



UNIVERSIDADE FEDERAL DE UBERLÂNDIA  
FACULDADE DE ENGENHARIA MECANICA



**PROJETO DE FIM DE CURSO**

**MÓDULO ROBÓTICO ACOPLADO A DRONE  
PARA INSTALAÇÃO DE ESFERA DE SINALIZAÇÃO**

**UNIVERSIDADE DE UBERLÂNDIA**  
**Faculdade de Engenharia Mecânica**

Pablo Barbosa dos Santos

Módulo robótico acoplado a drone  
para instalação de esfera de sinalização

Projeto de fim de curso apresentado no curso de graduação em Engenharia Mecatrônica da Universidade Federal de Uberlândia, como parte dos requisitos para a obtenção de título de BACHAREL EM ENGENHARIA MECATRÔNICA

**Área de concentração:** Engenharia Mecatrônica

**Orientador:** Prof. Dr. Rogério Sales Gonçalves



## UNIVERSIDADE FEDERAL DE UBERLÂNDIA

Coordenação do Curso de Graduação em Engenharia Mecatrônica  
Rodovia BR 050, KM 78, Bloco 1D, 2º andar - Bairro Glória, Uberlândia-MG, CEP  
38400-902  
Telefone: (34) 2512-6771 - www.mecanica.ufu.br - mecatronica@mecanica.ufu.br



### ATA DE DEFESA - GRADUAÇÃO

Curso de Graduação em:	Engenharia Mecatrônica				
Defesa de:	FEMEC42100 - Projeto de Fim de Curso II				
Data:	29/11/2021	Hora de início:	13:30	Hora de encerramento:	14:25
Matrícula do Discente:	11711EMT026				
Nome do Discente:	Pablo Barbosa dos Santos				
Título do Trabalho:	Módulo robótico acoplado a drone para instalação de esfera de sinalização				

Reuniu-se de forma remota, através da Plataforma Online Google Meet (<https://meet.google.com/vqb-waaj-jer>), a Banca Examinadora, designada pelo docente orientador, assim composta: Prof. Dr. Carlos Alberto Gallo - FEMEC/UFU; Profa. Dra. Vera Lúcia Donizeti de Sousa Franco - FEMEC/UFU; e Prof. Dr. Rogério Sales Gonçalves - FEMEC/UFU orientador do candidato.

Iniciando os trabalhos, o presidente da mesa, Prof. Dr. Rogério Sales Gonçalves, apresentou a Comissão Examinadora e o candidato, e concedeu ao discente a palavra, para a exposição do seu trabalho. A duração da apresentação do discente e o tempo de arguição e resposta foram conforme as normas do curso.

A seguir o(a) senhor(a) presidente concedeu a palavra, pela ordem sucessivamente, aos(às) examinadores(as), que passaram a arguir o(a) candidato(a). Ultimada a arguição, que se desenvolveu dentro dos termos regimentais, a Banca, em sessão secreta, atribuiu o resultado final, considerando o(a) candidato(a):

Aprovado sem nota.

As demandas complementares observadas pelos examinadores deverão ser satisfeitas no prazo máximo de 30 dias corridos a contar da data da defesa, para dar validade a esta aprovação.

Nada mais havendo a tratar foram encerrados os trabalhos. Foi lavrada a presente ata que após lida e achada conforme foi assinada pela Banca Examinadora.



Documento assinado eletronicamente por **Rogério Sales Gonçalves**,  
**Professor(a) do Magistério Superior**, em 29/11/2021, às 14:23, conforme



horário oficial de Brasília, com fundamento no art. 6º, § 1º, do [Decreto nº 8.539, de 8 de outubro de 2015](#).

---



Documento assinado eletronicamente por **Vera Lucia Donizeti de Sousa Franco, Presidente**, em 29/11/2021, às 14:24, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do [Decreto nº 8.539, de 8 de outubro de 2015](#).

---



Documento assinado eletronicamente por **Carlos Alberto Gallo, Professor(a) do Magistério Superior**, em 29/11/2021, às 14:24, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do [Decreto nº 8.539, de 8 de outubro de 2015](#).

---



A autenticidade deste documento pode ser conferida no site [https://www.sei.ufu.br/sei/controlador\\_externo.php?acao=documento\\_conferir&id\\_orgao\\_acesso\\_externo=0](https://www.sei.ufu.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0), informando o código verificador **3210311** e o código CRC **EB016333**.

---

Este projeto é dedicado à meus pais, amigos e a Faculdade de Engenharia Mecânica.  
Todos foram essenciais na minha formação e crescimento.

## **Agradecimentos**

Agradeço a todos os amigos que me ajudaram a fazer esse projeto possível. Agradeço também a minha família que sempre me apoiou em todas as minhas decisões dentro desse curso maravilhoso.

Agradeço a ANEEL e CELESC pela parceria na realização deste projeto com aportes técnicos e financeiros.

A Matheus Luiz, o qual devo um eterno agradecimento com toda ajuda durante essa jornada.

A minha noiva Alana, por todo apoio dado na conclusão desse processo.

E principalmente a meu professor e orientador Prof. Dr. Rogério Sales Gonçalves pela oportunidade e compreensão.

Pablo Barbosa dos Santos

”A possibilidade de realizarmos um sonho é o que torna a vida interessante.”

Paulo Coelho

---

## **RESUMO**

O presente projeto de graduação apresenta o projeto, desenvolvimento e testes de um módulo robótico acoplado a um drone, para instalação de esferas de sinalização em redes de alta tensão. A estrutura foi montada utilizando perfis estruturais de alumínio e foi desenvolvido todo o sistema de controle do módulo robótico. No projeto foi possível desenvolver a base para um protótipo viável para a aplicação requisitada.

Palavras Chave: Alta Tensão, Robô, Esfera de sinalização, Drone, Controle, Automação, Mecatrônica.

---

## **ABSTRACT**

This graduation project presents the design, development and testing of a robotic module coupled to a drone, for installation of warning spheres in high voltage networks. The structure was assembled using structural aluminum profiles and the entire robotic module control system was developed. In the research it was possible to develop the basis for prototype project viable for the required application.

Keywords: High voltage networks, Robot, Warning sphere, Drone, Control, Automation, Mechatronics.

# SUMÁRIO

<b>1</b>	<b>INTRODUÇÃO</b>	<b>1</b>
1.1	OBJETIVOS . . . . .	1
<b>2</b>	<b>REFERENCIAL TEÓRICO</b>	<b>2</b>
2.1	ESFERAS DE SINALIZAÇÃO . . . . .	2
2.2	ROBÔS ACOPLADOS À PLATAFORMA ELEVATÓRIA . . . . .	5
2.3	ROBÔS QUE SE MOVIMENTAM NOS CABOS . . . . .	7
<b>3</b>	<b>METODOLOGIA</b>	<b>9</b>
3.1	VERSÃO 1 . . . . .	9
3.2	VERSÃO 2 . . . . .	18
3.3	PROCEDIMENTOS DE CONSTRUÇÃO DA PARTE ELETRÔNICA . . . . .	23
3.3.1	RASPBERRY PI . . . . .	23
3.3.2	HELTEC LoRa . . . . .	24
3.3.3	PONTE H DUPLA L298N . . . . .	26
3.3.4	LM2596 CONVERSOR DC-DC STEP DOWN . . . . .	26
3.3.5	U2D2 POWER HUB BOARD SET . . . . .	27
3.3.6	BATERIA LIPO 3S TATTU . . . . .	28
3.4	SISTEMA DE CONTROLE . . . . .	29
3.5	PROCEDIMENTOS DE ADAPTAÇÃO DO DRONE . . . . .	37
<b>4</b>	<b>APRESENTAÇÃO E DISCUSSÃO DOS RESULTADOS</b>	<b>40</b>
<b>5</b>	<b>CONCLUSÃO</b>	<b>43</b>
	REFERÊNCIAS . . . . .	44
	<b>ANEXOS</b>	<b>47</b>

# LISTA DE FIGURAS

2.1	Esfera de sinalização convencional TEREX . . . . .	3
2.2	Esfera de sinalização por corda TEREX . . . . .	4
2.3	Esfera de sinalização por helicóptero TEREX . . . . .	5
2.4	Robô montado em plataforma elevatória . . . . .	6
2.5	conceito do braço robótico assistivo . . . . .	6
2.6	Trabalhadores efetuando reparo em linha viva . . . . .	7
2.7	Prototipo robótico para inspeção linhas de transmissão . . . . .	8
3.1	Perfil estruturado de alumínio 15x15mm . . . . .	10
3.2	Guia lateral em alumínio . . . . .	10
3.3	Parafuso Allen M3 . . . . .	11
3.4	Cantoneira 38x35x15mm . . . . .	11
3.5	Projeto CAD versão 1 . . . . .	11
3.6	Estrutura alumínio versão 1 . . . . .	12
3.7	Motor Robotis XM540-W270-R . . . . .	13
3.8	Micro atuador linear . . . . .	13
3.9	CAD Rolamento LMU8 . . . . .	14
3.10	CAD bocal . . . . .	15
3.11	CAD suporte motor . . . . .	15
3.12	CAD suporte motor . . . . .	15
3.13	CAD conjunto acoplamento . . . . .	16
3.14	Modulo versão 1 finalizado . . . . .	17
3.15	Modulo versão 1 finalizado . . . . .	17
3.16	CAD chapa de acoplamento suporte - drone . . . . .	18
3.17	CAD chapa alumínio para reforço e junção . . . . .	19
3.18	CAD mecanismo sustentação passivo . . . . .	19
3.19	CAD mecanismo acoplamento . . . . .	20
3.20	Eletroímã . . . . .	21
3.21	Modulo versão 2 . . . . .	21
3.22	Modulo versão 2 . . . . .	22
3.23	Placa Raspberry Pi 3b . . . . .	23
3.24	Mapeamento GPIO Raspberry Pi 3b . . . . .	24
3.25	Placa Heltec LoRa V2 . . . . .	25

3.26	Pinout Heltec LoRa V2 . . . . .	25
3.27	Módulo Ponte H L298N. . . . .	26
3.28	Módulo LM2596 Step Down . . . . .	27
3.29	Módulo Power Hub U2D2 . . . . .	28
3.30	Bateria TATTU 3S . . . . .	28
3.31	Diagrama da sequencia de comunicação . . . . .	29
3.32	Aplicativo <i>Arduino bluetooth controller</i> . . . . .	29
3.33	Configuração caracteres . . . . .	30
3.34	Configuração caracteres . . . . .	30
3.35	Mapeamento comandos . . . . .	31
3.36	Fluxograma algoritmo Lora <i>SENDER</i> . . . . .	32
3.37	Fluxograma algoritmo Lora <i>RECIEVER</i> . . . . .	33
3.38	Fluxograma algoritmo Raspberry Pi . . . . .	34
3.39	Caixa de controle com componentes eletrônicos . . . . .	35
3.40	Esquemático das ligações físicas dos componentes . . . . .	36
3.41	Caixa de controle com componentes eletrônicos em funcionamento . . . . .	37
3.42	Esquemático Drone EFT E610 10L Agricultural spraying . . . . .	38
3.43	Drone EFT E610 10L <i>Agricultural spraying</i> . . . . .	39
3.44	Drone EFT E610 10L <i>Agricultural spraying</i> . . . . .	39
4.1	Estrutura física para testes . . . . .	40
4.2	Estrutura física para testes . . . . .	41

## Capítulo 1

# INTRODUÇÃO

A dificuldade de manutenção em redes de alta tensão é evidente independentemente do método utilizado, sendo que em diversas situações há a necessidade de desligar o abastecimento de energia, o que gera prejuízos financeiros ao desabastecer locais com fornecimento de energia elétrica.

Atualmente, para inspeção são utilizados drones com câmeras de alta resolução, o que caracteriza-se como um enorme avanço, já que em passado recente essa inspeção era realizada visualmente, dessa maneira, a depender da distância da linha, o suporte de helicópteros era necessário. Outras alternativas existem e estão em operação, como robôs que se movimentam nos cabos, no entanto, ainda apresentam desvantagens, por exemplo, por se movimentarem com rodas e qualquer imperfeição/obstáculo no cabo impossibilita o avanço [Toussaint et al. 2009].

Nesse contexto, a intenção deste projeto de pesquisa é discutir uma possibilidade de solução ao problema apresentado baseada em drones-robôs como objeto de instalação de esferas de sinalização. Defende-se a aplicação dos drones-robôs a partir da perspectiva de que esses conseguem realizar o serviço sem interromper o fornecimento da rede, além de diminuir os riscos de acidentes com os trabalhadores. Como o módulo robótico é instalado do drone, este é capaz de transpor todos os obstáculos existentes. A aplicação em larga escala de uma tecnologia como essa, geraria mais segurança e uma redução considerável nos custos de manutenção.

O presente projeto está estruturado em cinco partes, a saber: introdução; referencial teórico; metodologia; apresentação e discussão de resultados; e conclusão.

### 1.1 Objetivos

O objetivo geral desse projeto é a construção do projeto mecatrônico de um módulo para instalação de esferas de sinalização. Para atingi-lo, conta-se com o objetivo específico de produção do projeto mecânico do módulo, sistema de controle, fabricação e montagem.

## Capítulo 2

# REFERENCIAL TEÓRICO

Nesta seção, será apresentada uma breve explicação da importância das redes de alta tensão e seus riscos. Além disso, exemplificam-se abordagens anteriores de uso de robôs em linhas de alta tensão.

Com o avanço do desenvolvimento urbano, a demanda por energia elétrica aumentou consideravelmente. Por isso, são utilizados cabos de alta tensão para a transmissão de energia entre a produção e o consumidor final. Somente no Brasil estima-se que até o ano de 2023 deva-se alcançar cerca de 185.500 km de linhas de transmissão [CBIE 2019].

Junto a essa quantidade enorme de linhas de alta tensão, vem os problemas habituais a elas, a saber: como reparar defeitos nos cabos, nos isoladores, limpeza do cabo; e o que nos mais interessa aqui, instalação de esferas de sinalização. Os trabalhadores cuja função é manter essas linhas de transmissão em funcionamento, além de estarem sob o risco constante de acidentes, configuram-se como mão de obra extremamente qualificada, escassa e dispendiosa.

### 2.1 Esferas de sinalização

As esferas de sinalização são esferas cujo objetivo é a sinalização visual para evitar acidentes na rede elétrica. Durante o dia, elas servem de alerta a helicópteros, aviões, balões, etc. Elas colaboram para evitar a colisão dos mesmos com os fios de alta tensão. Para isso, as esferas devem seguir a norma ABNT NBR 6535, que especifica como ela deve ser construída e instalada, como a esfera apresentada na figura 2.1. Dentre suas características está a alta durabilidade, a resistência às oscilações, a resistência ao vento e o sistema de drenagem fluvial [Ritz 2021a].

Existem 3 versões dessas esferas que são comumente utilizadas:

- Esfera Convencional

Para esse tipo de esfera, a instalação é manualmente, no solo, na construção ou manutenção da linha de transmissão.

Figura 2.1: Esfera de sinalização convencional TEREX



Fonte: [Ritz 2021b], 2021

- Esfera para instalação por corda ou robô

Esse tipo de esfera foi desenvolvida para dar maior produtividade a instalação, sendo que ela permite a instalação em linha viva diretamente do solo, através da utilização do kit para operação com corda.

Figura 2.2: Esfera de sinalização por corda TEREX



© 2015 Terex Corporation. Todos os direitos reservados.

Fonte: [Ritz 2021b], 2021

- Esfera para instalação por helicóptero

Estas esferas possuem um mecanismo especial para abertura e acoplamento ao cabo, acionado a partir de um único parafuso olhal localizado na parte superior, o qual é operado a partir de um bastão de manobra para cabos de alta tensão. É também dotada de um contrapeso para manter o parafuso voltado para a parte superior. A instalação no cabo terra (cabo guia) é realizada diretamente do helicóptero.

Figura 2.3: Esfera de sinalização por helicóptero TEREX



Fonte: [Ritz 2021b], 2021

Durante esse projeto foi utilizada uma esfera como da figura 2.3, pois por meio de seu mecanismo era possível propor um procedimento de automatização.

Para reduzir risco a vida e o custo pesquisam-se maneiras de reduzir a mão de obra humana nesses serviços por meio da utilização de ferramentas automatizadas [Gonçalves and Carvalho 2013]. Essas máquinas, além de reduzir a mão de obra humana, e minimizar os riscos a saúde do operador, também têm a intenção de serem mais eficientes. Simplificadamente, pode-se dizer que existem duas categorias de robôs, a saber: os acoplados aos caminhões com plataforma elevatória; e os que se movimentam nos fios.

## 2.2 Robôs acoplados à plataforma elevatória

Esse tipo de sistema utiliza plataformas de elevação, que em sua extremidade, estão instaladas um manipulador. O grande problema desse sistema é que ele não pode ser usado em locais como vales, regiões montanhosas e linhas muito altas, pois o braço do guindaste é limitado em comprimento e em regiões que pode trafegar, como pode ser observador na figura 2.4.

A tecnologia presente nesse tipo de equipamento é protegida por inúmeras patentes sendo essas patentes espalhadas por todo o mundo e com finalidades variadas.

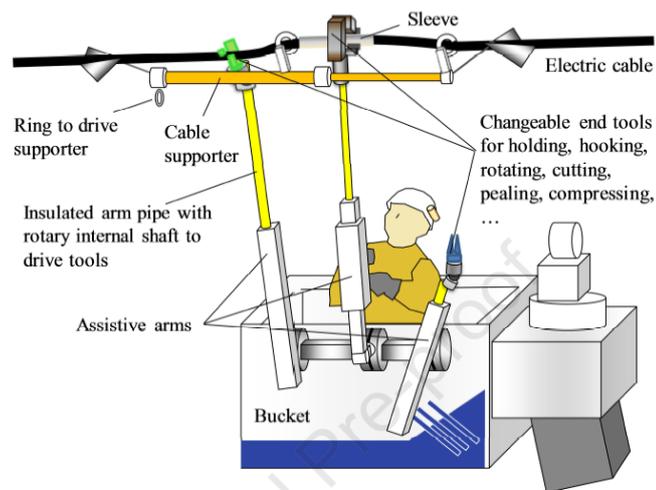
Figura 2.4: Robô montado em plataforma elevatória



Fonte: [Elizondo et al. 2010]

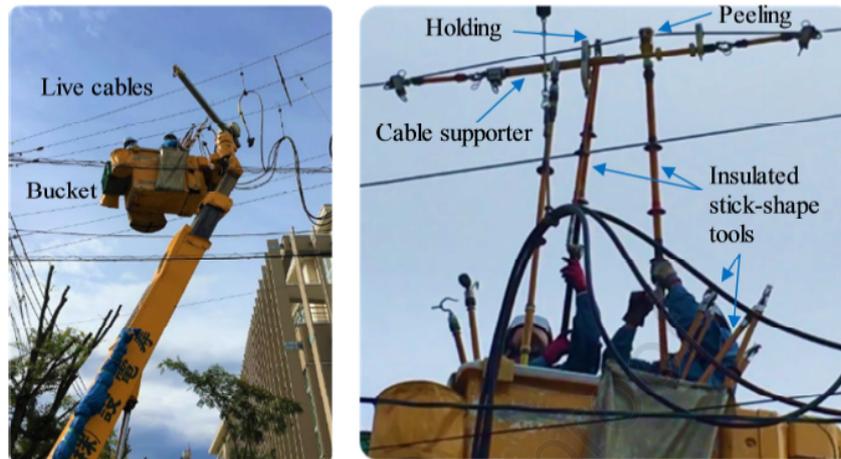
O robô demonstrado nas figuras 2.5 e 2.6, tem sua funcionalidade principal, em reparar cabos de alta tensão por meio de corte/conexão. O módulo robótico mantém o posicionamento dos fios por meio de garras, enquanto o operador repara o fio.

Figura 2.5: conceito do braço robótico assistivo



Fonte: [Seki et al. 2020]

Figura 2.6: Trabalhadores efetuando reparo em linha viva



Fonte: [Seki et al. 2020]

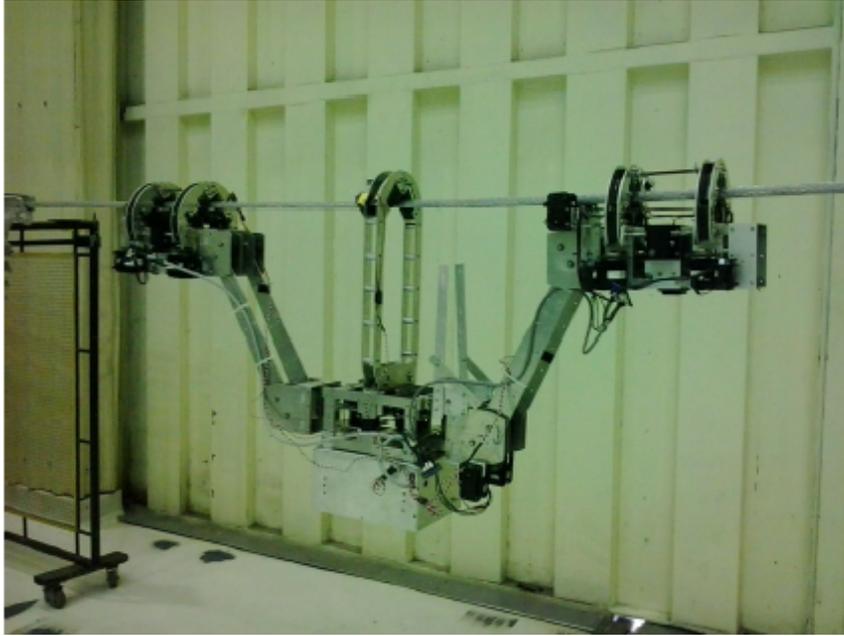
### 2.3 Robôs que se movimentam nos cabos

Nesse formato, os robôs são posicionados sobre o cabo por técnicos e percorrem o comprimento do fio até o ponto desejado. Esse tipo de robô, vem composto de um sistema capaz de o manter no fio, como garras ou rodas. Estes robôs tem a dificuldade em transpor obstáculos como amortecedores, isoladores, esferas de sinalização, etc.

Existem inúmeras versões para esses tipos de robôs, que variam em estabilidade (capacidade de se mover sem cair do fio, com segurança, e principalmente sem movimentos bruscos), controle e mobilidade (capacidade de transpor alguns obstáculos).

Um exemplo de robô é o desenvolvido por estudantes da UFMG em 2014, mostrado na figura 2.7 [LIMA and BOMFIM 2014] para inspeção de linhas de transmissão, o qual já permitia a transposição de alguns objetos previamente detectados por sensores presentes no próprio módulo.

Figura 2.7: Prototipo robótico para inspeção linhas de transmissão



Fonte: [LIMA and BOMFIM 2014]

A solução desenvolvida neste projeto, não encaixa em nenhum dos tipos descritos acima. Posto que é um módulo acoplado a um drone, portanto, não possui as limitações geográficas do primeiro e não precisa transpor objetos, pois será posicionado para instalação já no local correto não havendo a necessidade de percorrer o cabo. Outro fator importante é que o técnico responsável pela operação, estará sempre em local seguro, longe dos fios de alta tensão, evitando assim o risco de acidente humano.

## Capítulo 3

# METODOLOGIA

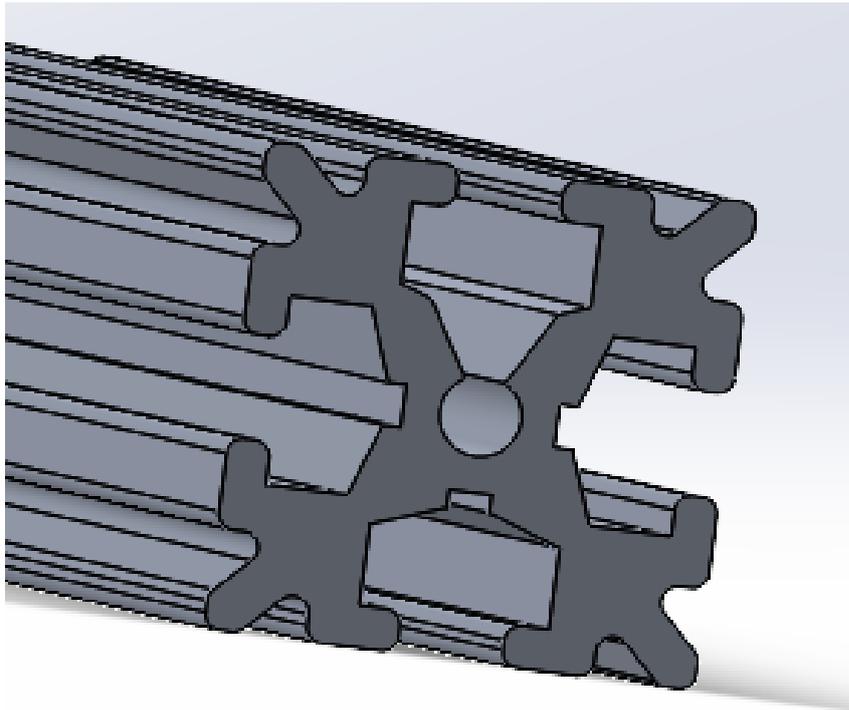
As etapas metodológicas aplicadas para o desenvolvimento deste projeto foram:

1. construção do Projeto CAD/CAE: nesta etapa foi feito o projeto CAD do módulo robótico, considerando a função da esfera a ser utilizada, e para a realização da simulação do processo de instalação;
2. confecção do Sistema de Controle: nesta etapa foi realizada a confecção do sistema de controle para acionamento do módulo;
3. fabricação do módulo: nesta etapa foi produzido o detalhamento do projeto e fabricação das peças;
4. montagem e testes: após a fabricação do módulo foi realizada a montagem do módulo e realizados testes experimentais e os ajustes necessários.

### 3.1 Versão 1

Desde os primeiros rascunhos a ideia era realizar a instalação da esfera com o módulo entrando lateralmente no cabo, na época, compreendia-se que seria a melhor opção. Com base nessa ideia foi pensado um suporte para acomodar a esfera da melhor maneira possível e de forma que facilitasse, também, a instalação da mesma, pois a partir do controlador do drone supunha-se que seria possível. Para isso, foi necessário definir qual a estrutura a ser utilizada para acomodar fisicamente todo o sistema. Essa primeira estrutura contou apenas com metais, sendo que boa parte da estrutura foi feita com perfis de alumínio estruturado de 15x15mm, como os da figura 3.1.

Figura 3.1: Perfil estruturado de alumínio 15x15mm



Fonte: O autor

Já as guias laterais, foram cortadas com 4mm de espessura cada. Foram elaboradas 4 guias as quais na montagem ficaram unidas por espaçadores fabricados em impressão 3D para diminuir peso.

Essas guias possuem como função o acoplamento do sistema ao cabo.

Figura 3.2: Guia lateral em alumínio



Fonte: O autor

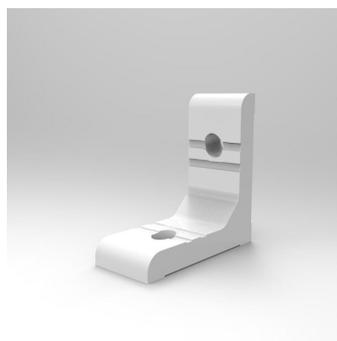
No alumínio estrutural foram realizados cortes para a montagem da estrutura. Toda a montagem foi feita no Laboratório de Automação e Robótica, sendo que as guias laterais foram cortadas/fabricadas externamente. Após ser feito o corte de todos os perfis com os tamanhos necessários, iniciou-se a montagem da estrutura utilizando para a união dos perfis, parafusos, porcas e arruelas M3, assim como as cantoneiras em Alumínio 38x38x15 mm, apresentados nas figuras 3.3 e 3.4.

Figura 3.3: Parafuso Allen M3



Fonte: [LUTEC 2021]

Figura 3.4: Cantoneira  
38x35x15mm



Fonte: [Forseti 2021]

Após a montagem finalizada, obtivemos um resultado similar ao modelo apresentado na figura 3.5, como pode ser verificado na figura 3.6, em seguida.

Figura 3.5: Projeto CAD versão 1



Fonte: O autor

Figura 3.6: Estrutura alumínio versão 1



Fonte: O autor

O próximo passo foi pensar em como se daria o acoplamento da esfera e todo conjunto necessário para o funcionamento correto do conjunto. Essa etapa, é considerada a parte chave deste suporte, uma vez que ela contém todo o conjunto necessário para acoplamento da esfera. Ademais, a parte que ao longo do desenvolvimento se provou a mais trabalhosa devido a sua necessidade de todas as peças encaixarem perfeitamente entre si.

O servo motor utilizado foi XM540-W270-R, escolhido para essa função devido a sua capacidade de torque (até 10,6 N.m), além do fato de sua precisão. Esse servo acoplado a um bocal é responsável pelo fechamento da esfera, pois ela tem um fechamento rosqueado.

Figura 3.7: Motor Robotis XM540-W270-R



Fonte: [ROBOTIS 2021]

Também, utilizou-se um micro atuador linear para elevar e abaixar o motor, como o apresentado na figura 3.8. Este é um dos menores atuadores lineares encontrados no mercado. alimentado com 12V, com curso de 15mm, força de 20N e velocidade de 30mm/s.

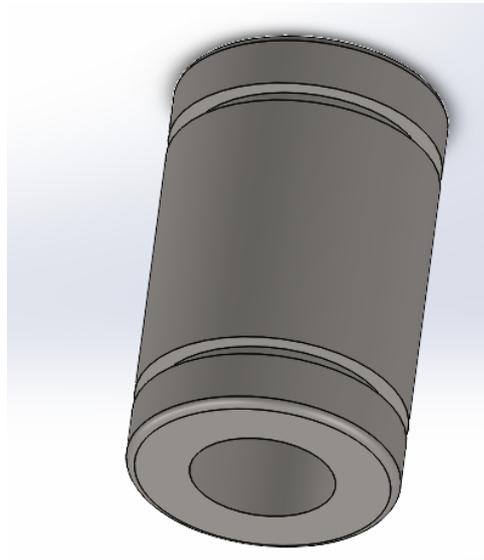
Figura 3.8: Micro atuador linear



Fonte: [VINITRÔNICA 2021]

O conjunto de elevação e rebaixamento construiu-se com ajuda de rolamentos Lmu 8mm, como o da figura 3.9, e eixos de 8mm de diâmetros para fixação. Esses são rolamentos padrão para construção de impressoras 3D.

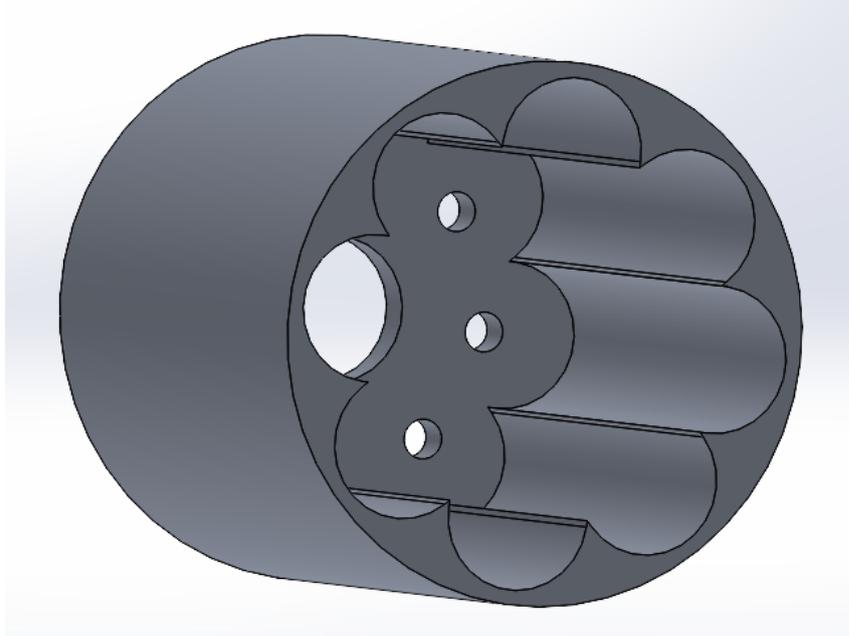
Figura 3.9: CAD Rolamento LMU8



Fonte: [Santos 2021]

A figura 3.10, mostra o bocal desenvolvido especialmente para encaixe no olhal da esfera, sendo que foi feito mais de um rasgo para facilitar o acoplamento dos dois. Esse bocal foi fabricado em impressão 3D.

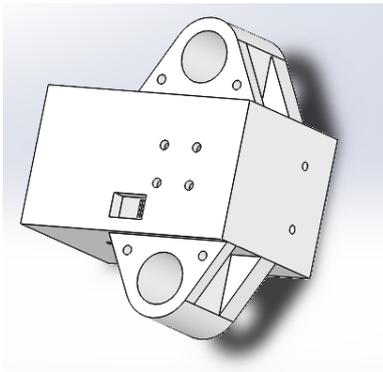
Figura 3.10: CAD bocal



Fonte: O autor

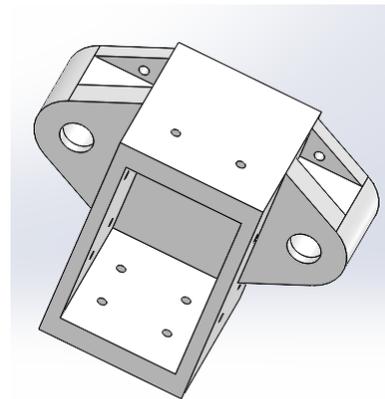
Foram fabricadas diversas peças em impressão 3D, sendo uma das mais importantes o suporte do motor, figuras 3.11 e 3.12, o qual fez a interligação de inúmeras partes do mesmo.

Figura 3.11: CAD suporte motor



Fonte: O autor

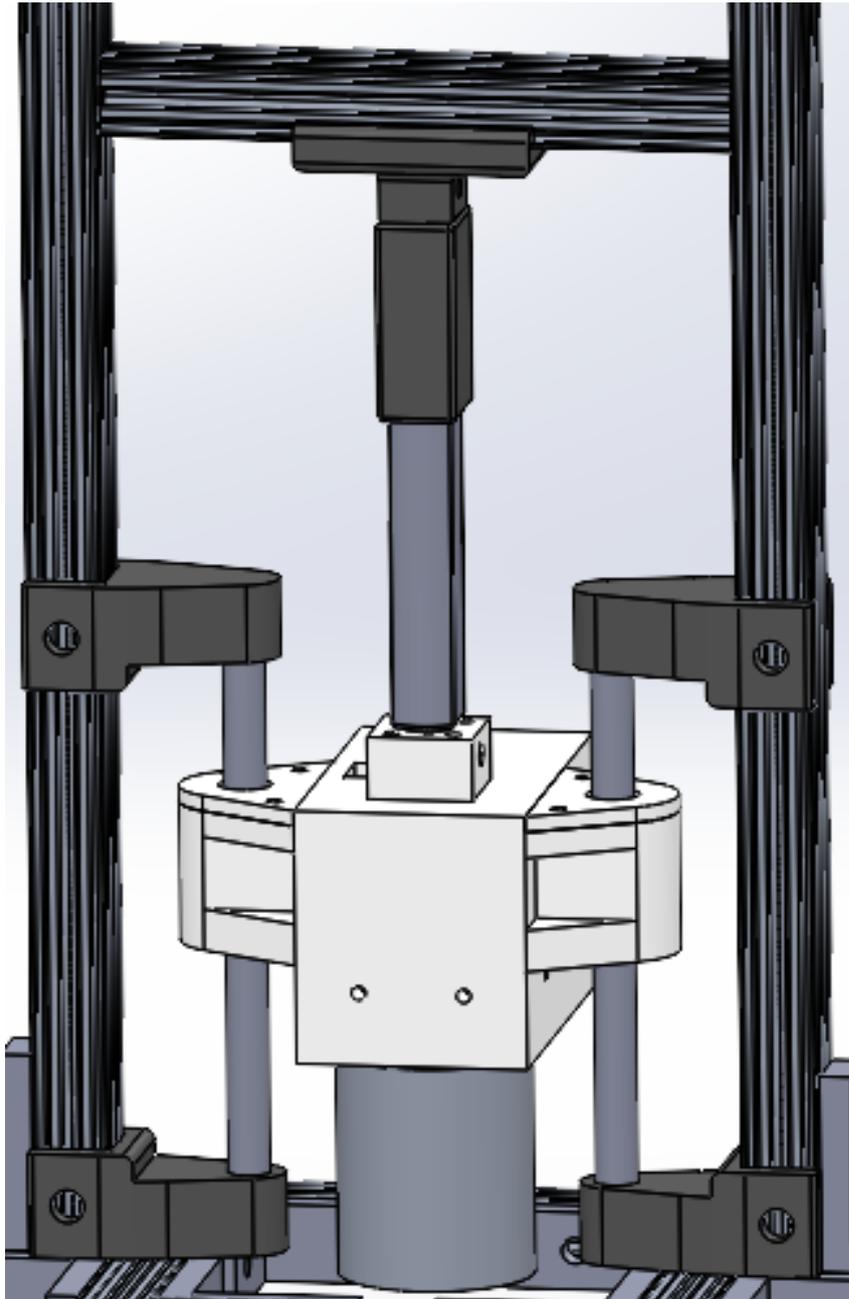
Figura 3.12: CAD suporte motor



Fonte: O autor

O resultado do conjunto como mostra a figura 3.13, provou-se funcional para acoplamento e desacoplamento da esfera.

Figura 3.13: CAD conjunto acoplamento



Fonte: O autor

Figura 3.14: Modulo versão 1 finalizado



Fonte: O autor

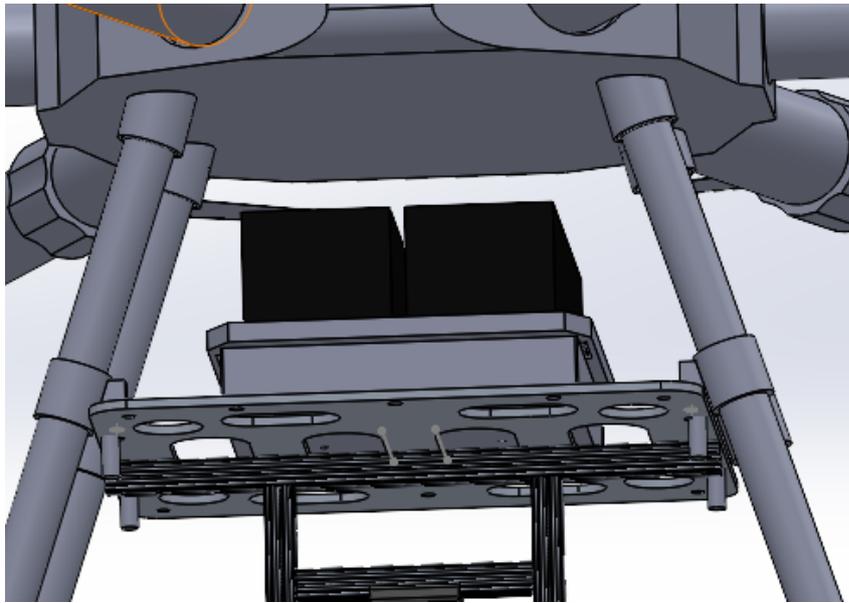
Figura 3.15: Modulo versão 1 finalizado



Fonte: O autor

Essa versão foi construída e testada no chão. Após todo seu funcionamento testado, o que ficou faltando nela foi o acoplamento ao drone, ao qual foi pensado para ser fixo em uma chapa de alumínio, conforme modelo mostrado na figura 3.16.

Figura 3.16: CAD chapa de acoplamento suporte - drone



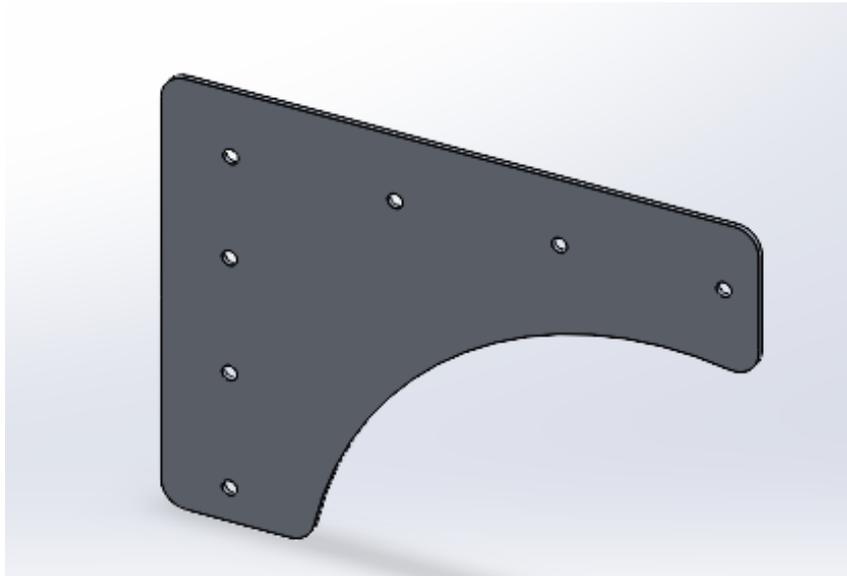
Fonte: O autor

No entanto, preferiu-se não prosseguir com tal versão, pois esse acoplamento mostrou-se falho, além do fato de a estrutura precisar de alguns ajustes para melhor acomodar a esfera. Dessa forma, decidiu-se retroceder alguns passos e reformular algumas peças que eventualmente poderiam ser problemas, dando início assim a preparação da versão 2.

## 3.2 Versão 2

Nessa versão, fez-se um reforço nas junções para assim, evitar possíveis movimentos indesejados devido ao aparafusamento da versão inicial concedendo uma maior resistência à parte superior. Para isso foram fabricadas as chapas apresentadas na figura 3.17.

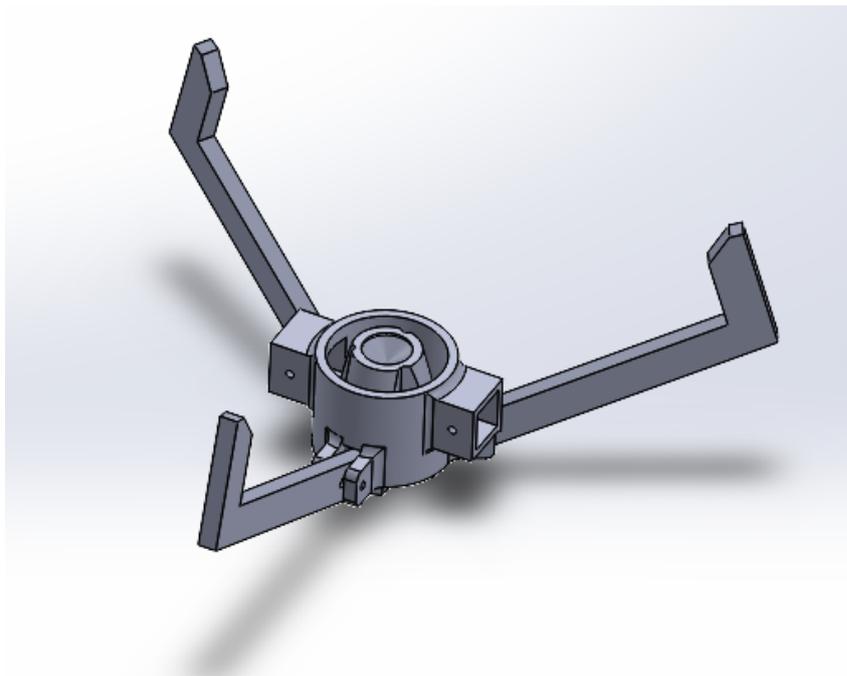
Figura 3.17: CAD chapa alumínio para reforço e junção



Fonte: O autor

Outra modificação importante foi a base que acomodava a esfera, como observa-se na figura 3.18. Na qual se projetou um mecanismo passivo auxiliar, que se movimenta com o peso da esfera, e quando liberada retornaria sua forma original, o que não interferiria na entrada ou saída da esfera.

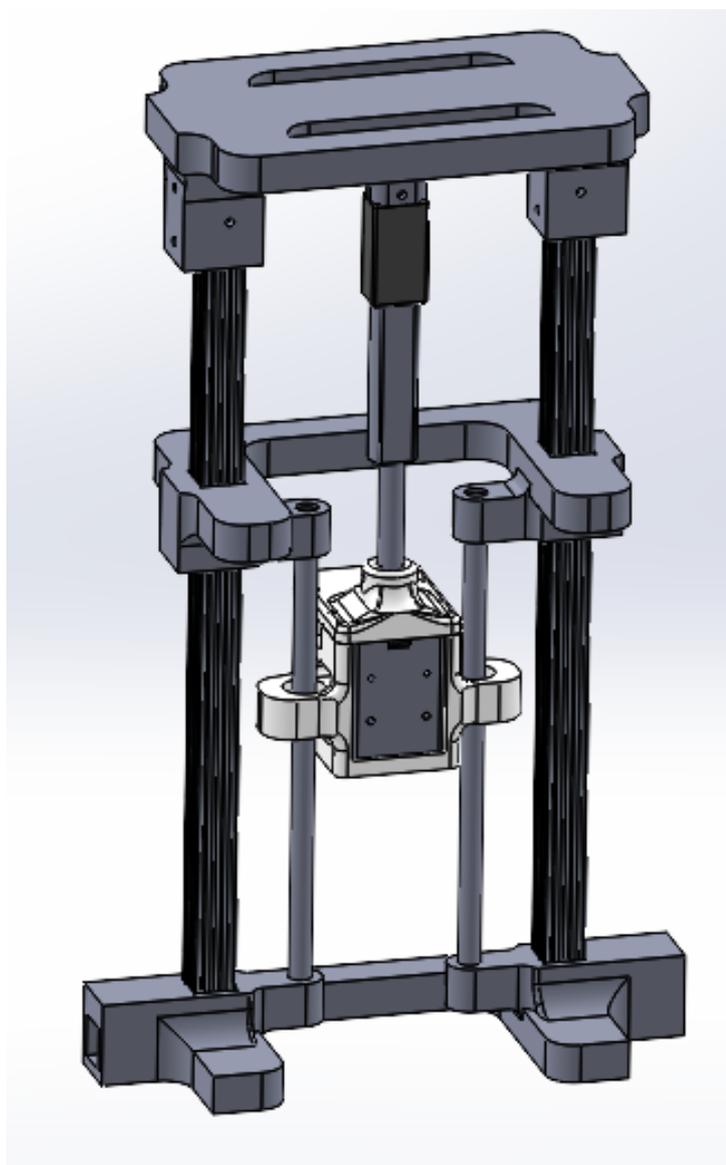
Figura 3.18: CAD mecanismo sustentação passivo



Fonte: O autor

A modificação mais importante diz respeito a uma alteração na parte superior, na qual se eliminou definitivamente o problema que ocorria às vezes referente ao olhal agarrar na saída do bocal. Para isso, seguiu-se a mesma ideia da versão anterior, mas com alteração na junção inferior possibilitando maior abertura. Além disso, realizou-se um reforço na parte superior dos eixos o que permitiu que eles ficassem completamente alinhados, além de deixar a movimentação do motor mais segura.

Figura 3.19: CAD mecanismo acoplamento



Fonte: O autor

Ademais, instalou-se na parte superior um eletroímã para acoplar e desacoplar o suporte ao

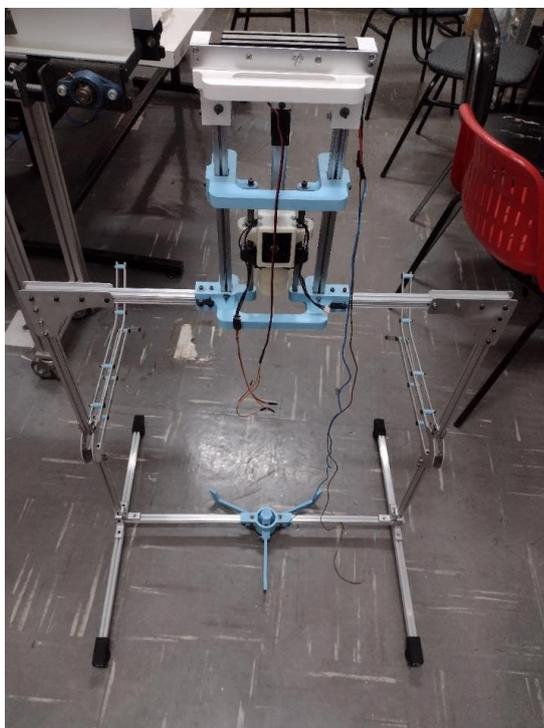
drone, que ofereceu mobilidade, além de permitir em caso de emergência um desacoplamento rápido e seguro. As figuras 3.21 e 3.22, em sequência, apresentam o resultado final.

Figura 3.20: Eletroímã



Fonte: [Projseg 2021]

Figura 3.21: Modulo versão 2



Fonte: O autor

Figura 3.22: Modulo versão 2



Fonte: O autor

### 3.3 Procedimentos de construção da parte eletrônica

#### 3.3.1 Raspberry Pi

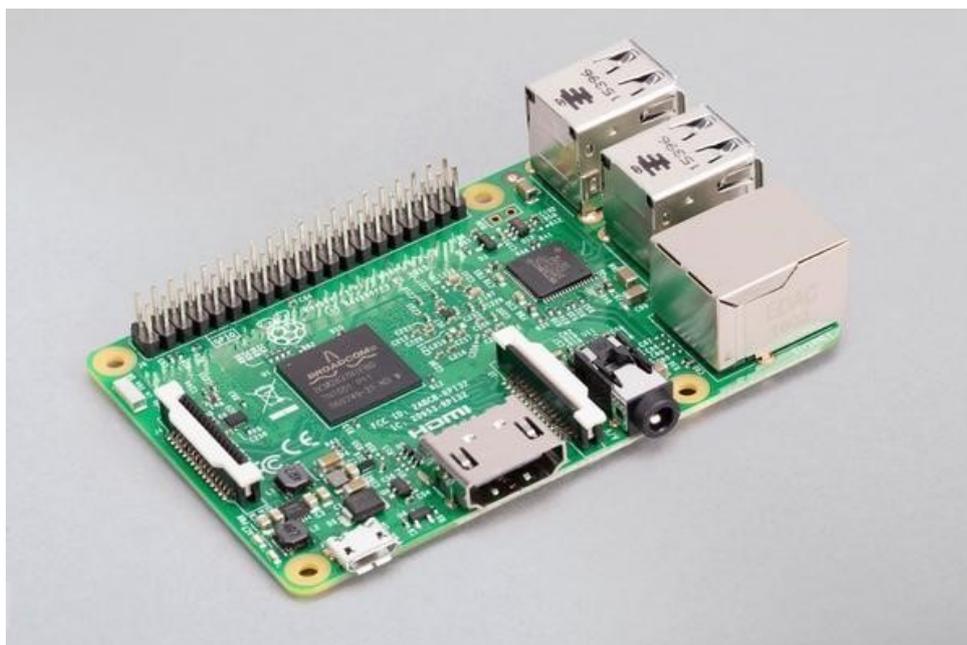
Atualmente, é praticamente impossível pensar qualquer solução que não envolva tecnologia. As soluções baseadas em ferramentas computacionais se provam cada vez mais efetivas, e com o avanço técnico-científico apresentam fácil utilização. Na condução deste projeto, por exemplo, houve uso de aparelhos celulares, uso extenso de computador e o mais importante para a solução um sistema embarcado.

Um sistema embarcado é um sistema controlado por um microcontrolador cujo propósito é atender uma solução específica. Assim, esse tipo de sistema tem a característica de ser de baixo custo, além de apresentar tamanho reduzido. Após pesquisa definiu-se pela utilização do Raspberry Pi 3b como controlador principal em toda a parte eletrônica do projeto, sobre ele discorre-se a seguir.

O Raspberry Pi é largamente utilizado em sistemas embarcados, na verdade, ele é um microcomputador. Isto posto, destaca-se que esse é um dispositivo criado no Reino Unido por uma organização sem fins lucrativos com intuito de estimular o ensino de computação nas escolas com preço acessível [Foundation 2021].

As placas Raspberry Pi, são computadores de baixo custo, que oferecem todas as funções de um computador comum, além de possuírem uma interface GPIO (*General Purpose Input/Output*) que permite a ligação de diversos componentes eletrônicos externos, figura 3.23.

Figura 3.23: Placa Raspberry Pi 3b

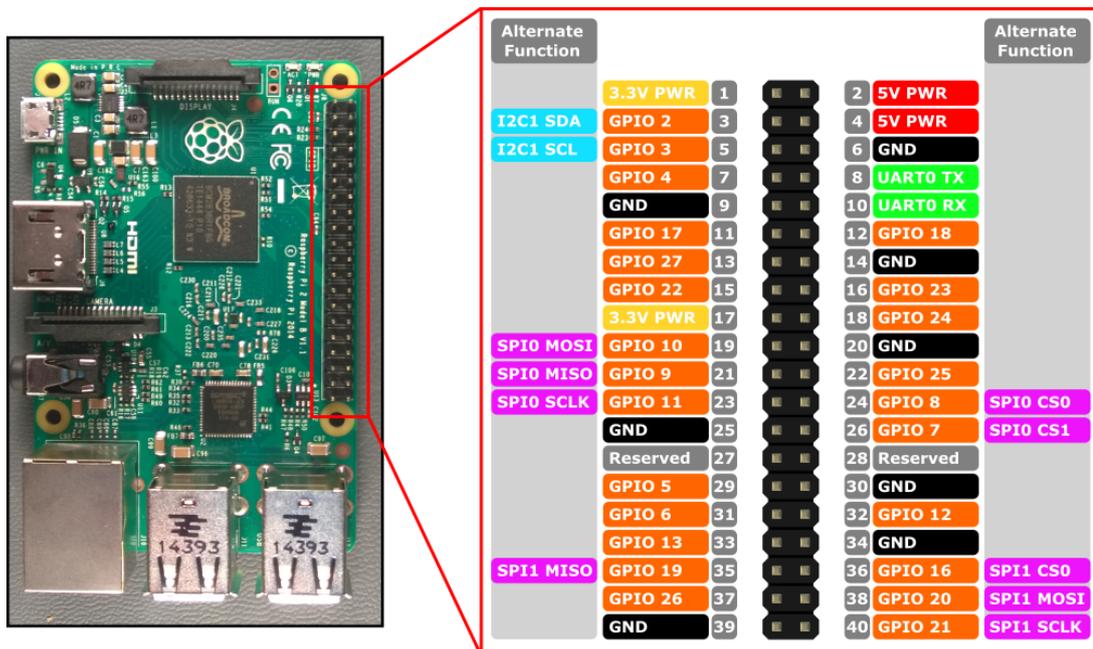


Fonte: [Foundation 2021]

A escolha pela aplicação/Raspberry deu-se por alguns motivos, entre eles, o baixo custo, a facilidade de desenvolvimento de aplicações em linguagem Python, e o fato de ter as entradas GPIO, necessárias para controlar os dispositivos presentes neste projeto como atuador linear e o eletroímã. O Raspberry provou-se a opção mais viável para a movimentação do motor utilizado também, pois possui um protocolo próprio. Essa movimentação nos testes iniciais com Shields próprios do fabricante evidenciaram-se de difícil utilização, posteriormente a pesquisas descobriu-se que era possível controlar o motor via Python, por isso o Raspberry Pi mostrou-se uma opção.

A figura 3.24 apresenta o detalhamento das entradas e saídas da placa Raspberry.

Figura 3.24: Mapeamento GPIO Raspberry Pi 3b



Fonte: [Docs 2021]

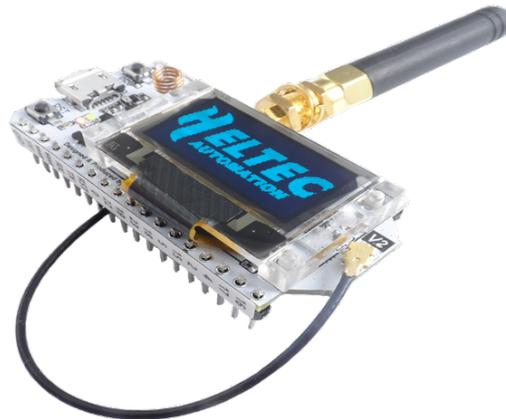
### 3.3.2 Heltec LoRa

Para a comunicação entre a base e o drone era necessário um tipo de conexão que permitisse longas distâncias, que conseguisse passar as mensagens/comandos desejados sem perdas. Após pesquisa, optou-se por utilizar a rede LoRa, que é uma categoria de comunicação sem fio de baixo consumo e longas distâncias. Outras possibilidades foram estudadas como a transmissão bluetooth, wifi-direct e LTE. No entanto, o custo benefício da LoRa, como a distância percorrida e principalmente o fato de enviar os pacotes de dados sem corromper ao destinatário a tornaram ideal.

A LaRaWAN é um protocolo de comunicação para redes LoRa que proporciona comunicação bidirecional entre dispositivos. Desse modo, um emissor de sinal pode enviá-lo a

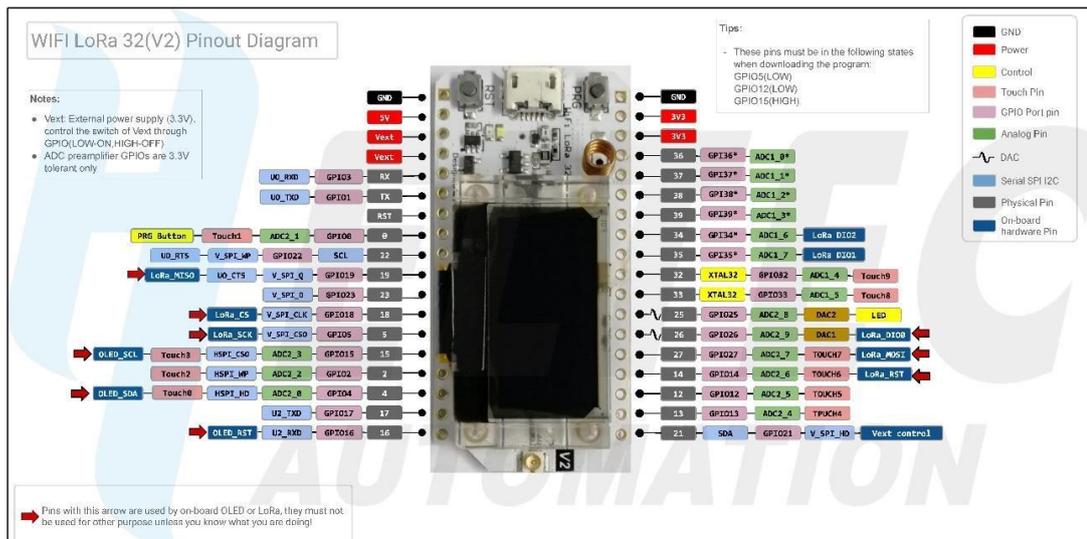
somente um dispositivo ou a qualquer dispositivo na área, por isso o suporte a criptografia no envio é tão importante. Para o projeto, foi escolhida a placa WiFi LoRa 32 (V2) da Heltec Automation (TM)[Heltec 2021], a qual conta com um módulo ESP 32, conexão bluetooth e Wifi, na mesma placa, sendo ser possível utilizar todas as conexões simultaneamente, tornando-a uma excelente opção, essa placa pode ser observada nas figuras 3.25 e 3.26.

Figura 3.25: Placa Heltec LoRa V2



Fonte: [Heltec 2021]

Figura 3.26: Pinout Heltec LoRa V2



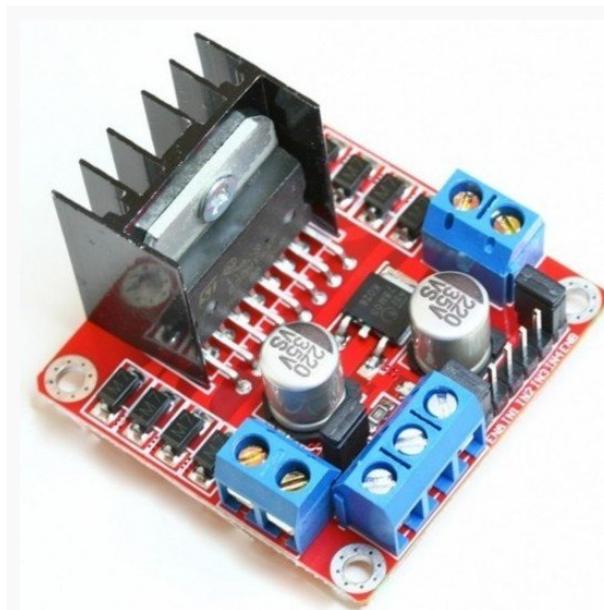
Fonte: [Heltec 2021]

Para o projeto, foram utilizadas duas placas como as da figura 3.25, uma para envio (*sender*) e outra para recebimento (*reciever*) dos dados. Mostrou-se necessário, também, para o devido funcionamento do circuito, a utilização de alguns componentes eletrônicos. A seguir, esses são enumerados.

### 3.3.3 Ponte H Dupla L298N

A Ponte H é um circuito originário da eletrônica de potência do tipo chopper. Portanto, seu uso é necessário na aplicação, pois a corrente transmitida a carga indutiva pode danificar o Raspberry PI. Dessa forma, utilizou-se a Ponte H com o intuito de controlar tanto o atuador linear, quanto o eletroímã na segunda versão. Essa inserção desse componente eletrônico provou-se uma opção barata (menos que R\$10,00) e eficaz de controlar tais equipamentos. A figura 3.27 , apresenta a Ponte H Dupla L298N.

Figura 3.27: Módulo Ponte H L298N.



Fonte: [STMicroelectronics 2021]

### 3.3.4 LM2596 Conversor DC-DC Step Down

A melhor forma de alimentar uma placa Raspberry PI é através da fonte original ligada a rede de energia, evitando quaisquer riscos à placa. Como na solução desenvolvida a placa foi instalada junto ao drone, essa forma de alimentação tornou-se inviável. Mostrou-se necessário a utilização de um regulador de tensão para reduzir a tensão dc de 12v para 5,3v, fundamental para a correta alimentação do Raspberry PI. Esse regulador de tensão tem as seguintes funções,

reduzir e estabilizar a tensão de 12v proveniente da bateria para 5,3v, utilizada para alimentar o Raspberry PI [Instruments 2021].

Dessa forma, o principal requisito desse regulador de tensão é que ele não sofresse flutuações significativas de tensão (acima de 0,3v). Caso a tensão varie ela corre o risco de entrar em modo de segurança o que desligaria todo o sistema. Por isso, esse módulo é de crucial importância, devido à tensão estabilizada na saída. Para tal necessidade foi alocado o módulo regulador de tensão LM2596 Conversor DC-DC Step Down, apresentado na figura 3.28.

O motivo de ser alimentado com 5,3v e não com 5v como seria o ideal, é que ao ligar o Raspberry a tensão tende a reduzir um pouco. Portanto, esse valor foi determinado experimentalmente de modo a manter exatos 5v na entrada mesmo após a placa ligada.

Figura 3.28: Módulo LM2596 Step Down

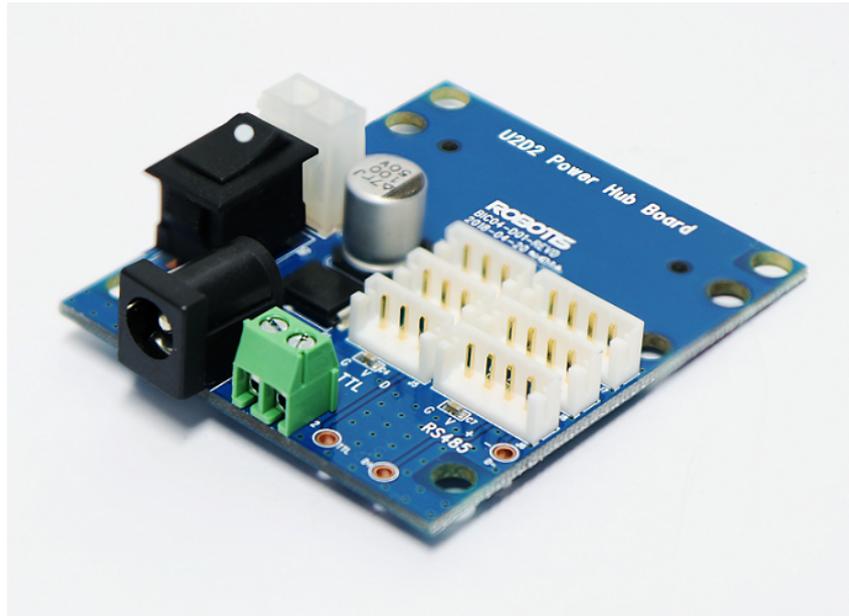


Fonte: [FilipeFlop 2021]

### 3.3.5 U2D2 Power Hub Board Set

O U2D2 *Power Hub Board Set*, mostrado na figura 3.29, é um módulo de propriedade da Robotis, utilizado para alimentar e transmitir informações do Raspberry PI ao motor. O Raspberry se conecta ao U2D2 via porta serial, que controla o motor [robotis 2021].

Figura 3.29: Módulo Power Hub U2D2



Fonte: [robotis 2021]

### 3.3.6 Bateria Lipo 3S TATTU

Para a alimentação de toda a parte eletrônica utilizou-se a Bateria Lipo Gens Ace linha Tattu 5200mAh 3S 11.1V 35C, apresentado na figura 3.30. Esta, a partir dos testes experimentais apresentou confiabilidade e tempo de uso suficiente para a aplicação.

Figura 3.30: Bateria TATTU 3S

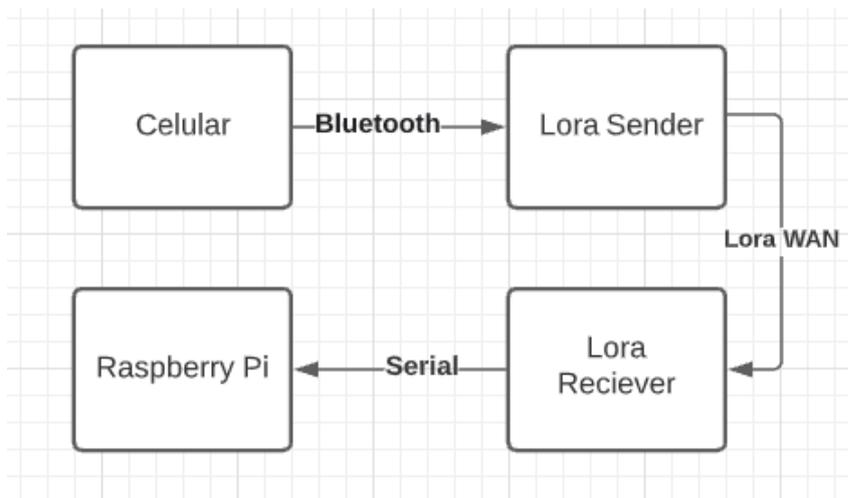


Fonte:[Hobby 2021]

### 3.4 Sistema de controle

A principal meta com a parte eletrônica do projeto era a de conseguir enviar comandos do solo para o drone sem perdas e em tempo real. Dessa forma, o grande problema a ser resolvido é que essa distância entre o operador e o drone poderia chegar até 400m. Foi então desenvolvido um sistema de controle como o descrito, na figura 3.31.

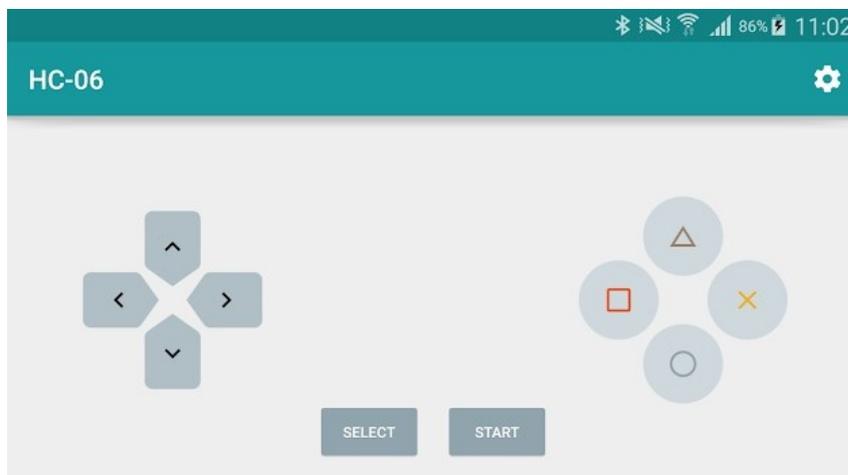
Figura 3.31: Diagrama da sequencia de comunicação



Fonte: O autor

A sequência de comunicação começa com um celular ao qual foi instalado o aplicativo *Arduino bluetooth controller* da empresa Giumig Apps por ser um aplicativo gratuito e de fácil acesso, disponível para download no Google Play.

Figura 3.32: Aplicativo *Arduino bluetooth controller*



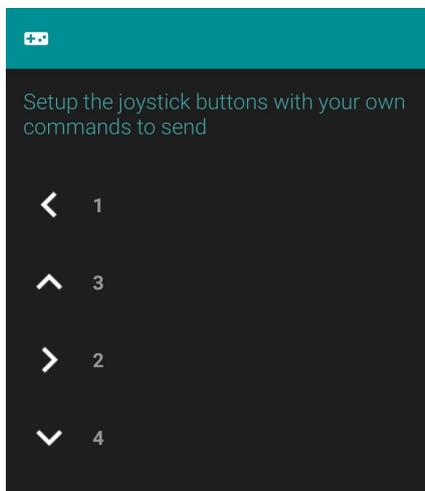
Fonte: O autor

Inicialmente a ideia era a de se desenvolver uma versão própria para o projeto no entanto, em buscas na internet foi encontrado tal aplicativo que supria todas as necessidades o qual economizou tempo de desenvolvimento que foi alocado em outras partes do projeto.

A configuração no formato de joystick como podemos ver na figura 3.32 é simplesmente para facilitar os comandos pois em vez de textos temos botões para enviar os comandos, esses por sua vez são convertidos em caracteres e enviados tudo automaticamente pelo aplicativo.

Antes do primeiro uso, é necessário fazer uma rápida configuração do aplicativo como pode-se ver nas figuras abaixo, assim pode-se definir o que será enviado ao pressionar cada botão.

Figura 3.33: Configuração caracteres



Fonte: O autor

Figura 3.34: Configuração caracteres



Fonte: O autor

Feita a configuração inicial este já está pronto para uso. Basta conectar via bluetooth a uma das placas LoRa (*Sender*). A partir disso, ao pressionar um dos botões é enviado um caractere para a LoRa(*Sender*) que interpreta o sinal recebido.

Na figura 3.35, é possível ver o mapeamento dos botões do aplicativo para os devidos comandos no módulo robótico.

Figura 3.35: Mapeamento comandos

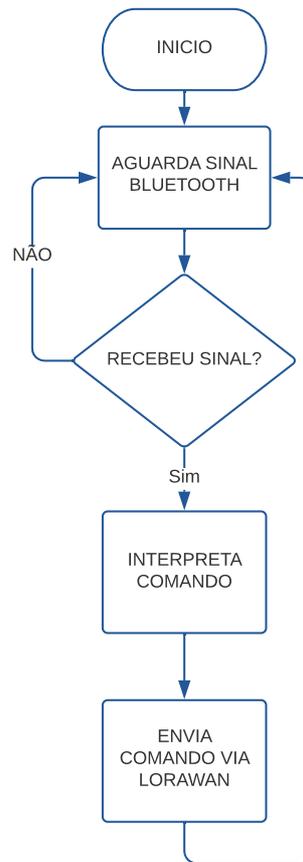
	FECHAMENTO DA ESFERA
	ABERTURA DA ESFERA
	AVANÇO ATUADOR LINEAR
	RECUO ATUADOR LINEAR
	ATIVAR ELETROIMÃ
	DESATIVAR ELETROIMÃ

Fonte: O autor

Recebido os comandos do aplicativo de celular, a LoRa interpreta estes e redireciona a informação via LoRaWan para a segunda LoRa(*reciever*).

A Figura 3.36 mostra um fluxograma da placa Lora *SENDER*

Figura 3 36 Fluxograma algoritmo LoRa *SENDER*

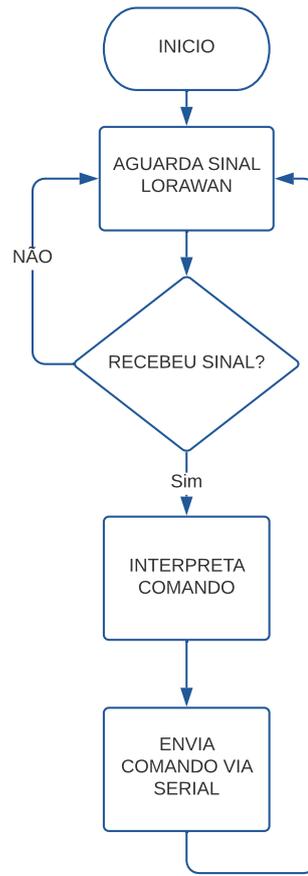


Fonte: O autor

Essa comunicação via LoRa Wan é fundamental pois é possível enviar comandos com enormes distâncias. Em testes experimentais foi conseguido cerca de 500m, no entanto com antenas mais potentes é possível facilmente ultrapassar essa barreira. Outro ponto importante é que essa rede não sofre interferência eletromagnética significativa da rede de transmissão de energia elétrica.

Para confirmar tal fato, foi feito um teste experimental onde enviamos sinais de uma LoRa, e recebidos em outra, através de uma subestação de energia, e conseguimos com sucesso enviar comandos sem perda.

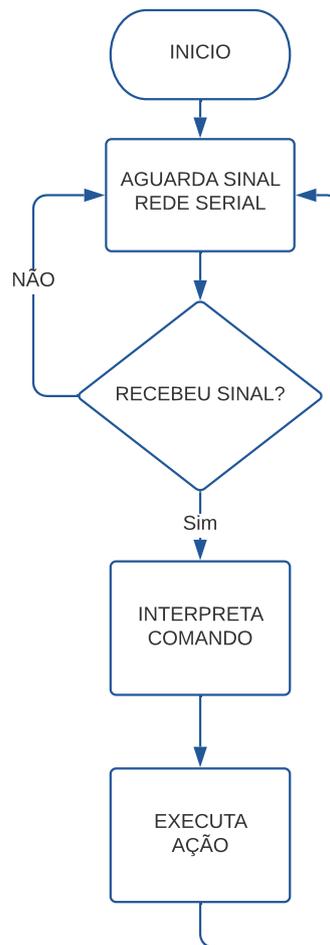
Na terceira etapa da comunicação recebemos os dados da LoRa sender via LoRaWan na LoRa (*receiver*). Essa LoRa por sua vez, tem a função de receber e interpretar os dados recebidos via LoRaWAN, esses dados interpretados são enviados via porta serial, para a última etapa da comunicação, a placa Raspberry Pi 3.37.



Fonte: O autor

A Raspberry PI captura essas informações recebidas via comunicação serial e a partir desses dados controla todos os equipamentos. Assim, o controle de todo o módulo está centralizado nela 3.38.

Figura 3.38: Fluxograma algoritmo Raspberry Pi



Fonte: O autor

Todos os algoritmos utilizados nas placas, estão apresentados na seção Apêndice.

A figura 3.39, apresenta a caixa de controle.

Figura 3.39: Caixa de controle com componentes eletrônicos



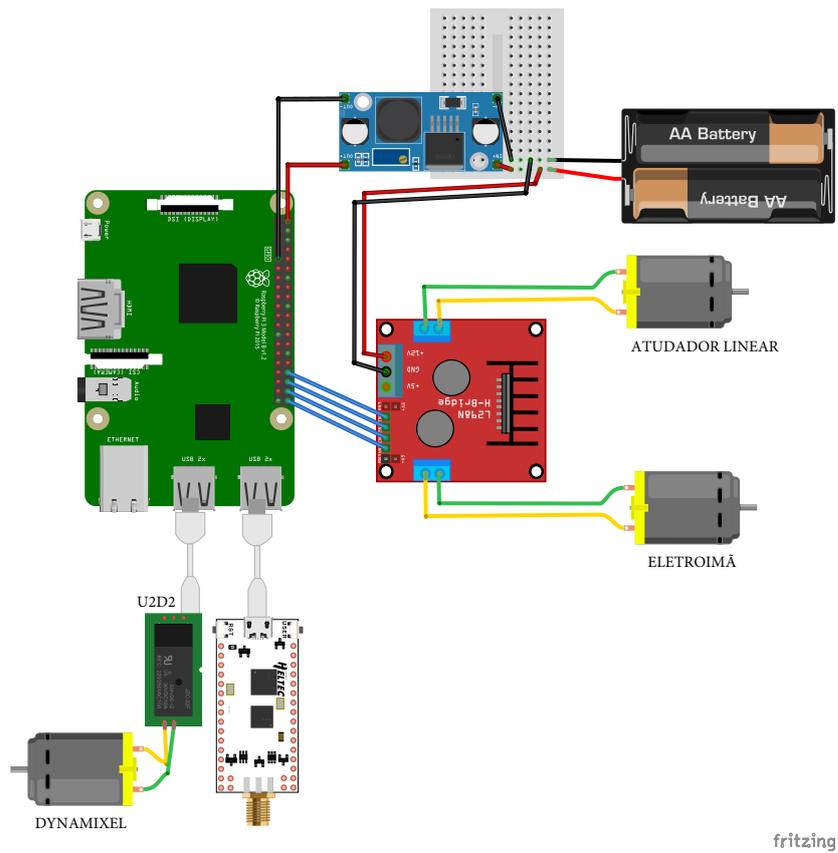
Fonte: O autor

A imagem acima, mostra a caixa de controle e todos seus componentes, a saber:

1. Raspberry PI 3B
2. Regulador Tensão Step Down Buck dc-dc Lm2596
3. Heltec WIFI LoRa 32(V2)
4. U2D2 Power Hub Board Set
5. Módulo Ponte H Dupla L298n

Na figura 3.40, é representado o esquemático das ligações físicas dos componentes, com todos os componentes utilizados no módulo com as devidas ligações.

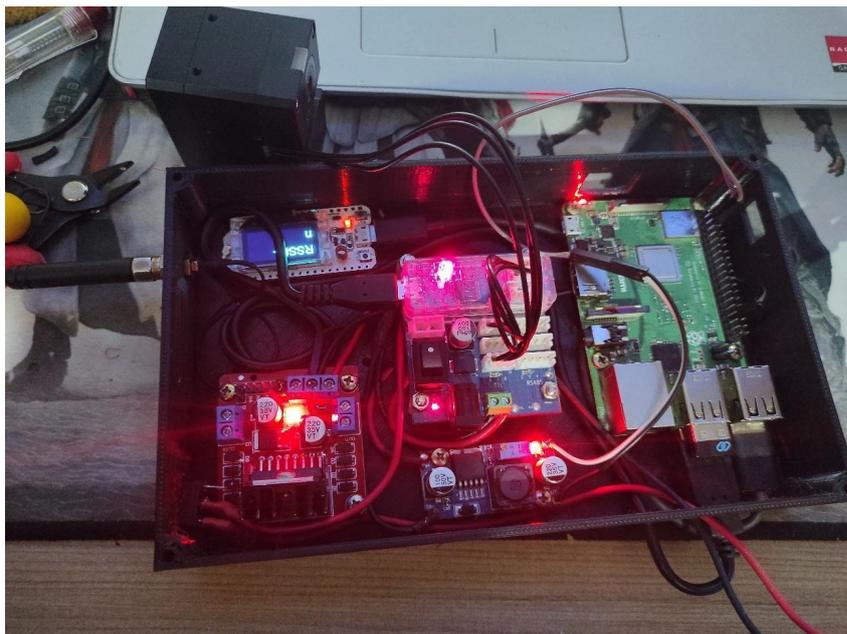
Figura 3.40: Esquemático das ligações físicas dos componentes



Fonte: O autor

A seguir, na figura 3.41, apresenta-se a caixa de controle em funcionamento.

Figura 3.41: Caixa de controle com componentes eletrônicos em funcionamento

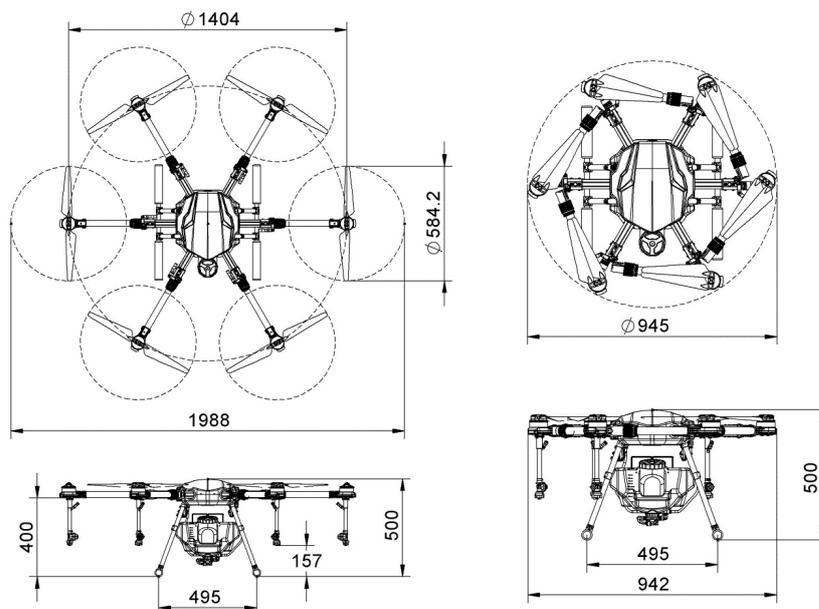


Fonte: O autor

### 3.5 Procedimentos de adaptação do drone

Para a realização dos testes, era necessário que o drone escolhido tivesse algumas características, entre elas: a capacidade de suportar o peso do módulo mais esfera; e, principalmente, ter estabilidade para voar com tanto peso acoplado a ele. O drone disponibilizado, proveniente de licitação, foi EFT E610 10L *Agricultural spraying*, figura 3.42. Esse drone tem uso inicialmente agrícola, portanto foi adaptado ao uso buscado pelo projeto.

Figura 3.42: Esquemático Drone EFT E610 10L Agricultural spraying



Fonte: Manual fornecido pelo fabricante

São os parâmetros do quadro:

- Distância entre eixos: 1404 mm;
- tamanho expandido: 1495 \* 1380 \* 500mm;
- tamanho dobrável: 945 \* 848 \* 500mm;
- capacidade do reservatório: 10L;
- peso do quadro: 5kg (excluindo o sistema de spray) ;
- motor: Hobbwying X6,6215;
- hélice: hélice dobrável;
- ESC: 80A FOC;
- tensão de alimentação: 12S;
- peso máximo de decolagem recomendado: 25kg.

As figuras 3.43 e 3.44, apresentam o drone.

Figura 3.43: Drone EFT E610 10L *Agricultural spraying*



Fonte: Manual fornecido pelo fabricante

Figura 3.44: Drone EFT E610 10L *Agricultural spraying*



Fonte: Manual fornecido pelo fabricante

Após a conclusão dessa parte, finalizou-se a parte física da versão 1, conforme se apresenta nas figuras 3.14 e 3.15.

## Capítulo 4

# APRESENTAÇÃO E DISCUSSÃO DOS RESULTADOS

Para testar a instalação da esfera apropriadamente foram efetuados diversos testes prévios antes de partir para o teste com voo devidamente. Nos testes em laboratório, o módulo conseguiu com sucesso realizar a abertura e fechamento da esfera liberando-a após o fechamento. Ademais, toda a parte eletrônica do módulo se provou eficiente.

Durante os testes de laboratório detectamos um problema: o tempo de fechamento da esfera se deu em cerca de 30 segundos, o que é um tempo extremamente alto para manter o drone estável à espera do fechamento da mesma. Mesmo com esse problema, decidiu-se testar em voo para ver quais possíveis erros ainda poderiam aparecer.

Dessa forma, para o teste de campo montou-se uma estrutura próxima à reitoria da UFU para simular, em altura segura, a instalação da esfera em fios de alta tensão, como pode-se ver na figura 4.1.

Figura 4.1: Estrutura física para testes



Fonte: O autor

Com a estrutura montada foi levantado voo com o drone. Após decolagem o drone foi posicionado logo acima do suporte para então ser abaixado e enviado o comando no celular para energizar o eletroímã. Com o drone e suporte conectados partiu-se para a primeira tentativa. O próximo passo foi o alinhamento das guias de suporte e a corda, que se mostraram muito eficientes no posicionando da corda no local correto para fechamento da esfera. Dessa forma, com a corda no local foi dado comando para que a esfera fechasse.

O controle do drone manteu-se estável por cerca de 25 segundos, levando a esfera a quase fechar completamente. Após esse período, o drone perdeu o controle e devido à força, o eletroímã não conseguiu mais segurar o suporte que veio ao chão, deixando a esfera na corda praticamente fechada.

Figura 4.2: Estrutura física para testes



Fonte: O autor

O teste relatado anteriormente, foi o melhor de todos os realizados. Em outros testes e observações, a estabilidade do drone configurou-se como um problema.

Em outro dia, testou-se a possibilidade de tentar estabilizar o drone com cabos de aço. Primeiramente com 5 metros de distância, depois 3 metros, e finalmente com 1 metro. Com o cabo de aço a estabilidade do drone se provou pior ainda, e por motivos de segurança nem sequer arriscou-se a instalação na corda. Por isso, abortou-se todas as tentativas antes mesmo de alinhar a guia.

Uma das preocupações iniciais, em relação a descarga da bateria que alimentaria o modulo, foi resolvida facilmente com o uso da bateria 3s TATTU 3.3.6. Após os testes feitos tanto em laboratório, quanto de campo foi medida a capacidade restante da bateria, e em todas elas a queda de tensão foi muito baixa, não se tornando um problema.

## Capítulo 5

# CONCLUSÃO

A solução desenvolvida nesse projeto atingiu os objetivos esperados no início de seu desenvolvimento e que estão definidos na introdução. O módulo robótico foi, fabricado, montado e devidamente testado. Ademais o sistema de controle utilizado apresentou-se muito eficiente, provando-se capaz de executar todos os comandos não exibindo falha em momento algum.

Durante o processo de realização do projeto, foi possível propor e validar modelos, descobrir e solucionar problemas mecânicos, elétricos e computacionais, além disso, ensaios foram realizados para testar metodologias e confirmar resultados esperados. Esse processo, permitiu a validação de conhecimentos absorvidos ao longo do curso de Engenharia Mecatrônica.

Por último, julga-se que a proposta maior de um projeto de fim de curso seja o aprendizado gerado ao aluno e isso, indiscutivelmente, foi alcançado. Por ser um projeto completo desde a idealização, desenho em CAD, construção e testes. Esse projeto propiciou muito aprendizado, além de deixar toda a base para uma futura versão, que pode ser desenvolvida de modo a corrigir os defeitos percebidos e relatados nesse projeto.

Como sugestão para projetos futuros, deixa-se indicado a mudança no sistema de posicionamento da esfera do horizontal para vertical. Além da procura por uma alternativa viável para o servo motor responsável por fechar a esfera. Por último, sugere-se a alteração do sistema de acoplamento do drone com o suporte.

# Referências

- [Aoshima et al. 1989] Aoshima, S., Tsujimura, T., and Yabuta, T. (1989). A wire mobile robot with multi-unit structure. In *Proceedings. IEEE/RSJ International Workshop on Intelligent Robots and Systems ' (IROS '89) 'The Autonomous Mobile Robots and Its Applications*, pages 414–421.
- [Aracil et al. ] Aracil, R., Penin, L., Feme, M., Jimenez, L., Barrientos, A., Santamaria, A., Martinez, P., and Tudun, A. ROBTET: a new teleoperated system for live-line maintenance. In *Proceedings of ESMO'95 - 1995 IEEE 7th International Conference on Transmission and Distribution Construction, Operation and Live-Line Maintenance*. IEEE.
- [CBIE 2019] CBIE (2019). Quantos quilômetros de linha de transmissão de energia temos no brasil? <https://cbie.com.br/artigos/quantos-quilometros-de-linha-de-transmissao-de-energia-temos-no-brasil#:~:text=0%20Brasil%20possui%20atualmente%20141.388,espalhadas%20por%20todo%20o%20territ%C3%B3rio.> (Accessed on 09/30/2021).
- [Docs 2021] Docs, M. (2021). Mapeamentos de pino do raspberry pi 2 & 3 - windows iot — microsoft docs. [https://docs.microsoft.com/pt-br/windows/iot-core/learn-about-hardware/pinmappings/pinmappingsrpi.](https://docs.microsoft.com/pt-br/windows/iot-core/learn-about-hardware/pinmappings/pinmappingsrpi) (Accessed on 10/19/2021).
- [Elizondo et al. 2010] Elizondo, D., Gentile, T., Candia, H., and Bell, G. (2010). Ground based robots for energized transmission line work-technology description, field projects and technical-economical justification of their application. In *2010 IEEE/PES Transmission and Distribution Conference and Exposition: Latin America (T&D-LA)*. IEEE.
- [FilipeFlop 2021] FilipeFlop (2021). Regulador de tensão lm2596 conversor dc-dc step down - filipeflop. [https://www.filipeflop.com/produto/regulador-de-tensao-lm2596-conversor-dc-dc-step-down/.](https://www.filipeflop.com/produto/regulador-de-tensao-lm2596-conversor-dc-dc-step-down/) (Accessed on 09/30/2021).
- [Forseti 2021] Forseti (2021). cantoneira em alumínio para perfil estrutural em alumínio v slot - forseti soluções. <https://loja.forsetisolucoes.com.br/cantoneira-em-aluminio-38x8x15-para-perfil-estrutural-em-aluminio.> (Accessed on 10/04/2021).

- [Foundation 2021] Foundation, R. P. (2021). Raspberry pi foundation - about us. <https://www.raspberrypi.org/about/>. (Accessed on 09/30/2021).
- [Gonçalves and Carvalho 2013] Gonçalves, R. S. and Carvalho, J. C. M. (2013). Review and latest trends in mobile robots used on power transmission lines. *International Journal of Advanced Robotic Systems*, 10(12):408.
- [Heltec 2021] Heltec (2021). Wifi lora 32 (v2) – heltec automation. <https://heltec.org/project/wifi-lora-32/>. (Accessed on 09/30/2021).
- [Hobby 2021] Hobby, A. (2021). Bateria tattú 5200mah 14.8v 35c 4s1p lipo xt60 — audiotec hobby. <https://www.audiotechobby.com.br/BATERIA-TATTU-5200MAH-14.8V-35C-4S1P-LIPO-XT60~7188~425~73~BATERIAS-CARREGADORES~BATERIA-LIPO>. (Accessed on 09/30/2021).
- [Instruments 2021] Instruments, T. (2021). Lm2596 simple switcher® power converter 150-khz 3-a step-down voltage regulator datasheet (rev. f). [https://www.ti.com/lit/ds/symlink/lm2596.pdf?ts=1633018945431&ref\\_url=https%253A%252F%252Fwww.google.com%252F](https://www.ti.com/lit/ds/symlink/lm2596.pdf?ts=1633018945431&ref_url=https%253A%252F%252Fwww.google.com%252F). (Accessed on 09/30/2021).
- [LIMA and BOMFIM 2014] LIMA, E. J. and BOMFIM, M. H. S. (2014). Robô autônomo para inspeção de linhas de alta tensão. *XX Congresso Brasileiro de Automatica*.
- [LUTEC 2021] LUTEC (2021). Lutec — parafuso allen cabeça cilíndrica m3 x 20 aço liga (embalagem 20 peças) - lutec parafusos e ferramentas. <https://www.lutecparafusoferramenta.com.br/parafuso-allen-cabeca-cilindrica-m3-x-20-aco-liga-embalagem-20-pecas>. (Accessed on 10/04/2021).
- [Projseg 2021] Projseg (2021). Fechadura eletromagnética trava eletroímã - blog da projseg. <https://blog.projseg.com.br/fechadura-eletromagnetica-trava-eletoima/>. (Accessed on 10/04/2021).
- [Ritz 2021a] Ritz (2021a). Terex ritz. <https://www.terexritz.com.br/produtos/ferramentas-manutencao-sistemas-eletricos/esferas-de-sinalizacao/>. (Accessed on 10/19/2021).
- [Ritz 2021b] Ritz, T. (2021b). Esfera para instalação por helicóptero - terex ritz. <https://www.terexritz.com.br/produtos/ferramentas-manutencao-sistemas-eletricos/esferas-de-sinalizacao/1-esfera-para-instalacao-por-helicoptero/>. (Accessed on 09/30/2021).
- [ROBOTIS 2021] ROBOTIS (2021). Dynamixel xm540-w270-r - robotis. <https://www.robotis.us/dynamixel-xm540-w270-r/>. (Accessed on 10/04/2021).

- [robotis 2021] robotis (2021). U2d2. <https://emannual.robotis.com/docs/en/parts/interface/u2d2/>. (Accessed on 09/30/2021).
- [Santos 2021] Santos, M. (2021). Rolamento linear lmu 8mm lm8 cnc 3d impressora 3d - ms santos — loja de acessórios e peças para impressoras laser - 3d - laser cnc. <https://www.mssantos.com.br/rolamento-linear-lmu-8mm-lm8-cnc-3d-impressora-3d>. (Accessed on 10/04/2021).
- [Seki et al. 2020] Seki, H., Nakayama, S., Uenishi, K., Tsuji, T., Hikizu, M., Makino, Y., Kakiuchi, A., and Kanda, Y. (2020). Development of assistive robotic arm for power line maintenance. *Precision Engineering*, 67:69–76.
- [STMicroelectronics 2021] STMicroelectronics (2021). Módulo controlador dupla ponte-h l298n. <https://www.alldatasheet.com/datasheet-pdf/pdf/22437/STMICROELECTRONICS/L298.html>. (Accessed on 09/30/2021).
- [Toussaint et al. 2009] Toussaint, K., Pouliot, N., and Montambault, S. (2009). Transmission line maintenance robots capable of crossing obstacles: State-of-the-art review and challenges ahead. *Journal of Field Robotics*, 26(5):477–499.
- [VINITRÔNICA 2021] VINITRÔNICA (2021). Micro atuador linear vx - vinitrônica. <https://www.vinitronica.com.br/micro-atuador-linear-vx>. (Accessed on 10/04/2021).

# Apêndice

## Algoritimo I.1: Codigo Raspberry Pi

```
1  #!/usr/bin/env python
2  # -*- coding: utf-8 -*-
3
4  import serial
5  from dynamixel_sdk import *
6  import os
7  import RPi.GPIO as GPIO
8  from time import *
9
10 GPIO.setmode(GPIO.BCM)
11 GPIO.setwarnings(False)
12
13 GPIO.setup(23, GPIO.OUT) # ATUDOR AVAN O
14 GPIO.setup(24, GPIO.OUT) # ATUADOR RECULO
15 GPIO.setup(22, GPIO.OUT) # SUPORTE ON/OFF
16
17
18 if os.name == 'nt':
19     import msvcrt
20
21     def getch():
22         return msvcrt.getch().decode()
23 else:
24     import sys
25     import tty
26     import termios
27     fd = sys.stdin.fileno()
28
29
30 DEVICENAME = '/dev/ttyUSB0'
31 DEVICENAME_2 = '/dev/ttyUSB1'
32 lora_receiver2 = serial.Serial('/dev/ttyUSB1', 115200)
33 lora_receiver = serial.Serial('/dev/ttyUSB0', 115200)
34
35 # Control table address dynamixel
36 TORQUE_ENABLE_T = 64
37 OPERATION_MODE = 11
```

```

38 GOAL_VELOCITY = 104
39 VELOCITY_LIMIT = 44
40 ADDR_PRO_GOAL_POSITION = 116
41 ADDR_PRO_PRESENT_POSITION = 132
42
43 # CONFIGURATION DYNAMIXEL
44 PROTOCOL_VERSION = 2.0
45 BAUDRATE = 57600
46
47 DXL_ID_1 = 1
48
49 # CONFIGURATION ENABLES
50 TORQUE_ENABLE = 1
51 TORQUE_DISABLE = 0
52 VELOCITY_CONTROL = 1
53 POSITION_CONTROL = 3
54
55 # CRITERIOS DE SEGURAN A
56 MAX_VEL = 1000
57 MIN_POS = 0
58 MAX_POS = 4000
59
60 packetHandler = PacketHandler(PROTOCOL_VERSION)
61
62
63 def port_conf(DXL_ID):
64
65     portHandler = PortHandler(DEVICENAME)
66     lora_receiver = serial.Serial('/dev/ttyUSB1', 115200)
67
68     if portHandler.openPort():
69         print('sucessed to open the port')
70         # print("Succeeded to open the port")
71
72     else:
73         print("Failed to open the port")
74         print("Press any key to terminate...")
75         getch()
76         quit()
77
78     if portHandler.setBaudRate(BAUDRATE):
79         print("Succeeded to change the baudrate")
80
81     else:
82         print("Failed to change the baudrate")
83         print("Press any key to terminate...")
84         getch()
85         quit()
86
87     dxl_comm_result, dxl_error = packetHandler.write1ByteTxRx(

```

```

88     portHandler, DXL_ID, TORQUE_ENABLE_T, TORQUE_DISABLE)
89
90     if dxl_comm_result != COMM_SUCCESS:
91         lora_receiver = serial.Serial('/dev/ttyUSB0', 115200)
92         portHandler.closePort()
93         print("saindo!")
94         # exit(1)
95         portHandler = PortHandler(DEVICENAME_2)
96
97         if portHandler.openPort():
98             print("Succeeded to open the port")
99         else:
100             print("Failed to open the port")
101             print("Press any key to terminate...")
102             getch()
103             quit()
104
105         if portHandler.setBaudRate(BAUDRATE):
106             print("Succeeded to change the baudrate")
107         else:
108             print("Failed to change the baudrate")
109             print("Press any key to terminate...")
110             getch()
111             quit()
112
113         dxl_comm_result, dxl_error = packetHandler.write1ByteTxRx(
114             portHandler, DXL_ID_1, TORQUE_ENABLE_T, TORQUE_DISABLE)
115
116         if dxl_comm_result != COMM_SUCCESS:
117             print("deu problema , n o trocou!")
118
119     return portHandler, lora_receiver
120
121
122 def Set_Operation_Mode_Velocity(DXL_ID):
123
124     # Changing Operation Mode to velocity mode
125     dxl_comm_result, dxl_error = packetHandler.write2ByteTxRx(
126         portHandler, DXL_ID, OPERATION_MODE, VELOCITY_CONTROL)
127
128     if dxl_comm_result != COMM_SUCCESS:
129         print("%s" % packetHandler.getTxRxResult(dxl_comm_result))
130     elif dxl_error != 0:
131         print("%s" % packetHandler.getRxPacketError(dxl_error))
132     else:
133         print(f'VeLOCITY mode -> id={DXL_ID}')
134
135     # Changing the velocity limit to close to maximum
136     dxl_comm_result, dxl_error = packetHandler.write4ByteTxRx(
137         portHandler, DXL_ID, VELOCITY_LIMIT, MAX_VEL)

```

```

138
139     # Enable torque
140     dxl_comm_result, dxl_error = packetHandler.write1ByteTxRx(
141         portHandler, DXL_ID, TORQUE_ENABLE_T, TORQUE_ENABLE)
142
143
144 def Set_Operation_Mode_Position(DXL_ID):
145
146     # Changing Operation Mode to velocity mode
147     dxl_comm_result, dxl_error = packetHandler.write2ByteTxRx(
148         portHandler, DXL_ID, OPERATION_MODE, POSITION_CONTROL)
149
150     if dxl_comm_result != COMM_SUCCESS:
151         print("%s" % packetHandler.getTxRxResult(dxl_comm_result))
152     elif dxl_error != 0:
153         print("%s" % packetHandler.getRxPacketError(dxl_error))
154     else:
155         print(f'Position mode -> id={DXL_ID}')
156
157     # Enable torque
158     dxl_comm_result, dxl_error = packetHandler.write1ByteTxRx(
159         portHandler, DXL_ID, TORQUE_ENABLE_T, TORQUE_ENABLE)
160     if dxl_comm_result != COMM_SUCCESS:
161         print("%s" % packetHandler.getTxRxResult(dxl_comm_result))
162     else:
163         print('toruq enable')
164
165
166 def Write_Velocity(DXL_ID, vel):
167
168     dxl_comm_result, dxl_error = packetHandler.write4ByteTxRx(
169         portHandler, DXL_ID, GOAL_VELOCITY, vel)
170     if dxl_comm_result != COMM_SUCCESS:
171         print("%s" % packetHandler.getTxRxResult(dxl_comm_result))
172     elif dxl_error != 0:
173         print("%s" % packetHandler.getRxPacketError(dxl_error))
174
175
176 def Write_position(DXL_ID, pos):
177
178     dxl_comm_result, dxl_error = packetHandler.write4ByteTxRx(
179         portHandler, DXL_ID, ADDR_PRO_GOAL_POSITION, pos)
180
181     if dxl_comm_result != COMM_SUCCESS:
182         print("%s" % packetHandler.getTxRxResult(dxl_comm_result))
183     elif dxl_error != 0:
184         print("%s" % packetHandler.getRxPacketError(dxl_error))
185
186
187 def Disable_Torque(DXL_ID):

```

```

188
189     # Disable Dynamixel Torque -- in case pre-enable
190     dxl_comm_result, dxl_error = packetHandler.writeByteTxRx(
191         portHandler, DXL_ID, TORQUE_ENABLE_T, TORQUE_DISABLE)
192
193     if dxl_comm_result != COMM_SUCCESS:
194         print("1-%s" % packetHandler.getTxRxResult(dxl_comm_result))
195         # closeport()
196         return 1
197     elif dxl_error != 0:
198         print("2-%s" % packetHandler.getRxPacketError(dxl_error))
199     else:
200         print(f'Torque has been successfully turn off! , ID_{DXL_ID}')
201
202         #portHandler, lora_receiver = port_conf(DXL_ID_1)
203
204         # SINAL VISUAL DO SUCESSO!
205
206
207     portHandler, lora_receiver = port_conf(DXL_ID_1)
208     Set_Operation_Mode_Velocity(DXL_ID_1)
209     Write_Velocity(DXL_ID_1, MAX_VEL)
210     sleep(2)
211     Write_Velocity(DXL_ID_1, 0)
212
213     flag = False
214     flag_a = False
215     flag_f = False
216     flag_2 = False
217
218     while True:
219         recebe = lora_receiver.readline()
220         a = int(recebe.decode("utf8"))
221         print(a)
222
223         if (a == 11 and flag == False):
224
225             print("AVAN O MOTOR!!")
226             Write_Velocity(DXL_ID_1, MAX_VEL)
227             flag_f = True
228             ms_f = time_ns()
229
230         if (flag_f == True):
231             print("motor fechando")
232
233             if(time_ns()-ms_f > 12000000000 and flag_2 == False): # 2s
234                 ms_f = time_ns()
235                 Write_Velocity(DXL_ID_1, MAX_VEL)
236                 GPIO.output(23, GPIO.HIGH)
237                 ms_a = time_ns()

```

```

238         flag_a = True
239         flag_2 = True
240         print("segunda rodada!")
241
242         if(flag_2 == True and time_ns()-ms_f > 14000000000): # 2s
243             Write_Velocity(DXL_ID_1, 0)
244             flag_f = False
245             flag_2 = False
246
247     elif(a == 10):
248         print("recuar_motor")
249         Write_Velocity(DXL_ID_1, -150)
250         flag = True
251         ms = time_ns()
252
253     if (flag == True):
254         if(time_ns()-ms > 8000000000):
255             Write_Velocity(DXL_ID_1, 0)
256             flag = False
257
258     elif(a == 12 and flag_a == False):
259         print("atuador_on")
260         ms_a = time_ns()
261         GPIO.output(24, GPIO.HIGH)
262         flag_a = True
263         print("atuador_on_3")
264
265     elif(a == 13 and flag_a == False):
266         print("atuador_off")
267         ms_a = time_ns()
268         GPIO.output(23, GPIO.HIGH)
269         flag_a = True
270
271     elif(a == 14):
272         print("MAG_ON")
273         GPIO.output(22, GPIO.HIGH)
274         sleep(0.1)
275
276     elif(a == 15):
277         print("MAG_OFF")
278         GPIO.output(22, GPIO.LOW)
279         Write_Velocity(DXL_ID_1, 0)
280         sleep(0.1)
281
282     elif (flag_a == True):
283         if(time_ns()-ms_a > 7500000000): # 2s
284             flag_a = False
285             GPIO.output(23, GPIO.LOW)
286             GPIO.output(24, GPIO.LOW)
287 a = 5

```

```

288
289 Disable_Torque (DXL_ID_1)
290 # Close port
291 portHandler.closePort ()
292 print ("Succeeded to close the port")
293
294
295 #
296 #####
297
298 # FOR FURTHER USE
299 """
300 # for i in range(0, 100):
301 #     recebe = lora_receiver.readline()
302 #     print (f'{recebe.decode("utf8")}')
303 #     time.sleep(0.2)
304
305     GPIO.output (24, GPIO.HIGH)
306     sleep (3)
307     GPIO.output (24, GPIO.LOW)
308     GPIO.output (23, GPIO.HIGH)
309     sleep (3)
310     GPIO.output (23, GPIO.LOW)
311     # sleep (3)
312
313     # Disable_Torque (DXL_ID_1)
314     # Set_Operation_Mode_Velocity (DXL_ID_1)
315     # Write_Velocity (DXL_ID_1, 500)
316     # sleep (3)
317     # Write_Velocity (DXL_ID_1, -500)
318     # sleep (3)
319     # Write_Velocity (DXL_ID_1, 0)
320     # sleep (3) """

```

## Algoritmo I.2: Código LoRa Sender

```
1 #include "heltec.h"
2 #include "images.h"
3 #include "BluetoothSerial.h"
4
5 char buf;
6 int a;
7
8 BluetoothSerial SerialBT;
9
10 #define BAND 915E6
11
12 unsigned int num = 0;
13 String rssi = "RSSI --";
14 String packSize = "--";
15 String packet;
16
17 void logo() //LOGO Heltec
18 {
19     Heltec.display->clear();
20     Heltec.display->drawXbm(0, 5, logo_width, logo_height, logo_bits);
21     Heltec.display->display();
22 }
23
24 void setup()
25 {
26     pinMode(LED, OUTPUT);
27
28     Heltec.begin(true /*DisplayEnable Enable*/, true /*Heltec.Heltec.
29         Heltec.LoRa Disable*/, true /*Serial Enable*/, true /*PABOOST
30         Enable*/, BAND /*long BAND*/);
31
32     Heltec.display->init();
33     Heltec.display->flipScreenVertically();
34     Heltec.display->setFont(ArialMT_Plain_16);
35     logo();
36     // AT  AQUI OK
37
38     delay(100);
39
40     Serial.begin(115200);
41     SerialBT.begin("LoRa_BT");
42 }
43
44 void loop()
45 {
46     if (Serial.available())
47     {
48         SerialBT.write(Serial.read());
49     }
50 }
```

```

47     }
48
49     if (SerialBT.available())
50     {
51         a = SerialBT.read();
52         Serial.write(a);
53     }
54
55     if (a == '1')
56     {
57         LoRa.beginPacket();
58         LoRa.setTxPower(14, RF_PACONFIG_PASELECT_PABOOST);
59         //LoRa.print("HIGH ");
60         num = 1011;
61         LoRa.print(num);
62         LoRa.endPacket();
63
64         Heltec.display->clear();
65         Heltec.display->setTextAlignment(TEXT_ALIGN_LEFT);
66         Heltec.display->setFont(ArialMT_Plain_24);
67
68         Heltec.display->drawString(20, 0, "MOTOR");
69         Heltec.display->drawString(40, 26, "ESQ");
70         Heltec.display->display();
71     }
72
73     if (a == '2')
74     {
75         LoRa.beginPacket();
76         LoRa.setTxPower(14, RF_PACONFIG_PASELECT_PABOOST);
77         num = 2022;
78         LoRa.print(num);
79         LoRa.endPacket();
80
81         Heltec.display->clear();
82         Heltec.display->setTextAlignment(TEXT_ALIGN_LEFT);
83         Heltec.display->setFont(ArialMT_Plain_24);
84
85         Heltec.display->drawString(20, 0, "MOTOR");
86         Heltec.display->drawString(40, 26, "DIR");
87         Heltec.display->display();
88     }
89     if (a == '3')
90     {
91         LoRa.beginPacket();
92         LoRa.setTxPower(14, RF_PACONFIG_PASELECT_PABOOST);
93         num = 3033;
94         LoRa.print(num);
95         LoRa.endPacket();
96

```

```

97         Heltec.display->clear ();
98         Heltec.display->setTextAlignment (TEXT_ALIGN_LEFT);
99         Heltec.display->setFont (ArialMT_Plain_24);
100
101         Heltec.display->drawString (20, 0, "ATUADOR");
102         Heltec.display->drawString (40, 26, "HIGH");
103         Heltec.display->display ();
104     }
105     if (a == '4')
106     {
107         LoRa.beginPacket ();
108         LoRa.setTxPower (14, RF_PACONFIG_PASELECT_PABOOST);
109         num = 4044;
110         LoRa.print (num);
111         LoRa.endPacket ();
112
113         Heltec.display->clear ();
114         Heltec.display->setTextAlignment (TEXT_ALIGN_LEFT);
115         Heltec.display->setFont (ArialMT_Plain_24);
116
117         Heltec.display->drawString (20, 0, "ATUADOR");
118         Heltec.display->drawString (40, 26, "LOW");
119         Heltec.display->display ();
120     }
121
122     if (a == '5')
123     {
124         LoRa.beginPacket ();
125         LoRa.setTxPower (14, RF_PACONFIG_PASELECT_PABOOST);
126         num = 5055;
127         LoRa.print (num);
128         LoRa.endPacket ();
129
130         Heltec.display->clear ();
131         Heltec.display->setTextAlignment (TEXT_ALIGN_LEFT);
132         Heltec.display->setFont (ArialMT_Plain_24);
133
134         Heltec.display->drawString (20, 0, "SUPORTE");
135         Heltec.display->drawString (40, 26, "HIGH");
136         Heltec.display->display ();
137     }
138
139     if (a == '6')
140     {
141         LoRa.beginPacket ();
142         LoRa.setTxPower (14, RF_PACONFIG_PASELECT_PABOOST);
143         num = 6066;
144         LoRa.print (num);
145         LoRa.endPacket ();
146

```

```

147         Heltec.display->clear ();
148         Heltec.display->setTextAlignment (TEXT_ALIGN_LEFT);
149         Heltec.display->setFont (ArialMT_Plain_24);
150
151         Heltec.display->drawString (20, 0, "SUPPORT");
152         Heltec.display->drawString (40, 26, "LOW");
153         Heltec.display->display ();
154     }
155
156     // // send packet
157     LoRa.beginPacket ();
158     LoRa.setTxPower (14, RF_PACONFIG_PASELECT_PABOOST);
159 //     LoRa.print ("Pacote: ");
160     LoRa.print (num);
161     LoRa.endPacket ();
162
163     num++;
164
165     a = 'a';
166
167     // delay(20); // wait for a second
168 }

```

### Algoritmo I.3: Código LoRa Reciever

```
1 #include "heltec.h"
2 #include "images.h"
3
4
5 #define BAND 915E6
6
7 String rssi = "RSSI --";
8 String packSize = "--";
9 String packet;
10
11 int LED1 = 32;
12 int LED2 = 33;
13 int LED3 = 35;
14
15 int pos; // Posi o Servo
16
17 void LoRaData()
18 {
19   Heltec.display->clear();
20   Heltec.display->setTextAlignment(TEXT_ALIGN_LEFT);
21   Heltec.display->setFont(ArialMT_Plain_24);
22   //Heltec.display->drawString(0, 18, "Pct="+ packSize + "b");
23   //Heltec.display->drawStringMaxWidth(0, 32, 128, packet);
24   Heltec.display->drawString(0, 0, rssi);
25   Heltec.display->display();
26 }
27
28 void cbk(int packetSize)
29 {
30   packet = "";
31   packSize = String(packetSize, DEC);
32   for (int i = 0; i < packetSize; i++)
33   {
34     packet += (char)LoRa.read();
35   }
36
37   rssi = "RSSI " + String(LoRa.packetRssi(), DEC);
38   LoRaData();
39 }
40
41 void setup()
42 {
43   Serial.begin(115200);
44
45   Heltec.begin(true /*DisplayEnable Enable*/, true /*Heltec.Heltec.Heltec.
46     LoRa Disable*/, true /*Serial Enable*/, true /*PABOOST Enable*/, BAND
47     /*long BAND*/);
```

```

47 Heltec.display->init ();
48 Heltec.display->flipScreenVertically ();
49 Heltec.display->setFont (ArialMT_Plain_16);
50
51 Heltec.display->drawString (0, 0, "Tudo OK!");
52 Heltec.display->drawString (0, 18, "Esperando Dados");
53 Heltec.display->display ();
54 delay (5);
55 LoRa.receive ();
56 }
57
58 void loop ()
59 {
60   int packetSize = LoRa.parsePacket ();
61   if (packetSize) {cbk (packetSize);}
62
63     if (packet == "1011")
64     {
65       Serial.println ("10");
66       delay (5);
67
68       Heltec.display->clear ();
69       Heltec.display->setTextAlignment (TEXT_ALIGN_LEFT);
70       Heltec.display->setFont (ArialMT_Plain_24);
71       Heltec.display->drawStringMaxWidth (0, 32, 128, "MOTOR ESQ");
72       Heltec.display->drawString (0, 0, rssi);
73       Heltec.display->display ();
74       delay (10);
75     }
76
77     else if (packet == "2022")
78     {
79       Serial.println ("11");
80       delay (5);
81
82       Heltec.display->clear ();
83       Heltec.display->setTextAlignment (TEXT_ALIGN_LEFT);
84       Heltec.display->setFont (ArialMT_Plain_24);
85       Heltec.display->drawStringMaxWidth (0, 32, 128, "MOTOR DIR");
86       Heltec.display->drawString (0, 0, rssi);
87       Heltec.display->display ();
88       delay (10);
89     }
90
91     else if (packet == "3033")
92     {
93       Serial.println ("12");
94       delay (5);
95
96       Heltec.display->clear ();

```

```

97     Heltec.display->setTextAlignment (TEXT_ALIGN_LEFT);
98     Heltec.display->setFont (ArialMT_Plain_24);
99     Heltec.display->drawStringMaxWidth (0, 32, 128, "ATUA HIGH");
100    Heltec.display->drawString (0, 0, rssi);
101    Heltec.display->display ();
102    delay (10);
103    }
104
105    else if (packet == "4044")
106    {
107    Serial.println ("13");
108    delay (5);
109
110    Heltec.display->clear ();
111        Heltec.display->setTextAlignment (TEXT_ALIGN_LEFT);
112        Heltec.display->setFont (ArialMT_Plain_24);
113        Heltec.display->drawStringMaxWidth (0, 32, 128, "ATUA LOW");
114        Heltec.display->drawString (0, 0, rssi);
115        Heltec.display->display ();
116    delay (10);
117    }
118
119    else if (packet == "5055")
120    {
121    Serial.println ("14");
122    delay (5);
123
124    Heltec.display->clear ();
125        Heltec.display->setTextAlignment (TEXT_ALIGN_LEFT);
126        Heltec.display->setFont (ArialMT_Plain_24);
127        Heltec.display->drawStringMaxWidth (0, 32, 128, "MAG ON");
128        Heltec.display->drawString (0, 0, rssi);
129        Heltec.display->display ();
130    delay (10);
131    }
132
133    else if (packet == "6066")
134    {
135    Serial.println ("15");
136    delay (5);
137
138    Heltec.display->clear ();
139        Heltec.display->setTextAlignment (TEXT_ALIGN_LEFT);
140        Heltec.display->setFont (ArialMT_Plain_24);
141        Heltec.display->drawStringMaxWidth (0, 32, 128, "MAG OFF");
142        Heltec.display->drawString (0, 0, rssi);
143        Heltec.display->display ();
144    delay (10);
145    }
146

```

```
147     else
148     {
149     Serial.println("80");
150     delay(5);
151
152     Heltec.display->clear();
153     Heltec.display->setTextAlignment(TEXT_ALIGN_LEFT);
154     Heltec.display->setFont(ArialMT_Plain_24);
155     Heltec.display->drawStringMaxWidth(0, 32, 128, "n");
156     Heltec.display->drawString(0, 0, rssi);
157     Heltec.display->display();
158     delay(10);
159     }
160
161 }
```