



**UGV (UNMANED GROUND VEHICLE) PARA MONITORAMENTO DE TALHÕES  
EM FRUTICULTURA E SILVICULTURA**

**Breno Alencar Araújo ([brenoaa@al.insper.edu.br](mailto:brenoaa@al.insper.edu.br))**

**Bruno Morales Balkins ([brunomb4@al.insper.edu.br](mailto:brunomb4@al.insper.edu.br))**

**Fernando Bichuette Assumpção ([fernandoba2@al.insper.edu.br](mailto:fernandoba2@al.insper.edu.br))**

**Giulia Carolina Martins de Sampaio ([giuliacms@al.insper.edu.br](mailto:giuliacms@al.insper.edu.br))**

**Trabalho de Conclusão de Curso**

**Relatório do Projeto Final de Engenharia**

**Versão Final**

**São Paulo - SP – Brasil**

**Fevereiro 2024**

**Breno Alencar Araújo**  
**Bruno Morales Balkins**  
**Fernando Bichuette Assumpção**  
**Giulia Carolina Martins de Sampaio**

**UGV (UNMANED GROUND VEHICLE) PARA MONITORAMENTO DE TALHÕES  
EM FRUTICULTURA E SILVICULTURA**

**Trabalho de Conclusão de Curso**  
**Relatório do Projeto Final de Engenharia**  
**Versão Final**

Relatório apresentado aos respectivos cursos de Engenharia (Computação/Mecânica/Mecatrônica) ou Ciência da Computação, como requisito parcial para a obtenção do título de Bacharel no respectivo curso.

Professor Orientador: Prof. Dr. Vinícius Licks

Mentor: Thiago Teixeira Santos

Coordenador TCC/PFE: Prof. Dr. Luciano Pereira Soares

**São Paulo - SP – Brasil**

Fevereiro 2024

**Breno Alencar Araújo**  
**Bruno Morales Balkins**  
**Fernando Bichuette Assumpção**  
**Giulia Carolina Martins de Sampaio**

**UGV (UNMANED GROUND VEHICLE) FOR MONITORING ORCHARDS IN  
FRUIT AND SILVICULTURE**

**Capstone Project Report**  
**Final Version**

Undergraduate Capstone Project Report submitted to the respective (Computer/Mechanical/Mechatronic) Engineering programs or Computer Science program in partial fulfillment of the requirements for the Bachelor's Degree.

Advisor: Prof. Dr Vinícius Licks

Mentor: Thiago Teixeira Santos

General Coordinator TCC/PFE: Prof. Dr. Luciano Pereira Soares

**São Paulo - SP – Brazil**

Fevereiro 2024

**Breno Alencar Araújo**

**Bruno Morales Balkins**

**Fernando Bichuette Assumpção**

**Giulia Carolina Martins de Sampaio**

**UGV (UNMANED GROUND VEHICLE) PARA MONITORAMENTO DE TALHÕES  
EM FRUTICULTURA E SILVICULTURA**

Trabalho de Conclusão de Curso apresentado aos programas de Graduação em Engenharia (Computação/Mecânica/Mecatrônica) e Ciência da Computação como requisito parcial para a obtenção do título de Bacharel no respectivo curso.

**Orientador:** Prof. Dr. Vinícius Licks

**Banca Examinadora**

---

Prof. Vinícius Licks

Insper

---

Prof. Fábio Ferraz Junior

Insper

---

Prof. Silvio Szafir

Insper

## Sumário

<b>RESUMO .....</b>	<b>10</b>
<b>ABSTRACT .....</b>	<b>11</b>
<b>INTRODUÇÃO .....</b>	<b>7</b>
ESCOPO DO PROJETO.....	9
RECURSOS .....	9
PLANEJAMENTO.....	9
<i>Work Breakdown Structure (WBS)</i> .....	10
<i>Cronograma</i> .....	11
MAPEAMENTO DOS STAKEHOLDERS .....	12
RISCOS ENVOLVIDOS.....	13
QUESTÕES ÉTICAS E PROFISSIONAIS.....	13
NORMAS TÉCNICAS .....	14
REVISÃO DO ESTADO DA ARTE .....	17
<i>Missão Mars Pathfinder e o Rover Sojourner</i> .....	17
<i>Evolução da tecnologia de mapeamento</i> .....	20
<i>Robot Operating System (ROS) na robótica</i> .....	22
<i>Mecanização agrícola e Revolução Verde</i> .....	23
<i>Integração de Unmanned Ground Vehicles (UGVs) na Agricultura</i> .....	24
<i>Impacto das inovações tecnológicas na produtividade agrícola</i> .....	27
<b>METODOLOGIA .....</b>	<b>28</b>
COLETA DE DADOS SOBRE O PROJETO.....	30
ANÁLISE DOS DADOS COLETADOS .....	33
<b>DETALHAMENTO TÉCNICO DO PROTÓTIPO .....</b>	<b>36</b>

<b>ELETRÔNICA.....</b>	<b>36</b>
<i>Esquemáticos e Detalhamento Geral de Funcionamento.....</i>	<i>36</i>
<i>Especificação dos Motores.....</i>	<i>40</i>
<i>Detalhamento Circuito de Emergência.....</i>	<i>42</i>
<i>Detalhamento Circuito Display.....</i>	<i>44</i>
<i>Etapa de Cabeamento .....</i>	<i>46</i>
<b>MECÂNICA .....</b>	<b>48</b>
<i>Detalhamento Estrutural do Chassi e Suspensão.....</i>	<i>48</i>
<i>Adaptações Necessárias para Montagem.....</i>	<i>49</i>
<i>Revisão de Erros Encontrados na Documentação do Oficial do GitHub.....</i>	<i>53</i>
<b>PROGRAMAÇÃO.....</b>	<b>54</b>
<i>Configuração da Raspberry Pi como Servidor e Comunicação SSH.....</i>	<i>54</i>
<i>Aquisição de Imagens de Monitoramento .....</i>	<i>75</i>
<b>RESULTADOS.....</b>	<b>78</b>
<b>CONCLUSÕES E TRABALHOS FUTUROS .....</b>	<b>88</b>
<b>REFERÊNCIAS .....</b>	<b>91</b>
<b>APÊNDICE A .....</b>	<b>96</b>
<b>APÊNDICE B .....</b>	<b>102</b>
<b>APÊNDICE C .....</b>	<b>103</b>
<b>APÊNDICE D .....</b>	<b>106</b>

## **Índice de Figuras**

Figura 1 – Espaçamento entre os talhões, usado para locomoção de pessoas e equipamentos..	7
Figura 2 – Estrutura WBS .....	10
Figura 3 – Rover Sojourner .....	17
Figura 4 – Mars Rover.....	19
Figura 5 – SLAM feito com câmeras RGB-D .....	21
Figura 6 - Trator a querosene Waterloo Boy .....	23
Figura 7 – UGV Burro Grande. ....	26
Figura 8 – Produtividade agrícola nos Estados Unidos.....	28
Figura 9 – Processo completo de desenvolvimento de produto. As etapas representadas em cinza não foram contempladas neste projeto.....	29
Figura 10 – Esquemático 3D do UGV. ....	30
Figura 11 - LiDAR Ouster 16 canais.....	32
Figura 12 - Esboço da solução.....	35
Figura 13 – Placas Eletrônicas.....	36
Figura 14 - Diagrama elétrico de alimentação.....	37
Figura 15 - Diagrama elétrico de sinais.....	38
Figura 16 - Placas após solda de componentes.....	40
Figura 17 – Motor Alocado no Robô .....	41
Figura 18 -- Servo Alocado no Robô.....	42
Figura 19 – Circuito de Emergência.....	43
Figura 20 – Sinal de Emergência. ....	44

Figura 21 – Ligações dos pinos SDA e SCL do display na placa de controle .....	45
Figura 22 – Exemplo de ligação entre conectores .....	47
Figura 23 - Estrutura do robô após montagem. ....	48
Figura 24 – Perfil de alumínio usado para montagem da suspensão.....	50
Figura 25 – Operação de rosqueamento. ....	50
Figura 26 – barras de alumínio para montagem do corpo. ....	51
Figura 27 – Marcação nas barras de alumínio para etapa de furação.....	52
Figura 28 – Representação da peça substituída e de sua substituta.....	53
Figura 29 – Peça identificada incorretamente. ....	54
Figura 30 – Funcionamento modo desktop. ....	55
Figura 31 – Funcionamento modo servidor.....	55
Figura 32 – Diagrama de Funcionamento Protocolo SSH .....	56
Figura 33 – Funcionamento do ROS. ....	58
Figura 34– Interface do Basicmicro Studio. ....	64
Figura 35 – Fluxograma de funcionamento da rotina de emergência. ....	71
Figura 36 – Fluxograma de funcionamento do UGV.....	73
Figura 37 – Imagens Visualizadas no Rviz. ....	76
Figura 38 – Imagens Visualizadas no rqt_image_view.....	76
Figura 39 – Fluxogramas de obtenção de imagens.....	77
<b>Figura 40 – Ganho excessivo de cambagem .....</b>	<b>79</b>
Figura 41 – Ângulos de Ackerman.....	80
Figura 42 – Ganho de cambagem ajustado.....	80

<b>Figura 43</b> – Exemplo de fixação da roda ao cubo.	81
Figura 44 – Fluxograma de funcionamento do deadman.	82
Figura 45 – Mapa dos botões do controle frente	82
Figura 46 – Mapa dos botões do controle trás	83
Figura 47 – Display conectado ao microcomputador exibindo “None%”	84
<b>Figura 48</b> – Modelagem 3D da base do suporte da câmera no Fusion360	85
<b>Figura 49</b> – Modelagem 3D do acoplamento para suporte da câmera	86
Figura 50 – Modelagem 3D do suporte do LIDAR no Fusion360	87
<b>Figura 51</b> – Suporte para câmera e LiDAR	88
Figura 52 – Robô completo.	89

## **Índice de Tabelas**

Tabela 1 - Cronograma de atividades .....	12
Tabela 2 - Matriz de Stakeholders.....	13
Tabela 3 - Normas técnicas ABNT e ASTM.....	16
Tabela 4 - Matriz de variantes de solução. ....	33
Tabela 5 – Comandos para inicialização. ....	75

## **RESUMO**

Este projeto tem como objetivo desenvolver um UGV, veículo terrestre não tripulado, para monitoramento de talhões em citricultura, pomicultura e silvicultura, inspirado no JPL Open Source Rover Project da NASA. O veículo foi rádio controlado e possui uma unidade computacional executando o framework ROS para monitoramento, integração e futuro controle autônomo. A utilização do framework trará a flexibilidade de se modificar os nós conforme necessidades específicas do cliente. Para isso, fez-se uso da metodologia de PDP (Processo de Desenvolvimento de Produtos) para abranger os requerimentos junto ao cliente. Avaliações técnicas preliminares mostraram ser efetivo o uso do veículo para monitoramento e movimentação pelos talhões.

**Palavras-chave:** UGV; ROS; Open Source Rover Project; Monitoramento de talhões em citricultura;

## **ABSTRACT**

This project aims to develop an Unmanned Ground Vehicle (UGV) for monitoring orchards in citrus, apple, and forestry cultivation, inspired by NASA's JPL Open Source Rover Project. The vehicle is radio-controlled and equipped with a computational unit running the ROS framework for monitoring, integration, and future autonomous control. The use of the framework will provide flexibility to modify nodes according to specific customer needs. To achieve this, the Product Development Process (PDP) methodology was employed to address requirements with the customer. Preliminary technical evaluations have shown the vehicle's effectiveness for monitoring and maneuvering through orchards.

**Key-words:** UGV; ROS; Open Source Rover Project; Monitoring of citrus plantations.

## Introdução

A Empresa Brasileira de Pesquisa Agropecuária (Embrapa) é uma empresa federal fundada em 1973 e vinculada ao Ministério da Agricultura e Pecuária (MAPA), voltada para o desenvolvimento tecnológico e de pesquisa aplicados a agropecuária brasileira, de forma compatível com as nuances do clima tropical e com as particularidades socioeconômicas das várias regiões de um país continental.

Na agricultura, seja ela fruticultura ou silvicultura, um elemento fundamental é o talhão. O talhão é a unidade mínima de cultivo de uma propriedade, geralmente delimitado por características naturais como relevo, tipo de solo, ou por características artificiais como estradas e cercas. (Duft, 2014)

Na fruticultura, os talhões são frequentemente usados para organizar a plantação de árvores frutíferas. Cada talhão pode conter uma única variedade de fruta, permitindo um manejo mais eficiente das árvores e uma colheita mais organizada. Além disso, os talhões permitem um controle mais eficaz de pragas e doenças, pois é mais fácil isolar um talhão afetado para tratamento. (Gebler, 2017)

**Figura 1** – Espaçamento entre os talhões, usado para locomoção de pessoas e equipamentos.



**Fonte:** Embrapa Agricultura Digital (2024).

Na silvicultura, os talhões são ainda mais importantes. Eles são usados para organizar a floresta em unidades de manejo, cada uma com seu próprio plano de manejo florestal. Isso permite que diferentes técnicas silviculturais sejam aplicadas em diferentes partes da floresta, dependendo das características específicas de cada talhão. Além disso, os talhões facilitam a realização de inventários florestais e o planejamento de operações de colheita. (Maran et al, 2020)

Nos talhões, o monitoramento durante o processo produtivo é essencial, possibilitando o planejamento da produção e da colheita, bem como o controle de pragas, garantindo maior qualidade do produto oferecido ao consumidor final e diminuindo as perdas por parte dos produtores.

O processo de monitoramento humano pode ser dificultado ou até mesmo inviabilizado dependendo de fatores como o tamanho e a quantidade de talhões. Segundo dados da Confederação da Agricultura e Pecuária do Brasil (CNA), em 2021 a produção nacional de frutas ultrapassou 41 milhões de toneladas, sendo que 81% dos estabelecimentos produtores se enquadram como agricultura familiar. De acordo com o IBGE (Instituto Brasileiro de Geografia e Estatística), o Brasil, que é líder no fornecimento de madeira para produção de papel e celulose, bateu recorde em 2022, com 99,7 milhões de metros cúbicos (IBGE, 2023). Os estados do Sudeste lideram ambas as produções, no entanto, apenas 16% dos estabelecimentos agropecuários são da agricultura familiar do Brasil (CNA Brasil, 2022). Dados que revelam a força do setor primário na economia brasileira e que grandes latifúndios, concentrados nessa região, vem demandando novas tecnologias para melhorar a produtividade.

Considerando tal demanda, o desenvolvimento de tecnologias no setor aplicadas ao monitoramento das plantações pode garantir que o procedimento seja realizado de maneira mais assertiva e rápida.

Este projeto parte desse problema e tem como objetivo oferecer uma alternativa para o departamento de Agricultura Digital da Embrapa, voltada ao monitoramento de talhões de fruticultura e silvicultura por meio da aplicação da robótica no ambiente produtivo.

## Escopo do projeto

Após a finalização do projeto será entregue o protótipo de um UGV (Unmanned Ground Vehicle), rádio controlado e voltado ao monitoramento dos talhões de fruticultura e silvicultura, capaz de abrigar os componentes requisitados pela empresa, sendo eles:

- um LiDAR de modelo Ouster os1 de 16 canais;
- 2 a 4 câmeras modelo Intel RealSense Depth D435i ou Logitech C920 HD PRO.

O projeto confeccionado será baseado no PJL Open Source Rover, um projeto aberto disponibilizado pela Nasa no repositório do GitHub (GitHub, 2024), sujeito a algumas alterações para comportar os componentes requisitados previamente citados.

O protótipo irá conter uma unidade computacional com sistema operacional baseado em Linux para controle do robô, executando a framework ROS2 (Robot Operating System) Humble e sendo capaz de se movimentar por terrenos irregulares e com aclividades, aplicável para utilização em talhões de tamanho comercial com tempo de operação de 30 minutos.

## Recursos

Os recursos necessários para o desenvolvimento do projeto serão materiais, financeiros e humanos. Os materiais serão as infraestruturas disponibilizadas pelos laboratórios da instituição e aqueles comprados com os recursos financeiros. Os recursos financeiros serão necessários para aquisição das peças. Os recursos humanos serão o auxílio dos técnicos dos laboratórios e o tempo e dedicação dos alunos e do professor orientador. Em anexo, planilha com os recursos materiais utilizados.

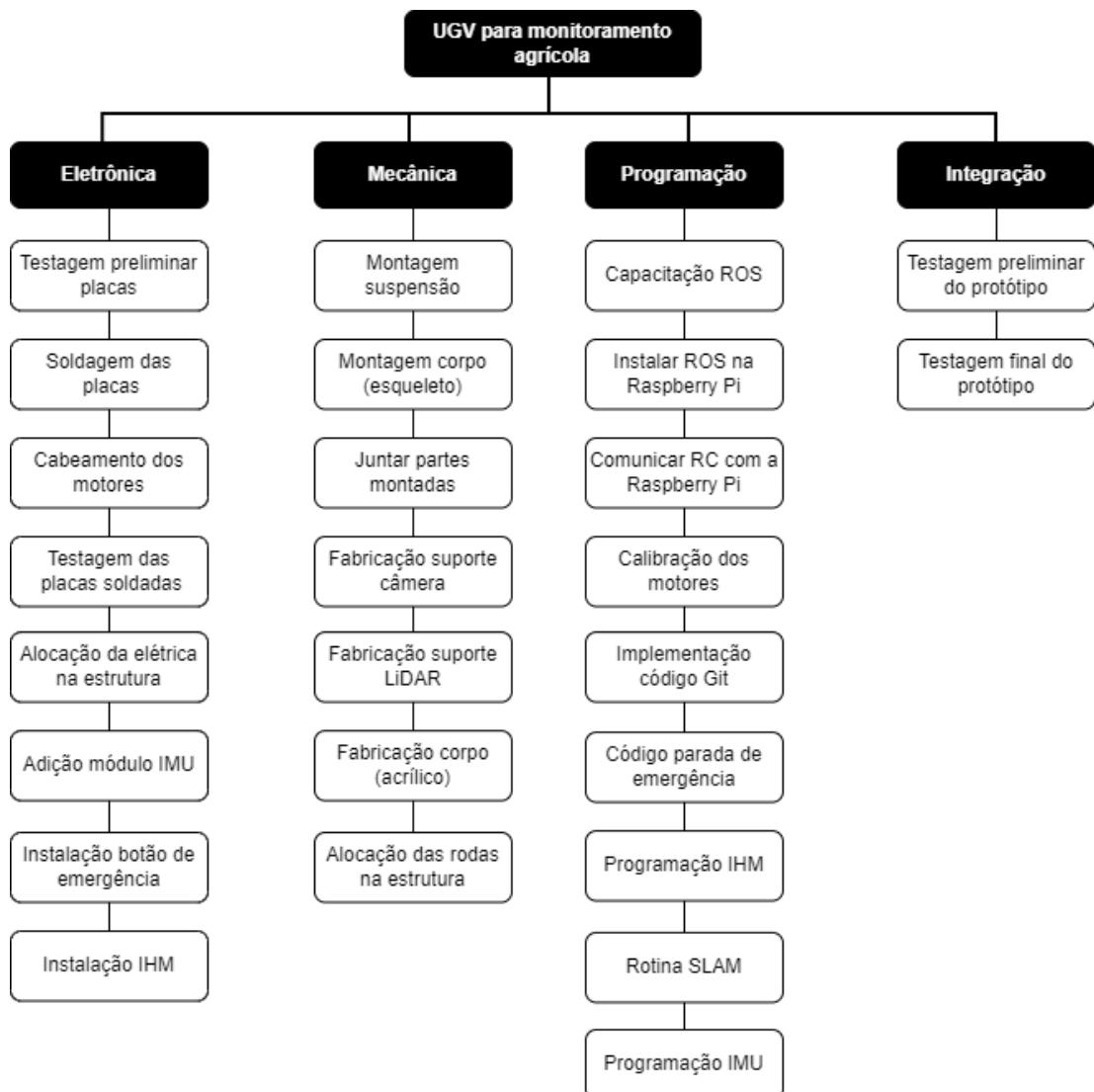
## Planejamento

O planejamento do projeto foi estruturado a partir de duas ferramentas: o *Work Breakdown Structure* (WBS), para divisão dos entregáveis em tarefas menores e o cronograma, usado para encaixar cada uma das tarefas dentro do prazo de entrega do projeto.

## Work Breakdown Structure (WBS)

Para elaboração da análise WBS (Figura 1), foi necessário quebrar o projeto em entregáveis e dividir esses entregáveis em pacotes menores de trabalho, respeitando as etapas definidas na metodologia. Para facilitar esse processo, os entregáveis foram considerados como sendo as próprias áreas de desenvolvimento do projeto (mecânica, eletrônica, programação e integração) e os pacotes de trabalho, cada uma das atividades necessárias para conclusão da área em questão.

**Figura 2 – Estrutura WBS.**



**Fonte:** Elaborado pelos autores (2024).

## Cronograma

Com o intuito de garantir cumprimento do prazo estipulado para conclusão do projeto foi estabelecido um cronograma (Tabela 1), de forma a ordenar as atividades necessárias e facilitar sua atribuição aos respectivos responsáveis. De forma geral, optou-se em condensar o desenvolvimento das áreas de mecânica e eletrônica na primeira metade do semestre, permitindo a conclusão de toda a parte estrutural do protótipo a tempo da banca intermediária. Assim, minimiza-se os riscos de atrasos das entregas por problemas com peças faltantes ou danificadas. Para a segunda metade do semestre, o foco será o desenvolvimento da programação necessária para o acionamento do robô e o desenvolvimento dos nós do framework ROS para integração.

O motivo para a adesão de tal divisão consiste no fato de que a elaboração da programação depende diretamente da conclusão das atividades predecessoras relacionadas as demais áreas. Além disso, é necessário um período de capacitação para as tarefas relacionadas ao tema por parte integrantes da equipe, garantindo um melhor domínio do ROS.

A seguir, é possível observar a divisão das atividades de todas as áreas ao longo dos períodos (semanas), iniciando na semana do dia 05/02 e finalizando dia 29/05. Em destaque, estão os períodos 8, 16 e 17, que representam as bancas intermediária e final.

**Tabela 1** - Cronograma de atividades.

**Fonte:** elaborado pelos autores (2024).

### Mapeamento dos stakeholders

Os stakeholders envolvidos na elaboração do projeto são representados pelos membros da equipe, pelo professor orientador e pelo mentor, um representante da Embrapa que possui a função de ser o intermediário na comunicação da equipe com a empresa. A seguir é possível visualizar a atribuição de cada um dos membros no desenvolvimento do projeto (Tabela 2).

**Tabela 2 - Matriz de Stakeholders.**

Matriz de Stakeholders			
Stakeholder	Função no Projeto	Interesse no Projeto	Atribuições
Breno Alencar Araújo	Equipe de projeto	Aprendizado e preparação para mercado de trabalho	E, G
Bruno Morales Balkins	Equipe de projeto	Aprendizado e preparação para mercado de trabalho	E, G
Fernando Bichuette Assumpção	Equipe de projeto	Aprendizado e preparação para mercado de trabalho	E, G
Giulia Carolina Martins de Sampaio	Equipe de projeto	Aprendizado e preparação para mercado de trabalho	E, G
Vinícius Licks	Orientador	Contribuição para instituição (Insper)	V
Thiago Teixeira Santos	Mentor (Embrapa)	Contribuição para desenvolvimento de projetos dentro da Embrapa	V

**Legenda**

E	Execução do Projeto
V	Validação de Resultados
G	Gestão do Projeto

**Fonte:** elaborado pelos autores (2024).

### Riscos Envolvidos

Visando facilitar a compreensão e gerenciamento de possíveis adversidades que poderiam afetar negativamente o projeto, foi elaborada uma planilha de gestão de riscos (Apêndice B), na qual foram analisados fatores como gravidade, probabilidade de ocorrência, prioridade e medida de prevenção para cada risco identificado.

A divisão desses riscos na tabela foi efetuada de acordo com a área do projeto a que eles estão relacionados, facilitando a atribuição de responsáveis. Observa-se que os riscos com ID 1 a 13 envolvem questões técnicas, sendo que 1 a 5 englobam a área de mecânica, 7 a 10, eletrônica e 11 a 13, programação. Do ID 14 em diante foram considerados fatores operacionais, organizacionais, externos ou de gestão.

### Questões Éticas e Profissionais

Durante a execução do projeto, foi discutido algumas questões éticas e profissionais que são necessárias para o desenvolvimento de veículos terrestres autônomos (UGVs).

Um aspecto ético-chave é assegurar que os trabalhadores das áreas de fruticultura e silvicultura tenham a sua privacidade e anonimato garantidos. O UGV deve focar-se apenas na

avaliação e preservação dos talhões e não no levantamento de dados de pessoas. Isso pode ser alcançado ao desenvolver algoritmos e sistemas, dos quais precisam estar focados apenas nas informações críticas para os talhões, sem tocar em nenhuma informação individual dos trabalhadores, a menos que seja com o seu consentimento explícito.

Outro aspecto ético importante é a proteção dos dados coletados pela UGV. Dados críticos sobre as plantações, o estado do solo e informações meteorológicas, devem ser protegidos contra vazamento ou mau uso. Isso inclui o desenvolvimento de protocolos de segurança, criptografia de dados e sistemas de gerenciamento de acesso para proteger a confidencialidade e integridade dos dados.

O desenvolvimento ético de UGVs para monitoramento em fruticultura e silvicultura também requer atenção ao impacto ambiental. Portanto, é necessário garantir que a operação desses veículos não prejudique o meio ambiente, através de práticas que evitem a contaminação do solo, água e ar. Além disso, deve haver a adoção de medidas que reduzam a probabilidade de acidentes que danifiquem a flora e fauna. Para isso, pode-se recorrer à tecnologia de sensoriamento remoto de baixo impacto, práticas de operação que causem mínima compactação do solo, e protocolos de emergência que evitem vazamentos de produtos químicos das baterias.

Inserir UGVs na agricultura pode levar à automação de tarefas anteriormente desempenhadas por trabalhadores humanos com a consequência potencial de que menos pessoas encontrem trabalho. Isso pode produzir um efeito substancial sobre as comunidades agrícolas, na medida em que o emprego agrícola é um ponto de apoio vital para sua subsistência. É fundamental prever os impactos sociais e econômicos desta mudança e desenvolver estratégias para atenuar quaisquer resultados negativos. Por exemplo, a reciclagem da força de trabalho em outras áreas de emprego ou a introdução de programas de apoio à colocação-profissional para aqueles trabalhadores desempregados e necessitados.

## Normas Técnicas

Para este projeto os integrantes estarão atentos as normas técnicas durante toda a extensão dele. Assim, estão listadas abaixo essas normas relacionadas tanto a construção e

montagem do UGV, quanto ao possível uso que está sendo pensado para ele de monitorar talhões e a proteção de dados por causa de captura de imagens.

As normas seguidas neste projeto estão sendo as abaixo:

#### NR 12 - Segurança no Trabalho em Máquinas e Equipamentos:

Esta norma estabelece requisitos para garantir a segurança na operação de máquinas e equipamentos, incluindo aspectos como proteções, dispositivos de segurança, manutenção, inspeção e treinamento dos trabalhadores.

Sobre as normas que estão sendo seguidas durante o desenvolvimento deste projeto, serão feitas manutenções e inspeções no robô durante toda a montagem dele para que não haja falhas mecânicas e/ou eletrônicas. Terá um botão de emergência de fácil acesso para qualquer eventual ocasionalidade. Todos os integrantes trabalhando ativamente neste projeto estão cientes de todos os riscos e tem capacidade e treinamento para trabalhar com tudo que será necessário. Manual de informação para o uso correto e seguro de todos os equipamentos envolvidos. Botão de liga e desliga de movimentação do robô (Deadman Switch) que impeça o funcionamento autônomo e dificulte o acionamento sem intenção.

#### NR 10 - Segurança em Instalações e Serviços em Eletricidade:

Esta norma regulamenta as atividades envolvendo eletricidade, estabelecendo medidas de proteção para prevenir acidentes elétricos, como choques e incêndios, além de exigir qualificação e treinamento para trabalhadores que lidam com instalações elétricas.

No projeto do UGV a NR 10 é essencial para garantir a integridade dos operadores e a eficiência do veículo. Seguindo as diretrizes desta norma, são implementadas medidas de prevenção de acidentes elétricos, inspeções periódicas e treinamentos para a equipe. Cuidados com os circuitos eletrônicos e baterias foram tomados bem como EPI's foram usados durante a manipulação de peças energizadas. Assim, este projeto que está sendo desenvolvido do UGV, está em conformidade com as melhores práticas de segurança, assegurando um ambiente de trabalho seguro e confiável.

Outras normas relacionadas ao design e confecção do robô estão apresentadas na tabela 3 abaixo:

**Tabela 3** - Normas técnicas ABNT e ASTM.

Organização	Norma	Descrição
<b>ABNT (Associação Brasileira de Normas Técnicas)</b>	ABNT ISO 10218-1	Especifica os requisitos e orientações para um projeto seguro com robótica.  Descreve os perigos básicos associados a robôs e provê requisitos para eliminar ou reduzir adequadamente esses riscos
	ABNT ISSO 10218-2	Requisitos de segurança para a integração de robôs e sistemas robotizados industriais
<b>ASTM</b>	F2910	Requisitos para design e construção de UAS(terrestres também aplicam)

**Fonte:** elaborada pelos autores (2024).

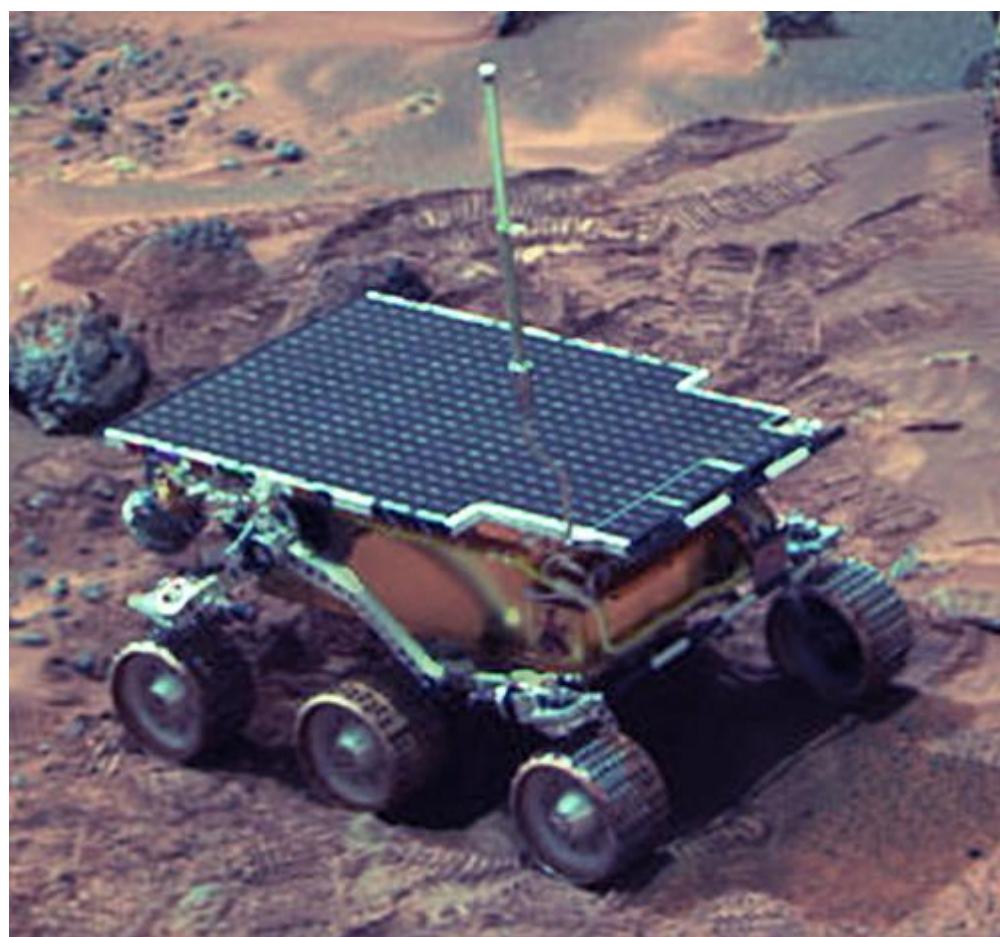
Também serão seguidas normas com relação ao uso do UGV, que terá câmera e fará imagens ao vivo 360 graus do ambiente em que for utilizado, de acordo com as normas seguidas na LGPD.

## Revisão do Estado da Arte

### Missão Mars Pathfinder e o Rover Sojourner

Em 4 de dezembro de 1996, foi lançado pela agência espacial americana (NASA) o rover Sojourner, como parte da missão Mars Pathfinder, com o objetivo de explorar a superfície de Marte. Essa missão pioneira marcou uma importante etapa na história da exploração espacial, pois o Sojourner foi o primeiro rover a operar de forma bem-sucedida no solo marciano. O local de pouso escolhido foi em uma região conhecida como Ares Vallis, uma das áreas mais rochosas do planeta. A escolha do local foi feita pois os cientistas acreditavam que ela ofereceria uma superfície relativamente segura para o pouso e uma ampla variedade de rochas. Por isso foi necessário que garantir que o rover pudesse operar em um terreno bastante acidentado (Nasa 2024)

**Figura 3 – Rover Sojourner**



**Fonte:** <https://www.space.com/17745-mars-pathfinder-sojourner-rover.html>. Acesso 15 mar. 2024.

Outro objetivo da missão era para também demonstrar possível desenvolver espaçonaves "mais rápidas, melhores e mais baratas", para contextualizar, as missões Viking custaram U\$1.06 bilhões em 1974 (The Planetary Society, 2024), já missão Mars Pathfinder teve um gasto total de \$ 265 milhões, incluindo o veículo de lançamento e as operações de missão. (NASA Space Science Data Coordinated Archive, 2024).

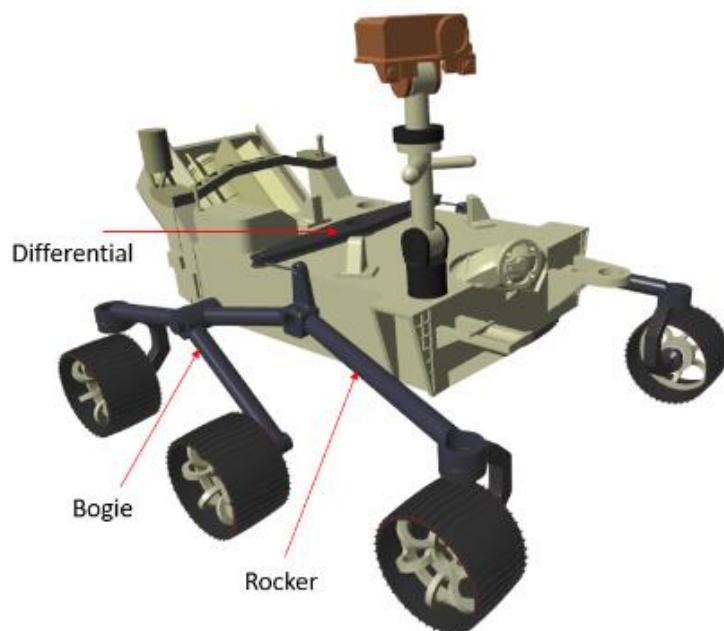
A grande distância entre a Terra e Marte faz com que as comunicações por rádio levem, em média, 13 minutos para viajar entre os dois planetas, impossibilitando o controle em tempo real do rover Sojourner. Por isso, o rover precisava operar de forma autônoma durante suas atividades em Marte. O Sojourner era capaz de navegar de forma independente usando um sistema de navegação autônomo. Ele recebia comandos da equipe de operações na Terra para se dirigir a pontos específicos de interesse na superfície marciana. Ao longo de sua rota, ele utilizava câmeras e lasers para detectar obstáculos, parando a cada 7 centímetros para capturar imagens com e sem laser, a fim de identificar mudanças no terreno, como rochas ou buracos. (Matijevic, 2024)

Quando encontrava um obstáculo, o rover podia girar para evitá-lo e depois retomar sua trajetória original. Ele também calculava continuamente sua posição relativa ao lander usando uma técnica chamada "dead reckoning" (navegação inercial), que envolvia medir a rotação das rodas e a mudança de orientação para determinar a distância percorrida e a direção da viagem. O rover também contava com sensores de contato nos para-choques dianteiros e traseiros, que funcionavam como uma linha de defesa final contra obstáculos não detectados pelos outros sistemas. Caso o rover não conseguisse atingir um ponto de passagem dentro do tempo determinado, ele parava e sinalizava um erro para a equipe de operações na Terra. (Matijevic, 2024)

O rover Sojourner, foi o primeiro a possuir um sistema de suspensão conhecido como rocker-bogie. Esse mecanismo consistia em três rodas de cada lado do rover, sendo uma roda dianteira apoiada por um balancim e duas rodas traseiras conectadas por um bogie. Esses componentes eram interligados por uma junta rotativa passiva, permitindo movimentos independentes das rodas em terrenos irregulares. A suspensão rocker-bogie foi projetada para

garantir que as seis rodas do rover permanecessem em contato com o solo, mesmo em terrenos acidentados. A simetria do sistema permitia uma distribuição equitativa da carga vertical entre as rodas, reduzindo o risco de afundamento em terrenos macios. Além disso, a geometria das rodas foi projetada para minimizar o risco de capotamento e garantir a estabilidade do rover em terrenos acidentados. (Cosenza et al. 2022)

**Figura 4 – Mars Rover**



**Fonte:** mathworks.com. Acesso em 12 mai. 2024.

O mecanismo diferencial presente na suspensão rocker-bogie permitia que o corpo do rover realizasse rotações de inclinação mínimas, contribuindo para a estabilidade do veículo em terrenos acidentados. Essa capacidade de adaptação a diferentes tipos de terreno era essencial para o sucesso das missões de exploração em Marte, onde o rover Sojourner enfrentava desafios variados. Por meio de uma análise cinemática detalhada, era possível determinar a configuração ideal da suspensão rocker-bogie com base no perfil do terreno. Essa análise prévia permitia ajustar a distribuição da carga vertical entre as rodas, garantindo a melhor performance do rover em diferentes condições de terreno. (Cosenza et al. 2022)

Dessa forma, a suspensão rocker-bogie foi essencial para a mobilidade, estabilidade e sucesso das missões de exploração do rover Sojourner em Marte. Desde a missão Mars

Pathfinder em 1997, esse tipo de suspensão tem sido amplamente utilizado nas missões de exploração marciana por sua capacidade de superar obstáculos em terrenos irregulares, assim como buracos grandes, com alturas próximas ao raio das rodas, e até mesmo obstáculos de altura superior. (Cosenza et al. 2022)

### Evolução da tecnologia de mapeamento

Como foi citado anteriormente, o rover Sojourner necessitava possuir um sistema autônomo, uma vez que a distância entre os planetas impossibilitava o controle em tempo real, então os cientistas da NASA utilizando a tecnologia da época conseguiram desenvolver um sistema autônomo. Com os avanços da tecnologia nas últimas décadas novos métodos para a criação de sistemas autônomos surgiram e não ficaram restritos a exploração espacial, um desses métodos é conhecido com SLAM (Simultaneous Localization And Mapping). O SLAM mapeia o ambiente em volta e localiza a posição dos veículos autônomos nesse mapa, assim é possível detectar obstáculos no caminho e evitá-los.

Existem diferentes tipos de SLAM, o Visual, o Lidar e o com múltiplos sensores. O SLAM Visual utiliza imagens adquiridas por câmeras para mapear o ambiente, essas câmeras podem ser desde câmeras RGB ou até aquelas que consigam ver profundidade, como as câmeras RGB-D esse tipo de SLAM pode ser implementado com baixo custo utilizando câmeras relativamente baratas. Além disso, uma vez que as câmeras fornecem uma grande quantidade de informações, que podem ser usadas para detectar pontos de referência (posições previamente medidas). (Mathworks, 2024)

**Figura 5** – SLAM feito com câmeras RGB-D



**Fonte:** Mathlab Visual SLAM. Acesso 12 maio 2024

O lidar é um sensor que utiliza lasers para fazer medições de distância. Esse dispositivo utiliza luz para coletar dados de uma superfície enviando um feixe de laser em direção a um alvo e medindo o tempo necessário para que esse sinal retorne. (Flyability, 2024)

Ao utilizar esses dados é possível gerar modelos 3D e mapas extremamente precisos, mais precisos que aqueles usando somente câmeras. Esse tipo de SLAM é capaz de criar uma nuvem de pontos, que é um conjunto de pontos de dados no espaço. Esses pontos são geralmente definidos em três dimensões (x, y, z) e são usados para representar a forma externa de um objeto ou cena no mundo real. É possível também utilizar uma variedade de outros sensores como IMUs (Unidades de Medição Inercial), GPS, radar e outros, junto com as câmeras e o lidar para criar mapas cada vez mais precisos e com mais detalhes. Esse tipo de SLAM é conhecido como multi-sensor. (Mathworks, 2024)

Essa nova tecnologia vem sendo usada em diversos produtos e setores, como em robôs aspirador, no setor de entretenimento, na medicina, UGV's, carros autônomos e outros. (Flybility, 2024)

## Robot Operating System (ROS) na robótica

O Sistema Operacional de Robôs, mais conhecido como ROS (do inglês, Robot Operating System), é um conjunto de bibliotecas de software e ferramentas que ajudam a construir aplicações de robótica. O ROS foi lançado inicialmente em 2007 pela Willow Garage, um laboratório de pesquisa em robótica, em colaboração com o Stanford Artificial Intelligence Laboratory e Open Robotics. Desde então tem sido mantido e aprimorado por uma comunidade global de engenheiros, desenvolvedores e entusiastas (Open Robotics, 2024).

O ROS é usado em uma variedade de aplicações de robótica, desde robôs móveis até braços robóticos industriais. Ele fornece serviços projetados para um cluster de computadores heterogêneos, como abstração de hardware, controle de dispositivos de baixo nível, implementação de funcionalidades comumente usadas, troca de mensagens entre processos e gerenciamento de pacotes (Open Robotics, 2024).

Embora o ROS não seja um sistema operacional em tempo real (RTOS), é possível integrá-lo com código de computação em tempo real. A falta de suporte para sistemas em tempo real foi abordada na criação do ROS 2, uma grande revisão da API do ROS que aproveita bibliotecas e tecnologias modernas para funções principais do ROS e adiciona suporte para código em tempo real e hardware de sistema embarcado (Open Robotics, 2024).

O ROS é usado para desenvolver uma ampla gama de aplicações de robótica. Ele fornece as ferramentas necessárias para criar simulações de robôs em qualquer ambiente, antes de implantar qualquer coisa no mundo real. As ferramentas como o Gazebo permitem criar simulações tão realistas quanto necessárias para cada aplicação (Open Robotics, 2024).

Além disso, o ROS é usado para desenvolver algoritmos de última geração para robôs, incluindo drivers de hardware, modelos de robôs, tipos de dados, planejamento, percepção, mapeamento e localização simultâneos (SLAM), ferramentas de simulação e outros algoritmos (Open Robotics, 2024).

A tecnologia ROS tem sido amplamente utilizada em várias aplicações de robótica, incluindo veículos autônomos, robôs de serviço, robôs industriais e até mesmo em missões espaciais. Com o avanço da tecnologia e a crescente demanda por robôs autônomos, o uso do ROS continuará a crescer no futuro. A capacidade do ROS de fornecer uma plataforma comum

para o desenvolvimento de aplicações de robótica torna-o uma escolha ideal para muitos projetos de robótica (Open Robotics, 2024).

### Mecanização agrícola e Revolução Verde

A mecanização agrícola se iniciou no século XVIII durante o processo da primeira revolução industrial, nessa época, com o aumento expressivo da população nas cidades, foi necessário que a oferta de alimento aumentasse em detrimento da diminuição de mão de obra no campo devido ao êxodo rural. No começo do século XX, começa a surgir os primeiros tratores movidos a combustíveis fosseis. A produção em massa desses e outros maquinários agrícolas ocasionou a queda de preço e popularizou o seu uso. (Junges, 2024)

**Figura 6** - Trator a querosene Waterloo Boy



**Fonte:** darcymaulsby.com Acesso 15 mar. 2024

No setor agropecuário, foram dados os primeiros passos para a automatização e produção em larga escala, que se tornaram comuns na indústria e na agricultura. Esse processo deu início à agroindústria, um fenômeno que ficou conhecido como Revolução Verde. No contexto da mecanização agrícola, a Revolução Verde desempenhou um papel crucial ao fornecer a base para a adoção de tecnologias avançadas. O aumento da produtividade e a

disponibilidade de excedentes agrícolas proporcionaram os recursos necessários para que os agricultores investissem em máquinas e equipamentos modernos, como tratores, colheitadeiras e sistemas de irrigação. (Brasil Escola, 2024)

Com esses avanços tecnológicos, na década de 1990, surgiu uma nova técnica de plantio, a Agricultura de Precisão. Esta técnica utiliza tecnologias modernas para aumentar a produtividade e a eficiência no uso de insumos agrícolas. (Instituto Agro, 2024)

A agricultura de precisão é uma abordagem baseada no uso intensivo de tecnologia para otimizar os processos agrícolas, desde o plantio até a colheita. Ela envolve o uso de sistemas de posicionamento global (GPS), sensores remotos, drones e softwares avançados para coletar e analisar dados sobre as condições do solo, clima e saúde das plantas. Essas informações são então utilizadas para tomar decisões precisas em tempo real, como a aplicação personalizada de fertilizantes e pesticidas, o manejo eficiente da irrigação e o monitoramento do crescimento das culturas. A agricultura de precisão permitiu aos agricultores maximizarem a produtividade, reduzir os custos de insumos e minimizar o impacto ambiental, promovendo uma gestão mais sustentável dos recursos agrícolas. (Embrapa, 2024)

### Integração de Unmanned Ground Vehicles (UGVs) na Agricultura

Uma notável evolução tecnológica na agricultura é a crescente integração de Unmanned Ground Vehicles (UGVs) nas operações agrícolas, representando uma inovação significativa. Os UGVs, também conhecidos como drones terrestres, estão se tornando cada vez mais proeminentes no setor, oferecendo uma gama de benefícios e funcionalidades avançadas. Projetados para operar de forma autônoma no terreno, esses veículos autônomos coletam dados e executam tarefas específicas com eficiência e precisão. Equipados com sensores e câmeras 3D, os UGVs têm a capacidade de mapear áreas cultivadas, monitorar o crescimento das plantas, identificar ervas daninhas e até mesmo realizar operações de pulverização e aplicação de pesticidas.

Uma das principais vantagens dos UGVs é sua habilidade de operar em terrenos acidentados ou de difícil acesso, superando desafios que máquinas tradicionais podem

enfrentar. Além disso, esses veículos autônomos podem ser programados para trabalhar de forma coordenada, aumentando a eficiência e reduzindo o tempo necessário para completar as tarefas agrícolas. A capacidade dos UGVs de realizar operações precisas e autônomas contribui significativamente para a otimização das atividades agrícolas, permitindo uma abordagem mais eficiente e sustentável no campo. (Katikaridis et al, 2022)

Fundada em 2017 na Filadélfia anteriormente conhecida como Augean Robotics, a Burro.AI é uma empresa que se especializa à construção de Veículos Terrestres Não Tripulados (UGVs). Composta por uma equipe de engenheiros e pesquisadores especializados. A missão da empresa é resolver o problema da mão de obra enfrentado pelos agricultores, tornando os robôs uma realidade. Para isso, começaram com uma plataforma robótica colaborativa chamada Burro, que ajuda as pessoas a trabalharem de forma mais produtiva. (Burro.AI, 2024)

Os veículos da Burro.AI podem operar em uma variedade de ambientes e lidar com terrenos acidentados e áreas de cultivo de difícil acesso. Sua estrutura robusta e capacidade de navegação autônoma permitem que eles se movam com facilidade em terrenos irregulares, garantindo eficiência e precisão em todas as etapas do processo agrícola.

Atualmente, a Burro.AI oferece quatro modelos diferentes de veículos autônomos, cada um projetado para atender a uma variedade de necessidades e aplicações. O modelo Original é o produto principal, projetado para ser compacto e adaptável para uma variedade de aplicações. O modelo XL possui uma plataforma mais larga, permitindo que os agricultores carreguem mais volume e cargas mais pesadas. O modelo Grande, o veículo mais poderoso da linha, foi projetado para rebocar mais, carregar mais e fazer mais em ambientes agrícolas. (Burro.AI, 2024)

Equipados com 12 câmeras e um Lidar, os veículos da Burro.AI podem mapear o ambiente 3D ao seu redor e identificar objetos. Eles utilizam tanto a visão computacional quanto o GPS de alta precisão para aprender e ensinar a si mesmos a navegar em novos ambientes. Isso significa que os veículos da Burro.AI podem operar em “trilhos digitais” através de configurações complexas durante todo o dia, mesmo na ausência de sinal GPS. (Burro.AI, 2024)

Os veículos da Burro.AI foram projetados para operar em uma variedade de ambientes, desde terrenos acidentados até áreas de cultivo de difícil acesso. Sua estrutura robusta e sua capacidade de navegação autônoma permitem que os veículos se movam com facilidade em terrenos irregulares e desafiadores, garantindo eficiência e precisão em todas as etapas do processo agrícola. (Burro.AI, 2024)

A grande diferença de proósito entre esse UGV e o que será construído está na finalidade: enquanto o veículo da Burr.AI está endereçado para atividades quem envolvem força trativa, o rover foca no monitoramento dos talhões.

**Figura 7 – UGV Burro Grande.**



**Fonte:** <https://burro.ai/burro-grande/>. Acesso 15 mar 2024.

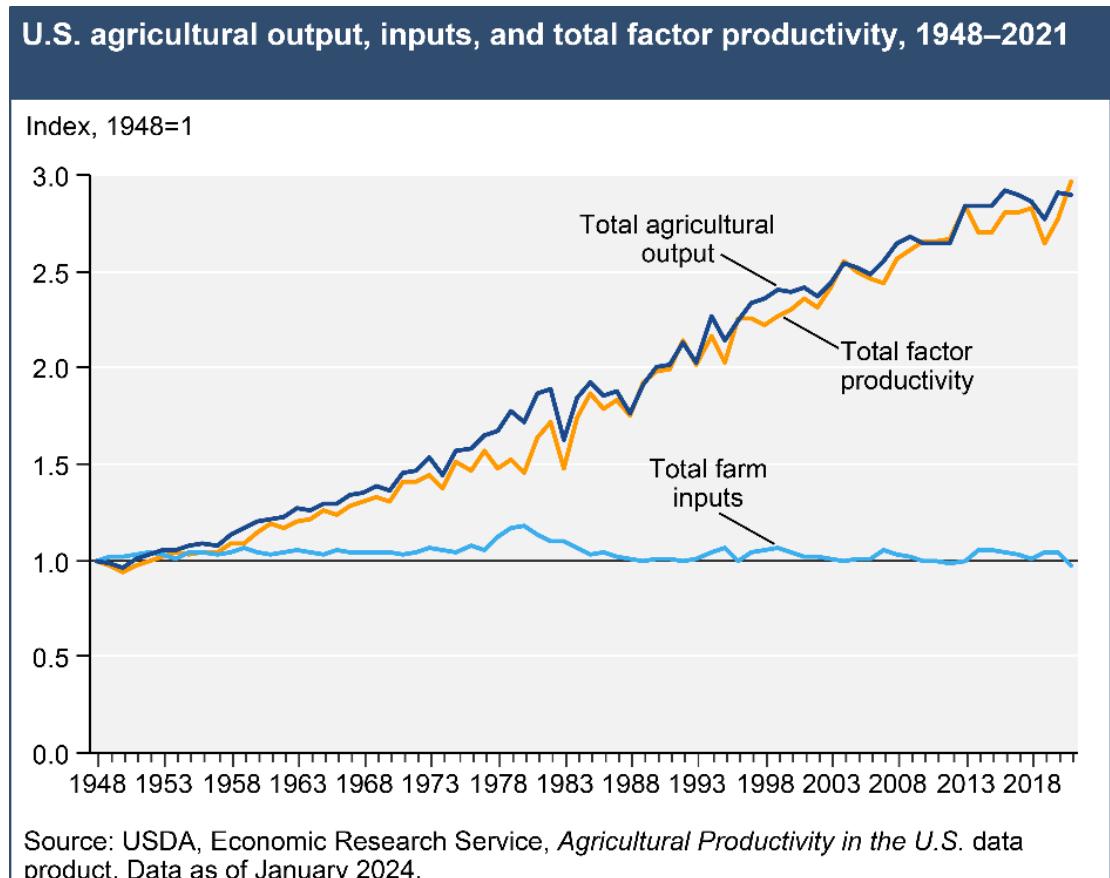
### **Impacto das inovações tecnológicas na produtividade agrícola**

As inovações tecnológicas resultantes das revoluções industriais transformaram profundamente o setor primário. De acordo com dados do Serviço de Pesquisa Econômica do Departamento de Agricultura dos EUA, é amplamente aceito que a produtividade agrícola nos EUA tem aumentado a uma taxa média anual de 1,46%. (Serviço de Pesquisa Econômica do Departamento de Agricultura dos EUA, 2024)

Esse aumento na produtividade agrícola provavelmente se refere à melhoria na eficiência da produção de alimentos, permitindo que mais alimentos sejam produzidos com menos insumos ao longo do tempo. Esse aumento na produtividade é atribuído a uma combinação de fatores, incluindo melhorias na genética das plantas, práticas agrícolas mais eficientes e o uso de tecnologias avançadas, como a agricultura de precisão. A adoção dessas tecnologias permitiu aos agricultores otimizarem a aplicação de insumos, como água e fertilizantes, melhorar o manejo de pragas e doenças e aumentar a eficiência da mão de obra. Como resultado, a agricultura moderna é capaz de produzir mais alimentos em menos terra e com menos insumos de forma recorde. (Serviço de Pesquisa Econômica do Departamento de Agricultura dos EUA, 2024)

Portanto, as inovações tecnológicas têm desempenhado um papel crucial na transformação da agricultura, permitindo aumentos significativos na produtividade e eficiência. Isso tem implicações importantes para a segurança alimentar e a sustentabilidade ambiental, à medida que a população mundial continua a crescer.

**Figura 8** – Produtividade agrícola nos Estados Unidos.



**Fonte:** ers.usda.gov Acesso 15 mar. 2024.

## Metodologia

Utilizou-se da metodologia PDP (Processo de Desenvolvimento de Produto) para elaboração e detalhamento das atividades. Ela é dividida em três macrofases: Pré-Desenvolvimento, Desenvolvimento e Pós-Desenvolvimento. Entre cada uma das fases, o método PDP de Rozenfeld et al. (2006) prevê gates, que, no contexto deste projeto, foram adaptados e alocados para as bancas intermediária e final após o projeto conceitual e o projeto detalhado, respectivamente.

**Figura 9** – Processo completo de desenvolvimento de produto. As etapas representadas em cinza não foram contempladas neste projeto.



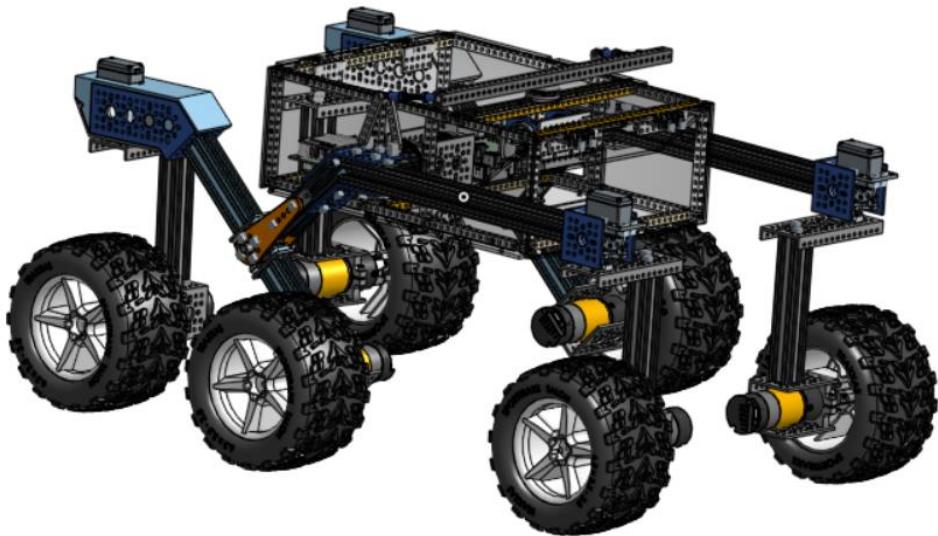
**Fonte:** os autores (2024).

O pré-desenvolvimento abrangeu o planejamento estratégico, em que o produto a ser desenvolvido e o escopo do projeto foram definidos. Fase essa já realizada pela Embrapa antes do início do semestre letivo e acertada com os integrantes do projeto durante as primeiras reuniões. É no pré-desenvolvimento também que a análise da viabilidade econômica, prazos e riscos foram acordados com a empresa e o mentor. Parte do projeto detalhado (aquisição) foi adiantada nessa fase, visando viabilizar o desenvolvimento do robô no tempo disponível.

O processo de desenvolvimento, dividido em cinco fases menores: projeto informacional, projeto conceitual, projeto detalhado, preparação para produção e lançamento do produto, abrangeu as etapas de execução do projeto. Sendo que as duas últimas etapas não serão contempladas neste projeto.

Durante o projeto informacional, abordado na seção de coleta de dados sobre o projeto, detalhou-se os requisitos para convertê-los em especificações de projeto, buscando entender os requisitos do cliente e investigando soluções semelhantes no mercado, como o Burro.AI citado na revisão do estado da arte. Durante o desenvolvimento do projeto conceitual as principais funções foram descritas e sugeriram-se possíveis soluções em uma matriz de solução para a criação de variantes, descritas na seção de análise de dados coletados. Ao final dessa etapa o produto já possuía alguma forma geométrica e estética definida, como a montagem no CAD (Computer Aided-Design) com os principais subsistemas definidos.

**Figura 10** – Esquemático 3D do UGV.



**Fonte:** OnShape Model. Acesso 10 fev. 2024

O projeto detalhado ainda pode ser dividido em três ciclos: detalhamento, em que se descrevem sistemas e componentes, como os hardwares de lógica e potência, a estrutura física e o software. Aquisição, em que há o contato com os fornecedores. Nessa etapa algumas compras de partes faltantes foram realizadas. Por fim, o ciclo de otimização, para avaliar e integrar os sistemas, realizando testes em conjunto e documentar os processos. Em paralelo ao detalhamento, há a preparação da produção e finaliza-se o macroprocesso com o lançamento do lote piloto e o lançamento do produto, etapas essas não contempladas nesse projeto.

O pós-desenvolvimento é dividido em duas fases: acompanhamento e descontinuidade. O primeiro para avaliar o desempenho do produto e o segundo para decidir sobre a retirada do produto do mercado. Essa fase ficará a cargo da empresa.

### Coleta de dados sobre o projeto

Durante a coleta de dados, para o projeto informacional, foram realizadas duas reuniões com o mentor: a primeira para melhor compreensão do problema e dos requisitos da empresa

em relação ao projeto e a segunda para formalizar o fechamento do escopo, iniciada na etapa de pré-desenvolvimento, garantindo um alinhamento entre as expectativas de ambas as partes.

Ao longo do processo de entrevistas se mostrou como uma necessidade um maior entendimento do ambiente de monitoramento (talhões), bem como do processo de monitoramento por terra em si, garantindo que o protótipo seja capaz de suportar as adversidades do ambiente de operação e atender aos requisitos especificados.

No processo de investigação do ambiente de operação descobriu-se que a distância entre as plantações (filas), ou seja, o caminho percorrido pelo robô, apresentaria uma largura de aproximadamente 1,8 m, podendo conter irregularidades no solo e crescimento de vegetação em seu centro, o que poderia gerar uma maior dificuldade no deslocamento do rover. Apesar de tal adversidade, não foi salientada a necessidade de se cruzar a vegetação nem de mudanças no vão livre (distância entre o solo e a parte mais baixa do robô), contanto que isso não interfira na qualidade da imagem registrada durante o monitoramento.

Quanto ao relevo dos talhões, constatou-se que não haveria a necessidade de que o protótipo suportasse altas declividades como ocorreria em plantações de café de montanha, por exemplo, onde as inclinações variam entre 30° e 40°. Todavia, o sensoriamento da inclinação se mostrou atrativo para a empresa, permitindo a elaboração de testes capazes de informar a declividade máxima suportada pelo robô, informação importante para avaliar a compatibilidade do rover com as plantações candidatas a aplicação dessa tecnologia e trazendo mais informações ao mapeamento realizado.

Considerando agora o processo de monitoramento propriamente dito, alguns tópicos devem ser analisados, dentre eles a qualidade da imagem necessária para o monitoramento e a forma de referenciamento utilizada pelo robô, importante para possibilitar a futura implementação do sistema de navegação autônoma, que embora não englobe o escopo deste projeto, foi considerada como um objetivo da empresa para futuras versões do protótipo.

Quanto as imagens coletadas, é necessário garantir que as câmeras sejam fixadas de forma estável no robô, implicando em uma melhor visualização das plantações, evitando que seu registro fique trêmulo em decorrência do deslocamento do protótipo. Também devem permitir ajustes manuais para que se adequem a situações diversas. Outra necessidade da

empresa é que seja possível captar as árvores ao longo de toda sua extensão, através de câmeras laterais. No processo de entrevistas, as câmeras selecionadas como as melhores alternativas para a empresa foram a Intel RealSense Depth D435i e a Logitech C920 HD PRO, a primeira apresenta duas câmeras de infravermelho (visão estéreo), uma câmera RGB e um projetor de matriz de pontos infravermelho para estimar distâncias e auxiliar no mapeamento com a fusão de sensores e a segunda apresenta uma câmera RGB.

Quanto ao referenciamento do robô, para implementação de próximas versões com sistema autônomo, o equipamento de preferência escolhido pela empresa foi o LiDAR Ouster de 16 canais (Figura 10). Este instrumento é importante para implementação do SLAM (Simultaneous Location and Mapping), sistema que permite ao robô construir uma representação do sistema ao seu redor a partir da fusão de sensores, junto à câmera e ao hodômetro (encoders) dos motores do rover, podendo estimar sua localização neste ambiente, possibilitando que o deslocamento ocorra de forma autônoma.

**Figura 11** - LiDAR Ouster 16 canais.



**Fonte:** <https://ouster.com/>. Acesso em 16 fev. 2024.

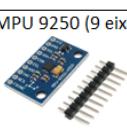
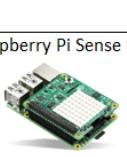
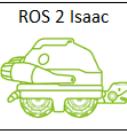
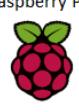
Assim como as câmeras estéreo, o LIDAR serve para medir a distância dele aos objetos. Ao lançar um feixe de lasers em torno dele, sensores captam o retorno desses raios devido a reflexão por outros objetos e calculam a distância com base no tempo entre envio e recepção dos feixes em determinada direção.

## Análise dos dados coletados

Antes da segunda entrevista, utilizada para o fechamento do escopo e da etapa do projeto informacional, referido na seção de metodologia, foi elaborada uma matriz contendo as principais variantes de solução, ou seja, algumas alternativas para os requisitos apresentados pelo representante da empresa ao longo do primeiro encontro.

A partir das informações fornecidas a respeito do ambiente e forma de monitoramento, foi possível analisar as hipóteses levantadas, selecionando as alternativas de solução mais apropriadas. A matriz contendo essas opções está representada na tabela 4 e, vale ressaltar, que embora tenha sido realizada a coleta de dados a respeito do LIDAR e seu funcionamento, o componente não foi incluído na análise, uma vez que seu modelo já havia sido definido no escopo do projeto. Isso vale para a estrutura mecânica e grande parte da eletrônica do robô, tendo em vista que o projeto JPL Open Source da Nasa já havia sido adotado como ponto de partida. No entanto, o estudo das possibilidades reforçou as escolhas adotadas.

**Tabela 4 - Matriz de variantes de solução.**

		Variantes de Solução			
		1	2	3	4
Funções	Monitoramento (câmera)	Intel Realsense Depth D435i 	Logitech C920 HD PRO 	-	-
	Sensoriamento inclinação	MPU 6050 (6 eixos) 	MPU 9250 (9 eixos) 	Via câmera (BML055)	Raspberry Pi Sense Hat 
ROS	ROS 1	ROS 2 Humble 	ROS 2 Iron 	ROS 2 Isaac 	
Linguagem de Programação	Python 	C++ 	-	-	
Sistema Operacional	Raspberry Pi OS 	Ubuntu 	-	-	
Display IHM	LCD 16x2 - módulo I2C 	TFT SPI 	Nextion NX8048P070-011C 	-	

**Fonte:** elaborado pelos autores (2024).

Analisando o primeiro e tópico matriz, referente ao tipo de câmera utilizada no monitoramento, foi escolhido o modelo Intel RealSense D435i, uma vez que esta foi a opção preferida pela empresa, por apresentar maior número de recursos, principalmente no que diz respeito a aplicação do SLAM.

Do ponto de vista do sensoriamento de inclinação, optou-se em utilizar o sensor BMI055 presente na própria câmera, diminuindo a quantidade de adaptações necessárias no sistema elétrico do robô, uma vez que a Raspberry Pi apresenta poucos pinos disponíveis.

Considerando a aplicação do ROS, foram analisadas algumas opções. O ROS1 e ROS2 versões Humble, Iron e Isaac. As versões Humble e Iron, são alternativas fornecidas pelo próprio repositório do projeto, facilitando a implementação dos códigos prontos e evitando retrabalho. Já o Isaac, sugestão do mentor, é uma versão do ROS2 criada pela Nvidia, com facilidades voltadas a implementação de inteligência artificial em aplicações de robótica. Após análise, a versão escolhida foi o ROS2 Humble, uma vez que é uma versão do ROS2 com suporte a longo prazo, além de estar disponível no repositório, facilitando a execução do projeto.

Do ponto de vista da linguagem de desenvolvimento, foram analisadas duas opções: Python e C++. Tendo em vista a experiência da equipe de projeto com cada uma das linguagens e as recomendações da empresa, o Python foi selecionado.

Analizando o sistema operacional escolhido, optou-se por usar o Ubuntu em vez do Raspberry Pi OS, tendo em vista que a segunda opção não apresentava suporte para a biblioteca do ROS. A opção rejeitada chegou a ser testada, porém houve dificuldade no processo de instalação do framework, levando ao abandono do sistema operacional.

Após a seleção do Ubuntu, outra escolha se fez necessária, sendo ela a modalidade da imagem instalada na Raspberry Pi, uma vez que o sistema operacional poderia ser instalado como desktop ou servidor. Inicialmente, a versão desktop foi adotada por ser mais intuitiva e fácil de trabalhar, todavia, após a necessidade de realizar a visualização das imagens captadas pela câmera, observou-se grande delay no funcionamento do equipamento. Para resolução desse problema, foi efetuada a transição do sistema operacional para a modalidade de servidor, permitindo o acesso remoto e poupança de recursos de processamento da Raspberry Pi.

Quanto ao display adotado, a principal limitação que levou a escolha do modelo LCD 16X2 foi a presença de um módulo I<sup>2</sup>C, que poderia ser aplicado com maior facilidade no projeto, uma vez que a placa de controle apresenta um barramento I<sup>2</sup>C disponível. Além disso, a facilidade de programação do display em questão também se mostrou atrativa.

Após decisão dos componentes usados foi possível realizar um esboço da solução (Figura 11). Note que os elementos numerados de 1 a 4 representam as câmeras com seus suportes, 5 o LiDAR, 6 o botão de emergência e 7 o display da IHM. O elemento 8, por sua vez, não foi discutido na matriz, mas representa um multímetro do projeto Open Source e não deve ser confundido com o display.

**Figura 12 -** Esboço da solução.



**Fonte:** elaborado pelos autores (2024).

A etapa de PDP seguinte ao projeto conceitual detalha o projeto em eletrônica, mecânica e software.

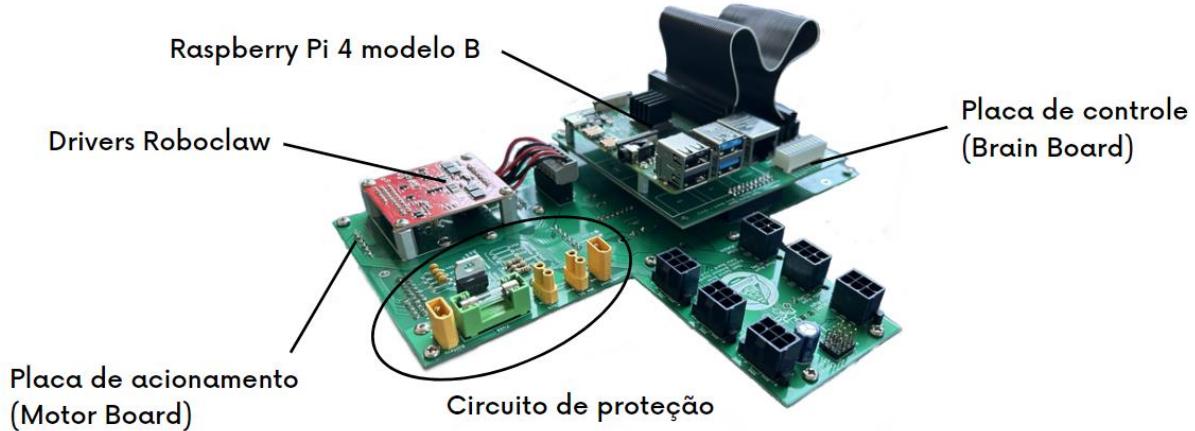
## Detalhamento Técnico do Protótipo

### Eletrônica

#### Esquemáticos e Detalhamento Geral de Funcionamento

A eletrônica do protótipo é composta por duas placas (figura 13): a placa de acionamento dos motores (Motor Board), onde estão localizados os circuitos de potência e a placa responsável por comandar todo o funcionamento do sistema (Brain Board), onde estão os circuitos de controle e está alocada uma Raspberry Pi 4, modelo B de 8 GB, com microprocessador quad-core de 64-bits ARM-Cortex A72, rodando a 1,5 GHz.

**Figura 13 – Placas Eletrônicas.**



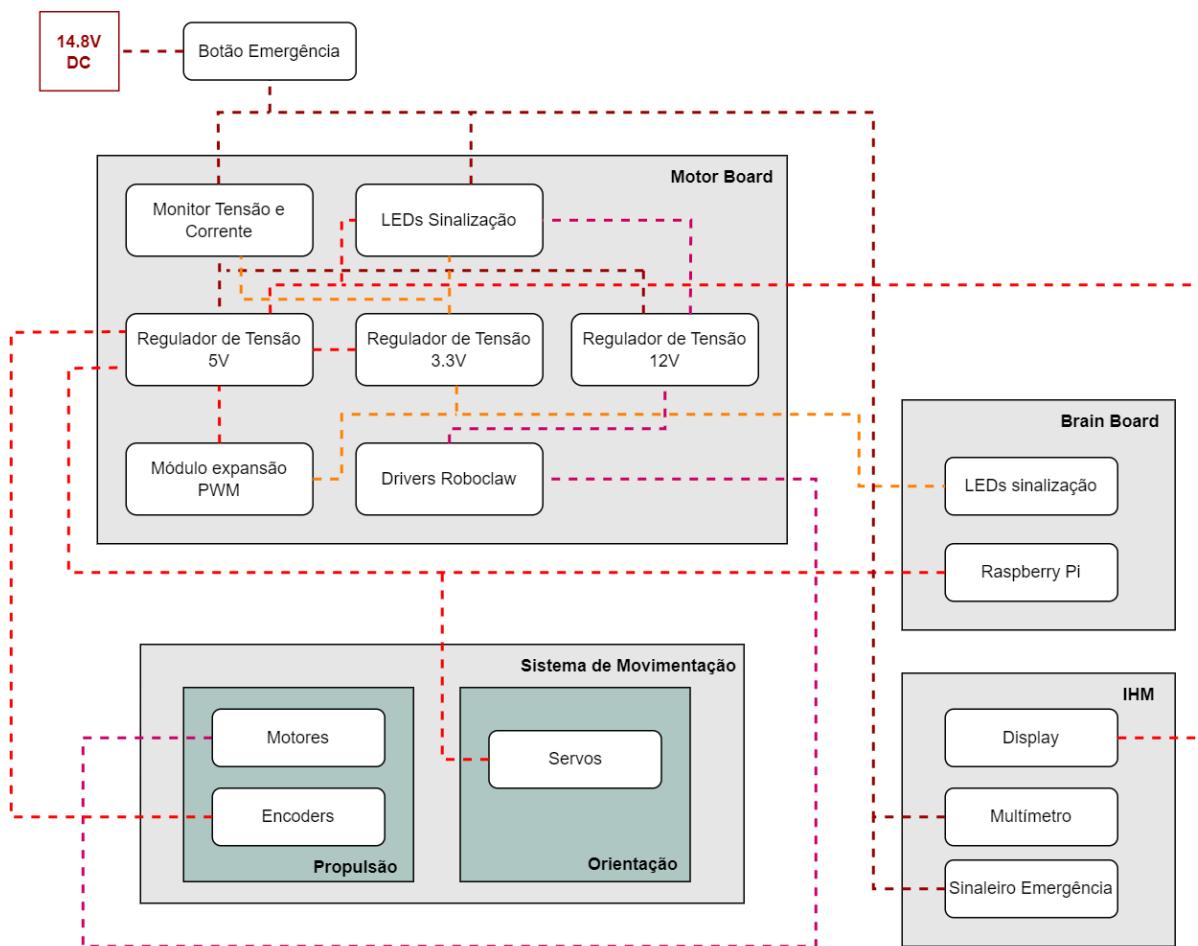
**Fonte:** elaborado pelos autores (2024).

Com o objetivo de facilitar a compreensão do funcionamento de cada uma das placas e sua interação, foram elaborados dois diagramas de blocos, o primeiro para representação do sistema de alimentação e o segundo para ilustrar os sinais enviados entre os componentes. Além dos diagramas, o esquemático dos circuitos elétricos de ambas as placas também está disponível no Apêndice C.

Analizando primeiro a alimentação do sistema (Figura 10), uma bateria fornecendo 14,8V DC é conectada diretamente à placa de acionamento, passando por um circuito de proteção, que possui um fusível, um resistor de 1 kΩ e um diodo, que atuam como proteção do

circuito dos motores. Além disso, um módulo de monitoramento de tensão e corrente (INA260) é utilizado para medição de sobrecorrente, além de um multímetro, que é utilizado para monitoramento do usuário dos dados de corrente e tensão.

**Figura 14** - Diagrama elétrico de alimentação.



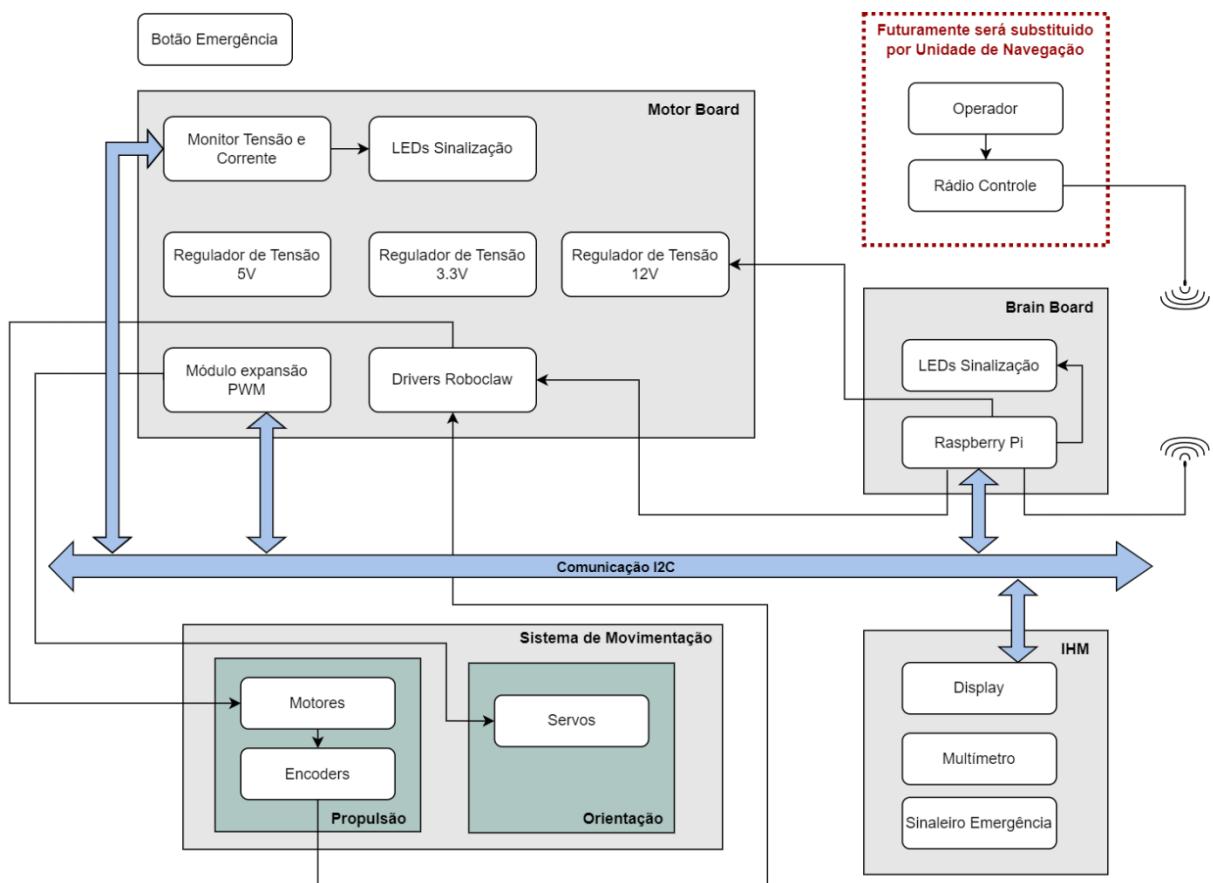
**Fonte:** elaborado pelos autores (2024).

A tensão 14,8V é usada na alimentação de dois reguladores, um step-down de 12V (D24V22Fx), usado para alimentar os drivers da Roboclaw e outro de 5V (D24V150Fx). Há também um terceiro regulador de tensão que converte a tensão de 5V para 3,3V (MIC2937A-3.3WT), alimentando o módulo de expansão de PWM (PCA9685), o monitor de tensão e corrente e os LED's de sinalização de ambas as placas. A tensão de 5V, por sua vez, também realiza a alimentação do módulo de expansão de PWM, dos servos, da Raspberry Pi, do display

da IHM e dos encoders dos motores. Os motores são alimentados pela Roboclaw via PWM com tensão máxima de 12V.

Considerando agora o diagrama de sinais do sistema (Figura 13), note que um novo bloco foi adicionado, o conjunto entre o operador e o rádio controle, responsável por enviar os comandos por meio de ondas de rádio até a Raspberry Pi, que recebe o sinal por meio de um módulo USB. Os dados recebidos pelo microprocessador são enviados na serial e usados para o controle dos motores e servos.

**Figura 15** - Diagrama elétrico de sinais.



**Fonte:** elaborado pelos autores (2024).

A malha de controle dos seis motores é comandada pela Raspberry Pi, que recebe a referência do rádio controle e envia os comandos para os drivers da Roboclaw, que realizam o acionamento via PWM. O fechamento da malha é realizado pelos encoders, que são

responsáveis pelo sensoriamento, enviando os dados coletados ao controlador presente no driver.

Os quatro servos, por sua vez, são controlados em malha aberta (fechada internamente) acionados por meio de um sinal PWM. Este sinal é fornecido pelo módulo PCA9685 (expansão), usado para ampliar a quantidade de servos que podem ser controlados, por meio da geração de 16 novos canais PWM. A comunicação desse módulo com a Raspberry Pi ocorre por meio do protocolo de comunicação I<sup>2</sup>C.

Considerando a comunicação do microcomputador com o módulo de monitoramento de tensão e corrente (INA260), nota-se que o envio de dados também ocorre por meio de um protocolo de comunicação I<sup>2</sup>C. O módulo também se comunica com os LED's de sinalização para alertar o usuário caso seu resistor de shunt interno detecte alguma alteração no sistema de alimentação.

Além do módulo de expansão e do monitor, a comunicação com o display da IHM, que será futuramente implementada, também ocorrerá de acordo com o protocolo I<sup>2</sup>C. O motivo para tal escolha é que a placa de controle apresenta um barramento I<sup>2</sup>C disponível, que poderia ser adaptado para a conexão do display com baixa quantidade de adaptações necessárias.

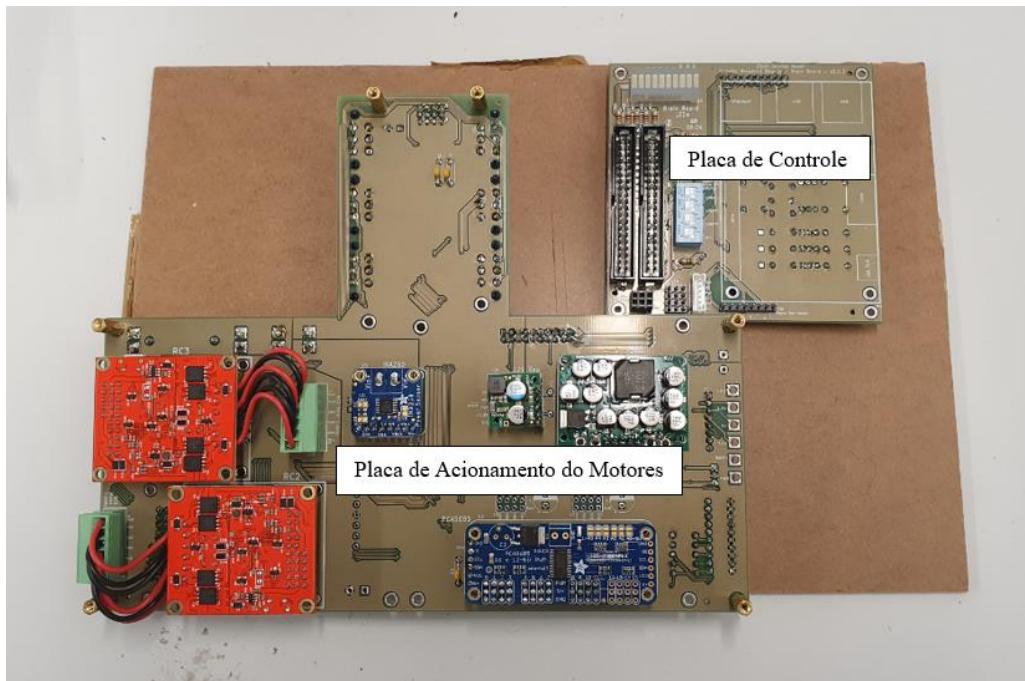
Outra observação importante em relação ao diagrama é que a Raspberry Pi também está responsável pelo controle do regulador de tensão de 12V por meio de um sinal de enable, ligando ou desligando o output de 12V.

Analizando um pouco mais a placa de controle, observa-se a presença de alguns LED's de sinalização, esses componentes possuem como finalidade a comunicação com o usuário, sinalizando por exemplo quando ocorre a transferência de dados pela serial, acendendo quando os bits são transferidos.

Por fim, o último fluxo de dados a ser analisado é o indicador do estado de emergência, no qual um sinal digital de nível lógico alto é enviado pelo botão de emergência ao microcomputador, que comunica a situação ao usuário por meio do display da IHM.

Uma vez conhecidos os componentes principais de cada placa, assim como seu funcionamento, são possíveis entender a lógica de controle e acionamento do dispositivo. A seguir estão ilustradas as duas placas após soldagem dos componentes (Figura 14).

**Figura 16** - Placas após solda de componentes.



**Fonte:** elaborado pelos autores (2024).

## Especificação dos Motores

Tendo em vista que os motores são responsáveis por fornecer a potência necessária para impulsionar o robô, o detalhamento e especificação desses atuadores se torna vital para verificar se os requisitos de projeto estão sendo atendidos.

Para estes componentes, existem dois tipos diferentes de motores sendo utilizados: os motores DC escovados, usados para propulsão do sistema e os servos, utilizados para a mudança de orientação do robô.

Os motores DC são do modelo “5203 Series Yellow Jacket Planetary Gear Motor”, da fabricante GoBilda e possuem torque máximo de 38 kg.cm. São utilizados 6 motores no total

para realizar a locomoção do robô, cada um equipado por um encoder magnético, que realiza a leitura através de um sensor de efeito hall. Essas leituras podem ser acompanhadas em um nó específico, o ‘roboclaw\_movemotor.py’, contido em: osr-rover-code/scripts/ no repositório apenas dos códigos do robô. (GitHub, 2024)

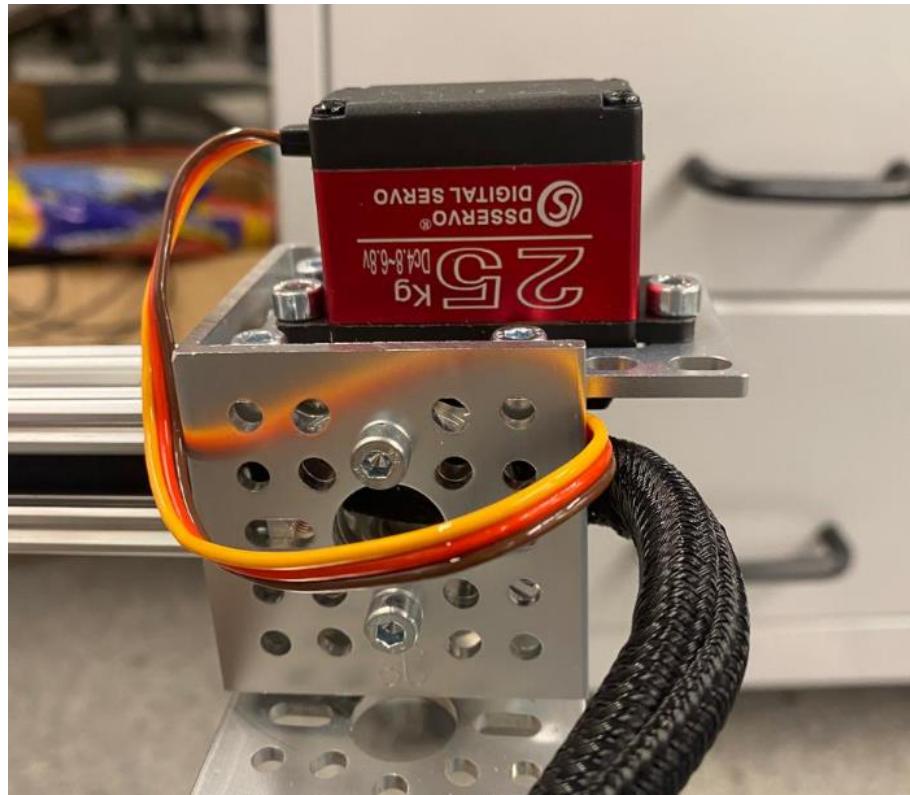
**Figura 17** – Motor Alocado no Robô



**Fonte:** elaborado pelos autores (2024).

Os servos motores adotados são do modelo “Dsservo DS3225” e apresentam torque máximo de 21 kg.cm com os 5 V fornecidos pela placa de acionamento. No total são necessários 4 atuadores desse tipo para realizar o controle da orientação do robô, possibilitando a dinâmica lateral.

**Figura 18 --** Servo Alocado no Robô

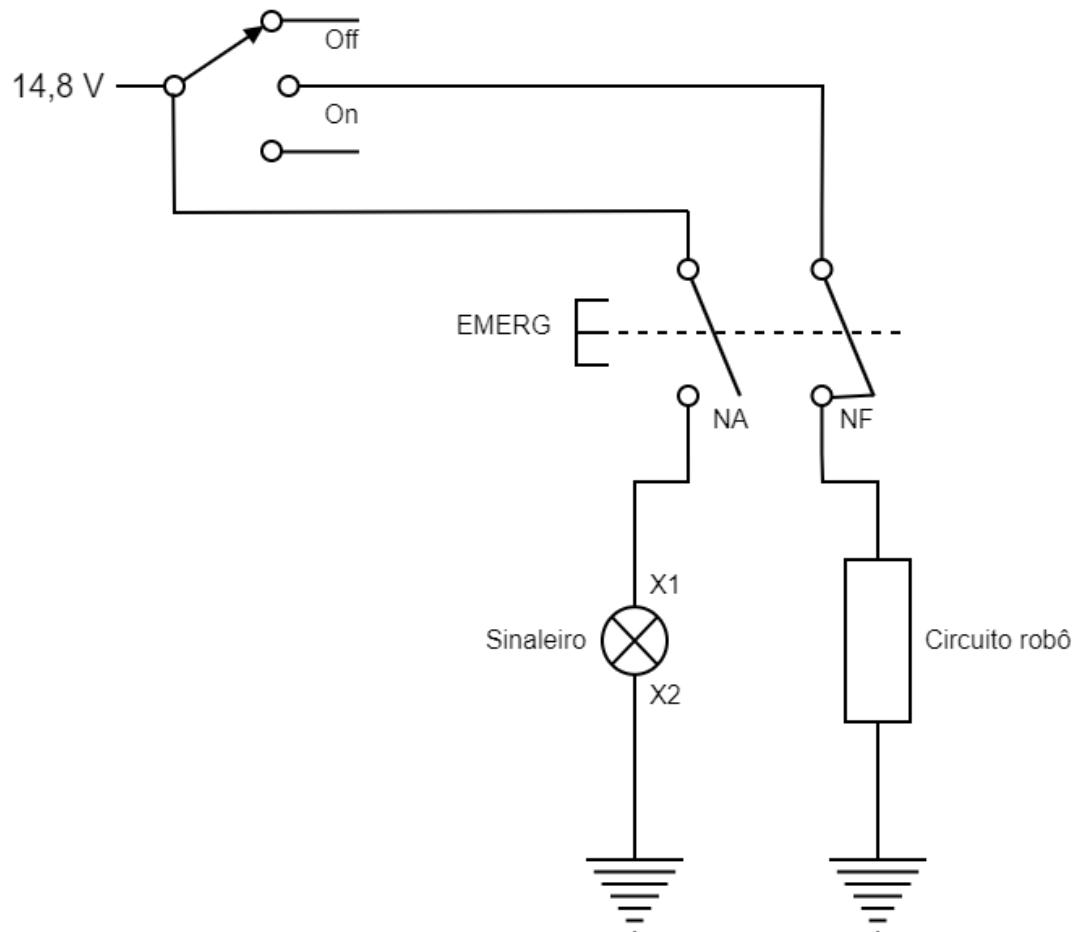


**Fonte:** elaborado pelos autores (2024).

#### Detalhamento Circuito de Emergência

Para assegurar a adequação do protótipo à norma de segurança do trabalho em máquinas e equipamentos (NR-12), se fez necessária a inclusão de um circuito de emergência (figura 17), contendo um botão de emergência físico e garantindo a parada total do robô, além da desenergização das placas de controle e potência, através de uma chave do tipo normalmente fechada, responsável por realizar o corte da alimentação fornecida para ambas as placas.

**Figura 19** – Circuito de Emergência.



**Fonte:** elaborado pelos autores (2024).

**Figura 20** – Sinaleiro de Emergência.



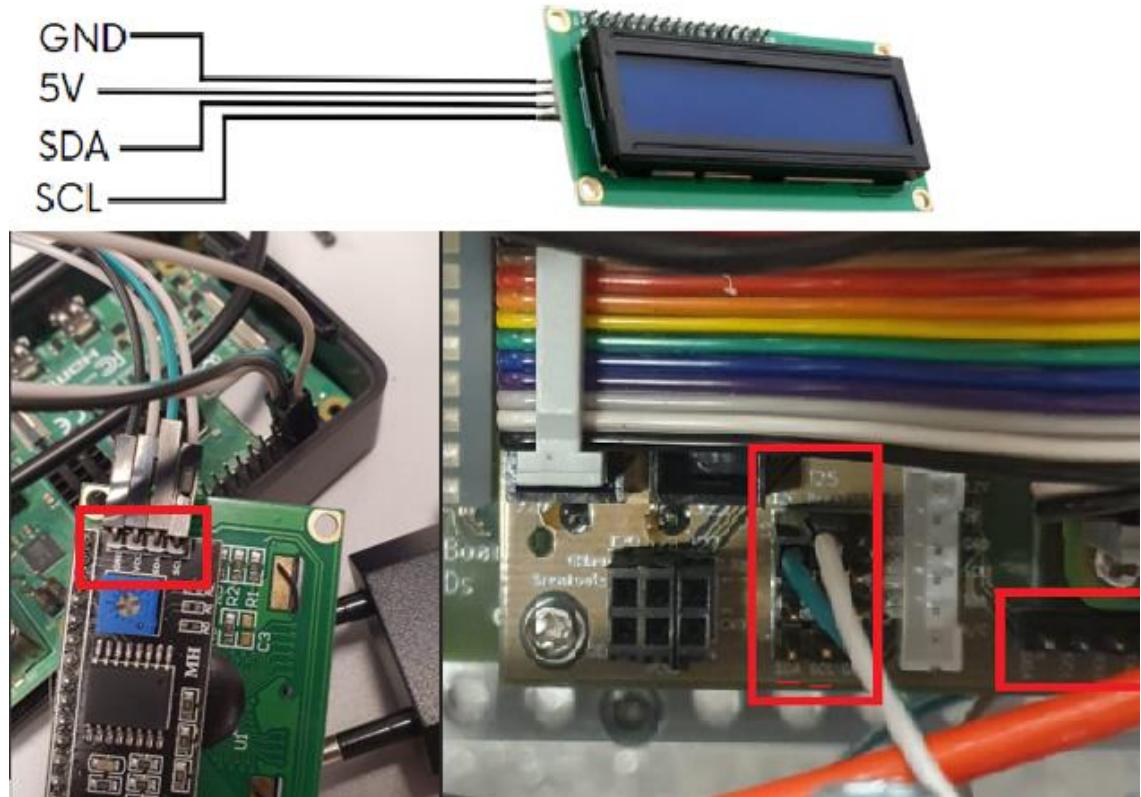
**Fonte:** elaborado pelos autores (2024).

Outro ponto que será posteriormente detalhado será a presença de um botão de emergência remoto, presente no sistema de rádio controle, possibilitando a desenergização dos atuadores e parada do robô à distância.

#### Detalhamento Circuito Display

Para exibição de informações relevantes a respeito do status do robô, como conectividade, percentual de bateria etc., instalou-se o display lcd 16x02 com módulo de conversão I<sup>2</sup>C, assim como previsto na matriz de solução do projeto. Foi necessário alimentá-lo com 5V em um dos pinos de alimentação disponíveis e assinalados pelo silkscreen da brain board (caixa vermelha da direita na figura abaixo), bem como conectar os pinos SDA e SCL.

**Figura 21** – Ligações dos pinos SDA e SCL do display na placa de controle



**Fonte:** elaborado pelos autores (2024).

Após conectar o display à placa, acessou-se o servidor via SSH e instalou-se a biblioteca RPi.GPIO.I2C-LCD pelo terminal:

```
ubuntu@embrapa:~$ sudo pip3 install RPi-GPIO-I2C-LCD
```

Com a biblioteca instalada, foi necessário buscar o endereço, (0x27), i2c do display usando o comando:

```
ubuntu@embrapa:~$ i2cdetect -y 1
```

No repositório proprietário do projeto no GitHub o script ‘ihm\_lcd.py’ contém a sequência de passos para exibição das informações.

A estratégia adotada para redução de consumo foi atualizar o display em um determinado tempo, para isso foi necessário adicionar via editor ‘nano’ um comando para executar o script a cada 5 minutos:

```
ubuntu@embrapa:~$ crontab -e
```

Na última linha adicionou-se o código:

```
*/5 * * * * /usr/bin/python3 /home/ubuntu/osr_ws/src/ihm_lcd.py
```

Dessa forma, o script foi executado de forma periódica para exibição de informações. Iterações futuras podem contemplar implementações mais sofisticadas com callbacks e fluxo de telas.

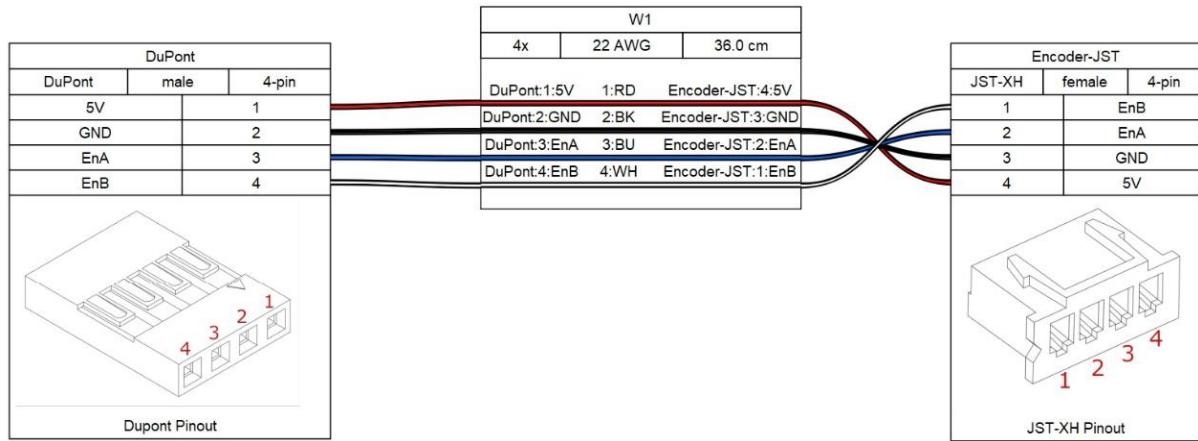
### Etapa de Cabeamento

O cabeamento necessário para alimentação, leitura de encoder e envio de sinais para os atuadores seguiu as instruções do repositório do GitHub em: open-source-rover/electrical/wiring. As dimensões sugeridas, no entanto, se apresentaram reduzidas, de forma que um excesso de pelo menos 10 cm foi adicionado em cada fio. O código de cores foi respeitado bem como a bitola dos fios de alimentação dos motores. Fios para sinais de controle foram substituídos por bitola de AWG duas unidas menor.

**Figura 22 – Exemplo de ligação entre conectores**

### Corner Encoder Extension Cable (x4)

#### Diagram



#### Bill of Materials

Id	Description	Qty	Unit	Designators
1	Cable, 4 x 22 AWG	36.0	cm	W1
2	Connector, DuPont, male, 4 pins	1		DuPont
3	Connector, JST-XH, female, 4 pins	1		Encoder-JST

**Fonte:** NASA. Open Source Rover. Disponível em: <https://github.com/nasa-jpl/open-source-rover>.

Acesso em: 27 fev. 2024.

## Mecânica

### Detalhamento Estrutural do Chassi e Suspensão

**Figura 23** - Estrutura do robô após montagem.



**Fonte:** elaborado pelos autores (2024).

A estrutura do robô é uma adaptação de veículos não tripulados que monitoram a superfície de marte e da lua. Ele possui uma suspensão do tipo Rocker-Bogie feita em perfil de alumínio, desenvolvido em 1996 para o rover Sojourner da Nasa e, mesmo transpassando por terrenos irregulares, seu chassi mantém-se nivelado a fim de minimizar a rolagem e arfagem. Ela é composta por dois eixos: o primeiro está fixo ao chassi e faz o braço maior (rocker) rotacionar. O segundo está na extremidade do anterior e faz o braço menor (bogie) rotacionar. Por eles atravessam os fios dos motores e encoders até a parte interna do chassi.

A dinâmica longitudinal de ultrapassagem de obstáculos (com as seis rodas motoras), sem a utilização de amortecedores, ocorre quando a roda dianteira é forçada contra o objeto pelas rodas central e traseira, então a suspensão rotaciona e a roda ultrapassa o objeto. Então a roda

central é puxada pela roda que ultrapassou e empurrada pela traseira, então a suspensão gira novamente. Por fim, a roda traseira é puxada pelas outras duas, rotaciona a suspensão e ultrapassa o obstáculo. Há uma barra perpendicular ao eixo fixo na porção superior do chassi, ela transmite a força de um lado da suspensão para o outro.

A dinâmica lateral é realizada por quatro eixos nas extremidades da suspensão, de forma que o rover pode realizar manobras de rotação (guinada) em espaços bem menores, como em torno do próprio eixo. Ângulos como camber, toe e caster apresentam grau nulo em deflexão estática, o que confere maior estabilidade na movimentação do veículo, maior área de contato das rodas com o solo e melhor manobrabilidade na faixa de velocidade que o rover se move.

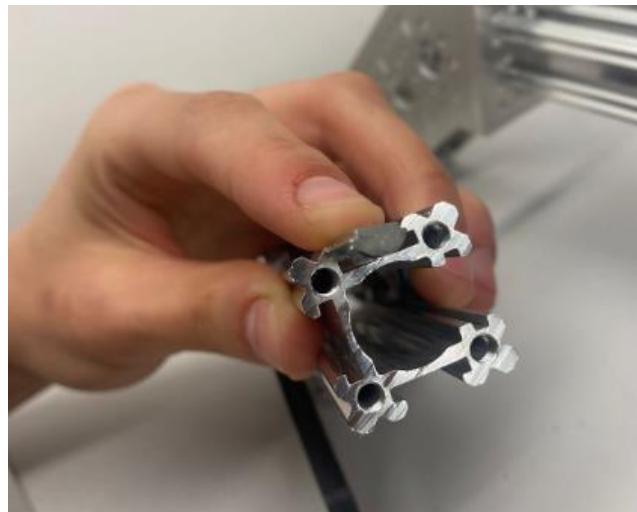
O chassi do robô é composto por barras de alumínio e chapas de acrílico onde estão guardadas as placas eletrônicas, baterias e IHM do dispositivo. Fixado a ela estão os suportes da câmera e do LiDAR. Dessa forma, a altura do centro de massa do robô fica acima da massa não-suspensa (rodas e suspensão), garantindo maior estabilidade ao robô. No entanto, o máximo de aclividade será limitado, em sua maioria, por essa altura, não permitindo a ele vencer ângulos superiores a 45°. Obstáculos que excedam resistência maior que o torque dos seis motores e possuam altura superior ao raio das rodas também não são vencidos.

## Adaptações Necessárias para Montagem

Ao longo da montagem da estrutura mecânica em alumínio, foi necessário realizar algumas adaptações, uma vez que os componentes eram importados e parte deles não estava disponível para a entrega. Para contornar esse problema, foram compradas peças equivalentes que permitissem adaptação para a aplicação proposta.

As primeiras peças a serem adaptadas foram os perfis de alumínio (figura 24) que compõem parte da suspensão do robô, uma vez que não foi possível encontrá-los nos comprimentos especificados na documentação oficial da Nasa, de 96mm. Como alternativa foram comprados perfis de tamanho superior, que foram cortados na dimensão correta especificada.

**Figura 24** – Perfil de alumínio usado para montagem da suspensão.



**Fonte:** elaborado pelos autores (2024).

Após o corte dos perfis, a rosca para alocação do parafuso M4 utilizado na montagem precisou ser refeita para todos os quatro furos de cada uma das peças cortadas. A operação de rosqueamento (figura 25) foi realizada manualmente e composta por três etapas, utilizando machos de tamanhos diferentes até chegar no tamanho desejado da rosca.

**Figura 25** – Operação de rosqueamento.



**Fonte:** elaborado pelos autores (2024).

As barras de alumínio (figura 26) utilizadas na montagem do corpo do robô também precisaram ser adaptadas, uma vez que também foram compradas em comprimento superior ao especificado e depois cortadas no tamanho correto, de 96 mm.

**Figura 26** – barras de alumínio para montagem do corpo.



**Fonte:** elaborado pelos autores (2024).

Após realização das operações de corte nas peças, foi necessário realizar a furação das barras, uma vez que o furo existente não era passante como no perfil. Além disso, também foi necessário realizar o rosqueamento em duas etapas, bem como ocorreu com a outra peça citada anteriormente.

Na operação de furação foram realizadas as seguintes etapas para garantir um melhor alinhamento do furo:

1. pintura da face a ser furada com um marcador.

2. Remoção da tinta para marcação do furo com auxílio de um marcador de altura vertical. Traçado realizado com dimensão equivalente à metade do lado da face, garantindo que as retas se cruzem no centro.
3. Realização de uma marcação central para garantir que a broca não seja inserida de forma desalinhada, etapa realizada com um equipamento de punção de centro, uma morsa e um martelo.
4. Furação com auxílio de uma fresadora manual de bancada, buscando alinhar a ponta da broca com a marcação realizada na etapa anterior. Avançar e recuar a broca durante o processo, garantindo escoamento do cavaco e dissipação do calor gerado.

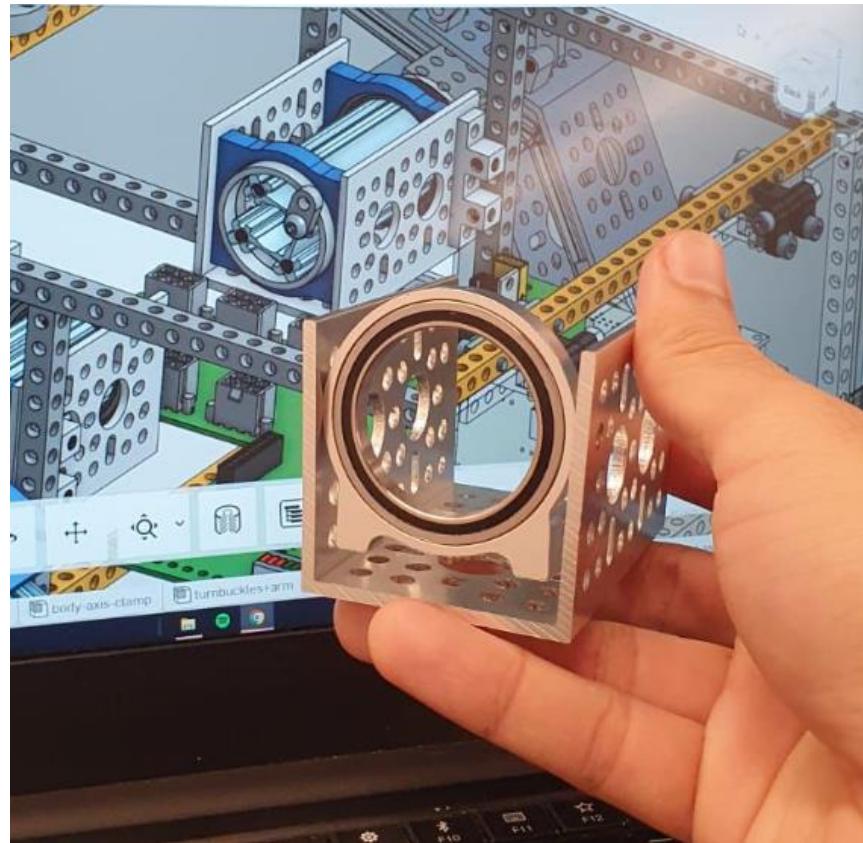
**Figura 27** – Marcação nas barras de alumínio para etapa de furação.



**Fonte:** elaborado pelos autores (2024).

A peça com o rolamento para encaixe da suspensão não estava disponível no modelo original da documentação da NASA, dessa forma foi adotada outra com diâmetro interno equivalente (Figura 28). Embora a substituta não encaixe perfeitamente e cause uma leve deformação estrutural, optou-se por não realizar nenhuma adaptação pois os componentes poderiam ser danificados e não possuíam substitutos. O efeito de deformação resultante não comprometeu o funcionamento do rover.

**Figura 28** – Representação da peça substituída e de sua substituta.

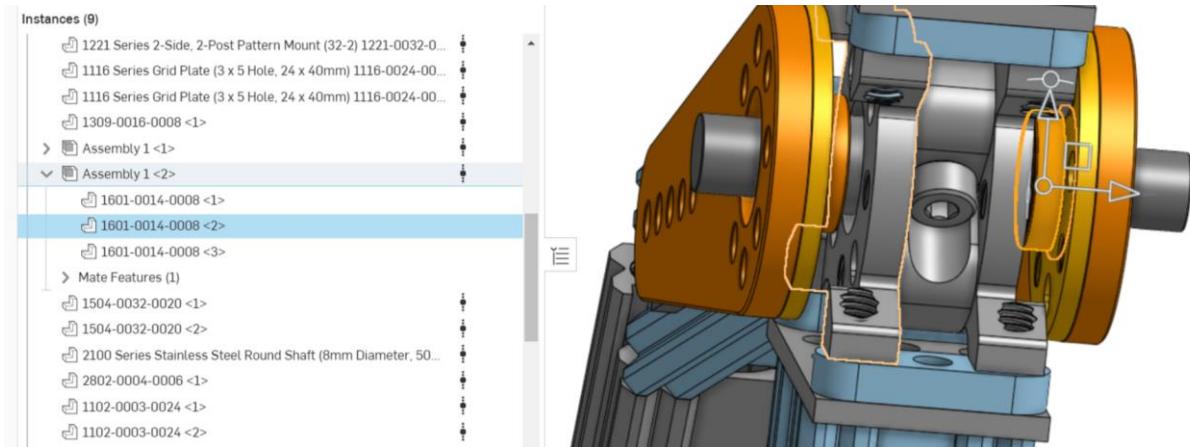


**Fonte:** elaborado pelos autores (2024).

#### Revisão de Erros Encontrados na Documentação do Oficial do GitHub

Um dos componentes da suspensão foi identificado erroneamente na documentação oficial, com código inicial 1601, que na realidade corresponde ao componente número 1611. Na visualização 3D da montagem, a peça identificada com código correspondente ao informado na documentação oficial apresenta diâmetro interno com valor inferior a 8mm, não montando no eixo especificado.

**Figura 29** – Peça identificada incorretamente.



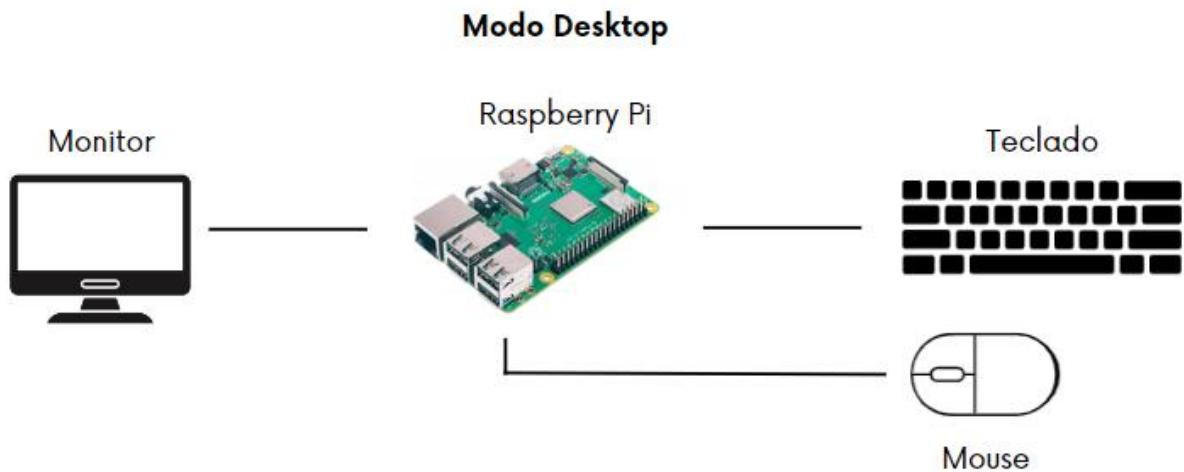
**Fonte:** elaborado pelos autores (2024).

## Programação

### Configuração da Raspberry Pi como Servidor e Comunicação SSH

Existem duas possibilidades para a instalação do Ubuntu na Raspberry Pi: o desktop (Figura 30) e o server (Figura 31). Na primeira opção, o sistema operacional apresenta um ambiente equipado com uma interface gráfica e outras aplicações que possibilitam o acesso por parte do usuário de forma mais visual e intuitiva, uma vez que a Raspberry Pi passa a atuar como um computador com a conexão de um mouse, um teclado e de um display. Na segunda opção, na configuração de servidor, a manipulação pelo usuário é menos intuitiva. Considerando a ausência da interface gráfica, existe a necessidade de configuração de um acesso remoto, permitindo que o servidor seja acessado por outros computadores.

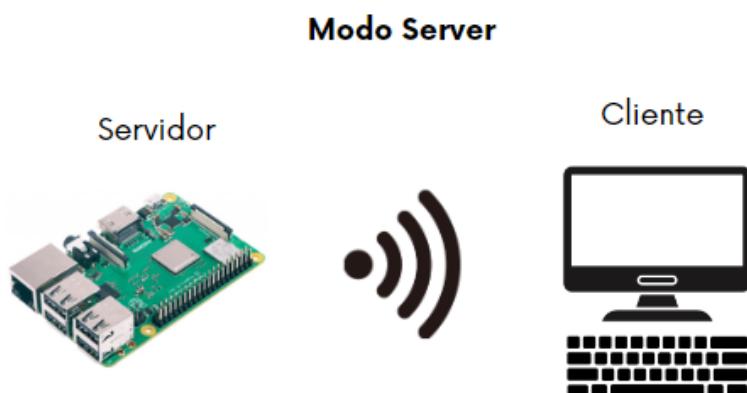
**Figura 30** – Funcionamento modo desktop.



**Fonte:** elaborado pelos autores (2024).

Como citado anteriormente na seção de Análise dos dados coletados, ocorreu um processo de transição ao longo do semestre entre as duas opções apresentadas. Inicialmente optou-se pelo modo desktop por sua maior intuitividade e facilidade na aplicação de comandos prontos do GitHub, uma vez que não era necessário digitar os códigos manualmente no terminal e as instruções do repositório podiam ser facilmente visualizadas usando apenas a Raspberry Pi.

**Figura 31** – Funcionamento modo servidor.



**Fonte:** elaborado pelos autores (2024).

Após inicialização da etapa de captação das imagens fornecidas pela câmera, o desktop deixou de se mostrar como uma opção vantajosa, dada a presença de um delay excessivo na atualização das imagens fornecidas, bem como uma velocidade de processamento significantemente reduzida. O problema se apresentou principalmente durante a utilização do Rviz2, uma interface gráfica do ROS2 utilizada para visualização de diversos tópicos, com plugins que possibilitam o monitoramento das imagens.

Com mudança para servidor, o problema de processamento das imagens deixou de ocorrer uma vez que passou a ser utilizada a interface gráfica do próprio computador para efetuar o processamento. Para implementação do servidor, o processo de conexão ocorreu com base no protocolo de comunicação SSH (Secure Socket Shell), via rede.

Segundo a SSH Academy (2023), o protocolo SSH (Secure Socket Shell) funciona em um modelo cliente-servidor, no qual o cliente (computador) inicializa a conexão, utilizando uma chave pública para contatar o servidor (Raspberry Pi), que após abertura de um canal seguro permite o acesso ao seu sistema operacional. Um diagrama explicativo do funcionamento do protocolo está disponível na figura 19.

**Figura 32 – Diagrama de Funcionamento Protocolo SSH**



**Fonte:** SSH Academy (2023). Disponível em: <https://www.ssh.com/academy/ssh/protocol>. Acesso em: 10 mai. 2024.

Em decorrência da transição para o servidor, a conexão do robô por protocolo SSH passou a ocorrer por meio de uma rede pública, disponível automaticamente após alimentação da Raspberry Pi. Para garantir maior segurança, uma senha de acesso foi incluída.

Após conexão do usuário na rede do robô utilizando a chave correta, é necessária a inicialização da comunicação SSH no terminal, através do nome de usuário e do endereço de IP da Raspberry Pi. Após finalização dessa etapa, todos os tópicos do robô passam a estar disponíveis para o cliente e podem ser acessados e editados remotamente pelo computador.

Para efetuar a conexão com a raspberry no modelo servidor, seguir as etapas:

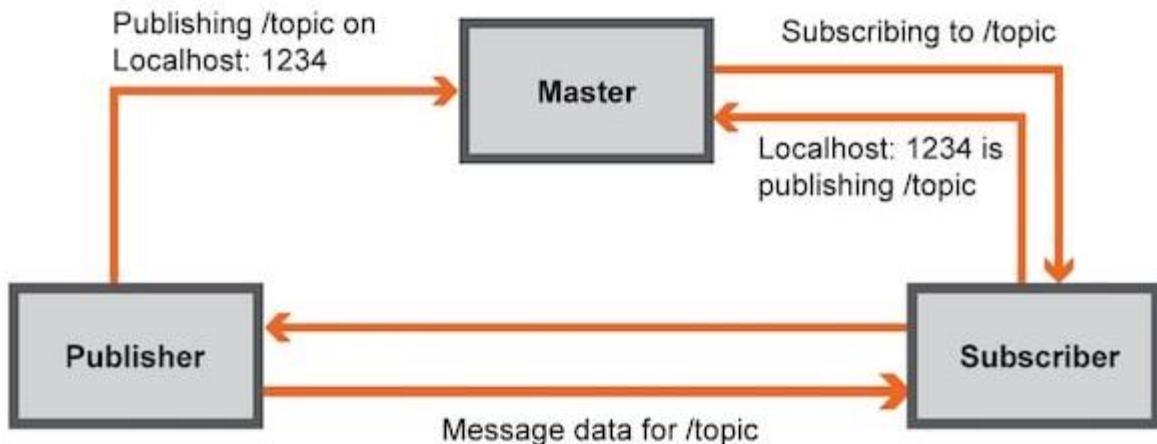
1. Alimentar a raspberry pi
2. Acessar a rede “Embrapa”
3. Colocar a senha da rede
4. Acessar prompt de comando e requisitar a comunicação SSH com o comando “ssh embrapa@10.42.0.1”
5. Fornecer senha para iniciar a comunicação

## Framework

Para programar o UGV foi utilizado um framework ROS (Robot Operating System), ele é baseado na dinâmica de nós. Estes são, em sua essência, arquivos executáveis dentro do pacote ROS, que se comunicam através de tópicos. O nó pode ser um publisher ou um subscriber. Um publisher é um nó que precisa publicar as mensagens, enquanto um subscriber é um nó que é privilegiado para acessar as mensagens.

Para gerenciar esse ambiente há um Mestre no ROS ele é uma entidade centralizada que desempenha um papel crucial na coordenação da comunicação entre os nós, sendo responsável pelo registro e busca de nomes para o restante do sistema. Sem o Mestre, os nós não seriam capazes de encontrar uns aos outros ou trocar mensagens.

**Figura 33 – Funcionamento do ROS.**



**Fonte:** <https://www.allaboutcircuits.com/technical-articles/an-introduction-to-robot-operating-system-ros/>  
Acesso em 18 mar. 2024.

No caso do projeto o ROS Humble foi instalado na Raspberry Pi. O sistema operacional escolhido para executar o framework e acessar os arquivos foi o Ubuntu. Foi criado um nó para ler os dados recebidos pelo controle e publicar a informação no tópico do controle. Os outros dois nós que controlam os motores de direção (servo motor) e sentido (motores DC) vão receber a informação do tópico do controle e executar a tarefa necessária.

### Configuração da Raspberry Pi

#### Instalando um Sistema Operacional

A configuração da Raspberry Pi começou com a instalação do sistema operacional Ubuntu, escolhido por sua compatibilidade com o ROS e seu suporte a diversos pacotes de software necessários para o desenvolvimento do rover. Primeiro, a imagem do sistema operacional foi baixada do site oficial do Ubuntu Server. Em seguida, um software como o Raspberry Pi Imager foi utilizado para gravar essa imagem em um cartão SD. Este passo é crucial, pois a imagem do sistema operacional contém todos os arquivos necessários para que a Raspberry Pi funcione corretamente. Após a gravação, o cartão SD foi inserido na Raspberry

Pi, que foi ligada e inicializada. Durante a configuração da imagem, foi habilitado o SSH, e configurado o usuário e senha, a rede Wi-Fi.

Para facilitar o acesso, é atribuído um endereço IP estático à Raspberry Pi. A partir de outro computador, é possível conectar com a Raspberry Pi usando SSH com o comando ssh `ubuntu@embrapa` ou `ssh embrapa@10.42.0.1`, onde é solicitada uma senha para estabelecer a conexão com uma janela de terminal na Raspberry Pi.

### Instalando o ROS

Durante a inicialização, foram seguidas as instruções na tela para configurar a rede Wi-Fi e outras preferências básicas. Estas configurações iniciais garantem que a Raspberry Pi possa se conectar à internet e receber atualizações de software. A seguir, o sistema operacional foi atualizado utilizando os comandos:

```
ubuntu@embrapa:~$ sudo apt update  
ubuntu@embrapa:~$ sudo apt upgrade
```

Este passo é importante para garantir que todos os pacotes de software estejam na versão mais recente e tenham as últimas correções de segurança e melhorias de desempenho.

Com o sistema operacional atualizado, foi iniciada a instalação do ROS Humble. A instalação do ROS Humble é feita seguindo vários passos descritos na documentação oficial. Primeiramente, a chave do repositório ROS foi adicionada ao sistema com o comando:

```
ubuntu@embrapa:~$ sudo apt install software-properties-common  
ubuntu@embrapa:~$ sudo add-apt-repository universe  
ubuntu@embrapa:~$ sudo apt update  
ubuntu@embrapa:~$ sudo apt install curl  
ubuntu@embrapa:~$ curl -sSL  
https://raw.githubusercontent.com/ros/rosdistro/master/ros.asc | sudo apt-key add -
```

Esta chave garante que os pacotes baixados do repositório ROS sejam autênticos e seguros. Em seguida, o repositório ROS foi adicionado à lista de fontes do sistema utilizando o comando:

```
ubuntu@embrapa:~$ sudo sh -c 'echo "deb http://packages.ros.org/ros2/ubuntu
$(lsb_release -cs) main" > /etc/apt/sources.list.d/ros2-latest.list'
```

Com o repositório adicionado, as listas de pacotes foram atualizadas com:

```
ubuntu@embrapa:~$ sudo apt update
```

Para instalar o ROS Humble, foi utilizado o comando:

```
ubuntu@embrapa:~$ sudo apt install ros-humble-desktop
```

Após a instalação, é necessário configurar o ambiente para utilizar o ROS. Isto é feito adicionando o ROS ao ambiente de sistema utilizando o comando:

```
ubuntu@embrapa:~$ source /opt/ros/humble/setup.bash
```

Este comando carrega as configurações do ROS, permitindo que seus comandos e ferramentas sejam utilizados. Para que essa configuração seja carregada automaticamente em novas sessões de terminal, a linha:

```
ubuntu@embrapa:~$ echo "source /opt/ros/humble/setup.bash" >> ~/.bashrc
```

foi adicionada ao arquivo “.bashrc”.

Além do ROS, foram instaladas dependências adicionais necessárias para o projeto do rover, utilizando os comandos:

```
ubuntu@embrapa:~$ sudo apt install python3-colcon-common-extensions
```

```
ubuntu@embrapa:~$ sudo apt install python3-argcomplete
```

Estas ferramentas são essenciais para a construção e gerenciamento de pacotes ROS. Com o ROS instalado o processo de setup, que envolve a configuração e inicialização dos nós ROS que compõem o sistema do rover.

## Configuração do Ambiente ROS e Construção do Código do Rover

Com o ROS instalado, prossegue-se para a criação de um workspace ROS para o código do rover. Cria-se o diretório do workspace e navega-se até ele com:

```
ubuntu@embrapa:~$ mkdir -p ~/osr_ws/src && cd ~/osr_ws
```

Este comando cria um diretório para o workspace do ROS e navega até ele. Depois é preciso carregar as configurações do ROS para esse novo workspace. Isso é feito usando comando:

```
ubuntu@embrapa:~$ source /opt/ros/humble/setup.bash
```

Em seguida, é necessário clonar o repositório do código do rover. Primeiramente foi instalado o Git, que é uma ferramenta de controle de versão usada para clonar repositórios, na Raspberry Pi por meio do comando:

```
ubuntu@embrapa:~$ sudo apt install git
```

Em seguida, clona-se o repositório do código do rover do GitHub para o diretório “src”:

```
ubuntu@embrapa:~$ cd ~/osr_ws/src
```

```
ubuntu@embrapa:~$ git clone https://github.com/nasa-jpl/osr-rover-code.git
```

É preciso baixar também o rosdep, que é uma ferramenta que instala automaticamente as dependências de pacotes ROS, e depois inicializá-lo no sistema e atualizar sua base de dados.

```
ubuntu@embrapa:~$ sudo apt install python3-rosdep
```

```
ubuntu@embrapa:~$ sudo rosdep init
```

```
ubuntu@embrapa:~$ rosdep update
```

Em seguida é instalado todas as dependências dos pacotes ROS no diretório src, assim como, bibliotecas Python adicionais necessárias para o projeto do rover.

```
ubuntu@embrapa:~$ rosdep install --from-paths src --ignore-src --rosdistro=humble -y
```

```
ubuntu@embrapa:~$ pip3 install adafruit-circuitpython-servokit smbus ina260
```

Em seguida, constroem-se os pacotes ROS:

```
ubuntu@embrapa:~$ colcon build --symlink-install
```

O colcon é uma ferramenta de construção para ROS 2 que compila os pacotes no workspace e cria links simbólicos para facilitar o desenvolvimento. Adicionam-se os arquivos gerados ao caminho para que o ROS possa encontrá-los.

```
ubuntu@embrapa:~$ source install/setup.bash
```

Este comando carrega as configurações do ROS no ambiente atual, incluindo os pacotes recém-compilados.

### Configurando a Comunicação Serial na Raspberry Pi

Para garantir a comunicação adequada entre a Raspberry Pi e os controladores de motor Roboclaw, configura-se a interface serial. Habilita-se a comunicação Serial e I2C utilizando o comando:

```
ubuntu@embrapa:~$ sudo raspi-config
```

O raspi-config é uma ferramenta de configuração para a Raspberry Pi, usada para habilitar interfaces como Serial e I2C. No menu de configuração, habilita-se as interfaces I2C e Serial. Caso o raspi-config não esteja instalado, utiliza-se os comandos para instalá-lo:

```
ubuntu@embrapa:~$ sudo apt-get install raspi-config
```

Este comando instala o raspi-config no sistema.

É necessário desabilitar serviço serial-getty@ttyS0.service para evitar conflitos entre os dispositivos seriais. Também é preciso impedir que esse serviço seja iniciado automaticamente, para isso usa-se os seguintes comandos:

```
ubuntu@embrapa:~$ sudo systemctl stop serial-getty@ttyS0.service
```

```
ubuntu@embrapa:~$ sudo systemctl disable serial-getty@ttyS0.service
```

```
ubuntu@embrapa:~$ sudo systemctl mask serial-getty@ttyS0.service
```

Este último comando máscara o serviço serial-getty@ttyS0, prevenindo que ele seja iniciado manualmente.

### Copiando as Regras do udev

Para configurar automaticamente os dispositivos necessários, copia-se as regras do udev. Estas regras definem como o sistema deve configurar dispositivos de hardware quando eles são conectados. É copiado o arquivo de regras do repositório do rover para o sistema com o comando:

```
ubuntu@embrapa:~$ cd ~/osr_ws/src/osr-rover-code/config
ubuntu@embrapa:~$ sudo cp serial_udev_ubuntu.rules /etc/udev/rules.d/10-local.rule
```

O primeiro comando navega até o diretório de configuração do código do rover, enquanto o segundo comando copia o arquivo de regras do udev para o diretório /etc/udev/rules.d/. Este arquivo configura as regras necessárias para a comunicação serial com os controladores Roboclaw. É preciso então recarregar as regras do udev para que as novas configurações entrem em vigor:

```
ubuntu@embrapa:~$ sudo udevadm control --reload-rules && sudo udevadm trigger
```

O primeiro comando recarrega as regras do udev, enquanto o segundo comando, ativa as regras recarregadas, garantindo que os dispositivos sejam configurados corretamente.

### Adicionando o Usuário aos Grupos tty e dialout

Para garantir que o usuário tenha as permissões necessárias para acessar dispositivos seriais, adiciona-se o usuário aos grupos tty e dialout. Esses grupos controlam o acesso aos dispositivos seriais no Linux. O primeiro comando adiciona o usuário atual ao grupo tty, já o segundo adiciona o usuário atual ao grupo dialout:

```
ubuntu@embrapa:~$ sudo adduser $USER tty
```

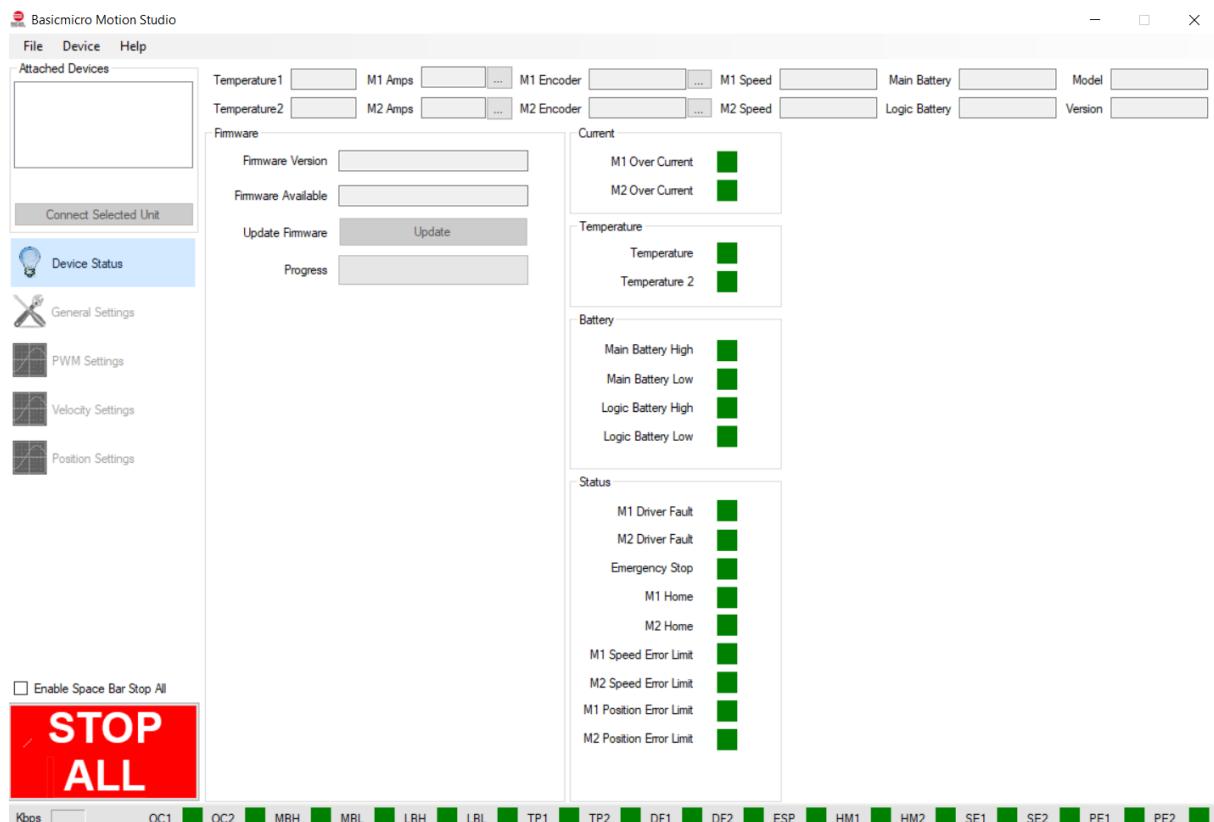
```
ubuntu@embrapa:~$ sudo adduser $USER dialout
```

Após adicionar o usuário aos grupos, é necessário reiniciar a sessão SSH ou reiniciar a Raspberry Pi para que as mudanças tenham efeito.

### Configurando os Drives da Roboclaw

Para configurar os drives da Roboclaw, utiliza-se o software disponível no site da BasicMicro. O software BasicMicro Motion Studio permite ajustar diversas configurações dos controladores Roboclaw, como endereços, taxas de comunicação serial e modos de operação. Inicialmente, baixa-se e instala-se o software no computador. Em seguida, conecta-se o controlador Roboclaw ao computador via USB.

**Figura 34–** Interface do Basicmicro Studio.



**Fonte:** elaborado pelos autores (2024).

No BasicMicro Motion Studio, seleciona-se a porta COM correspondente ao controlador Roboclaw e clica-se em "Connect". Uma vez conectado, ajusta-se o endereço de cada controlador para valores únicos (por exemplo, 128, 129, 130). Configura-se a taxa de comunicação serial para 115200 bits/s e habilita-se o "Multi-Unit Mode" em todos os controladores, garantindo que possam operar corretamente no mesmo barramento serial.

Após configurar todos os controladores, salva-se as configurações clicando no Device, localizado no canto superior esquerdo e depois em Write Settings e depois desconecta-se o controlador Roboclaw do computador. Em seguida, reconecta-se ao sistema do rover para teste e operação.

### Testando a Comunicação Serial com os Controladores de Motor Roboclaw

Para verificar se a comunicação serial com os controladores de motor Roboclaw está funcionando corretamente, utiliza-se um script de teste fornecido no repositório do rover. Navega-se até o diretório de scripts e executa-se o script roboclawtest.py para cada endereço de motor (endereços são 128, 129 e 130). Os comandos são:

```
ubuntu@embrapa:~$ cd ~/osr_ws/src/osr-rover-code/scripts
ubuntu@embrapa:~$ python3 roboclawtest.py 128
ubuntu@embrapa:~$ python3 roboclawtest.py 129
ubuntu@embrapa:~$ python3 roboclawtest.py 130
```

Os scripts devem retornar informações sobre o controlador de motor, como a versão do firmware e o estado atual. Cada comando deve retornar algo como:

```
(1, 'USB Roboclaw 2x7a v4.1.34\n')
(1, 853, 130)
```

Se o script parecer travar ou retornar apenas zeros dentro dos parênteses, pode haver um problema de comunicação com o roboclaw para aquele endereço específico. Alguns passos de solução de problemas incluem:

1. Verificar se as instruções na seção de configuração da comunicação serial foram seguidas corretamente e se os dispositivos seriais estão configurados corretamente na Raspberry Pi.
2. Certificar-se de que todos os controladores Roboclaw foram configurados corretamente, incluindo o endereço, taxa de comunicação serial e a opção "Enable Multi-Unit Mode".
3. Desconectar todos os controladores de motor, exceto um, e depurar exclusivamente com ele.

Seguindo esses passos, o sistema do rover foi configurado e os nós ROS ficaram prontos para serem utilizados, permitindo a comunicação entre os diferentes componentes do UGV e a execução das tarefas necessárias para seu funcionamento.

#### Bringup do Rover: Configuração Detalhada

Para a inicialização do rover utilizando o código fornecido pelo repositório da NASA JPL, é necessário seguir uma série de etapas detalhadas. Os próximos tópicos descrevem, minuciosamente, os procedimentos para a configuração e bringup do rover, conforme especificado na documentação oficial disponível no repositório. Essas etapas são fundamentais para garantir que todos os componentes do rover estejam operando de maneira integrada e eficiente, proporcionando um desempenho adequado às necessidades do projeto.

#### Configurando os Parâmetros do Rover

Primeiramente, é necessário configurar os parâmetros específicos do rover no arquivo params.yaml. Este arquivo deve ser ajustado para refletir as características físicas e os detalhes de hardware do rover, como a base das rodas, a largura entre as rodas, a velocidade máxima, o ângulo de direção máximo, a porta serial e a taxa de baud.

```
rover:
    ros__parameters:
        rover_dimensions:
            d1: 0.177 # [m]
```

```

d2: 0.310 # [m]

d3: 0.274 # [m]

d4: 0.253 # [m]

wheel_radius: 0.075 # [m], needs to be a float

drive_no_load_rpm: 223.0 # no load speed for the drive motors.

NOTE: needs to be a float value

```

O arquivo params.yaml está localizado no diretório de configuração do código do rover.

### Calibração dos Servos de Canto

A calibração dos servos de canto é fundamental para garantir que os ângulos de direção sejam precisos. O processo de calibração envolve o ajuste dos valores de ângulo mínimo e máximo dos servos, bem como a verificação de sua resposta aos comandos.

Primeiramente, envia-se os motores dos servos para suas posições zero. Como o intervalo de movimento dos servos é de 0 a 300, a posição padrão deve ser no meio desse intervalo, em 150. Executam-se os seguintes comandos:

```

ubuntu@embrapa:~$ cd ~/osr_ws/src/osr-rover-code/scripts

ubuntu@embrapa:~$ python3 calibrate_servos.py 0 150

ubuntu@embrapa:~$ python3 calibrate_servos.py 1 150

ubuntu@embrapa:~$ python3 calibrate_servos.py 2 150

ubuntu@embrapa:~$ python3 calibrate_servos.py 3 150

```

Com cada linha, espera-se que o servo motor traseiro direito (0), depois o dianteiro direito (1), seguido pelo dianteiro esquerdo (2) e finalmente o traseiro esquerdo (3) movam-se para um ângulo de 150. Se o servo não se mover, isso pode significar que ele já está nesse ângulo ou que há um problema com a fiação, PCB ou conexão com o PCA9685.

Com cada motor montado, pode-se prosseguir com o alinhamento de cada conjunto de canto para que as rodas estejam perfeitamente retas. Faz-se isso enviando cada conjunto de canto para diferentes ângulos até que visualmente pareça correto. Utiliza-se o mesmo script para isso. Para um motor de cada vez, executa-se o script com um valor ligeiramente diferente. Por exemplo, move-se o motor traseiro direito para 160 graus:

```
ubuntu@embrapa:~$ python3 calibrate_servos.py 0 160
```

Se ele se mover mais para longe do centro, tente 140; se estiver se aproximando, tente 165 ou 170. Repete-se isso até que a roda esteja exatamente no zero. Quando encontrar o valor onde está centralizado, anote-o, pois será o valor inicial do servo.

Repete-se esse procedimento para os motores nos canais 1, 2 e 3. Ao final, deve-se ter quatro valores, por exemplo, [135, 155, 125, 162]. Em seguida, é necessário informar ao software quais são esses valores para que, ao comandar os motores dos cantos para girar, eles utilizem esses valores como os centros verdadeiros. Abre-se o arquivo osr\_mod\_launch.py com o seguinte comando:

```
ubuntu@embrapa:~$nano~/osr_ws/src/osr-rover_code/ROS/osr Bringup/launch/  
osr_mod_launch.py
```

Procura-se a linha que diz parameters= [ {'centered\_pulse\_widths': [165, 134, 135, 160]} ]. Substituem-se os valores usando os 4 valores encontrados. Após salvar essas alterações no arquivo os motores dos servos dos cantos estão calibrados e prontos para uso.

## Mapeando Botões e Eixos do Controle Remoto para o Movimento do Rover

Cada controle remoto possui diferentes configurações de eixos e botões, tornando essencial a personalização para que cada função do rover seja devidamente mapeada. É necessário definir a velocidade máxima do rover, qual eixo controla o movimento para frente e para trás, qual eixo controla o movimento para a esquerda e para a direita, e qual eixo permite a rotação no próprio eixo. É necessário essas mudanças no arquivo “osr-rover-code/ROS/osr Bringup/launch/osr\_mod\_launch.py”. Os valores no nó joy\_to\_twist que são de interesse incluem:

```

{"scale_linear": X.XX}, # escala aplicada à velocidade linear, em m/s:
drive_motor_rpm * 2pi / 60 * raio da roda * fator de redução

{"scale_angular": X.XX}, # escala aplicada à velocidade angular, em rad/s:
scale_linear / min_radius

{"scale_linear_turbo": X.XX}, # escala aplicada à velocidade linear em
modo turbo, em m/s

{"enable_button": X}, # botão que habilita o movimento

{"axis_linear.x": X}, # eixo do controle remoto para movimento linear

{"axis_angular.yaw": X}, # eixo do controle remoto para rotação

{"axis_angular.pitch": X}, # eixo do controle remoto para rotação no
próprio eixo

```

Define-se inicialmente os valores de escala (primeiros três) baixos (por exemplo, 0.05) para confirmar facilmente se o rover se move conforme esperado, evitando possíveis acidentes. Nos próximos passos, após confirmar que tudo está funcionando corretamente, ajustam-se esses valores de escala para configurações mais apropriadas.

Dependendo do modelo e fabricante do controle remoto, os botões e eixos (joysticks) podem ser representados de forma diferente. Para isso, roda-se o nó joy do ROS 2 e verifica-se a funcionalidade dos controles. Certifica-se de que o joystick está conectado e, em seguida, executa-se no terminal:

```
ubuntu@embrapa:~$ ros2 run joy joy_node
```

Esse comando confirma a conexão com o gamepad. Para verificar, abre-se um novo terminal separado e escuta-se por mensagens provenientes do joystick pelo tópico:

```
ubuntu@embrapa:~$ ros2 topic echo /joy
```

Deve-se ver uma sequência constante de mensagens sendo impressas. Movendo um eixo ou pressionando um botão no controle, observa-se uma mudança na saída. Na janela onde as mensagens do comando ros2 topic echo estão sendo impressas, anota-se qual índice na lista

de eixos/botões muda ao movimentar o eixo e abre o arquivo osr\_mod\_launch.py, para mudar os parâmetros do joy.

Para configurar o controle do movimento para frente, move-se o eixo correspondente e observa-se qual número na lista de eixos do terminal muda. A contagem começa em zero, então, se o eixo que muda for o terceiro da lista, o número do eixo é 2 (3-1=2). Insere-se esse número (neste caso 2) após axis\_linear.x.

Realiza-se o mesmo procedimento para os eixos que serão utilizados para girar e para a rotação no próprio eixo, utilizando axis\_angular.yaw para girar e axis\_angular.pitch para rotação no próprio eixo. Para o botão que será utilizado para habilitar qualquer movimento, anota-se o número do botão (novamente começando a contagem do zero) e insere-se esse número em enable\_button.

O controle remoto que foi disponibilizado foi o Spektrum DXS, ao fazer esses passos para mapear o controle e determinar quais eixos e botões são os mais intuitivos os parâmetros dos joy no osr\_mod\_launch.py ficaram assim configurados:

```
parameters = [
    {"scale_linear.x": 0.12},
    {"axis_linear.x": 1},
    {"axis_angular.yaw": 2},
    {"axis_angular.pitch": 4
    {"scale_angular.yaw": 1.25},
    {"scale_angular.pitch": 0.25},
    {"scale_angular_turbo.yaw": 0.0},
    {"scale_linear_turbo.x": 0.0},
    {"enable_button": 0},
    {"enable_turbo_button": 1}
```

]

## Implementação de uma Rotina de Emergência

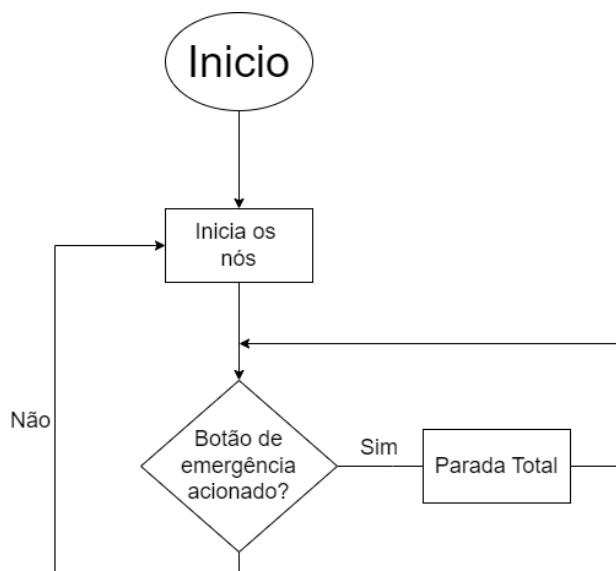
A criação de uma rotina de emergência no parâmetro joy foi essencial para garantir a segurança operacional do rover. Esta rotina tem como objetivo parar qualquer tipo de movimentação no rover em situações críticas, proporcionando uma camada adicional de segurança tanto para o operador quanto para o ambiente ao redor.

Desenvolveu-se um botão que, ao ser acionado no controle, sobrescreve todos os valores e zera todas as ações do sistema. Para isso, utilizou-se os parâmetros enable\_button\_turbo e scale\_linear\_turbo.

Quando enable\_button\_turbo é acionado (enable\_button\_turbo = 1), o valor de scale\_linear\_turbo é ajustado para 0.0, o que resulta na parada completa do rover. Essa escolha foi feita para garantir uma resposta imediata e segura em emergências. A vantagem de usar scale\_linear\_turbo com valor zero é que ele proporciona uma maneira simples e eficiente de anular qualquer comando de movimento, parando o rover de forma rápida e segura.

O fluxograma abaixo ilustra o funcionamento da rotina de emergência.

**Figura 35** – Fluxograma de funcionamento da rotina de emergência.



**Fonte:** elaborado pelos autores (2024).

No fluxograma, inicia-se o processo iniciando os nós do sistema. Em seguida, verifica-se continuamente se o botão de emergência foi acionado. Se o botão for acionado, o sistema executa uma parada total, interrompendo todos os movimentos do rover. Caso contrário, o sistema continua monitorando o estado do botão de emergência.

Essa implementação garante que o rover possa ser parado imediatamente em caso de emergência, proporcionando uma camada adicional de segurança durante a operação. Utilizar o botão de turbo (enable\_button\_turbo) para acionar a parada de emergência foi uma escolha estratégica, pois esse botão é facilmente acessível e sua função original pode ser modificada sem impactar negativamente outras operações do rover.

### Bringup Manual do Rover

Para iniciar manualmente o rover, é necessário executar uma série de comandos no terminal. Este processo envolve a inicialização dos nós do ROS 2 necessários para o funcionamento do rover. Abaixo estão os passos detalhados para realizar o bringup manual:

Primeiramente, navega-se até o diretório de trabalho do ROS, que contém o workspace do rover:

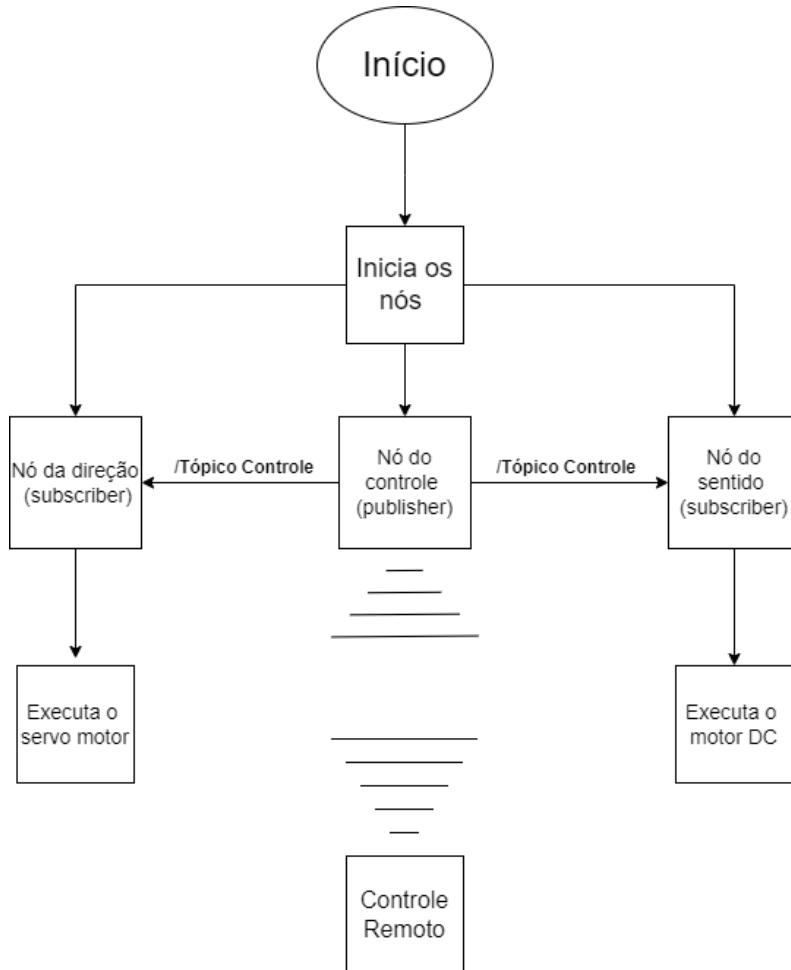
```
ubuntu@embrapa:~$ cd ~/osr_ws
```

Este comando muda o diretório atual para o workspace do ROS, onde o código do rover está localizado. Em seguida, executa-se o comando de bringup, que inicializa todos os nós necessários para operar o rover:

```
ubuntu@embrapa:~$ ros2 launch osr Bringup osr.launch.py
```

Este comando lança o arquivo osr.launch.py, que contém as instruções para inicializar os diferentes nós do sistema ROS 2. O arquivo de launch é configurado para iniciar todos os nós essenciais para a operação do rover, incluindo aqueles responsáveis pelo controle dos motores, leitura de sensores e comunicação com o controle remoto.

**Figura 36** – Fluxograma de funcionamento do UGV.



**Fonte:** elaborado pelos autores (2024).

Durante este processo, verifica-se no terminal se todos os nós foram inicializados corretamente. Mensagens de erro ou avisos são analisados e resolvidos conforme necessário. Este procedimento manual permite uma maior flexibilidade para testes e ajustes finos antes de automatizar o processo de bringup.

#### Bringup Automático com Script de Lançamento

Para eliminar a necessidade de acesso via SSH para executar o rover, é necessário configurar o Raspberry Pi (RPi) para iniciar automaticamente o código do rover ao ser ligado. Durante a inicialização, quaisquer parâmetros modificados em osr\_mod\_launch.py e na pasta config serão aplicados automaticamente.

Iniciar scripts automaticamente no boot utilizando ROS apresenta dificuldades adicionais em comparação a scripts normais devido às configurações de permissão padrão no RPi e à impossibilidade de executar ROS como usuário root. A estratégia utilizada será criar um serviço que inicia nosso script roslaunch, configurando-o para executar automaticamente no boot do robô.

Existem dois scripts na pasta init\_scripts. O primeiro é o arquivo bash que executa o arquivo de lançamento do ROS, e o segundo cria um serviço do sistema para iniciar esse script bash. Para configurar isso, seguem os passos abaixo:

Navegar até a pasta init\_scripts:

```
ubuntu@embrapa:~$ cd ~/osr_ws/src/osr-rover-code/init_scripts
```

Criar links simbólicos para capturar atualizações nesses arquivos no serviço:

```
ubuntu@embrapa:~$ sudo ln -s $(pwd)/launch_osr.sh/usr/local/bin/
launch_osr.sh
```

```
ubuntu@embrapa:~$ sudo ln -s $(pwd)/osr_paths.sh /usr/local/bin/osr_paths.sh
```

Copiar o arquivo de serviço para o diretório de serviços do systemd:

```
ubuntu@embrapa:~$ sudo cp osr_startup.service/etc/systemd/system/
osr_startup.service
```

Ajustar as permissões do arquivo de serviço:

```
ubuntu@embrapa:~$ sudo chmod 644 /etc/systemd/system/osr_startup.service
```

Instalar o serviço de inicialização do OSR no Pi e deixá-lo pronto para uso. A tabela abaixo apresenta alguns comandos úteis para gerenciar esse serviço:

**Tabela 5** – Comandos para inicialização.

Descrição	Comando
Iniciar serviço	<code>sudo systemctl start osr_startup.service</code>
Parar serviço	<code>sudo systemctl stop osr_startup.service</code>
Habilitar serviço (executa na inicialização do RPi)	<code>sudo systemctl enable osr_startup.service</code>
Desabilitar serviço (não executa na inicialização do RPi)	<code>sudo systemctl disable osr_startup.service</code>
Verificar status do serviço	<code>sudo systemctl status osr_startup.service</code>
Ver lista de serviços ao vivo	<code>sudo journalctl -f</code>

**Fonte:** elaborado pelos autores (2024).

Feito os testes e verificado que o rover está funcionando corretamente foi habilitado o serviço de inicialização no robô com o comando abaixo:

```
ubuntu@embrapa:~$ sudo systemctl enable osr_startup.service
```

### Aquisição de Imagens de Monitoramento

Para realização do monitoramento, do ponto de vista da captação de imagens, foi necessário realizar alguns passos de setup antes de efetuar a conexão da câmera com a Raspberry Pi.

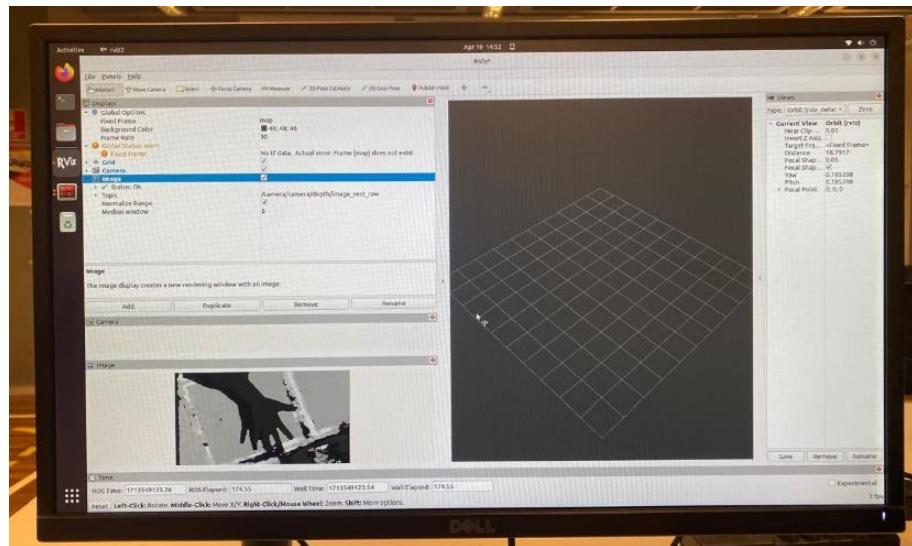
A primeira etapa desse processo consiste em incluir a biblioteca da RealSense no ROS2, para o processo de instalação, foi utilizada a biblioteca oficial da câmera no GitHub<sup>1</sup>, fornecida pela fabricante Intel.

---

<sup>1</sup> Fonte: INTEL. 2024. Disponível em: <https://github.com/IntelRealSense/realsense-ros>. Acesso em: 11 mai. 2024.

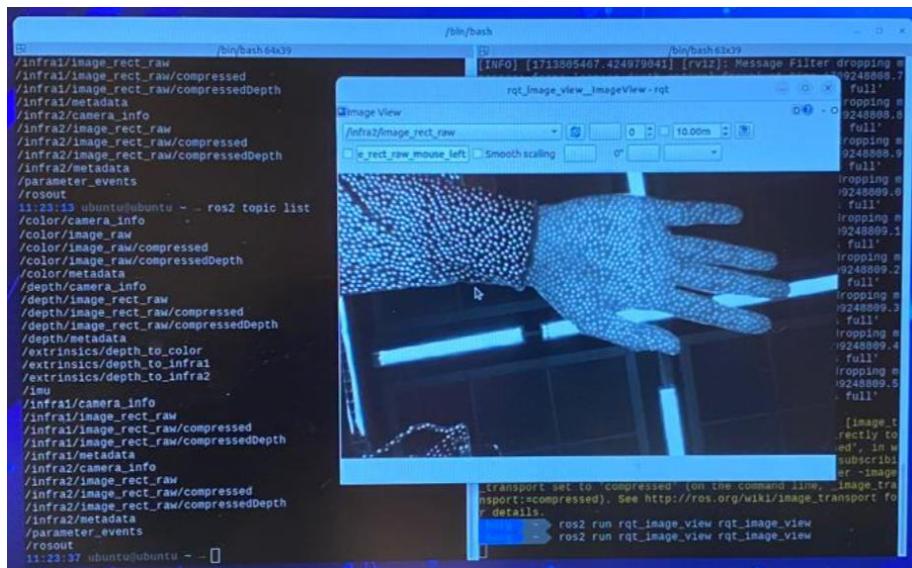
Com a inclusão da biblioteca, é possível conectar a câmera, realizar sua inicialização e verificar as imagens publicadas em seus tópicos através de alguma interface gráfica integrada ao ROS2, como o Rviz ou o plugin rqt\_image\_view.

**Figura 37** – Imagens Visualizadas no Rviz.



**Fonte:** elaborado pelos autores (2024).

**Figura 38** – Imagens Visualizadas no rqt\_image\_view.

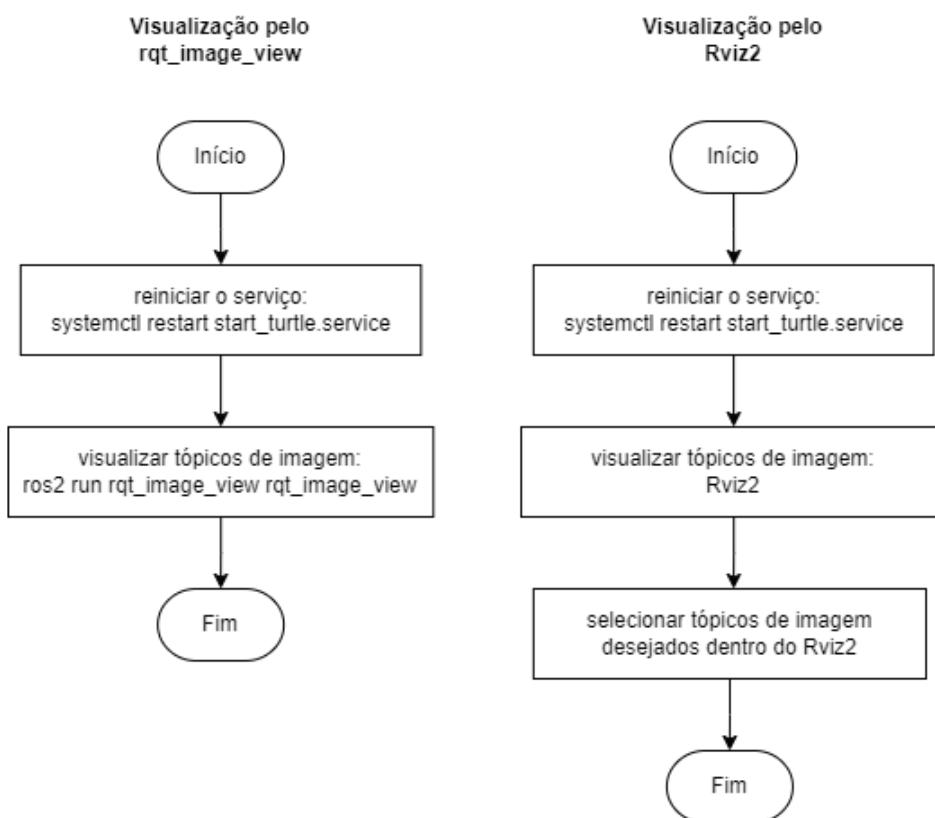


**Fonte:** elaborado pelos autores (2024).

Uma vez que as imagens fornecidas estavam em formato RAW, outra etapa adotada foi ativar os nodes responsáveis por seu transporte para publicarem e se inscreverem em tópicos de imagem comprimida, como JPG e PNG. Para realização desse processo foi feita a instalação de outro plugin, chamado compressed\_image\_transport.

Para representar as etapas de conexão da câmera para visualização dos tópicos de imagem tanto no rqt\_image\_view quanto no Rviz foram elaborados dois fluxogramas (Figura 39). Nota-se a necessidade de reiniciar o serviço para garantir a conexão da câmera, que pode ser verificada através do comando “ros2 topic list”, caso os tópicos de imagem estejam listados na resposta do terminal, a câmera está corretamente conectada.

**Figura 39** – Fluxogramas de obtenção de imagens.



**Fonte:** elaborado pelos autores (2024).

Para estabilização e fixação da câmera e do lidar, foram desenvolvidos suportes em PLA e furos na tampa superior foram realizados para o cabeamento desses dispositivos.

## Resultados

Finalizando o projeto detalhado realizou-se a montagem do esqueleto do robô em alumínio, bem como a solda de todos os componentes nas placas de controle e acionamento. Após a conclusão desta etapa isso iniciou-se os testes após a aquisição das peças faltantes.

Os atuadores responderam aos pulsos PWM enviados pelos drivers e as leituras dos encoders chegaram ao microprocessador. As referências de posição dadas aos servomotores foram seguidas por eles. Os sinais de rádio do controle foram lidos por um dos nós de teste. Neste ponto, todos os principais nós já foram validados separadamente, restando executar o nó que se comunica com todos os outros.

Com o RPi configurado, iniciou-se a preparação do rover. Foram tomadas algumas precauções, como colocar o corpo do rover em uma caixa para que as rodas pudessem girar sem que o rover atingisse algo acidentalmente. Por fim, calibrou-se os servos de canto. Os motores de servo em cada canto têm codificadores absolutos que lembram onde as rodas estão mesmo quando a energia está desligada. No entanto, ainda foi necessário calibrá-los pela primeira vez para que soubessem onde está a posição neutra.

Após a calibração dos servos de canto, o rover estava pronto para ser testado. O software foi carregado e os parâmetros foram ajustados para corresponder às especificações do rover. Os testes iniciais foram realizados com o rover em uma caixa para evitar danos acidentais.

Ao colocar em prática o nó de comunicação, o rover inicialmente se movia apenas em uma direção. Para aprimorar sua funcionalidade, o código foi modificado permitindo que o rover se movesse para frente e para trás, controlado pelo joystick esquerdo do controle remoto. Além disso, a velocidade do rover foi configurada para variar de acordo com a posição do joystick, proporcionando um controle mais preciso e eficiente do veículo.

Imediatamente após as primeiras manobras, notou-se um problema durante as curvas executadas pelo rover. A roda interna, que recebia uma maior força devido à dinâmica da curva, apresentou um ganho de cambagem excessivo conforme o esterçamento aumentava. Isso

indicava uma distribuição de força desequilibrada que poderia comprometer a integridade estrutural do rover e sua capacidade de manobra.

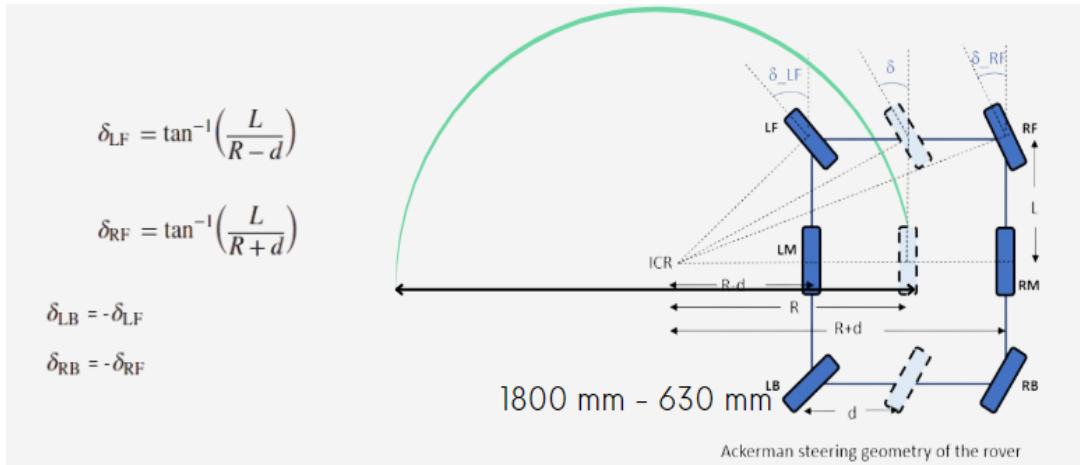
**Figura 40** – Ganho excessivo de cambagem



**Fonte:** elaborado pelos autores (2024).

Diante dessa situação, decidiu-se intervir no código de controle do rover. A solução encontrada foi aumentar o raio de curva. Isso permitiu uma distribuição mais uniforme da força entre as rodas durante as curvas, minimizando o estresse na roda interna e preservando a estrutura do rover. Essa modificação não só resolveu o problema observado, mas também melhorou a eficiência e a segurança das manobras do rover. Como a bitola externa maior do rover (largura) é de aproximadamente 630 mm e considerando o distanciamento máximo entre as filas de 1,8 m, o maior raio de curvatura do centro de massa do rover, para que esse consiga retornar apenas esterçando as rodas ao estar mais próximo de uma das filas é de  $\frac{(1800-630)}{2} = 585$  mm. Com esse valor de raio de curva foi possível alterar os ângulos de Ackerman das rodas e deixá-los como descrito no final da seção de mapeamento do controle. (MathWorks, 2024).

**Figura 41** – Ângulos de Ackerman.



**Fonte:** elaborado pelos autores (2024).

Esses novos valores de esterçamento foram alterados em ‘rover.py’ contido em: osr-rover-code/ROS/osr\_control/osr\_control/, para que self.min\_radius e self.max\_radius correspondessem ao novo raio de curva encontrado.

**Figura 42** – Ganho de cambagem ajustado.



**Fonte:** elaborado pelos autores (2024).

Devido ao torque dos motores, houve a necessidade de modificar a fixação das rodas. Antes a fixação foi estabelecida entre o eixo do motor e a roda com um parafuso central

pressionando a roda entre uma arruela e o cubo de roda acoplado ao eixo do motor. Após a nova fixação, as rodas foram furadas e fixadas, por dois parafusos cada, aos cubos de roda. No entanto foi verificada que essa estratégia, sugerida no repositório oficial do projeto, é passível de desalinhamentos.

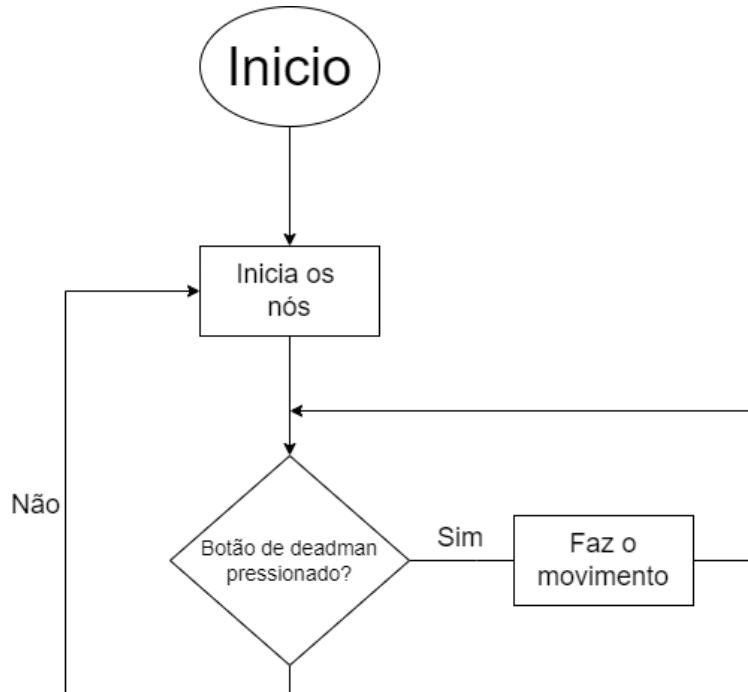
**Figura 43** – Exemplo de fixação da roda ao cubo.



**Fonte:** elaborado pelos autores (2024).

Com o objetivo de garantir a segurança durante a operação do rover, decidiu-se implementar um recurso conhecido como “botão deadman”. Este botão funciona como um interruptor de segurança, permitindo que o rover se movimente apenas quando este botão estiver pressionado. Isso proporciona um controle mais seguro, pois o rover só se movimenta quando há uma ação intencional do operador.

**Figura 44** – Fluxograma de funcionamento do deadman.



**Fonte:** elaborado pelos autores (2024).

Além disso, adicionou-se um botão de parada total ao sistema. Este botão, redundante e retentivo, serve como uma medida de segurança adicional. Quando acionado, ele interrompe imediatamente todos os movimentos do rover, independentemente de quaisquer outros comandos que possam estar sendo enviados ao mesmo tempo. Este botão de emergência garante que o rover possa ser rapidamente desativado em caso de qualquer situação imprevista ou potencialmente perigosa.

Essas medidas de segurança foram cuidadosamente projetadas para garantir que o rover possa ser operado de maneira segura e eficaz, minimizando o risco de danos ao equipamento ou ao ambiente ao seu redor.

**Figura 45** – Mapa dos botões do controle frente



**Fonte:** Controle DXS Acesso em 13 maio 2024

**Figura 46** – Mapa dos botões do controle trás



**Fonte:** Controle DXS Acesso em 13 maio 2024.

Como descrito no esboço de solução e na seção Detalhamento Circuito Display, um display LCD 16x02 com módulo de conversão paralelo para I<sup>2</sup>C foi utilizado para exibir informações do sistema. Ele foi conectado à placa de controle pelos pinos I<sup>2</sup>C disponíveis e pelos pinos de alimentação. O script em python para exibição das informações no display roda periodicamente, podendo esse tempo ser alterado, isolado do ROS, trazendo uma segurança a mais ao usuário ao ter um indicador do sistema linux separado dos nós do robô.

**Figura 47** – Display conectado ao microcomputador exibindo “None%”



**Fonte:** elaborado pelos autores (2024).

Como requisitado pelo cliente, também foram desenvolvidos alguns suportes para câmeras e um LIDAR, utilizando o software Fusion 360 para modelagem e a tecnologia de impressão 3D para materialização do projeto.

O material utilizado para a impressão 3D foi o PLA (poliácido láctico), um polímero de origem renovável amplamente utilizado na manufatura aditiva devido à sua biodegradabilidade e propriedades mecânicas adequadas. O PLA apresenta uma relação favorável entre rigidez e peso, sendo fácil de imprimir, o que contribui para a eficiência no processo de prototipagem e na qualidade final dos suportes. Este material é derivado de fontes naturais como milho e cana-de-açúcar, o que reduz o impacto ambiental em comparação com polímeros tradicionais. A facilidade de uso do PLA também se deve à sua baixa temperatura de extrusão e à ausência de deformações significativas durante a impressão, tornando-o uma escolha popular em diversas aplicações industriais e educacionais (Tumer, 2021).

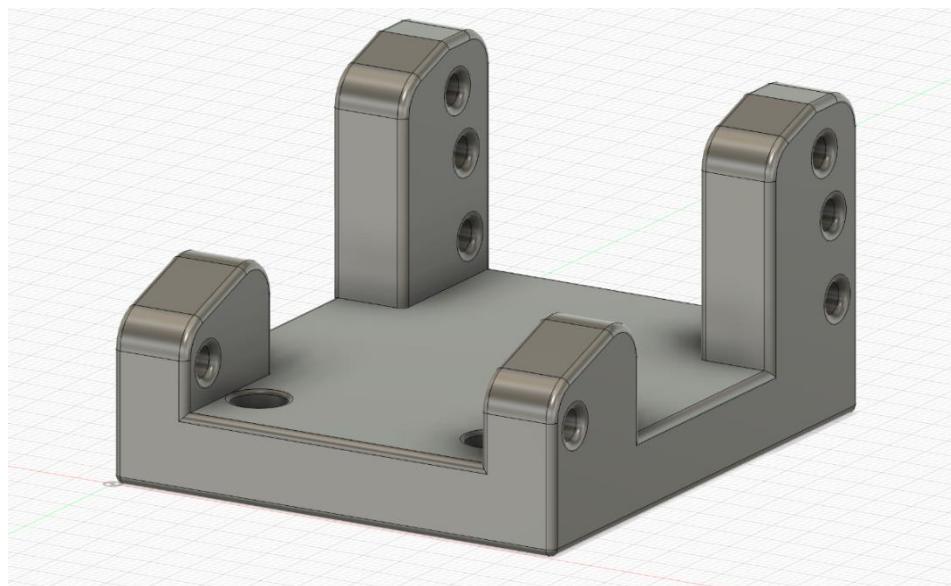
Este suporte foi pensado principalmente na funcionalidade, mas também levado em conta o design e o espaço que ele tinha que ser encaixado, além da praticidade da câmera poder mudar de angulação quando necessário.

Posteriormente, a impressão 3D permitiu a materialização do projeto. A utilização desta tecnologia possibilitou iterações rápidas de prototipagem e refinamento do design, resultando em um produto que atende tanto aos requisitos estéticos quanto aos funcionais.

A escolha do PLA proporcionou leveza e resistência ao suporte, garantindo estabilidade na captura de imagens. Comparado ao alumínio, o PLA oferece uma alternativa mais leve, não sobrecarregando o rover nem alterando seu centro de gravidade, além de permitir ajustes personalizados conforme as demandas específicas de cada situação de filmagem.

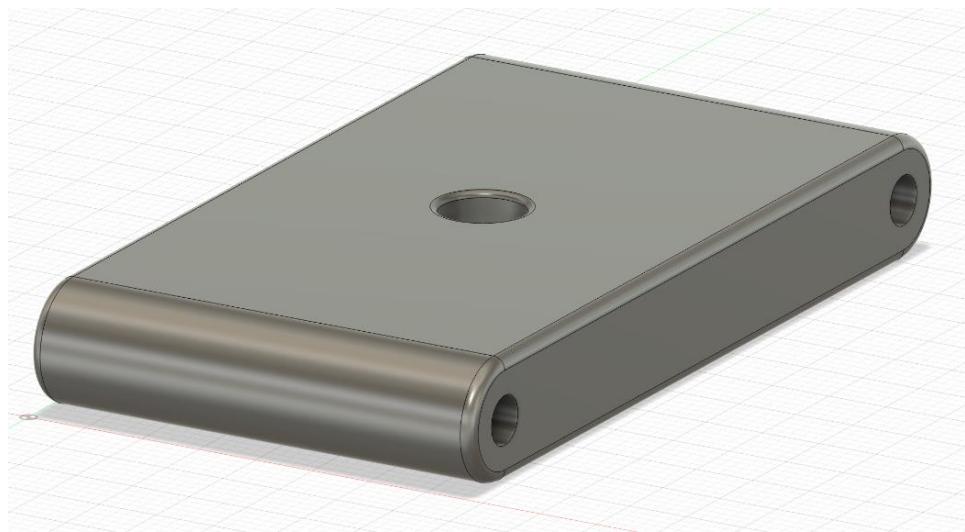
Abaixo imagens 3D dos modelos feitos das duas peças do suporte da câmera Realsense D435I da Intel e do Lidar Ouster os-1 de 16 canais.

**Figura 48** – Modelagem 3D da base do suporte da câmera no Fusion360



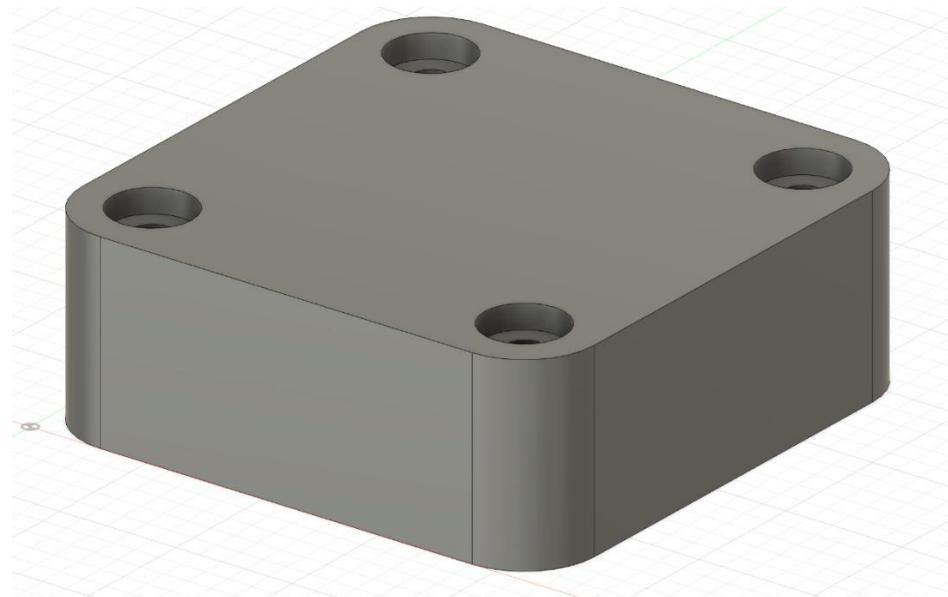
**Fonte:** elaborado pelos autores (2024).

**Figura 49** – Modelagem 3D do acoplamento para suporte da câmera



**Fonte:** elaborado pelos autores (2024).

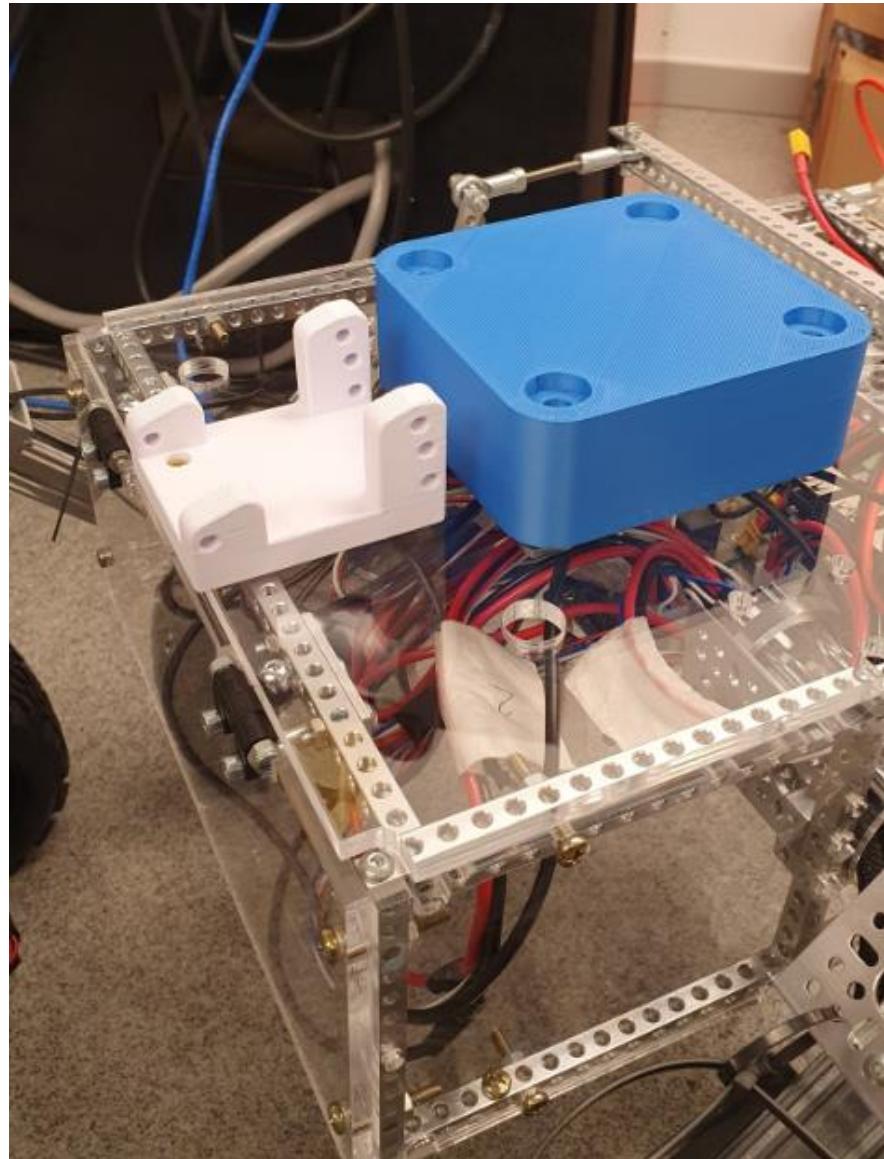
**Figura 50–** Modelagem 3D do suporte do LIDAR no Fusion360



**Fonte:** elaborado pelos autores (2024).

A seguir, é possível visualizar ambos os suportes montados na estrutura de acrílico do robô.

**Figura 51 – Suporte para câmera e LiDAR**

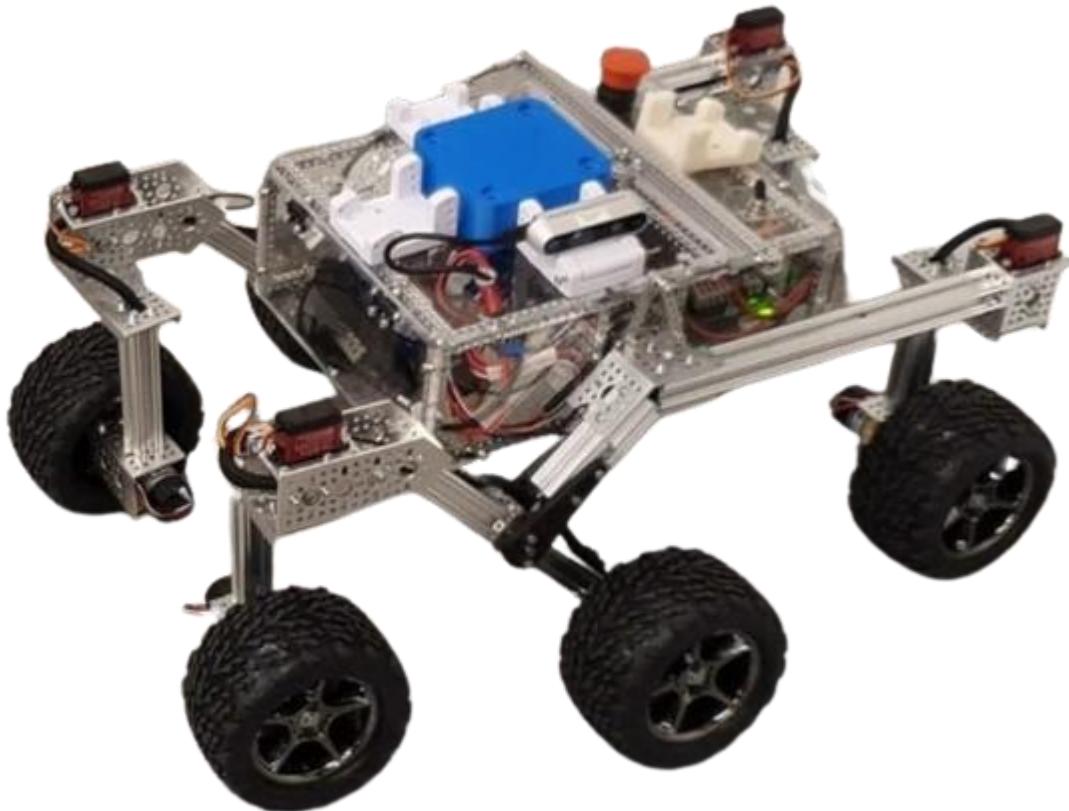


**Fonte:** elaborado pelos autores (2024).

## **Conclusões e trabalhos futuros**

Realizados os testes finais, conclui-se que o objetivo estabelecido em parceria com o cliente foi atingido, uma vez que o robô apresenta toda a estrutura necessária para se movimentar em terrenos irregulares, bem como para abrigar componentes necessários durante a etapa de automatização, com compatibilidade com os modelos de câmera e de LiDAR requisitados pela empresa.

**Figura 52** – Robô completo.



**Fonte:** elaborado pelos autores (2024).

Como recomendação para os futuros responsáveis pela etapa de automatização foram sugeridas algumas melhorias nos seguintes tópicos de cada subsistema:

- Em mecânica será oportuno buscar por alternativas para melhorar a fixação das rodas e alinhamento da suspensão, uma vez que alguns pneus ficaram levemente desalinhados após a montagem da roda no cubo, o que poderia levar a um desgaste irregular da estrutura. A substituição por outro de maior diâmetro e rigidez do pneu também garantiria ultrapassagem de obstáculos maiores. Além disso, durante a compra da estrutura do robô, não foi entregue uma das peças de alumínio responsável por estruturar o corpo, que embora não seja vital para seu funcionamento garante melhor fixação das placas de acrílico. Uma sugestão seria de comprar ou usinar uma peça equivalente, ou, até mesmo, encontrar novas soluções nacionais para a estrutura do robô. Proteções

adicionais podem ser necessárias em alguns ambientes, principalmente para vedar a entrada de corpos estranhos e umidade nos conectores dos atuadores e no corpo do robô.

- Em eletrônica, eventualmente pode vir a ocorrer algum tipo de mau contato envolvendo os terminais tubulares presentes no circuito de emergência, caso ocorra a frequente abertura do corpo do robô. Existe também a possibilidade de interrupção no cabeamento da suspensão, uma vez que ele entra em contato com a estrutura metálica ao longo da passagem por terrenos irregulares, e embora protegido por uma malha expansiva, pode vir a ser danificado pelo uso contínuo.
- Em programação, alguns outros features poderiam ser adicionados. Por exemplo, poderíamos implementar uma função de segurança para garantir que, em caso de perda de sinal com o controle, o rover interrompa imediatamente sua movimentação. Adicionalmente, poderíamos criar rotinas para mapear o ambiente utilizando a técnica de Localização e Mapeamento Simultâneos (SLAM), com o auxílio de câmeras e LiDAR, o que possibilitaria a navegação autônoma do rover.

Foi criado um repositório<sup>2</sup> no GitHub chamado, unmaned-ground-vehicle-2024.1, contendo todas as informações detalhadas sobre o projeto, incluindo códigos, documentação e instruções de uso. Este repositório serve como uma plataforma colaborativa para o desenvolvimento contínuo e aperfeiçoamento do UGV, permitindo que outros pesquisadores e desenvolvedores contribuam e accessem os recursos do projeto.

---

<sup>2</sup> Fonte: Elaborado pelos autores. 2024. Disponível em: <https://github.com/pfeinsper/unmaned-ground-vehicle-2024.1.git>.

## Referências

BRITTO, VINÍCIUS. Valor de produção da silvicultura e da extração vegetal cresce 11,9% e atinge recorde de R\$ 33,7 bilhões. 2023. Disponível em: <https://agenciadenoticias.ibge.gov.br/agencia-noticias/2012-agencia-de-noticias/noticias/37963-valor-de-producao-da-silvicultura-e-da-extracao-vegetal-cresce-11-9-e-atinge-recorde-de-r-33-7-bilhoes>. Acesso em 25 fev. 2024.

BURRO. Company. 2024. Disponível em: <https://burro.ai/company/>. Acesso em: 08 maio. 2024.

BURRO. Technical Specifications Sheet (Gen 8). Disponível em: <https://burro.ai/wp-content/uploads/2024/02/Technical-Specifications-Sheet-Gen-8.pdf>. Acesso em: 08 maio. 2024.

BURRO. Technology. Disponível em: <https://burro.ai/technology/>. Acesso em: 08 maio. 2024.

CAMARGO LAMPARELLI, Rubens Augusto. 2022. Disponível em: <https://www.embrapa.br/agencia-de-informacao-tecnologica/cultivos/cana/producao/avanco-tecnologico/agricultura-de-precisao>. Acesso em 08 maio. 2024.

COSENZA, Chiara et al. Spring-Loaded Rocker-Bogie Suspension for Six Wheeled Rovers. Disponível em: [https://www.researchgate.net/publication/362493224\\_Spring-Loaded\\_Rocker-Bogie\\_Suspension\\_for\\_Six\\_Wheeled\\_Rovers](https://www.researchgate.net/publication/362493224_Spring-Loaded_Rocker-Bogie_Suspension_for_Six_Wheeled_Rovers). Acesso em: 07 maio. 2024

Datasheet PCA9685. Disponível em: <https://www.nxp.com/docs/en/datasheet/PCA9685.pdf>. Acesso em 16 fev. 2024.

DUFT, Daniel. O que é talhão. 2014. Disponível em: <https://www.inteliagro.com.br/o-que-e-talhao/>. Acesso em: 15 maio 2024

FLYABILITY. What is Simultaneous Localization and Mapping (SLAM)? Disponível em: <https://www.flyability.com/blog/simultaneous-localization-and-mapping>. Acesso em: 06 maio. 2024.

GEBLER, Luciano. Fruticultura de precisão: uma análise dos desafios e uma nova visão de produção. 2017. Disponível em: <https://www.infoteca.cnptia.embrapa.br/infoteca/bitstream/doc/1069932/1/GeblerAgapomiN275P102017.pdf>. Acesso em: 15 maio 2024

GITHUB. unmaned-ground-vehicle-2024.1. Disponível em: <https://github.com/pfeinsper/unmaned-ground-vehicle-2024.1.git>. Acesso em: 9 jun 2024

GITHUB. open-source-rover. Disponível em: <https://github.com/nasa-jpl/open-source-rover>. Acesso em: 2 fev. 2024.

GITHUB. osr-rover-code. Disponível em: <https://github.com/nasa-jpl/osr-rover-code>. Acesso em: 2 fev. 2024.

IBGE. Agência de Notícias. Disponível em: [https://agenciadenoticias.ibge.gov.br/media/com\\_mediaibge/arquivos/963e3c571b4e164d31832b38f292ebbc.pdf](https://agenciadenoticias.ibge.gov.br/media/com_mediaibge/arquivos/963e3c571b4e164d31832b38f292ebbc.pdf). Acesso em: 11 mai. 2024.

INA260 Precision Digital Current and Power Monitor With Low-Drift, Precision Integrated Shunt. Disponível em: [https://www.ti.com/lit/ds/symlink/ina260.pdf?ts=1710599397988&ref\\_url=https%253A%252F%252Fwww.google.com%252F](https://www.ti.com/lit/ds/symlink/ina260.pdf?ts=1710599397988&ref_url=https%253A%252F%252Fwww.google.com%252F). Acesso em 16 fev. 2024.

INTEL. 2024. Disponível em: <https://github.com/IntelRealSense/realsense-ros>. Acesso em: 11 mai. 2024.

JUNGES, Rafael. Uma Breve História da Mecanização Agrícola. 2019. Disponível em <https://www.austertecnologia.com/single-post/mecanizacao-agricola-historia>. Acesso em: 08 maio. 2024

KATIKARIDIS, Dimitrios et al. UAV-Supported Route Planning for UGVs in Semi-Deterministic Agricultural Environments. Agronomy. 2022. Disponível em:

[https://www.researchgate.net/publication/362771341\\_UAV-Supported\\_Route\\_Planning\\_for\\_UGVs\\_in\\_Semi-Deterministic\\_Agricultural\\_Environments](https://www.researchgate.net/publication/362771341_UAV-Supported_Route_Planning_for_UGVs_in_Semi-Deterministic_Agricultural_Environments). Acesso em: 08 maio. 2024.

MARAN, Jessica et al. Ordenamento florestal por talhões: metodologia apoiada em SIG e silvicultura para o manejo de florestas nativas. 2020. Disponível em: <https://ainfo.cnptia.embrapa.br/digital/bitstream/item/215359/1/Augsta-8379-Article-Text-39644-1-10-20200628-1.pdf>. Acesso em: 15 maio 2024

MATIAS, Átila. "Revolução Verde"; Brasil Escola. Disponível em: <https://brasilescola.uol.com.br/geografia/revolucao-verde.htm>. Acesso em 05 maio. 2024.

MATHWORKS. SLAM (Simultaneous Localization and Mapping). Disponível em: <https://www.mathworks.com/discovery/slam.html>. Acesso em: 06 maio. 2024.

MathWorks. Modeling and Control of a Mars Rover. Disponível em: [https://www.mathworks.com/help/sm/ug/mars\\_rover.html](https://www.mathworks.com/help/sm/ug/mars_rover.html). Acesso em: 11 mai 2024.

MATIJEVIC, Jake Autonomous Navigation and the Sojourner Microrover. Science, v. 280, n. 5362, p. 454-455, ano 1998. Disponível em: [https://www.science.org/doi/full/10.1126/science.280.5362.454?casa\\_token=EthL-s842fYAAAAA%3A3D7DIBJI-bwQ-UqiA-OA46aNay9CXwwvci8sF-Pbqueux3UiL4yBe9bxwkKmVvdv4nvWPDBw33ERzFQ](https://www.science.org/doi/full/10.1126/science.280.5362.454?casa_token=EthL-s842fYAAAAA%3A3D7DIBJI-bwQ-UqiA-OA46aNay9CXwwvci8sF-Pbqueux3UiL4yBe9bxwkKmVvdv4nvWPDBw33ERzFQ). Acesso em: 07 maio. 2024.

NASA. Mars Pathfinder. Disponível em: <https://science.nasa.gov/mission/mars-pathfinder>. Acesso em: 07 maio. 2024.

NASA. Mars Pathfinder Rover. Disponível em: <https://nssdc.gsfc.nasa.gov/nmc/spacecraft/display.action?id=MESURPR>. Acesso em: 07 maio. 2024.

NASA. Open Source Rover. Disponível em: <https://github.com/nasa-jpl/open-source-rover>. Acesso em: 27 fev. 2024.

OPEN ROBOTICS. ROS Documentation. Disponível em: <https://docs.ros.org/en/foxy>. Acesso em: 11 maio. 2024.

PINTO DA SILVA, Gerarda Beatriz. Agricultura de Precisão: conceitos básicos e aplicações práticas. 2019. Disponível em: <https://institutoagro.com.br/agricultura-de-precisao/>. Acesso em 08 maio. 2024.

Raspberry Pi 4 Model B. Disponível em: <https://datasheets.raspberrypi.com/rpi4/raspberry-pi-4-datasheet.pdf>. Acesso em 16 fev. 2024.

**ROZENFELD, H.; FORCELLINI, F. A.; AMARAL, D. C.; et al. Gestão de Desenvolvimento de Produto: uma referência para a melhoria do processo.** 1 ed. São Paulo: Saraiva, 2006.

SSH ACADEMY. 2023. Disponível em: <https://www.ssh.com/academy/ssh/protocol>. Acesso em: 10 mai. 2024.

TAWIL, YAHYA. An Introduction to Robot Operating System (ROS). <https://www.allaboutcircuits.com/technical-articles/an-introduction-to-robot-operating-system-ros/> Acesso em 18 mar 2024

THE PLANETARY SOCIETY. Viking 1 and 2, NASA's first Mars landers. Disponível em: <https://www.planetary.org/space-missions/viking>. Acesso em: 07 maio. 2024.

TUMER, E.; ERBIL, H. Extrusion-Based 3D Printing Applications of PLA Composites: A Review. Disponível em <https://www.mdpi.com/2079-6412/11/4/390>. Acesso em: 08 jun. 2024

UNITED STATES DEPARTMENT OF AGRICULTURE - ECONOMIC RESEARCH SERVICE. Summary of recent findings. 2024. Disponível em: <https://www.ers.usda.gov/data-products/agricultural-productivity-in-the-u-s/summary-of-recent-findings/#:~:text=It%20is%20widely%20agreed%20that,1.46%20percent%20over%20the%20period>. Acesso em: 15 mar. 2024.

V.FONSECA, LETÍCIA. Fruticultura Brasileira: Diversidade e sustentabilidade para alimentar o Brasil e o Mundo. 2022. Disponível em: <https://cnabrasil.org.br/noticias/fruticultura-brasileira-diversidade-e-sustentabilidade-para-alimentar-o-brasil-e-o-mundo>. Acesso em: 25 fev. 2024.

WANG LING, SUN. NJUKI, ERIC. NEHRING, RICHARD. MOSHEIM ROBERTO. Agricultural Productivity in the U.S. <https://www.ers.usda.gov/data-products/agricultural-productivity-in-the-u-s/summary-of-recent-findings/#:~:text=It%20is%20widely%20agreed%20that,1.46%20percent%20over%20the%20period.> Acesso em 15 mar. 2024.

## Apêndice A

<b>Parts for drive wheel assembly</b>	
short name	link
wheel	Traxxas 5374X Talon Tires, Gemini Wheels, Black Chrome - Dollar Hobby
clamping mount	1401 Series 2-Side, 2-Post Clamping Mount (43mm Width, 36mm Bore) - <a href="#">gobILDA</a>
motor	5203 Series Yellow Jacket Planetary Gear Motor (26:9:1 Ratio, 24mm Length)
REX bore hub	\$42.99
2 Hole U channel	1310 Series Hyper Hub (8mm REX™ Bore)
	\$7.99
	\$4.99
	1120 Series U-Channel (2 Hole, 72mm Length) - <a href="#">gobILDA</a>
	\$29.94

<b>Parts for corner assembly</b>	
short name	link
144mm goRail	1109 Series goRAIL (144mm Length) - <a href="#">gobILDA</a>
4 Hole U channel	1121 Series Low-Side U-Channel (4 Hole, 120mm Length) - <a href="#">gobILDA</a>

<b>Parts for rocker bogie assembly</b>	
control arm	link
1hole uchannel	1120 Series U-Channel (1 Hole, 48mm Length) - <a href="#">gobILDA</a>
servoblock	ServoBlock™ (Standard Size, 25 Tooth Spline, Hub Shaft) - <a href="#">gobILDA</a>
servo	2000 Series Dual Mode Servo (25-2, Torque) - <a href="#">gobILDA</a>
8mm Sonic Hub	1309 Series Sonic Hub (8mm Bore) - <a href="#">gobILDA</a>
flanged bearing 2pack	1611 Series Flanged Ball Bearing (8mm ID x 14mm OD, 5mm Thickness) - <a href="#">gobILDA</a>
Pack - <a href="#">gobILDA</a>	\$3.99
joint pattern spacer	1504 Series 32mm OD Pattern Spacer (2mm Length) - <a href="#">gobILDA</a>
joint 8mm shaft	2100 Series Stainless Steel Round Shaft (8mm Diameter, 50mm Length)
joint 3hole flat beam 2pack	1102 Series Flat Beam (3 Hole, 24mm Length) - 2 Pack - <a href="#">gobILDA</a>
joint pattern mount	1221 Series 2-Side, 2-Pot Pattern Mount (32-2) - <a href="#">gobILDA</a>
steel flat bracket	\$4.99
3x5 grid plate	1126 Series Steel Flat Bracket (1-1) - 2 Pack - <a href="#">gobILDA</a>
angle bracket	\$2.49
96mm open goRail	1116 Series Grid Plate (3 x 5 Hole, 24 x 40mm) - <a href="#">gobILDA</a>
288mm open goRail	\$1.29
45deg bracket	\$5.99
90deg steel bracket 2pack	1111 Series Angle Pattern Bracket (3-1) - <a href="#">gobILDA</a>
control arm	\$3.39
8mm ID Spacer (10mm OD, 6mm Length)	1118 Series Open goRAIL (96mm Length) - <a href="#">gobILDA</a>
8mm ID Spacer (10mm OD, 4mm Length)	\$5.89
	1111 Series Angle Pattern Bracket (1-1) - <a href="#">gobILDA</a>
	\$2.99
	1137 Series Steel Flat Grid Bracket (1-1) - 2 Pack - <a href="#">gobILDA</a>
	\$3.49
	Plastic Hub-Mount Control Arm (72mm Length) - <a href="#">gobILDA</a>
	\$4.99
	1514 Series 8mm ID Spacer (10mm OD, 6mm Length) - 4 Pack - <a href="#">gobILDA</a>
	\$2.19
	1522 Series 8mm ID Spacer (10mm OD, 4mm Length) - 4 Pack - <a href="#">gobILDA</a>
	\$2.99

<b>Parts for body assembly</b>		<a href="#">link</a>	cost per part	total # req	total cost	Número Oracle
control arm						
120mm open goRail	<a href="#">1118 Series Open goRAIL (120mm Length) - goBILDA</a>	\$4.99	2	\$9.98	UC.26776	
2hole Uchannel	<a href="#">1120 Series U-Channel (2 Hole, 72mm Length) - goBILDA</a>	\$4.99	2	\$9.98	UC.26777	
32mm bearing	<a href="#">1604 Series 2-Side, 2-post pillow Block (32mm Bore, 6mm Length) - 2 Pack - goBILDA</a>	\$12.99	4	\$51.96	UC.26778	
32mm 6mm spacer 2 pack	<a href="#">1506 Series 32mm ID Spacer (36mm OD, 6mm Length) - 2 Pack - goBILDA</a>	\$3.49	3	\$10.47	UC.26779	
29 hole beam	<a href="#">1106 Series Square Beam (29 Hole, 232mm Length) - goBILDA</a>	\$5.09	5	\$25.45	UC.26780	
41 hole beam	<a href="#">1106 Series Square Beam (41 Hole, 328mm Length) - goBILDA</a>	\$6.99	4	\$27.96	UC.26781	
12 hole beam	<a href="#">1106 Series Square Beam (12 Hole, 96mm Length) - goBILDA</a>	\$2.99	8	\$23.92	UC.26782	
41 hole u-beam beam	<a href="#">1101 Series U-Beam (41 Hole, 328mm Length) - goBILDA</a>	\$3.89	1	\$3.89	UC.26783	
7 hole flat beam 2pack	<a href="#">1102 Series Flat Beam (7 Hole, 56mm Length) - 2 Pack - goBILDA</a>	\$2.49	2	\$4.98	UC.26784	
dual block mount	<a href="#">1205 Series Dual Block Mount (1-4) - 2 Pack - goBILDA</a>	\$5.99	2	\$11.98	UC.26785	
90deg threaded gusset 4pack	<a href="#">1203 Series Block Mount (1-1) - 4 Pack - goBILDA</a>	\$4.99	1	\$4.99	UC.26786	
50mm threaded rod	<a href="#">2808 Series Stainless Steel Threaded Rod (M4 x 0.7mm, 50mm Length) - 2 Pack - goBILDA</a>	\$1.49	1	\$1.49	UC.26787	
ball linkage 2pack	<a href="#">2913 Series Steel Ball Linkage (Female M4 x 0.7mm, 24.1mm Length) - 2 Pack - goBILDA</a>	\$4.99	2	\$9.98	UC.26788	
Idler Bearing Hub	<a href="#">Idler Bearing-Hub (32mm OD, 16mm Height) - goBILDA</a>	\$12.99	1	\$12.99	UC.26789	
plastic hinges	<a href="#">Plastic Hinge - 2 Pack - goBILDA</a>	\$4.99	2	\$9.98	UC.26790	
8mm standoffs	<a href="#">1501 Series M4 x 0.7mm Standoff (6mm OD, 8mm Length) - 4 Pack</a>	\$2.49	2	\$4.98	UC.26791	
22mm standoffs	<a href="#">1501 Series M4 x 0.7mm Standoff (6mm OD, 22mm Length) - 4 Pack</a>	\$3.49	2	\$6.98	UC.26792	
plastic spacer	<a href="#">1500 Series Plastic Spacer (6mm ID x 8mm OD, 1mm Thickness) - 12 Pack</a>	\$2.79	1	\$2.79	UC.26793	
3x5 grid plate	<a href="#">1116 Series Grid Plate (3 x 5 Hole, 24 x 40mm)</a>	\$1.29	4	\$5.16	UC.26794	
2 hole flat beam	<a href="#">1102 Series Flat Beam (2 Hole, 16mm Length) - 2 Pack</a>	\$1.79	2	\$3.58	UC.26795	
				4		
<b>Parts for general assembly</b>		<a href="#">link</a>	cost per part	total # req	total cost	Número Oracle
short name						
hurricane nuts	<a href="#">Hurricane Nut for goRAIL - 25 Pack - goBILDA</a>	\$9.99	1	\$9.99	UC.26796	
wire grommets	<a href="#">Plastic Grommet (14-1-12 Pack - goBILDA</a>	\$1.99	1	\$1.99	UC.26797	
M4 washers	<a href="#">Zinc Plates Steel Washer (M4 x 8mm OD) - 25 Pack - goBILDA</a>	\$1.99	1	\$1.99	UC.26798	
M4 nuts	<a href="#">Zinc Plates Steel Hex Nut (M4 x 7) - 25 Pack - goBILDA</a>	\$2.49	1	\$2.49	UC.26799	
M4 locknuts	<a href="#">Zinc Plates Steel Locknut (M4 x 7) - 25 Pack - goBILDA</a>	\$2.99	1	\$2.99	UC.26800	
M4x6 button screws	<a href="#">M4x6 Button Screw 25-pack</a>	\$2.99	2	\$5.98	UC.26801	
M4x6 button screws	<a href="#">M4x10 Button Screw 25-pack</a>	\$3.39	3	\$10.17	UC.26802	
M4x6 button screws	<a href="#">M4x16 Button Screw 25-pack</a>	\$3.79	2	\$7.58	UC.26803	
M4x6 socket screws	<a href="#">M4x8 Socket Screw 25-pack</a>	\$3.19	2	\$6.38	UC.26804	
M4x8 socket screws	<a href="#">M4x10 Socket Screw 25-pack</a>	\$3.19	3	\$9.57	UC.26805	
M4x10 socket screws	<a href="#">M4x12 Socket Screw 25-pack</a>	\$3.19	1	\$3.19	UC.26806	
M4x12 socket screws	<a href="#">M4x16 Socket Screw 25-pack</a>	\$3.19	2	\$6.38	UC.26807	
M4x20 socket screws	<a href="#">M4x20 Socket Screw 25-pack</a>	\$3.09	1	\$3.09	UC.26808	

## Parts for electrical assembly

short name	link	cost per part	total # req	total cost	Número Oracle
A1: CAP CER 10000PF 50V X7R RADIAL	<a href="#">399-98865-1-ND</a>	\$0.164	20	\$3.28	AF.20478
A2: CAP ALUM 100uF 20% 50V RADIAL	<a href="#">399-18272-1-ND</a>	\$0.248	10	\$2.48	AF.20479
A3: CAP CER 0.1uF 50V X7R RADIAL	<a href="#">399-14065-1-ND</a>	\$0.211	10	\$2.11	AF.20480
A4: CAP ALUM 10uF 20% 50V RADIAL	<a href="#">399-ESK106M050AC3DACT-ND</a>	\$0.162	10	\$1.62	AF.20481
A5: DIODE SCHOTTKY 25V 10A TO220AC	<a href="#">497-2738-5-ND</a>	\$1.93	2	\$3.86	AF.20482
A6: DIODE GEN PURP 75V 300MA DO35	<a href="#">1655-1N4148CT-ND</a>	\$0.108	10	\$1.08	AF.20483
A7: FUSEHOLDER BLOCKS - PCB - CLIP C	<a href="#">732-11376-ND</a>	\$0.68	1	\$0.68	AF.20484
A10: CONN HEADER VERT 12POS 2.54MM	<a href="#">TSW-104-07-F-T-ND</a>	\$1.5	4	\$6.0	AF.20485
A11: CONN HEADER VERT 8POS 2.54MM	<a href="#">S1012E-08-ND</a>	\$0.47	4	\$1.88	AF.20486
A12: CONN HDR 20POS 0.1 TIN PCB	<a href="#">S7078-ND</a>	\$1.3	1	\$1.3	AF.20487
A13: CONN HEADER VERT 40POS 2.54MM	<a href="#">S9175-ND</a>	\$0.73	2	\$1.46	AF.20488
A14: CONN HEADER VERT 6POS 2MM	<a href="#">455-1708-ND</a>	\$0.36	2	\$0.72	AF.20489
A16: CONN HEADER VERT 20POS 2.54MM	<a href="#">WM65548-ND</a>	\$2.52	1	\$2.52	AF.20490
A17: CONN HDR 2POS 0.1 TIN PCB	<a href="#">S7000-ND</a>	\$0.32	5	\$1.6	AF.20491
A19: TERM BLK 2P SIDE ENT 5.08MM PCB	<a href="#">ED2580-ND</a>	\$0.73	2	\$1.46	AF.20492
A20: CONN HEADER VERT 6POS 4.2MM	<a href="#">1726750613-ND</a>	\$2.27	6	\$13.62	AF.20493
A21: TRANS PNP 40V 0.2A TO92-3	<a href="#">2N3906TAFSCT-ND</a>	\$0.285	10	\$2.85	AF.20495
A22: RES 100 OHM 5% 1/4W AXIAL	<a href="#">CF14JT100RCT-ND</a>	\$0.056	10	\$0.56	AF.20496
A23: RES 10K OHM 5% 1/4W AXIAL	<a href="#">CF14JT10K0CT-ND</a>	\$0.056	10	\$0.56	AF.20497
A24: RES 4.7K OHM 5% 1/4W AXIAL	<a href="#">CF14JT4K70CT-ND</a>	\$0.0404	25	\$1.01	AF.20498
A25: RES 0 OHM JUMPER 1/4W AXIAL	<a href="#">CD14ZT0R00CT-ND</a>	\$0.079	10	\$0.79	AF.20499
A26: RES 68 OHM 5% 1/8W AXIAL	<a href="#">CF18JT68R0CT-ND</a>	\$0.045	10	\$0.45	AF.20500
A27: RES 82 OHM 5% 1/4W AXIAL	<a href="#">CF14JT82R0CT-ND</a>	\$0.056	10	\$0.56	AF.20501
A28: RES 1K OHM 5% 1/4W AXIAL	<a href="#">CF14JT1K0CT-ND</a>	\$0.056	10	\$0.56	AF.20502
A29: RES 680 OHM 5% 1/4W AXIAL	<a href="#">CF14JT680R0CT-ND</a>	\$0.056	10	\$0.56	AF.20503
A30: RES 220 OHM 5% 1/4W AXIAL	<a href="#">CF14JT220RCT-ND</a>	\$0.056	10	\$0.56	AF.20504
A32: SWITCH SLIDE DIP SPDT 50MA 24V	<a href="#">CT206124-ND</a>	\$1.25	1	\$1.25	AF.20505
A37: LED BAR GRAPH 10SEG 570NM GRN	<a href="#">754-2186-ND</a>	\$3.26	2	\$6.52	AF.20506
A38: IC REG LINEAR 3.3V 750mA TO220-3	<a href="#">576-2235-ND</a>	\$3.02	2	\$6.04	AF.20507
B1: FUSE GLASS 10A 125VAC 5X20MM	<a href="#">F3631-ND</a>	\$1.21	2	\$2.42	AF.20508
B9: SWITCH TOGGLE ON-OFF-ON	<a href="#">708-3060-ND</a>	\$3.28	1	\$3.28	AF.20509
B11: TERM BLOCK HDR 24POS VERT 5MM	<a href="#">ED1682-24-ND</a>	\$3.33	1	\$3.33	AF.20510
A9: TERM BLOCK HDR 6POS VERT 3.5MM	<a href="#">WM25701-ND</a>	\$2.02	3	\$6.06	AF.20511
B2: TERM BLOCK PLUG 6POS STR 3.5MM	<a href="#">WM13033-ND</a>	\$2.59	3	\$7.77	AF.20512

## Extra Parts

The PCBs	
Brain Board	\$62,00
Motor Board	\$92,00

The body plates	
COMPRAR 3 CHAPAS DE ACRÍLICO 3MM X 2M X 1M PEÇAS:	<a href="https://www.tudodemacrilico.com/chapa-placa-de-acrilico-2-x-1-metros-3mm-cristal-virgem.html">https://www.tudodemacrilico.com/chapa-placa-de-acrilico-2-x-1-metros-3mm-cristal-virgem.html</a>
Top Rear (254mm x 84mm)	
Front Plate Standoff (254mm x 115mm)	
Top Front (254mm x 232mm)	
Bottom Plate (254mm x 328mm)	
Side Plate (112mm x 328mm)	
Back Plate Standoff (254mm x 115mm)	
Gap Slat (126,85mm x 18,5mm)	

A gamepad or remote controller	
Spektrum WS2000	\$44,99
Spektrum DXs	\$114,99

Standoff kit for the boards	
HELIFOUNDER 420 Pieces M2.5 M3 M4 Male Female Hex Brass Spacers Standoffs Screws Nuts Assortment Kit with a Tweezers	\$90,00
	1
	\$90,00 AF.20526

Wiring	
Rolo fio 18AWG vermelho	R\$ 2,50 50m
Rolo fio 18AWG preto	R\$ 2,00 50m
Rolo fio 20AWG branco	R\$ 1,50 50m
Rolo fio 20AWG vermelho	R\$ 1,50 50m
Rolo fio 20AWG preto	R\$ 1,50 50m
Rolo fio 22AWG branco	R\$ 1,50 50m
Rolo fio 22AWG preto	R\$ 1,50 50m
Rolo fio 22AWG vermelho	R\$ 1,50 50m
Rolo fio 22AWG azul	R\$ 1,50 50m

Malha náutica expansível	Sugestão: 8mm ( <a href="https://loja.termotubos.com.br/malha-nautica-expansiva-autofechamento-8-0mm-mae?parceiro=2491&amp;gclid=EA1aiQobChMloZnLrpD9ggMVG1hAB3tK8gyEAQYBCABEgK3-vD_Bw&amp;variant_id=6355">https://loja.termotubos.com.br/malha-nautica-expansiva-autofechamento-8-0mm-mae?parceiro=2491&amp;gclid=EA1aiQobChMloZnLrpD9ggMVG1hAB3tK8gyEAQYBCABEgK3-vD_Bw&amp;variant_id=6355</a> )	R\$ 18,00	25m	R\$ 450,00	AF.20525			
GPIO Ribbon Cable for Raspberry Pi Model A+/B+/Pi 2/XT30 Connector Pack (FH-MC x 5, MH-FC x 5)	<a href="https://www.adafruit.com/product/1988">https://www.adafruit.com/product/1988</a> <a href="https://www.gobilida.com/xt30-connector-pack-fh-mc-x-5-mh-fc-x-5/">https://www.gobilida.com/xt30-connector-pack-fh-mc-x-5-mh-fc-x-5/</a> <a href="https://www.amazon.com/gp/product/B0875MBLNH/ref=ppx_yo_dt_b_search_asin_title?ie=UTF8&amp;psc=1">https://www.amazon.com/gp/product/B0875MBLNH/ref=ppx_yo_dt_b_search_asin_title?ie=UTF8&amp;psc=1</a> <a href="https://www.digikey.com/en/products/detail/dfrobot/FI0586/9559255">https://www.digikey.com/en/products/detail/dfrobot/FI0586/9559255</a> <a href="https://www.gobilida.com/xt30-lead-fh-mc-300mm-length/">https://www.gobilida.com/xt30-lead-fh-mc-300mm-length/</a> <a href="https://www.gobilida.com/xt30-lead-mh-fc-300mm-length/">https://www.gobilida.com/xt30-lead-mh-fc-300mm-length/</a> <a href="https://www.gobilida.com/encoder-breakout-cable-4-pos-ist-xh-mh-fc-to-4-x-1-pos-tjc8-mh-fc-300mm-length/">https://www.gobilida.com/encoder-breakout-cable-4-pos-ist-xh-mh-fc-to-4-x-1-pos-tjc8-mh-fc-300mm-length/</a> <a href="https://www.amazon.com/460PCS-XH2.54-Connector-Terminal-Connectors/dp/B09DBGVX5C/ref=sr_1_1_sspa?crid=2NX4ZXFZG0CK&amp;keywords=ist%2Bconnector%2Bkit%2B4%2Bpin&amp;qid=1690482910&amp;sprefix=ist%2Bconnector%2Bkit%2B4%2Bpin%2Cads%2C140&amp;sr=8-1-spons&amp;ssp_csd=d2lkz2v0TmFzT1zcF9hdGy&amp;th=1">https://www.amazon.com/460PCS-XH2.54-Connector-Terminal-Connectors/dp/B09DBGVX5C/ref=sr_1_1_sspa?crid=2NX4ZXFZG0CK&amp;keywords=ist%2Bconnector%2Bkit%2B4%2Bpin&amp;qid=1690482910&amp;sprefix=ist%2Bconnector%2Bkit%2B4%2Bpin%2Cads%2C140&amp;sr=8-1-spons&amp;ssp_csd=d2lkz2v0TmFzT1zcF9hdGy&amp;th=1</a>	\$2,95 \$4,99 \$11,99 \$1,90 \$1,99 \$1,99 \$3,99 \$1,99	1 1 1 1 2 2 6	\$2,95 \$4,99 \$11,99 \$1,90 \$3,98 \$3,98 \$23,94	AF.20524 AF.20523 AF.20522 AF.20521 AF.20513 AF.20514 AF.20515			
460PCS JST XH2.54 Connector kit 2/3/4/5/6Pin Plug with 3.5mm Bullet Lead (MH-FC, 300mm Length)	<a href="https://www.amazon.com/Twidec-Connector-Terminals-Crimping-Connectors/dp/B0B152WRSW/ref=sr_1_1_sspa?keywords=dupont+connectors+kit&amp;qid=1690482984&amp;ssprefix=DUPont%2Camps%2C190&amp;sr=8-1-spons&amp;ssp_csd=d2lkz2v0TmFzT1zcF9hdGy&amp;psc=1">https://www.amazon.com/Twidec-Connector-Terminals-Crimping-Connectors/dp/B0B152WRSW/ref=sr_1_1_sspa?keywords=dupont+connectors+kit&amp;qid=1690482984&amp;ssprefix=DUPont%2Camps%2C190&amp;sr=8-1-spons&amp;ssp_csd=d2lkz2v0TmFzT1zcF9hdGy&amp;psc=1</a> <a href="https://www.gobilida.com/3-5mm-bullet-lead-mh-fc-300mm-length/">https://www.gobilida.com/3-5mm-bullet-lead-mh-fc-300mm-length/</a> <a href="https://www.gobilida.com/3-5mm-bullet-connector-pack-mh-fc-x-5-fh-mc-x-5/">https://www.gobilida.com/3-5mm-bullet-connector-pack-mh-fc-x-5-fh-mc-x-5/</a> <a href="https://www.amazon.com/Davitu-Electrical-Equipments-Supplies-Transparent/dp/B089W96QWJ/ref=sr_1_6?keywords=3.5mm+bullet+connector+kit%2Caps%2C152&amp;sr=8-6">https://www.amazon.com/Davitu-Electrical-Equipments-Supplies-Transparent/dp/B089W96QWJ/ref=sr_1_6?keywords=3.5mm+bullet+connector+kit%2Caps%2C152&amp;sr=8-6</a>	\$7,88	1	\$7,88	AF.20517			
3.5mm Bullet Connector Pack (MH-FC x 5, FH-MC x 5)	<a href="https://www.gobilida.com/3-5mm-bullet-lead-mh-fc-300mm-length/">https://www.gobilida.com/3-5mm-bullet-lead-mh-fc-300mm-length/</a> <a href="https://www.gobilida.com/3-5mm-bullet-connector-pack-mh-fc-x-5-fh-mc-x-5/">https://www.gobilida.com/3-5mm-bullet-connector-pack-mh-fc-x-5-fh-mc-x-5/</a>	\$1,99	1	\$1,99	AF.20518			
Davitu Electrical Equipments Supplies - 120pcs 3.5mm	<a href="https://www.amazon.com/Davitu-Electrical-Equipments-Supplies-Transparent/dp/B089W96QWJ/ref=sr_1_6?keywords=3.5mm+bullet+connector+kit%2Caps%2C152&amp;sr=8-6">https://www.amazon.com/Davitu-Electrical-Equipments-Supplies-Transparent/dp/B089W96QWJ/ref=sr_1_6?keywords=3.5mm+bullet+connector+kit%2Caps%2C152&amp;sr=8-6</a>	\$10,00	1	\$10,00	AF.20520			
<b>Threadlocker</b>	<a href="https://www.gobilida.com/locrite-threadlocker-blue-242-6ml/">https://www.gobilida.com/locrite-threadlocker-blue-242-6ml/</a> <a href="https://www.lojadmecanico.com.br/produto/4028/32/235/trava-porcas-e-parafusos-locrite-277">https://www.lojadmecanico.com.br/produto/4028/32/235/trava-porcas-e-parafusos-locrite-277</a>	R\$ 86,00	1	R\$ 86,00	AF.20536			

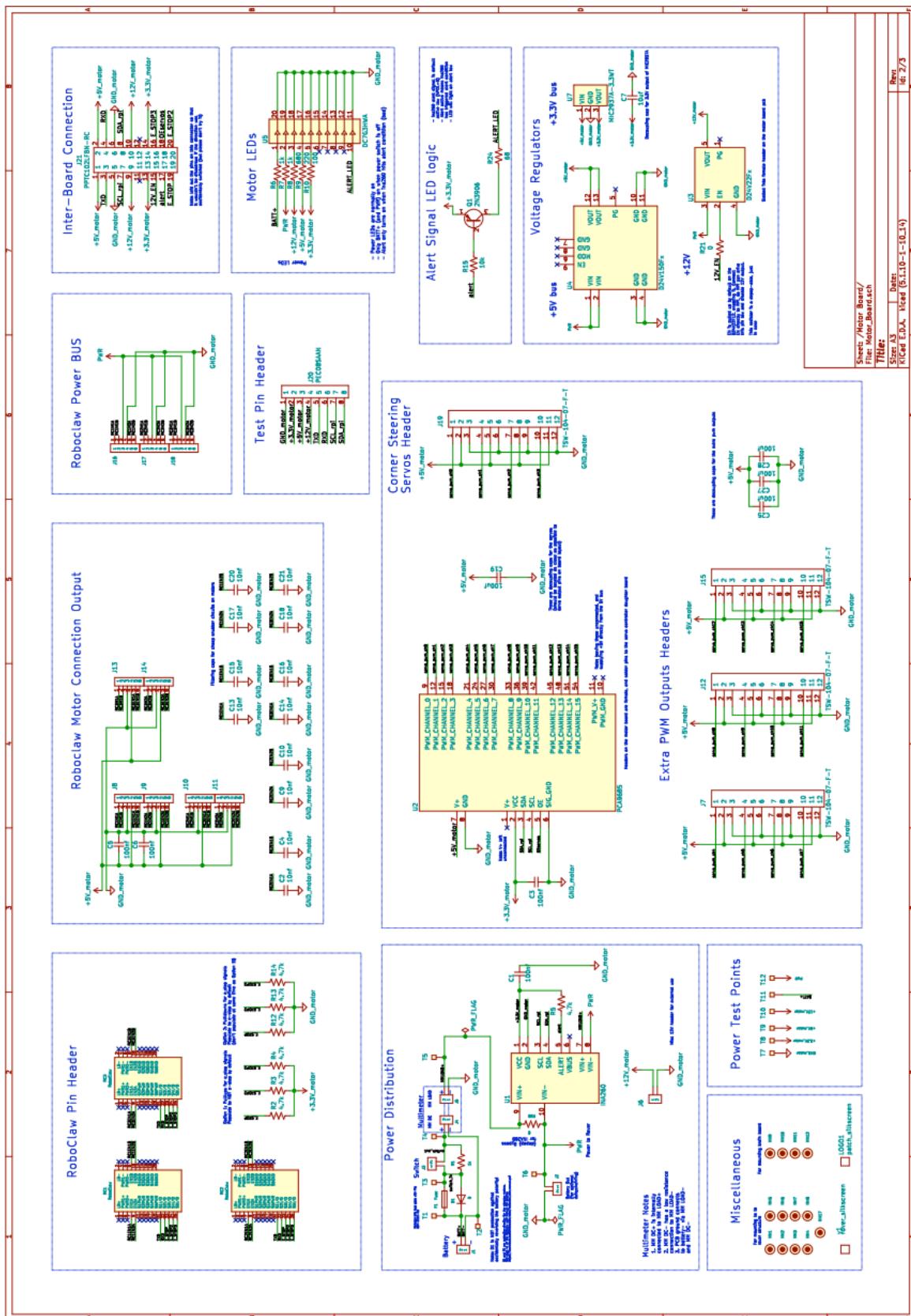
<b>Tools</b>	
7mm Combination Nut Driver	<a href="https://www.gobilida.com/7mm-combination-nut-driver/">https://www.gobilida.com/7mm-combination-nut-driver/</a>
Wera Tools 2,5mm Ball-End Hex-Plus L-Key	<a href="https://www.gobilida.com/wera-tools-2-5mm-ball-end-hex-plus-l-key/">https://www.gobilida.com/wera-tools-2-5mm-ball-end-hex-plus-l-key/</a>
Wera Tools 3mm Ball-End Hex-Plus L-Key	<a href="https://www.gobilida.com/wera-tools-3mm-ball-end-hex-plus-l-key/">https://www.gobilida.com/wera-tools-3mm-ball-end-hex-plus-l-key/</a>
<b>Electrical</b>	
1X 5V Regulator : [Pololu]	<a href="https://www.pololu.com/product/2881">https://www.pololu.com/product/2881</a>
1X 12V Regulator	<a href="https://www.pololu.com/product/2855">https://www.pololu.com/product/2855</a>
1 X PCA9685 corner	<a href="https://www.mouser.com/ProductDetail/Adafruit/Adafruit_Grayscale_9179Q%3D%3D">https://www.mouser.com/ProductDetail/Adafruit/Adafruit_Grayscale_9179Q%3D%3D</a>
3 X Roboclaw 2x7A Motor Controller	<a href="https://www.basicmicro.com/Roboclaw-2x7A-Motor-Controller_p_55.html">https://www.basicmicro.com/Roboclaw-2x7A-Motor-Controller_p_55.html</a>
1 X Power Measurement Unit INA260	<a href="https://www.adafruit.com/product/4226?clcid=CiwKCAiw3djqgBtBNtEWATPrYalOm_zc9GhzRH1NWN4bSRwDg84sihSh3lEqjOrivQ3l_kRBscTAyhOCB0QQAvD_BwE">https://www.adafruit.com/product/4226?clcid=CiwKCAiw3djqgBtBNtEWATPrYalOm_zc9GhzRH1NWN4bSRwDg84sihSh3lEqjOrivQ3l_kRBscTAyhOCB0QQAvD_BwE</a>
1 X PCA9685 corner motor driver	<a href="https://www.mouser.com/ProductDetail/Adafruit/Adafruit_Grayscale_9179Q%3D%3D">https://www.mouser.com/ProductDetail/Adafruit/Adafruit_Grayscale_9179Q%3D%3D</a>
DC power meter	<a href="https://www.amazon.com/gp/product/B017FSED91/">https://www.amazon.com/gp/product/B017FSED91/</a>
batteries	<a href="https://www.amazon.com/gp/stores/page/25B7018D-26C0-4E43-Bc6D-EFAF737D8F5E?ingress=2&amp;visitId=1b4115fa-2240-424d-a0e6-a4668320819ce&amp;ref_=ast_bln">https://www.amazon.com/gp/stores/page/25B7018D-26C0-4E43-Bc6D-EFAF737D8F5E?ingress=2&amp;visitId=1b4115fa-2240-424d-a0e6-a4668320819ce&amp;ref_=ast_bln</a>
charger for the battery	<a href="https://www.amazon.com/Hobby-Fans-Professional-Balance-Discharger/dp/B09XC91BWJ/ref=sr_1_6?eyeword=Lipo+charger&amp;qid=169084744&amp;sr=8-6">https://www.amazon.com/Hobby-Fans-Professional-Balance-Discharger/dp/B09XC91BWJ/ref=sr_1_6?eyeword=Lipo+charger&amp;qid=169084744&amp;sr=8-6</a>
Lipo safe bag	<a href="https://www.robocore.net/bateria/lipo-sack?gad_source=1&amp;clcid=CiwKCAiA982WnBhAYEiwA2WvhOqx0gNRppqeyFheqGLWE5NfgBbbLzI_aAE TNxVh63era4bzDBOZn0Cl-wQAvD_BwE">https://www.robocore.net/bateria/lipo-sack?gad_source=1&amp;clcid=CiwKCAiA982WnBhAYEiwA2WvhOqx0gNRppqeyFheqGLWE5NfgBbbLzI_aAE TNxVh63era4bzDBOZn0Cl-wQAvD_BwE</a>
Raspberry Pi 4 model B 8gb	<a href="https://www.makerhero.com/produto/raspberry-pi-4-model-b/">https://www.makerhero.com/produto/raspberry-pi-4-model-b/</a>

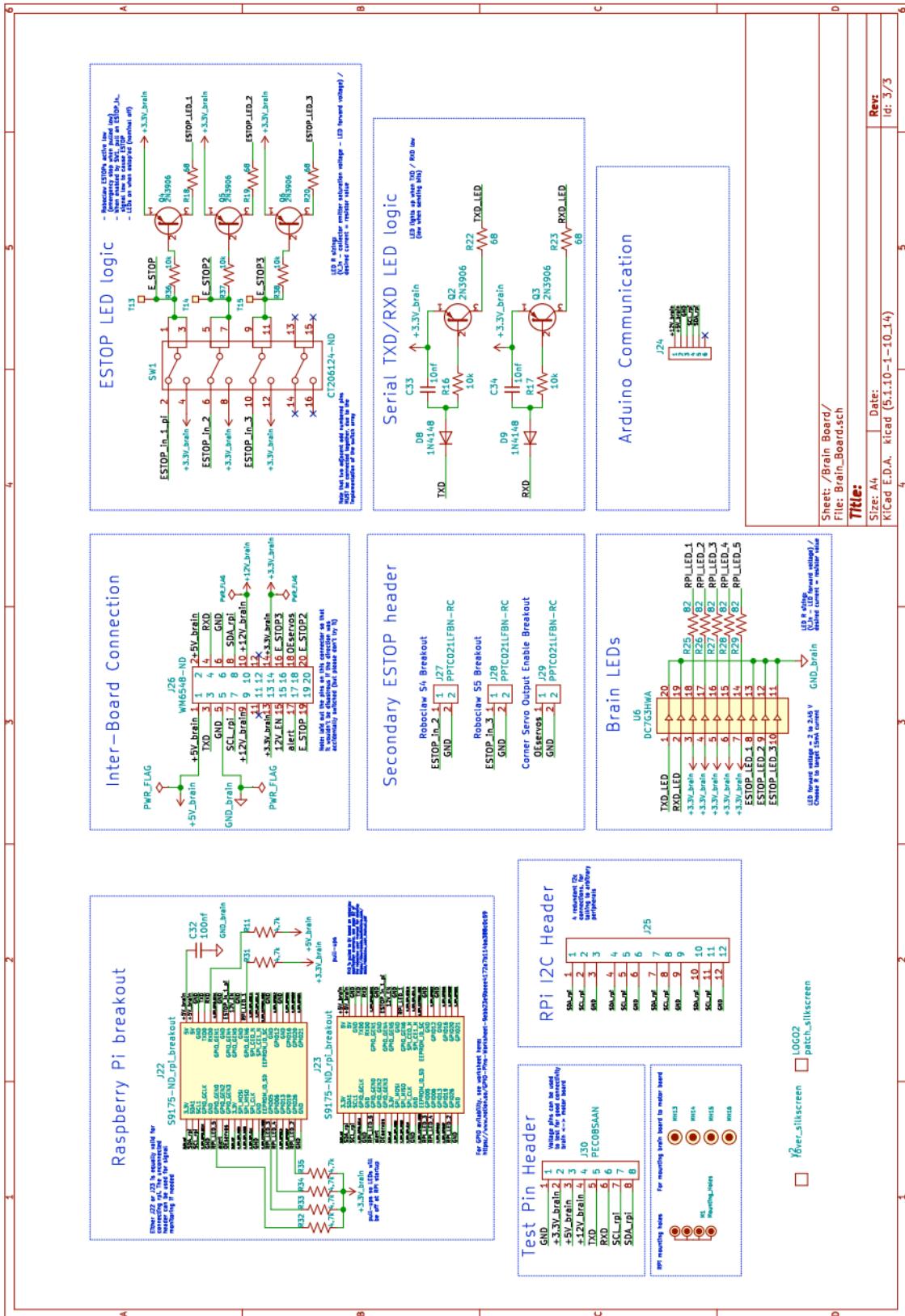
Planilha com lista de componentes: [Lista de componentes.xlsx](#)

## Apêndice B

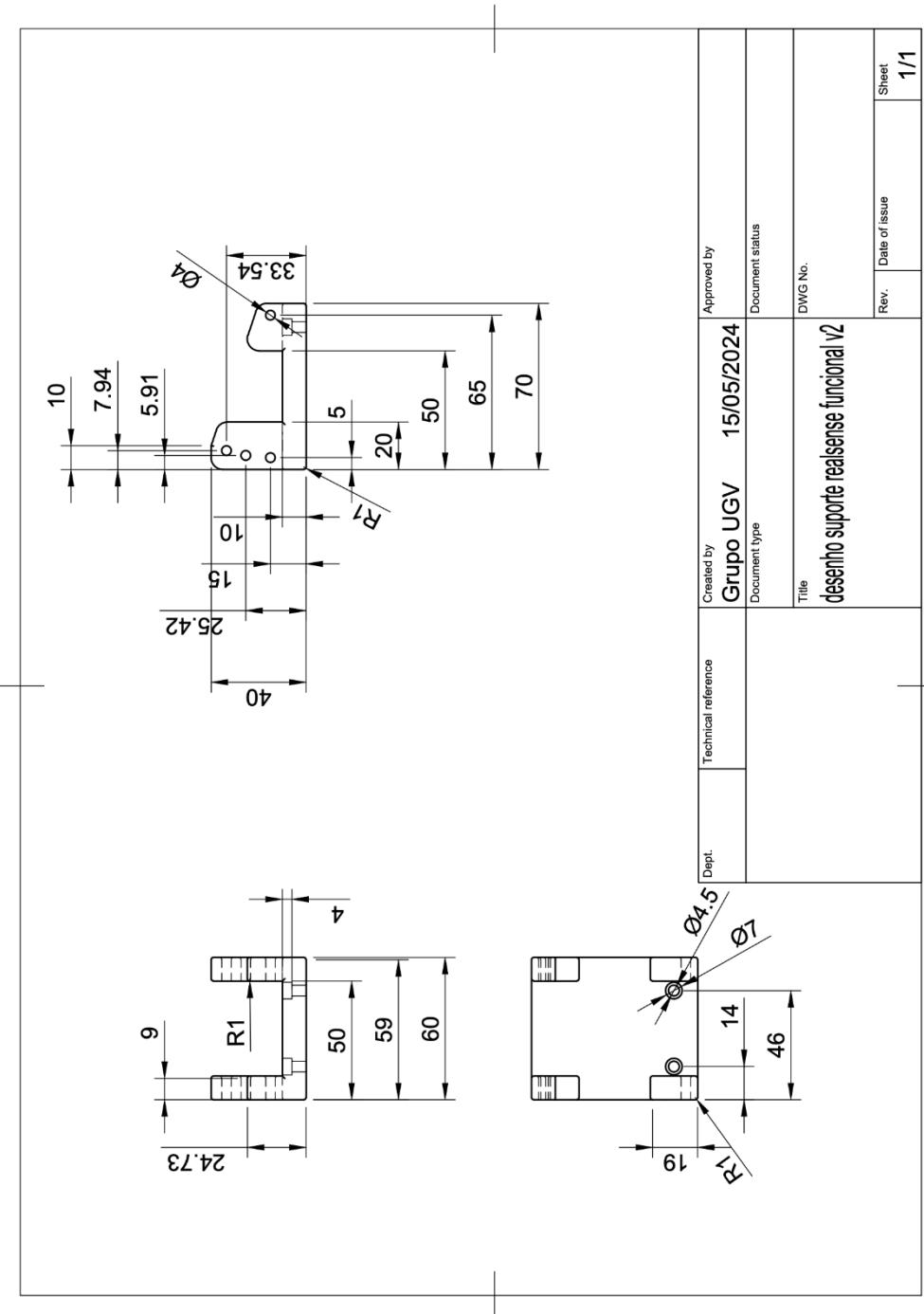
ID do risco	Descrição do risco	Impacto	Índice de gravidade do impacto (G)				Probabilidade de ocorrência do risco (O)	Avaliação da prioridade do risco (P=Média x G x O)	Medida de Prevenção (resposta ao risco)	Responsável
			Escopo	Qualidade	Tempo	Custo				
			Moderado	Baixo	Muito Baixo	0,1	Muito Baixa	0,1	Breno	
1	Pecas encontradas defetivas	Atraso na finalização da parte mecânica do projeto, atrasando as demandas à áreas	Moderado	Baixo	Muito Alto	0,5	Muito Baixa	0,02	Comprar substituta, usar na inspeção ou arrumar o defeito	Breno
2	Pecas encontradas à menos	Atraso na finalização da parte mecânica do projeto, atrasando as demandas à áreas	Moderado	Baixo	0,8	0,4	Média	0,40	Comprar substituta, usar na inspeção ou arrumar o defeito	Breno
3	Pecas contadas incorretamente	Atraso na finalização da parte mecânica do projeto, atrasando as demandas à áreas	Moderado	Baixo	0,4	0,2	Média	0,20	Comprar substituta, usar na inspeção ou arrumar o defeito	Breno
4	Pecas rosqueadas ou furadas incorretamente	Atraso na finalização da parte mecânica do projeto, atrasando as demandas à áreas	Moderado	Baixo	Alto	Moderado	Média	0,20	Comprar substituta, usar na inspeção ou arrumar o defeito	Breno
5	Quebra de alguma estrutura durante testes em campo	Atraso pela necessidade de reconstrução e possibilidade de encontrar peças necessárias a seguir desenvolver a programação	Moderado	Muito Alto	Alto	0,8	Baixa	0,24	Comprar substituta, usar na inspeção ou arrumar o defeito	Breno
6	Demora na entrega da placa	Atraso para finalizar a eletrônica do projeto e não conseguir desenvolver a programação	Baixo	Baixo	Moderado	0,1	Baixa	0,18	Adiar outras tarefas do projeto, passar mais tempo na capacitação em RÓS para evitar futuros atrasos com a programação	Giulia
7	Queima de componentes eletrônicos	Atraso para finalizar a eletrônica do projeto e não conseguir desenvolver a programação	Muito Baixo	Muito Baixo	Moderado	0,05	Moderado	0,10	Pedir componentes e trabalhar em outras áreas do projeto.	Giulia
8	Componentes eletrônicos não entregues	Atraso para finalizar a eletrônica do projeto e não conseguir desenvolver a programação	Muito Baixo	Muito Baixo	Moderado	0,05	Moderado	0,10	Pedir componentes e trabalhar em outras áreas do projeto.	Giulia
9	Placa não entrega	Grande atraso para finalizar a eletrônica do projeto e não conseguir desenvolver a programação	Alto	Moderado	Muito Alto	0,2	Baixa	0,06	Comprar placas novas	Giulia
10	Queima de motores das rodas que não possuem reteava	Necessidade de encontrar motores novos, atrasando a conclusão da etapona e programação	Baixo	0,4	Alto	Moderado	Baixa	0,12	Comprar motores ou conseguir motores similares no insper ou na empresa	Giulia
11	Dificuldades na instalação de Linux e ROS na Raspberry Pi	Atraso na finalização da programação e conclusão do projeto	Baixo	Moderado	Muito Baixo	0,1	Moderado	0,10	Pedir auxílio aos técnicos do laboratório de robótica, pesquisar materiais e cursos on-line	Fernando
12	Dificuldades com a comunicação do raspberry pi, controle com a Raspberry Pi	Atraso na finalização da programação e conclusão do projeto	Moderado	Moderado	Muito Baixo	0,2	Moderado	0,10	Pedir auxílio aos técnicos do laboratório de robótica, pesquisar materiais e cursos on-line	Fernando
13	Bugs de código	Atraso na finalização da programação e conclusão do projeto	Baixo	Baixo	Muito Baixo	0,1	Moderado	0,18	Pedir auxílio aos técnicos do laboratório de robótica, pesquisar materiais e cursos on-line	Fernando
14	Laboratórios lados e com disponibilidade de uso equipamentos	Dificuldades para acesso aos técnicos e auxilio na resolução das problemáticas. Atraso no cronograma	Baixo	Baixo	Moderado	0,1	Alta	0,14	Reservar máquinas com antecedência e dar preferência a horários mais vazios	Bruno
15	Atraso para conclusão do projeto e sobreprazo de outros membros	Atraso para conclusão do projeto e sobreprazo de outros membros	Muito Baixo	Moderado	Alto	0,4	Muito Baixa	0,36	Conversar com integrantes e em último caso com o orientador	Giulia
16	Eros no material disponível do Github	Atraso na montagem do protótipo	Alto	Moderado	Alto	0,2	Alta	0,28	Validar as informações com antecedência e comparar o CAD com foto real do robô e auxiliar aos técnicos	Fernando
17	Problemas de comunicação com a empresa	Atraso para validação de diferentes etapas do projeto	Alto	Moderado	Muito Baixo	0,4	Muito Alta	0,12	Marcar reuniões com antecedência e falar com os problemas para que a empresa corrija o problema	Bruno

## Apêndice C

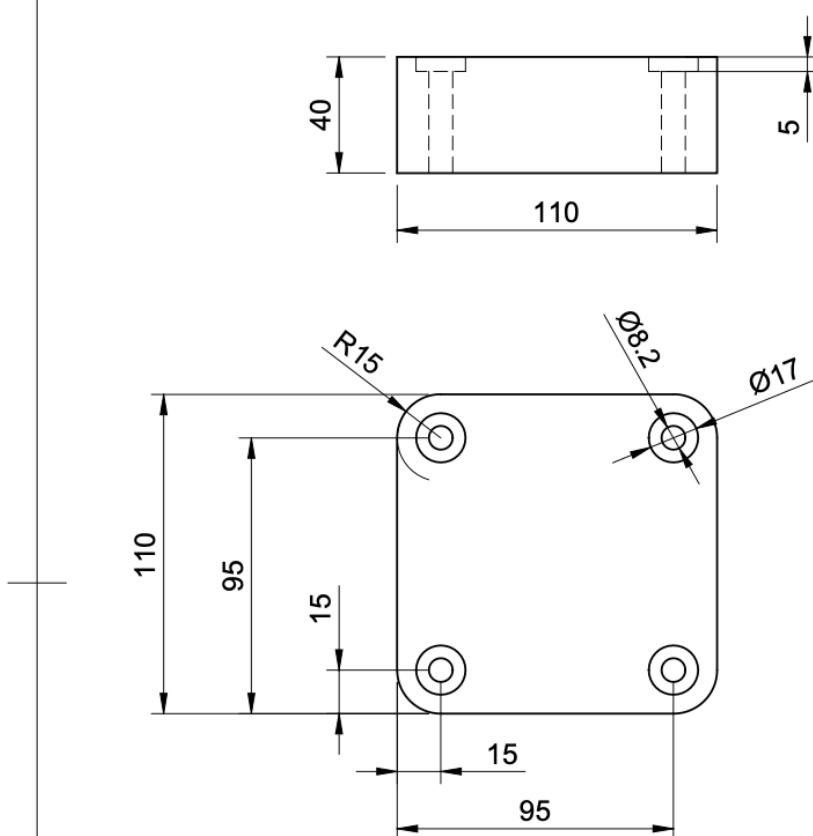




Fonte: NASA. Open Source Rover. Disponível em: [https://github.com/nasa-jpl/open-source-rover/blob/master/electrical/pcb/control\\_board/documentation/v2.0.1/schematics.pdf](https://github.com/nasa-jpl/open-source-rover/blob/master/electrical/pcb/control_board/documentation/v2.0.1/schematics.pdf). Acesso em: 27 fev. 2024.



## **Apêndice D**



Dept.	Technical reference	Created by <b>Grupo UGV</b> 15/05/2024	Approved by
	Document type	Document status	
	Title <b>suporte_lidar v1</b>	DWG No.	
	Rev.	Date of issue	Sheet <b>1/1</b>

