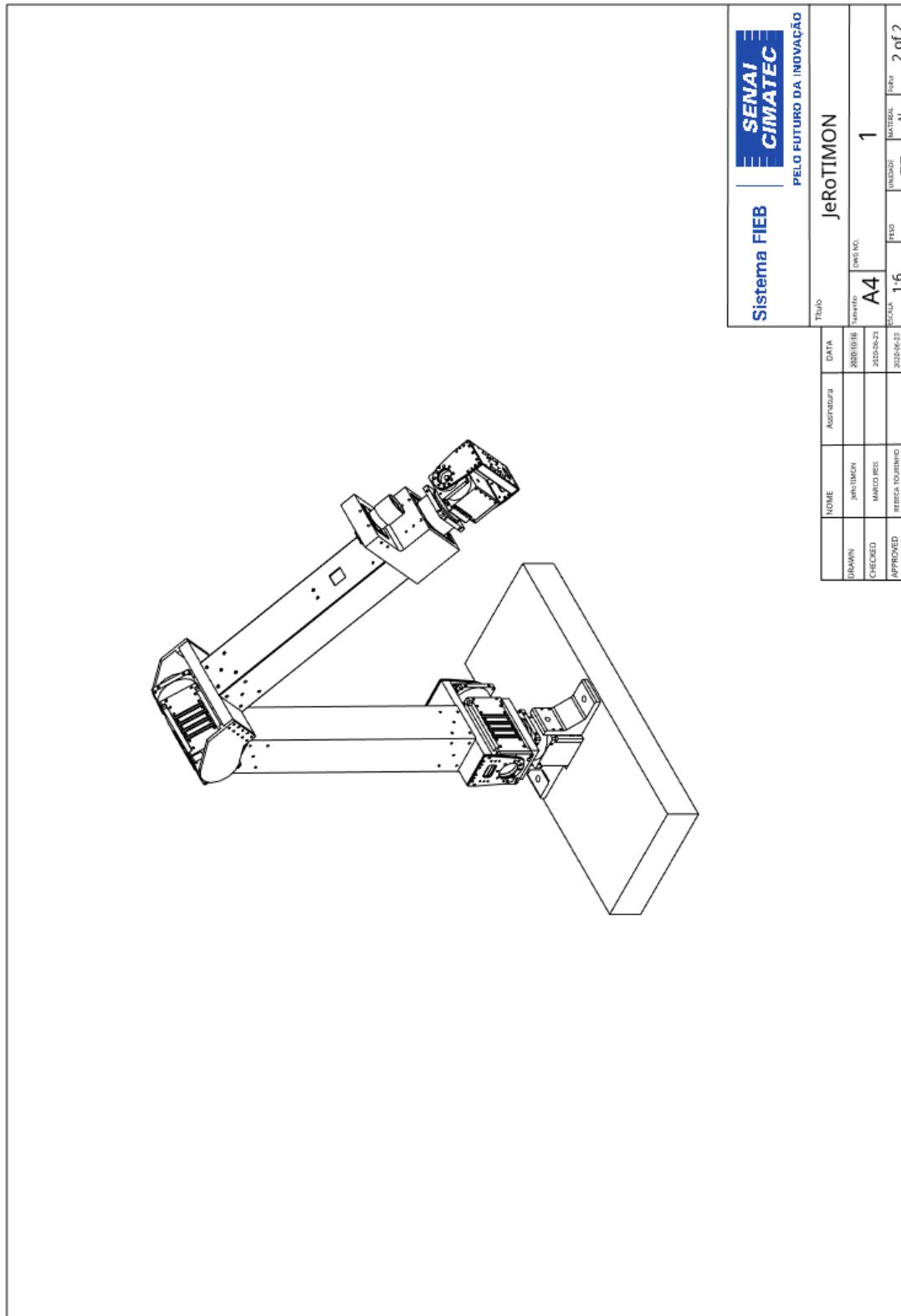
O desenho do manipulador foi todo elaborado utilizando o *software* “*OnShape*”. A Figura 39 mostra o desenho técnico do manipulador com as vistas: frontal, lateral esquerda, superior e isométrica, contendo todos os detalhes necessários para execução do projeto. A figura 40 mostra a vista isométrica expandida do manipulador em sua posição *home*.

Figura 39: Desenho técnico do JeRoTIMON, vistas: frontal, lateral esquerda e superior.



Fonte: Autoria própria.

Figura 40: Desenho técnico do JeRoTIMON, vista isométrica.

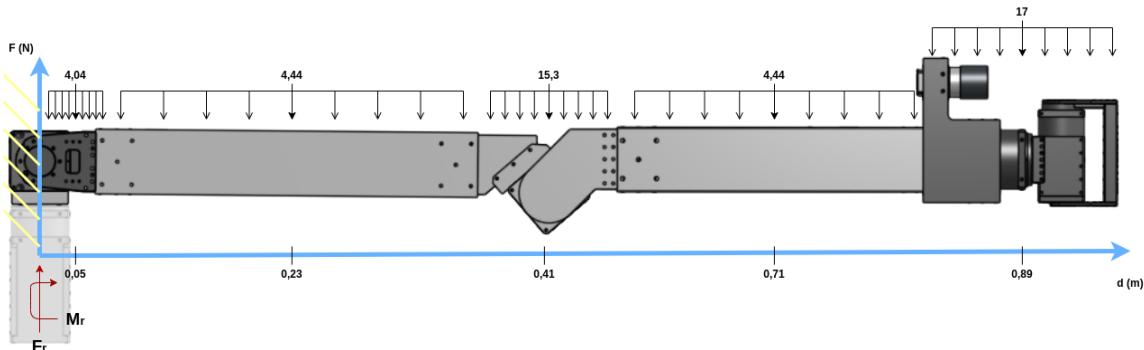


Fonte: Autoria própria.

F.2 Analise estática

Para calcular a carga máxima suportada pelo manipulador foi realizada uma análise estática. Para isso, o mesmo foi posto na situação em que exige um maior esforço das juntas 0 e 1, conforme visto na Figura 41.

Figura 41: Analise estática - condição de maior esforço exigido.



fonte: Autoria Própria

Para analisar os esforços sofridos pela junta 1 (motor 2), admiti-se a extremidade esquerda como engastada e em sua situação de equilíbrio estático, como visto na Figura 41. A força resultante no eixo y é calculada da seguinte forma:

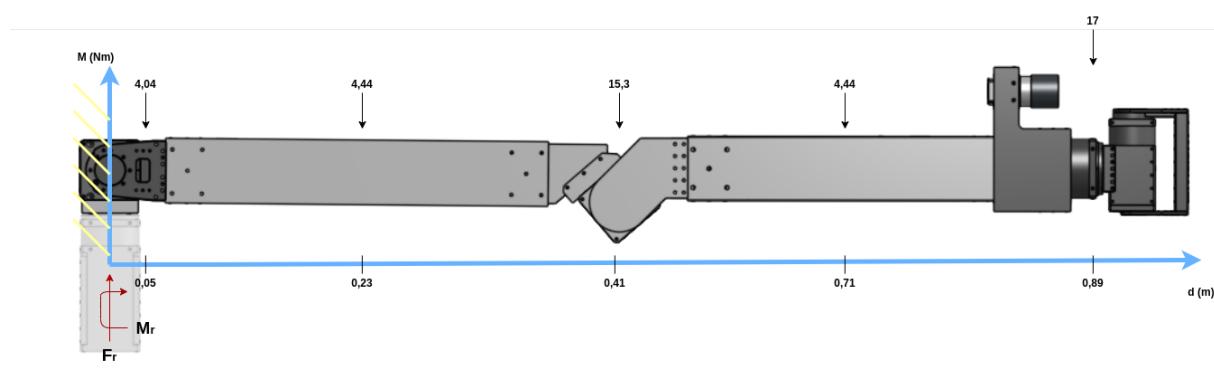
$$\sum F_{ry} = 0$$

$$F_{ry} - 4,04 - 4,44 - 15,3 - 4,44 - 17 = 0$$

$$F_{ry} = 45,22 \text{ [N]}$$

Para calcular o torque sofrido pela junta, as cargas distribuídas de cada componente do robô foram colocadas no seu centróide, conforme visto na Figura 42.

Figura 42: Analise estática - condição de maior esforço exigido (cargas no centróide).



fonte: Autoria Própria

Em seguida, foi calculado o momento torsor gerado nos motores 1 e 2 por cada carga pontual da seguinte maneira:

$$\sum M_r = 0$$

$$M_r - (4,04 \times 0,05) - (4,44 \times 0,23) - (15,3 \times 0,41) - (4,44 \times 0,71) - (17 \times 0,89) = 0$$

$$M_r = \mathbf{25,8} \text{ [N x m]}$$

Por fim, foi calculada a carga máxima suportada pelo manipulador, também conhecida por *payload*. Para este cálculo, foi necessário obter e comparar as informações sobre o torque máximo fornecido pelo motor que compõe a junta 1 (PH54-200-S500-R) que é de 44,7 N x m, segundo o *datasheet* disponível no anexo D. Sendo o alcance máximo de 0,89 m, temos:

$$T_{max/motor} = T_{sofrido} - T_{payload}$$

$$44,7 = 25,8 - (F_{payload} \times 0,89)$$

$$F_{payload} = 21,23[N]$$

De acordo com a Segunda Lei de Newton e considerando a aceleração da gravidade como 10 m/s^2 , obtemos uma carga máxima suportada pelo manipulador de:

$$F_r = m \times a$$

$$21,23 = m \times 10$$

$$m = \mathbf{2,1} \text{ [kg]}$$

APÊNDICE G

Algoritmo de busca e acionamento do painel

```
#include <ros/ros.h>
#include <geometry_msgs/PoseWithCovarianceStamped.h>
#include <moveit/move_group_interface/move_group_interface.h>
#include <moveit/planning_scene_interface/
    planning_scene_interface.h>
#include <moveit_msgs/DisplayRobotState.h>
#include <moveit_msgs/DisplayTrajectory.h>
#include <moveit_msgs/AttachedCollisionObject.h>
#include <moveit_msgs/CollisionObject.h>
#include <moveit_visual_tools/moveit_visual_tools.h>
#include <trajectory_msgs/JointTrajectory.h>
#include <trajectory_msgs/JointTrajectoryPoint.h>
#include <tf/transform_listener.h>
#include <tf/tf.h>
#include <geometry_msgs/Pose.h>
#include <moveit_msgs/PlanningScene.h>
#include <moveit_msgs/ApplyPlanningScene.h>

int main(int argc, char** argv)
{
    ros::init(argc, argv, "push_button_real");
    ros::NodeHandle nh;
    ros::AsyncSpinner spinner(2);
    spinner.start();
    static const std::string PLANNING_GROUP = "arm";
    moveit::planning_interface::MoveGroupInterface move_group(
        PLANNING_GROUP);
    moveit::planning_interface::PlanningSceneInterface
        planning_scene_interface;
    const robot_state::JointModelGroup* joint_model_group =
        move_group.getCurrentState()->getJointModelGroup(
            PLANNING_GROUP);
    namespace rvt = rviz_visual_tools;
    moveit_visual_tools::MoveItVisualTools visual_tools(
        "invisible_link");
```

```
visual_tools.deleteAllMarkers();
visual_tools.loadRemoteControl();
Eigen::Isometry3d text_pose = Eigen::Isometry3d::Identity();
text_pose.translation().z() = 1.75;
visual_tools.publishText(text_pose, "MoveGroupInterface Demo",
    rvt::WHITE, rvt::XLARGE);
visual_tools.trigger();
ROS_INFO_NAMED( "Reference frame: %s", move_group.
    getPlanningFrame().c_str());
ROS_INFO_NAMED( "End effector link: %s", move_group.
    getEndEffectorLink().c_str());

ros::Time start_time = ros::Time::now();

//JUNTAS
moveit::core::RobotStatePtr current_state = move_group.
    getCurrentState();
std::vector<double> joint_group_positions;
current_state->copyJointGroupPositions(joint_model_group,
    joint_group_positions); //COMANDO NO DOMÍNIO DAS JUNTAS
joint_group_positions[0] = 0.0;
joint_group_positions[1] = -0.12;
joint_group_positions[2] = 2.2;
joint_group_positions[3] = 1.57;
joint_group_positions[4] = 0.68;
move_group.setJointValueTarget(joint_group_positions);
moveit::planning_interface::MoveGroupInterface::Plan my_plan;
move_group.plan(my_plan);
while(move_group.plan(my_plan).val == -1){
    move_group.plan(my_plan);
}
//move_group.execute(my_plan)

sleep(5);

// Define variables for xyz for the box
double transform_fake_button_x;
double transform_fake_button_y;
double transform_fake_button_z;
```

```
double transform_fake_button_or_x;
double transform_fake_button_or_y;
double transform_fake_button_or_z;
double transform_fake_button_or_w;
// Define a collision object ROS message.
moveit_msgs::CollisionObject collision_object; //floor
collision_object.header.frame_id = move_group.getPlanningFrame()
();
moveit_msgs::CollisionObject collision_object1; //box
collision_object1.header.frame_id = move_group.
    getPlanningFrame();
// The id of the object is used to identify it.
collision_object.id = "floor";
collision_object1.id = "box";
// Define a box to add to the world.
// floor
shape_msgs::SolidPrimitive primitive;
primitive.type = primitive.BOX;
primitive.dimensions.resize(3);
primitive.dimensions[0] = 2.0;
primitive.dimensions[1] = 3.0;
primitive.dimensions[2] = 0.0;
// Define a pose for the box (specified relative to frame_id)
geometry_msgs::Pose box_pose;
box_pose.orientation.w = 0.0;
box_pose.position.x = 0.0;
box_pose.position.y = 0.0;
box_pose.position.z = 0.0;

tf::TransformListener listener;
//GETTING FAKE BUTTON POSE
tf::StampedTransform transform, transform_button_;
int i=2;
posicao:
try{
std :: cout << i;
i = i+1;
listener.waitForTransform("/invisible_link", "/fake_botao",

```

```
    ros::Time(0), ros::Duration(2.0));
listener.lookupTransform("/invisible_link", "/fake_botao",
                       ros::Time(0), transform);

transform_fake_button_x = transform.getOrigin().x();
transform_fake_button_y = transform.getOrigin().y();
transform_fake_button_z = transform.getOrigin().z();
transform_fake_button_or_x = transform.getRotation().x();
transform_fake_button_or_y = transform.getRotation().y();
transform_fake_button_or_z = transform.getRotation().z();
transform_fake_button_or_w = transform.getRotation().w();

}

catch (tf::TransformException &ex) {

if(i == 1){
moveit::core::RobotStatePtr current_state = move_group.
    getCurrentState();
std::vector<double> joint_group_positions;
current_state->copyJointGroupPositions(joint_model_group,
    joint_group_positions);
joint_group_positions[0] = 0;
joint_group_positions[1] = -0.006;
joint_group_positions[2] = 2.052;
joint_group_positions[3] = 1.57;
joint_group_positions[4] = 0.68;
move_group.setJointValueTarget(joint_group_positions);
move_group.plan(my_plan);
while(move_group.plan(my_plan).val == -1){
    move_group.plan(my_plan);
}
// move_group.execute(my_plan);
sleep(5.0);
}

if(i == 2){
moveit::core::RobotStatePtr current_state = move_group.
    getCurrentState();
std::vector<double> joint_group_positions;
```

```
current_state->copyJointGroupPositions(joint_model_group ,  
    joint_group_positions);  
joint_group_positions[0] = 0;  
joint_group_positions[1] = -0.4999;  
joint_group_positions[2] = 2.499;  
joint_group_positions[3] = 1.57;  
joint_group_positions[4] = 0.68;  
move_group.setJointValueTarget(joint_group_positions);  
move_group.plan(my_plan);  
while(move_group.plan(my_plan).val == -1){  
    move_group.plan(my_plan);  
}  
// move_group.execute(my_plan);  
sleep(5.0);  
}  
  
if(i == 3){  
moveit::core::RobotStatePtr current_state = move_group.  
    getCurrentState();  
std::vector<double> joint_group_positions;  
current_state->copyJointGroupPositions(joint_model_group ,  
    joint_group_positions);  
joint_group_positions[0] = 0.4;  
joint_group_positions[1] = -0.769;  
joint_group_positions[2] = 2.757;  
joint_group_positions[3] = 1.57;  
joint_group_positions[4] = 0.68;  
move_group.setJointValueTarget(joint_group_positions);  
move_group.plan(my_plan);  
while(move_group.plan(my_plan).val == -1){  
    move_group.plan(my_plan);  
}  
// move_group.execute(my_plan);  
sleep(5.0);  
}  
  
if(i == 4){  
moveit::core::RobotStatePtr current_state = move_group.  
    getCurrentState();
```

```
std :: vector<double> joint_group_positions;
current_state->copyJointGroupPositions(joint_model_group ,
    joint_group_positions);
joint_group_positions[0] = -0.4;
joint_group_positions[1] = -0.769;
joint_group_positions[2] = 2.757;
joint_group_positions[3] = 1.57;
joint_group_positions[4] = 0.68;
move_group.setJointValueTarget(joint_group_positions);
move_group.plan(my_plan);
while(move_group.plan(my_plan).val == -1){
    move_group.plan(my_plan);
}
// move_group.execute(my_plan);
i=0;
sleep(5.0);
}

goto posicao;

}

//GETTING BUTTON POSE
try{
    // ros::Time now = ros::Time::now();
    listener.waitForTransform("/invisible_link", "/botao_press",
        ros::Time(0), ros::Duration(2.0));
    listener.lookupTransform("/invisible_link", "/botao_press",
        ros::Time(0), transform_button_);
    std::cout<<"Pose do botão: x: "<< transform_button_.getOrigin
    () .x()<<
        " y: "<< transform_button_.getOrigin
    () .y()<<
        " z: "<< transform_button_.getOrigin
    () .z();

}

catch ( tf::TransformException &ex) {
i=0;
goto posicao;
```

```
}

ROS_INFO( "Aruco Detectado" );
ros::Duration delta_t1 = ros::Time::now() - start_time;
double delta_t1_sec = delta_t1.toSec();

std::cout << "Tempo de busca: " << delta_t1_sec;

if ( transform.getRotation().x() <= 0.5) {
    ROS_INFO( "Caixa na Horizontal" );
    // box
    shape_msgs::SolidPrimitive primitive1;
    primitive1.type = primitive.BOX;
    primitive1.dimensions.resize(3);
    primitive1.dimensions[0] = 0.2;
    primitive1.dimensions[1] = 0.3;
    primitive1.dimensions[2] = 0.2;
    // Define a pose for the box (specified relative to frame_id
    )
    geometry_msgs::Pose box_pose1;
    box_pose1.position.x = transform_fake_button_x ;
    box_pose1.position.y = transform_fake_button_y;
    box_pose1.position.z = transform_fake_button_z-0.354;
    box_pose1.orientation.w = transform_fake_button_or_w;
    box_pose1.orientation.x = transform_fake_button_or_x;
    box_pose1.orientation.y = transform_fake_button_or_y;
    box_pose1.orientation.z = transform_fake_button_or_z;
    //Collision box
    collision_object.primitives.push_back(primitive);
    collision_object.primitive_poses.push_back(box_pose);
    collision_object.operation = collision_object.ADD;
    collision_object1.primitives.push_back(primitive1);
    collision_object1.primitive_poses.push_back(box_pose1);
    collision_object1.operation = collision_object.ADD;

    std::vector<moveit_msgs::CollisionObject> collision_objects;
    collision_objects.push_back(collision_object);
    collision_objects.push_back(collision_object1);
```

```
planning_scene_interface.addCollisionObjects(  
    collision_objects);  
  
geometry_msgs::Pose target_pose;  
target_pose.orientation.w = transform.getRotation().w();  
target_pose.orientation.x = transform.getRotation().x();  
target_pose.orientation.y = transform.getRotation().y();  
target_pose.orientation.z = transform.getRotation().z();  
target_pose.position.x = transform.getOrigin().x();  
target_pose.position.y = transform.getOrigin().y();  
target_pose.position.z = transform.getOrigin().z();  
move_group.setGoalPositionTolerance(0.001);  
move_group.setGoalOrientationTolerance(0.35);  
move_group.setPlanningTime(25);  
move_group.setPoseTarget(target_pose);  
  
move_group.plan(my_plan);  
while (move_group.plan(my_plan).val != 1){  
    move_group.plan(my_plan);  
}  
// move_group.execute(my_plan);  
  
sleep(16.0);  
  
moveit::core::RobotStatePtr current_state_1 = move_group.  
    getCurrentState();  
std::vector<double> joint_group_positions_garra;  
current_state_1->copyJointGroupPositions(joint_model_group,  
    joint_group_positions_garra);  
joint_group_positions_garra[3] = 1.57;  
joint_group_positions_garra[4] = 1.21;  
move_group.setJointValueTarget(joint_group_positions_garra);  
moveit::planning_interface::MoveGroupInterface::Plan my_plan  
;  
move_group.plan(my_plan);  
while(move_group.plan(my_plan).val == -1){  
    move_group.plan(my_plan);
```

```
}

sleep(11.0);
// move_group.execute(my_plan);

//PRESS BUTTON

target_pose.position.z -=0.175;
std::vector<geometry_msgs::Pose> waypoints;
waypoints.push_back(target_pose);
moveit_msgs::RobotTrajectory trajectory;
const double jump_threshold = 0.0;
const double eef_step = 0.1;
move_group.computeCartesianPath(waypoints, eef_step,
                               jump_threshold, trajectory);
sleep(5.0);

geometry_msgs::Pose pose_real;
pose_real = move_group.getCurrentPose().pose;
std::cout << "Pose Real: " << pose_real.position;

target_pose.position.z +=0.11;
move_group.setPoseTarget(target_pose);
move_group.plan(my_plan);
while (move_group.plan(my_plan).val != 1){
    move_group.plan(my_plan);
}

else
{ // box
    shape_msgs::SolidPrimitive primitive1;
    primitive1.type = primitive.BOX;
    primitive1.dimensions.resize(3);
    primitive1.dimensions[0] = 0.2;
    primitive1.dimensions[1] = 0.3;
    primitive1.dimensions[2] = 0.2;
    // Define a pose for the box (specified relative to frame_id
}
```

```
geometry_msgs::Pose box_pose1;
    box_pose1.orientation.w = transform_fake_button_or_w;
    box_pose1.orientation.x = transform_fake_button_or_x;
    box_pose1.orientation.y = transform_fake_button_or_y;
    box_pose1.orientation.z = transform_fake_button_or_z;
    box_pose1.position.x = transform_fake_button_x -0.1;
    box_pose1.position.y = transform_fake_button_y+0.318;
    box_pose1.position.z = transform_fake_button_z - 0.1;
//Collision box
collision_object.primitives.push_back(primitive);
collision_object.primitive_poses.push_back(box_pose);
collision_object.operation = collision_object.ADD;
collision_object1.primitives.push_back(primitive1);
collision_object1.primitive_poses.push_back(box_pose1);
collision_object1.operation = collision_object.ADD;

std::vector<moveit_msgs::CollisionObject> collision_objects;
collision_objects.push_back(collision_object);
collision_objects.push_back(collision_object1);

planning_scene_interface.addCollisionObjects(
    collision_objects);

geometry_msgs::Pose target_pose;

target_pose.orientation.w = transform.getRotation().w();
target_pose.orientation.x = transform.getRotation().x();
target_pose.orientation.y = transform.getRotation().y();
target_pose.orientation.z = transform.getRotation().z();
target_pose.position.x = transform.getOrigin().x();
target_pose.position.y = transform.getOrigin().y();
target_pose.position.z = transform.getOrigin().z();
move_group.setGoalPositionTolerance(0.001);
move_group.setGoalOrientationTolerance(1.0);
move_group.setPlanningTime(25);
move_group.setPoseTarget(target_pose);
move_group.plan(my_plan);
while (move_group.plan(my_plan).val != 1){
```

```
move_group.plan(my_plan);
}
// move_group.execute(my_plan);
sleep(10.0);
moveit::core::RobotStatePtr current_state_1 = move_group.
    getCurrentState();
std::vector<double> joint_group_positions_garra;
current_state_1->copyJointGroupPositions(joint_model_group,
    joint_group_positions_garra);
joint_group_positions_garra[3] = -2.12;
joint_group_positions_garra[4] = 1.21;
move_group.setJointValueTarget(joint_group_positions_garra);
moveit::planning_interface::MoveGroupInterface::Plan my_plan
    ;
move_group.plan(my_plan);
while(move_group.plan(my_plan).val == -1){
    move_group.plan(my_plan);
}
sleep(10.0);

//PRESS BUTTON
move_group.setGoalOrientationTolerance(0.9);
float or_w, or_x, or_y, or_z, x, y;
or_x = move_group.getCurrentPose().pose.orientation.x;
or_y = move_group.getCurrentPose().pose.orientation.y;
or_z = move_group.getCurrentPose().pose.orientation.z;
or_w = move_group.getCurrentPose().pose.orientation.w;
target_pose.orientation.w = or_w;
target_pose.orientation.x = or_x;
target_pose.orientation.y = or_y;
target_pose.orientation.z = or_z;
target_pose.position.y += 0.172;
target_pose.position.z += 0.018;
move_group.setPoseTarget(target_pose);
// moveit::planning_interface::MoveGroupInterface::Plan
my_plan4;
move_group.plan(my_plan);
while (move_group.plan(my_plan).val != 1){
    move_group.plan(my_plan);
```

```
    }

    sleep(10.0);

    target_pose.position.y -=0.2;
    move_group.setPoseTarget(target_pose);
    move_group.plan(my_plan);
    while (move_group.plan(my_plan).val != 1){
        move_group.plan(my_plan);
    }
    // move_group.execute(my_plan);

}

//BACK TO HOME POSITION

current_state->copyJointGroupPositions(joint_model_group,
    joint_group_positions);
joint_group_positions[0] = 0.0;
joint_group_positions[1] = -0.12;
joint_group_positions[2] = 2.2;
joint_group_positions[3] = 0;
joint_group_positions[4] = 0.0;
move_group.setJointValueTarget(joint_group_positions);

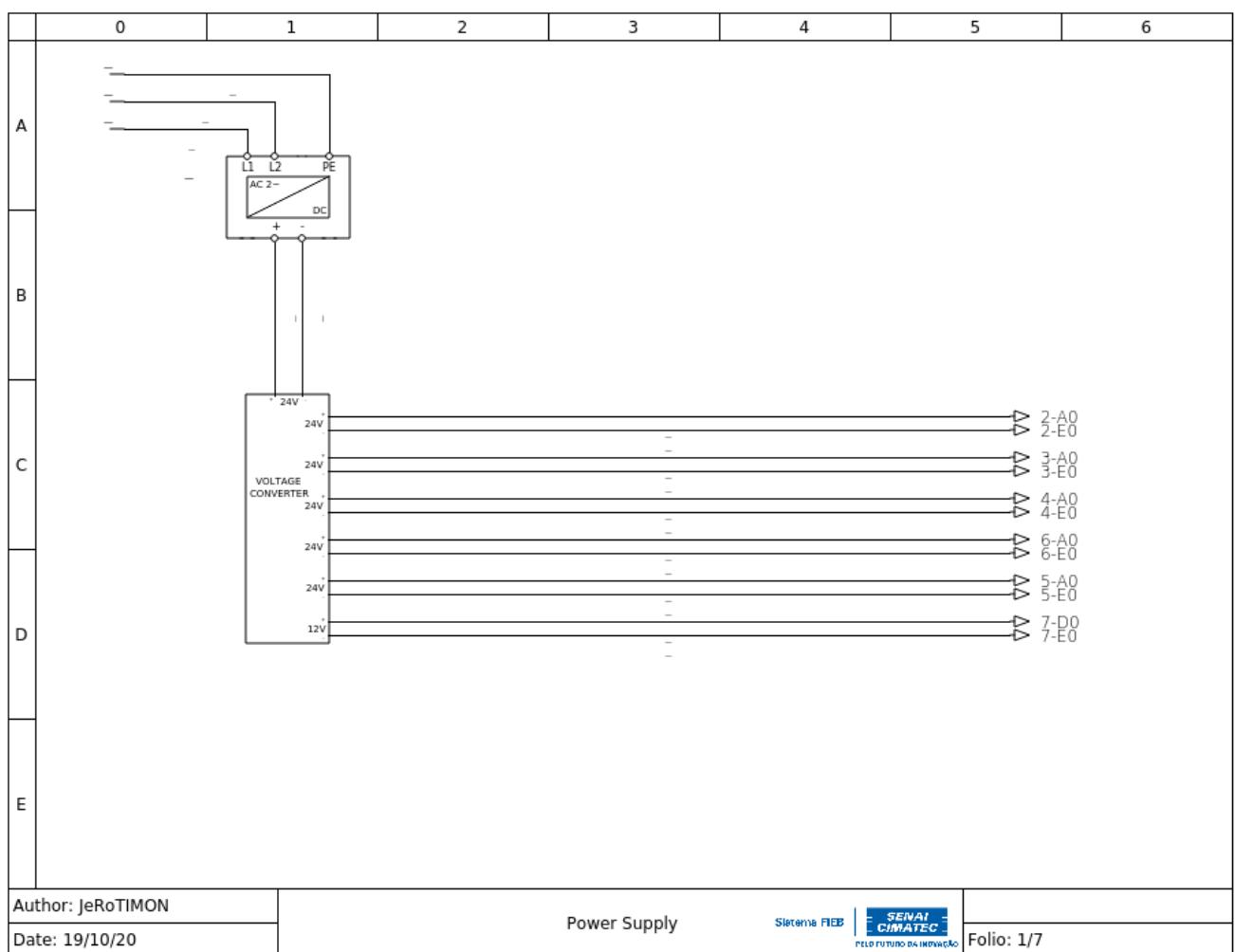
move_group.plan(my_plan);
while (move_group.plan(my_plan).val != 1){
    move_group.plan(my_plan);
}
// move_group.execute(my_plan);
sleep(5.0);
ros::Duration delta_t = ros::Time::now() - start_time;
double delta_t_sec = delta_t.toSec();

std::cout<< "Tempo de missão: " << delta_t_sec;

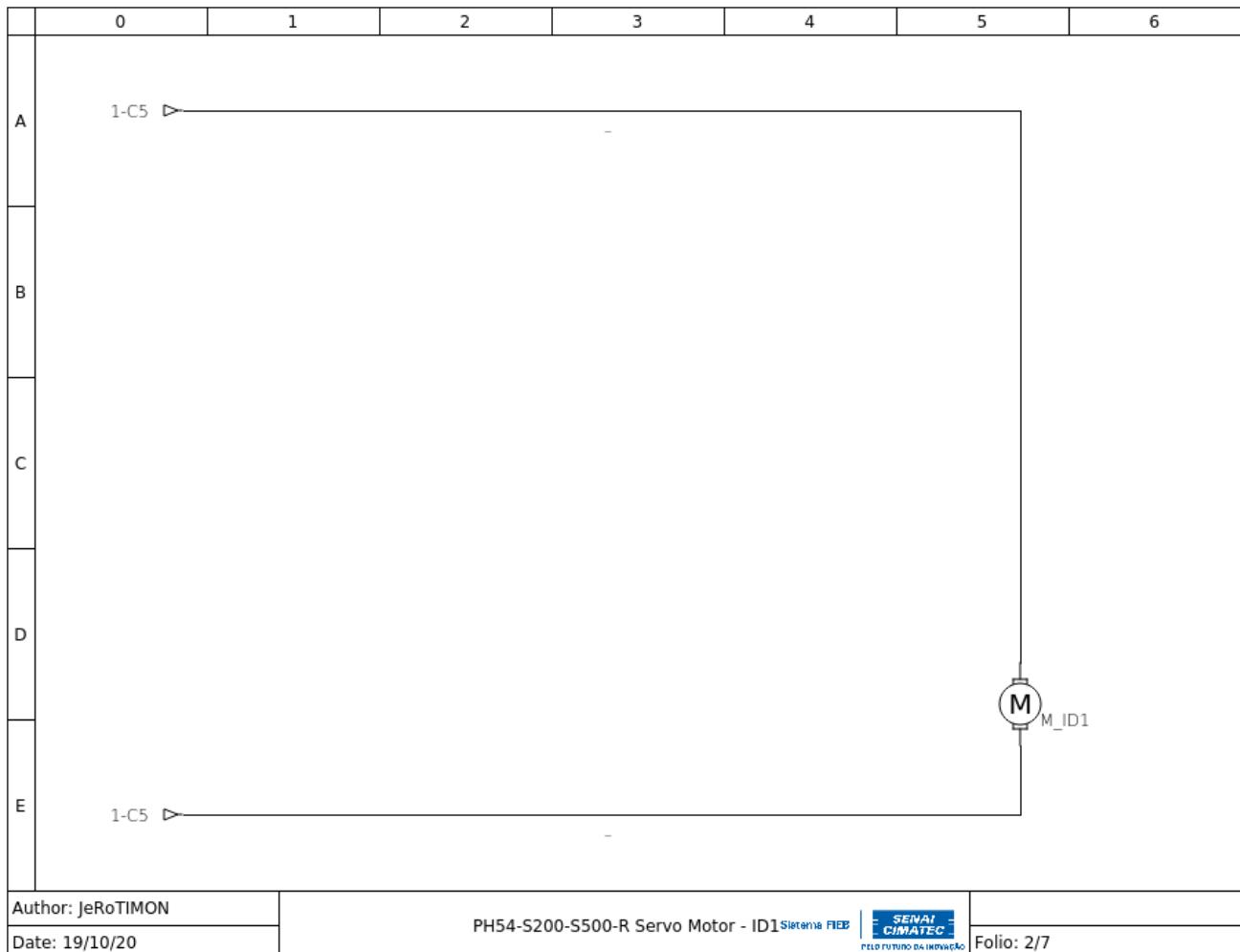
ros::shutdown();
return 0;
}
```

APÊNDICE H

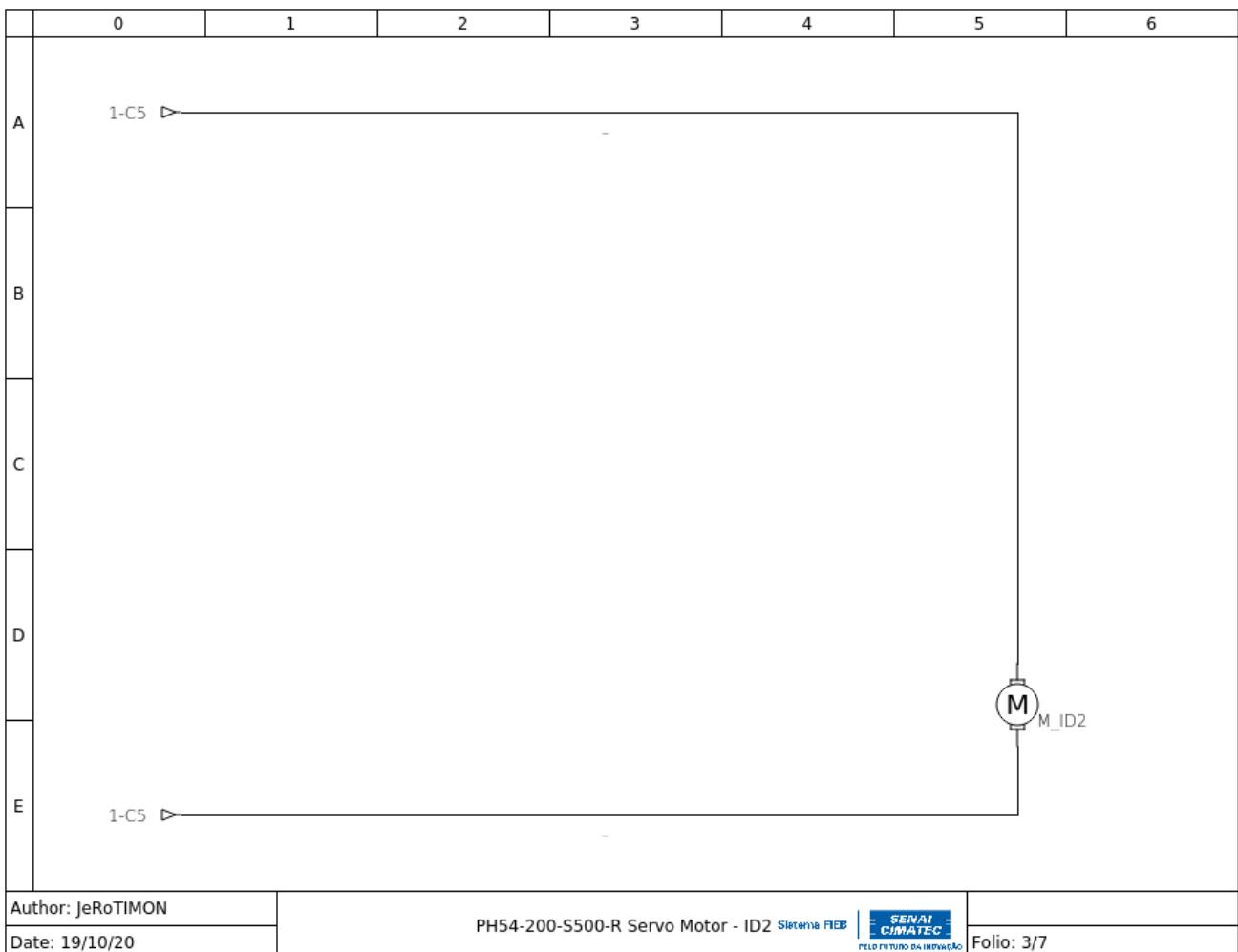
Diagrama elétrico do JeRoTIMON



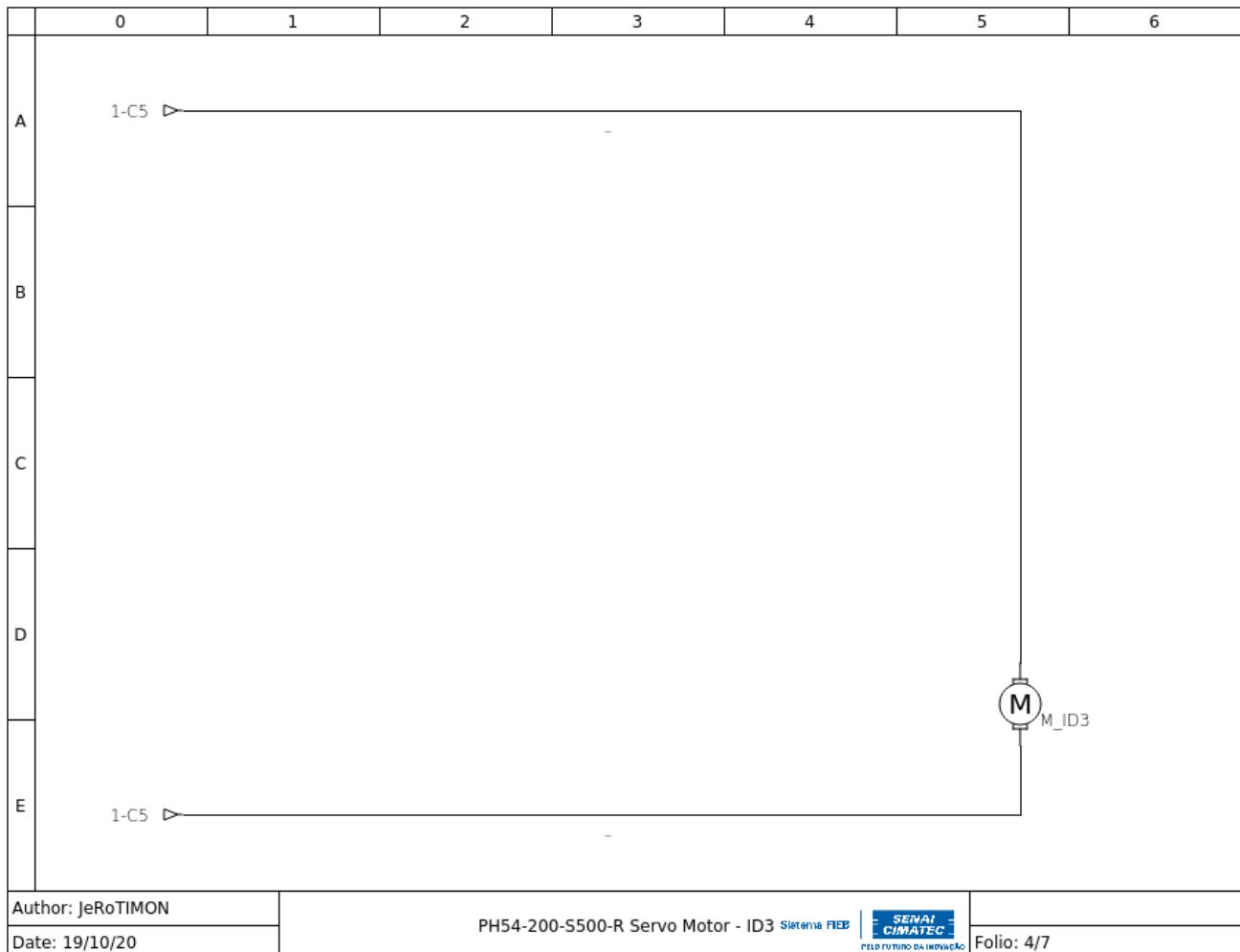
Apêndice B



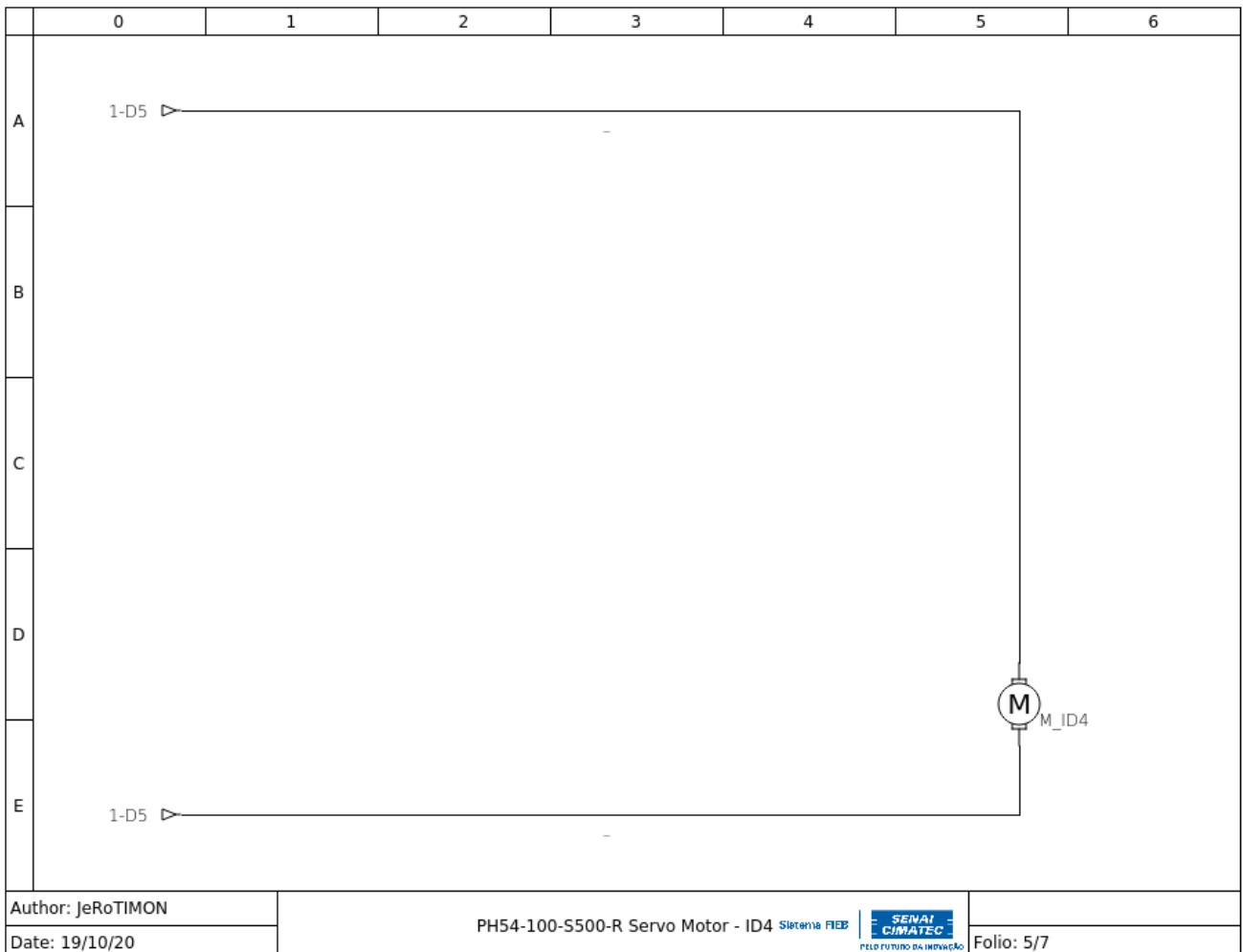
Apêndice B



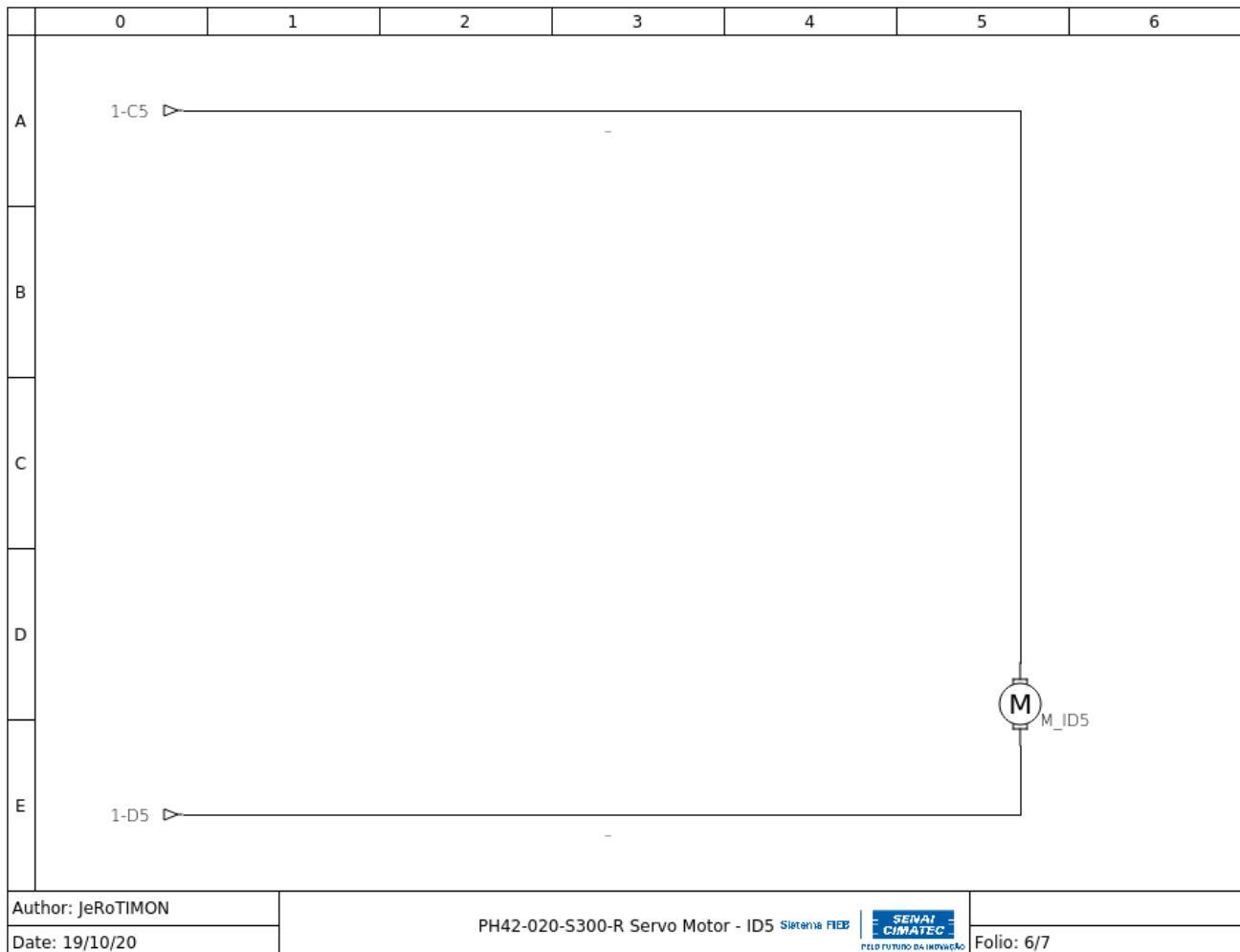
Apêndice B



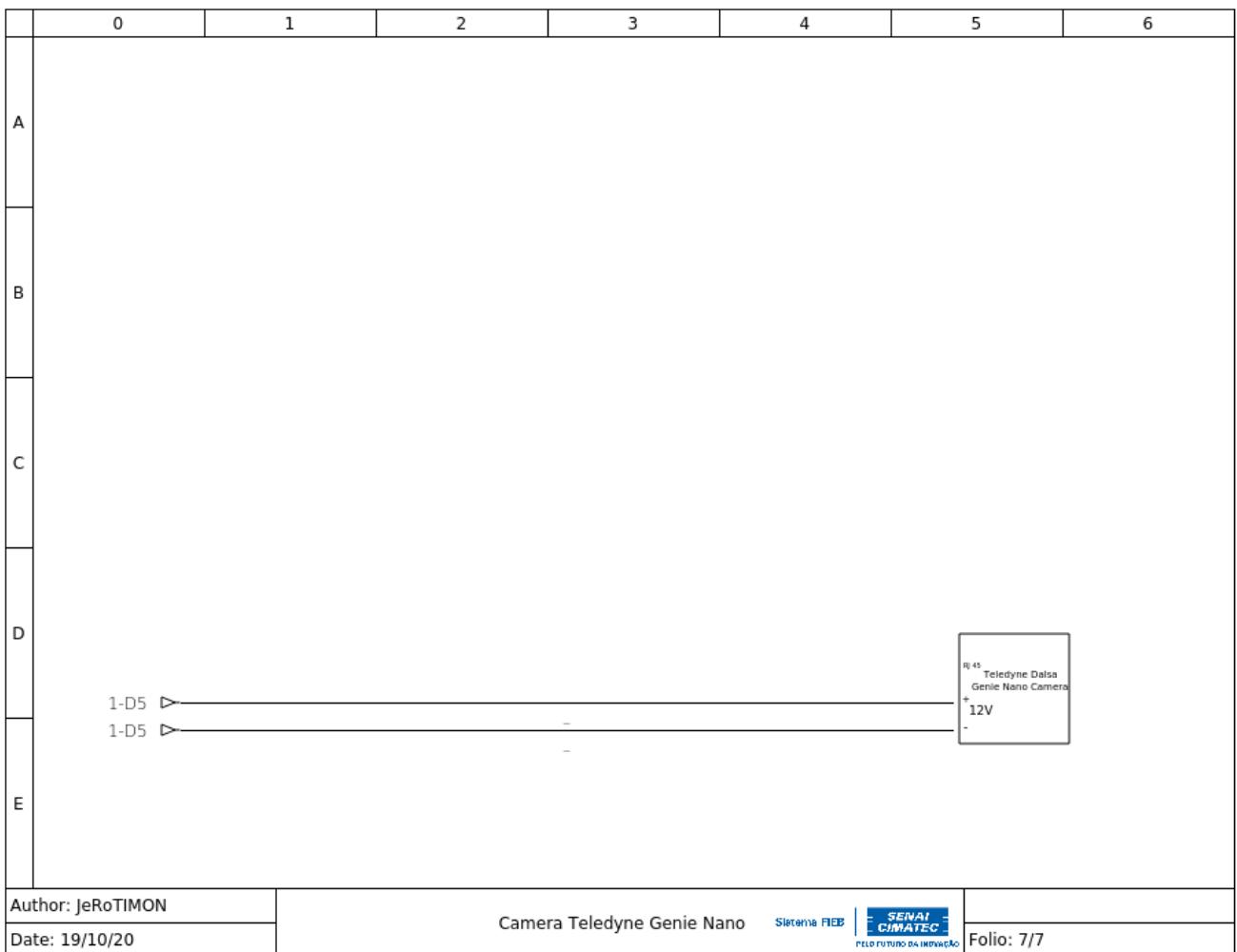
Apêndice B



Apêndice B

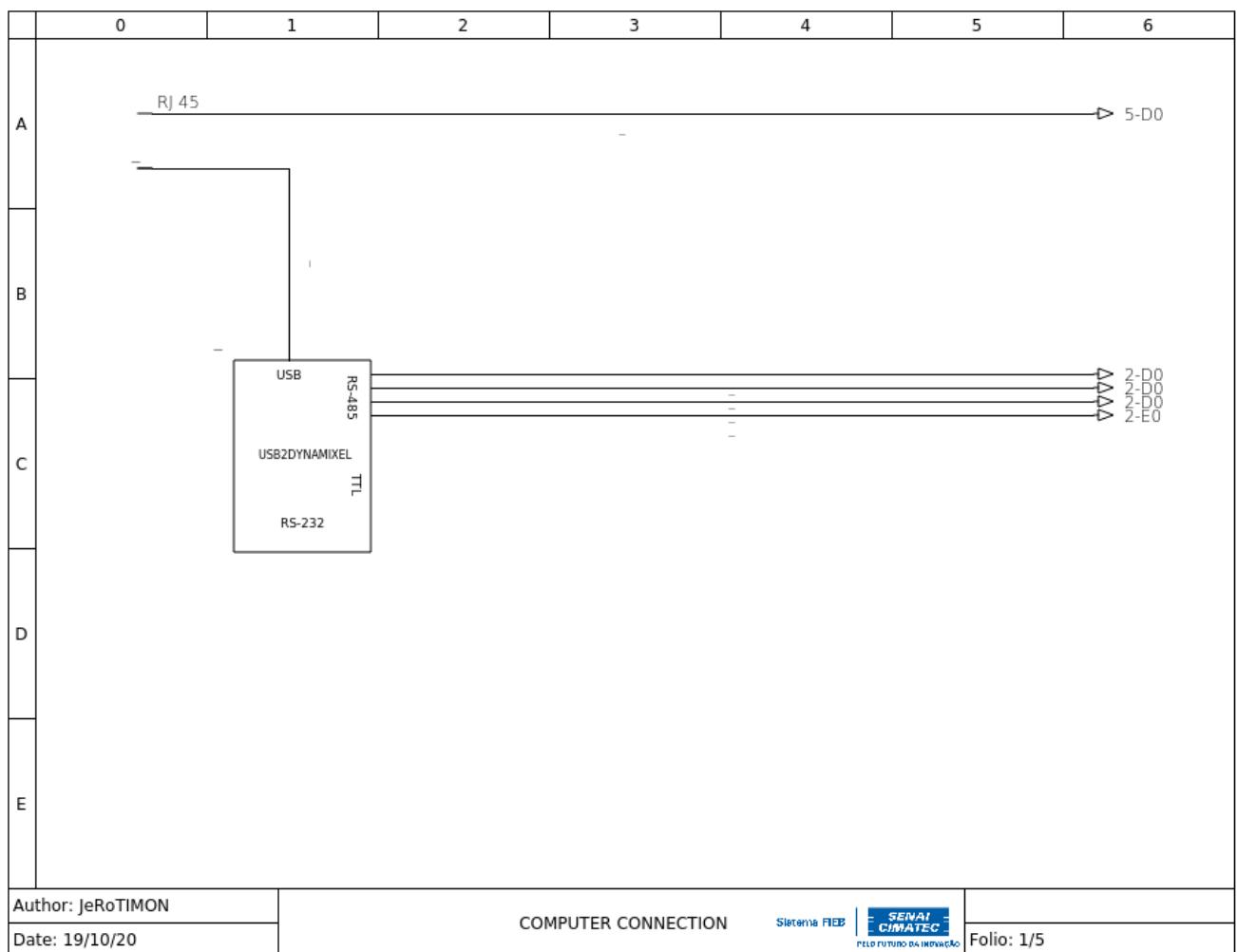


Apêndice B

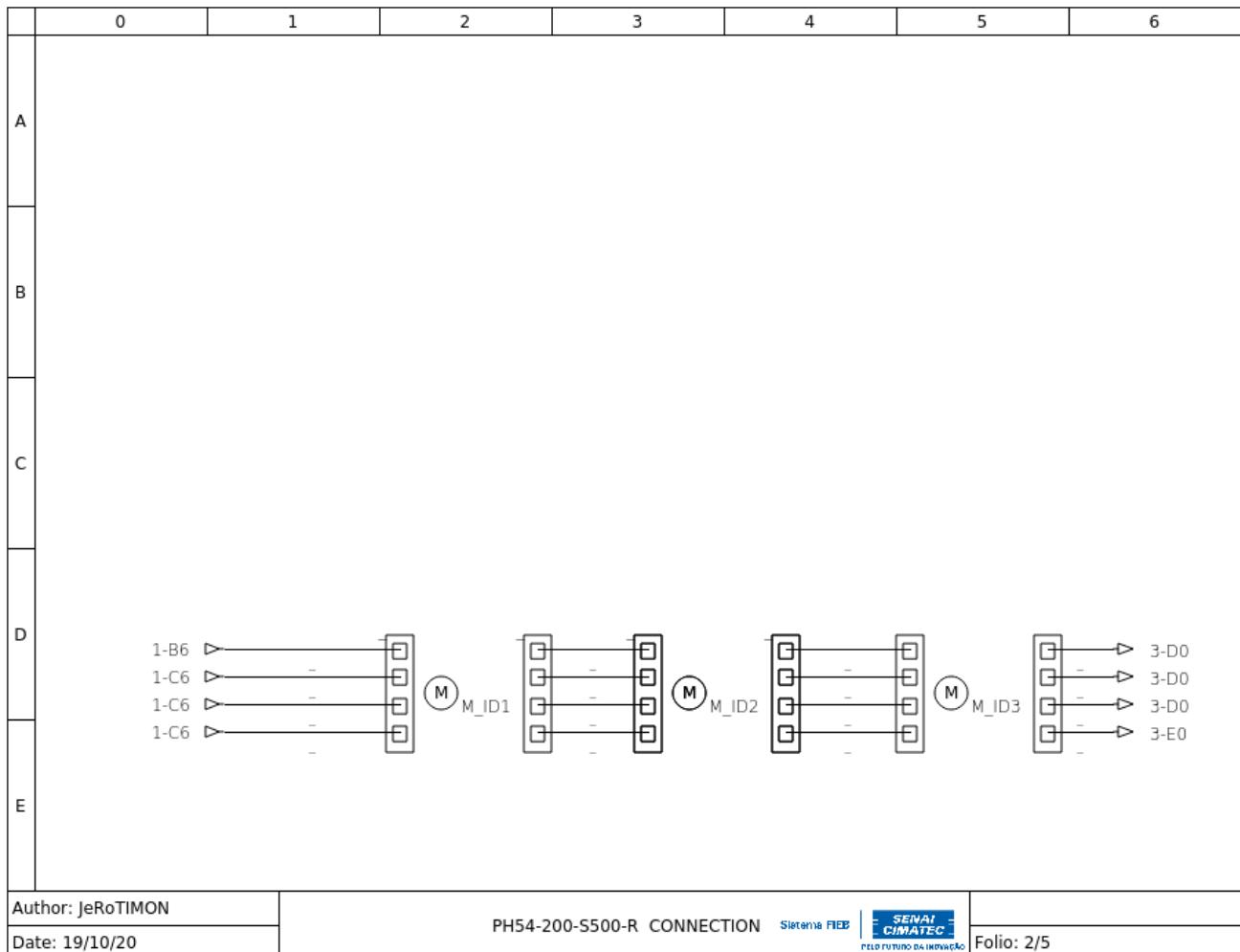


APÊNDICE I

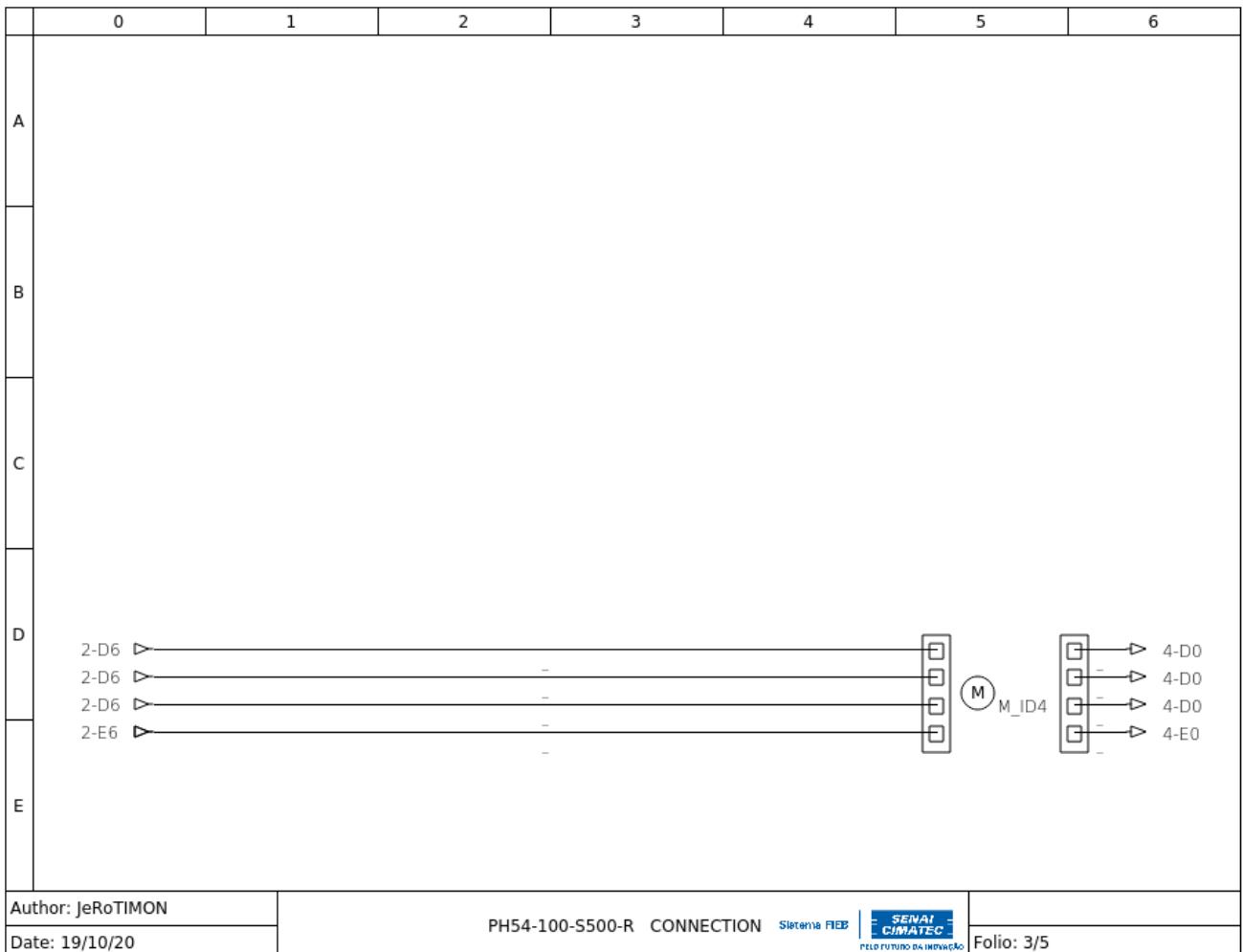
Diagrama de conexão do JeRoTIMON



Apêndice B



Apêndice B



Apêndice B

	0	1	2	3	4	5	6
A							
B							
C							
D	3-D6 ►						
	3-D6 ►	-					
	3-D6 ►	-					
	3-E6 ►	-					
E							

Author: JeRoTIMON
Date: 19/10/20

PH42-020-S300-R CONNECTION Sistema FIEB | SENAI CIMATEC
FELIZ FUTURO DA INovação

Folio: 4/5

Apêndice B

	0	1	2	3	4	5	6
A							
B							
C							
D	1-A6	►		-			RJ 45 Teledyne Dalsa Genie Nano Camera + 12V -
E							
Author: JeRoTIMON		CAMERA CONNECTION			Sistema FIEP SENAI CIMATEC PELO FUTURO DA INOVAÇÃO		Folio: 5/5
Date: 19/10/20							

ANEXO A

Especificações da câmera Dalsa Genie Nano

Specifications: M2590, M2590-NIR, C2590

Supported Features	M2590, M2590-NIR	Nano-C2590	
Resolution	2592 x 2048		
Sensor	OnSemi Python5000 P1 (5.1M)		
Pixel Size	4.8 µm x 4.8 µm		
Shutter type	Full frame electronic global shutter function		
Full Well charge	10ke (max)		
Firmware option (Field programmable)	Standard Design Monochrome (factory default)	Standard Design Bayer (factory default)	RGB-Output Design
Max. Internal Frame Rate Full Resolution (2592 x 2048)	51.8 fps (Fast Readout Enable) 24.7 fps (Normal Readout Enable)		
Maximum Sustained Frame Rate Output (with TurboDrive v1)*	42.7 fps (8-bit) 24.9 fps (10-bit)		N/A
Maximum Sustained Frame Rate Output (without TurboDrive)	22 fps (8-bit)		5.5 fps (RGBA) 8.7 fps (RGB) 11 fps (Yuv422) 22 fps (8-bit mono)
Pixel Data Formats	Mono 8-bit Mono 10-bit	Bayer 8-Bit Bayer 10-Bit	RGBA 32-bit RGB 24-bit Yuv422 16-bit Mono 8-bit
Trigger to Exposure Minimum delay (Synchronous Exposure Alignment)	8 µs if exposureAlignment = Synchronous With No Overlap between the new exposure and the previous readout 26.2 µs if exposureAlignment = Synchronous With Overlap between the new exposure and the previous readout		
Trigger to Exposure Minimum delay (Reset Exposure Alignment)	3 µs		
Trigger to Exposure Start jitter (best case with Synchronous Exposure Alignment)	Up to 1 line time		
Trigger to Exposure Start jitter (Reset Exposure Alignment) †	0 µs		
Exposure Time Minimum (see "exposureTimeActual" in Sensor Control)	87 µs (increment steps of 1µs)		
Min. Time from End of Exposure to Start of Next Exposure (second frame)	49 µs – Normal Readout 47 µs – Fast Readout		
Horizontal Line Time:	11.33 µs – Normal Readout 9.33 µs – Fast Readout		
Readout Time	23242 µs – Normal Readout for 2592 x 2048 Add 76µs when overlapping Exposure and Readout 19142 µs – Fast Readout for 2592 x 2048 Add 64µs when overlapping Exposure and Readout <i>Specifically: (Horizontal line time at current resolution * number of lines) + (3 * (line time of the 2590 model))</i>		
Auto-Brightness	Yes , with Auto-Exposure and AGC (FPGA Gain)		
Black offset control	Yes (in DN)		

Gain Control	In-sensor Analog Gain (1.0x to 8x) in 11 gain steps (1.0, 1.14, 1.33, 1.6, 2.0, 2.29, 2.67, 3.2, 4.0, 5.33, 8.0) In-sensor Digital Gain (1x to 32x) in 0.01x steps In-FPGA Digital Gain (1x to 4x) in 0.007x steps	
Binning Support	Yes In-FPGA (summing and average, 2x2, 4x4) Yes In- Sensor (averaging 2x2)	No
Color Correction Support	No	Yes
Decimation Support	No	
Defective Pixel Replacement	Yes, up to 512 positions	
Image Correction	No	
Image Flip Support	Yes, In-Sensor, Vertical Only	
Multi-ROI Support	Yes, in Sensor, up to 16 ROI (mutually exclusive with binning)	
On-Board Image Memory	90MB	
Output Dynamic Range (dB)	62.1 dB (in 10-Bit Pixel Format)	
SNR (dB)	39.8 dB (in 10-Bit Pixel Format)	

*TurboDrive internal limitation of 250MB/sec

† Note: The actual internal minimum exposure may be different than what is programmed. Use the feature "exposureTimeActual" from the [Sensor Control](#) category to read back the actual sensor exposure. The exposure start sensor event is delayed 4 µs from the actual start.

Firmware Files for Models 1280, 1930, 2590

The latest firmware files for all Nano models are available on the Teledyne DALSA support web site:
<http://www.teledynedalsa.com/imaging/support/downloads/firmware/>

The firmware files for these models are listed below. The xx denotes the build number.

M1280, M1930, M2590

- Standard
"Genie_Nano_OnSemi_Python_0.3M-0.5M-1.3M-2M-5M_Mono_STD_Firmware_5CA18.xx.cbf"

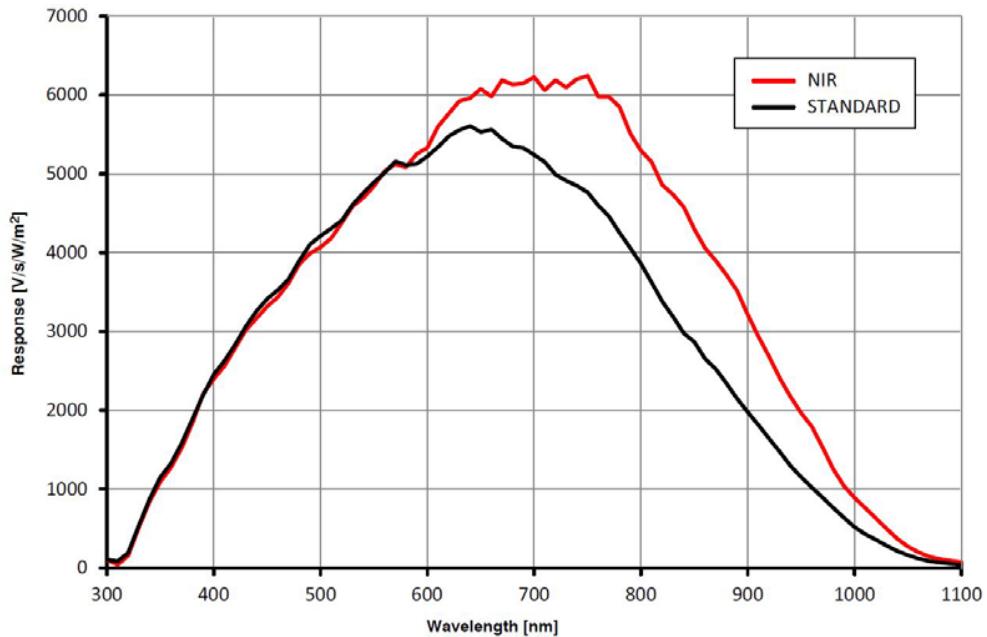
C1280, C1930, C2590

- Bayer Output
"Genie_Nano_OnSemi_Python_0.3M-0.5M-1.3M-2M-5M_Bayer_STD_Firmware_6CA18.xx.cbf"
- RGB Output
"Genie_Nano_OnSemi_Python_0.3M-0.5M-1.3M-2M-5M_RGB_Output_Firmware_6CA18.xx.cbf"

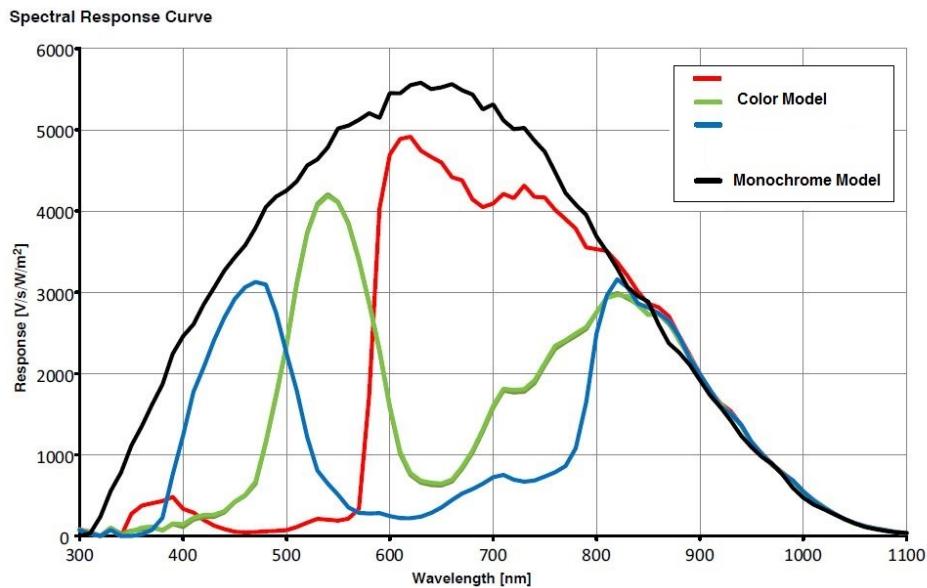
Spectral Response (Python 4.8 µm series)

Model specific specifications and response graphics for the On-Semi Python (VGA to 5M) series are provided here. The response curves describe the sensor, excluding lens and light source characteristics.

On-Semi Python Series (with 4.8 µm pixels) — Monochrome and NIR



On-Semi Python Series (with 4.8 µm pixels) — Monochrome and Color



ANEXO B

Especificações do motor *Dynamixel PH42-020-S300-R*

Item	Especificação
Microcontrolador	ARM CORTEX-M4 (168 [MHz], 32Bit)
Motor	Coreless (Maxon)
Taxa de transmissão	9,600 [bps] ~10.5 [Mbps]
Modos de operação	Torque Control Mode
	Velocity Control Mode
	Position Control Mode
	Extended Position Control Mode
	PWM Control Mode (Voltage Control Mode)
Massa	340 [g]
Dimensões (L x A x P)	42 x 84 x 42 [mm]
Resolução	607,500 [pulsos/rev]
Relação de transmissão	303.75:1
Folga	<6 [arcmin], 0.1 [°]
Carga radial	280 [N] (10 [mm] afastado da face do eixo)
Carga axial	100 [N]
Velocidade sem carga	32.7 [rev/min]
Corrente sem carga	0.57 [A]
Velocidade contínua	29.2 [rev/min]
Torque contínuo	5.1 [Nm]
Corrente contínua	1.5 [A]
Potência de saída	20 [W]
Temperatura de operação	5 ~55 [°C]
Tensão de operação	24.0 [V]
Sinal de comando	Pacotes digitais
Protocolo de comunicação	RS485 Comunicação serial asíncrona (8bit, 1 bit de parada, sem paridade)
Conexão física	RS485 Barramento multiponto
ID	253 ID (0 ~252)
Corrente de espera	80 [mA]

Tabela 24: Especificações do motor *Dynamixel HP42-020-S300-R*

Enter Search Terms



DYNAMIXEL

PLATFORM

STEAM

SOFTWARE

PARTS

FAQ

Apêndice B

PH42-020-S300-R

- 1. Specifications >
- 2. Control Table >
- 3. How to Assemble >
- 4. Maintenance
- 5. Reference >

[Edit on GitHub](#)

ROBOTIS



Robot Source



GitHub



TOP



PH42-020-S300-R

1. Specifications

Item	Specifications
MCU	ARM CORTEX-M4 (168 [MHz], 32Bit)
Motor	Coreless (Maxon)
Baud Rate	9,600 [bps] ~ 10.5 [Mbps]
Operating Modes	Torque Control Mode Velocity Control Mode Position Control Mode Extended Position Control Mode PWM Control Mode(Voltage Control Mode)
Weight	340 [g]
Dimensions (W x H x D)	42 x 84 x 42 [mm]
Resolution	607,500 [pulse/rev]
Gear Ratio	303.75:1
Backlash	< 6 [arcmin], 0.1 [°]
Radial Load	280 [N] (10 [mm] away from the horn)
Axial Load	100 [N]
No Load Speed	32.7 [rev/min]
No Load Current	0.57 [A]
Continuous Speed	29.2 [rev/min]
Continuous Torque	5.1 [Nm]
Continuous Current	1.5 [A]
Output	20 [W]
Operating Temperature	-5 ~ 55 [°C]
Input Voltage	24.0 [V]
Command Signal	Digital Packet
Protocol Type	RS485 Asynchronous Serial Communication (8bit, 1stop, No Parity)
Physical Connection	RS485 Multidrop Bus

Apêndice B

PH42-020-S300-R

- 1. Specifications
- 2. Control Table
- 3. How to Assemble
- 4. Maintenance
- 5. Reference

Item	Specifications
ID	253 ID (0 ~ 252)
Standby Current	30 [mA]

These specifications are calculated based on the specifications of the core motor. Please consult ROBOTIS for the long term use or special use, or else refer to the Performance Graph for general use.



TOP

**DANGER**

(May cause serious injury or death)

- Never place items containing water, flammables, and solvents near product.
- Never place fingers, arms, toes, and other body parts near product during operation.
- Cut power off if product emits strange odors or smoke.
- Keep product out of reach of children.
- Check the power's polarity before wiring.

**CAUTION**

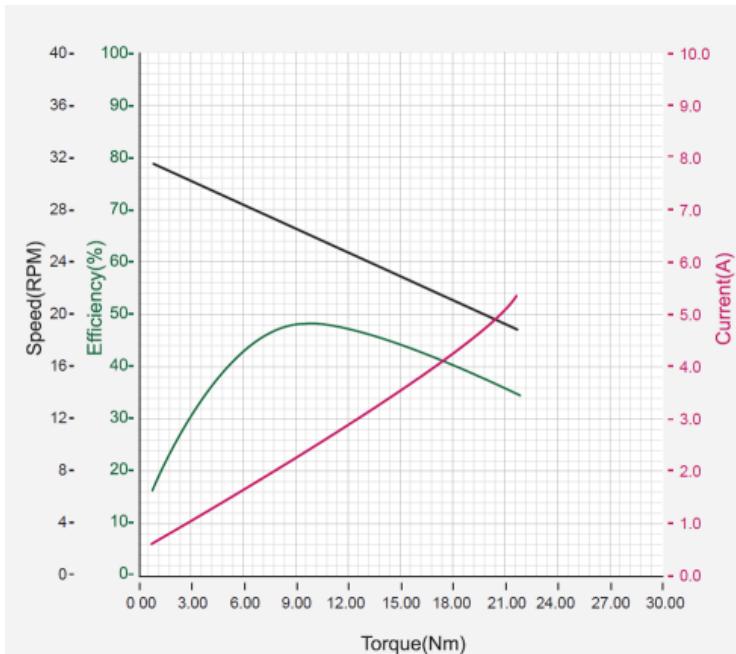
(May cause injury or damage to product)

- Do not operate the product at a temperature exceeding -5 ~ 55 [°C] range.
- Do not insert sharp blades nor pins during product operation.

**ATTENTION**

(May cause injury or damage to product)

- Do not disassemble or modify product.
- Do not drop or apply strong shock to product.

1. 1. Performance Graph[Show Enlarged Graph](#)

Enter Search Terms



Apêndice B

PH42-020-S300-R

- 1. Specifications >
- 2. Control Table >
- 3. How to Assemble >
- 4. Maintenance
- 5. Reference >



TOP

NOTE : The Max Torque and the Stall Torque of Performance Graph are different in measurement methods. Stall torque is a measured value of the momentary torque that it can reach. This is generally how RC servos are measured. The Performance graph is also called as N-T curves, which is measured with the gradually increasing load. The actual motor operation environment is closer to the performance graph, not stall torque method. For this reason, the performance graph is broadly used in the industrial field. Generally, Max Torque of the Performance Graph is less than the Stall Torque.

CAUTION : When supplying power

- It is recommended using ROBOTIS controller or SMPS2DYNAMIXEL.
- Do not connect or disconnect DYNAMIXEL when power is being supplied.
- In case of DYNAMIXEL PRO and DYNAMIXEL-P series, please supply power through 24V power port.

2. Control Table

The Control Table is a structure of data implemented in the device. Users can read a specific Data to get status of the device with Read Instruction Packets, and modify Data as well to control the device with WRITE Instruction Packets.

WARNING : DYNAMIXEL-P series use different Control Table from DYNAMIXEL PRO series. Please pay attention when replacing DYNAMIXEL PRO with DYNAMIXEL-P series.

2. 1. Control Table, Data, Address

The Control Table is a structure that consists of multiple Data fields to store status or to control the device. Users can check current status of the device by reading a specific Data from the Control Table with Read Instruction Packets. WRITE Instruction Packets enable users to control the device by changing specific Data in the Control Table. The Address is a unique value when accessing a specific Data in the Control Table with Instruction Packets. In order to read or write data, users must designate a specific Address in the Instruction Packet. Please refer to [Protocol 2.0](#) for more details about Instruction Packets.

NOTE : Two's complement is applied for the negative value. For more information, please refer to [Two's complement](#) from Wikipedia.

2. 1. 1. Area (EEPROM, RAM)

The Control Table is divided into 2 Areas. Data in the RAM Area is reset to initial values when the power is reset(Volatile). On the other hand, data in the EEPROM Area is maintained even when the device is powered off(Non-Volatile).

Data in the EEPROM Area can only be written to if Torque Enable(512) is cleared to '0'(Off).

2. 1. 2. Size

The Size of data varies from 1 ~ 4 bytes depend on their usage. Please check the size of data when updating the data with an Instruction Packet. For data larger than 2 bytes will be saved according to [Little Endian](#).

2. 1. 3. Access

The Control Table has two different access properties. 'RW' property stands for read and write access permission while 'R' stands for read only access permission. Data with the read only property cannot be changed by the WRITE Instruction. Read only property('R') is generally used for measuring and monitoring purpose, and read write property('RW') is used for controlling device.

ANEXO C

Especificações do motor Dynamixel PH54-100-S500-R

Item	Especificação
Microcontrolador	ARM CORTEX-M4 (168 [MHz], 32Bit)
Motor	BLDC (Maxon)
Taxa de transmissão	9,600 [bps] ~10.5 [Mbps]
Modos de operação	Controle de torque
	Controle de velocidade
	Controle de posição
	Controle de posição extendida (mais do que uma revolução)
	Controle por PWM (Controle de tensão)
Massa	740 [g]
Dimensões (L x A x P)	54 x 108 x 54 [mm]
Resolução	1,003,846 [pulsos/rev]
Relação de transmissão	501.923 : 1
Folga	<6 [arcmin], 0.1 [°]
Carga radial	370 [N] (10 [mm] afastado da face do eixo)
Carga axial	130 [N]
Velocidade sem carga	33.3 [rev/min]
Corrente sem carga	1.13 [A]
Velocidade contínua	29.2 [rev/min]
Torque contínuo	25.3 [Nm]
Corrente contínua	5.5 [A]
Potência de saída	100 [W]
Temperatura de operação	5 ~55 [°C]
Tensão de operação	24.0 [V]
Sinal de comando	Pacotes digitais
Protocolo de comunicação	RS485 Comunicação serial asíncrona (8bit, 1 bit de parada, sem paridade)
Conexão física	RS485 Barramento multiponto
ID	253 ID (0 ~252)
Corrente de espera	40 [mA]

Tabela 25: Especificações do motor *Dynamixel PH54-100-S500-R*

ROBOTIS e-Manual

DYNAMIXEL PLATFORM STEAM SOFTWARE PARTS FAQ

Enter Search Terms 

[Edit on GitHub](#)  ROBOTIS  Robot Source  GitHub 

[TOP](#)

PH54-100-S500-R

- 1. Specifications >
- 2. Control Table >
- 3. How to Assemble >
- 4. Maintenance >
- 5. Reference >



PH54-100-S500-R

1. Specifications

Item	Specifications
MCU	ARM CORTEX-M4 (168 [MHz], 32Bit)
Motor	BLDC (Maxon)
Baud Rate	9,600 [bps] ~ 10.5 [Mbps]
Operating Modes	Torque Control Mode Velocity Control Mode Position Control Mode Extended Position Control Mode PWM Control Mode(Voltage Control Mode)
Weight	740 [g]
Dimensions (W x H x D)	54 x 108 x 54 [mm]
Resolution	1,003,846 [pulse/rev]
Gear Ratio	501.923 : 1
Backlash	< 6 [arcmin], 0.1 [°]
Radial Load	370 [N] (10 [mm] away from the horn)
Axial Load	130 [N]
No Load Speed	33.3 [rev/min]
No Load Current	1.13 [A]
Continuous Speed	29.2 [rev/min]
Continuous Torque	25.3 [Nm]
Continuous Current	5.5 [A]
Output	100 [W]
Operating Temperature	-5 ~ 55 [°C]
Input Voltage	24.0 [V]

PH54-100-S500-R

- 1. Specifications >
- 2. Control Table >
- 3. How to Assemble >
- 4. Maintenance
- 5. Reference >

Item	Specifications
Command Signal	Digital Packet
Protocol Type	RS485 Asynchronous Serial Communication (8bit, 1stop, No Parity)
Physical Connection	RS485 Multidrop Bus
ID	253 ID (0 ~ 252)
Standby Current	40 [mA]

1 These specifications are calculated based on the specifications of the core motor. Please consult ROBOTIS for the long term use or special use, or else refer to the Performance Graph for general use.

**DANGER**

(May cause serious injury or death)

- Never place items containing water, flammables, and solvents near product.
- Never place fingers, arms, toes, and other body parts near product during operation.
- Cut power off if product emits strange odors or smoke.
- Keep product out of reach of children.
- Check the power's polarity before wiring.

**CAUTION**

(May cause injury or damage to product)

- Do not operate the product at a temperature exceeding -5 ~ 55 [°C] range.
- Do not insert sharp blades nor pins during product operation.

**ATTENTION**

(May cause injury or damage to product)

- Do not disassemble or modify product.
- Do not drop or apply strong shock to product.

1. 1. Performance Graph

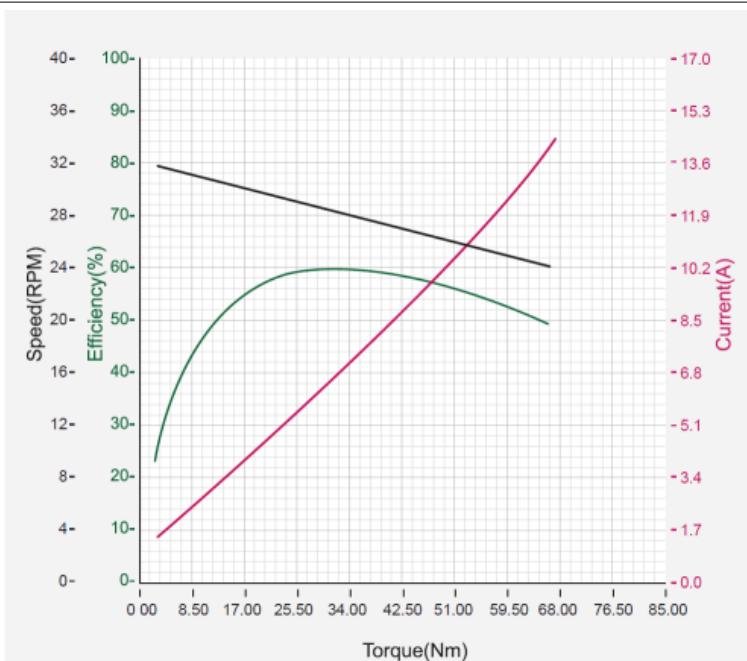
TOP

🔍

G
▲ TOP

PH54-100-S500-R

- 1. Specifications ➤
- 2. Control Table ➤
- 3. How to Assemble ➤
- 4. Maintenance ➤
- 5. Reference ➤



[Show Enlarged Graph](#)

NOTE : The Max Torque and the Stall Torque of Performance Graph are different in measurement methods. Stall torque is a measured value of the momentary torque that it can reach. This is generally how RC servos are measured. The Performance graph is also called as N-T curves, which is measured with the gradually increasing load. The actual motor operation environment is closer to the performance graph, not stall torque method. For this reason, the performance graph is broadly used in the industrial field. Generally, Max Torque of the Performance Graph is less than the Stall Torque.

CAUTION : When supplying power

- It is recommended using ROBOTIS controller or SMPS2DYNAMIXEL.
- Do not connect or disconnect DYNAMIXEL when power is being supplied.
- In case of DYNAMIXEL PRO and DYNAMIXEL-P series, please supply power through 24V power port.

2. Control Table

The Control Table is a structure of data implemented in the device. Users can read a specific Data to get status of the device with Read Instruction Packets, and modify Data as well to control the device with WRITE Instruction Packets.

WARNING : DYNAMIXEL-P series use different Control Table from DYNAMIXEL PRO series. Please pay attention when replacing DYNAMIXEL PRO with DYNAMIXEL-P series.

2. 1. Control Table, Data, Address

The Control Table is a structure that consists of multiple Data fields to store status or to control the device. Users can check current status of the device by reading a specific Data from the Control Table with Read Instruction Packets. WRITE Instruction Packets enable users to control the device by changing specific Data in the Control Table. The Address is a unique value when accessing a specific Data in the Control Table with Instruction Packets. In order to read or write data, users must

ANEXO D

Especificações do motor Dynamixel PH54-200-S500-R

Item	Especificação
MCU	ARM CORTEX-M4 (168 [MHz], 32Bit)
Motor	BLDC (Maxon)
Taxa de transmissão	9,600 [bps] ~10.5 [Mbps]
Modos de operação	Modo de controle de torque Modo de controle de velocidade Modo de controle de posição Modo de controle de posição estendida Modo de controle PWM (modo de controle de tensão)
Peso	855 [g]
Dimensões (c x l x h)	54 x 126 x 54 [mm]
Resolução	1,003,846 [pulse/rev]
Relação de engrenagem	501.923 : 1
Folga	<6 [arcmin], 0.1 [°]
Carga radial	370 [N] (10 [mm] afastado da face do eixo)
Carga axial	130 [N]
Velocidade (sem carga)	33.1 [rev/min]
Corrente (sem carga)	1.65 [A]
Velocidade contínua	29.0 [rev/min]
Torque contínuo	44.7 [Nm]
Corrente contínua	9.3 [A]
Saída	200 [W]
Temperatura de operação	5 ~55 [°C]
Tensão operacional	24.0 [V]
Sinal de comando	Pacote digital
Tipo de protocolo	RS485 Comunicação Serial Assíncrona (8bit, 1stop, Sem paridade)
Conexão física	RS485 Barramento multiponto
ID	253 ID (0 ~252)
Corrente de repouso	40 [mA]

Tabela 26: Especificações do motor *Dynamixel PH54-200-S500-R*

Enter Search Terms



PH54-200-S500-R

- 1. Specifications
- 1. 1. Performance Graph
- 2. Control Table
- 3. How to Assemble
- 4. Maintenance
- 5. Reference

[Edit on GitHub](#)

ROBOTIS



Robot Source



GitHub



TOP

PH54-200-S500-R

1. Specifications

Item	Specifications
MCU	ARM CORTEX-M4 (168 [MHz], 32Bit)
Motor	BLDC (Maxon)
Baud Rate	9,600 [bps] ~ 10.5 [Mbps]
Operating Modes	Torque Control Mode Velocity Control Mode Position Control Mode Extended Position Control Mode PWM Control Mode(Voltage Control Mode)
Weight	855 [g]
Dimensions (W x H x D)	54 x 126 x 54 [mm]
Resolution	1,003,846 [pulse/rev]
Gear Ratio	501.923 : 1
Backlash	< 6 [arcmin], 0.1 [°]
Radial Load	370 [N] (10 [mm] away from the horn)
Axial Load	130 [N]
No Load Speed	33.1 [rev/min]
No Load Current	1.65 [A]
Continuous Speed	29.0 [rev/min]
Continuous Torque	44.7 [Nm]
Continuous Current	9.3 [A]
Output	200 [W]
Operating Temperature	-5 ~ 55 [°C]
Input Voltage	24.0 [V]
Command Signal	Digital Packet
Protocol Type	RS485 Asynchronous Serial Communication (8bit, 1stop, No Parity)
Physical Connection	RS485 Multidrop Bus

Apêndice B

Enter Search Terms



PH54-200-S500-R

- 1. Specifications
- 1. 1. Performance Graph
- 2. Control Table
- 3. How to Assemble
- 4. Maintenance
- 5. Reference



TOP

1 These specifications are calculated based on the specifications of the core motor. Please consult ROBOTIS for the long term use or special use, or else refer to the Performance Graph for general use.

**DANGER**

(May cause serious injury or death)

- Never place items containing water, flammables, and solvents near product.
- Never place fingers, arms, toes, and other body parts near product during operation.
- Cut power off if product emits strange odors or smoke.
- Keep product out of reach of children.
- Check the power's polarity before wiring.

**CAUTION**

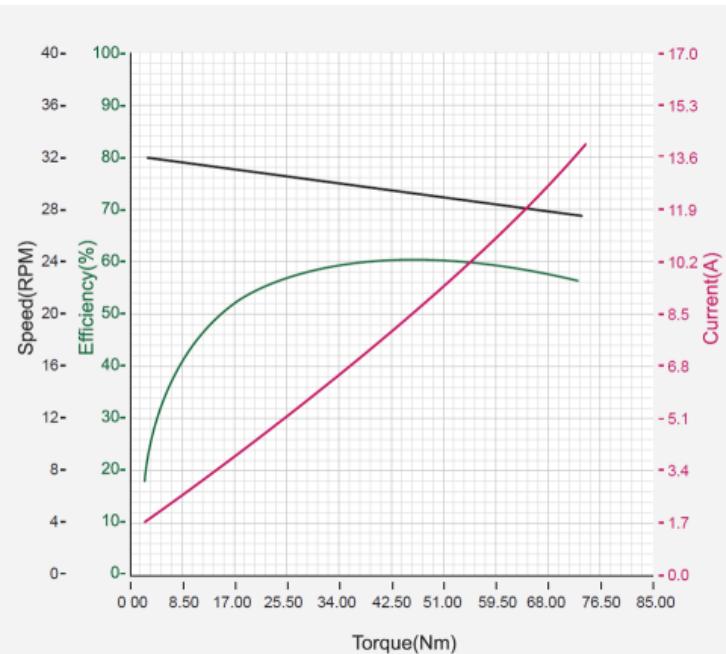
(May cause injury or damage to product)

- Do not operate the product at a temperature exceeding -5 ~ 55 [°C] range.
- Do not insert sharp blades nor pins during product operation.

**ATTENTION**

(May cause injury or damage to product)

- Do not disassemble or modify product.
- Do not drop or apply strong shock to product.

1. 1. Performance Graph[Show Enlarged Graph](#)

Apêndice B

Enter Search Terms	
PH54-200-S500-R	
1. Specifications	▼
1. 1. Performance Graph	▶
2. Control Table	▶
3. How to Assemble	▶
4. Maintenance	▶
5. Reference	▶

NOTE : The Max Torque and the Stall Torque of Performance Graph are different in measurement methods. Stall torque is a measured value of the momentary torque that it can reach. This is generally how RC servos are measured. The Performance graph is also called as N-T curves, which is measured with the gradually increasing load. The actual motor operation environment is closer to the performance graph, not stall torque method. For this reason, the performance graph is broadly used in the industrial field. Generally, Max Torque of the Performance Graph is less than the Stall Torque.



TOP

CAUTION : When supplying power

- It is recommended using ROBOTIS controller or SMPS2DYNAMIXEL.
- Do not connect or disconnect DYNAMIXEL when power is being supplied.
- In case of DYNAMIXEL PRO and DYNAMIXEL-P series, please supply power through 24V power port.

2. Control Table

The Control Table is a structure of data implemented in the device. Users can read a specific Data to get status of the device with Read Instruction Packets, and modify Data as well to control the device with WRITE Instruction Packets.

WARNING : DYNAMIXEL-P series use different Control Table from DYNAMIXEL PRO series. Please pay attention when replacing DYNAMIXEL PRO with DYNAMIXEL-P series.

2. 1. Control Table, Data, Address

The Control Table is a structure that consists of multiple Data fields to store status or to control the device. Users can check current status of the device by reading a specific Data from the Control Table with Read Instruction Packets. WRITE Instruction Packets enable users to control the device by changing specific Data in the Control Table. The Address is a unique value when accessing a specific Data in the Control Table with Instruction Packets. In order to read or write data, users must designate a specific Address in the Instruction Packet. Please refer to [Protocol 2.0](#) for more details about Instruction Packets.

NOTE : Two's complement is applied for the negative value. For more information, please refer to [Two's complement](#) from Wikipedia.

2. 1. 1. Area (EEPROM, RAM)

The Control Table is divided into 2 Areas. Data in the RAM Area is reset to initial values when the power is reset(Volatile). On the other hand, data in the EEPROM Area is maintained even when the device is powered off(Non-Volatile).

Data in the EEPROM Area can only be written to if Torque Enable(512) is cleared to '0'(Off).

2. 1. 2. Size

The Size of data varies from 1 ~ 4 bytes depend on their usage. Please check the size of data when updating the data with an Instruction Packet. For data larger than 2 bytes will be saved according to [Little Endian](#).

2. 1. 3. Access

The Control Table has two different access properties. 'RW' property stands for read and write access permission while 'R' stands for read only access permission. Data with the read only property cannot be changed by the WRITE Instruction. Read only property('R') is generally used for measuring and monitoring purpose, and read write property('RW') is used for controlling device.

Read me do Desafio 2.5

Apêndice C

Learn Git and GitHub without any code!

Using the Hello World guide, you'll start a branch, write comments, and open a pull request.

[Read the guide](#)

[Brazilian-Institute-of-Robotics / timon_hm-2-5](#) (Private)

0 stars 0 forks

Star

Watch

Code

Issues

Pull requests

Actions

Projects

Wiki

Security

Insights

master

...



leolima21 ...

on May 23



[View code](#)

README.md



BIR 2.5 CHALLENGE - Timon-HM Team

This package refers to challenge 2.5 proposed by the BIR (Brazilian Institute of Robotics). In this challenge, four DARwin-OP robots provided by Robotis was used to solve the missions.

The robots made two different missions:

1. Made a synchronized march during 2 meters.

2. Made relay race where every robot ran
during 2 meters to change the baton to the
Appendix C
next robot.

Requirements

Software

- Ubuntu 18.04 LTS
- ROS Melodic (OpenCV included)
- OpenCV



Dependencies

This original package was used to develop the Challenge-2.5 package:

ROBOTIS OP 2 ROS packages

Packages

To reproduce these results it's necessary install these packages below.

- DARwin-OP:

```
$ git clone https://github.com/HumaRobotics/darwin_gazebo.git
```

- Control packages:

```
$ sudo apt-get install ros-melodic-ros-control ros-melodic-gazebo-ros-control ros-melodic-controller-manager ros-melodic-effort-controller ros-melodic-joy-* ros-melodic-joint-state-controller ros-melodic-joint-state-publisher
```

- Hector gazebo packages:

```
$ sudo apt-get install ros-melodic-hector-gazebo ros-melodic-hector-gazebo-plugins
```

Mission 1: March

Demo

Apêndice C



Launch Gazebo world :

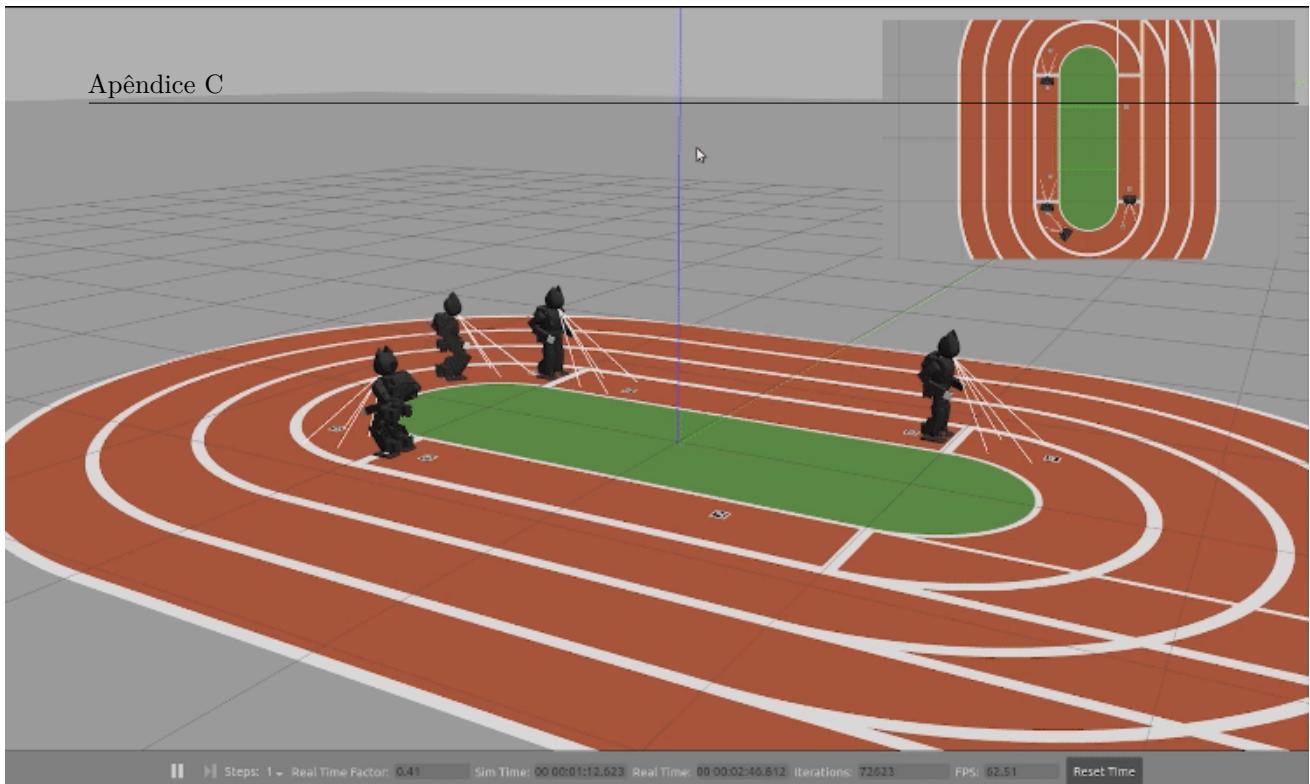
```
$ rosrun multi_robot main_march.launch
```

Execute the mission 1 : *

```
$ rosrun multi_robot march.py
```

Mission 2: Relay Race

Demo



Launch Gazebo world :

```
$ rosrun multi_robot main_race.launch
```

Execute mission 2 : *

```
$ rosrun multi_robot race.py
```

*Before executing the mission, don't forget to press play in Gazebo.

Releases

No releases published

[Create a new release](#)

Packages

No packages published

[Publish your first package](#)

Contributors 5



240

Apêndice C

Languages

- C++ 44.0%
- Python 38.7%
- CMake 17.3%

UGV SACI: Integrado com Detecção Visual e Manipulador

UGV SACI: INTEGRADO COM DETECÇÃO VISUAL E MANIPULADOR

Relatório do Projeto UGV

Autores:

Anderson Queiroz do Vale
Jéssica Lima Motta
Mateus Santos Cerqueira

Facilitadores:

Lucas Cruz da Silva
Rebeca Tourinho Lima
Tiago Pereira de Souza
Marco Antonio dos Reis

**Salvador
Bahia, Brasil**

Novembro de 2020

Título: UGV Saci: integrado com detecção visual e manipulador	
PROD. TEC. CCRoSA - 001 / 2020	Versão
Classificação: () Confidencial (X) Restrito () Uso Interno () Público	01

Informações Confidenciais - Informações estratégicas para o CCRoSA e Senai Cimatec.

Seu manuseio é restrito a usuários previamente autorizados pelo Gestor da área.

Informações Restritas - Informação cujo conhecimento, manuseio e controle de acesso devem estar limitados a um grupo restrito de pesquisadores que necessitam utilizá-la para exercer suas atividades profissionais.

Informações de Uso Interno - São informações destinadas à utilização interna por pesquisadores e parceiros.

Informações Públicas - Informações que podem ser distribuídas ao público externo, o que, usualmente, é feito através dos canais apropriados.

Dados Internacionais de Catalogação na Publicação (CIP)

Anderson Queiroz do Vale Jéssica Lima Motta Mateus Santos Cerqueira
000 Lucas Cruz da Silva Rebeca Tourinho Lima Tiago Pereira de Souza Marco Antonio dos Reis
UGV Saci: integrado com detecção visual e manipulador Salvador Bahia, Brasil Novembro de 2020
Keywords: 1. Mobile robots . 2. Manipulator. 3. Computer vision. 000

SUMÁRIO EXECUTIVO

O projeto do UGV Saci- Desafio 3.0, também conhecido como **Saci** configura-se: sob o Programa de Formação de Novos Talentos do [Centro de Competência de Robótica e Sistemas Autônomos \(CCRoSA\)](#) do Serviço Nacional de Aprendizagem Industrial, Departamento Regional da Bahia - Senai/DR/BA, sendo este o principal fomentador do programa. O presente trabalho tem como impulsionador principal a capacitação de novos pesquisadores preparados para solucionar os mais diversos problemas relacionados a robótica e sistemas autônomos.

O projeto foi considerado como início técnico no dia 04 de Novembro de 2020.

O prazo de execução planejado foi de 30 dias.

RESUMO

O Saci integra o veículo autônomo da *Clearpath Robotics Warthog* equipado com sensores (câmeras, LiDAR e GPS) e o manipulador robótico JeRoTIMON, com o propósito de transformá-lo em um robô autônomo. Este robô foi construído com o intuito de que o mesmo tivesse navegação autônoma para realizar investigação em ambiente externo e construir um mapa deste, detectasse a "bomba" escondida nesse ambiente, e realizasse o desarme da bomba através do manipulador. Para a simulação foram utilizados o software *Gazebo* e a ferramenta de visualização *Rviz*, e para o manipulador foi utilizado *MoveIt*. Este robô também possui sua versão real onde foi possível realizar testes e verificar seu desempenho em campo.

ABSTRACT

Saci is an *UGV*, called *Warthog* made by *Clearpath Robotics* equipped with sensors (cameras, LiDAR and GPS) and with a robotic manipulator JeRoTIMON. This robot was built with the intention that it had autonomous navigation to carry out research in an external environment and to build a map of it, to detect the "bomb"hidden in that environment, and to disarm this "bomb"through the manipulator. For the simulation, the software *Gazebo* and the visualization tool *Rviz* were used, and for the manipulator *MoveIt* was used. This robot also has its real version where it was possible to carry out tests and verify its performance in the field.

LISTA DE FIGURAS

Figura 1:	Frames para o manipulador Elbow.	16
Figura 2:	Simulação do desafio no <i>Gazebo</i>	17
Figura 3:	Arquitetura geral do sistema.	20
Figura 4:	Modelo esquemático.	21
Figura 5:	Plataforma veicular Warthog, Clearpath Robotics.	22
Figura 6:	NUC Xi7 (frontal), Intel.	23
Figura 7:	NUC Xi7 (traseira), Intel.	23
Figura 8:	Lidar VLP-16, Velodyne.	24
Figura 9:	GPS SMART6-L, NovAtel.	25
Figura 10:	Câmera Basler.	26
Figura 11:	IMU SEN-14001, Sparkfun.	26
Figura 12:	<i>Switch</i> TL-SG105, TP-Link.	27
Figura 13:	Estrutura analítica do protótipo.	27
Figura 14:	Diagrama da localização.	28
Figura 15:	Diagrama representando o mapeamento.	29
Figura 16:	Diagrama representando o planejamento.	31
Figura 17:	Diagrama das interações das funcionalidades com a Navegação.	32
Figura 18:	Representação da bomba.	33
Figura 19:	Identificação com filtro de cor.	33
Figura 20:	Identificação da bomba.	34
Figura 21:	Fluxograma representando a detecção.	35
Figura 22:	Representação da identificação da bomba.	36
Figura 23:	Diagrama da manipulação.	37
Figura 24:	Integração simplificado das funcionalidades.	37
Figura 25:	Vista frontal e lateral da plataforma móvel e seus componentes.	39
Figura 26:	Basler ace acA4600.	40
Figura 27:	Lente Kowa LM8HC.	40
Figura 28:	Manipulador.	41
Figura 29:	Análise a partir do Diagrama de Ishikawa.	46
Figura 30:	Especificações da câmera Basler ace.	57

LISTA DE TABELAS

Tabela 1:	Parâmetros dos links para o manipulador Elbow.	16
Tabela 2:	Especificações da câmera Basler ace.	25
Tabela 3:	Lições aprendidas	45

LISTA DE SÍMBOLOS E ABREVIATURAS

CCRoSA Centro de Competência de Robótica e Sistemas Autônomos

OMPL Open Motion Planning Library

OpenCV Open Source Computer Vision

LiDAR Light Detection And Ranging

URDF Unified Robot Description Format

NUC Next Unit of Computing

ROS *Robot Operating System*

GPS *Global Positioning System*

IMU *Inertial Measurement Unit*

UGV *Unmanned Ground Vehicle*

DoF *Degrees of Freedom*

SLAM *Simultaneous Localization And Mapping*

LIDAR *Light Detection And Ranging*

UART *Universal Asynchronous Receiver/Transmitter*

CPU *Central Process Unit*

SOTA *State Of The Art*

EKF Extended Kalman Filter

RRT Rapidly-exploring Random Tree

SUMÁRIO

1 INTRODUÇÃO	11
1.1 Objetivos	11
1.2 Justificativa	11
1.3 Organização do relatório	12
2 CONCEITO DO SISTEMA	13
2.1 Parâmetros básicos	13
2.1.1 Requisitos do cliente	13
2.1.2 Requisitos técnicos	14
2.1.3 Estudo do estado da arte	14
2.1.4 Ambiente de operação	17
3 DESENVOLVIMENTO DO SISTEMA	19
3.1 Descrição do sistema	19
3.1.1 Arquitetura geral	19
3.2 Modelo Esquemático	20
3.3 Especificação de componentes	21
3.3.1 UGV - Warthog	21
3.3.2 Computador central (NUC)	22
3.3.3 <i>LiDAR</i>	24
3.3.4 Sistema de georreferenciamento	24
3.3.5 Câmera Basler ace acA4600	24
3.3.6 IMU	25
3.3.7 <i>Switch</i>	26
3.3.8 Estrutura analítica do protótipo	27
3.4 Funcionalidades	27
3.4.1 Localização	28
3.4.1.1 Dependências	28
3.4.1.2 Saídas	29
3.4.2 Mapeamento	29

3.4.2.1 Dependências	30
3.4.2.2 Saídas	30
3.4.3 Planejamento	30
3.4.3.1 Dependências	30
3.4.3.2 Saídas	31
3.4.4 Navegação	31
3.4.4.1 Dependências	32
3.4.4.2 Saídas	32
3.4.5 Detecção	32
3.4.5.1 Dependências	33
3.4.5.2 Saídas	34
3.4.5.3 Dependências	34
3.4.5.4 Saídas	34
3.4.6 Manipulação	35
3.4.6.1 Dependências	35
3.4.6.2 Saídas	36
3.5 Arquitetura de software	36
4 IMPLEMENTAÇÃO	39
4.1 Estrutura física	39
4.1.1 Câmera	39
4.1.2 LiDAR	40
4.1.3 Warthog	40
4.1.4 Manipulador	41
5 RESULTADOS E DISCUSSÕES	43
6 GESTÃO DO CONHECIMENTO	45
6.1 Lições aprendidas	45
6.2 Análise através do Diagrama de Ishikawa	45
6.3 Guia de uso	46
7 CONCLUSÃO	49

REFERÊNCIAS	51
ANEXO A Warthog	53
ANEXO B Câmera Basler ace acA4600	57
ANEXO C Lente Kowa LM8HC	67
ANEXO D Velodyne LiDAR Puck LITE	69

1 INTRODUÇÃO

A robótica é um ramo da tecnologia que engloba área de eletrônica, mecânica e computação, com graus de teoria de controle, microeletrônica, inteligência artificial, fatores humanos e de produção (PIMENTA, 2009). A busca por robôs autônomos equipados com sensores e atuadores, como por exemplo manipuladores robóticos, para realizar atividades que colocam em risco a vida de seres humanos, seja pelo desempenho de uma atividade perigosa ou por o local ser de difícil acesso, vem se tornando o foco de pesquisas na área de robótica.

Este relatório descreve o processo de modelagem e construção de um veículo autônomo desenvolvido no Centro de Competência em Robótica e Sistemas Autônomos (CCRoSA) do SENAI CIMATEC e é destinado ao programa de formação Novos Talentos. São descritas as etapas de concepção, simulação, implementação física e testes.

1.1 Objetivos

O objetivo do projeto é desenvolver um **UGV** também conhecido como veículo terrestre não tripulado. E este deve possuir navegação autônoma, capaz de explorar um ambiente externo onde este possui marcadores visuais para limitar a exploração do robô, identificar uma bomba por meio de uma câmera RGB e proceder com a função de desarmá-la a partir de um braço robótico. Para isso, os objetivos específicos são:

- Realizar estudo do Estado da Arte(**SOTA**) sobre **UGVs**, detecção visual de objetos e integração com manipuladores.
- Modelagem para a simulação.
- Parametrizar o pacote de reconhecimento de marcadores visuais.
- Desenvolver um pacote de configuração do manipulador no *MoveIt*.
- Desenvolver o código para realizar a missão.
- Realizar a simulação em software.
- Implementar a versão física do protótipo.
- Realização de ajuste de parâmetros e testes.

1.2 Justificativa

A busca por meios de realizar atividades em locais que a presença do ser humano torna-se difícil, arriscado e até mesmo impossível, com precisão e eficiência, têm se tornado

prioridade. Além da capacidade do robô de interagir com o ambiente a partir da captura e análise de estímulos visuais ([LEITE, 2005](#)).

Há uma demanda crescente, porém falta profissionais habilitados à desenvolver pesquisas e aplicações na área da robótica. O presente trabalho tem como impulsionador principal a capacitação de novos pesquisadores preparados para solucionar os mais diversos problemas relacionados a robótica e sistemas autônomos.

Este projeto traz a utilização de veículos autônomos que sejam capazes de identificar marcadores visuais para saber o seu limite de exploração e realizar a tarefa de desarmar uma bomba. Espera-se que este projeto seja continuado e que seus resultados sejam compartilhados na comunidade científica, contribuindo para a construção de outros *UGVs* com características e/ou objetivos semelhantes.

1.3 Organização do relatório

Este documento visa agrupar todos os conteúdos utilizados para o desenvolvimento desse projeto desde a fase conceitual até a implementação e testes. O relatório está organizado em oito capítulos, sendo este de introdução e descrição da justificativa/motivação dos objetivos e da organização do documento.

No capítulo [2](#), Conceito do Sistema, são descritos parâmetros básicos do projeto, dentre eles os requisitos do cliente, requisitos técnicos e o estudo do estado da arte.

O capítulo [3](#), Desenvolvimento do Sistema, apresenta a descrição do sistema onde serão apresentados a arquitetura geral, especificações técnicas, o ambiente de operação do manipulador e a estrutura analítica do protótipo. Além disso, trará as especificações funcionais que compõem o sistema, sua arquitetura de software e o que foi desenvolvido para simulação.

O capítulo [4](#), Implementação, explica a construção física do *UGV*. São expostos seus parâmetros de configuração e sua estrutura.

O capítulo [5](#), Resultados, são apresentados os resultados alcançados nesse projeto.

No capítulo [6](#), Gestão do Conhecimento, é feito um estudo sobre as lições aprendidas, foi realizada uma análise usando o Diagrama de Ishikawa, além de apresentar o guia de uso.

Por fim, o capítulo [7](#) apresenta a conclusão do relatório.

2 CONCEITO DO SISTEMA

A norma técnica (ISO-8373, 2012) criada para padronizar o vocabulário referente aos robôs e dispositivos robóticos operando em ambientes industriais e não industriais, define o robô autônomo como uma máquina na qual o mecanismo programável, geralmente, em um ou mais eixos com graus de liberdade, hábil para execução de tarefas usando estado físico e/ou sensores sem intervenção humana. Em outras palavras, robô autônomo é um equipamento programável baseado em atuadores e sensores, projetado para realizar uma variedade de atividades, assim como realização de diversos processos industriais (ISO-8373, 2012).

O desenvolvimento deste projeto está situado na concepção de plataformas móveis que podem ser utilizadas na navegação em ambientes abertos, detecção e desarmamento de um tipo específico de bomba. Esta plataforma terá modularidade para diversas outras aplicações, tornando possível adaptá-la para ambientes externos e internos devido a diversidade de sensores inseridos no seu corpo. Além dos sensores, o veículo possui um manipulador robótico para realizar as tarefas de corte do fio ou pegar um objeto.

Neste capítulo serão tratados os requisitos solicitados pelo cliente, os requisitos técnicos do projeto e o estudo do estado da arte sobre [UGV](#).

2.1 Parâmetros básicos

Nesta seção serão detalhados os requisitos solicitados pelo cliente, referentes ao projeto, no qual a tarefa precisa ser realizada em um ambiente externo. Além disso, são exibidos os requisitos técnicos que tratam das especificações do sistema e uma breve revisão teórica de conceitos relacionados ao [UGV](#).

2.1.1 Requisitos do cliente

Requisitos pré-determinados pelo cliente consistem em exigências de funcionamento que devem ser observadas ao final do projeto, para que se considere um sucesso a concepção deste. Para tal, foram determinadas algumas características desejáveis no projeto:

1. A plataforma móvel usada será o *Warthog*;
2. O *Warthog* sempre será o principal meio para conclusão da solução;
3. O desenvolvimento da solução deve ser implementado no computador externo à plataforma;
4. A equipe usará a câmera RGB (Basler), [LiDAR](#) (Velodyne) e [SLAM](#) (EKF SLAM);

5. O desenvolvimento do **SLAM** deve ter cinco características principais: mapeamento, localização, planejamento de trajetória, destino alvo e desvio de obstáculos;
6. A busca pela bomba e desativação da plataforma móvel devem ser realizados de forma autônoma.

2.1.2 Requisitos técnicos

Os requisitos técnicos de um projeto são especificações necessárias para o funcionamento esperado do projeto. Podem ser sobrepostos aos requisitos do cliente em caso de conflito entre o esperado pelo cliente e o necessário para que o projeto seja bem sucedido, objetivando manter o projeto o mais eficiente dentro do escopo planejado. Os requisitos foram:

1. O manipulador utilizado para a missão será o do projeto JeRoTimon;
2. O *Warthog* deve se localizar no mundo;
3. A plataforma móvel deve identificar uma região que não é de interesse a partir da detecção de um marco fiducial;
4. Deve ser feito o uso do *framework ROS*;
5. Integrar um minicomputador (NUC) ao sistema para realizar o processamento;
6. O manipulador de desativação da bomba será posicionado em frente ao robô móvel;
7. A plataforma deve identificar a “bomba” a partir de uma câmera RGB e desabilitá-la, aproximando o *end effector* do fio com a coloração específica;

2.1.3 Estudo do estado da arte

Com a necessidade da automatização de tarefas repetitivas e atividades de alto risco, a implementação de robôs autônomos tem crescido exponencialmente. (**ROBOS-AUTONOMOS, s.d.**).

A partir da definição dada por (**MATARIĆ et al., 2007**) sobre um robô móvel autônomo, é possível ter uma noção dos componentes essenciais que o compõe. Os sensores são fundamentais para que o robô seja capaz de perceber o ambiente em que ele se encontra. Da mesma forma, atuadores são fundamentais para que o robô consiga atuar sobre este ambiente. Por fim, para que o robô seja autônomo, é necessário um sistema de controle não supervisionado.

De acordo com (**TILBURY; ULSOY, 2010**), os *UGVs* possuem muitas potenciais aplicações, desde uso em ações militares como por exemplo descarte de material bélico

explosivo, vigilância e reconhecimento em combate, até seu uso industrial e doméstico, como a limpeza de pisos, ou tarefas especiais como operações de resgate.

Para realizar as operações de forma autônoma o *UGV* precisa de métodos de planejamento de caminho e estes possuem duas classificações: método de planejamento global e método de planejamento local (SEDIGHI et al., 2004). Para o método de planejamento global, o *UGV* requer conhecimento sobre o ambiente e assume que o terreno é estático. Em contrapartida, para o planejamento do caminho local, o ambiente é parcialmente conhecido ou até mesmo completamente desconhecido. E para este método o *UGV* utiliza a informação proveniente de sensores que enviam esses sinais em tempo real durante sua navegação local (WANG et al., 2005). Os sensores também auxiliam ao veículo autônomo a se movimentar no ambiente sem colidir com os obstáculos encontrados.

Para que esses robôs possuem navegação autônoma uma das técnicas utilizadas é o uso da visão computacional, onde esta pode ser definida como um conjunto de técnicas e teorias com o objetivo de obtenção de informação por meio de imagens, sendo de modo geral, o estudo da capacidade das máquinas em visualizar o mundo real (tridimensional), por meio de sensores, câmeras e outros dispositivos que extraem informação dos ambientes (ALVES, 2005). A identificação de imagem monocromática pode ser feita a partir de uma função bidimensional, $f(x,y)$, onde x e y são coordenadas espaciais no plano e o número de pixels e sua intensidade em tons de cinza variam de acordo com a coloração imagem formada no plano. (GONZALEZ; WOODS, 2000). Um objeto pode ser encontrado de diversas maneiras, a partir da identificação de características específicas como: cor, geometria, tamanho e composição do material. Para realizar o controle da tarefa de identificação, a "bomba" é identificada a partir de um algoritmo escrito em *Python* e que utiliza bibliotecas do *OpenCV*.

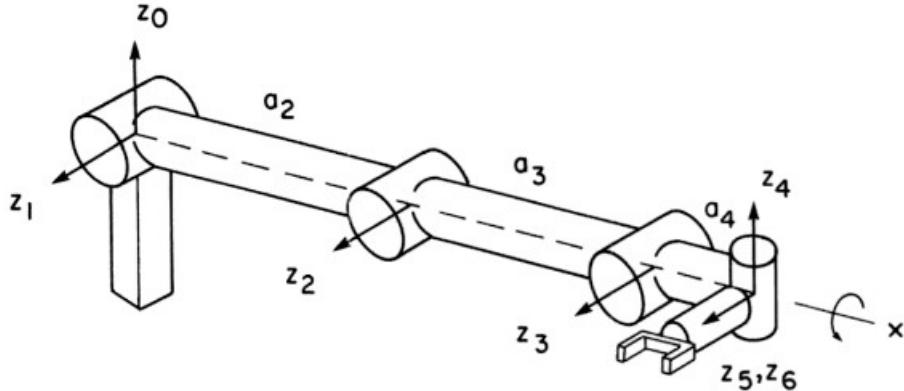
Na parte da atuação para o desarme da "bomba" foi utilizado um manipulador robótico e muitas pesquisas têm sido realizadas nesta área. Em (HERNANDEZ-MENDEZ et al., 2017) é descrito o desenvolvimento de um manipulador robótico com 3 *DoF* e dois dedos independentes. Este robô foi desenvolvido com o propósito de manipular objetos, cujas localizações são conhecidas, e transportá-los de uma localidade para outra utilizando *ROS*. Os autores utilizaram o *MoveIt* para tratar do planejamento de trajetória e o *Rviz* como ferramenta de visualização. Além disso, foi desenvolvido um controlador de posição e força para o *end effector*¹ durante o processo de *pick and place*². Os experimentos realizados trouxeram bons resultados para o que foi proposto, sendo ressaltada a necessidade de adicionar um sistema de visão que permita identificar e localizar o objeto alvo.

O estudo da dinâmica e da cinemática apresenta a síntese do projeto de um dispositivo,

¹ Na robótica, um endeffector é o dispositivo no final de um braço robótico, projetado para interagir com o meio ambiente.

² Sequência de movimentos na qual o manipulador robótico pega determinado objeto e o transfere a uma pose alvo.

Figura 1: Frames para o manipulador Elbow.



Fonte: ([PAUL, 1981](#)).

expressando matematicamente as relações de movimento de um mecanismo na execução de determinada tarefa.

Para a análise cinemática, utiliza-se os parâmetros de Denavit-Hartenberg (D-H). Este realiza uma cadeia cinemática espacial através da fixação de sistemas de referência aos elos ([PAUL, 1981](#)). Esta notação de Denavit-Hartenberg é uma ferramenta comumente utilizada para sistematizar a descrição cinemática de sistemas mecânicos articulados com n graus de liberdade ([JR; DENAVIT; HARTENBERG, 1964](#)). Na Tabela 1 tem-se a cadeia cinemática através dos parâmetros D-H para o manipulador da Figura 1, nesta tabela tem-se o elo (*link*), θ refere-se ao ângulo de rotação no eixo z e α rotação no eixo x, a refere-se a translação ao longo do eixo de rotação x e d translação ao longo do eixo z. Na cinemática direta é possível observar que a partir dos ângulos das juntas do manipulador encontra-se a posição e orientação da ferramenta com relação a estação de trabalho. Contudo, quando projeta-se um manipulador, é fundamental realizar os cálculos da cinemática inversa. Nesta, por sua vez, são obtidos dados de posição e orientação da ferramenta com relação a estação de trabalho e calcula-se os ângulos das juntas ([CRAIG, 2012](#)).

Tabela 1: Parâmetros dos links para o manipulador Elbow.

Link	Variável	α	a	d	$\cos \alpha$	$\sin \alpha$
1	θ_1	90°	0	0	0	1
2	θ_2	0°	a_2	0	1	0
3	θ_3	0°	a_3	0	1	0
4	θ_4	-90°	a_4	0	0	-1
5	θ_5	90°	0	0	0	1
6	θ_6	0°	0	0	1	0

Fonte: ([PAUL, 1981](#)).

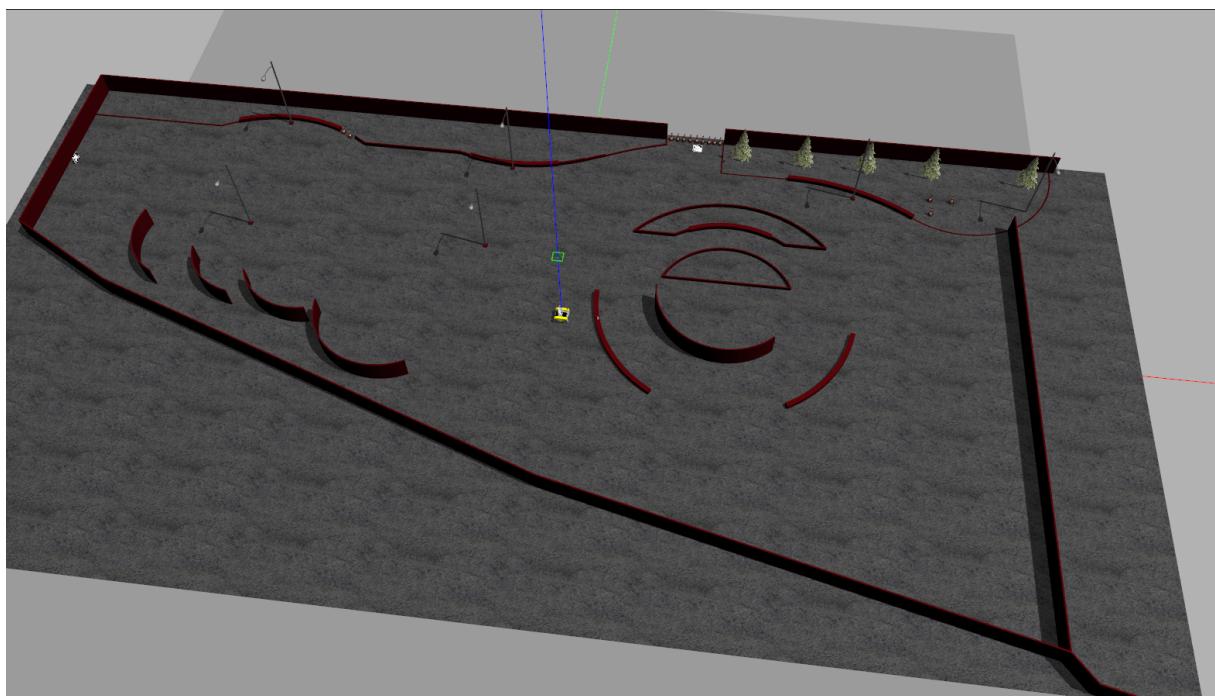
O controle da cinemática e o planejamento da movimentação evitam que o manipulador colida com outros elementos do seu espaço de trabalho. Para tal é necessário aplicar técnicas de controle e utilização de sensores, estes últimos fazem com que o robô tenha capacidade de percepção coletando dados tanto internos, sobre seu sistema mecânico, quanto externos, no meio ambiente em qual está inserido (SCIAVICCO; SICILIANO, 2012).

A dinâmica do manipulador dependerá do seu design mecânico, logo esta definição irá variar de um manipulador para outro, e depende também de já haver definido as formulações cinemáticas (NORTON, 2001). Estas por sua vez, fornecem os elementos geométricos de um sistema que são a base para se obter as energias que compõem as equações dinâmicas.

2.1.4 Ambiente de operação

O ambiente para a realização do desafio, onde serão incluídos o *Warthog*, equipado com os sensores e o manipulador, os *ArUcos* e a "bomba", é no ambiente externo entre os prédios 3 e 4 do SENAI CIMATEC. Na Figura 2 tem-se o ambiente de operação simulado no Gazebo para que possam ser realizados testes de navegação, exploração, detecção visual, reconhecimento de marcadores fiduciais e atuação do manipulador, e realização de ajustes e validação dos parâmetros aplicados.

Figura 2: Simulação do desafio no *Gazebo*



Fonte: Autoria própria

3 DESENVOLVIMENTO DO SISTEMA

Nesta seção serão detalhados o que foi utilizado no desenvolvimento do **UGV**, abordando os sistemas que o compõem em *software* e em *hardware*. Será mostrado a arquitetura geral do sistema, as especificações dos componentes e as principais funcionalidades.

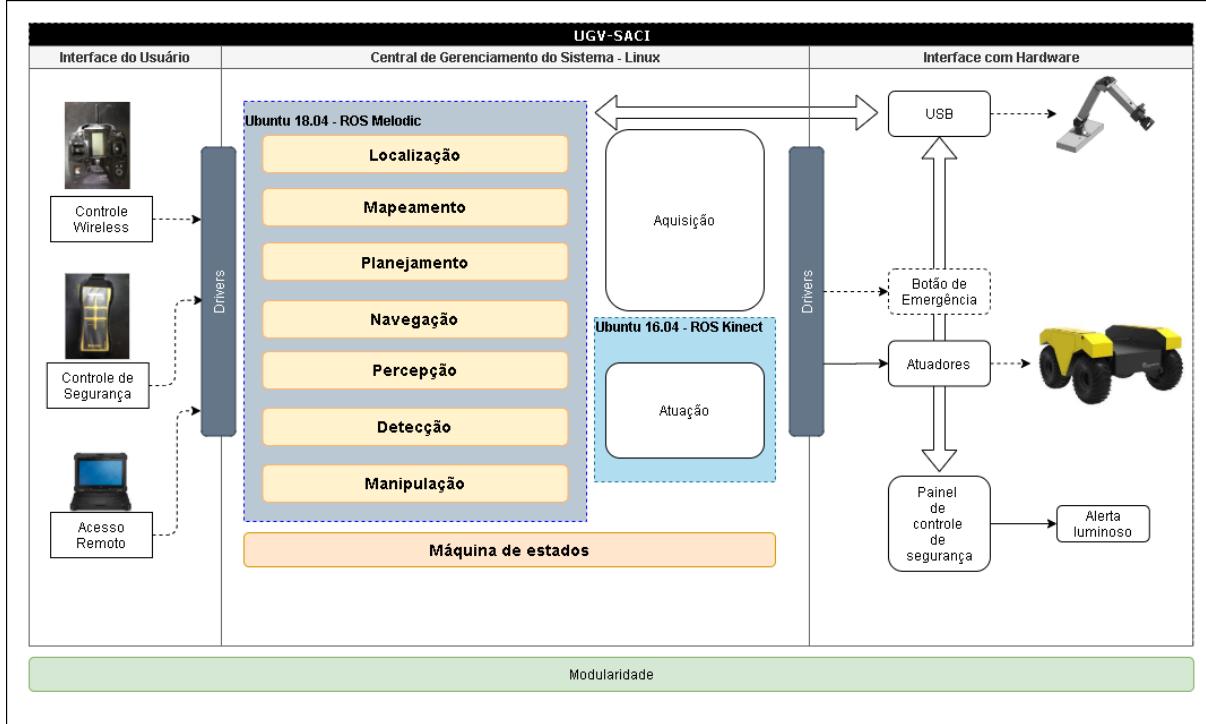
3.1 Descrição do sistema

O **UGV** é um veículo desenvolvido para atender aos desafios relacionados ao desarmamento de bombas. Esses desafios incluem navegação autônoma *outdoor*, detecção e estimativa da posição do objeto e manipular o objeto através de um manipulador robótico. Os pacotes que constituem este robô foram concebidos através do *software* de simulação *Gazebo*, da ferramenta de visualização *Rviz* e do *framework* **ROS**. O uso dessas ferramentas possibilitam que uma grande variedade de atividades, que venham a ser realizadas no mundo real, tenham sido previamente testadas em um ambiente simulado.

3.1.1 Arquitetura geral

A Figura 3 mostra a arquitetura geral do sistema que faz um panorama geral da relação entre interface de usuário, a central de gerenciamento do sistema e o *hardware*. A interface de usuário é responsável pelo contato direto com o usuário, sendo possível intervir na movimentação através do *joystick*, em que você pode movimentar o **UGV**, ou através do controle de segurança, no qual é possível impor um estado de parada de movimento do **UGV**. Por um *notebook* é possível realizar o acesso remoto ao sistema, podendo assim monitorar as tarefas realizadas de forma autonoma pelo **UGV**.

Figura 3: Arquitetura geral do sistema.



Fonte: Autoria própria.

A central de gerenciamento do sistema possui duas variações da distribuição *Linux - Ubuntu*. Para o *Ubuntu 18.04* foi utilizado uma [Next Unit of Computing \(NUC\)](#) com o [ROS Melodic](#). Essa [NUC](#) é responsável pela aquisição de alguns sensores e pelo processamento das funcionalidades presentes na arquitetura (Figura 3.) Para o *Ubuntu 16.04* foi utilizado o próprio sistema de processamento do [UGV](#). Ele é encarregado na atuação dos motores e na aquisição dos sensores do próprio fabricante ([Global Positioning System \(GPS\)](#), Encoder e Resolver). Essa decisão de adicionar uma [NUC](#) ao [UGV](#) foi para reduzir o processamento do próprio sistema e a possibilidade de adicionar funcionalidades que precise de um poder de processamento maior. A máquina de estado é responsável em gerenciar as funcionalidades, passando para atuação do manipulador ou do [UGV](#).

3.2 Modelo Esquemático

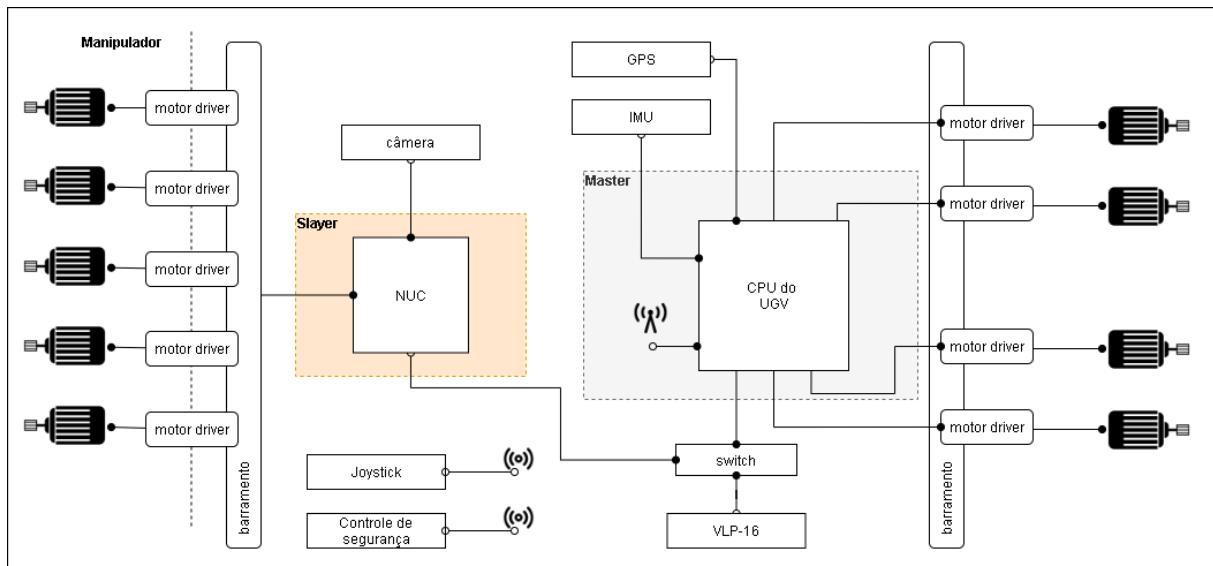
O modelo esquemático, representado na Figura 4, mostra os componentes utilizados no desenvolvimento do projeto, com intuito de representar a forma como estão conectados.

A ligação entre a [NUC](#) e a [Central Process Unit \(CPU\)](#) do [UGV](#) é conectada através do *switch*. A câmera e o manipulador robótico é ligado diretamente na [NUC](#). O [Light Detection And Ranging \(LIDAR\)](#), que é representado pelo VLP-16 na Figura 4, é conectado no *switch*, mas seu processamento é feito exclusivamente na [NUC](#). O mesmo ocorre para o

GPS e a *Inertial Measurement Unit* (IMU), a CPU do UGV coleta esses dados e lança-os nos seus respectivos tópicos para serem utilizados posteriormente pela NUC.

A unidade de processamento do UGV fica aguardando os comandos de movimentação que serão enviados pela NUC ou pelo joystick. O joystick envia informações para um receptor de radiofrequência conectado no computador central. Se o joystick estiver ligado e a NUC enviar comandos de movimento do UGV, irá ocorrer conflito com o joystick com o slayer e o UGV não vai conseguir se movimentar.

Figura 4: Modelo esquemático.



Fonte: Autoria própria.

3.3 Especificação de componentes

A especificação de componentes refere-se a uma relação e descrição dos componentes de hardware e software que constituem o sistema. Os componentes selecionados para compor o sistema são descritos a seguir.

3.3.1 UGV - Warthog

Para que os sensores necessários ao mapeamento sejam integrados e tenham mobilidade sobre uma plataforma robusta, compacta e resistente a condições adversas como impacto, poeira e umidade, foi selecionado um veículo modelo *Warthog* (Figura 5) do fornecedor *Clearpath Robotics*. Esse veículo apresenta interface para comunicação através de uma API para integração ao framework ROS, oferece recursos para controle que possibilitam a teleoperação da plataforma, possui capacidade de carga de 272kg e dimensões adequadas (1.52 m x 1.38 m x 0.83 m) para suportar a instrumentação que equipará o protótipo. Este

modelo possui autonomia elétrica de 2.5 horas, atendendo à necessidade de 2 horas de operação, ângulo de escalada de até 45°, o que possibilita ultrapassar pequenos obstáculos, capacidade de carga de até 272 Kg, além de grau de proteção contra poeira e água IP67 compatíveis com o projeto.

A plataforma inclui freios, suspensão articulada passiva geométrica, saídas de tensão regulada para 5/12/24 V, saída não-regulada 48V, computador embarcado Mini-ITX i5 com 8GB de RAM e 120 SSD, Ubuntu 16.04 e ROS Kinect instalados e configurados para operar o veículo, modelo de simulação 3D para o Gazebo, além de um GPS e uma unidade inercial.

Figura 5: Plataforma veicular Warthog, Clearpath Robotics.



Fonte: Clearpath Robotics.

3.3.2 Computador central (NUC)

Dado o volume de informações necessárias para as atividades de navegação e controle dos diversos equipamentos ligados ao UGV, foi definido um computador central portátil para esse gerenciamento.

Esse computador é o modelo Xi7 da Intel (Figuras 6 e 7). Suas principais configurações são:

- Processador Intel Core i5-5250U com frequência de 1.6 GHz (com possibilidade de trabalhar até 2.7 GHz) e 3 M de memória cache;
- 16 GB de memória RAM DDR3L com frequência de 1333 MHz;
- Placa de vídeo integrada HD Intel 4200 com resolução máxima de 2560x1600@60Hz;
- 4 portas USB (2 portas 2.0 e 2 portas 3.0);
- Saída de vídeo HDMI;
- Saída de áudio e entrada de microfone 3.5 mm;
- Placa de rede que suporta cabeamento RJ45 e Wi-Fi com velocidade de 10/100/1000 Mbps;
- Armazenamento interno via SSD de 256 GB.

Figura 6: NUC Xi7 (frontal), Intel.



Fonte: Intel.

Figura 7: NUC Xi7 (traseira), Intel.



Fonte: Intel.

3.3.3 *LiDAR*

Dispositivo óptico responsável pela detecção de obstáculos no ambiente. O modelo selecionado para esta aplicação foi o VLP-16, da fabricante *Velodyne*, conforme Figura 8. Possui um alcance de até 100 m podendo realizar um *scan* de 360°. Possui resolução angular de 2°, com um campo de visão de ± 15°, 16 canais, tem massa equivalente a 830 g e dimensões de 71 mm de altura e 88,9 mm de diâmetro. Ele trabalha com uma frequência de escaneamento de 5 - 20 Hz e possui classe de laser 1, ou seja, inofensivo aos olhos humanos.

Figura 8: Lidar VLP-16, Velodyne.



Fonte: Velodyne.

3.3.4 Sistema de georreferenciamento

No processo de mapeamento da rota de trabalho é necessário o registro da geolocalização do UGV para que ele possa confirmar a posição correta da atividade de exploração e estimar melhor sua localização. Dito isto, o **GPS** é responsável pelo fornecimento de informações sobre o posicionamento do robô.

O dispositivo utilizado foi o modelo *SMART6-L* da fabricante *NovAtel*, como mostra a Figura 9. É um módulo **GPS** com dimensões de 81 mm de altura e 155 mm de diâmetro. Possui uma frequência de leitura de 20 Hz, precisão de 15 cm e tem um peso de 470 g. Sua conectividade é *Universal Asynchronous Receiver/Transmitter* (**UART**), podendo ser alimentado por uma tensão regulada de 12 V.

3.3.5 Câmera Basler ace acA4600

A ficha resumo contendo as principais especificações da câmera *Basler ace acA4600* está disposta na Tabela 2. A Figura 10 mostra a câmera *Basler*. Essa câmera permite customizar sua imagem a partir de um *software* desenvolvido pelo próprio fabricante, chamado *Pylon*. A partir desse *software* é possível reduzir o tamanho da imagem, definir a



Figura 9: GPS SMART6-L, NovAtel.

escala de cor utilizada e atribuir algumas outras funcionalidades à câmera. Dessa forma, a câmera *Basler* pode ser utilizado em diversas aplicações, devido a sua customização para se adaptar ao tipo de processamento utilizado.

Tabela 2: Especificações da câmera Basler ace.

Item	Especificação
Especificação de câmera	acA4600
Resolução	4608 × 3288 [px]
Modelo do sensor	On-Semi MT9F002
Taxa de captura (padrão)	10 [fps]
Tamanho de pixel	1,4 × 1,4 μ m
Alcance dinâmico	62,1 [dB]
Círculo máximo de imagem	Formato óptico 1/2,3"
Encaixe de lente	C-Mount
Protocolo de comunicação	USB 3.0 (5 [Gbit/s])
Porta de comunicação	USB
Tensão de operação	5 [V]
Temperatura de operação	0 ~ 50 [$^{\circ}$ C]
Massa	< 80 [g]
Dimensões (L x A x P)	29,3 x 29 x 29 [mm ³]
Potência necessária	~ 2,5 W a 5 V _{DC} , máximo de 2,8 W

Fonte: Basler.

3.3.6 IMU

A **IMU** (Figura 11) é um dispositivo eletrônico capaz de medir e informar valores de aceleração linear, velocidade angular e rotação do robô. O modelo escolhido foi o SEN-14001, da fabricante *Sparkfun*.

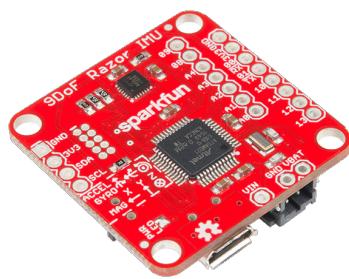
O módulo SEN-14001 combina um microprocessador SAMD21 com um sensor MPU-9250, de 9 graus de liberdade. O MPU-9250 possui três sensores de 3 eixos, um acelerômetro,

Figura 10: Câmera Basler.



Fonte: Basler.

Figura 11: IMU SEN-14001, Sparkfun.



Fonte: Sparkfun.

giroscópio e magnetômetro, que permitem detectar aceleração linear, velocidade de rotação angular e vetores de campo magnético.

O microprocessador integrado, SAMD21 da Atmel, é um microcontrolador ARM Cortex-M0+ de 32 bits. O modelo também inclui um soquete para cartão micro SD e chave de controle de energia.

3.3.7 *Switch*

É um dispositivo capaz de distribuir comunicação *ethernet* com os diversos componentes do sistema. O modelo TL-SG105 da TP-Link (Figura 12) foi escolhido por apresentar um número de portas suficiente e velocidade de comunicação necessária para a aplicação. Ele possui 5 portas de conexão com velocidades de 10 Mbps, 100 Mbps e 1000 Mbps. Seus protocolos e padrões operacionais são: IEEE 802.3 / 802.3u / 802.3ab / 802.3x CSMA / CD. Tem 400 gramas e possui dimensões de 99.8 mm x 98 mm x 25 mm, podendo operar em temperaturas de 0 °C até 40 °C e umidade entre 10% a 90% (sem condensação).

Figura 12: *Switch* TL-SG105, TP-Link.

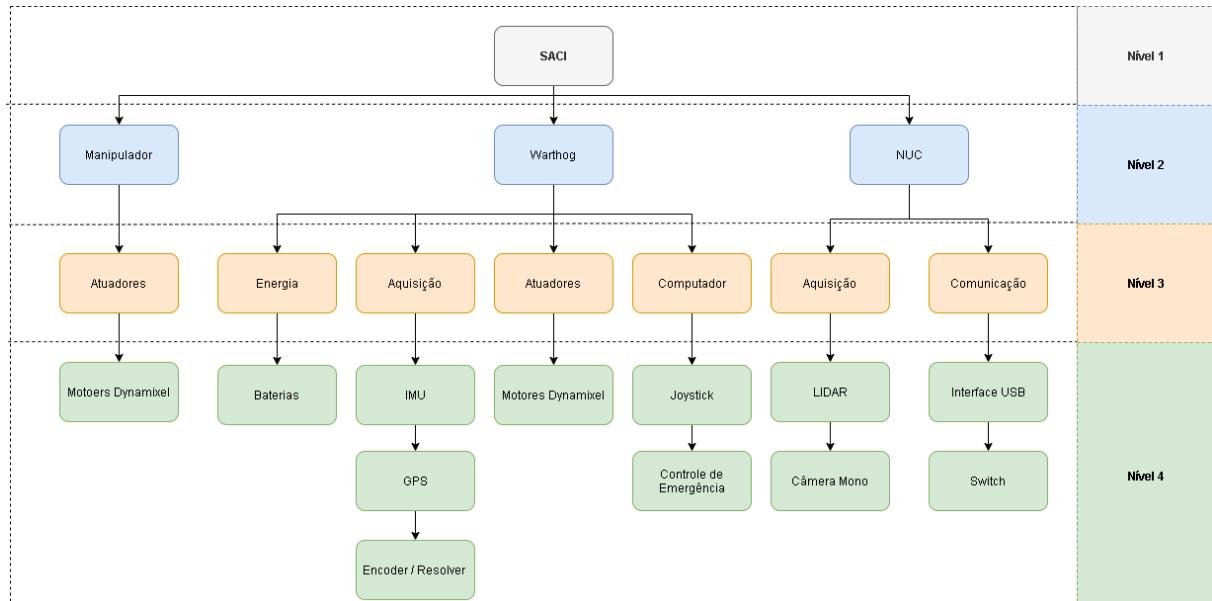


Fonte: TP-Link.

3.3.8 Estrutura analítica do protótipo

A estrutura analítica do protótipo mostrada na Figura 13 exibe as relações sistemáticas entre as partes que compõem o manipulador, o *warthog* e a NUC. A estrutura hierárquica possui quatro níveis: o primeiro, referente a aplicação principal SACI; o segundo nível mostra os principais sistemas da aplicação; o terceiro, mostra os principais subsistemas de cada módulo; o quarto nível mostra os componentes que cada sistema utiliza.

Figura 13: Estrutura analítica do protótipo.



Fonte: Autoria própria.

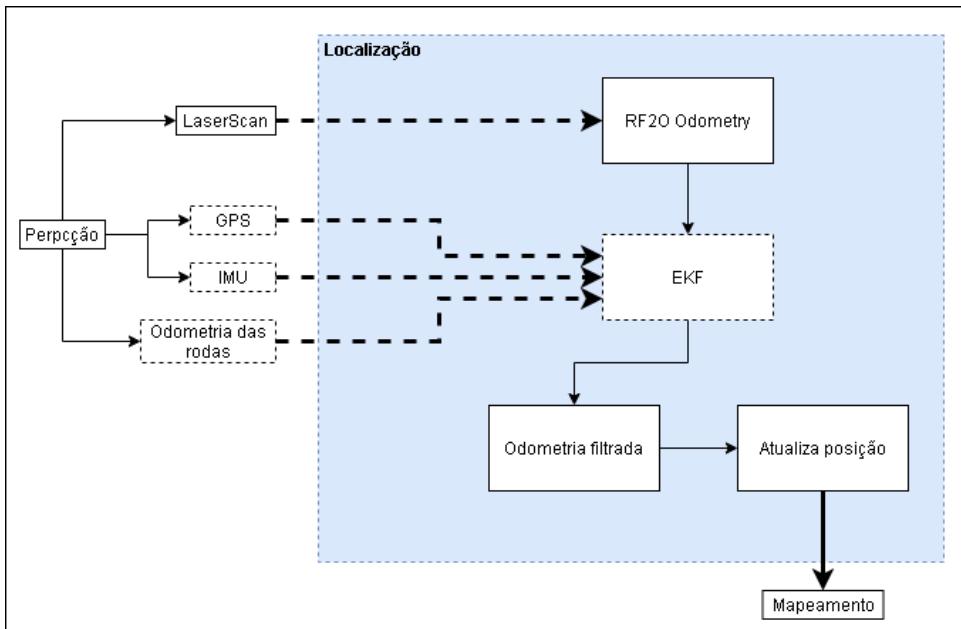
3.4 Funcionalidades

Nesta sessão serão detalhados as funcionalidades desenvolvidas no projeto. As funcionalidades presentes no projeto são: Localização, Mapeamento, Planejamento, Navegação, Percepção, Detecção, Manipulação e Atuação.

3.4.1 Localização

A localização engloba o constante monitoramento da posição e orientação do UGV dentro do ambiente no qual está contido. A localização serve de base para outras funcionalidades que permitem prover autonomia ao robô. A funcionalidade de localização está representada na Figura 14.

Figura 14: Diagrama da localização.



Fonte: Autoria própria.

Para estimar a odometria a partir dos dados da *Point Cloud* do *LIDAR* foi utilizado o pacote *rf2o laser odometry*. Esse pacote estima a odometria 2D com base em varreduras planares a laser. Para isso, foi preciso converter as informações da *Point cloud*, que estão em 3D para 2D, como mostra no diagrama da Figura 14, no qual essa conversão é feita pelo pacote *Point Cloud to Laser scan*.

Com os dados do *LIDAR* devidamente ajustado, o pacote *rf2o laser odometry* estima a odometria e envia-o para o *Extended Kalman Filter (EKF)* fazer a fusão com os sensores da *IMU*, *GPS* e a odometria da roda do UGV, tornando a odometria ainda mais precisa. Nessa parte está com as linhas pontilhadas no diagrama devido a esse método funcionar apenas na simulação, no real foi descartado o EKF por causa do conflito gerado com os pacotes internos do UGV. Assim, os sensores que seriam utilizados como entrada do EKF também não foram utilizados.

3.4.1.1 Dependências

A localização tem como dependência a percepção. A funcionalidade no real depende apenas dos dados enviados pelo LIDAR, já na simulação são utilizados a IMU e a odometria

da roda do UGV. O GPS não é utilizado na simulação devido a não ser possível reproduzir essas informações no ambiente simulado.

3.4.1.2 Saídas

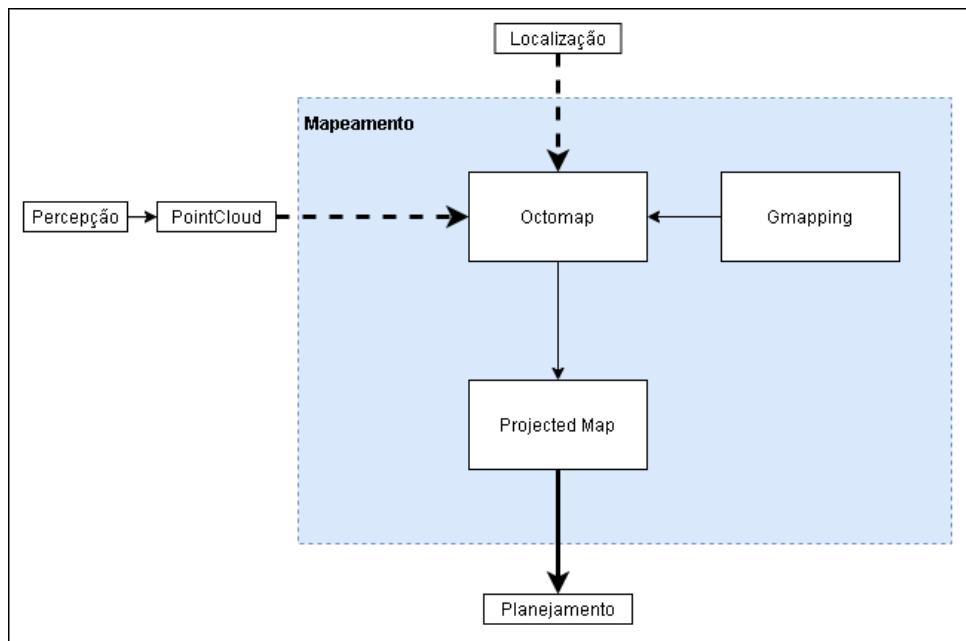
A funcionalidade irá atualizar a posição do UGV, reduzindo o erro gerado pela derivada da função que calcula a odometria. Dessa forma o UGV pode navegar com segurança, pois ele sabe sua real localização e distância entre os obstáculos.

3.4.2 Mapeamento

Dentre as funcionalidades mais utilizadas na robótica móvel, encontra-se a funcionalidade de mapeamento de ambientes. A capacidade de um robô de mapear o ambiente ao seu redor é essencial para que o mesmo consiga realizar a tarefa de movimentação de forma autônoma em ambientes internos ou externos. Para que tal tarefa torne-se possível, é imprescindível que o robô seja equipado com sensores capazes de fornecer boas informações espaciais do ambiente, assim como dotado de um algoritmo capaz de realizar a integração desses dados sensoriais de forma precisa e em todo instante.

A seguir é possível visualizar um diagrama, na Figura 15, representando o processo de mapeamento de um ambiente por parte do robô. A partir de sensores devidamente escolhidos e que sejam compatíveis com o algoritmo de mapeamento utilizado, é possível realizar, em todo instante, o mapeamento da área percorrida pelo robô.

Figura 15: Diagrama representando o mapeamento.



Fonte: Autoria própria.

Como mostra na Figura 15, o pacote responsável em criar o mapa é o *OctoMap*. Ele é

um método que, a partir da sua localização, mapeia uma área utilizando um LIDAR 3D, mas ele converte essa informação de 3D para 2D, construindo uma representação gráfica do mapa em 2D. Assim, com um mapa em mãos, o UGV pode realizar sua navegação de forma segura, realizando todas as trajetórias feitas pelo planejamento.

3.4.2.1 Dependências

O método *OctoMap* tem como dependências a localização, um *frame* de referência do mapa, que é criado pelo *Gmapping*, e os dados do LIDAR em forma de *Point Cloud*. O *Gmapping*, apesar de ser um *SLAM* que fornece localização e mapeamento diretamente, para esta aplicação está servindo apenas para criar o *frame* do mapa para referência do robô. Esse método é essencial porque não é possível criar essa referência diretamente.

3.4.2.2 Saídas

A funcionalidade tem como saída um mapa global representativo do ambiente percorrido pelo robô.

3.4.3 Planejamento

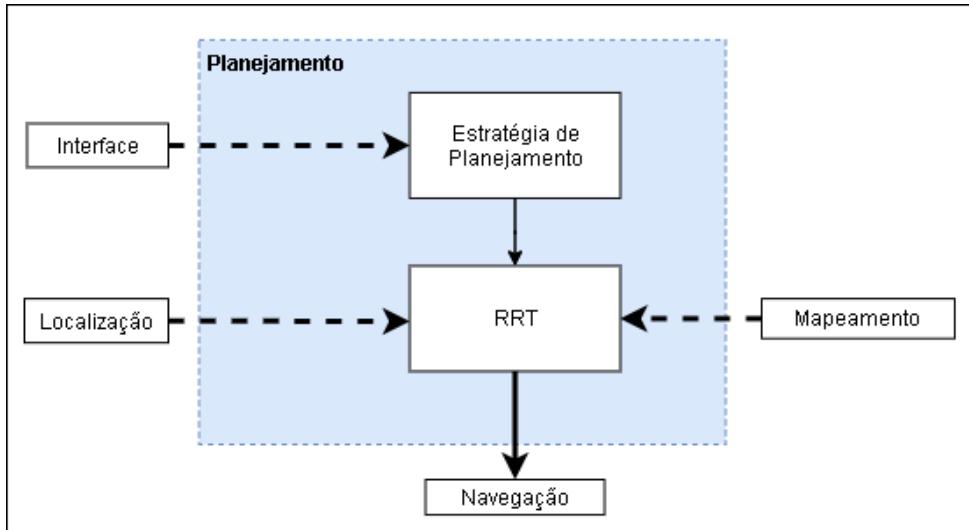
O planejamento de trajetória do UGV deve assegurar que, dadas posições inicial e final ou uma determinada área, seja traçada uma rota de forma que a estratégia de planejamento seja cumprida. Além disto, como o UGV estará realizando suas tarefas em ambientes não necessariamente estáticos, o planejamento deve também levar em consideração dados referentes à detecção de obstáculos caso haja a necessidade de replanejar a rota inicial com o intuito de evitar possíveis colisões.

A Figura 16 exibe o diagrama de funcionamento da funcionalidade. Através da interface, é solicitado ao robô que planeje uma trajetória para cumprir a sua missão, delimitando uma região de exploração. Com base no mapa global, será gerada uma trajetória inicial pelo planejador. Como mostra no diagrama, o *Rapidly-exploring Random Tree (RRT)* é um planejador de trajetória para explorar um ambiente. Ele busca pelos caminhos que ainda não foram explorados dentro de uma região definida pelo usuário.

3.4.3.1 Dependências

O planejamento depende da informação sobre qual será a missão, proveniente da Interface; odometria do robô, proveniente da Localização e o mapa global, provenientes do Mapeamento.

Figura 16: Diagrama representando o planejamento.



Fonte: Autoria própria.

3.4.3.2 Saídas

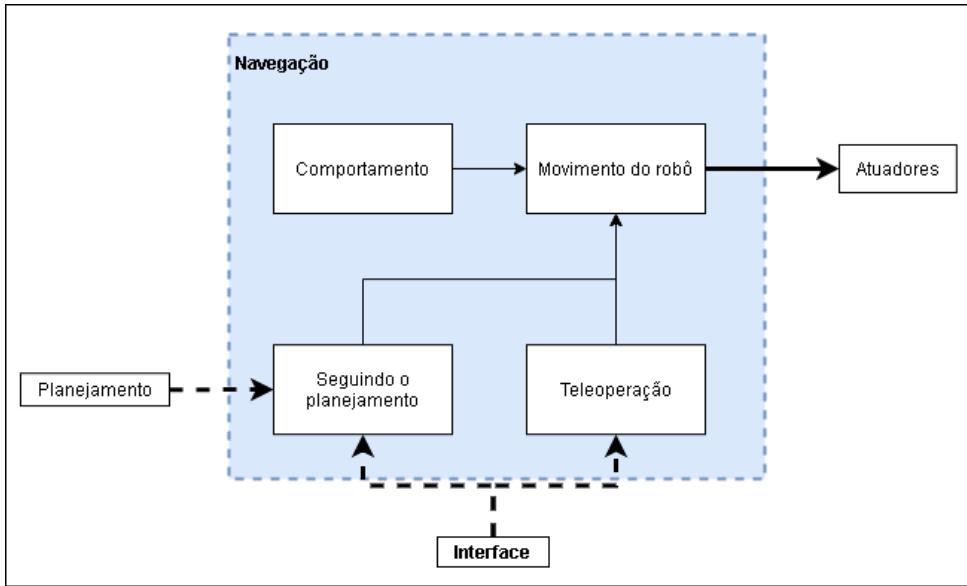
A saída desta funcionalidade será composta por trajetórias que irão para a Navegação do robô.

3.4.4 Navegação

Na navegação de um robô móvel autônomo é utilizado a combinação de três competências fundamentais para transitar de forma segura no ambiente: A localização, o planejamento e a percepção. A capacidade de localização do robô estabelece sua própria posição e orientação dentro do quadro de referência. O planejamento da rota utiliza da localização, extraindo a posição atual do robô, e a da posição do objetivo, traçando uma rota que deverá ser perseguida pelo robô. A percepção será a capacidade do robô em detectar objetos, um bloqueio ou uma situação de perigo, evitando assim os danos de colisão e realizar uma navegação segura. Portanto, a navegação de robôs móveis autônomos precisa dessas funcionalidades para oferecer os meios no qual o robô autônomo se move de forma segura de um local para outro em qualquer ambiente.

A funcionalidade navegação está representada pelo diagrama de interação entre as funcionalidades da Figura 17. O modo de navegação do robô será escolhido pela interface que poderá ser autônomo (Seguindo o plano) ou manual (Teleoperação). O movimento do robô está destacado como sendo o principal método da navegação. Nele está contido a ligação com os atuadores e a saída da funcionalidade.

Figura 17: Diagrama das interações das funcionalidades com a Navegação.



Fonte: Autoria própria.

3.4.4.1 Dependências

O sistema da navegação possui as seguintes funcionalidades como dependências: Planejamento e Interface. O planejamento deve fornecer os dados que serão a trajetória até o objetivo do UGV. O comportamento será quem irá decidir a forma da navegação, controlando assim a força e a velocidade dos atuadores. A interface irá monitorar a navegação e decidir se será feito de forma autônoma ou teleoperado. O State-machine realizar o controle de todas as funcionalidades.

3.4.4.2 Saídas

Como saída da funcionalidade de navegação será enviado os sinais de movimento dos motores para os atuadores.

3.4.5 Detecção

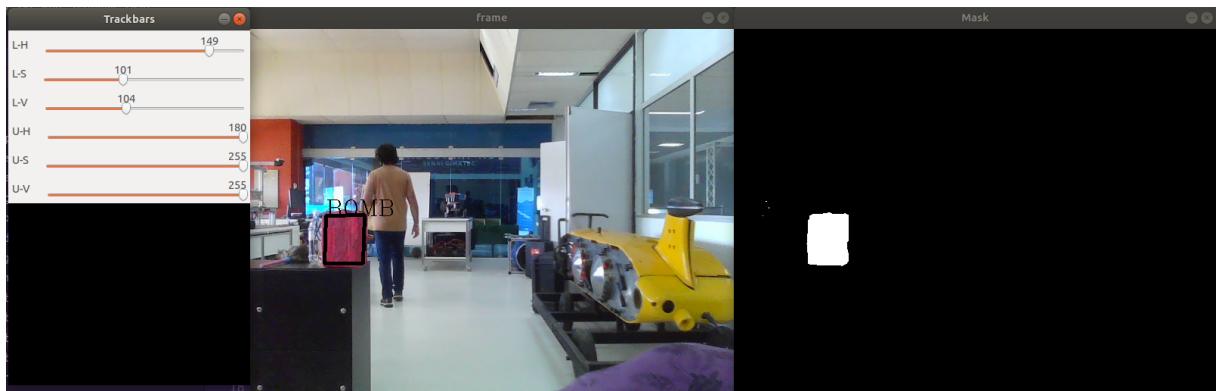
A detecção de um objeto específico, "bomba", demonstrada na figura 18 foi feito usando a biblioteca [OpenCV](#), uma multiplataforma desenvolvida pela Intel, livre ao uso acadêmico e comercial. A detecção inicial foi feita a partir de um filtro de coloração com o intuito de identificar uma caixa de cor única e específica demonstrada na figura 19. A aplicação foi falha quando a coloração do ambiente externo se igualava com a do objeto.

A aplicação da identificação do objeto com características específicas demonstrada na figura 18 foi feita a partir do uso de detecção de objetos usando classificadores em cascata baseados em recursos Haar, nessa abordagem é realizado um treinamento de função

Figura 18: Representação da bomba.



Figura 19: Identificação com filtro de cor.

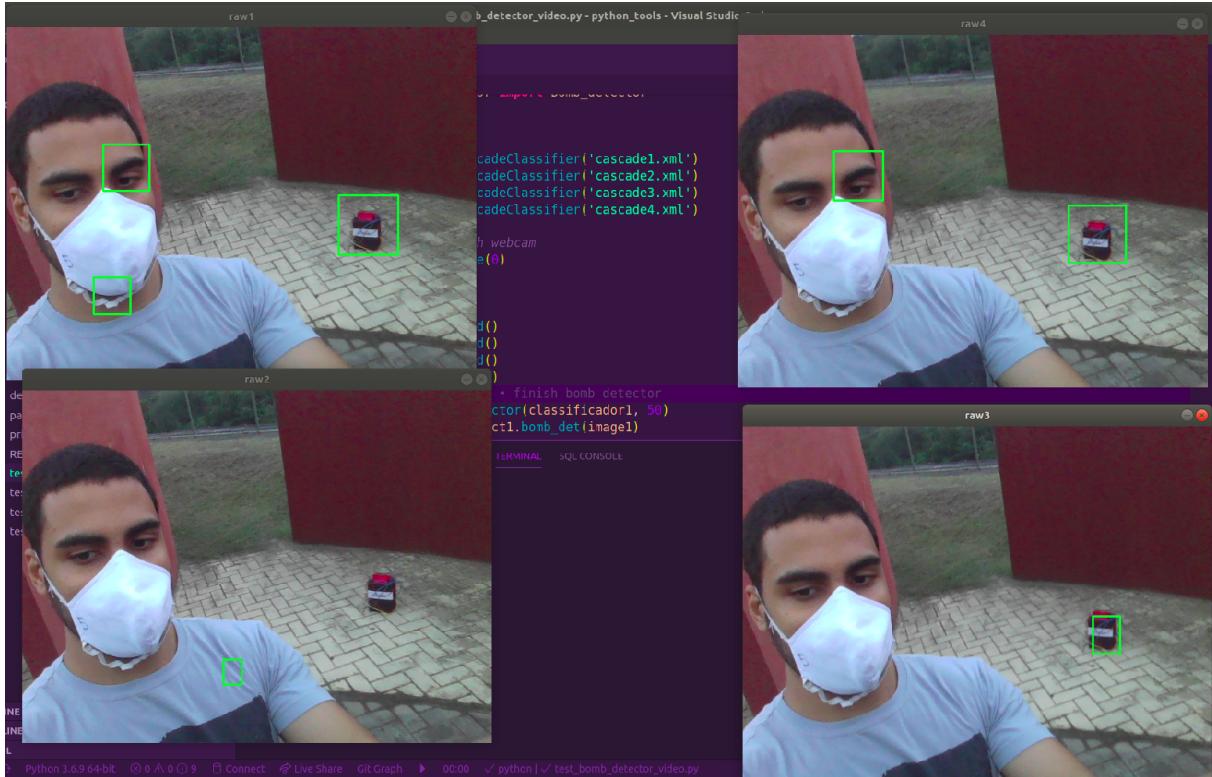


cascata, usando muitas imagens negativas, cujo objeto de interesse não esteja nela, e as imagens positivas, as quais cujo o objeto esteja inserido no ambiente. O treinamento e os métodos para identificação foram feitos usando a biblioteca [OpenCV](#), a identificação da imagens positivas e negativas faz com que algoritmo de treinamento procure características semelhantes na identificação do objeto.

3.4.5.1 Dependências

Esse tipo de identificação tem como dependência um arquivo em formato XML que indica a característica do treinamento feito com as imagens, o uso desta aplicação com funções da biblioteca [OpenCV](#) é utilizado para a identificação de características específicas da imagem podendo ajustar a detecção do objeto com uso de filtros da biblioteca, como demonstrada na figura 21

Figura 20: Identificação da bomba.



3.4.5.2 Saídas

Como saída da detecção o código feito em python te retorna a posição x e y da imagem em pixel e a largura e a altura do *bounding box*, quadrado que identifica o objeto, demonstrado de maneira representativa na figura 22

3.4.5.3 Dependências

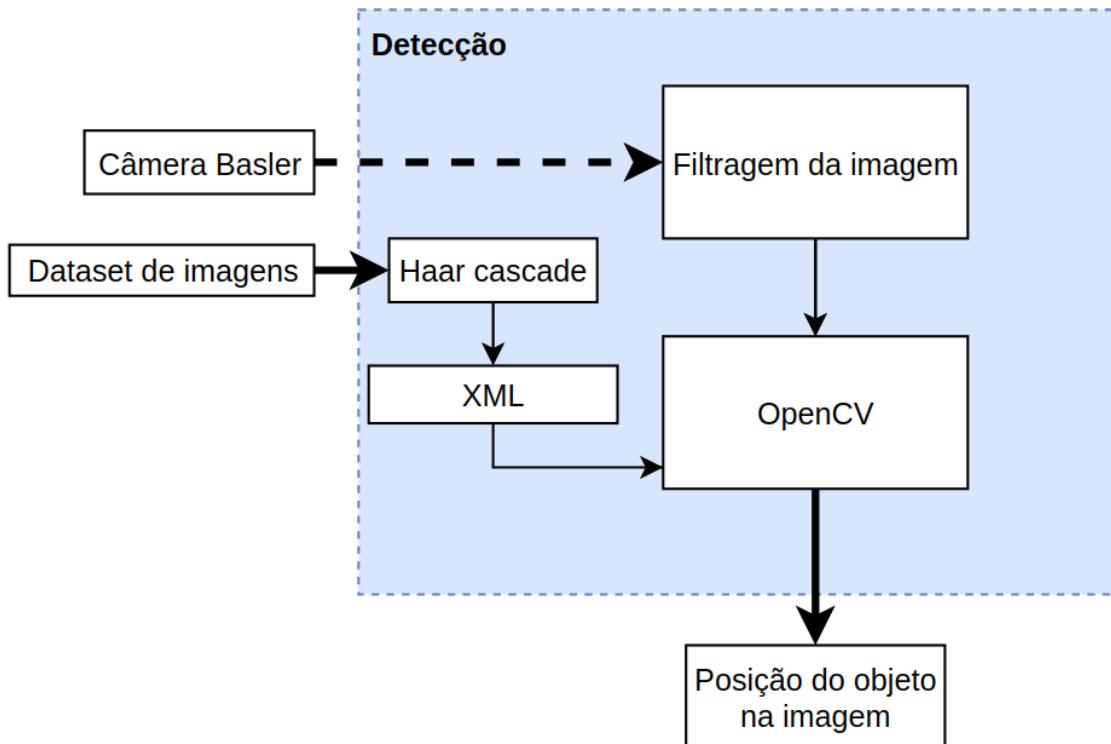
A funcionalidade de percepção depende dos dados oriundos dos sensores que estão incorporados na estrutura do *UGV*.

3.4.5.4 Saídas

As principais saídas desta funcionalidade são:

- *Image msg*: Mensagem que contém as informações sobre as propriedades da imagem que será enviada para localização para que possa ser realizada a detecção visual de objetos.
- *LaserScan msg*: Mensagem que será enviada para a localização no qual é utilizado para odometria.
- *PointCloud msg*: Mensagem que será enviada para o mapeamento pois contem

Figura 21: Fluxograma representando a detecção.



informações do ambiente.

- *Imu msg*: Mensagem contendo informações da orientação, aceleração, velocidade, posição do UGV que será enviada para o sistema de localização.
- *NavSatFix msg*: Mensagem contendo as informações da posição global e relativa do robô e deverá ser enviada para o sistema de localização.

3.4.6 Manipulação

A funcionalidade de manipulação está ligada ao manipulador robótico, em que será responsável pela movimentação do braço. Nela é utilizado o pacote do *Moveit* para controle e planejador de trajetória do manipulador. O diagrama da Figura 23 mostra a aplicação desta funcionalidade.

3.4.6.1 Dependências

A funcionalidade de manipulação fica aguardando uma posição e orientação para realizar seu objetivo.

Figura 22: Representação da identificação da bomba.



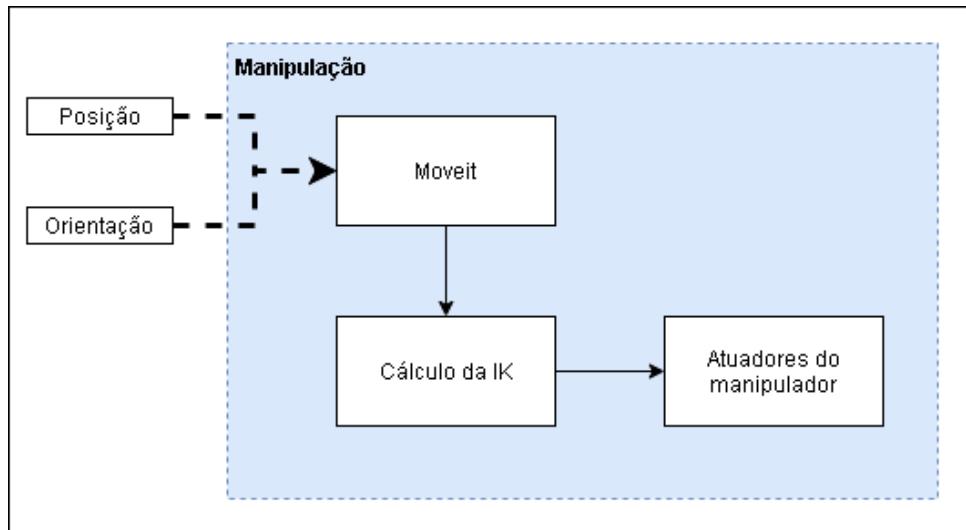
3.4.6.2 Saídas

Como saída, será enviado as posições ângulares de cada junta, estabelecendo assim uma posição e orientação final do *end effector*.

3.5 Arquitetura de software

O UGV foi desenvolvido para atuar em conjunto com o *ROS*, isto é, segue o propósito de conectar diferentes módulos, como câmeras, motores, sensores e códigos. A proposta é realizar conexão das funcionalidades com os modelos externos (*hardware*) de modo que estabeleça uma navegação estável e uma precisão na estimativa da posição do objeto detectado. A Figura 24 mostra a arquitetura simplificada das funcionalidades. Nele é possível ver a interação de uma funcionalidade com a outra. A máquina de estado, representado na Figura 24, está conectado na navegação, na detecção, no manipulador e nos atuadores. Essa configuração é devido a máquina ter controle apenas nessas funcionalidades

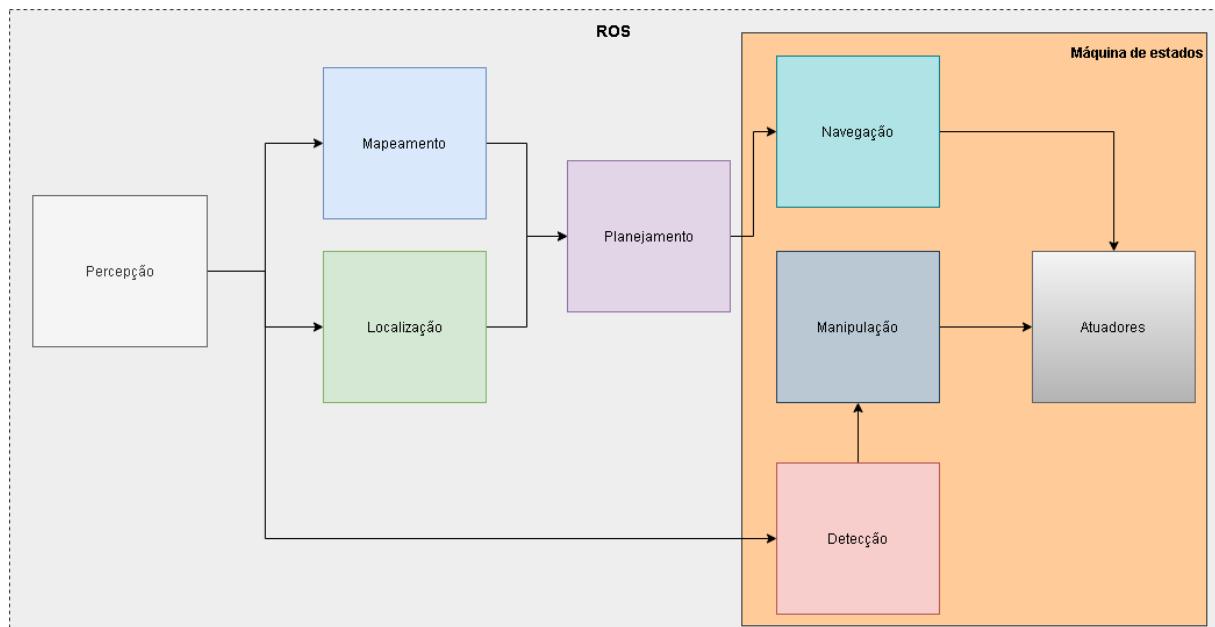
Figura 23: Diagrama da manipulação.



Fonte: Autoria própria.

listadas, e as demais são gerenciadas pelo próprio *framework* do *ROS*.

Figura 24: Integração simplificado das funcionalidades.



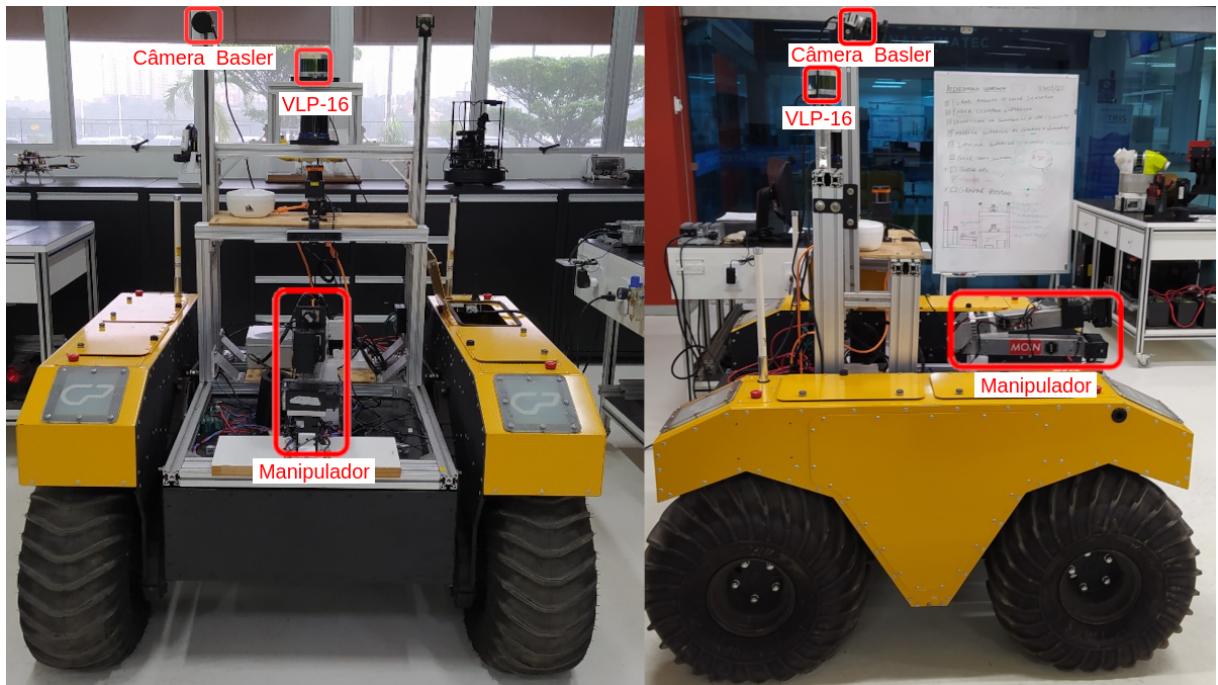
Fonte: Autoria própria.

4 IMPLEMENTAÇÃO

4.1 Estrutura física

A estrutura física do projeto Saci foi baseada no uso do [UGV](#), *Warthog*, demonstrado na figura 25 montado com algumas câmeras e sensores e neste projeto fará uso da câmera Basler anexo B [LiDAR](#) velodyne vlp-16 anexo D.

Figura 25: Vista frontal e lateral da plataforma móvel e seus componentes.



Fonte: Autores.

4.1.1 Câmera

Para prover o sistema com capacidade de detecção de objeto e *tag* para desta forma, obter dados necessários para aquisição de pose e identificação de objetos, utilizou-se uma câmera Basler ace acA4600 (Figura 26) acoplada a lente Kowa LM8HC(Figura 27).

Figura 26: Basler ace acA4600.



Fonte: ([BASLER, 2016](#))

Figura 27: Lente Kowa LM8HC.



Fonte: Kowa American Corp

4.1.2 LiDAR

O sensor [LiDAR](#) Velodyne VLP-16 usado para detecção de obstáculos e mapeamento. O sensor gera uma nuvem de pontos que caracteriza e identifica o ambiente ao seu redor gerando assim, um mapa com posicionamento de obstáculos em sua trajetória. As demais informações encontram-se no anexo [D](#).

4.1.3 Warthog

Um [UGV](#) de grande porte foi utilizado para aplicação do desafio e comunicação com o uso da *framework* [ROS](#) para transporte de equipamentos e execução do desafio proposto. As características e demais informações encontram-se no *datasheet* anexo [A](#)

4.1.4 Manipulador

O manipulador desenvolvido no projeto Jerotimon demonstrado na figura 28 foi implementado na parte frontal do UGV com a função de desarmamento da "bomba", a partir da aproximação do *end effector* de um fio com coloração específica.

Figura 28: Manipulador.



Fonte: Autores

5 RESULTADOS E DISCUSSÕES

Os resultados obtidos no desenvolvimento do projeto de detecção e desarmamento da bomba consistiram em conhecimento, devido ao protótipo não ter se comportado de forma estável no ambiente real. Não houve coleta de dados a partir dos testes realizados, porém, notou-se que o comportamento instável do UGV estava ligado a estrutura e a falta de controle no movimento. Essa instabilidade no movimento gerava um movimento de balanço e o LIDAR passava a detectar o chão como obstáculo, impedindo que o UGV navegassem por aquela trajetória previamente estabelecida.

A detecção foi feita em 3 formas de treinamento. Na primeira, 3 mil imagens aleatórias, de dimensão 100x100 pixels as quais geravam falso positivos e a filtragem não aprimorava a detecção da "bomba", por esse motivo não havia uma detecção precisa da "bomba". Na segunda, 20 mil imagens do ambiente, medindo 195x108 pixels onde o UGV trabalharia gerou outros falso positivos na imagem, porém isso se resolvia com filtragem devido a rigorosidade da semelhança do objeto, melhorando a sua detecção. Na terceira, 5 mil imagens variadas do ambiente eram formadas onde o UGV trabalharia de tamanho 500x500 pixels onde teria uma baixa filtragem, haveria uma detecção mais fácil e menos errônea, principalmente no ambiente de trabalho do UGV onde esses testes foram realizados em poucas imagens, e a filmagem com câmera de detecção no ambiente para aprimorar a precisão da identificação da "bomba".

Também, não foi possível utilizar o pacote do EKF para fazer o SLAM devido ao conflito com o próprio EKF do Warthog. Esse método EKF apresentava problemas na transmissão da mensagem e alguns sensores não eram conectados na arvore de TF. Essa conexão é essencial para a troca de mensagens no padrão *ROS*. Sem essas mensagens, os sensores enviam as informações, porém, as funcionalidades não conseguem coletar esses dados. Por motivos desconhecidos, existe um problema estabelecido na criação da arvore de TF, o que impede nas trocas de mensagens. Dessa forma, para que o mapeamento, a localização e as demais funcionalidades consigam utilizar os dados provenientes dos sensores, foi preciso remover o nó do EKF do warthog e conectar novamente a arvore de TF.

6 GESTÃO DO CONHECIMENTO

Neste capítulo estão descritas as lições aprendidas que foram adquiridas durante o processo de desenvolvimento do protótipo desde a etapa conceitual até a fase de testes com o modelo real. Na seção 6.2 está o Diagrama de Ishikawa, onde foi realizada uma análise para descobrir quais foram as causas que levaram a entrega do projeto incompleto.

Na seção 6.3 traz as informações essenciais para realizar a simulação do *UGV* e a utilização modelo real. Este guia tem o propósito de auxiliar o usuário na replicação dos experimentos realizados neste relatório.

6.1 Lições aprendidas

A Tabela 3 mostra as lições aprendidas durante todo o projeto, identificando para cada lição um: Tema, Fase, Impacto, O que ocorreu?, Como foi resolvido?, Resultados e Recomendações para os próximos projetos. Essa análise tem como foco o objetivo deste estudo e a correção dos impactos negativos para os projetos subsequentes.

Tabela 3: Lições aprendidas

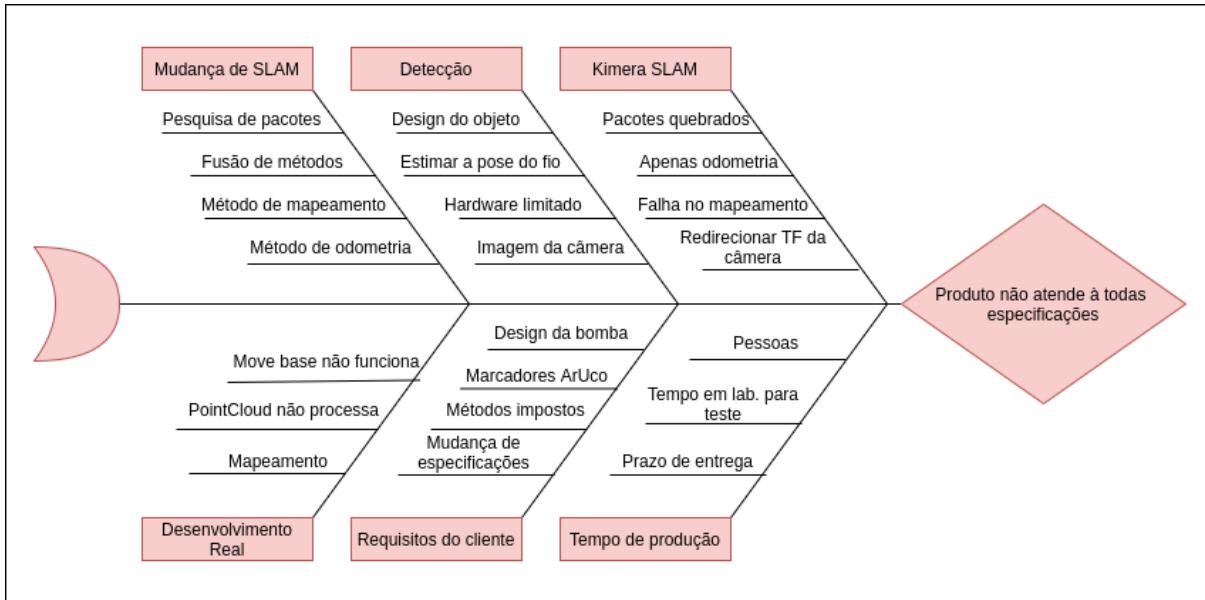
LIÇÕES APRENDIDAS						
Tema	Fase	Impacto	O que ocorreu?	Como foi resolvido?	Resultados	Recomendações para os próximos projetos
Gestão	Planejamento	Negativo	Planejamento do tempo de projeto não foi baseado nas ferramentas utilizadas	Foi testado outro método	SLAM não funciona	Conhecer as ferramentas antes de estipular o prazo para o projeto
Gestão	Execução	Negativo	Tempo insuficiente para a realização de testes	Tentou-se realizar testes sempre que havia disponibilidade do UGV	Não atende a todas especificações	Estipular prazo maior para a fase de testes
Gestão	Execução	Negativo	Alterações nos requisitos do cliente	Testou-se outros métodos de detecção	Não foi possível pegar a pose da bomba	Quando verificar que as modificações solicitadas vão causar atraso no projeto, solicitar nova data de entrega
Tecnológico	Execução	Negativo	O frame Map não envia a pose para o move base do robô	Outros métodos foram testados e foi pedido ajuda	Utilizou-se o gmapping para fazer a conexão entre os frames map e odom	Usar um método de SLAM que realize essa conexão

Fonte: Autoria própria.

6.2 Análise através do Diagrama de Ishikawa

Próximo ao prazo final do projeto foi observado que o mesmo seria entregue porém sem atender a todas funcionalidades do mesmo. Logo, foi realizada uma análise do que causou isso. Para tal foi utilizado o Diagrama de Ishikawa mostrado na Figura 29, onde foram levantados os problemas decorridos da Mudança de SLAM, da detecção, do Kimera SLAM, do Tempo de Produção, dos Requisitos do Cliente e do Desenvolvimento Real, que por sua vez resultaram em um produto que não atende à todas as especificações do projeto.

Figura 29: Análise a partir do Diagrama de Ishikawa.



Fonte: Autoria própria.

6.3 Guia de uso

Para replicar a simulação do *UGV*, ou a utilização do modelo real, é necessário seguir os passos descritos no link https://github.com/Brazilian-Institute-of-Robotics/wbm_ekfslam. Neste link estão todas as orientações para o uso. Recomenda-se a utilização do Ubuntu 18.04 LTS e o *ROS* Melodic Morenia, e que os pacotes mencionados sejam instalados no mesmo *workspace*.

Para criar um *workspace*:

- Criar um *workspace* para adicionar dentro deste os pacotes que serão usados na simulação.

```
$ mkdir nomeoworkspace_ws
```

- Entrar no *workspace*:

```
$ cd nomeoworkspace_ws
```

- Criar a pasta *source*¹:

```
$ mkdir src
```

¹ Pasta onde contém os arquivos fonte.

- Entrar no *source*:

```
$ cd src
```

- Clonar os pacotes citados no link do repositório citado anteriormente

- Retornar para a raiz do *workspace*:

```
$ cd ..
```

- Compilar o *workspace*:

```
$ catkin_make
```

- Ativar o ambiente virtual do *workspace*:

```
$ source devel/setup.bash
```

Após realizar os procedimentos citados o *workspace* estará configurado para executar a simulação, ou o uso do modelo real. Os *launches* que deveram ser lançados e sua ordem também estão descritos no repositório desse projeto.

7 CONCLUSÃO

O presente relatório retratou desde a concepção, simulação e construção do Saci, um robô autônomo que utiliza a plataforma da *Clearpath*, o *Warthog*, integrado com sensores e um manipulador robótico com 5 *DoF*,e que utiliza o *framework ROS*, capaz de explorar e navegar no ambiente externo de forma autônoma, identificar os *ArUcos* como regiões a não serem exploradas, detectar a "bomba"e efetuar o seu desarme com o manipulador. A estrutura física do robô foi concebida de acordo com os modelos *URDF* projetados.

Como citado anteriormente, o Saci utiliza o pacote *RS2O_Laser_Odometry* para estabelecer a odometria através do *laser* e juntamente com o pacote do *OctoMap* para realizar o mapeamento do ambiente externo. Unindo a odometria do *laser* com a *IMU* utilizando o método de EKF é estimado uma localização mais precisa. Dessa forma, com a localização e o mapa gerado foi feito o *EKF_SLAM*. Também foi utilizado o pacote *RRT* que é o planejador para onde o *UGV* deve ir.

Para possibilitar o uso da ferramenta *Moveit*, arquivos de configuração foram gerados e sua comunicação com o modelo do manipulador simulado no *Gazebo* foi estabelecida. Para resolver as equações de cinemática inversa optou-se pelo plugin TRAC-IK e para o planejamento de trajetória foi utilizada a biblioteca *OMPL*.

Foi desenvolvido em linguagem Python um pacote capaz de comunicar o sistema de detecção da "bomba"e o sistema de execução do manipulador. A utilização da câmera Basler ace aca4600 possibilitou a identificação da "bomba"com precisão, tornando o sistema capaz de localizar a mesma e planejar uma trajetória que leve o *end effector* do manipulador até o alvo estabelecido.

Para o aprimoramento das funcionalidades do robô é necessário realizar ajustes no pacote de navegação, para que o *UGV* tenha uma estabilidade no seu movimento, melhoramento na detecção do objeto estabelecido e estimativa da posição desse objeto.

REFERÊNCIAS

- ALVES, G. T. M. *Um Estudo das Técnicas de Obtenção de Forma a partir de Estéreo e Luz Estruturada para Engenharia*. Tese (Doutorado) — PUC-Rio, 2005. Citado na página 15.
- BASLER, P. *pylon Camera Software Suite*. 2016. Disponível em: <<https://www.baslerweb.com/en/products/software/>>. Citado na página 40.
- CRAIG, J. J. Robótica. 3^a edição. *Rev. Atual*, 2012. Citado na página 16.
- GONZALEZ, R. C.; WOODS, R. E. *Processamento de imagens digitais*. [S.l.]: Editora Blucher, 2000. Citado na página 15.
- HERNANDEZ-MENDEZ, S. et al. Design and implementation of a robotic arm using ros and moveit! In: IEEE. *2017 IEEE International Autumn Meeting on Power, Electronics and Computing (ROPEC)*. [S.l.], 2017. p. 1–6. Citado na página 15.
- ISO-8373. *Robots and robotic devices — Vocabulary*. Geneva, CH, 2012. Citado na página 13.
- JR, J. J. U.; DENAVIT, J.; HARTENBERG, R. An iterative method for the displacement analysis of spatial mechanisms. 1964. Citado na página 16.
- LEITE, A. C. *Controle Híbrido de Força e Visão de um Manipulador Robótico Sobre Superfícies Desconhecidas*. Tese (Doutorado) — Tese de Mestrado, Departamento de Engenharia Elétrica, 2005. Citado na página 12.
- MATARIĆ, M. J. et al. *The robotics primer*. [S.l.]: MIT press, 2007. Citado na página 14.
- NORTON, C. *Os mecanismos da escrita criativa*. [S.l.: s.n.], 2001. Citado na página 17.
- PAUL, R. P. *Robot manipulators: mathematics, programming, and control: the computer control of robot manipulators*. [S.l.]: Richard Paul, 1981. Citado na página 16.
- PIMENTA, T. T. Controle de manipuladores robóticos. *PUC-Rio de Janeiro, Departamento de Engenharia de Controle e automação*, 12 2009. Citado na página 11.
- ROBOS-AUTONOMOS. s.d. <<https://www.voitto.com.br/blog/artigo/robos-autonomos>>. Acessado: 07-12-2020. Citado na página 14.
- SCIAVICCO, L.; SICILIANO, B. *Modelling and control of robot manipulators*. [S.l.]: Springer Science & Business Media, 2012. Citado na página 17.
- SEDIGHI, K. H. et al. Autonomous local path planning for a mobile robot using a genetic algorithm. In: IEEE. *Proceedings of the 2004 congress on evolutionary computation (IEEE Cat. No. 04TH8753)*. [S.l.], 2004. v. 2, p. 1338–1345. Citado na página 15.
- TILBURY, D.; ULSOY, A. Reliable operations of unmanned ground vehicles: Research at the ground robotics reliability center. In: *Proceedings of IARP workshop on technical challenges for dependable robots in human environments*. [S.l.: s.n.], 2010. p. 27–32. Citado na página 14.

WANG, M. et al. Fuzzy logic based robot path planning in unknown environment. In: IEEE. *2005 International Conference on Machine Learning and Cybernetics*. [S.l.], 2005. v. 2, p. 813–818. Citado na página 15.

ANEXO A

Warthog

Maiores informações sobre o *datasheet* da plataforma móvel *warthog* desenvolvida pela *Clearpath Robotics Inc* anexado a seguir.



Hardware Overview

Please see Figure 1 for a view of some of Warthog's key external-facing features.

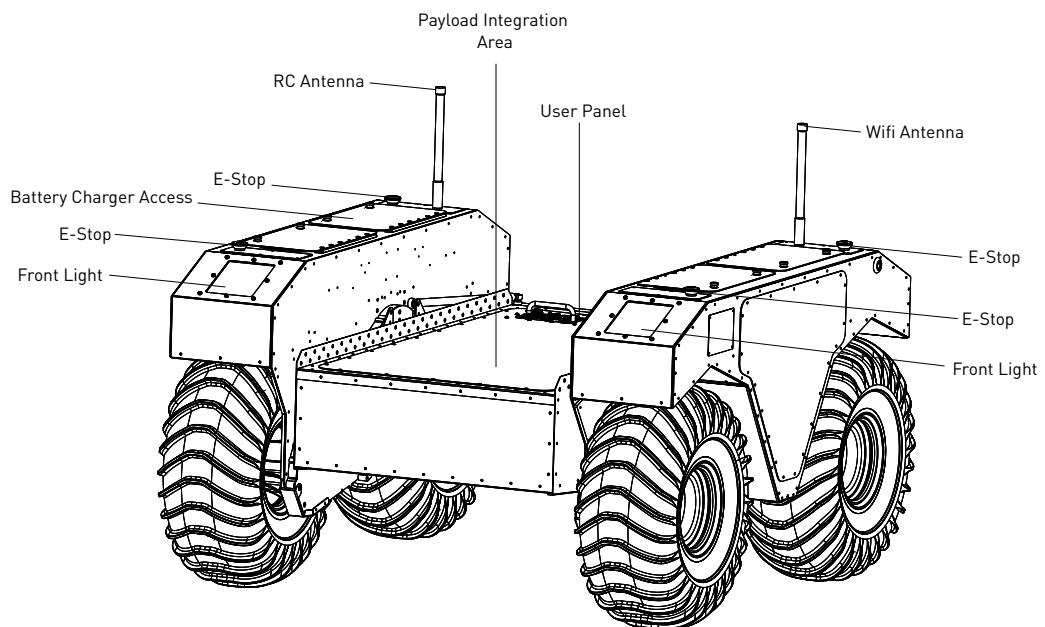


Figure 1: Warthog Hardware Overview



Battery Charger

Please see Appendix for information about the provided charger.

Bilge Pumps

The Warthog has two bilge pumps, with one situated in each of the drive units underneath the motor.

The bilge pumps are used to remove water from the main chassis and drive units during operation in water. At initial startup, an audible sound can be heard from the bilge pumps being initiated. These pumps are an automatic pumping system that check for water levels inside the drive units. The pumps automatically come on every two minutes and check for water levels. If the water level inside the drive unit exceeds a predetermined level, the pumps will remain on, otherwise they will shut off. During prolonged use in water, some water may appear in the drive unit. This is normal and acceptable.

This outlet leads from the pump to the exterior of the Warthog. It is used to remove excess water from the chassis and drive units. No obstructions should be placed in front of or around this area. Obstruction of water flow may result in damage to the internal electrical components and loss of function in the Warthog.

User Panel

The User Panel provides access to the user power panel, as well as USB, serial, and ethernet ports. The power panel can be used to power your payloads. The USB 3.0 and ethernet ports are connected directly to the onboard PC. To connect a device to the onboard network, it's suggested to give it a static IP in the 192.168.131.x subnet, avoiding IPs in use by the following pre-existing devices:

192.168.131.1	Onboard PC (all ports, br0 network interface).
192.168.131.2	Ethernet-connected MCU.
192.168.131.14	Front-facing LIDAR (optional).
192.168.131.13	Rear-facing LIDAR (optional).

Table 1: Warthog Onboard Network Devices

For more information on electrical integration, please see subsection 4.3 on page 23.

Payload Integration Area

All payloads should be mounted to the central chassis when traversing through water to prevent rolling. The primary payload of the unit should be placed on the central chassis. If necessary loading can be placed on the drive units however payloads should not exceed 50 lbs on each drive unit.

For more information and guidance on mounting payload structures on top of Warthog, please refer to subsection 4.2 on page 22.



Technical Specifications

Key specifications of Warthog are shown in Table 2.

External Dimensions (L x W x H)	1.52 x 1.38 x 0.83 m (4.9 x 4.5 x 2.72 ft)
Base Weight	280 kg (551 lbs)
Ground Clearance	254 mm (10 in)
Max Payload	272 kg (600 lbs)
Max Incline	35-45°
Max Speed	18 km/h (11 mph)
Suspension	Geometric Passive Articulation
Drive Configuration	4x4 Skid Steer
Operating Environment	Outdoor
Traction	24" Argo tire (24" Turf tire or 12" wide Quad Track System optional)
Battery Chemistry	AGM sealed lead acid (Li-ion optional)
Capacity	105 Ah at 48 V, expandable to 200 Ah with Li-ion option
Nominal Run Time	Lead acid: 2.5 hrs, Li-ion: 6 hrs
Charge Time	4 Hours approx
User Power	5 V, 12 V Fused (24 V, 48 V optional)
Control Modes	Remote control, computer controlled velocity commands, indoor/outdoor autonomy packages
Feedback	Battery voltage, motor currents, wheel odometry, control system status, temperature, safety status
Communication	Ethernet, USB, Remote Control, Wi-Fi
Drivers and APIs	Packaged with ROS Indigo (includes RViz, Gazebo support), Matlab API available

Table 2: Warthog System Specifications

ANEXO B

Câmera Basler ace acA4600

A ficha resumo contendo as principais especificações da câmera fotográfica Basler ace acA4600 está disposta na figura 30

Maiores informações e desenho mecânico deste componente estão dispostos no *datasheet* anexado a seguir.

Figura 30: Especificações da câmera Basler ace.

Item	Especificação
Especificação de câmera	acA4600
Resolução	4608 × 3288 [px]
Modelo do sensor	On-Semi MT9F002
Taxa de captura (padrão)	10 [fps]
Tamanho de pixel	1,4 × 1,4 μ m
Alcance dinâmico	62,1 [dB]
Círculo máximo de imagem	Formato óptico 1/2,3"
Encaixe de lente	C-Mount
Protocolo de comunicação	USB 3.0 (5 [Gbit/s])
Porta de comunicação	USB
Tensão de operação	5 [V]
Temperatura de operação	0 ~ 50 [°C]
Massa	< 80 [g]
Dimensões (L x A x P)	29,3 x 29 x 29 [mm ³]
Potência necessária	~ 2,5 W a 5 V _{DC} , máximo de 2,8 W

Fonte: Autores.

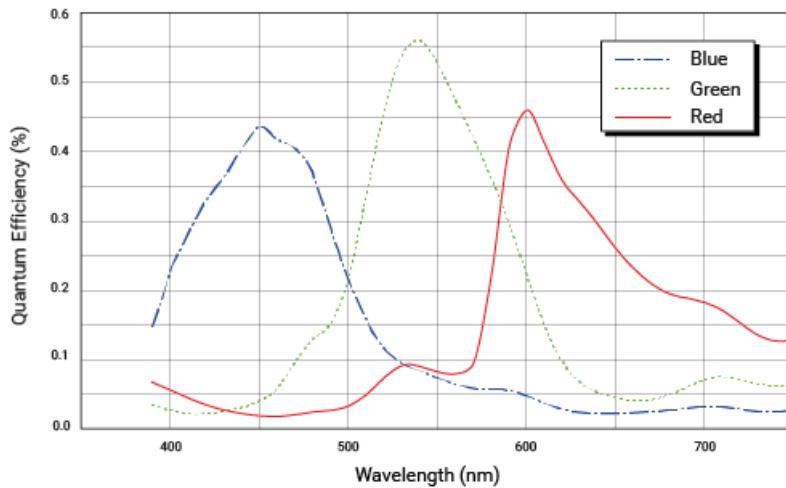
Specifications

General Specifications

Specification	acA4600-10uc
Resolution (H x V Pixels)	4608 x 3288
Sensor Type	ON Semiconductor MT9F002 Progressive scan CMOS Rolling shutter
Optical Size	1/2.3"
Effective Sensor Diagonal	8.0 mm
Pixel Size (H x V)	1.4 µm x 1.4 µm
Frame Rate (at Default Settings)	10 fps
Product Line	 ace classic
Mono / Color	Color
Image Data Interface	USB 3.0, nominal max. 5 Gbit/s (SuperSpeed)
Pixel Formats	See Pixel Format .
Synchronization	Via hardware trigger Via software trigger Via free run
Exposure Time Control	Programmable via the camera API
Camera Power Requirements	Nominal 5 VDC supplied via the camera's USB 3.0 port ≈2.5 W (typical) @ 5 VDC ≈2.8 W (max.)
I/O Lines	1 opto-coupled input line

	1 opto-coupled output line 2 general purpose I/O (GPIO) lines
Lens Mount	C-mount
Size (L x W x H)	29.3 mm x 29 mm x 29 mm (without lens mount or connectors) 48.2 mm x 29 mm x 29 mm (with lens mount and connectors)
Weight	<80 g
Conformity	CE (includes RoHS), UL Listed, FCC, GenICam 2.x (including PFNC 2.x and SFNC 2.x), IP30, USB3 Vision, REACH The EU Declaration of Conformity is available on the Basler website .
Software	Basler pylon Camera Software Suite (version 4.0 or higher) Available for Windows, Linux x86, Linux ARM, and OS X
Accessories	Cables for your camera model Lenses for your camera model Additional accessories for your camera model

Spectral Response



The spectral response curve excludes lens characteristics, light source characteristics, and IR cut filter characteristics.

IR Cut Filter

Color cameras are equipped with an IR cut filter. The filter is mounted in a filter holder inside the lens mount.

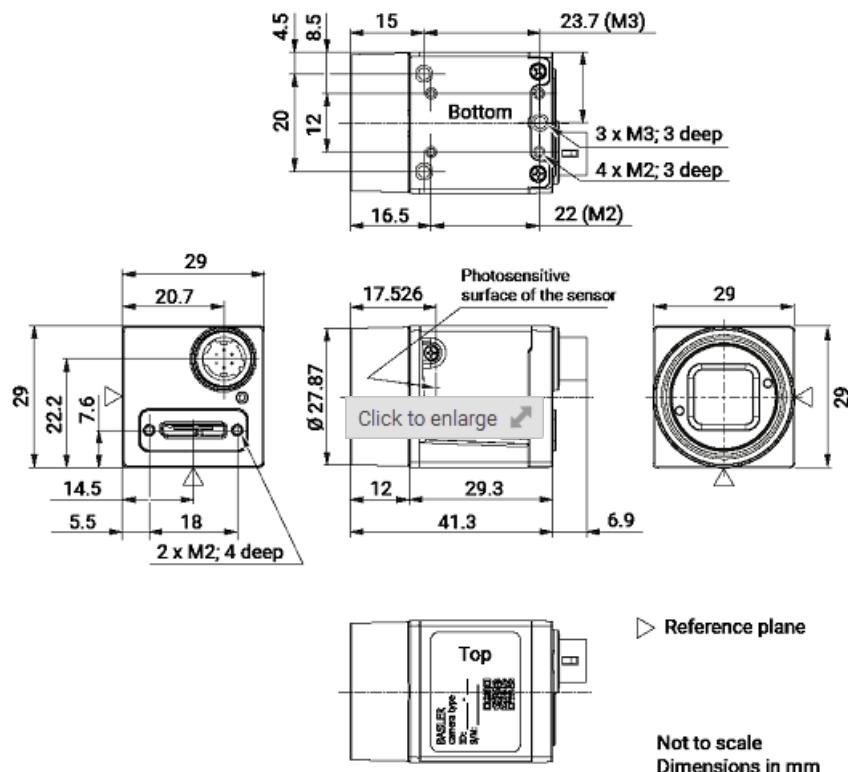
The IR cut filter has the following spectral characteristics:

Wavelength [nm]	Transmittance
450–610	$T_{\min} > 90 \%$
450–620	$T_{\text{avg}} > 93 \%$
645 ± 10	$T = 50 \%$
700–1070	$T_{\max} < 4 \%$
690–1070	$T_{\text{avg}} < 1 \%$

The filter holder can be removed. For more information, see the [ACE IR Cut Filter Holder Removal Procedure](#) application note.

Mechanical Specifications

Camera Dimensions and Mounting Points



Maximum Allowed Lens Intrusion

→ See [Maximum Allowed Lens Intrusion](#).

Mounting Instructions

→ See [Mounting Instructions](#).

Stress Test Results

→ See [Stress Test Results](#).

Requirements

Environmental Requirements

Temperature and Humidity

Housing temperature during operation	0–50 °C (32–122 °F)
Humidity during operation	20–80 %, relative, non-condensing
Storage temperature	-20–80 °C (-4–176 °F)
Storage humidity	20–80 %, relative, non-condensing
Housing temperature according to UL 60950-1	max. 70 °C (158 °F)
Ambient temperature according to UL 60950-1	max. 30 °C (86 °F)

UL 60950-1 test conditions: no lens attached to camera; no heat dissipation measures; ambient temperature kept at 30 °C (86 °F).

Heat Dissipation

→ See [Providing Heat Dissipation](#).

Electrical Requirements

DANGER
Electric Shock Hazard
WARNING
Fire Hazard
NOTICE
Incorrect voltage can damage the camera.

Camera Power

You must supply camera power that complies with the Universal Serial Bus 3.0 specification.

The camera's nominal operating voltage is 5 VDC, effective on the camera's connector.

Opto-Coupled I/O Input Line

Voltage	Description
30 VDC	Absolute maximum. This voltage must never be exceeded. Doing so may damage the camera and voids the warranty.
0–24 VDC	Safe operating range.
0–1.4 VDC	Indicates a logical 0 (with inverter disabled).
>1.4–2.2 VDC	Region where the logic level transition occurs; the logical status is not defined in this region.
>2.2 VDC	Indicates a logical 1 (with inverter disabled).
<ul style="list-style-type: none"> Minimum current: 5 mA Current draw: 5–15 mA 	

Opto-Coupled I/O Output Line

Voltage	Description
30 VDC	Absolute maximum. This voltage must never be exceeded. Doing so may damage the camera and voids the warranty.
3.3–24 VDC	Safe operating range.
<3.3 VDC	Unreliable I/O output.
<ul style="list-style-type: none"> Leakage current: <60 μA. Actual leakage depends on operating temperature and production spread of electronic components. Maximum load current: 50 mA Minimum load current: Not specified. Consider the following: <ul style="list-style-type: none"> Leakage current will have a stronger effect when load currents are low. Propagation delay of the output increases as load currents decrease. Higher-impedance circuits tend to be more susceptible to EMI. Higher currents cause higher voltage drops in long cables. 	

General Purpose I/O Lines

NOTICE

Applying incorrect electrical signals to the camera's GPIO line can severely damage the camera.

Operation as Input

Voltage	Description
30 VDC	Absolute maximum. This voltage must never be exceeded. Doing so may damage the camera and voids the warranty.
0–5 VDC	Safe operating range. The minimum external pull-up voltage is 3.3 VDC.
0–0.8 VDC	Indicates a logical 0 (with inverter disabled).
>0.8–2.0 VDC	Region where the logic level transition occurs; the logical status is not defined in this region.
>2.0 VDC	Indicates a logical 1 (with inverter disabled).
<ul style="list-style-type: none"> Current draw (high-level): <100 µA Sink current: Your application must be able to accept 2 mA sink current from the GPIO input line without exceeding 0.8 VDC. 	

Operation as Output

Voltage	Description
30 VDC	Absolute maximum. This voltage must never be exceeded. Doing so may damage the camera and voids the warranty.
3.3–24 VDC	Safe operating range.
<3.3 VDC	Unreliable GPIO output.
<ul style="list-style-type: none"> Internal pull-up resistor: $\approx 2\text{ k}\Omega$, with open collector. Many applications will have to provide an additional pull-up resistor. Residual voltage ("on" state): $\approx 0.4\text{ V}$ at 50 mA and 25 °C (77 °F) housing temperature. Actual residual voltage depends on operating temperature, load current, and production spread of electronic components. Leakage current: <60 µA. Actual leakage depends on operating temperature and production spread of electronic components. Maximum load current: 50 mA Minimum load current: Not specified. However, consider the following: <ul style="list-style-type: none"> Leakage current will have a stronger effect when load currents are low. Propagation delay of the output increases as load currents decrease. Higher-impedance circuits tend to be more susceptible to EMI. 	

- Higher currents cause higher voltage drops in long cables.

Circuit Diagrams

→ See [Circuit Diagrams for Basler ace Cameras](#).

Cable Requirements

USB 3.0 Cable

- Use a high-quality USB 3.0 cable with a Micro-B plug.
- To [avoid EMI](#), cables must be shielded, as specified in the USB 3.0 standard.
- Basler recommends using USB 3.0 cables from the [Basler Vision Components](#) range.

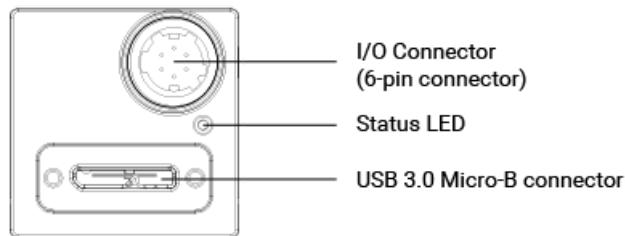
For more information about recommended USB 3.0 cables, see the [Recommended Accessories for Basler USB 3.0 Cameras](#) document.

I/O Cable

- The I/O cable must be shielded.
- The I/O cable must have a cross-section at least 0.14 mm^2 (close to AWG26).
- Use twisted pair wire cables.
- Maximum recommended cable length: 10 m
- Camera-side connector: Hirose micro plug (part number HR10A-7P-6S) or equivalent
- Close proximity to strong magnetic fields should be avoided.
- Basler recommends using I/O cables from the [Basler Vision Components](#) range:
 - [GPIO cable, 10 m](#) (yellow cable): For use with the [GPIO lines](#) of your camera. To avoid interferences due to crosstalk, the opto-coupled I/O lines are not connected.
 - [Opto-I/O cable, 10 m](#) (blue cable): For use with the [opto-coupled I/O lines](#) of your camera. To avoid interferences due to crosstalk, the GPIO lines are not connected.
 - [Opto-GPIO Y-cable, 2 x 10 m](#) (yellow-blue cable): Allows you to use the [GPIO lines](#) and the [opto-coupled I/O lines](#) simultaneously without interferences due to crosstalk. There are two separate wires to split both I/O types.

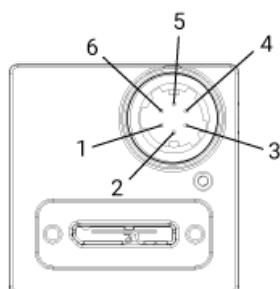
Physical Interface

Camera Connectors and Status LED



6-pin connector	Hirose micro receptacle (part number HR10A-7R-6PB) Recommended mating connector: Hirose micro plug (part number HR10A-7P-6S)
USB 3.0 Micro-B connector	Standard USB 3.0 Micro-B connector with screw lock Recommended mating connector: Standard connector with screws
Status LED	Indicates camera operation (LED lit = camera operating).

Connector Pin Numbering and Assignments



Pin	Line	Function
1	Line 3	General purpose I/O (GPIO) line
2	Line 1	Opto-coupled I/O input line
3	Line 4	General purpose I/O (GPIO) line
4	Line 2	Opto-coupled I/O output line
5	-	Ground for opto-coupled I/O lines
6	-	Ground for General Purpose I/O (GPIO) lines

Precautions

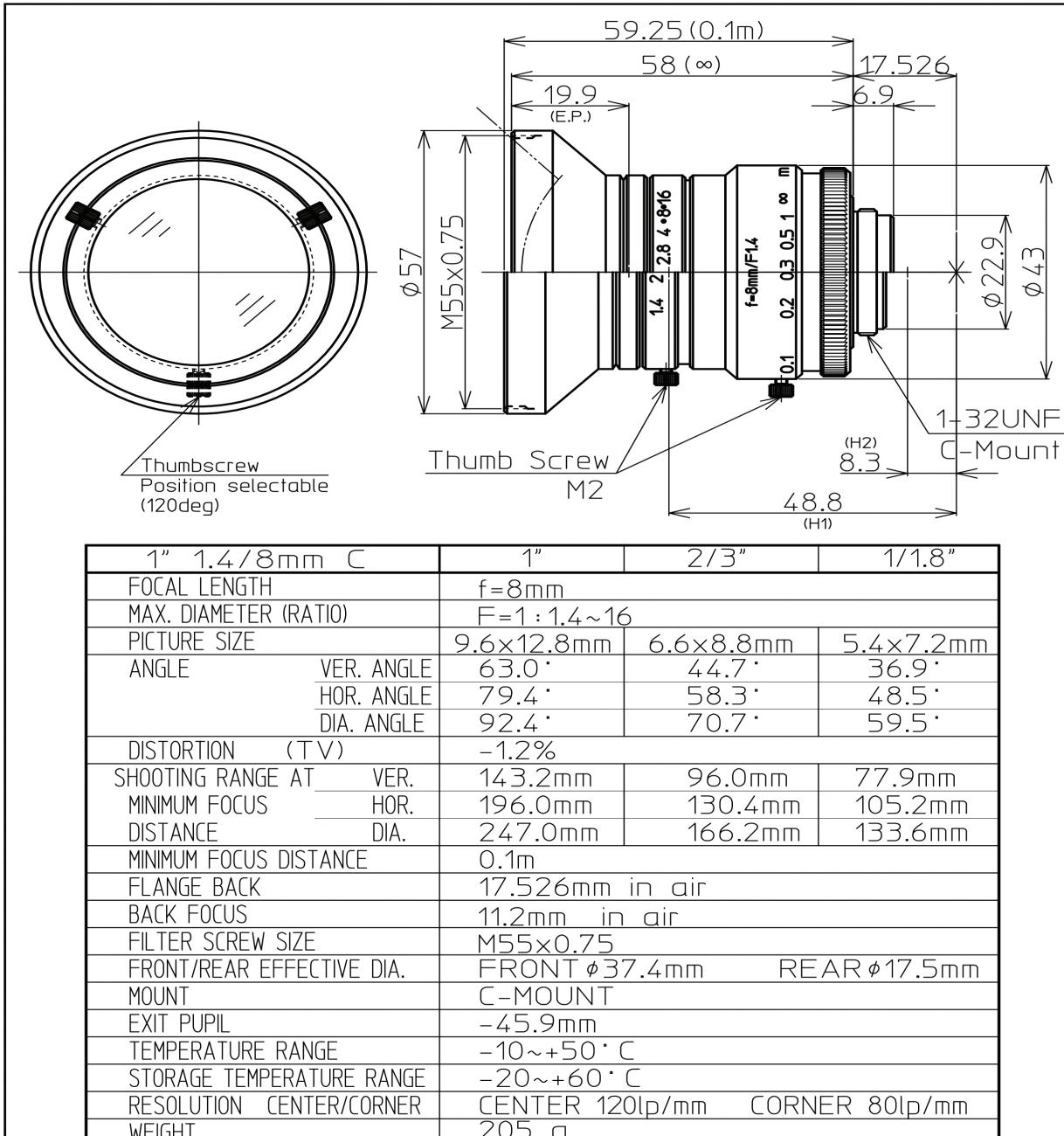
→ See [Safety Instructions for Basler ace Cameras](#).

Installation

ANEXO C

Lente Kowa LM8HC

As principais especificações da lente Kowa LM8HC utilizada em conjunto com a câmera fotográfica está disposta no *datasheet* do anexo a seguir.



NOTE: Specifications and availability are subject to change without notice



	1/1	HR975NCN-3H
2015.7.30		1" F1.4 f=8mm
2014.12.25		LM8HC
2013.4.15		

ANEXO D

Velodyne LiDAR Puck LITE

Maiores informações e características físicas demostrado no *datasheet* anexado a seguir.

Puck LITE™

LIGHT WEIGHT REAL-TIME 3D LiDAR SENSOR



Puck LITE

Our Lightest Sensor Ever

Velodyne LiDAR's Puck LITE is a lighter version of the VLP-16 Puck for applications that demand a lower weight to meet their requirements. Aside from the weight, the Puck LITE has identical performance to the VLP-16. The sensor retains Velodyne's patented 360° surround view to capture real-time 3D LiDAR data that includes distance and calibrated reflectivity measurements.

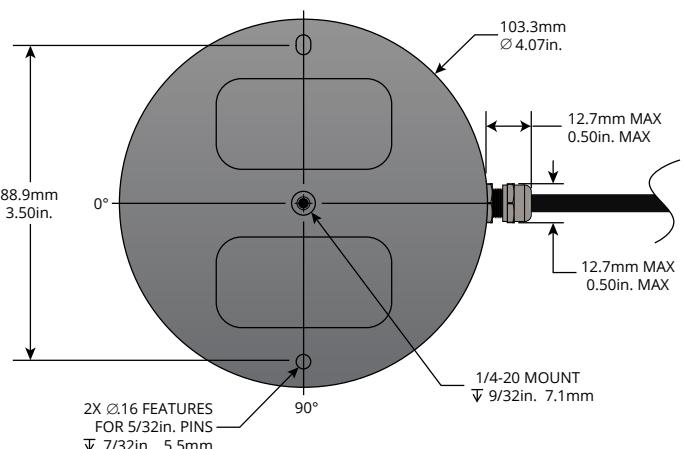
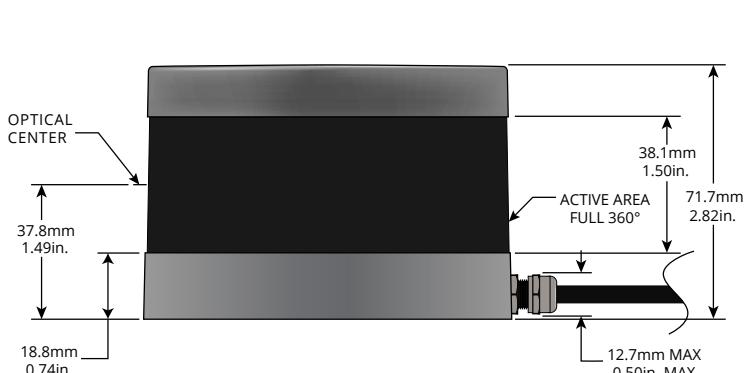
Unprecedented Field of View and Point Density

The Puck LITE has a range of 100 m with dual return mode to capture greater detail in the 3D image with a low power consumption. A compact footprint and an industry leading weight for a LiDAR sensor with high resolution makes it ideal for UAV/drone and mobile applications in the areas of 3D mapping/imaging, inspection and navigation.

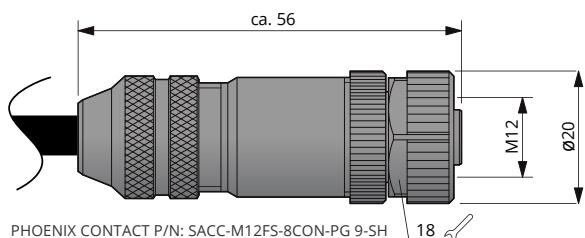
The Puck LITE supports 16 channels and generates approximately 300,000 points/second from a 360° horizontal field of view and a 30° vertical field of view ($\pm 15^\circ$ from the horizon). The Puck LITE has no visible rotating parts and is encapsulated in a package that allows it to operate over a wide temperature range and environmental conditions.



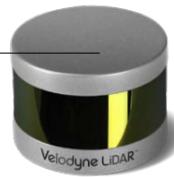
DIMENSIONS *(Subject to change)*



M12 CONNECTOR OPTION



For other connector options contact
Velodyne Sales (sales@velodyne.com)



Light Weight Real-Time 3D LiDAR Sensor

The Puck LITE provides high definition 3-dimensional information about the surrounding environment.

Specifications:

Sensor:	<ul style="list-style-type: none"> • 16 Channels • Measurement Range: 100 m • Range Accuracy: Up to ± 3 cm (Typical)¹ • Field of View (Vertical): +15.0° to -15.0° (30°) • Angular Resolution (Vertical): 2.0° • Field of View (Horizontal): 360° • Angular Resolution (Horizontal/Azimuth): 0.1° – 0.4° • Rotation Rate: 5 Hz – 20 Hz • Integrated Web Server for Easy Monitoring and Configuration
Laser:	<ul style="list-style-type: none"> • Laser Product Classification: Class 1 Eye-safe per IEC 60825-1:2007 & 2014 • Wavelength: 903 nm
Mechanical/ Electrical/ Operational	<ul style="list-style-type: none"> • Power Consumption: 8 W (Typical)² • Operating Voltage: 9 V – 18 V (with Interface Box and Regulated Power Supply) • Weight: ~590 g (without Cabling and Interface Box) • Dimensions: See diagram on previous page • Environmental Protection: IP67 • Operating Temperature: -10°C to +60°C³ • Storage Temperature: -40°C to +105°C
Output:	<ul style="list-style-type: none"> • 3D LiDAR Data Points Generated: <ul style="list-style-type: none"> - Single Return Mode: ~300,000 points per second - Dual Return Mode: ~600,000 points per second • 100 Mbps Ethernet Connection • UDP Packets Contain: <ul style="list-style-type: none"> - Time of Flight Distance Measurement - Calibrated Reflectivity Measurement - Rotation Angles - Synchronized Time Stamps (μs resolution) • GPS: \$GPRMC and \$GPGGA NMEA Sentences from GPS Receiver (GPS not included)

63-9286 Rev-H

For more details and ordering information, contact Velodyne Sales (sales@velodyne.com)

1. Typical accuracy refers to ambient wall test performance across most channels and may vary based on factors including but not limited to range, temperature and target reflectivity.

2. Operating power may be affected by factors including but not limited to range, reflectivity and environmental conditions.

3. Operating temperature may be affected by factors including but not limited to air flow and sun load.



CLASS 1 LASER PRODUCT

Copyright ©2018 Velodyne LiDAR, Inc. Specifications are subject to change. Other trademarks or registered trademarks are property of their respective owners.

**Analise estatística R&R da simulação do robô
Darwin OP**

AVALIAÇÃO DO SISTEMA DE MEDAÇÃO - TIMON 2.5

Laboratório de Robótica e Sistemas Autônomos - RoSA

Jéssica Lima Motta
Leonardo Mendes de Souza Lima
Miguel Felipe Nery Vieira
Vinícius José Gomes de Araujo Felismino

Introdução

Este documento tem como objetivo analisar o sistema de medição de dados coletados durante os testes realizados na etapa de corrida de revezamento do desafio 2.5C, utilizando o método de análise de variância (ANOVA).

Coleta dos Dados

Foram realizados 3 testes consecutivos em 4 máquinas diferentes, exibidas na Tabela 1.

Tabela 1: Máquina utilizadas por operador

	Máquina 1	Máquina 2	Máquina 3	Máquina 4
Nome	Jéssica	Leonardo	Miguel	Vinícius
Processador	i7-950H	FX 6300	i7-4790	i7-8550U
Memória	8 GB RAM	4 GB RAM	16 GB RAM	8 GB RAM
GPU	GTX 1660 Ti	GTX 750 Ti	GT 730	GeF MX150

Cada operador tornou-se responsável em registrar o tempo que cada robô Darwin-OP levava para realizar o seu percurso na corrida de revezamento. Os dados medidos encontram-se exibidos na Tabela 2.

Tabela 2: Dados Medidos por Operadores

Máquina	Robô	Teste	Tempo (s)	Máquina	Robô	Teste	Tempo (s)
1	1	1	62.210	3	1	1	66.589
1	1	2	67.055	3	1	2	67.673
1	1	3	65.496	3	1	3	66.132
1	2	1	68.529	3	2	1	68.500
1	2	2	64.352	3	2	2	63.283
1	2	3	67.405	3	2	3	68.312
1	3	1	68.063	3	3	1	64.767
1	3	2	69.005	3	3	2	69.628
1	3	3	64.155	3	3	3	63.055
1	4	1	66.685	3	4	1	66.536
1	4	2	65.966	3	4	2	66.139
1	4	3	67.553	3	4	3	68.835
2	1	1	66.311	4	1	1	62.480
2	1	2	67.995	4	1	2	65.503
2	1	3	65.864	4	1	3	64.498
2	2	1	67.928	4	2	1	69.168
2	2	2	62.444	4	2	2	63.142
2	2	3	66.577	4	2	3	67.336
2	3	1	65.796	4	3	1	65.806
2	3	2	68.764	4	3	2	68.634
2	3	3	66.984	4	3	3	62.443
2	4	1	66.772	4	4	1	67.633
2	4	2	64.137	4	4	2	65.668
2	4	3	68.172	4	4	3	67.806

Para avaliar a precisão do sistema de medição foi realizado um estudo de Repetibilidade e Reprodutibilidade (R&R) do sistema. Esse estudo compara a variação do sistema de medição com a variação total do processo. Se a variação do sistema de medição for grande em relação à variação das peças, o sistema provavelmente não proverá informações úteis e consequentemente não poderá fazer a distinção correta entre as mesmas.

Interpretação dos resultados obtidos

Para o cálculo dos componentes de variância é utilizado o procedimento ANOVA. A Tabela 3 exibe os graus de liberdade (Df), soma dos quadrados (Sum Sq), média dos quadrados (Mean Sq), valor da distribuição F, e o P-valor para cada uma das fontes.

Tabela 3: ANOVA com Dois Fatores com Interação

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
Robô	3	137.67	45.89	37.216	2.12e-05
Máquina	3	4.31	1.44	1.165	0.376
Robô:Máquina	9	11.10	1.23	0.963	0.487
Repetibilidade	32	40.95	1.28		
Total	47	194.02			

Caso o P-valor seja maior do que 0.05, o termo será considerado estatisticamente irrelevante. Desta forma, como o P-valor de Robô:Máquina é igual a 0.487, é gerada a Tabela 4 desconsiderando a interação. O elevado p-valor de Máquina (0.348) indica que essa fonte do sistema não é estatisticamente significativa e portanto pode ser desconsiderada caso haja o interesse em reduzir a variabilidade do sistema de medição (COMO..., 2020). Pode-se verificar também a quantidade de graus de liberdade disponível para estimar a repetibilidade do medidor, sendo recomendado um valor que esteja de 30 a 45 e no nosso caso igual a 41.

Tabela 4: ANOVA com Dois Fatores sem Interação

	Df	Sum Sq	Mean	F Value	Pr>F
Robô	3	137.67	45.89	36.147	1.37 e-11
Máquina	3	4.31	1.44	1.131	0.348
Repetibilidade	41	52.05	1.27		
Total	47	194.02			

Na tabela 5, chamada Gage R&R, temos a coluna de componentes da variância (Varcomp) e a coluna que representa o percentual de contribuição da variância do respectivo componente (%Contrib). Nota-se que a soma da repetibilidade com a reprodutibilidade corresponde a um total de 25.66% de variabilidade do sistema e isso permite concluir que o sistema de medição pode estar comprometido, uma vez que, este percentual deveria ser muito pequeno. O %Contrib peça a peça é de 74.34% e é responsável pela maior parte da variabilidade do sistema, quanto maior esse valor mais facilmente o sistema conseguiria distinguir as peças.

A %Contrib é baseada nas estimativas dos componentes da variância. Cada valor em Varcomp é dividido pela variação total e depois multiplicado por 100 (ESTUDOS..., 2020).

Tabela 5: Gage R&R

	Varcomp	%Contrib
Total Gage R&R	1.28339665	25.66
—Repetibilidade	1.26949803	25.38
—Reprodutibilidade	0.01389862	0.28
—Máquina	0.01389862	0.28
Peça a Peça	3.71826828	74.34
Variação Total	5.00166493	100

A tabela 6 nos fornece a referência final para aprovar ou rejeitar o sistema de medição. Esta tabela informa os valores calculados para as fontes de variação, desvios padrão e variáveis do estudo. Para encontrar os valores que estão na coluna variável do estudo, seguimos a recomendação da *Automotive Industry Action Group* (AIAG): multiplicamos o valor do desvio padrão por 6 em uma distribuição normal, já que dentro de 6 desvios padrão estão cerca de 99,73% dos dados (ESTUDOS..., 2020). Por fim, a coluna %Var do estudo divide a variável do estudo pela variação total para descobrir quanto da variação do sistema de medição representa da variação total do sistema. Abaixo da tabela, o número de categorias distintas informa quantas categorias diferentes o sistema de medição foi capaz de discriminar de acordo com as variações apresentadas.

Tabela 6: Avaliação das medições

	Desvio padrão	Var do estudo	%Var do estudo
Total Gage R&R	1.1328710	6.7972259	50.66
—Repetibilidade	1.1267200	6.7603202	50.38
—Reprodutibilidade	0.1178924	0.7073545	5.27
—Máquina	0.1178924	0.7073545	5.27
Peça a Peça	1.9282812	11.5696870	86.22
Variação Total	2.2364402	13.4186414	100

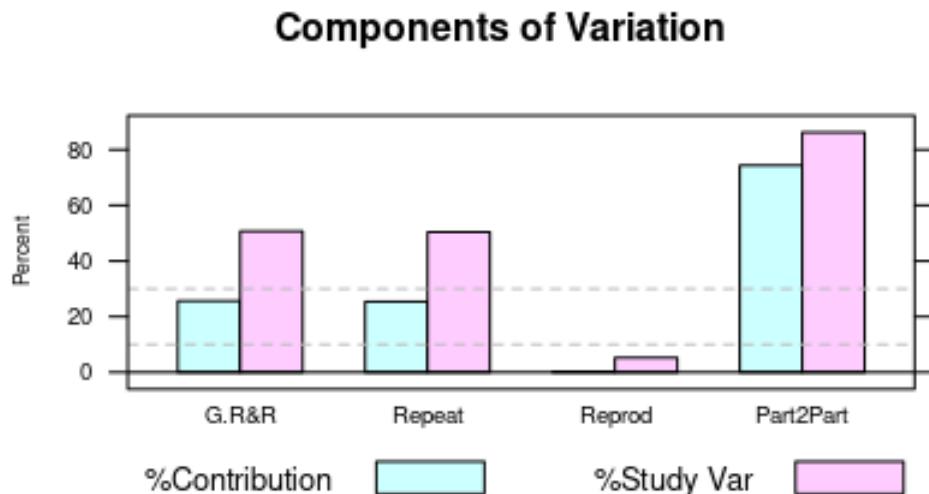
Número de categorias distintas - 2

Para que o sistema de medição seja considerado aceitável, é esperado que o mesmo possua valor menor que 10% em seu Gage R&R. O sistema de medição analisado neste documento está acima de 50% em seu Gage R&R total, logo, é classificado como inaceitável e deve ser melhorado. É recomendado pela AIAG que o numero de categorias distintas seja maior ou igual a 5 para uma ideal diferenciação. O valor calculado de 2, é considerado insuficiente.

A Figura 1 representa o gráfico dos componentes da variação onde cada agrupamento de barras representa uma origem de variação. A altura das barras, representa percentualmente, o quanto cada componente é responsável. As barras azuis representam o percentual de contribuição para a variação geral e as rosas a contribuição para a variação no estudo.

Em um bom sistema de medição, a maior parte de variação está em *Part-to-Part*(Peça a Peça). No nosso caso, há uma grande variação atribuída ao percentual do sistema de medição (G.R&R) e por isso será necessário corrigir o sistema de medição.

Figura 1: Componentes da variação



Já na figura 2 podemos verificar a amplitude da amostra por operador. Observe que segundo a tabela 1 temos quatro operadores diferentes e é exatamente isto que encontra-se disposto aqui. Cada ponto representa a diferença entre a maior e a menor leitura observada para cada uma das três medições realizadas em cada robô. Embora haja um certo padrão comportamental entre os quatro gráficos, valores muito distantes como o do robô 1 no operador 2 indicam a existência de dificuldades nesta medição.

Figura 2: Amplitude da Amostra por Máquina

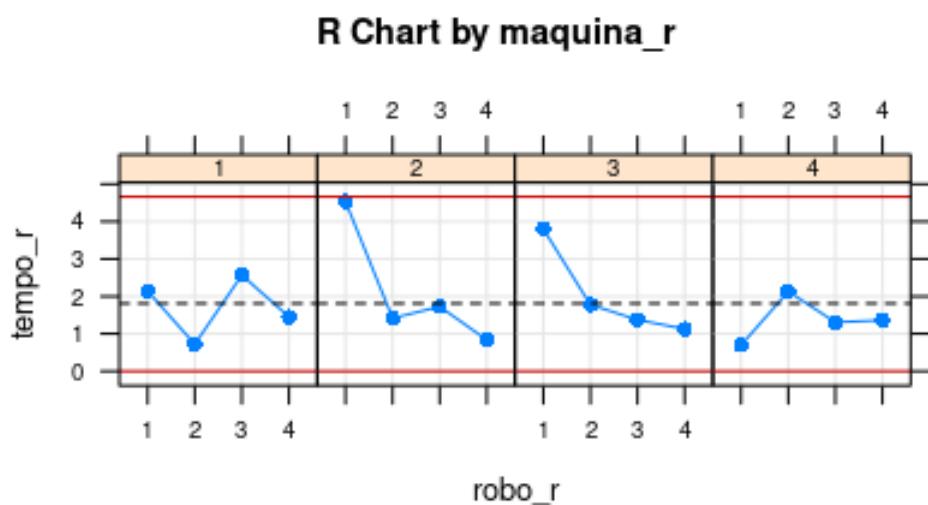
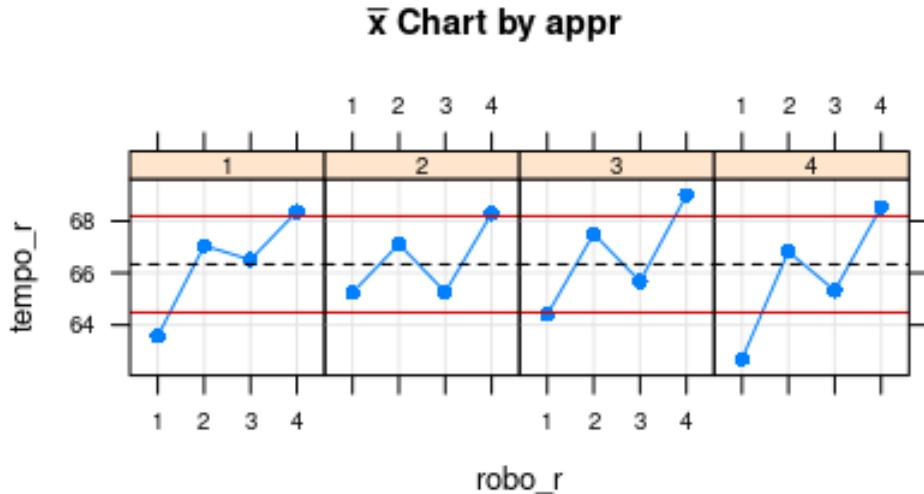


Figura 3: Média da amostra por máquina



Na Figura 3 é plotada a média das medições com o mesmo intuito do gráfico exibido na Figura 2. Mas neste caso a média é utilizada. Neste gráfico estão exibidas as médias do tempo para cada robô. Podemos observar que há um comportamento similar entre as máquinas 2 e 3, porém as máquinas 1 e 4 exibem médias diferentes do que se esperava para o sistema.

Na seção *tempo_r* by *robo_r* exibida na figura 4 encontram-se distribuídas em torno do ponto central (média) as leituras efetuadas para cada robô, sendo traçada uma reta ligando estas médias. Pode-se notar que existem alguns pontos distantes do ponto central, o que pode significar um erro de medição, uma medição ruim ou errada.

Figura 4: Tempo por Robô

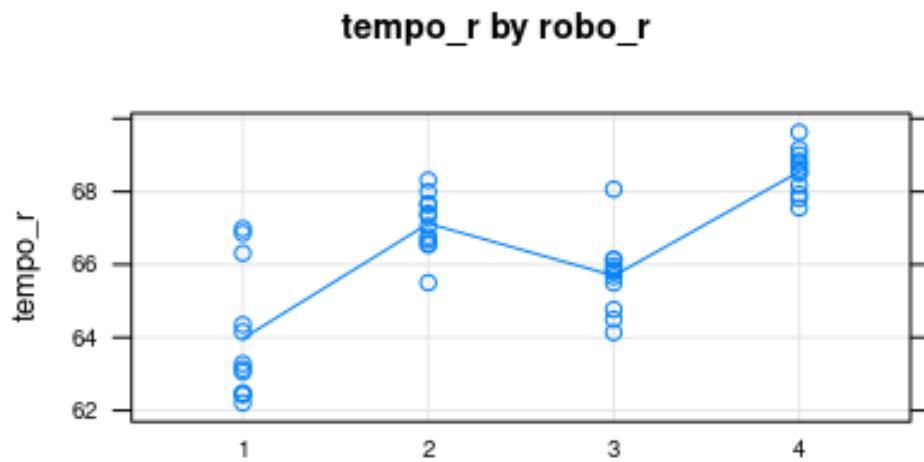
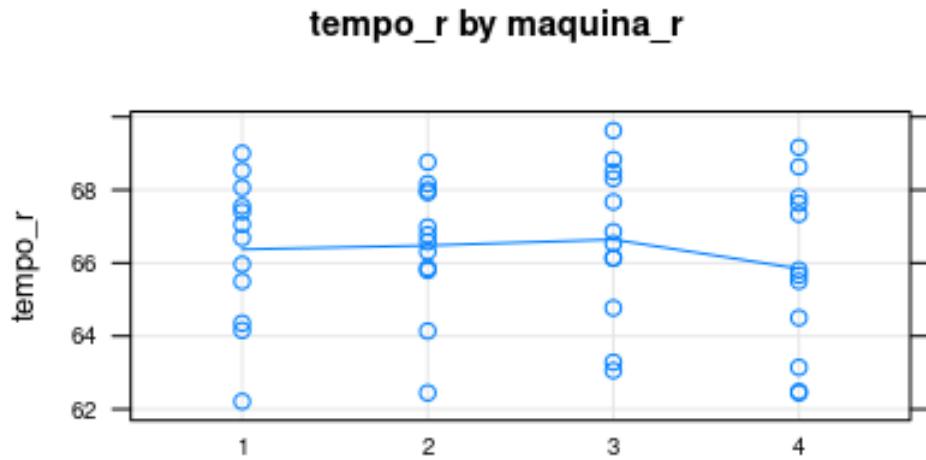
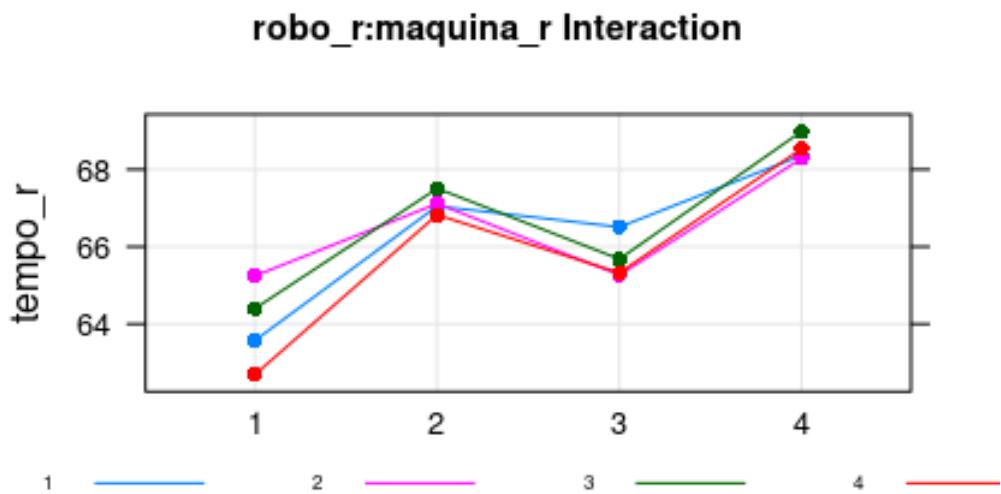


Figura 5: Tempo por máquina



Na Figura 5 cada um dos testes têm distribuídos em torno da linha azul central (a média) as leituras efetuadas. Note que as leituras se distribuem em torno das leituras em azul, o que significa que estão muito distantes do alvo o que representa um erro de leitura, uma leitura ruim ou errada. As medidas de tempo encontradas para cada um dos operadores encontram-se exibidos nesse gráfico, onde cada valor encontrado é representado por um círculo e uma reta passa por seus valores médios. Quanto menor o ângulo entre cada uma das máquinas melhor para o sistema, pois significa que não há grande diferença entre as capacidades de operação das máquinas.

Figura 6: Média da amostra por máquina



Na seção 6 está apresentada uma junção dos gráficos exibidos na Figura 3 a fim de fazer uma comparação entre as médias encontradas para cada robô. É usado para checar se existe uma relação entre máquina e um robô específico. Caso alguma máquina tenha

uma dificuldade com algum tipo de robô, será possível observar uma discrepância entre as linhas em determinado robô. No gráfico apresentado, verifica-se que há muita diferença entre cada máquina.

Conclusão

O valor encontrado na porcentagem de variação do estudo classifica o sistema de medição como inaceitável e a não conformidade com o numero de categorias distintas também é um fator que contribui negativamente com os resultados encontrados nos levando a concluir que algum dos operadores deve ter cometido algum erro durante as suas medições. É recomendada como possível solução a retomada dos dados com realização de novos testes.

REFERÊNCIAS

COMO Planejar, Analisar e Interpretar os Resultados de um Estudo de Gage RR Expandido. 2020. <<https://www.minitab.com/pt-br/Published-Articles/Como-Planejar,-Analizar-e-Interpretar-os-Resultados-de-um-Estudo-de-Gage-R-R-Expandido/>>, note = Accessed: 2020-07-20. Citado na página 3.

ESTUDOS de medição para dados contínuos. 2020. <<https://www.minitab.com/uploadedFiles/Documents/sample-materials/FuelInjectorNozzles-PT.pdf>>, note = Accessed: 2020-07-20. Citado 2 vezes nas páginas 3 e 4.

**Planejamento de Experimentos (DOE) -Helicóptero
de Papel (TIMON-HM)**

**PLANEJAMENTO DE EXPERIMENTOS (DOE) -
HELICÓPTERO DE PAPEL (TIMON-HM)**
Laboratório de Robótica e Sistemas Autônomos - RoSA

Autores:
Jéssica Lima Motta
Leonardo Mendes de Souza Lima
Miguel Felipe Nery Vieira
Vinícius José Gomes de Araujo Felismino

**Salvador
Bahia, Brasil**

Setembro de 2020

RESUMO

O presente documento tem como objetivo aplicar os conceitos de Planejamento de Experimento, do inglês [Design of Experiments \(DOE\)](#), a um modelo de helicóptero de papel. O propósito principal foi identificar quais são os fatores que mais influenciam seu tempo de voo e como estas variáveis podem melhorar o seu desempenho. Durante o processo, foi medido o seu tempo de voo em duas alturas diferentes, além disto, foram adicionados adesivos e um clipe em sua estrutura a fim de verificar a influência da variação destes parâmetros no resultado final. Para realizar o estudo estatístico dos dados obtidos durante os testes, foi utilizada a ferramenta R, uma linguagem de programação voltada à manipulação, análise e visualização de dados.

LISTA DE SÍMBOLOS E ABREVIATURAS

DOE Design of Experiments

SUMÁRIO

1 INTRODUÇÃO	4
2 PLANEJAMENTO DE EXPERIMENTO COM VÁRIOS FATORES	5
2.1 Caracterização do problema	5
2.2 Escolha dos fatores de influência e níveis, e listar restrições .	5
2.3 Seleção das variáveis de resposta	6
2.4 Determinação de um modelo de planejamento de experimento	6
2.5 Condução do experimento	6
2.6 Análise dos dados	6
2.7 Conclusões e recomendações	6
3 EXPERIMENTO	8
4 INTERPRETAÇÃO DOS RESULTADOS OBTIDOS	11
5 CONCLUSÃO	16
REFERÊNCIAS	17

1 INTRODUÇÃO

Diferentes métodos constituem a prática da melhoria contínua. É de suma importância que os administradores conheçam estas ferramentas para que haja sempre redução de desperdícios, aumento da eficiência e controle dos processos ([NORTEGUBISIAN, 2020](#)).

Planejamento de Experimentos ([DOE](#)) é uma das técnicas utilizadas para estudar um produto ou processo, e assim, identificar os fatores que mais influenciam seu comportamento. Através deste método deve-se obter a mais otimizada configuração para a construção da peça ou elaboração do procedimento ([PETENATE, 2020](#)).

O desenvolvimento de um experimento bem executado deve explicitar os fatores-chave do processo, assim como a combinação dos fatores que fazem o processo funcionar de maneira aceitável. A variabilidade do processo, ou seja, a diferença entre o que esperamos de algo e o que realmente acontece também é um ponto a ser observado pelo executor do [DOE](#).

O resultado que determina uma característica ou elemento do experimento é chamado de **variável de resposta**. Por ser um método de abordagem repetitiva, é necessário realizar ciclos de testes para alcançar um bom resultado. Estes ciclos devem possuir três etapas: **Rastreamento** - fase de delimitação de variáveis e do campo de atuação; **Projeto factorial completo** - fase de combinação de fatores e níveis de fatores e **Projeto de superfície** - Modelagem dos resultados obtidos.

O [DOE](#) pode ser aplicado em duas situações: planejamento de experimentos e correção de processos defeituosos. Um processo desenvolvido desde o início com esta aplicação garante que sua produção e gestão sejam sempre melhoradas e tenham custos e tempo reduzidos. É uma ferramenta de melhoria contínua bastante eficaz, desde que se tome os devidos cuidados com as etapas do experimento ([PETENATE, 2020](#)).

2 PLANEJAMENTO DE EXPERIMENTO COM VÁRIOS FATORES

Antes de realizar o experimento foi necessário um estudo prévio sobre o **DOE**. Neste planejamento determina-se quais as configurações são eficientes para determinado processo. Segundo (**COLEMAN; MONTGOMERY, 1993**) as etapas para o desenvolvimento de um Planejamento de Experimento na Indústria devem ser as seguintes:

- Caracterização do problema;
- Escolha dos fatores de influência e níveis, e listar restrições;
- Seleção das variáveis de resposta;
- Determinação de um modelo de planejamento de experimento;
- Condução do experimento;
- Análise dos dados;
- Conclusões e recomendações.

Nesta seção será avaliado cada item mencionado.

2.1 Caracterização do problema

Nesta etapa é necessário desenvolver as ideias acerca do problema e sobre os objetivos específicos do experimento. É fundamental a participação de toda a equipe de qualidade, engenharia, clientes e operadores, para fazer um relato preciso sobre o problema. Dessa forma será possível uma melhor compreensão do processo e a busca por uma provável uma solução para o problema.

2.2 Escolha dos fatores de influência e níveis, e listar restrições

Os fatores de influência e os níveis são escolhidos após se obter uma boa definição do problema e a elaboração do objetivo do experimento. O responsável pelo experimento deverá determinar quais fatores devem variar, os intervalos nos quais esses fatores variarão e os níveis em que cada rodada será realizada. Quando se tem por objetivo fazer uma varredura dos fatores ou caracterização do processo, e este ainda não está amadurecido, melhor manter baixo o número de níveis, geralmente dois (**ROZENFELD, 2014**). É fundamental a investigação de todos os fatores que possam ser importantes.

É necessário listar e rotular as interações conhecidas e supostas, e as restrições no experimento, como métodos de aquisição de dados, duração, materiais, facilidade de alterar a variável de controle, tipo de experimento, etc.

2.3 Seleção das variáveis de resposta

Nesta etapa o responsável pelo experimento irá escolher a(s) variável/variáveis que fornece informação útil sobre o processo. Geralmente, tem-se como variável de resposta a média ou o desvio padrão, ou ambos, da característica medida. A capacidade do medidor também interferirá nessa etapa pois, caso seja baixa, apenas efeitos grandes serão detectados, ou será necessária replicação do experimento. O embasamento para selecionar a variável resposta pode vir da teoria, de especialistas ou da experiência.

2.4 Determinação de um modelo de planejamento de experimento

A escolha do planejamento leva em consideração o tamanho da amostra, seleção de uma ordem adequada de rodadas para as tentativas experimentais, e se a formação de blocos ou outras restrições de aleatorização estão envolvidas ([DÁVILA, 2018](#)). Podem-se citar dentre os métodos de planejamento: Fatorial, Completely aleatorizado com um único fator, Fatorial 2^k em blocos, Fatorial 2^k fracionário, Blocos aleatorizados, Blocos incompletos balanceados, blocos incompletos parcialmente balanceados, Quadrados latinos, Quadrados de Youden, Hierárquico e Superfície de resposta ([MONTGOMERY; RUNGER, 2013](#)).

2.5 Condução do experimento

Nesta etapa é de extrema importância o monitoramento do processo de forma a garantir que seja feito de acordo com o que foi planejado. Os erros no procedimento experimental que ocorrem durante a condução do experimento destruirão a validade do mesmo, requerendo uma repetição dos testes.

2.6 Análise dos dados

Para analisar os dados deve-se empregar métodos estatísticos como ANOVA, regressão, plots e t test,e com isso obter resultados e conclusões. Se o experimento foi planejado e executado corretamente, logo, não serão encontradas dificuldades nas análises decorrentes do tipo de método estatístico aplicado.

2.7 Conclusões e recomendações

Após os dados serem analisados, o experimento deverá apresentar conclusões práticas sobre os resultados e recomendar uma ação. Nessa etapa são usados métodos gráficos para apresentar os resultados a outras pessoas envolvidas no processo. Devem ser realizadas sequências de acompanhamento, a análise dos resíduos decorrente do processo e testes de

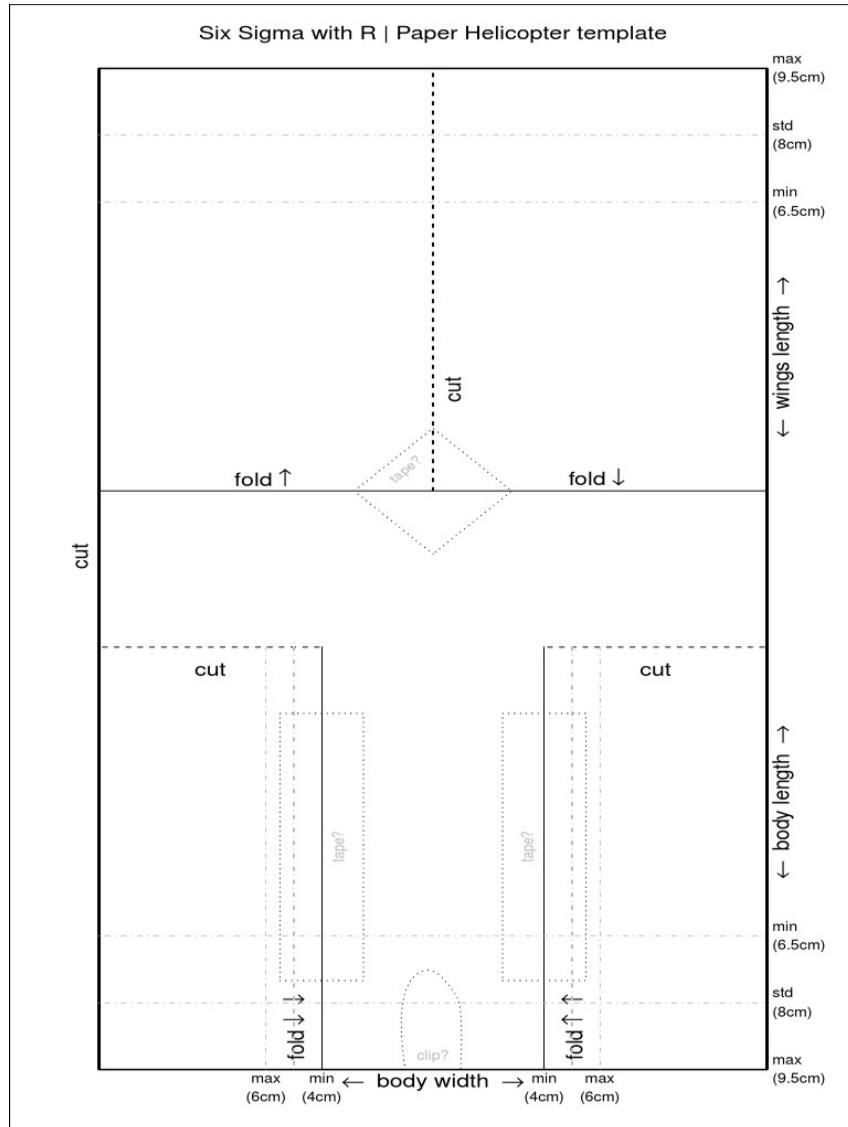
confirmação, para validar as conclusões do experimento.

3 EXPERIMENTO

Para aplicar os conceitos vistos na seção 2, foi proposto um desafio que consiste em modelar um experimento contendo um helicóptero de papel. O objetivo deste desafio é observar como a saída desejada, neste caso o tempo de voo, está relacionada às variáveis de entrada: altura, clipe e adesivos.

Para conceber o protótipo do helicóptero para o estudo, foi utilizado o modelo proposto pela metodologia *SixSigma*, conforme visto na Figura 1.

Figura 1: Modelo do helicóptero de papel.

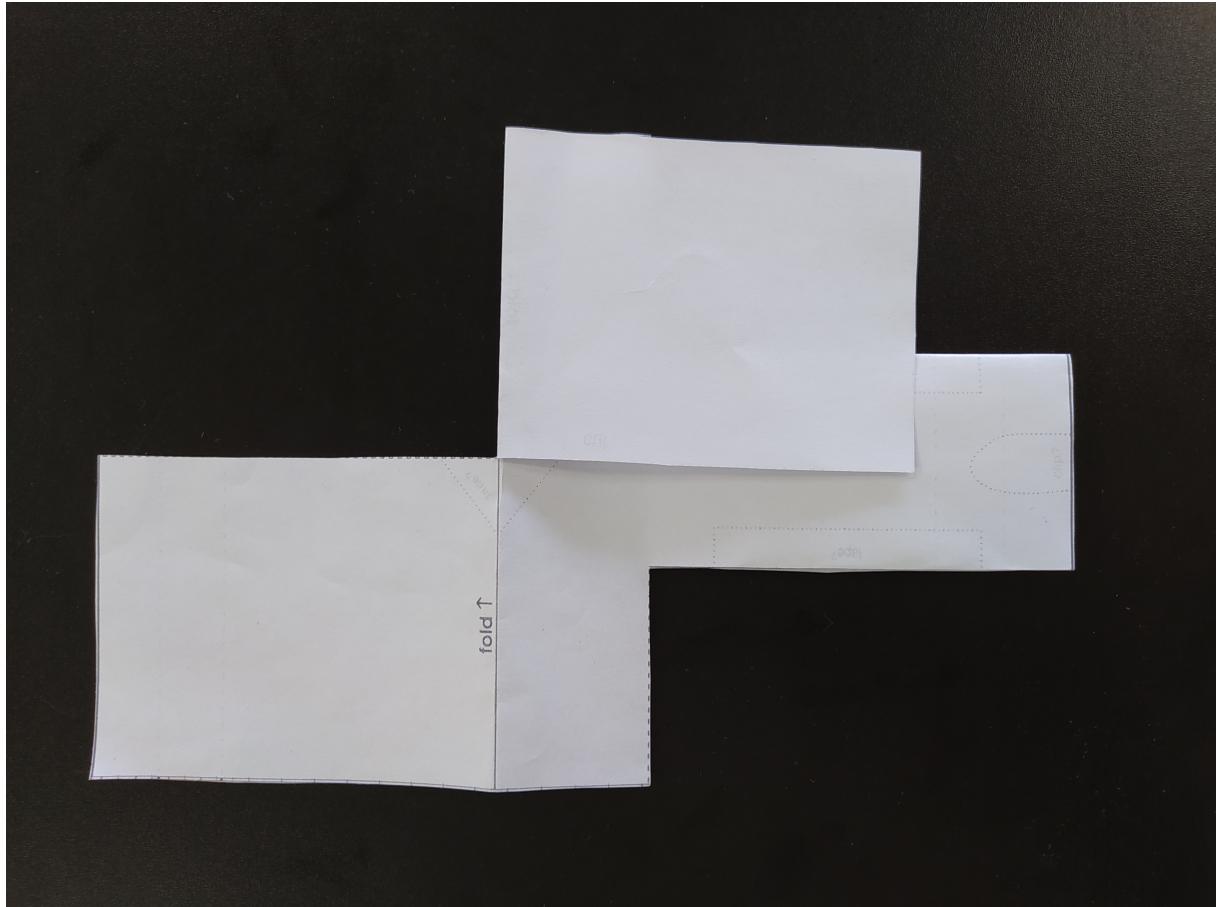


Fonte: ([LOPEZ, 2018](#))

Seguindo as recomendações do *template*, foi obtido como modelo de configuração inicial o helicóptero visto na Figura 2. Este apresenta as asas e o corpo com o comprimento

máximo de 0,095 m que não deverá ser alterado. O seu tempo de voo é medido desde o momento em que é lançado da altura definida até o momento em que atinge o solo.

Figura 2: Helicóptero de papel.



Fonte: Autoria própria.

Para realizar os testes foram considerados alguns fatores que influenciam no tempo de voo, conforme vistos na Tabela 1.

Tabela 1: Fatores considerados para alterar a estrutura.

Fatores	Configuração atual	Alteração permitida
Comprimento (asa e corpo) (m)	0,095	Não
Clipe	Não	Sim
Altura (m)	2,10	1,30
Adesivo (Asa)	Não	Sim
Adesivo (Corpo/Esquerdo)	Não	Sim
Adesivo (Corpo/Direito)	Não	Sim

Fonte: Autoria própria.

Por fim, foi construída a Tabela 2 que contém o tempo de voo para cada uma das pos-

síveis combinações dos fatores. Para as variáveis *clipe*, *Ad_top*, *Ad_esquerda* e *Ad_direita* o simbolo “+” indica a sua presença enquanto o “-” representa a sua ausência, já para a variável *Altura* o “+” retrata sua configuração inicial de 2,10 metros e o “-” representa a altura de 1,30 metros. O próximo passo é utilizar a ferramenta R para realizar o estudo de planejamento de experimentos (**DOE**) e analisar qual das configurações está exercendo uma maior influência no experimento, que será discutido na seção 4.

Tabela 2: Dados do experimento.

Altura	Clipe	Ad_top	Ad_esquerda	Ad_direita	Tempo
-	-	-	-	-	1,27
+	-	-	-	-	1,57
-	+	-	-	-	1,10
+	+	-	-	-	1,70
-	-	+	-	-	1,31
+	-	+	-	-	1,82
-	+	+	-	-	1,30
+	+	+	-	-	1,75
-	-	-	+	-	1,36
+	-	-	+	-	1,92
-	+	-	+	-	1,52
+	+	-	+	-	1,71
-	-	+	+	-	1,40
+	-	+	+	-	1,83
-	+	+	+	-	1,32
+	+	+	+	-	1,74
-	-	-	-	+	1,50
+	-	-	-	+	1,91
-	+	-	-	+	1,23
+	+	-	-	+	1,55
-	-	+	-	+	1,42
+	-	+	-	+	2,04
-	+	+	-	+	1,35
+	+	+	-	+	1,68
-	-	-	+	+	1,44
+	-	-	+	+	1,58
-	+	-	+	+	1,25
+	+	-	+	+	1,73
-	-	+	+	+	1,32
+	-	+	+	+	1,86
-	+	+	+	+	1,17
+	+	+	+	+	1,63

Fonte: Autoria própria.

4 INTERPRETAÇÃO DOS RESULTADOS OBTIDOS

O modelo de regressão linear encontrado, considerando a interação entre dois elementos, é disposto a seguir.

```
##  
## Call:  
## lm(formula = tempo ~ (altura + clipe + ad_top + ad_esquerda +  
##     ad_direita) + altura * clipe + altura * ad_top + altura *  
##     ad_esquerda + altura * ad_direita + clipe * ad_top + clipe *  
##     ad_esquerda + clipe * ad_direita + ad_top * ad_esquerda +  
##     ad_top * ad_direita + ad_esquerda * ad_direita,  
##     data = helicoptero)  
##  
## Residuals:  
##      Min       1Q   Median      3Q      Max  
## -0.180625 -0.055312 -0.009375  0.059687  0.120625  
##  
## Coefficients:  
##                                     Estimate Std. Error t value Pr(>|t|)  
## (Intercept)                 1.24188   0.07069 17.569 6.99e-12 ***  
## altura+                   0.42125   0.07903  5.330 6.77e-05 ***  
## clipe+                    -0.04125   0.07903 -0.522  0.60885  
## ad_top+                     0.07875   0.07903  0.996  0.33386  
## ad_esquerda+                0.17625   0.07903  2.230  0.04040 *  
## ad_direita+                 0.19125   0.07903  2.420  0.02779 *  
## altura+:clipe+              -0.03250   0.07069 -0.460  0.65186  
## altura+:ad_top+              0.09500   0.07069  1.344  0.19771  
## altura+:ad_esquerda+        -0.04000   0.07069 -0.566  0.57932  
## altura+:ad_direita+         -0.02000   0.07069 -0.283  0.78085  
## clipe+:ad_top+              -0.03750   0.07069 -0.531  0.60304  
## clipe+:ad_esquerda+         0.06750   0.07069  0.955  0.35382  
## clipe+:ad_direita+          -0.14250   0.07069 -2.016  0.06092 .  
## ad_top+:ad_esquerda+        -0.13500   0.07069 -1.910  0.07425 .  
## ad_top+:ad_direita+         -0.00500   0.07069 -0.071  0.94448  
## ad_esquerda+:ad_direita+   -0.21000   0.07069 -2.971  0.00901 **  
## ---  
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```

## 
## Residual standard error: 0.09996 on 16 degrees of freedom
## Multiple R-squared:  0.9161, Adjusted R-squared:  0.8375
## F-statistic: 11.65 on 15 and 16 DF,  p-value: 6.57e-06

```

Pode-se observar que, para o nosso modelo, as variáveis que possuem relevância estatística, ou seja $\text{Pr} < 0.05$ são: altura ($\text{Pr} = 6.77\text{e-}05$), ad_esquerda ($\text{Pr} = 0.04040$), ad_direita ($\text{Pr} = 0.02779$) e ad_esquerda:ad_direita ($\text{Pr} = 0.00901$).

Considerando as variáveis que possuem relevância estatística, a equação linear que representa o modelo é descrita da seguinte forma:

$$\begin{aligned} tempo &= \text{média}(tempo) + \frac{\text{coef}(altura)}{2} \text{altura} + \frac{\text{coef}(ad_esquerda)}{2} ad_esquerda + \\ &\quad \frac{\text{coef}(ad_direita)}{2} ad_direita + \frac{\text{coef}(ad_esquerda : ad_direita)}{2} ad_esquerda : ad_direita \end{aligned}$$

Desta forma, fazendo as devidas substituições, temos que:

$$\begin{aligned} tempo &= 1.54 + \frac{0.42125}{2} \text{altura} + \frac{0.17625}{2} ad_esquerda + \frac{0.19125}{2} ad_direita + \\ &\quad \frac{(-0.21)}{2} ad_esquerda : ad_direita \end{aligned}$$

, logo:

$$\begin{aligned} tempo &= 1.54 + 0.210625 \text{altura} + 0.088125 ad_esquerda + 0.095625 ad_direita - \\ &\quad 0.105 ad_esquerda : ad_direita \end{aligned} \tag{4.1}$$

É fácil de verificar na equação 4.1 que as variáveis *altura*, *ad_direita* e *ad_esquerda* influenciam positivamente no tempo de voo (possuem coeficientes positivos) enquanto a interação entre as variáveis *ad_esquerda:ad_direita* influencia negativamente (possui coeficiente negativo). Desta forma, considerando que nossas variáveis de entrada assumam apenas valores de -1 ou 1, para encontrar o maior valor de tempo de voo devemos atribuir valor positivo às variáveis *altura*, *ad_direita* e *ad_esquerda* e valor negativo à interação *ad_esquerda:ad_direita*, o que resulta em:

$$tempo_{\max} = 1.54 + 0.2106 * (1) + 0.0881 * (1) + 0.0931 * (1) - 0.105 * (-1) = 2.04 \text{seg}$$

De forma análoga, para encontrar o menor valor de tempo de voo devemos atribuir valor negativo às variáveis *altura*, *ad_direita* e *ad_esquerda* e valor positivo à interação *ad_esquerda:ad_direita*, resultando em :

$$tempo_min = 1.54 + 0.2106*(-1) + 0.0881*(-1) + 0.0931*(-1) - 0.105*(1) = 1.15seg$$

Os valores esperados para cada teste de lançamento do helicóptero (*Tempo*), bem como os previstos pelo modelo de regressão linear (*Tempo_p*) e seus respectivos resíduos podem ser visualizados na Tabela 3.

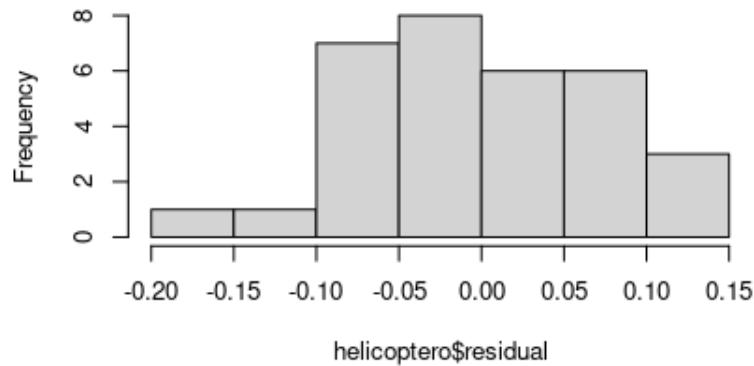
Tabela 3: Valores Previstos e Resíduos

Tempo	Tempo_p	Resíduo
1,27	1,241875	0,028125
1,57	1,663125	-0,093125
1,1	1,200625	-0,100625
1,7	1,589375	0,110625
1,31	1,320625	-0,010625
1,82	1,836875	-0,016875
1,3	1,241875	0,058125
1,75	1,725625	0,024375
1,36	1,418125	-0,058125
1,92	1,799375	0,120625
1,52	1,444375	0,075625
1,71	1,793125	-0,083125
1,4	1,361875	0,038125
1,83	1,838125	-0,008125
1,32	1,350625	-0,030625
1,74	1,794375	-0,054375
1,5	1,433125	0,066875
1,91	1,834375	0,075625
1,23	1,249375	-0,019375
1,55	1,618125	-0,068125
1,42	1,506875	-0,086875
2,04	2,003125	0,036875
1,35	1,285625	0,064375
1,68	1,749375	-0,069375
1,44	1,399375	0,040625
1,58	1,760625	-0,180625
1,25	1,283125	-0,033125
1,73	1,611875	0,118125
1,32	1,338125	-0,018125
1,86	1,794375	0,065625
1,17	1,184375	-0,014375
1,63	1,608125	0,021875

Fonte: Autoria própria.

De posse desses valores, para garantir a veracidade do modelo de regressão linear encontrado deve-se então realizar a análise dos seus resíduos, os quais espera-se possuírem distribuição normal e aleatoriedade em torno da regressão obtida. A Figura 3 exibe o histograma dos resíduos calculados e pode-se observar a distribuição normal dos valores, conforme o esperado.

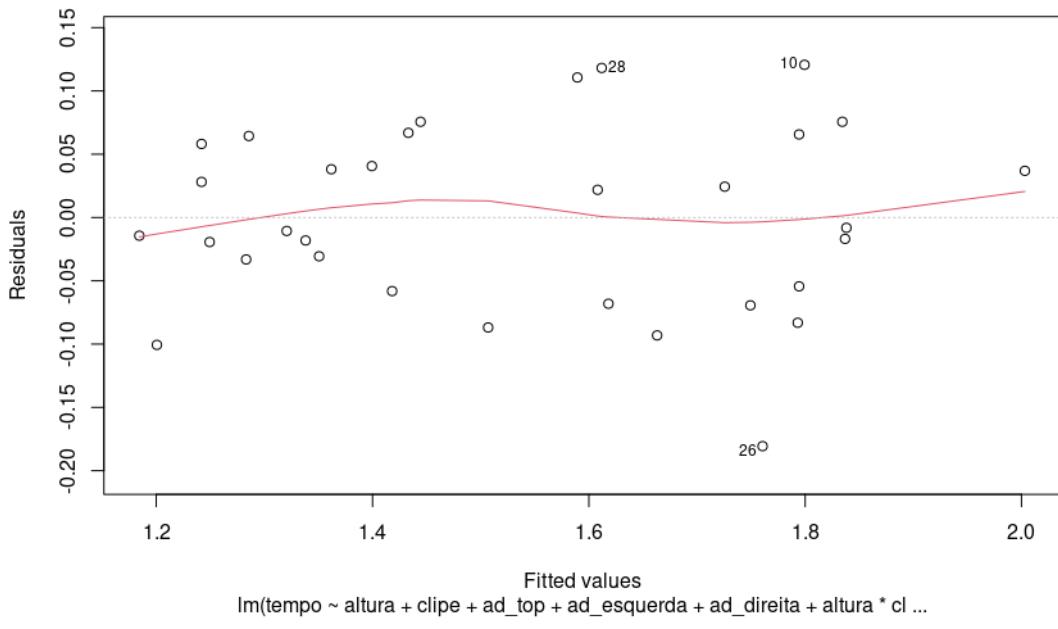
Figura 3: Histograma dos resíduos



Fonte: Autoria própria.

Ao usar a função **plot()** para o nosso modelo, o primeiro gráfico gerado é o *Residuals vs Fitted*, exibido na Figura 4, que dá uma indicação se há padrões não-lineares nos resíduos. Pode-se verificar na figura que o nosso modelo exibe uma regressão linear através de um certo número dos pontos, o que o valida sobre este aspecto.

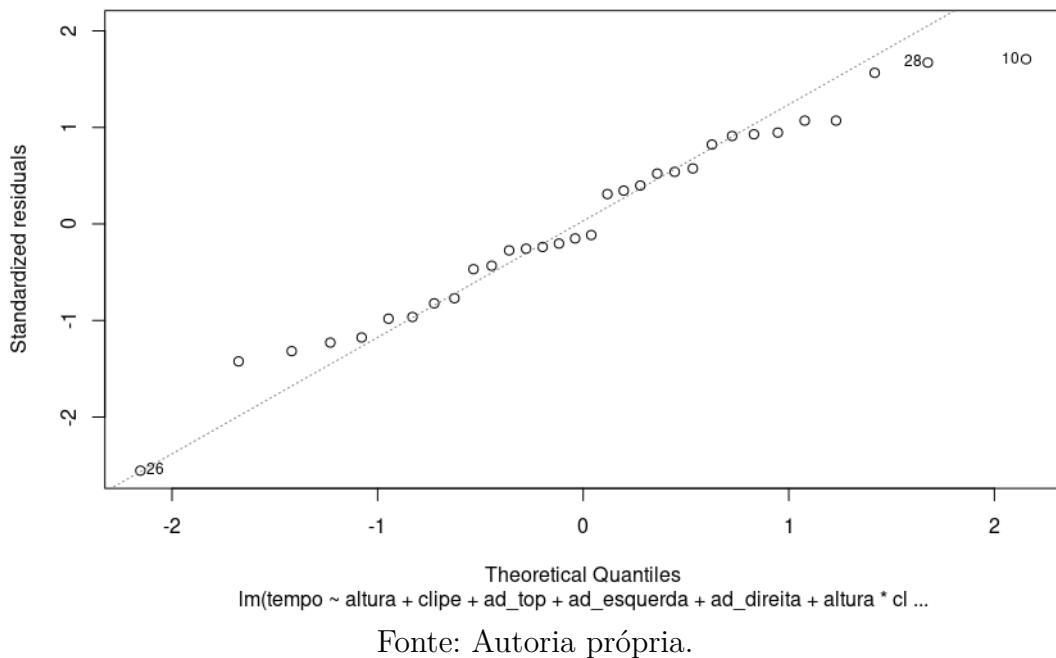
Figura 4: *Residuals vs Fitted*



Fonte: Autoria própria.

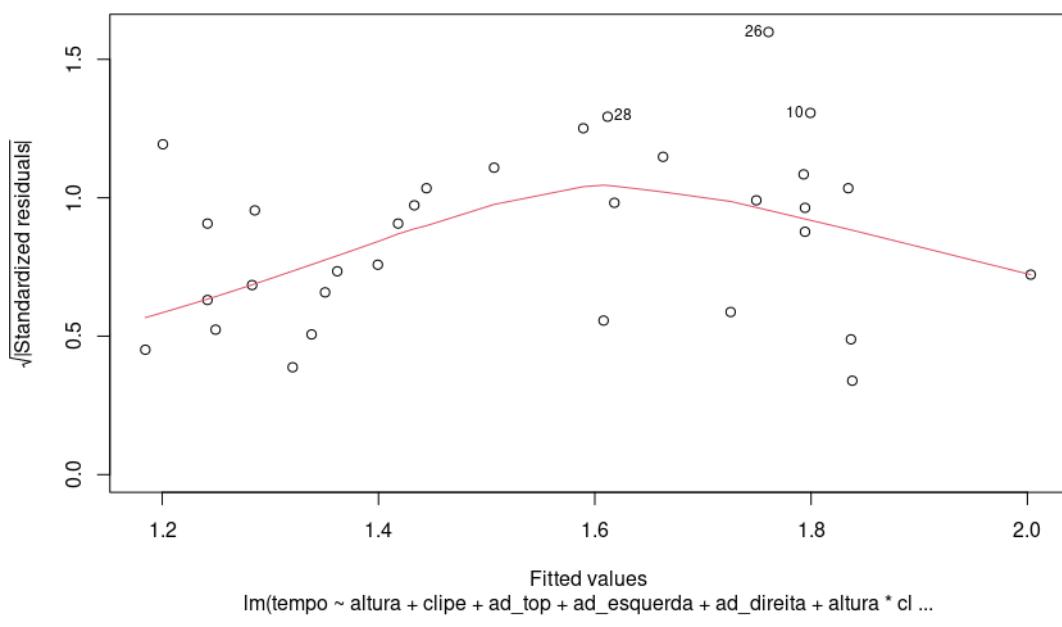
Uma outra forma de verificar a distribuição normal dos resíduos é através da Normal Q-Q. Podemos visualizar na Figura 5 a Normal Q-Q para o nosso modelo, na qual os resíduos seguem próximos à linha reta em diagonal, sendo uma boa indicação de que encontram-se normalmente distribuídos.

Figura 5: Normal Q-Q



É necessário verificar também se os resíduos possuem homocedasticidade, ou seja possuam variância comum ao longo da regressão. Pode-se observar no gráfico exibido na Figura 6 que os resíduos apresentam-se aleatoriamente pela linha, sem concentrar-se nem ao topo nem abaixo da mesma, comportamento que se assemelha ao esperado.

Figura 6: *Scale-Location*



5 CONCLUSÃO

Com base na pesquisa elaborada neste relatório, foi possível constatar que o experimento com helicóptero de papel possui eficiência na implementação dos conceitos relativos ao **DOE**. Durante o processo experimental pôde-se observar a importância de uma escolha acertada para as variáveis de entrada e saída, bem como da aleatoriedade de cada teste ao evitar ao máximo, por exemplo, a interferência do vento durante a coleta de dados.

A partir da análise realizada é possível então inferir que o modelo de regressão linear encontrado na equação 4.1 é válido para o helicóptero de papel utilizado, com as variáveis *altura*, *ad_direita* e *ad_esquerda* possuindo relevância para a resposta observada na variável de saída (*tempo*), sendo sugerido a otimização destes fatores para a construção de um helicóptero que possua um tempo de voo maior.

REFERÊNCIAS

- COLEMAN, D. E.; MONTGOMERY, D. C. A systematic approach to planning for a designed industrial experiment. *Technometrics*, Taylor & Francis Group, v. 35, n. 1, p. 1–12, 1993. Citado na página 5.
- DÁVILA, V. H. V. H. L. *Experimentos planejados para a melhoria do processo*. 2018. <<https://www.ime.unicamp.br/~hlachos/planejamento.pdf>>, Accessed: 2020-09-21. Citado na página 6.
- LOPEZ, E. *Design of the Paper Helicopter*. 2018. <<https://cran.r-project.org/web/packages/SixSigma/vignettes/HelicopterInstructions.pdf>>, Accessed: 2020-09-18. Citado na página 8.
- MONTGOMERY, D. C.; RUNGER, G. C. Estatística aplicada e probabilidade para engenheiros, 5^a. Ed. Rio de Janeiro: Editora LTC, 2013. Citado na página 6.
- NORTEGUBISIAN. *O que é design of experiments e como aplicar!* 2020. <<https://www.nortegubisian.com.br/blog/entenda-o-que-e-design-of-experiments-e-como-aplicar>>, Accessed: 2020-09-18. Citado na página 4.
- PETENATE, M. *Entenda o que é DOE - Design of experiments*. 2020. <<https://www.escolaedti.com.br/entenda-o-que-e-doe>>, Accessed: 2020-09-18. Citado na página 4.
- ROZENFELD, H. *Planejamento de Experimentos (DOE) - USP*. 2014. <<http://www5.eesc.usp.br/portaldeconhecimentos/index.php/por/Conteudo/Planejamento-de-Experimentos-DOE>>, Accessed: 2020-09-21. Citado na página 5.

**Artigo Manipulador Robótico TIMON-HM- Evento
SAPCT 2020**

PROJETO E SIMULAÇÃO DE UM MANIPULADOR ROBÓTICO COM 5 GRAUS DE LIBERDADE E SISTEMA DE VISÃO INTEGRADO

Jéssica Lima Motta¹, Leonardo Mendes de Souza Lima², Miguel Felipe Nery Vieira³, Vinicius José Gomes de Araújo Felismino⁴; Tiago Pereira de Souza⁵, Lucas Cruz da Silva⁶

¹Bolsista; Programa novos talentos - competência robótica e sistemas autônomos; jessica.motta@fbter.org.br

²Bolsista; Programa novos talentos - competência robótica e sistemas autônomos; leonardo.lima@fbter.org.br

³Bolsista; Programa novos talentos - competência robótica e sistemas autônomos; miguel.vieira@fbter.org.br

⁴Bolsista; Programa novos talentos - competência robótica e sistemas autônomos; vinicius.felismino@fbter.org.br

⁵Engenheiro Eletricista; Centro Universitário SENAI CIMATEC; Salvador-BA; tiago.souza@fieb.org.br

⁶Mestre em Engenharia Elétrica; Centro Universitário SENAI CIMATEC; Salvador-BA; lucas.cs@fieb.org.br

RESUMO

Timon-HM é um manipulador projetado, simulado e construído com o intuito de atender às demandas relacionadas ao reconhecimento de marcadores visuais e acionamento de interruptores, chaves ou botões. Posteriormente, este mesmo manipulador robótico será integrado ao robô *WARTHOG*, desenvolvido pela *Clearpath Robotics*, com o propósito de realizar a atividade de investigação em ambiente externo, desta vez participando de uma simulação de busca e desarme de bombas. O pacote de simulação do manipulador foi construído através do software *Gazebo* aliado ao *MoveIt* e a ferramenta de visualização *Rviz*. Testes foram realizados considerando diferentes posições e orientações do alvo, verificando-se a capacidade do sistema de atender à demanda solicitada.

PALAVRAS-CHAVE: Manipulador; ROS; Gazebo; Reconhecimento visual; MoveIt.

1. INTRODUÇÃO

A distinção entre um robô industrial e automação fixa ainda está indefinida, porém tem-se que se um dispositivo mecânico é flexível à programação podendo realizar diversas aplicações, então, provavelmente este é um robô industrial.¹ O aumento do uso de robôs industriais é em consequência do seu custo que vem sofrendo grande declínio nas últimas décadas.² Hoje, além de serem mais eficientes, estão mais rápidos, flexíveis e precisos. A utilização de robôs nas tarefas permite mais segurança aos seres humanos e confere maior confiabilidade, repetibilidade e qualidade no trabalho.

Muitas pesquisas vêm sendo realizadas na área de manipuladores robóticos. Em (HERNANDEZ-MENDEZ et al., 2017) é descrito o desenvolvimento de um manipulador robótico com 3 DoF (*Degrees of Freedom* - Graus de Liberdade) e dois dedos independentes. Este robô foi desenvolvido com o propósito de manipular objetos, cujas localizações são conhecidas e transportá-los de uma localidade para outra utilizando ROS. Este trabalho serviu de base para a concepção e modelagem de manipuladores como o Timon-HM. O manipulador descrito neste trabalho foi desenvolvido com o intuito de complementar o rol de estudos dedicados a manipuladores robóticos autônomos. Utilizou-se o *MoveIt* para tratar do planejamento de trajetória e o *Rviz* como ferramenta de visualização.³

O propósito deste projeto é fazer com que este braço robótico possua a capacidade de reconhecer um marcador por meio de uma câmera RGB e em sequência realizar intervenções no ambiente. A escolha do framework *Robot Operating System* (ROS)⁴ se deu por conta de o mesmo possuir suficiente conjunto de bibliotecas e ferramentas, código aberto e uma grande comunidade disponível na web. Para o componente de reconhecimento, a biblioteca *ArUco* ofereceu todos os dicionários e recursos para a implementação do código de detecção. Ao final do projeto, é esperado que este dispositivo seja capaz de realizar todas as funções projetadas tanto em ambiente simulado quanto no mundo real. Consequentemente, apresentará, como benefícios, a capacitação de profissionais pertencentes à área de desenvolvimento e operação de manipuladores robóticos e também contribuirá para a criação de outros manipuladores com características e objetivos semelhantes.

2. METODOLOGIA

Este trabalho tem caráter teórico e experimental. A técnica de modelagem e simulação adotada permite criar, em computadores, ambientes virtuais os quais imitam o comportamento de praticamente qualquer tipo de sistema.⁵

Todos os desenhos foram feitos em 3D com a ajuda do software *OnShape*⁶, um sistema CAD 2D/3D em nuvem que permite que todos em uma equipe trabalhem juntos usando apenas um navegador da Internet.⁷ Em seguida, um arquivo URDF foi construído contendo todos os elementos presentes no robô, incluindo características de sensores e parâmetros de montagem.

Os modelos virtuais do manipulador e do laboratório foram utilizados para simular a execução de tarefas no *Gazebo*⁸, *Rviz*⁹ e *MoveIt*¹⁰, simuladores de robótica capazes de realizar cálculos e captar informações a respeito do ambiente virtual. Para resolver a cinemática inversa, utilizou-se no *MoveIt* o plugin TRAC-IK, um método alternativo ao habitual uso da inversa Jacobiana. Este método se adequa bem a manipuladores que possuem limitações em suas juntas, ao contrário de algoritmos baseados no teorema de Newton. Foi utilizado o OMPL (*Open Motion Planning Library*), uma coleção de algoritmos de planejamento de movimentação geralmente usada no *MoveIt*.

Testes em ambiente simulado possibilitam a previsão de problemas que possam surgir no mundo real e assim solucionar virtualmente complicações relacionadas ao limite de movimentos em cada junta, evitando que a estrutura física seja danificada.

Com o fim da realização dos testes virtuais, o modelo será implementado fisicamente e controlado pelo ROS, framework empregado no manipulador.

3. RESULTADOS E DISCUSSÃO

Através dos softwares mencionados, foram definidas as principais características presentes no manipulador. O alcance máximo de 979 mm de distância indica até onde o braço pode chegar carregando na ponta um peso que não pode ultrapassar 1,94 kg. A tensão de operação de 24 V segue as especificações presentes nos motores Dynamixel empregados. Por meio dos softwares de simulação citados, foi possível encontrar o peso estimado de 7,38 kg para o manipulador que pode ser visto na Figura 1 (a). Na imagem, o mesmo está em cima da mesa de operação, em posição inicial e de frente para o seu objetivo.

Foram realizados alguns testes de planejamento de trajetória e resultados promissores para o cumprimento da tarefa foram encontrados como o descrito na Figura 1 (b). Uma sequência de poses foi enviada, planejada e executada a partir da classe *MoveGroupInterface*, que proporciona diversas operações e configurações de objetivos para juntas ou poses, planeja movimentos e é capaz de adicionar objetos ao ambiente e ao robô. A imagem da câmera foi extraída e é exibida na Figura 1 (c), sendo utilizada para determinação da pose do botão em relação ao *end effector* do manipulador.

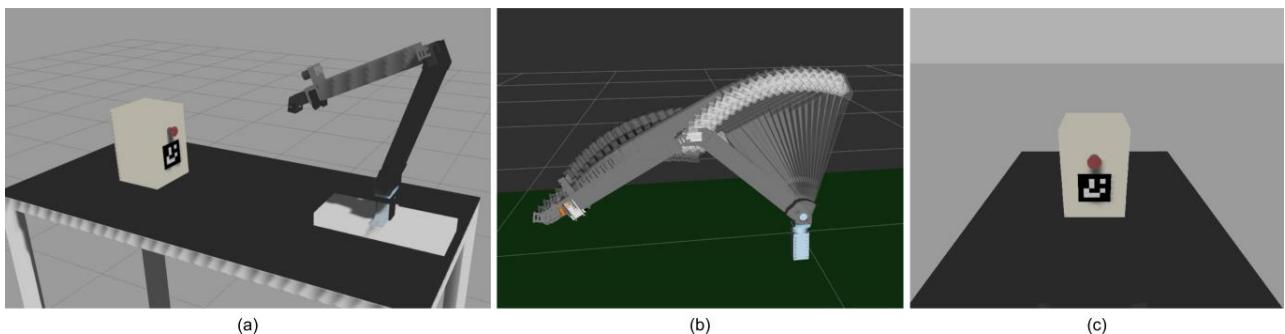


Figura 1: (a) Manipulador realizando tarefa no *Gazebo*, (b) Planejamento de trajetória com *MoveIt*, (c) Visão a partir da câmera integrada (*Rviz*).

4. CONSIDERAÇÕES FINAIS

Os métodos empregados neste projeto apresentaram resultados esperados em ambientes gerados pelo *Gazebo*, *MoveIt* e *Rviz*, sendo o manipulador capaz de acionar o painel elétrico a partir da detecção de um marcador visual, pendente a realização dos testes no mundo real. Necessitam-se ainda verificações do manipulador robótico em diferentes tarefas, como realizar movimentos mais complexos e procurar por outros pontos que estejam dentro ou fora do seu espaço de trabalho. Adiante, propõe-se finalizar o desenvolvimento e integração do sistema de visão que será aplicado, assim como a montagem final utilizando partes reais. A eficácia

do efetuador final será avaliada conforme resultados encontrados e ajustes serão realizados caso este não atenda as especificações desejadas.

5. REFERÊNCIAS

- ¹ CRAIG, John J. Robótica. 3^a edição. **Rev. Atual**, 2012.
- ² HERNÁNDEZ-ORDOÑEZ, Martín et al. An education application for teaching robot arm manipulator concepts using augmented reality. **Mobile Information Systems**, v. 2018, 2018.
- ³ HERNANDEZ-MENDEZ, Sergio et al. Design and implementation of a robotic arm using ROS and MoveIt!. In: **2017 IEEE International Autumn Meeting on Power, Electronics and Computing (ROPEC)**. IEEE, 2017. p. 1-6.
- ⁴ SIMPLÍCIO, Paulo Victor Galvão; LIMA, Beatriz Rêgo. Manipuladores robóticos industriais. **Caderno de Graduação-Ciências Exatas e Tecnológicas-UNIT**, v. 3, n. 3, p. 85, 2016.
- ⁵ ROS. Ros, 2010. Disponível em: <https://www.ros.org/>. Acesso em: 03 de abril de 2020.
- ⁶ ONSHAPE. Capterra, 2019. Disponível em: <https://www.capterra.com.br/software/164681/onshape>. Acesso em: 02 de abril de 2020.
- ⁷ FREITAS, Paulo J. de. Introdução a modelagem e Simulação de Sistemas. **Florianópolis, SC, Brasil: Visual Books**, p. 2-14, 2001.
- ⁸ GAZEBO. Gazebo, 2002. Disponível em: <http://gazebosim.org/>. Acesso em: 03 de abril de 2020.
- ⁹ RVIZ. Rviz, 2010. Disponível em: <http://wiki.ros.org/rviz>. Acesso em: 03 de abril de 2020.
- ¹⁰ MOVEIT. MoveIt, 2016. Disponível em: <https://moveit.ros.org/>. Acesso em: 03 de abril de 2020.

Certificado de Participação do Evento SAPCT 2020

Acesse <https://doity.com.br/validar-certificado> para verificar se este certificado é válido. Código de validação: ZPBUZMZ



**INTELIGÊNCIA
ARTIFICIAL**

CERTIFICADO

Certificamos que **JÉSSICA LIMA MOTTA** participou do V Seminário de Avaliação de Pesquisa Científica e Tecnológica (SAPCT) e IV Workshop de Integração e Capacitação em Processamento de Alto Desempenho (ICPAD), no dia 5 de junho de 2020, com carga horária de 6 hora(s).

Jailson Bittencourt de Andrade

Pró-reitor de pós-graduação e pesquisa do Centro Universitário SENAI CIMATEC



**Artigo publicado TRIS: Thermal Remote
Identification System of Feverish People- Evento
SIINTEC 2020**

TRIS: THERMAL REMOTE IDENTIFICATION SYSTEM OF FEVERISH PEOPLE

Diogo A. Martins, Pedro P. V. Tecchio, João V. Torres Borges, Nelson A. Ferreira Neto, Ana C. B. de Jesus, Jéssica L. Motta, Aziel M. de Freitas Júnior, Mateus S. de Cerqueira, Tiago P. de Souza

Robótica e Sistemas Autônomos, SENAI CIMATEC, Brazil

Abstract: The COVID-19 pandemic has pressed for many technological fronts in the combat against the corona virus spreading. Among those fronts, the TRIS project raises up as a virus propagation control tool by assisting in the process of detecting feverish people, enabling the conduct of a medical thermal screening procedure. Among the functionalities proposed by TRIS, the essential modules are face detection and recognition, as well as, temperature estimation. In this paper, we point out the challenges and problems faced during its development and provide a comparison with other thermal systems used for face temperature estimation.

Keywords: COVID-19; artificial intelligence; face detection; temperature estimation, thermal screening

TRIS: SISTEMA TÉRMICO REMOTO DE IDENTIFICAÇÃO DE PESSOAS FEBRIS

Resumo: A pandemia de COVID-19 instigou muitas frentes tecnológicas no combate à propagação do corona vírus. Dentre essas frentes, o projeto TRIS surge como uma ferramenta de controle de propagação do vírus ao auxiliar no processo de detecção de pessoas febris, possibilitando a realização do procedimento médico de triagem térmica. Dentre as funcionalidades propostas pelo TRIS, os módulos essenciais são detecção e reconhecimento de faces, além de estimativa de temperatura. Neste artigo, apontamos os desafios e problemas enfrentados durante seu desenvolvimento e fazemos uma comparação com outros sistemas térmicos usados para estimativa de temperatura facial.

Palavras-chave: COVID-19; inteligência artificial; detecção de faces; estimativa de temperatura, triagem por temperatura

1. INTRODUCTION

Infectious diseases are a constant challenge for health and science. The emblematic case in the 21st century is the new coronavirus that causes the COVID-19 disease, which despite the fact that science has already advanced and circumvented situations such as Ebola and the H1N1 influenza virus, faces endless difficulties about the capability of the new virus mutation. 18,902,735 cases of COVID-19 and 709,511 deaths have been confirmed by World Health Organization, until August 7, 2020 [1] and this numbers tend to increase if there are no efforts to prevent coronavirus proliferation until a vaccine release.

In order to implement new strategies for the coronavirus prevention, a system named TRIS - Thermal Remote Identification System was developed. TRIS is an artificial intelligence based tool for remote temperature measurement that uses facial recognition on surveillance cameras and thermal cameras over crowded or intense pedestrian traffic places with the purpose of reducing infections caused by coronavirus.

This paper will describe and explain the challenges of developing technology to prevent coronavirus at times of pandemic. First of all, it will be presented the methodology, techniques and principles used as face detection, facial recognition, how the temperature is estimated, how the system interface works and data capture. Then the third section reports the results and analysis achieved with images and graphics.

2. METHODOLOGY

2.1. Face Detection

The face detection module determines the location, size and estimated pose of a face in a digital image. Face detection is carried out with RetinaFace, a robust single-stage face detector, which performs pixel-wise face localization on various scales of faces by taking advantages of joint extra-supervised and self-supervised multi-task learning [2]. RetinaFace outperforms the state-of-the-art average precision on WIDER FACE benchmark [3] and provides implementation examples in its repository.

This module was implemented in TensorFlow 2.0+ with ResNet50 as the convolutional backbone architecture used for feature extraction. The neural network model was trained and validated with WIDER FACE dataset, achieving an average precision of 83.55% on the hard test set. Within the TRIS system, the face detection module receives images from cameras as input, executes the inference and returns the set of bounding boxes corresponding to the faces found.

The Face Detection and subsequent recognition are done in images acquired by a Logitech C270 camera, Figure 1a, and by a FLIR ADK, Figure 1b. The Logitech C270 has a resolution of 1280 x 720, 30 FPS (Frames Per Second) and a FOV (Field of View) of 60°. The FLIR ADK has a resolution of 640 x 512, being able to operate with 30 or 60 FPS and a FOV of 50°.

Figure 1: TRIS cameras



Figure 1a: Logitech c270



Figure 1b: FLIR ADK

2.2. Facial Recognition

The system is also equipped with facial recognition capabilities, enabling the tracking of individuals of interest (the feverish ones). This feature is based on the *face_recognition* framework [4]. It works by processing the RGB captured frames as an input, then it proceeds to assign *face signatures* to each detected face. The framework uses a deep learning trained model, that ran through approximately 3 million faces of more than 7000 different individuals, in order to give the algorithm an ability to recognize where are the most distinctive features in a face, which then enables recognition [5].

As a next step of this project, all successfully detected faces on each camera captured frame will have their signatures compared against database stored face signatures and the registered individuals whose temperature are beyond the safe threshold will be contacted and informed of the fact.

2.3. Temperature Estimation

In order to estimate human face temperature from infrared radiometric images, one needs to first convert these images which contains values proportional to the amount of infrared radiation generated by scene objects to temperature values. Some cameras already provide thermal images, which are the ones with temperature values for each pixel, others do not. In this work a camera that did not have this feature FLIR ADK was used, Figure 1b. It has a resolution of 640 x 512, 30 or 60 FPS and a FOV of 50°. Its thermal sensitivity is 50 mK or less.

Because of that, a simple experiment of heating a body of water was developed to obtain the camera calibration curve that relates radiance to temperature values. The water temperature was measured using a Minipa ET-2082B with the provided thermocouple. This equipment has a 1 °C resolution over the range of measured temperatures, which was around 10 to 100 °C. Later on, this calibration curve was used to estimate the temperature values throughout this work.

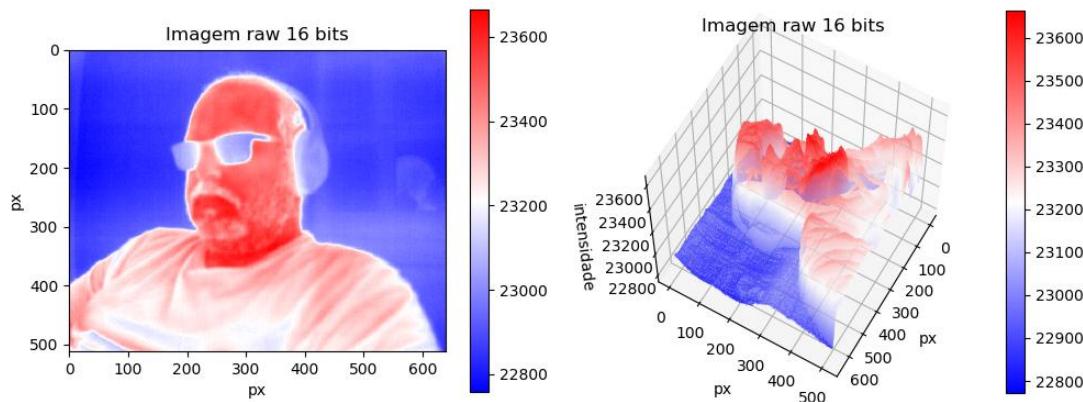
After temperature conversion, one needs to estimate the face temperature from the bounding boxes of each image given by the face detection and recognition algorithm. We propose a straightforward statistical approach of only using data from the 75th to 95th percentiles which are then averaged to obtain an estimate. This estimate is later compared with a given temperature threshold to decide if the person has a fever or not. Another approach, would be to utilize a calibrated "black body" as a thermal reference on the scene captured by the thermal camera.

2.4. Dataset Capture

While still under study and implementation, the dataset capture feature will allow one to record a set of RGB and infrared radiance video files or still frames. It is already possible to store video files from both cameras using image manipulation and video compression techniques.

While most color cameras and video files utilize three layers of unsigned 8 bits values per pixel, raw radiance images are received from the infrared thermal camera as a single layer of unsigned 16 bits values per pixel, as seen on Figure 2. Therefore, in order to utilize standard video compression encoders to store uncompressed radiance images, the single layer of 16 bits values per pixel were segregated in two layers of 8 bits values per pixel by utilization of bit-shifting. These two layers were then coupled with another zeroed layer to form a standard RGB image, which was then compressed as usual.

Figure 2. Thermal image example in raw 16 bits



3. RESULTS AND DISCUSSION

3.1. Operation of the Graphical User Interface - GUI

The TRIS GUI allows a verified user control of the system operation. User verification is done via login and password credentials check. By command of the user, the system also shows, in real time, both RGB and infrared camera streams with or without bounding boxes marking all detected faces and their estimated temperatures. In Figure 3 is shown a picture of the graphical user interface.

Figure 3: TRIS interface.



The system may show in the bottom half of the interface the faces of people classified with a feverish state. The user is able to choose the display order by decreasing temperature values or by increasing detection time. If a face can be matched to a person in the system database, his or her identity will be shown along with his or her face and temperature. If no match occurs, only the detected face and temperature is shown. All the above information will be kept by the system so that it can be consulted at any time.

3.2. Inference Optimization

In order to meet real-time operating requirements, the optimization of the neural network graph was performed through the TensorFlow integration with NVIDIA TensorRT (TF-TRT). Two optimized models were generated, one with the neural network weights represented in 32-bit floating point values (FP32) and another with half-precision float point values (FP16). The performance of the system with TensorFlow and TF-TRT optimized models can be seen in Tables 1 and 2.

Table 1. System performance values: Development environment Ubuntu 18.04, AMD Ryzen 7 1800x, 16GB DDR4, NVIDIA RTX 2070 Super.

	TensorFlow (FPS)	TF-TRT - FP32 (FPS)	TF-TRT - FP16 (FPS)
Minimum	2.57	30.96	31.09
Mean	3.15	34.38	34.56
Standard deviation	0.17	1.77	1.76

Maximum	3.67	39.06	39.72
----------------	------	-------	-------

Table 2. System performance values: Test environment Ubuntu 18.04, Intel Core i7-9750H, 8GB DDR4, NVIDIA GTX 1650.

	TensorFlow (FPS)	TF-TRT - FP32 (FPS)	TF-TRT - FP16 (FPS)
Minimum	3.39	16.33	20.45
Mean	4.17	23.45	23.72
Standard deviation	0.12	1.56	1.16
Maximum	4.44	26.24	27.09

3.3. Thermal Camera Curve Calibration Experiment

Both heating water experiments results are displayed in Figures 4 and 5. As can be seen in these figures, a simple linear function was enough to provide a good enough fitting model with R-squared values greater than 0.99 in both cases. But, although both experiments can be fitted to a linear model, the resulting linear functions are distinct in both slope and intercept values. It is relevant to point that both experiments were done using the same materials, but in different days.

Figure 4. First thermal camera curve calibration experiment

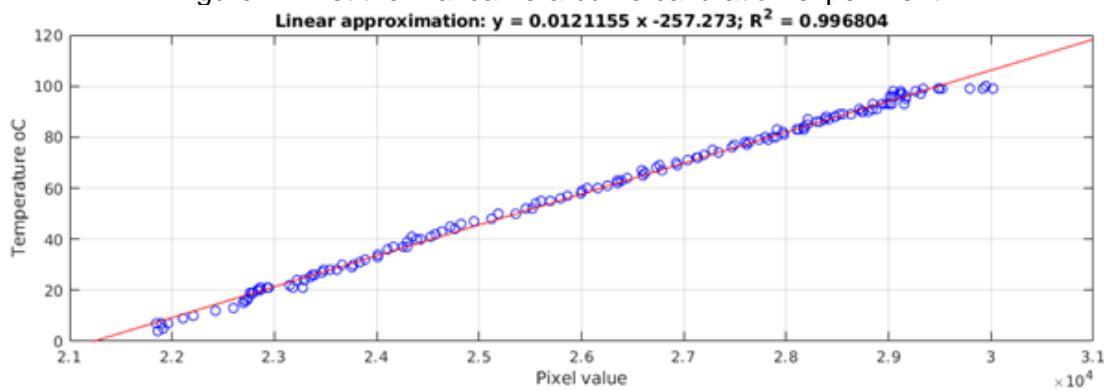
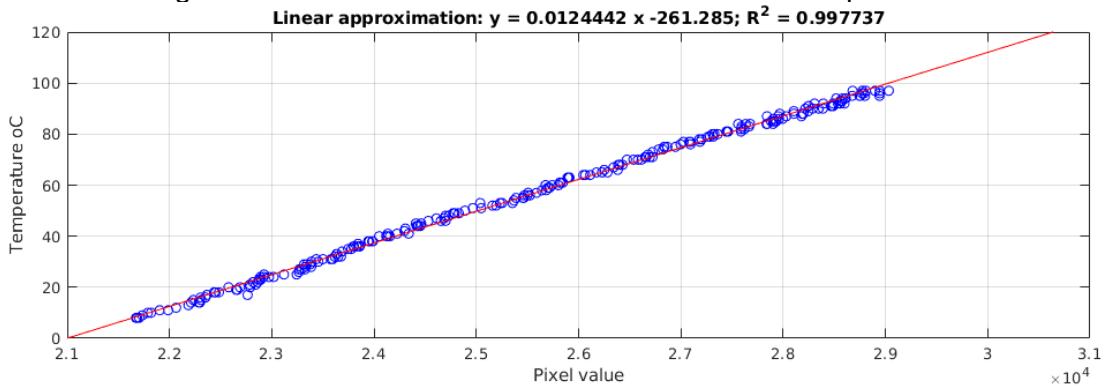
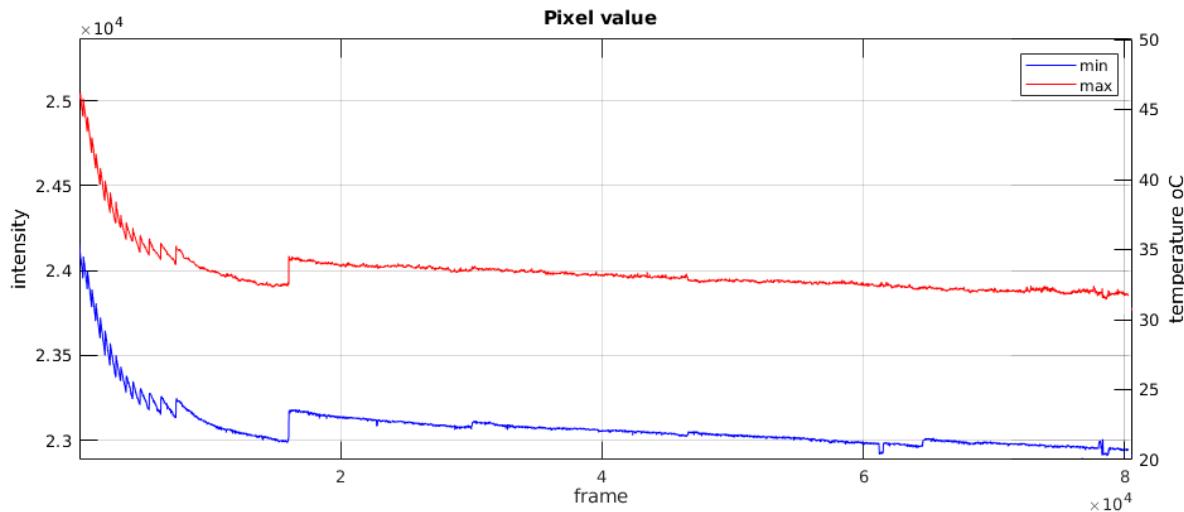


Figure 5. Second thermal camera curve calibration experiment



These results seem to indicate that the calibration curve may be approached as a linear function by themselves, but directly applying these linear models during further development of this project did no provide consistent results. Temperature differences from ground truth to estimated values of up to 5 °C were found during adverse ambient temperature and sunlight illumination. Thus, TRIS is a solution to be tested under controlled environmental conditions, with suitable distances between cameras and pedestrian and far from sunlight exposition. These results seem to indicate that a more robust non-linear model which takes in consideration other factors besides object instantaneous temperature and radiance values is required for greater accuracy and reliability.

Figure 6. Infrared camera behavior of static scene over time (frames)



Moreover, as can be seen from Figure 6, the existing infrared camera has pixel intensity drift over different image frames taken from the same static scene. This behavior causes further issues in the temperature estimation process, it is not easily corrected and usually requires an improved hardware configuration.

4. CONCLUSION

TRIS utilizes state of the art techniques like RetinaFace neural network architecture for face detection and inference optimization with TensorRT. These combined with our temperature calibration experiments allows TRIS to provide its user with an estimated temperature for every detected person passing through its line of sight in real time.

While the current solution does not conform with the desired accuracy (about ± 0.3 °C), it is important to notice that commercially available state of the art thermographic cameras have accuracy around ± 2 °C [6,7]. Moreover, this accuracy happens while taking account of more robust physical models and improved calibration methods. Existing commercial solutions for human fever screening make use of the same calibration methods in a much more restrained temperature range centered around normal human body temperature [8,9]. These systems can also use external temperature reference sources known as blackbodies in order to further increase their accuracy [9]. The results shown in this work do not use these methods yet.

More accurate temperature estimations will be achieved during the current system development using a more reliable calibration procedure, by taking account of

emissivity, reflectivity and transmissivity constraints, as well as, environment variations and by making use of external temperature references and a more appropriate thermal camera. TRIS is expected to be employed as sensory tool to allow for control the spread of diseases which cause feverish states, specially the coronavirus. It is currently being tested at SENAI CIMATEC in one of its laboratories in order to verify its performance and reliability. These tests will provide data for further development.

Acknowledgments

The authors would like to thank SENAI CIMATEC for its support on materials and funding for this research.

5. REFERENCES

- ¹ BASTOS, L., 2020. *OPAS/OMS Brasil - Folha Informativa – COVID-19 (Doença Causada Pelo Novo Coronavírus)* | OPAS/OMS. [online] Pan American Health Organization / World Health Organization. Available at: <https://www.paho.org/bra/index.php?option=com_content&view=article&id=6101:covid19&Itemid=875>. Accessed on: 10 Aug. 2020.
- ² DENG, Jiankang et al. RetinaFace: Single-Shot Multi-Level Face Localization in the Wild. In: **Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition**. 2020. p. 5203-5212.
- ³ YANG, S. et al. *Wider face: A face detection benchmark*. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.
- ⁴ GEITGEY, Adam. *Face recognition*. Available at: <https://github.com/ageitgey/face_recognition>. Accessed on: 10 Aug. 2020.
- ⁵ KING, Davis. *Dlib Models*. Available at: <<https://github.com/davisking/dlib-models>>. Accessed on: 10 Aug. 2020.
- ⁶ FLIR® Systems, Inc. *FLIR A400/A700 Thermal Cameras with Smart Sensor Configuration*. Available at: <<https://www.flir.com/products/a400-a700-smart-sensor/?model=85900-0000-T300293>> Accessed: 13 Sep. 2020.
- ⁷ Fluke Corporation. *Fluke RSE600 mounted infrared camera*. Available at: <<https://www.fluke.com/en/product/thermal-cameras/rse600>> Accessed: 13 Sep. 2020.
- ⁸ FLIR® Systems, Inc. *FLIR A700 EST Thermal Screening Solutions*. Available at: <<https://www.flir.com/products/flir-a700-est/?model=85902-0101>> Accessed: 13 Sep. 2020.
- ⁹ Seek Thermal. *Seek Scan™*. Available at: <<https://www.thermal.com/seekscan.html>> Accessed: 13 Sep. 2020.

Certificado de Participação do Evento SIINTEC 2020

Acesse <https://doity.com.br/validar-certificado> para verificar se este certificado é válido. Código de validação: 9ZUMUM-A



International Symposium
on Innovation and Technology

Challenges in science, technology and innovation after COVID-19

Sistema FIEB



CERTIFICADO

Certificamos que o trabalho TRIS: THERMAL REMOTE IDENTIFICATION SYSTEM OF FEVERISH PEOPLE dos autores DIOGO ALEXANDRE MARTINS, PEDRO PAULO VENTURA TECCCHIO, JOÃO VICTOR TORRES BORGES, NELSON ALVES FERREIRA NETO, ANA CAROLINA BARRETO DE JESUS, JÉSSICA LIMA MOTTA, AZIEL MARTINS DE FREITAS JÚNIOR, MATEUS SANTOS DE CERQUEIRA, TIAGO PEREIRA DE SOUZA foi aceito para publicação no **VI International Symposium on Innovation and Technology (SIINTEC)**, realizado no período de 21/10/2020 a 23/10/2020.

Profa. Dra. Lilian Lefol Nani Guarieiro
Coordenadora SIINTEC 2020