CENTRO UNIVERSITÁRIO INTERNACIONAL UNINTER
ESCOLA SUPERIOR POLITÉCNICA
BACHARELADO EM ENGENHARIA DA COMPUTAÇÃO

ALEXANDRE BIDA
CARLOS ALBERTO BORGES PADOVEZI JUNIOR
PAULO RYCARDO TEODORO CHRISTOFF

MAPEAMENTO DE AMBIENTES COM LIDAR

TRABALHO DE CONCLUSÃO DE CURSO

CURITIBA
2020

ALEXANDRE BIDA
CARLOS ALBERTO BORGES PADOVEZI JUNIOR
PAULO RYCARDO TEODORO CHRISTOFF

MAPEAMENTO DE AMBIENTES COM LIDAR

Trabalho de Conclusão de Curso apresentado como requisito parcial à obtenção do título de Bacharel em Engenharia da Computação pela Escola Superior Politécnica do Centro Universitário Internacional Uninter.

Orientador: Prof. Me. Charles Way Hun Fung.

CURITIBA
2020

Este projeto é dedicado aos nossos familiares, amigos e professores que sempre nos incentivaram durante este período.

AGRADECIMENTOS

Nosso primeiro agradecimento é direcionado aos nossos pais que nos incentivaram e apoiaram mesmo nos momentos mais difíceis, nos mostrando que todo esforço vale a pena.

Agradecemos aos professores do Centro Universitário Internacional UNINTER que nos ajudaram e sempre se mostraram dispostos a sanar nossas dúvidas em todas as etapas do Bacharelado principalmente nesta última.

Somos gratos ao professor Charles, nosso orientador, por nos ensinar e instruir durante todo o processo de desenvolvimento deste projeto.

Agradecemos também aos nossos amigos e colegas de turma, pela troca de conhecimentos, pelos momentos de descontração e por sempre estarem ao nosso lado com apoio e incentivo durante a graduação.

“O sucesso é um professor perverso.
Ele seduz as pessoas inteligentes e as faz
pensar que jamais vão cair”. (GATES, Bill)

RESUMO

BIDA, Alexandre; JUNIOR, Carlos Alberto Borges Padovezi; CHRISTOFF, Paulo Rycardo Teodoro. **MAPEAMENTO DE AMBIENTES COM LIDAR:** Trabalho de Conclusão de Curso (Bacharelado em Engenharia da Computação) – Centro Universitário Internacional Uninter. Curitiba, 2020.

O avanço da tecnologia, principalmente na área de robótica, possibilita a substituição de pessoas em tarefas que ofereçam riscos a sua integridade física, que necessitem de muita precisão e em tarefas que exigem esforço repetitivo. Este projeto tem como objetivo utilizar uma plataforma robótica juntamente com um sensor LIDAR para realizar o mapeamento de ambientes e uma câmera para captura de imagens sem a necessidade do contato físico humano no local. O sistema é desenvolvido e executado na plataforma do ROS (*Robotic Operational System*) através do sistema operacional Ubuntu, os dados são enviados pelo sensor, tratados pelo *RaspBerry Pi* e publicados em um arquivo de imagem.

Palavras-chave: ROS, LIDAR, robótica, sensor, mapeamento de ambientes.

ABSTRACT

BIDA, Alexandre; JUNIOR, Carlos Alberto Borges Padovezi; CHRISTOFF, Paulo Rycardo Teodoro. **MAPPING ENVIRONMENTS WITH LIDAR:** Course Completion Work (Bachelor of Computer Engineering) - Uninter International University Center. Curitiba, 2020.

The advancement of technology, especially in the area of robotics, makes it possible to replace people in tasks that pose risks to their physical integrity, that require a lot of precision and in tasks that require repetitive effort. This project aims to use a robotic platform together with a LIDAR sensor to perform the mapping of environments and a camera for capturing images without the need for human physical contact at the site. The system is developed and executed on the ROS platform (Robotic Operational System) through the Ubuntu operating system, the data is sent by the sensor, treated by RaspBerry Pi and published in an image file.

Keywords: ROS, LIDAR, robotics, sensor, environment mapping.

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE TABELAS

LISTA DE SIGLAS

API	Interface de Programação de Aplicativos (*Application Programming Interface*)
DC	Corrente Direta/Contínua (*Direct Current*)
GPIO	Entrada/Saída de Propósito Geral (*General Purpose Input/Output*)
GPS	Sistema de Posicionamento Global (*Global Positioning System*)
HDMI	Interface multimídia de alta definição (*High-Definition Multimedia Interface*)
HTML	Linguagem de Marcação de Hipertexto (*Hypertext Markup Language*)
IFR	Federação Internacional de Robótica (*International Federation of Robotics*)
IP	Protocolo de Internet (*Internet Protocol*)
NPM	Gerenciador de Pacotes do Node (*Node Package Manager*)
PWM	Modulação da Largura do Pulso (*Pulse Width Modulation*)
SO	Sistema Operacional (Operational System)
USB	Porta Serial Universal (*Universal Serial Bus*)
WS	Área de trabalho (Work Space)

LISTA DE ACRÔNIMOS

BIT	Dígito Binário (*Binary Digit*)
DARPA	Agência de Projetos de Pesquisa Avançada de Defesa (*Defense Advanced Research Projects Agency*)
DOM	Modelo de Objeto de Documento (*Document Object Model*)
JSON	Notação de Objeto *JavaScript* (*JavaScript Object Notation*)
LIDAR	Detecção e Alcance da Luz (*Light Detection And Ranging*)
LADAR	Detecção e Alcance a Laser (*Laser Detection And Ranging*)
LASER	Amplificação de Luz por Emissão Estimulada de Radiação (*Light Amplification by Stimulated Emission of Radiation*)
ROS	Sistema Operacional de Robôs (*Robot Operating System*)
SLAM	Localização e mapeamento simultâneos (*Simultaneous Localization and Mapping*)
WIFI	Fidelidade sem Fio (Wireless Fidelity)

SUMÁRIO

1. INTRODUÇÃO

Há anos o homem vem procurando e desenvolvendo novas soluções tecnológicas que otimizem e agilizem suas atividades, isso se tornou possível graças aos "avanços de *hardware* e *software* desenvolvidos. Em termos de *hardware*, os computadores e dispositivos embarcados vêm sendo miniaturizados, tendo seus custos reduzidos e sua capacidade de processamento aumentada. Além disso, eles têm se tornado mais robustos e precisos, consumindo menos energia e alcançando maior autonomia. Tudo isso reflete diretamente na área de *software*. A maior quantidade e precisão das informações coletadas sobre a área de atuação do robô permitem o desenvolvimento de novos algoritmos nas áreas de controle, tomada de decisão, processamento de imagens, reconhecimento de voz, entre outros" (OSÓRIO, PRESTES, ROMERO e WOLF, 2014). Esta tecnologia se faz presente nos mais diversos segmentos da sociedade, tornando as cidades cada vez mais conectadas e aumentando o conforto dos que a utilizam, alguns exemplos das aplicações tecnológicas de acordo com MURPHY (2000, pg. 16, apud OSÓRIO, PRESTES, ROMERO e WOLF) são: "tarefas que possam colocar a vida do ser humano em risco (nucleares, espaciais, militares), substituir os humanos nas tarefas repetitivas e tediosas, uso humanitário (busca e resgate, remoção de minas terrestres), tarefas diárias (ajudante, aspirador de pó, limpeza de vidros), cirurgias minimamente invasivas (vide laparoscopia), educação de jovens, na forma de professores robôs, ajudar o desenvolvimento cognitivo de pessoas autistas". Existe um fator em comum entre as áreas anteriormente citadas, algo que não obrigatoriamente precisa estar inserido no cotidiano da população atualmente para que possamos mensurar a evolução, mas que com o passar do tempo será uma tendência cada vez maior o uso de robôs em nossas atividades.

"O estudo dos robôs ganhou força no século XX, porém ao longo da história da humanidade algumas invenções foram responsáveis por inspirar o surgimento desta área de estudo, como por exemplo, o cachorro mecânico de brinquedo encontrado no Egito datado de 1390 – 1353 a.C." (Adaptado de AYRES, 2007) que está em exposição no *Metropolitan Museum of Art* (Museu Metropolitano de Arte) localizado em Nova York.

Figura 1: Cachorro mecânico de brinquedo.

Fonte: Metropolitan Museum of Art.

"A palavra *Robot* (robô) é oriunda da palavra *Robota* que em tcheco significa trabalho árduo, foi utilizada pela primeira vez em 1920 na peça de ficção científica *Rossumovi Univerzální Roboti* (em inglês *Rossum's Universal Robot* e em português Robô Universal de Rossum) do dramaturgo tcheco Carel Karpec" (Adaptado de AYRES, 2007). O contexto da peça se passa na criação de robôs humanizados por parte do cientista Rossum a partir de uma substância recém-descoberta, para que realizassem tarefas pesadas e repetitivas.

Continuando no campo da literatura, "o termo robótica (estudo dos robôs) foi introduzido no ano de 1924 pelo escritor Isaac Asimov em sua obra *Runaround*" (Adaptado de MAIA, 2003). Nesta obra o autor atribui aos robôs comportamento e características dos seres humanos, porém esses atributos deveriam ser pré-programados seguindo diversas regras. De acordo com Asimov a robótica deve seguir as seguintes diretrizes, que ficaram conhecidas como as Leis da Robótica:

1. Um robô não pode ferir um ser humano ou, por omissão, permitir que um ser humano sofra algum mal;
2. Um robô deve obedecer às ordens que lhe sejam dadas por seres humanos, exceto nos casos em que tais ordens contrariem a Primeira Lei;
3. Um robô deve proteger sua própria existência, desde que tal proteção não entre em conflito com a Primeira e a Segunda Lei.

Esta é uma nova era da robótica, onde existem máquinas ou sistemas computadorizados capazes de interagir de maneira autônoma com os seres humanos, robôs que utilizam sensores (temperatura, ultrassom, infravermelho...) para se localizarem e para captarem dados do ambiente que estão inseridos, por exemplo: "existem robôs sendo usados em cirurgias de alta precisão, como o robô Da Vinci; em limpezas de ambientes (aspirando o pó), como o robô Roomba da iRobot ou o Navibot da Samsung; na área de entretenimento, como o Lego MindStorms, Aibo da Sony, Pleo da Innvo Labs, RoboSapiens e Rovio da WowWee, entre muitos outros" (OSÓRIO, PRESTES, ROMERO e WOLF, 2014).

É possível citar como exemplo de evolução tecnológica a Agência de Projetos de Pesquisa Avançada de Defesa DARPA (*Defense Advanced Research Projects Agency*), uma agência criada por cientistas e militares norte-americanos em resposta ao lançamento do Sputnik por parte do governo russo. A DARPA investe não só em tecnologia militar, investe também em reconhecimento de voz, tradução automatizada de idiomas, em dispositivos de GPS e em outras tecnologias e pesquisas. Em 2004 foi lançado um desafio com o objetivo de estimular o desenvolvimento de veículos autônomos com propósito militar, as equipes participantes do desafio deveriam criar um veículo que percorresse de forma autônoma o percurso de Barstow na Califórnia até Primm em Nevada, porém nenhum veículo conseguiu completar o percurso e nenhuma das equipes recebeu o prêmio de 1 milhão de dólares. No ano seguinte em 2005 cinco equipes concluíram o percurso estabelecido, quem ficou com o prêmio de US $2 milhões foi a equipe Stanford Racing Team que percorreu as 132 milhas estabelecidas em 6 horas e 52 minutos, o menor tempo de acordo com o site da DARPA.

No ano de 1987 foi criada a Federação Internacional de Robótica a IFR (*International Federation of Robotics*), uma instituição sem fins lucrativos com o propósito de estimular pesquisas em robótica e fornecer dados do mercado de robôs para pesquisa, estudo e estatísticas.

Segundo a Federação Internacional de Robótica atualmente existem cinco grandes mercados de robôs voltados para a indústria, localizados nos seguintes países: China, Japão, Estados Unidos, Coreia do Sul e Alemanha, esses cinco países juntos totalizam 74% das instalações de robôs do planeta. O continente asiático é responsável pelo maior crescimento na quantidade de instalações, entre 2008 e 2018 houve um aumento de 371,7% de 60.000 para 283.000 unidades, conforme ilustra a Figura 2.

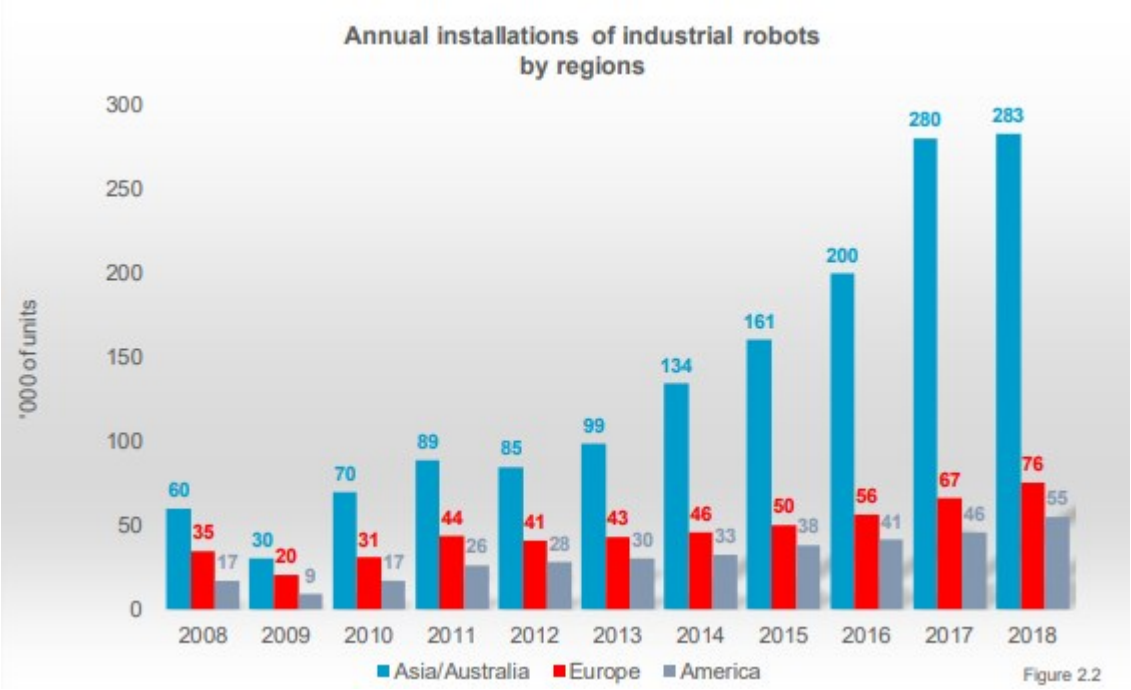

Figura 2: Instalações anuais de robô industriais.

Fonte: IFR.

Conforme visto anteriormente o maior uso de robôs está presente nas indústrias, porém de acordo com dados da IFR há um crescimento em cerca de 90% nas vendas de robôs para atividades domésticas entre 2017 e 2018 e a previsão é que neste ano (2019) haja um aumento de 44% em comparação com 2018. A expectativa é que esse crescimento continue, para 2020, 2021 e 2022 as vendas estão estimadas em 25,5 milhões de unidades, 37,2 milhões de unidades e 55,1 milhões de unidades para cada ano respectivamente.

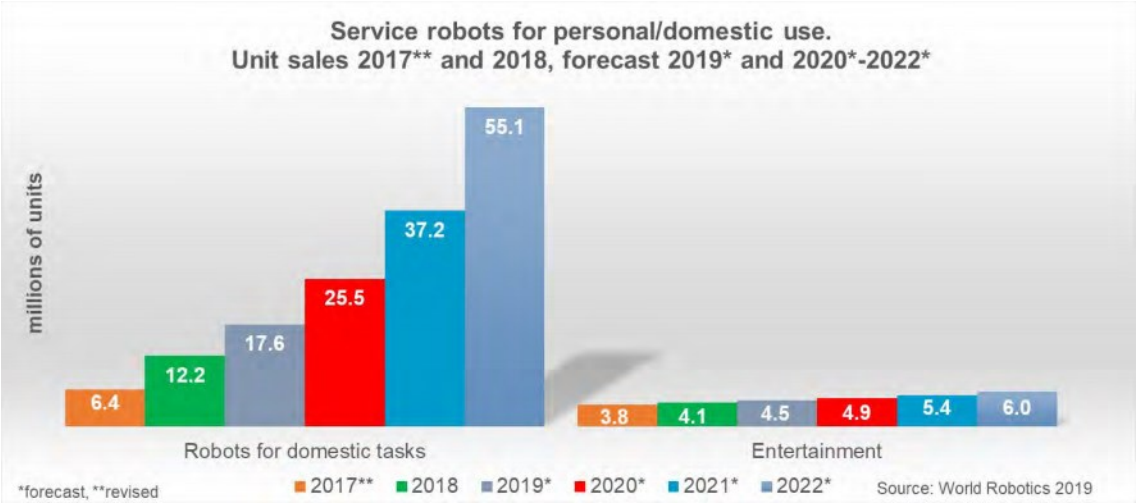

Figura 3: Robôs para uso pessoal/doméstico.

Fonte: Müller - IFR.

Como visto anteriormente a tendência é de que o uso de robôs se torne cada vez mais comum em diversas áreas, destaca-se a utilização da robótica para mapeamento de ambientes

fechados, já que para ambientes amplos e abertos existem muitas técnicas de mapeamento e localização como por exemplo o gps. Para que o robô possa atuar coletando informações do ambiente, ele deve possuir sensores, estes são escolhidos com base na necessidade do projeto, neste caso, pode-se utilizar dispositivos que emitem ondas de luz ou ondas sonoras para localizar objetos em um local desconhecido, os dados coletados são tratados por outra parte do sistema do robô ou por outro sistema que não está inserido no controle do robô.

O autor (FERREIRA, 2014) ressalta a importância do mapeamento de ambientes para algumas áreas específicas: “A geração de mapas de ambientes pode ser útil em uma gama de aplicações como engenharia civil, arquitetura, entretenimento, projetos de plantas industriais e processos de engenharia reversa”.

1.1 PROBLEMA

Em locais de difícil acesso para pessoas, como por exemplo um duto de ar que se encontra obstruído ou tubulações de gás que estão rompidas. A fim de prestar manutenção, para que possa ser encontrada com precisão a origem desse problema, sem a necessidade de colocar em risco a saúde de uma pessoa. Este projeto busca o desenvolvimento de um sistema que realize o mapeamento bidimensional de ambientes inóspitos, insalubres. Dessa forma, diminuindo a necessidade de humanos entrarem nestes mesmos ambientes.

1.2 JUSTIFICATIVA

Para mapear ambientes fechados é necessário que um robô percorra o local em que está inserido e colete dados através de seus sensores. Estes dados são tratados por um sistema que gera o mapa do ambiente e exibe uma imagem que representa a visão frontal do robô.

Com o mapa e a imagem apresentados para o usuário, será possível analisar algumas características do ambiente nos eixos cartesianos, obstruções, obstáculos, distância de suas extremidades em relação a plataforma robótica e permitirá ao usuário a visualização do ambiente.

1.3 OBJETIVO GERAL

Desenvolver um sistema de controle para um protótipo, composto por um sensor LIDAR e uma câmera acoplados a uma base robótica, o sistema deve possuir uma interface acessível pelo computador, que permita através da interação com o usuário controlar o próprio robô, esse controle será realizado através de uma rede Wi-Fi, ou seja, a comunicação entre a base robótica e a interface será realizada pela rede. Conforme o robô se desloca pelo local, ele realiza o mapeamento do ambiente, então os resultados são exibidos e atualizados com um pequeno intervalo para o usuário.

1.4 OBJETIVOS ESPECÍFICOS

- Desenvolver um algoritmo de controle para uma base robótica através do ROS;

- Desenvolver a comunicação entre a base robótica e a interface que recebe os comandos das movimentações, utilizando Wi-Fi;
- Desenvolver um algoritmo que realize a montagem do mapeamento dos dados registrados pelo sensor LIDAR;
- Desenvolver um algoritmo que realize a montagem da imagem capturada pela câmera;
- Desenvolver uma interface de visualização em tempo real da imagem formada através dos dados captados pelo sensor e pela câmera;
- Desenvolver a comunicação entre o *Raspberry* e o LIDAR.

2. FUNDAMENTAÇÃO TEÓRICA

Para a utilização de robôs nas mais diversas tarefas é necessário que eles obtenham dados sobre si mesmos e sobre o ambiente no qual estão inseridos, por exemplo, dados relacionados a temperatura, pressão, velocidade, posição e etc. Desta forma as informações adquiridas são utilizadas como parâmetro para os cálculos programados.

2.1 SENSORES

De acordo com os autores (OSÓRIO, PRESTES, ROMERO e WOLF, 2014) "sensores são também chamados de transdutores (*transducer*) porque transformam a energia associada com uma dada medida em outra forma de energia. Basicamente, um sensor recebe energia (som, pressão, luz) e a transforma em um sinal analógico ou digital que pode ser usado pelo robô".

2.2 LIDAR (*Light Detection And Ranging*)

Sensores de distância baseados em *laser* são comumente chamados no meio acadêmico de LIDAR (*Light Detection And Ranging*) e em contextos militares, de LADAR (*Laser Detection And Ranging*). É uma tecnologia que permite detectar objetos remotamente, usando para isso *laser* pulsado (OSÓRIO, PRESTES, ROMERO e WOLF, 2014).

Existem duas maneiras de calcular a distância. A primeira delas é denominada tempo de voo, na qual se mede o tempo entre a emissão de um pulso de luz e a recepção do pulso refletido, como mostra a Figura 4. A distância entre o dispositivo LIDAR e a superfície pode ser então calculada utilizando a seguinte expressão:

$$d = \frac{1}{2} ct \tag{1}$$

na qual t é o tempo medido entre a emissão e recepção do pulso e c ≈ 299.792.458 m/s é a velocidade da luz (2008, KEMENY e TURNER apud FERREIRA).

A segunda estratégia de cálculo da distância tem como princípio a diferença de fase entre uma onda emitida e a onda refletida pela superfície. Neste caso, a onda de luz a ser emitida e modulada senoidalmente e incidida sobre a superfície, como mostra a Figura 4. A componente da onda refletida em direção ao sensor é então detectada e comparada com a onda emitida para

determinação da diferença de fase entre as duas. Dado que as ondas emitida e refletida possuem mesma frequência *f*, é possível determinar a distância entre o dispositivo de mapeamento e a superfície a partir da seguinte expressão:

$$d = \frac{c\varphi}{4\pi f} \quad (2)$$

Na qual φ é a diferença de fase entre as ondas emitida e recebida em radianos (2008, KEMENY e TURNER apud FERREIRA).

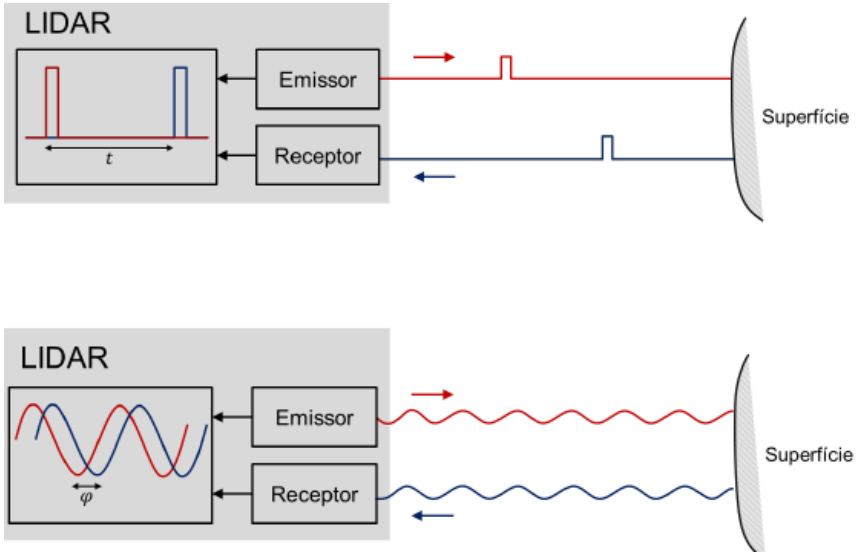

Figura 4: Funcionamento do LIDAR.

Fonte: FERREIRA.

2.3 MOTORES ELÉTRICOS

De acordo com o autor (PETRUZELLA, 2013) um motor elétrico converte energia elétrica em energia mecânica usando campos magnéticos que interagem entre si. Os motores elétricos são usados para uma variedade de operações nas áreas residenciais, comerciais e industriais.

2.3.1 MOTOR D.C.

O autor (OLIVEIRA, 2014) descreve que motores DC "são motores elétricos operando à base de corrente contínua. Eles podem ser encontrados em diferentes tipos, alguns dos principais são: motores com escova (*brushed*) e motores sem escova (*brushless*)".

Os motores *brushed* são constituídos por um eixo girante contendo um ou mais enrolamentos de fios conectados a comutadores acionados por escovas em meio a um campo magnético fixo (CONDIT, 2010, apud OLIVEIRA, 2014).

Motores *brushless* utilizam na verdade corrente alternada para produzir um campo magnético girante em enrolamentos fixos que propulsiona um rotor de imãs permanentes (YEDAMALE, 2003 apud OLIVEIRA, 2014). Estas correntes são geradas, no entanto, a partir de circuitos inversores alimentados por tensões elétricas CC.

2.4 PONTE H

Para um controle de velocidade e sentido de rotação, a forma mais utilizada é o acionamento através de circuitos de chaveamento em ponte H (CONDIT, 2010 apud OLIVEIRA, 2014).

Uma ponte H é constituída de quatro elementos de chaveamento dispostos como no circuito da Figura 5. Fechando chaves em posições diametralmente opostas (S1:S4 ou S2:S3) na ponte H, o motor gira em sentido horário ou anti-horário, dependendo do par de chaves ligadas. As outras chaves são mantidas abertas, nesse caso. Por outro lado, fechando chaves paralelas (S1:S3 ou S2:S4), o motor tem uma parada forçada. Enquanto que, se todas as chaves forem abertas, o motor tem uma parada gradual em movimento livre (OLIVEIRA, 2014).

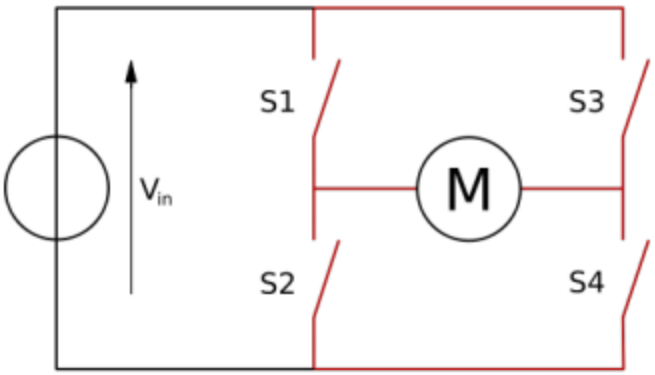

Figura 5: Ponte H.
Fonte: OLIVEIRA, 2014.

2.5 ROS (*ROBOT OPERATING SYSTEM*)

O sistema operacional robótico (ROS) é uma estrutura flexível para escrever software de robô. É uma coleção de ferramentas, bibliotecas e convenções que visam simplificar a tarefa de criar um comportamento robótico complexo e robusto em uma ampla variedade de plataformas robóticas.

Os Autores (BOHREN, ROMERO e GVDHOORN, 2010, 2014, 2017 apud ROSSI, 2019) dizem que o ROS permite executar projetos robóticos de forma individual ou grupal, permite criação de pacotes em múltiplas linguagens, provê serviços de um sistema operacional, controle de dispositivos, comunicação entre processos e é capaz de proporcionar a intercomunicação de hardwares

2.6 *RASPBERRY PI*

De acordo com MONK (2013, apud FREITAS, FRUTOSO e PEREIRA), o *Raspberry Pi*, é um minicomputador que executa principalmente o sistema operacional Linux. Possui portas USB nas quais pode-se conectar um teclado e um *mouse* e uma saída de vídeo HDMI (High-Definition Multimedia Interface) onde pode-se conectar uma TV ou um monitor. É possível utilizar o *Raspberry* Pi de diversas formas, desde que isso também possa ser feito em um computador desktop *Linux*, com algumas limitações. Pode ser utilizado para editar documentos de texto, navegar na Internet e jogar jogos.

2.7 *PWM* (*PULSE WIDTH MODULATION*)

O *PWM* é descrito pelo autor (REIS, 2017) como “uma técnica utilizada para permitir o controle da energia fornecida a equipamentos elétricos, como servomotores e dispositivos de iluminação. Também pode ser usada para codificar mensagens para transmissão”.

BRAGA explica o funcionamento e a utilização do *PWM* da seguinte maneira:

> “Se levarmos em conta que a potência e, portanto, a velocidade de um pequeno motor DC depende da tensão aplicada podemos usar de um artifício interessante para variar essa potência sem, entretanto, modificar a tensão aplicada ao motor. A ideia consiste em trabalhar com o tempo como uma segunda variável no circuito de controle.

> Se aplicarmos ao motor pulsos retangulares que tenham a tensão nominal do motor, mas que durem 50% do tempo, ou seja, um ciclo ativo de 50%, podemos dizer que a potência média do motor será 50% da máxima.
> No entanto, como cada pulso tem a tensão máxima nominal, o motor não sente com a inércia que ocorre quando aplicamos baixas tensões, mantendo seu torque.
> Para aumentar a potência aplicada, obtendo-se maior velocidade basta aumentar a largura do pulso, e para diminuir a velocidade ou a potência aplicada, basta diminuir a velocidade do pulso.
> Em suma, podemos controlar a velocidade, mas mantendo o torque numa faixa mais próxima do máximo, modulando os impulsos aplicados em sua largura, daí o nome dessa técnica amplamente usada nas aplicações de mecatrônica em todos os níveis.
> Veja então que um motor que tenha esse tipo de controle pode girar sem quase perder o torque a partir do zero, conseguindo-se rotações muito baixas, impossíveis de obter com um controle linear".

Os motores DC são controlados pelo PWM, esta técnica permite simularmos valores analógicos de tensão através dos pulsos de sinal digital, utilizando ondas quadradas que variam entre nível lógico alto e nível lógico baixo. Observando a Figura 7, é possível notar pulsos permaneceram 50% e 25% do tempo, portanto o primeiro obteve maior potência aplicada e consequentemente maior velocidade, para diminuir esta velocidade basta diminuir a largura do pulso.

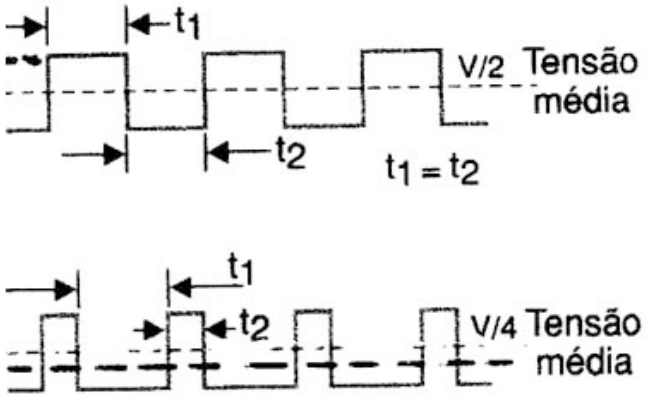

Figura 6: Pulsos *PWM*.

Fonte: BRAGA.

2.8 SLAM

O SLAM (*Simultaneous localization and mapping*) é uma técnica utilizada por robôs para se localizarem e ao mesmo tempo gerarem mapas. "O problema de localização e mapeamento simultâneos é um problema difícil e ainda não está totalmente resolvido,

entretanto, existem boas soluções já empregadas em navegações de diferentes tipos de veículos autônomos, como robô aéreos, terrestres e subaquáticos, produzindo resultados bem satisfatórios. Os principais problemas presentes na utilização dessa técnica estão relacionados ao custo computacional e ao correto relacionamento que deve ser feito entre os novos marcos observados e os já adicionados ao mapa, uma vez que se essa associação for feita" (MACHARET, 2009).

2.9 HECTOR SLAM

De acordo com a Wiki do Hector Slam, "o hector_mapping é uma abordagem SLAM que pode ser usada sem odometria, bem como em plataformas que exibem movimento de rotação / inclinação (do sensor, da plataforma ou de ambas). Ele aproveita a alta taxa de atualização dos sistemas LIDAR modernos, como o Hokuyo UTM-30LX e fornece estimativas de pose 2D na taxa de varredura dos sensores (40Hz para o UTM-30LX). Embora o sistema não forneça a capacidade explícita de fechamento de loop, é suficientemente preciso para muitos cenários do mundo real. O sistema foi usado com sucesso em robôs terrestres não tripulados, veículos de superfície não tripulados, dispositivos portáteis de mapeamento e dados registrados de UAVs quadrotor".

3. MATERIAIS E MÉTODOS

Esta seção trata das informações referentes aos dispositivos de *hardware* e as tecnologias utilizadas no desenvolvimento do *software* da aplicação. Trata também da organização do projeto, de como foi realizada a modelagem dos fluxos tendo como base os requisitos do sistema.

3.1 MATERIAIS

Para a construção física e lógica do protótipo alguns materiais foram necessários, aqui são apresentados esses materiais e como foram utilizados.

3.1.1 *HARDWARE*

Esta seção trata de alguns componentes físicos do projeto: motores DC, pontes H, LIDAR, *Raspberry Pi*, fonte de alimentação e a câmera para o *Raspberry Pi*.

3.1.1.1 Motor DC

O motor utilizado é o szdlt hm-gm-25-370 possui 6 pinos de conexão e as seguintes especificações:

Tabela 1: Especificações do motor DC.

CARACTERÍSTICA	**VALOR**
Tensão nominal	9 V – DC
Velocidade sem carga	11.500 RPM (max)
Corrente sem carga	180 mA (max)
Pino 1 (Cabo verde)	Segundo sensor de efeito *Hall*
Pino 2 (Cabo laranja)	Primeiro sensor de efeito *Hall*
Pino 3 (Cabo amarelo)	GND do sensor de efeito *Hall*
Pino 4 (Cabo branco)	VCC do sensor de efeito *Hall*
Pino 5 (Cabo vermelho)	GND
Pino 6 (Cabo preto)	VCC

Fonte: Os Autores.

Figura 7: Motor DC.

Fonte: Os autores.

3.1.1.2 L298N

A ponte H é utilizada para determinar para qual sentido os motores rotacionarão, no protótipo são utilizadas duas pontes, cada uma é responsável pelo acionamento de dois motores, pois possuem dois canais para operação. Ela recebe sinais vindos da *Raspberry* e então realiza o chaveamento de acordo com os comandos recebidos.

Os dois circuitos integrados utilizados são do modelo L298N e possuem as seguintes características:

Tabela 2: Características L298N.

SÍMBOLO	**PARÂMETRO**	**VALOR (máximo)**
Vs	Tensão de alimentação	50 V
Vss	Tensão de alimentação lógica em nível alto	7 V
	Tensão de alimentação lógica em nível baixo	1,5 V
Vi, Ven	Tensão de entrada e de ativação	-0,3 até 7 V
I	Corrente CC	4 A
Ptot	Potência Total Dissipada	25 W
Top	Temperatura de Operação da junção	-25 ºC até 130 ºC

Fonte: Adaptado de STMICROELECTRONICS.

Sua relação de entradas e saídas, juntamente com sua descrição:

Tabela 3: Relação de entradas e saídas.

NOME	**FUNÇÃO**
Sentido A e B	Possuem um resistor de carga conectado ao terro para controlar a corrente
Input 1 e 2	Entradas do sentido A
Input 3 e 4	Entradas do sentido B
Out 1 e 2	Saídas do sentido A

Out 3 e 4	Saídas do sentido B
Vs	Tensão de alimentação nas saídas
Vss	Tensão de alimentação para os terminais lógicos
GND	Terra
EnA e EnB	Habilitam as entradas A e B respectivamente

Fonte: Adaptado de STMICROELECTRONICS.

Seu esquemático com a pinagem descrita na tabela anterior está na Figura 8 e a imagem do componente na Figura 9:

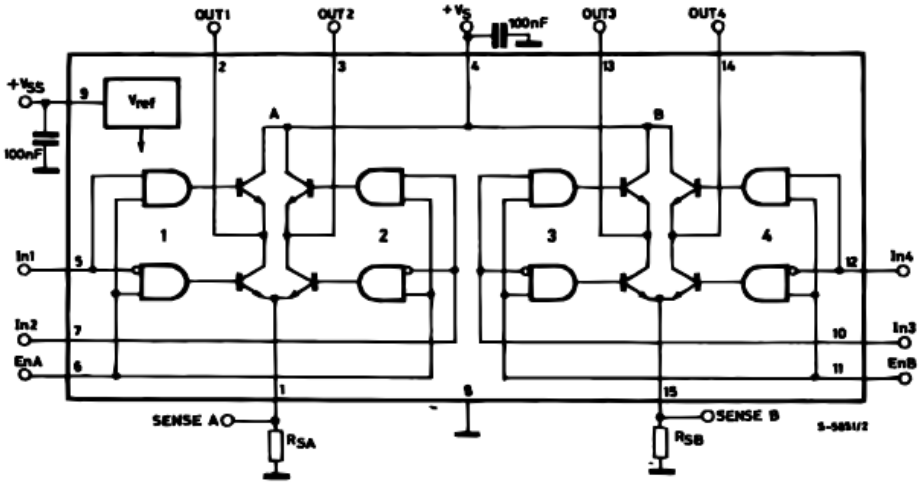

Figura 8: Esquemático L298N.

Fonte: STMICROELECTRONICS.

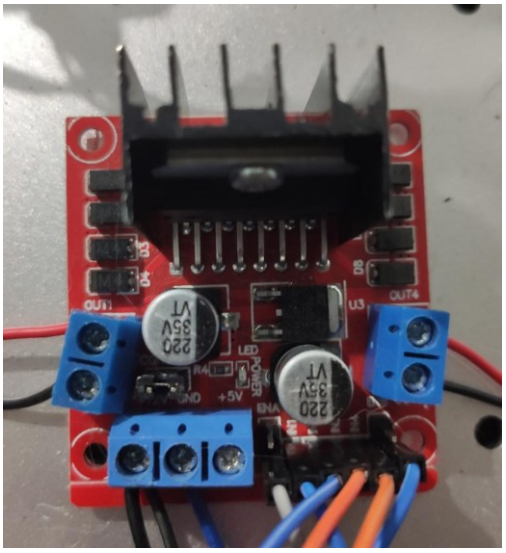

Figura 9: Ponte H.

Fonte: O Autores.

3.1.1.3 RPLIDAR-A1

O LIDAR é um sensor a laser usado para medição de distância, o modelo utilizado neste projeto realiza a varredura onidirecional em 360º do ambiente, as medidas são feitas com base no princípio da triangulação. De acordo com DEHONG (2017, apud ROSSI) o princípio da triangulação consiste em “o sensor emite um feixe LASER que é refletido e captado por um sensor fotossensível presente no equipamento. Dessa forma, quando o objeto se aproxima ou distância do emissor, o feixe de luz refletido toca o sensor fotossensível em outra posição, onde, pelo princípio de triangulação geométrica é estimada a posição do objeto”. A imagem do LIDAR está exposta a seguir, juntamente com a Tabela 4 que mostra suas características.

Figura 10: RPLIDAR-A1.

Fonte: Os Autores.

Tabela 4: Características RPLIDAR-A1.

PARÂMETRO	VALOR (máximo)
Alcance	12 m
Amostrar / Segundo	8000
Ângulo de escaneamento	360º
Taxa de varredura	2 Hz – 10 Hz
Tensão de alimentação	5 V

Fonte: Os Autores.

3.1.1.4 *Raspberry Pi*

Conforme mencionado anteriormente, o *Raspberry* minicomputador que utiliza na maioria das vezes um sistema operacional *Linux*, seus recursos são mais limitados que um computador comum, algumas de suas características são:

Tabela 5: Tabela de característica da Raspberry.

ESPECIFICAÇÕES	CARACTERÍSTICA
Processador	Broadcom BCM2837B0, Cortex-A53 (ARMv8) SoC de 64 bits a 1.4GHz
Alimentação	5V / 2.5ª DC
Memória	SDRAM LPDDR2 de 1 GB
WIFI e Bluetooth	LAN sem fio de 2,4 GHz e 5 GHz IEEE 802.11.b / g / n / ac, Bluetooth 4.2
Portas USB	4 portas USB 2.0
GPIO	GPIO de 40 pinos

Fonte: Os Autores.

Na Figura 11 é exibido o *Raspberry Pi 3,* que foi utilizado neste projeto. A Figura 12 mostra a numeração dos pinos, que vão de 1 a 40 e suas *labels*. Os pinos que podem ser programados para entrada ou saída de dados são os que possuem a *label* GPIO seguido de sua numeração específica.

Figura 11: *Raspberry* Pi.

Fonte: *RASPBERRYPI.*

Figura 12: Pinagem do *Raspberry*.

Fonte: MICROSOFT.

3.1.1.5 UNIPOWER UP1250

A bateria utilizada para alimentar o sistema é a UNIPOWER UP1250 que possui as seguintes especificações técnicas:

Tabela 6: Características da Fonte.

CARACTERÍSTICA	VALOR
Tensão da Bateria	12 V
Capacidade	5 Ah
Resistência Interna	21 mΩ
Peso	1,7 Kg

Fonte: Os Autores.

Figura 13: Bateria.

Fonte: Os Autores.

3.1.1.6 *Raspberry Pi Camera*

A *Raspberry Pi Camera* foi feita especificamente para aplicações do *Raspberry Pi*, a Figura 14 exibe este componente e suas especificações estão na tabela a seguir:

Tabela 7: Especificações *Raspberry Pi Camera*.

ESPECIFICAÇÃO	DESCRIÇÃO
Tamanho	25x24 mm (sem o cabo)
Tamanho do Sensor	3,67 x 2,74 mm
Vídeo	1080p a 30 fps
Campo de visão	2,0 x 1,33 m
Ângulo de visão	54 x 41 graus

Fonte: Os Autores.

Figura 14: *Raspberry Pi Camera*.

Fonte: Thomsen.

3.2 METODOLOGIA

Após o apanhado histórico que ressaltou a importância da utilização de robôs em nosso cotidiano, foram estudados os componentes do projeto, com o intuído de entender o funcionamento de cada um destes elementos. Os estudos citados forneceram a base de conhecimento necessária para realizar a implementação deste sistema.

Este documento aborda a modelagem do sistema a partir dos aspectos de *hardware* e *software* e trata de toda sua implementação. A seguir está exposto o diagrama de blocos que representa a visão geral do sistema:

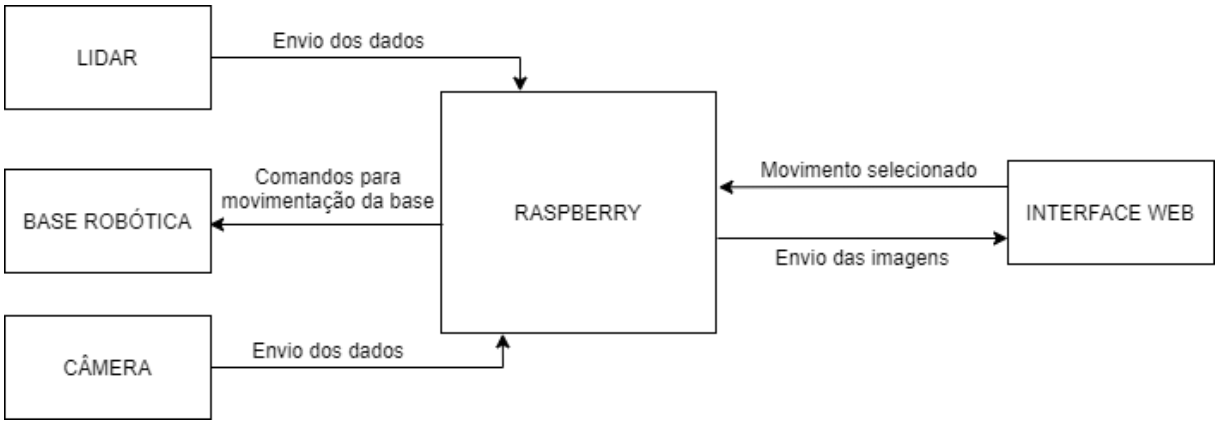

Figura 15: Diagrama de blocos - Visão Geral do Sistema.

Cada um dos blocos possui diferentes aplicações:

- O LIDAR é responsável pela leitura dos dados provenientes do ambiente externo e então os envia para o *RASPBERRY*;
- A parte móvel, representada pela BASE ROBÓTICA, recebe os comandos vindos da INTERFACE WEB por intermédio do *RASPBERRY* e realiza o movimento;
- O *RASPBERRY* recebe e processa os dados captados pelo LIDAR e pela câmera, então os encaminha para a INTERFACE WEB utilizando um protocolo de comunicação HTTP, recebe também comandos vindos da INTERFACE WEB e com base neles envia as instruções para a BASE ROBÓTICA;
- A INTERFACE WEB realiza o intermédio entre o usuário e o protótipo, tem a responsabilidade de transmitir para o *RASPBERRY* o movimento selecionado pelo utilizador, através das teclas W, S, A, D e deve exibir as informações processadas que recebe como resposta;

3.2.1 Requisitos de *Hardware*

Para que o protótipo do projeto possa ser desenvolvido, alguns requisitos de hardware foram seguidos, o quadro a seguir exibe estes requisitos:

IDENTIFICAÇÃO	COMPONENTE	DESCRIÇÃO
RH 01	*Raspberry Pi* 3	Utilizado para as seguintes funções: 1. Montar a imagem do mapeamento com os dados do LIDAR; 2. Montar a imagem da câmera com os dados da câmera; 3. Enviar comandos para a movimentação da base robótica.
RH 02	LIDAR	Capturar os dados do ambiente para gerar a imagem do mapeamento.
RH 03	Câmera	Capturar imagens para exibição na interface web.
RH 04	Ponte H	Realizar o interfaceamento entre a *Raspberry* e os motores.
RH 05	Motor DC	Movimentar o protótipo.
RH 06	Base Robótica	Representa a parte móvel do projeto: 1. Plataforma metálica; 2. Esteiras; 3. Motores DC.
RH 07	Bateria	Alimentar o sistema.
RH 08	Circuito Regulador de Tensão	Circuito responsável por regular a tensão para os componentes de *hardware*.

Quadro 1: Requisitos de Hardware.

Fonte: Os Autores.

Conforme o Quadro 1, nota-se que o protótipo é composto pelos seguintes itens: Base robótica, motor DC, ponte H, LIDAR, *RaspBerry Pi* 3, circuito regulador de tensão e uma bateria, conforme mostra o diagrama de blocos a seguir:

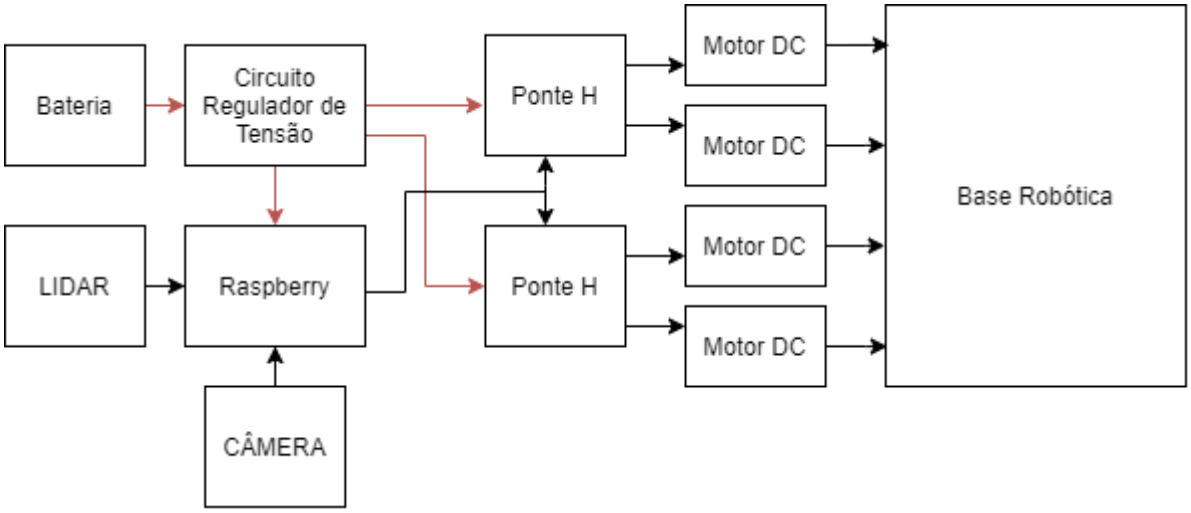

Figura 16: Diagrama de blocos do protótipo.

Fonte: Os Autores.

3.2.2 Especificações de *Hardware*

A base robótica representa a parte móvel do sistema, ela é composta pelos itens que estão relacionados na tabela a seguir:

Tabela 8: Parte móvel do sistema.

ITEM	FUNÇÃO
Plataforma metálica	É utilizada como base para que os componentes que constituem o protótipo sejam acoplados.
Esteiras	Quatro esteiras que reproduzem a rotação dos motores e por consequência movimentam o protótipo.
Motores DC	Quatro motores são utilizados para movimentar as esteiras.

Fonte: Os Autores.

Através da imagem a seguir podemos ver a base robótica descrita na tabela anterior:

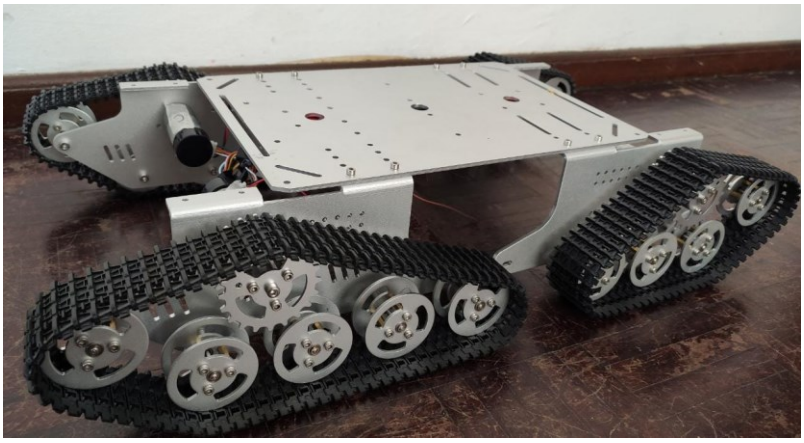

Figura 17: Base Robótica.

Fonte: Os Autores.

Para que o *Raspberry* possa interagir com as pontes H são utilizados seus pinos de GPIO (Figura 12), eles podem ser programados para receber ou enviar dados. Neste caso, são utilizados para enviar comandos para as pontes H.

As pontes H (L298N) recebem os comandos do *Raspberry* através de seus pinos de entrada (IN1, IN2, IN3, IN4, EnA e EnB).

A tabela a seguir mostra a relação entre a saída dos pinos de GPIO e as entradas das pontes H e a Figura 18 exibe as conexões:

Tabela 9: Relação dos terminais de GPIO e das pontes H.

NÚMERO DO GPIO	**RASPBERRY (TERMINAL)**	**ENTRADA PONTE H**	**PONTE H (TERMINAL)**
17	11	IN1	05
27	13	IN2	07
02	3	EnA	06
22	15	IN3	10
23	16	IN4	12
07	26	EnB	11
20	38	IN1	05
21	40	IN2	07
26	37	EnA	06
16	36	IN3	10
19	35	IN4	12
08	24	EnB	11

Fonte: Os Autores.

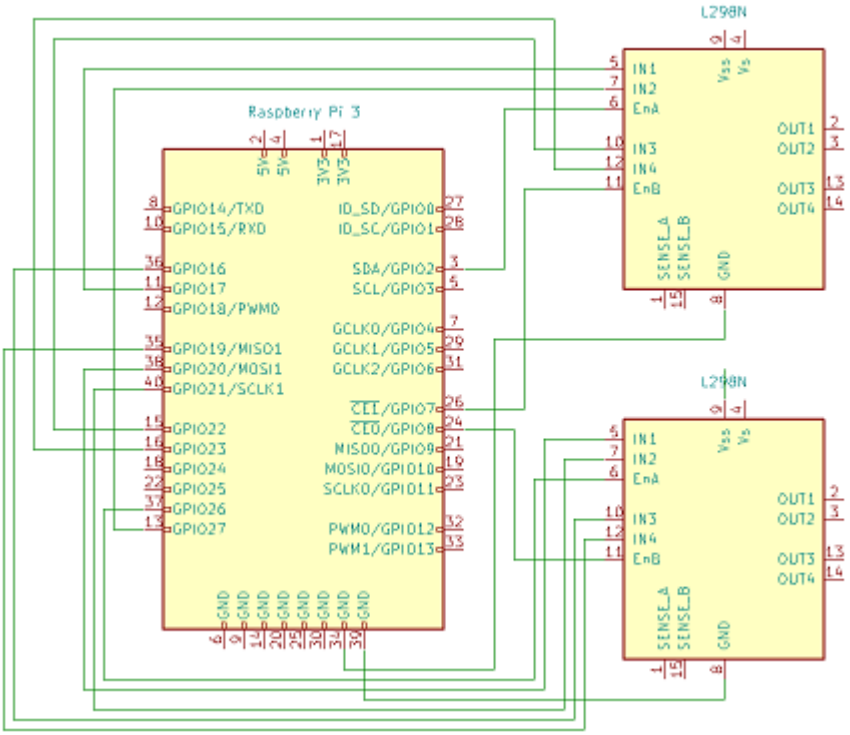

Figura 18: Conexões entre *Raspberry* e pontes H.

Fonte: Os Autores.

Após o recebimento dos comandos, as pontes H realizam o chaveamento necessário para o acionamento dos motores, então o protótipo é movimentado.

A ativação dos motores é feita através das saídas (OUT1, OUT2, OUT3 e OUT4) de ambas as pontes H (L298N), como mostra a Figura 19 e a tabela a seguir:

Tabela 10: Relação dos terminais das pontes H com os motores.

SAÍDA PONTE H	PONTE H (TERMINAL)	MOTOR DC (TERMINAL)
OUT1	02	01
OUT2	03	02
OUT3	13	02
OUT4	14	01
OUT1	02	01
OUT2	03	02
OUT3	13	02
OUT4	14	01

Fonte: Os Autores.

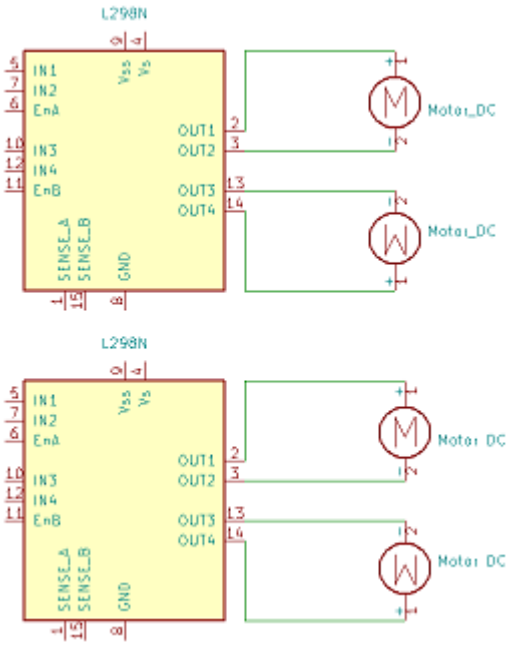

Figura 19: Conexões entre pontes H e os motores DC.

Fonte: Os Autores.

Nota-se na Figura 19 que as saídas OUT1 e OUT4 são ligadas na conexão positiva dos motores e as saídas OUT2 e OUT3 estão na conexão negativa dos motores, sendo assim, é possível realizar sua rotação em ambos os sentidos.

Para alimentar os motores são necessários 9V em cada motor e na *Raspberry* são necessários 5V, então foi criado um circuito regulador de tensão, para garantir estes valores. Este circuito recebe 12V vindos da bateria e utilizando o LM7805 regula 5V para a *Raspberry*, através de sua entrada micro USB e usando dois LM7809 são destinados 9V aos motores por intermédio das pontes H, este circuito está exposto na figura a seguir:

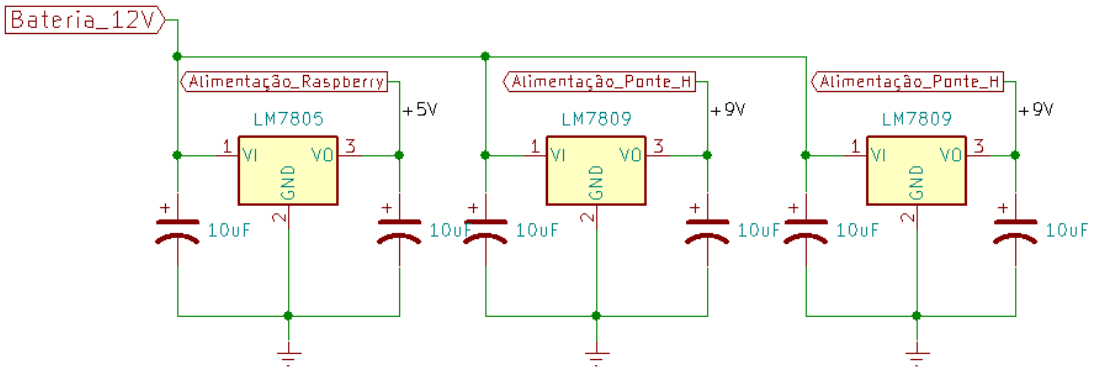

Figura 20: Esquemático do Circuito Regulador de Tensão.

Fonte: Os Autores.

Ambos os reguladores de tensão possuem três pinos, que neste caso ficam na mesma posição independentemente do modelo.

Tabela 11: Relação de pinos LM7805 e LM7809.

PINO	FUNÇÃO
1	Entrada
2	GND
3	Saída

Fonte: Os Autores.

O circuito regulador de tensão implementado está na Figura 21.

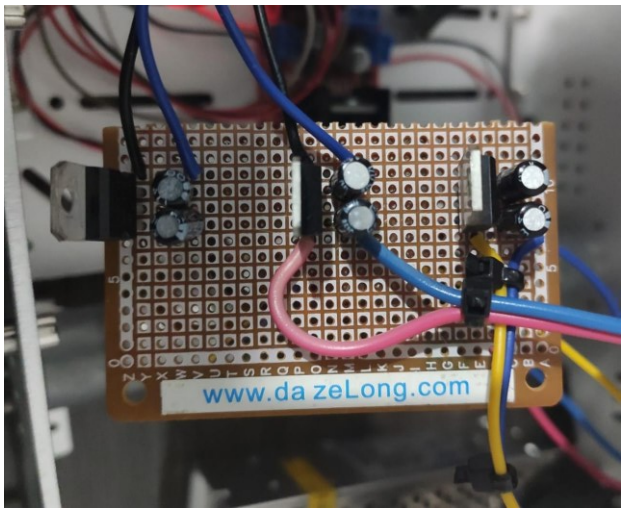

Figura 21: Circuito Regulador de Tensão.

Fonte: Os Autores.

O LIDAR é conectado na *Raspberry* via entrada USB, então não há a necessidade de utilização de uma fonte externa para alimentação, o mesmo ocorre para a câmera, porém ela se conecta ao *Raspberry* via cabo *flat*.

O esquemático do *hardware* contendo os componentes (*Raspberry*, pontes H, motores DC e o regulador de tensão) está exposto na Figura 22.

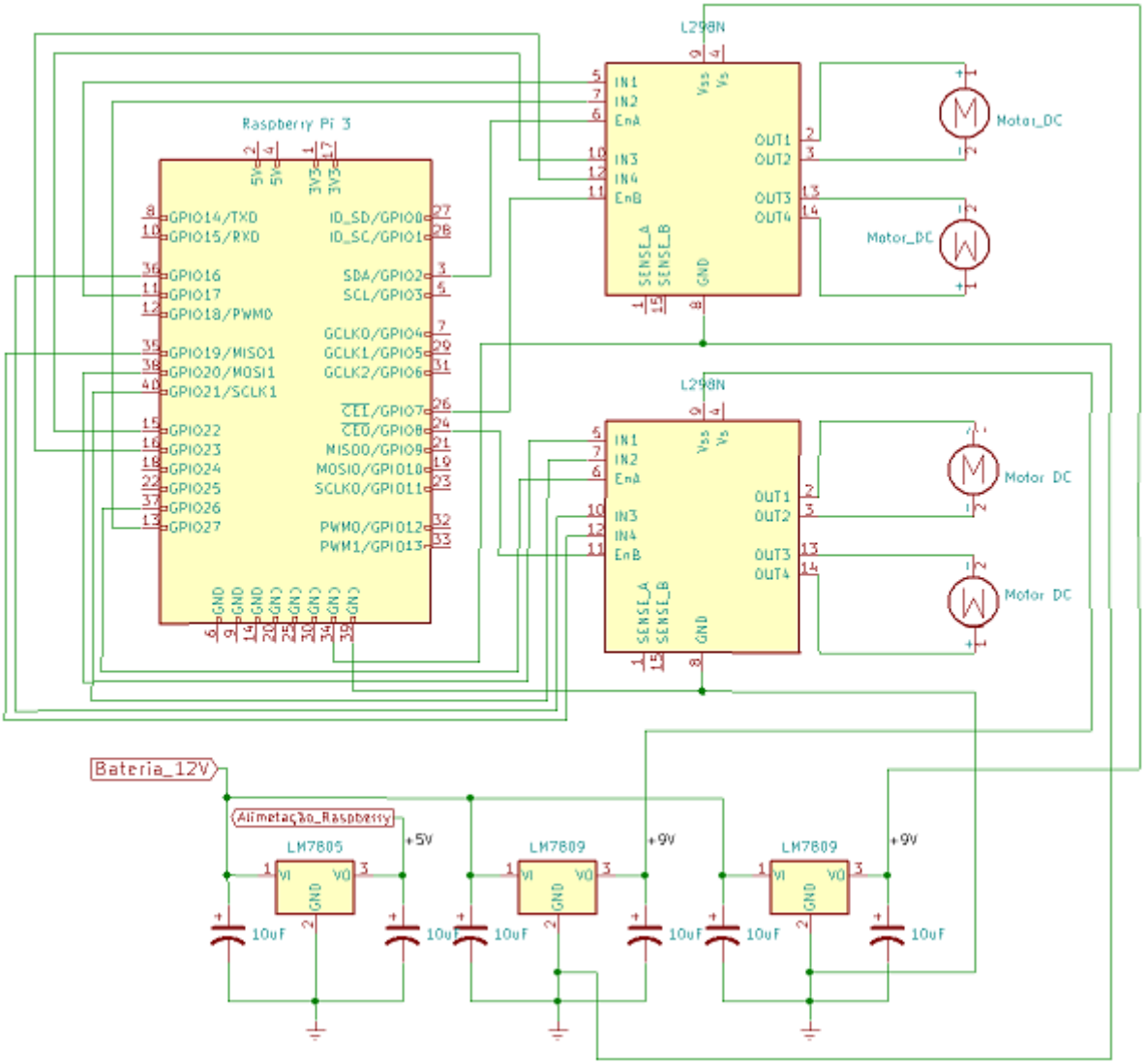

Figura 22: Esquemático do hardware.

Fonte: Os Autores.

3.2.3 *SOFTWARE*

Nesta seção são abordados os itens relacionados aos *softwares* implementados no projeto, itens que nos mostram como é realizado o controle do protótipo e o processamento dos dados.

3.2.3.1 Requisitos Funcionais

Nesta etapa são apresentados os requisitos funcionais e não funcionais do sistema, através dos quadros a seguir:

Nº Requisito	RF 01	Título	Selecionar a direção do movimento.
Descrição	O usuário seleciona a direção para movimentação.		
Definição de Casos de Uso			
UC01	Título	Selecionar a direção do movimento.	
	Fluxo Principal	O usuário seleciona através das teclas (W, S, A, D) a direção em que o protótipo deve se mover. W – Frente; S – Trás; A – Esquerda; D – Direita;	
	Fluxo(s) Alternativo(s)	Não realiza o movimento.	

Quadro 2: Requisito Funcional - Envio de comandos para o protótipo.

Fonte: Os Autores.

Nº Requisito	RF 02	Título	Enviar o comando para o protótipo
Descrição	Envio do movimento selecionado pelo usuário		
Definição de Casos de Uso			
UC02	Título	Enviar o comando para o protótipo	
	Fluxo Principal	Sistema envia o comando para movimentação do protótipo com base na escolha do usuário.	
	Fluxo(s) Alternativo(s)	Não envia o comando.	

Quadro 3: Requisito Funcional – Movimentação do protótipo.

Fonte: Os Autores.

Nº Requisito	RF 03	Título	Movimentar
Descrição	Movimentação do protótipo na direção escolhida pelo usuário.		
Definição de Casos de Uso			
UC03	Título	Movimentar	
	Fluxo Principal	Opções de movimentação: 1. Frente; 2. Trás; 3. Esquerda; 4. Direita.	
	Fluxo(s) Alternativo(s)	Não realiza o movimento.	

Quadro 4: Requisito Funcional – Tratamento dos dados do LIDAR.

Fonte: Os Autores.

Nº Requisito	RF 04	Título	Montar o mapeamento
Descrição	Montagem do mapeamento com dados coletados pelo LIDAR.		
Definição de Casos de Uso			
UC04	Título	Montar o mapeamento	
	Fluxo Principal	Fluxograma da Figura 31.	
	Fluxo(s) Alternativo(s)	Não realiza a montagem do mapeamento.	

Quadro 5: Requisito Funcional - Interação com o usuário.

Fonte: Os Autores.

Nº Requisito	RF 05	Título	Montar a imagem da câmera
Descrição	Montagem da imagem com os dados da câmera		
Definição de Casos de Uso			
UC05	Título	Montar a imagem da câmera	
	Fluxo Principal	Fluxograma da Figura 29.	
	Fluxo(s) Alternativo(s)	Não monta a imagem	

Quadro 6: Requisito Funcional - Exibir as imagens.

Fonte: Os Autores.

Nº Requisito	RF 06	Título	Exibir o mapeamento na interface web
Descrição	Exibe a imagem do mapeamento pronta na interface web		
Definição de Casos de Uso			
UC06	Título	Exibir o mapeamento na interface web	
	Fluxo Principal	Traz a imagem do mapeamento do diretório do *Rviz*.	
	Fluxo(s) Alternativo(s)	Não exibe o mapeamento.	

Quadro 7: Requisito Funcional - Varredura do ambiente.

Fonte: Os Autores.

Nº Requisito	RF 07	Título	Exibir a imagem da câmera na interface web
Descrição	Exibe a imagem da câmera na interface web		
Definição de Casos de Uso			
UC07	Título	Exibir a imagem da câmera na interface web	
	Fluxo Principal	Traz a imagem do repositório da câmera.	
	Fluxo(s) Alternativo(s)	Não exibe a imagem.	

Quadro 8: Requisito Funcional - Envio dos dados do LIDAR.

Fonte: Os Autores.

3.2.3.2 Requisitos Não Funcionais

Nº Requisito	RNF 01	Título	Instalação do ROS Desktop
Descrição	Para o sistema funcionar, a versão *Melodic* do ROS deve ser instalada.		

Quadro 9: Requisito Não Funcional - Instalação do ROS.

Fonte: Os Autores.

Nº Requisito	RNF 02	Título	Clonagem do Repositório da Slamtec
Descrição	Uso do repositório do fabricante do LIDAR para acessar suas funcionalidades		

Quadro 10: Requisito Não Funcional - Repositório Slamtec.

Fonte: Os Autores.

Nº Requisito	RNF 03	Título	Instalação do NodeJs
Descrição	Instalação do NodeJs na versão 14.5.0 para criar um ambiente de execução *JavaScript* assíncrono orientado a eventos.		

Quadro 11: Requisito Não Funcional - Instalação do NodeJs.

Fonte: Os Autores.

Nº Requisito	RNF 04	Título	Instalação do *JavaScript*
Descrição	Instalação da versão 14.5.0 do *JavaScript*, para manipulação os dados em tela.		

Figura 23: Requisito Não Funcional - Instalação do *JavaScript*.

Fonte: Os Autores.

Nº Requisito	RNF 05	Título	Instalação do *Express*
Descrição	Instalação da versão 4.17.1 do *Express* para uso de ferramentas para aplicativos web.		

Quadro 12: Requisito Não Funcional - Instalação do Express.

Fonte: Os Autores.

Nº Requisito	RNF 06	Título	Instalação do *React*
Descrição	Instalação da versão 16.13.1 do React - Biblioteca *JavaScript* para interfaces de usuário		

Quadro 13: Requisito Não Funcional - Instalação do React

Fonte: Os Autores.

Nº Requisito	RNF 07	Título	Instalação do *Rviz*
Descrição	Instalação da versão 1.13.12 do *Rviz*, para a montagem do mapeamento.		

Quadro 14: Requisito Não Funcional – Instalação do *Rviz*.

Fonte: Os Autores.

Nº Requisito	RNF 08	Título	Tempo de autonomia da aplicação
Descrição	O sistema deve manter sua estabilidade durante o uso		

Quadro 15: Requisito Não Funcional - Estabilidade do sistema.

Fonte: Os Autores.

Nº Requisito	RNF 09	Título	Tempo de resposta do sistema
Descrição	O sistema deve responder ao movimento do protótipo selecionado pelo usuário com o menor tempo de *delay* possível.		

Quadro 16: Requisito Não Funcional - Sistema responsivo.

Fonte: Os Autores.

Nº Requisito	RNF 10	Título	Segurança dos Dados
Descrição	Os dados trafegados pelo sistema não possuem criptografia.		

Quadro 17: Requisito Não Funcional – Segurança dos dados.

Fonte: Os Autores.

3.2.3.3 Caso de Uso

O diagrama de caso de uso a seguir nos mostra como o usuário interage com o sistema e a base robótica.

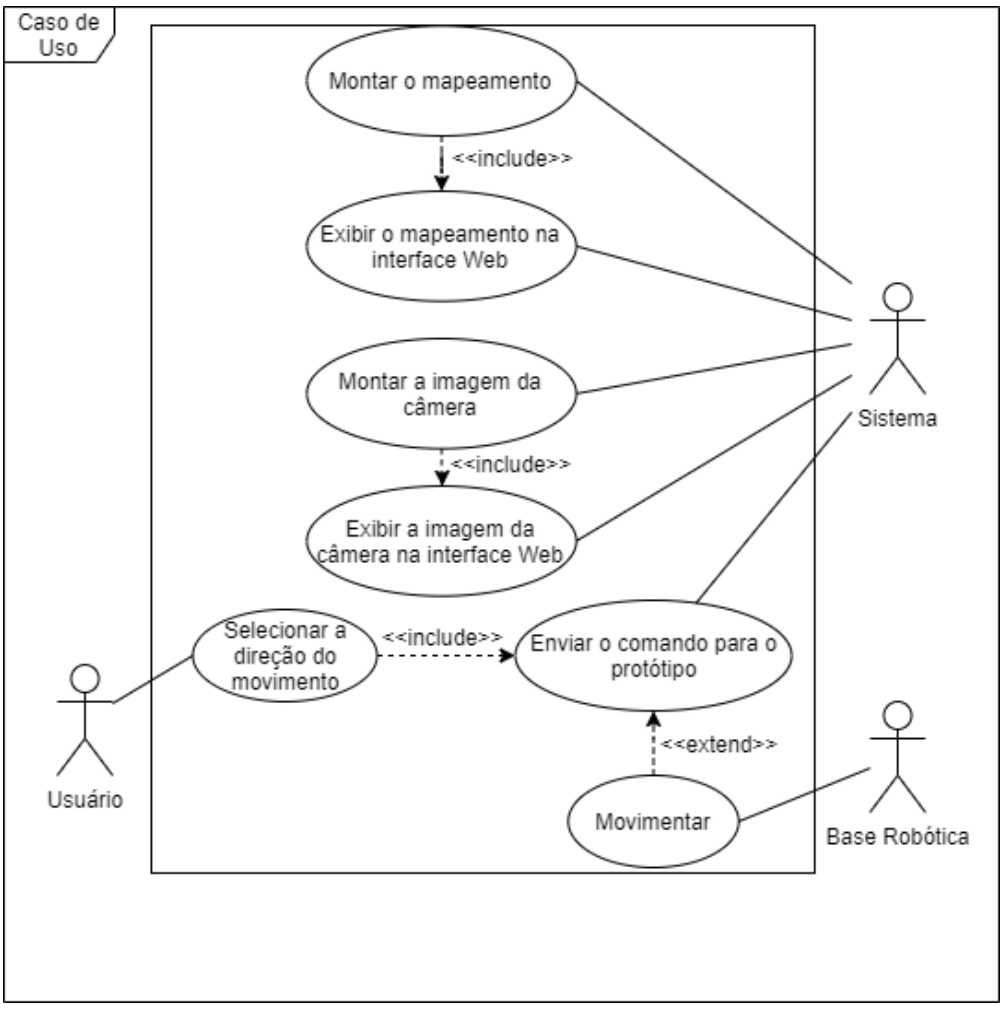

Figura 24: Diagrama Caso de Uso do sistema.

Fonte: Os Autores.

3.2.3.4 Interface Web

A interface criada é composta por seis elementos, cada um com funções diferentes. Podemos ver estes elementos na Figura 25, os principais componentes são os seguintes:

1. Componente do Mapeamento:
    a. É o componente que busca a imagem mapeada pelo LIDAR e plota na interface atualizando os frames enquanto a interface está aberta;
2. Componente da Câmera
    a. Componente que requisita e plota a imagem que está sendo capturada pela câmera da *Raspberry* pelo comando *raspistill*.
3. Componente de *Log*
    a. Componente responsável por exibir na tela os movimentos que estão sendo selecionados pelo usuário através das teclas anteriormente citadas.

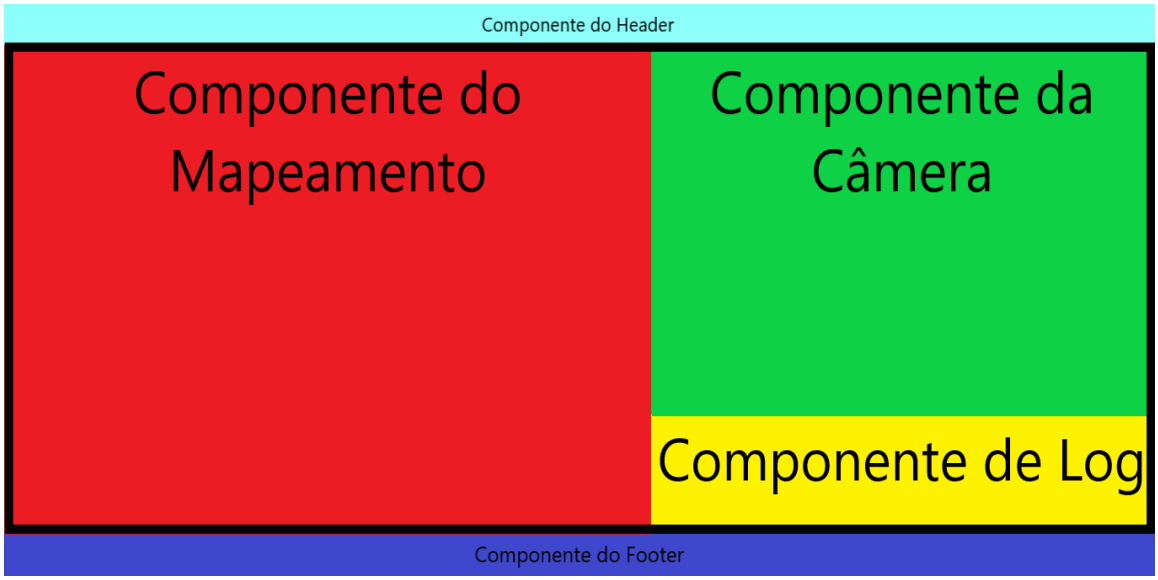

Figura 25: Composição da Interface Web.

Fonte: Os Autores.

A Figura 26 apresenta a interface *WEB* desenvolvida, com os componentes de tela anteriormente citados.

Figura 26: Inteface WEB.

Fonte: Os Autores.

Para a interface exibir a imagem do mapeamento, alguns passos devem ser seguidos, após a inicialização da API e da instância do *Express* App, a interface começa a esperar pelas requisições, a interface web ao receber a requisição faz a busca dos últimos mapeamentos exibidos pelo *Rviz* e então converte a imagem para base64, o fluxograma a seguir ilustra esta descrição.

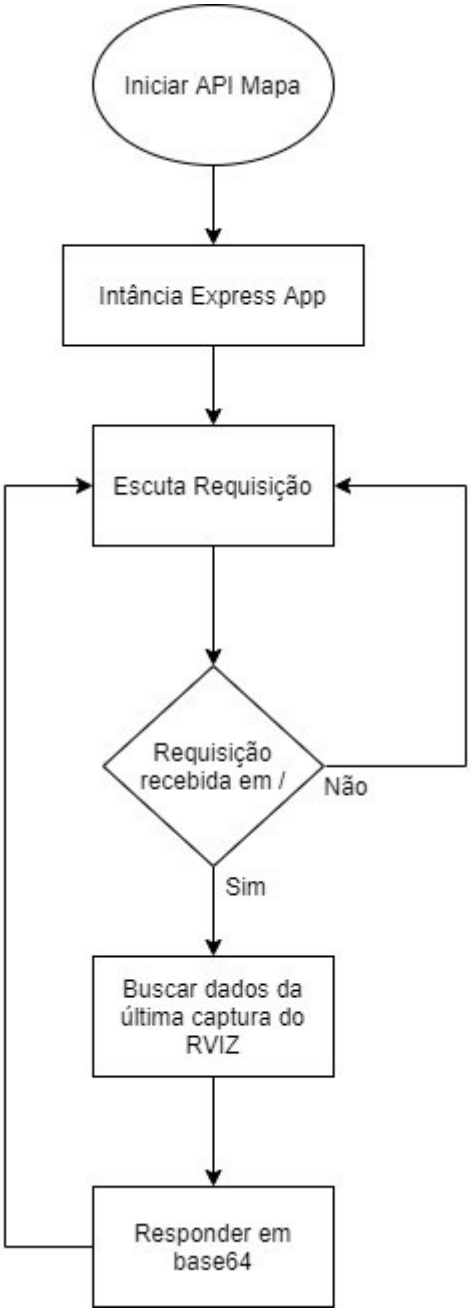

Figura 27: Busca do mapeamento pela interface web.

Fonte: Os Autores.

3.2.3.5 Uso da câmera

Para uso da câmera, é utilizado a linha de comando fornecida pelo sistema operacional, o *raspistill*. Porém o mesmo apresenta algumas limitiações na captura de vídeo em tempo real. Pelo fato de não ter uma ferramenta direcionada a *streaming* da vídeo, é indicado que faça um *timelapse* das imagens capturadas, e assim temos uma sequências de imagens. O comando utilizado para começar e configurar o *timelapse* é o mostrado na figura abaixo:

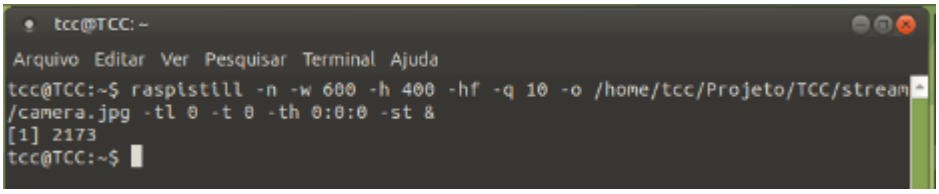

Figura 28: Comando para utilizar o *timelapse* das imagens.

Fonte: Os Autores.

Na linha de comando são passados alguns parâmetros de configurações para que funcione coerentemente com o sistema, esses parâmetros estão na tabela a seguir:

Tabela 12: Parâmetros do *Raspistill*.

COMANDO *RASPISTILL*	
PARÂMETRO	**DESCRIÇÃO**
-n	Parâmetro para não abrir o *preview* localmente.
-w	Parâmetro para especificar o comprimento da imagem em *pixels*.
-h	Parâmetro para especificar a altura da imagem em *pixels*.
-hf	Parâmetro para inverter a imagem horizontalmente.
-g	Parâmetro para especificar que a qualidade da imagem.
-o	Parâmetro para especificar o local e a extensão que a imagem capturada deve ficar.
-tl	Parâmetro para especificar que a captura e gravação da imagem do *timelapse* deve ser o mais rápido possível, isso se entende por conta do digito 0 (zero).
-th	Parâmetro passado para configurar o *thumbnail* onde a configuração é a seguinte (x:y:qualidade).

-st	Força o recalculo das estatísticas na captura feita.

Fonte: Os Autores.

E paralelamente com a ativação da câmera, é iniciado a API que fica esperando alguma consulta para disponibilizar o valor da imagem em base64, demonstrado na figura abaixo:

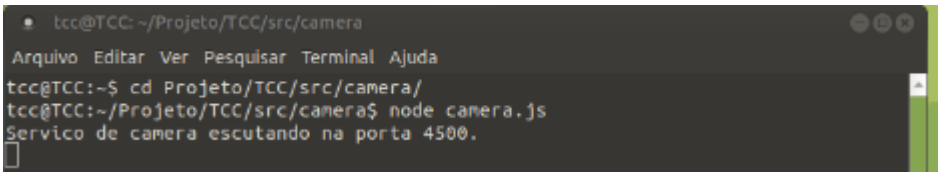

Figura 29: Inicialização da API da câmera.

Fonte: Os Autores.

A interface web realiza uma consulta *GET* na API da câmera onde o retorno em JSON contém a imagem codificada em base64, que é colocado na *tag* <img> do HTML configurando com o *source* coletado.

Após a inicialização da API da câmera, é realizada a instancia de um *Express* APP, para que sejam importados os módulos de aplicação com as funções necessárias.

Com isso é configurado um processo através de um dos módulos importados no *Express*, o "*listen*", para que acione uma porta de entrada fazendo assim o processo criado aguardar uma requisição HTTP.

Com a requisição HTTP recebida, o processo que estava aguardando, realiza uma busca no local onde está sendo salvo a imagem pelo *raspistill*. Assim é pego o valor da imagem e realiza o *encode* de base64.

Tendo o valor da imagem base64, o processo responde a requisição para a origem com o valor obtido.

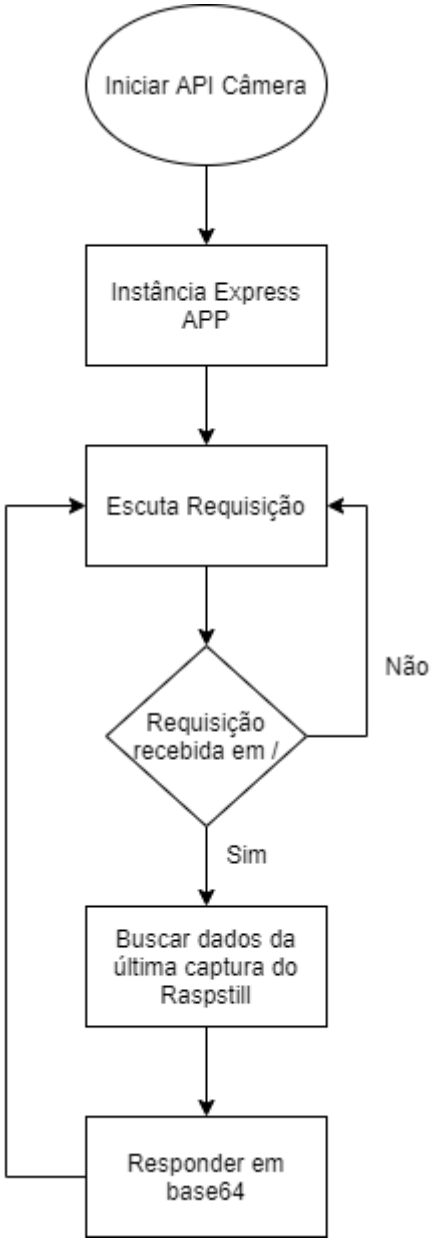

Figura 30: Fluxograma do uso da câmera.

Fonte: Os Autores.

3.2.3.6 Mapeamento

Para utilização do *RPLidar* deve-executar o *script* de inicialização do mesmo, será feita uma verificação se a porta USB onde ele está conectado tem liberação de leitura e escrita, caso tenha, inicia o nó /rplidarNode e inicia o tópico /scan.

Figura 31: Fluxograma de Execução do LIDAR.

Fonte: Os Autores.

Para realização do mapeamento deve ser inicializado o *script* com as configurações do *RPLidar,* será feita a validação da execução do /*rplidarNode e /scan,* será feita a inicialização do nó /*hector_mapping,* do tópico /*map* e será executado o *software Rviz* para a exibição gráfica do mapeamento.

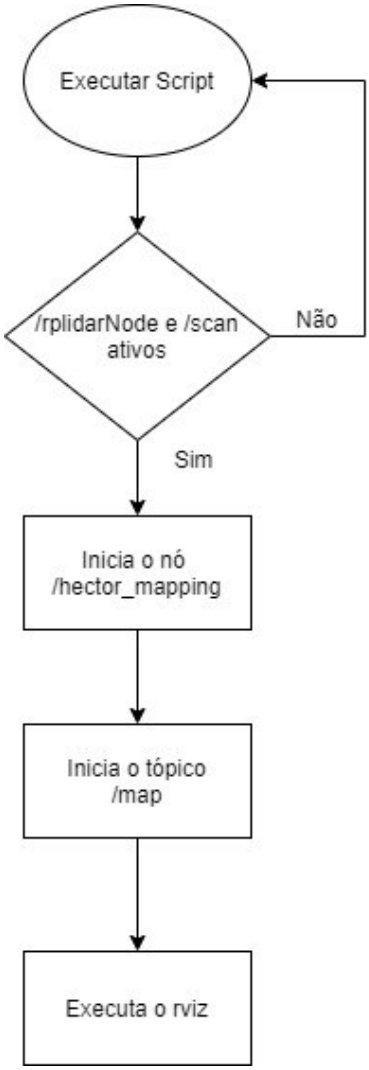

Figura 32: Fluxograma da criação do mapeamento.

Fonte: Os Autores.

3.2.3.7 Movimentação do Protótipo

O processo para a movimentação da base robótica através da interface *web* é baseado em três etapas, onde a primeira é a necessidade da interação do usuário que deve realizar a ação de apertar uma tecla de comando referente ao movimento. As teclas de movimento são W para movimentar para frente, S para movimentar para trás, A para girar para esquerda e D para girar para a direita, conforme mostra a Figura 33.

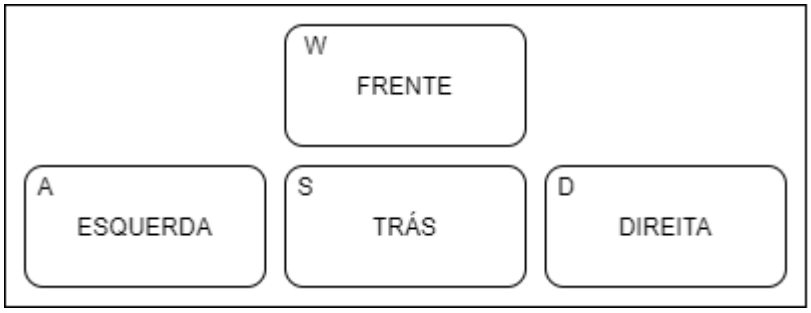

Figura 33: Teclas de comando e a direção do movimento.

Fonte: Os Autores.

A segunda etapa consiste no reconhecimento da tecla de ação, após sua validação a requisição é enviada para a API movimentar_base.js. A terceira etapa inicia um *script* em python passando a informação de direção do movimento, realizando então a movimentação e finalizando o *script*, as três etapas estão detalhadas no fluxograma a seguir:

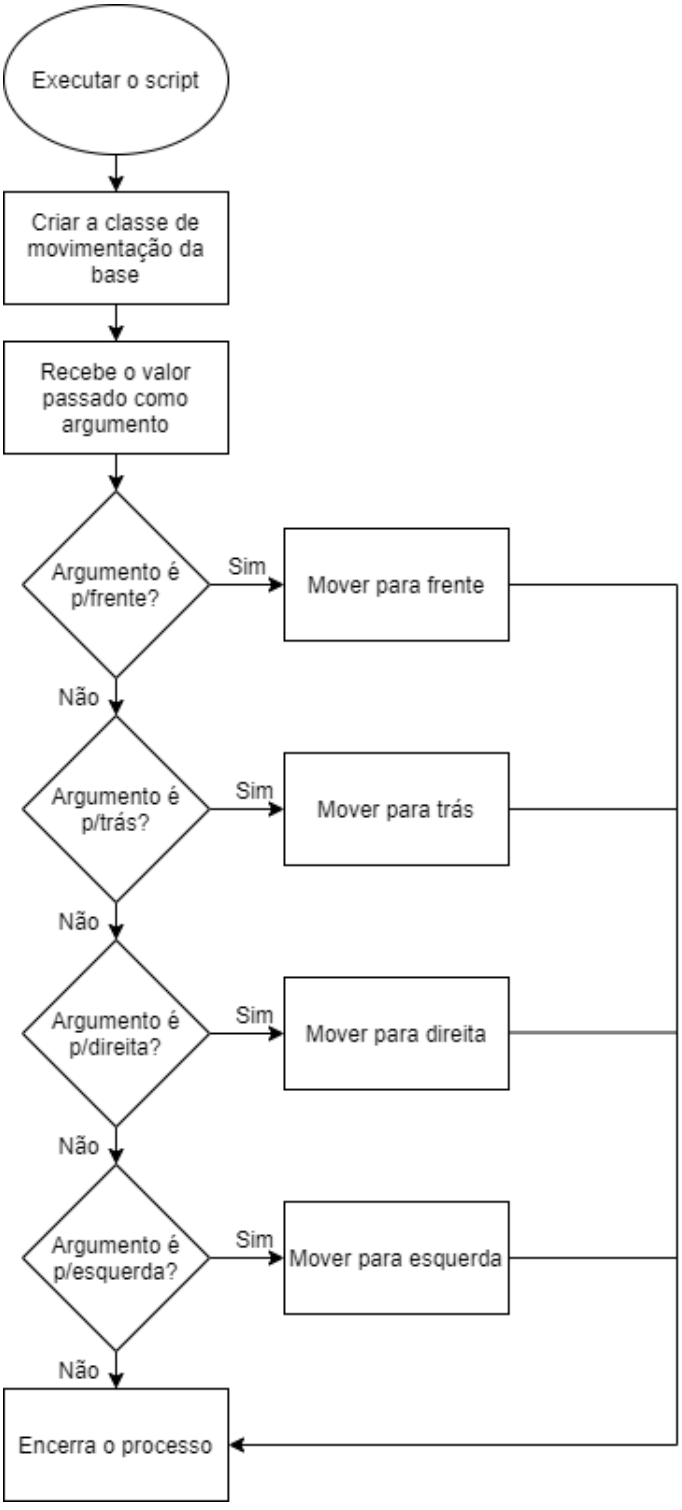

Figura 34: Três etapas do processo de movimentação da base robótica.

Fonte: Os Autores.

O fluxograma que detalha o processo da API de movimentação, movimentar_base.js, anteriormente detalhado está exposto a seguir na Figura 35:

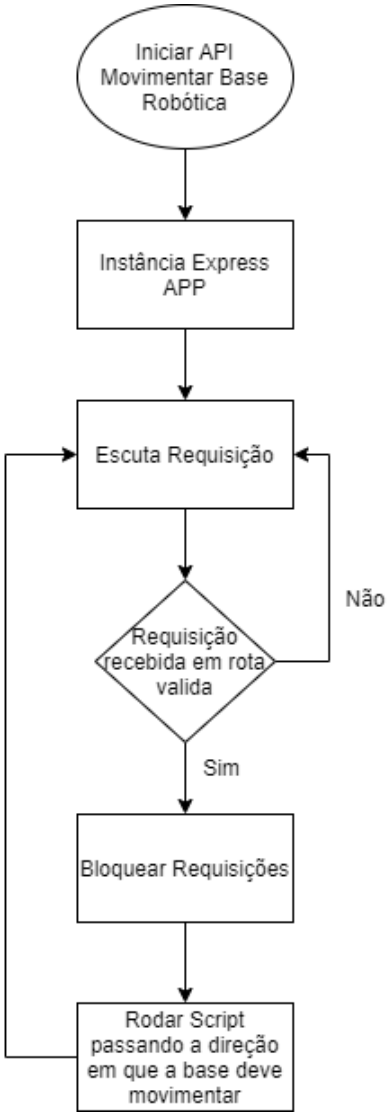

Figura 35: Fluxograma da movimentação do protótipo.

Fonte: Os Autores.

3.2.3.8 Instalação do ROS

A versão do sistema operacional utilizada foi a *Ubuntu Mate 18.04.2*, pois esta é uma versão direcionada ao *hardware* do *Raspberry.*

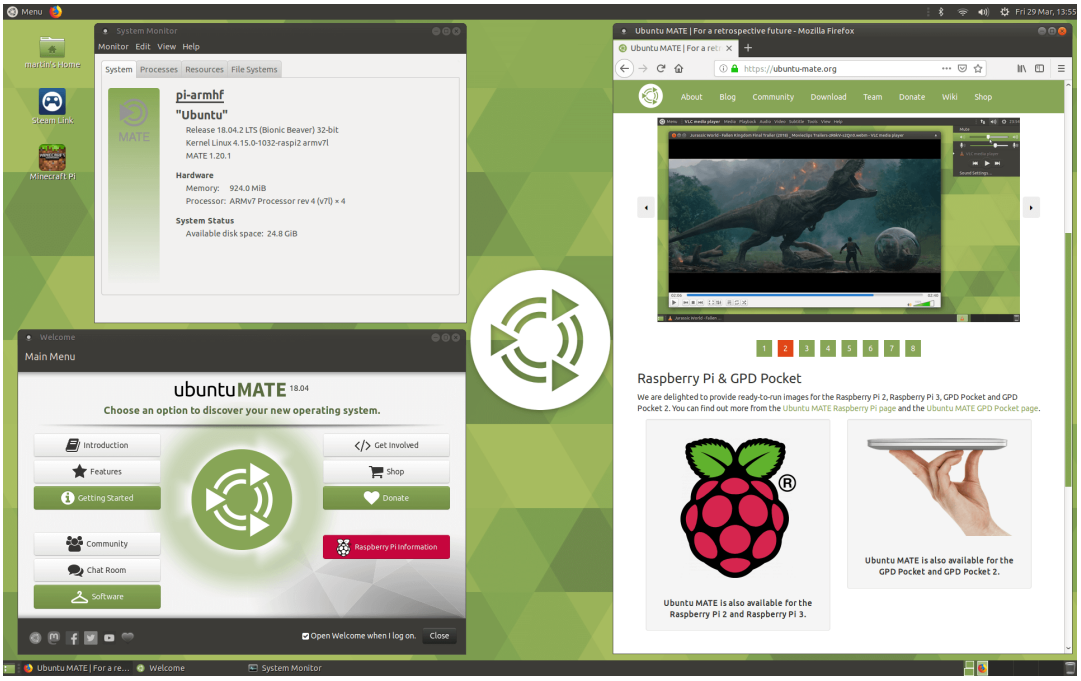

Figura 36 Tela inicial sistema operacional ubuntu Mate 18.04

Fonte: UBUNTU.

Após a inicialização do sistema operacional é necessário realizar todas as atualizações, com o seguinte comando no terminal:

sudo apt-get update && sudo apt-get upgrade

Esse comando irá baixar e instalar todas as atualizações do sistema operacional. Para instalar o ROS foi escolhida a versão *Melodic,* para sua instalação foram seguidos os passos descritos na documentação oficial do ROS.

Foi criado também uma *work space* (WS) seguindo a documentação oficial do ROS.

RPLidar A1M8: Após a criação da *WS* deve-se clonar o repositório do *RPLidar* no *github* da *Slamtec* (fabricante do *RPLidar*), que contém os arquivos necessários para integração do sistema com o *hardware,* para a pasta *src* que foi criada dentro de sua *WS*.

Hector Slam: Clonar do *github* o repositório do *HectorSlam* para a pasta *src* da sua *WS,* arquivos responsáveis pelo mapeamento e localização.

Para utilização do *RPLidar* deve-se liberar a porta USB onde ele está ligado para leitura e escrita, através do seguinte comando no terminal do S.O. com permissão de administrador do sistema:

sudo chmod 666 /dev/ttyUSB0

Para a criação da estrutura do projeto é necessário realizar o seguinte comando dento da raiz do seu diretório *WS:*

catkin_make

No diretório do *RPLidar* que está dentro da pasta *src* é necessária a criação de um arquivo com as configurações de leitura do *RPLidar A1M8* no formato *.launch* e também a criação de um arquivo neste formato dento do diretório *hector slam* para utilização dos dados obtidos para criar o mapa.

4 RESULTADOS E DISCUSSÕES

Para iniciar o mapeamento são abertas três seções no terminal, uma inicia o *ROS* e as outras duas executam os arquivos criados.

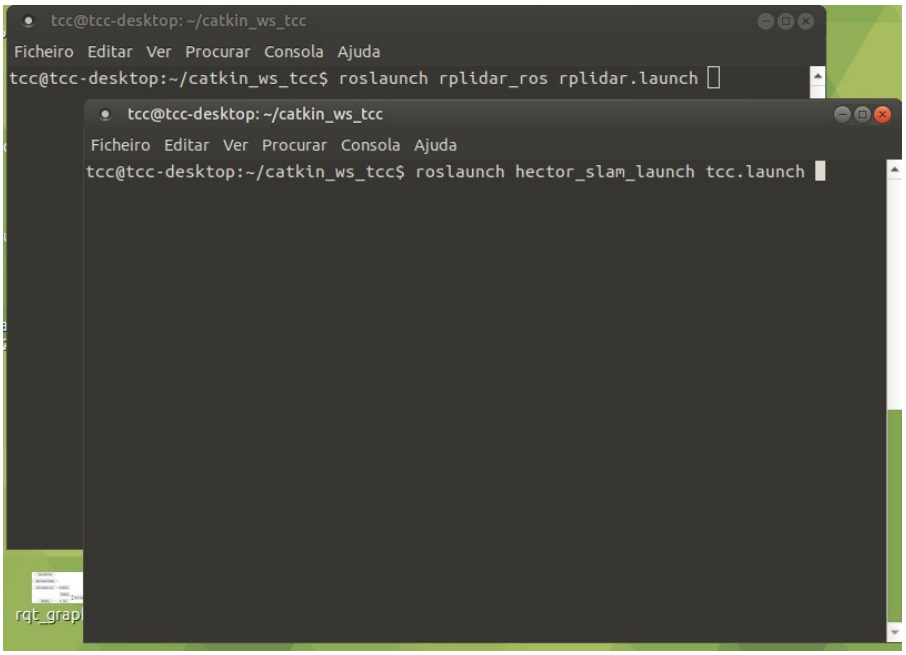

Figura 37: Execução dos Scripts.

Fonte: Os Autores

Após a execução será aberto o *software rviz* nativo da instalação do *ROS* para a visualização em tempo real do mapeamento.

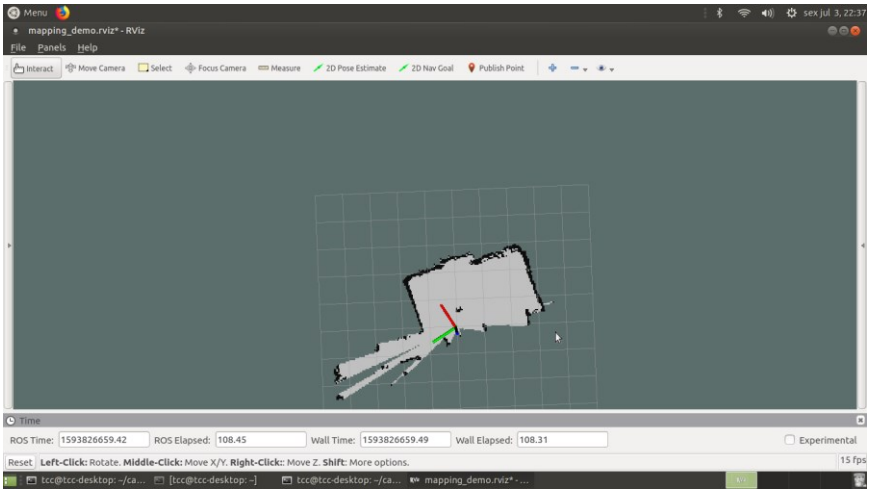

Figura 38: Mapeamento sendo realizado.

Fonte: Os Autores

Durante a execução é possível verificar os nós e tópicos utilizados pelo *ROS*, criando a estrutura a seguir:

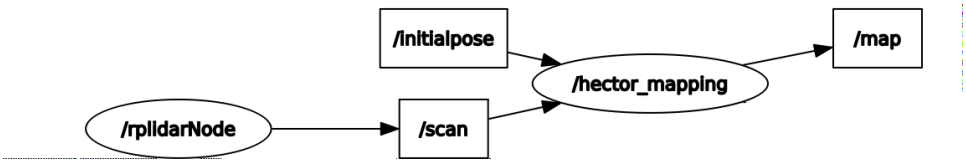

Figura 39: Nós e tópicos ativos durante a execução.

Fonte: Os Autores.

1. nó */rplidarNode* é um programa que realiza o interfaceamento entre o sensor e o microcontrolador.
2. tópico */scan* recebe os dados de leitura obtidos pelo *RPLidar.*
3. tópico /*initialpose* recebe a posição inicial do sensor.
4. nó /*hector_mapping* recebe os dados de posição inicial e a leitura do sensor.
5. tópico /*map* utiliza os dados de posição inicial, onde o sensor está, e também a montagem dos dados obtidos pelo sensor.

A Figura 40 apresenta o estado inicial da interface ao começar o mapeamento. O teste demonstrado foi realizado com a base robótica inicialmente em um corredor de dois metros de comprimento e noventa centímetros de largura e finalizando na sala de cinco metros de comprimento e três de largura, o mapeamento final está demonstrado na Figura 41.

Ao iniciar a interface, é possível observar que contém um breve mapeamento que é constituído dos dados capturados pelo sensor no momento em que é ligado.

Figura 40: Primeiro mapeamento realizado.

Fonte: Os Autores.

O componente de mapeamento busca a imagem e apresenta na interface, sendo possível visualizar a evolução do mapa gerado. A taxa de atualização do mapa é de um *frame* a cada

cinco segundos, o que deixa claro visualizando a evolução do mapa, os novos perímetros mapeados.

Em questão de velocidade, é necessário considerar o peso da base robótica e o peso dos equipamentos que carrega, com isso temos:

Tabela 13: Peso dos componentes.

ITEM	**PESO - KG**
Base Robótica	1,550
Bateria 12V	1,700
LIDAR	0,190
Raspberry	0,060
Total	3,500

Fonte: Os Autores.

Com a base suportando três quilos e quinhentas gramas, conseguiu realizar o percurso de dois metros em vinte segundos, podendo então chegar ao valor de dez centímetros por segundo, sendo essa velocidade constante.

Ao realizar comando de movimento, é registrado no log, assim criando um histórico de quais foram os movimentos realizados para chegar onde está. Foi considerado realizar o log dessa maneira para ser possível voltar pelo mesmo caminho, sendo necessário somente refazer os comandos de forma inversa.

O componente da câmera apresenta a imagem capturada pela câmera conectada a *Raspberry.* A taxa de atualização da imagem está fixada em um *frame* por segundo, podendo assim identificar objetos e obstáculos pelo caminho em frente a base.

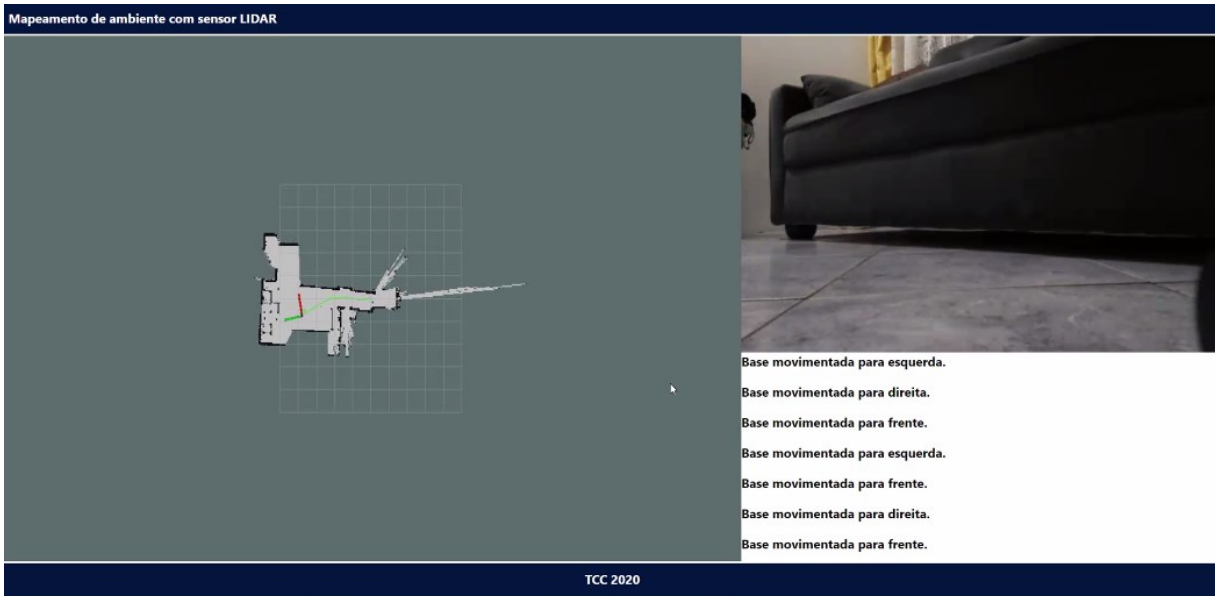

Figura 41: Mapeamento realizado.

Fonte: Os Autores.

Para a realização do mapa apresentado na Figura 41, foi necessário o tempo de dois minutos e 14 segundos.

Porém o mapeamento de ambientes maiores demorará cada vez mais, proporcionalmente ao seu tamanho.

Visando a autonomia da base robótica, é necessário evidenciar que a bateria utilizada, UP1250, tem a capacidade de 5A/h, os consumos realizados pelos componentes da base:

Tabela 14: Consumo do sistema.

ITEM	**mA/h**
Raspberry	330
LIDAR	400
Motores	720
Total	1450

Fonte: Os Autores.

Realizando a divisão da capacidade da bateria, 5A/h, pelo consumo total dos componentes utilizados, 1.45A/h. Temos o total de três horas e vinte e sete minutos de autonomia.

O protótipo pronto, com todos componentes e suas funcionalidades integrados está exposto a seguir nas Figuras 42 e 43.

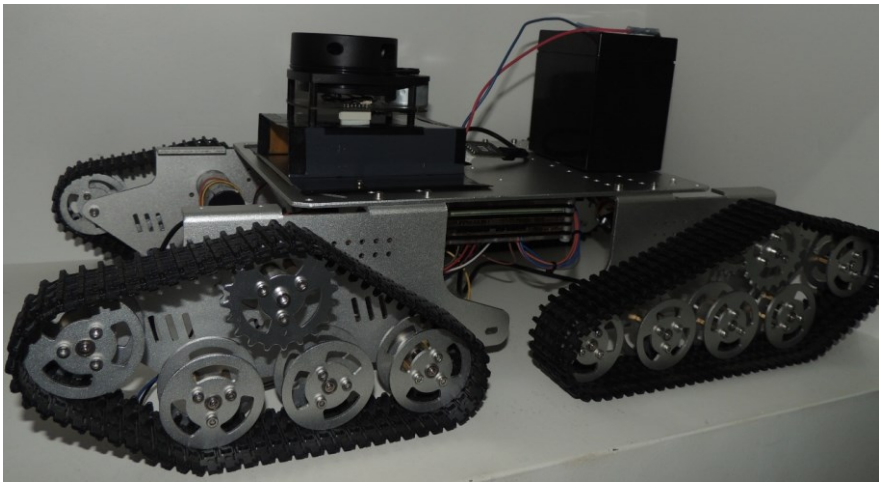

Figura 42: Protótipo Finalizado.

Fonte: Os Autores.

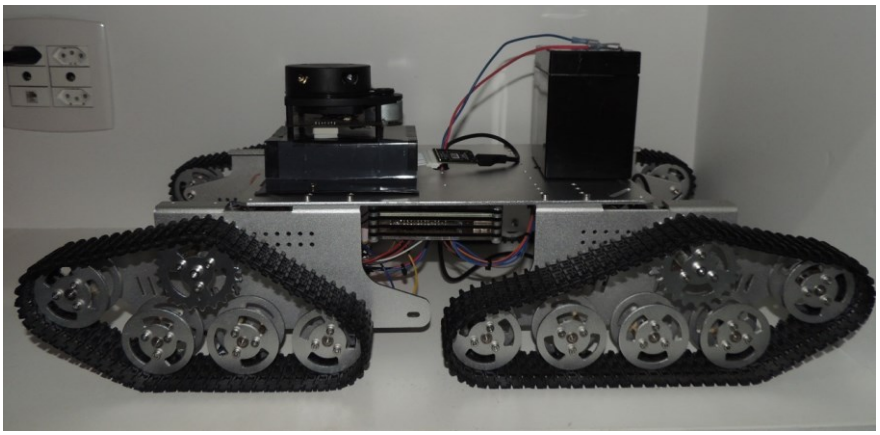

Figura 43: Protótipo Finalizado.

Fonte: Os Autores.

5 CONCLUSÃO

Neste trabalho foi apresentado o desenvolvimento de um sistema de mapeamento de ambientes fechados, ele é constituído de uma base robótica em que nela está acoplado um LIDAR para realizar o mapeamento, este robô é controlado pelo usuário através de uma interface *WEB*.

Os resultados obtidos foram satisfatórios, uma vez que foi possível realizar a movimentação da base por intermédio da interface *WEB*, o registro do mapeamento utilizando o *rviz* e a exibição na tela da imagem capturada pela câmera.

O projeto nos ajudou em nosso desenvolvimento profissional, acadêmico e pessoal através da aplicação prática dos conteúdos ministrados pelos professores e pelo conhecimento adquirido através dos estudos realizados em paralelo durante toda a graduação.

5.2 PRINCIPAIS DIFICULDADES

Pode-se evidenciar algumas dificuldades ou empecilhos que mais tomaram tempo no momento da procura de solução, como:

- Movimentar a base ao reconhecer o comando pelo navegador:
    - O fato de o comando vir de outro IP da rede dificulta a execução do movimento que deve ser executada localmente. Sendo assim necessário ter uma função que ative a funcionalidade pelo server;
    - Quando o servidor recebe o comando com o movimento da base, é necessário executar o movimento em um curto período de tempo, pois não é possível deixar um canal de transferência em tempo real.
- Adicionar imagem da câmera em tempo real na interface:
    - Como a transferência de imagem em tempo real requer velocidade de transmissão e gravação da imagem no diretório, o foco foi mostrar a imagem mesmo que em *time lapse* para demonstração do que está sendo capturado pela câmera.
- Demonstrar a imagem gerada pelo LIDAR na interface web.
    - A imagem gerada pelo *rviz* não é em formato aceitável pelo HTML, então foi preciso utilizar de softwares terceiros para a aplicação da extensão da imagem.

As maiores dificuldades enfrentadas estão relacionadas a parte de *software*, mais precisamente na integração entre o mapeamento e exibição em tela.

5.3 MELHORIAS FUTURAS

Um próximo passo seria o uso um *hardware* que possua suas especificações técnicas mais robustas do que o *Raspberry* utilizado, levando em consideração algumas lentidões e o pequeno *delay* da comunicação. Com o upgrade no *hardware* será possível executar e exibir em tela o mapeamento com maior velocidade de resposta.

REFERÊNCIAS

OSÓRIO, F; PRESTES, E; ROMERO, R. A. F; WOLF D. **ROBÓTICA MÓVEL** – Rio de Janeiro: LTC – Livros Técnicos e Científicos Editora, 2014.

AYRES, M. **Conheça a história dos robôs**. Disponível em: <https://tecnologia.uol.com.br/ultnot/2007/10/01/ult4213u150.jhtm> Acesso em: 05 de outubro de 2019.

METROPOLITAN MUSEUM OF ART. **Mechanical Dog**.Disponível em: <https://www.metmuseum.org/art/collection/search/544519> Acesso em: 05 deoutubro de 2019.

MAIA, D. V. A. **Automação Industrial e Robótica**. Disponível em: <http://professor.pucgoias.edu.br/SiteDocente/admin/arquivosUpload/17829/material/ARTIGO08.pdf> Acesso em: 01 de novembro de 2019.

DARPA. **The Grand Challenge**. Disponível em: <https://www.darpa.mil/about-us/timeline/-grand-challenge-for-autonomous-vehicles> Acesso em: 05 de outubro de 2019.

INTERNATIONAL FEDERATION OF ROBOTICS. **Executive Summary World Robotics 2019 Industrial Robots**. Disponível em: <https://ifr.org/downloads/press2018/Executive%20Summary%20WR%202019%20Industrial%20Robots.pdf> Acesso em: 05 de outubro de 2019.

INTERNATIONAL FEDERATION OF ROBOTICS; MÜLLER, C. **Domestic Service Robots**. Disponível em: <https://ifr.org/downloads/press2018/IFR%20World%20Robotics%20Presentation%20-%2018%20Sept%202019.pdf> Acesso em: 05 de outubro de 2019.

FERREIRA, G. T. **SISTEMA DE MAPEAMENTO TRIDIMENSIONAL DE AMBIENTES** – Trabalho de Conclusão de Curso – Escola de Engenharia de São Carlos - Universidade de São Paulo, São Paulo, 2014.

MOSCA, G; TIPLER, P. A.**FÍSICA PARA CIENTISTAS E ENGENHEIROS** – Volume 2 Eletricidade e Magnetismo, Óptica - Sexta Edição - Rio de Janeiro: LTC – Livros Técnicos e Científicos Editora, 2009.

PETRUZELLA, F. **Motores Elétricos e Acionamento** – Porto Alegre - RS: AMGH Editora Ltda., 2013.

OLIVEIRA, R. S. **Projeto da Eletrônica Embarcada para um Robô Móvel Aplicado a Atividades de Terapia Ocupacional**. Universidade Federal do Rio de Janeiro – Departamento de Eletrônica e de Computação, Rio de Janeiro – RJ, 2014. Disponível em: <http://monografias.poli.ufrj.br/monografias/monopoli10009552.pdf> Acesso em: 19 de novembro de 2019.

CARVALHO, A. F; MILLÉO, L. M. **DESENVOLVIMENTO DE UM SENSOR DE CORRENTE ELÉTRICA A PARTIR DE UM SENSOR DE EFEITO HALL**. Trabalho de Conclusão de Curso, Universidade Tecnológica Federal do Paraná – Departamento de Eletrônica, Ponta Grossa – PR, 2010. Disponível em: <http://repositorio.roca.utfpr.edu.br/jspui/bitstream/1/8512/1/PG_COAUT_2017_2_06.pdf> Acesso em: 19 de novembro de 2019.

FARINHAKI, R; JÚNIOR, A. M; NEIVA, E. C. R; NETO, A. L. R. **SISTEMA DE MEDIÇÃO DE CAMPO MAGNÉTICO BASEADO NO EFEITO HALL E ARDUINO**. Universidade Tecnológica Federal do Paraná – Departamento Acadêmico de Eletrônica, Curitiba – PR, 2010. Disponível em: <http://paginapessoal.utfpr.edu.br/msergio/portuguese/ensino-de-fisica/oficina-de-integracao-ii/oficina-de-integracao-ii/Monog-10-1-Efeito-Hall.pdf> Acesso em: 19 de novembro de 2019.

LIMA, G. F; SALAZAR, A. O; SILVA, A. E. M. **DESENVOLVIMENTO DE TACÔMETRO MAGNÉTICO PARA MEDIÇÃO DE VELOCIDADE EM PIGs INSTRUMENTADOS**. Universidade Federal do Rio Grande do Norte – Laboratório de Avaliação de Medição em Petróleo, 2017. Disponível em: <http://www2.ifrn.edu.br/ojs/index.php/HOLOS/article/download/5202/pdf> Acesso em: 19 de novembro de 2019.

ROS. **ROS**. Disponível em: <https://www.ros.org/about-ros/> Acesso em: 30 de março de 2020.

REIS, F. **O que é PWM – Pulse Width Modulation**. Disponível em: <http://www.bosontreinamentos.com.br/eletronica/curso-de-eletronica/curso-de-eletronica-o-que-e-pwm-pulse-width-modulation/> Acesso em: 30 de março de 2020.

BRAGA, N. C. **Controle PWM de Motor DC (MEC139)**. Disponível em: <http://www.newtoncbraga.com.br/index.php/robotica/5534-mec139> Acesso em: 30 de março de 2020.

WIKI HECTOR SLAM. **hector_slam**. Disponível em: <http://wiki.ros.org/hector_slam> Acesso em: 30 de março de 2020.

ROSSI, T. X. **MAPEAMENTO TRIDIMENSIONAL DE AMBIENTES INTERNOS UTILIZANDO UM SENSOR LIDAR**. Universidade Federal de Ouro Preto – Escola de Minas – Colegiado do curso de Engenharia de Controle e Automação, 2019.

RASPBERRYPI. ***Raspberry*** **Pi 3** ***Model*** **B+**. Disponível em: <https://www.raspberrypi.org/products/raspberry-pi-3-model-b-plus/> Acesso em: 23 de maio de 2020.

MICROSOFT. **Mapeamentos de PIN do Raspberry Pi 2 & 3**. Disponível em: <https://docs.microsoft.com/pt-br/windows/iot-core/learn-about hardware/pinmappings/pinmappingsrpi> Acesso em: 23 de maio de 2020.

FRUTOSO, E. B; JUNIOR, F. V; PEREIRA, G. R. **ARDUINO E *RASPBERRY* PI: UMA COMPARAÇÃO DE ESPECIFICAÇÕES E APLICAÇÕES DE MINICOMPUTADORES**. Disponível em: <http://eventos.ifc.edu.br/micti/wp-content/uploads/sites/5/2014/08/ARDUINO%C2%AE-E-RASPBERRY-PI%C2%AE-UMA-COMPARA%C3%87%C3%83O-DE-ESPECIFICA%C3%87%C3%95ES-E-APLICA%C3%87%C3%95ES-DE-MINICOMPUTADORES.pdf> Acesso em 23 de maio de 2020.

STMICROELECTRONICS. ***Datasheet*** **L298N**. Disponível em: <https://www.sparkfun.com/datasheets/Robotics/L298_H_Bridge.pdf> Acesso em: 20 de maio de 2020.

UBUNTU. **Ubuntu MATE for *Raspberry Pi***. Disponível em: < https://ubuntu-mate.org/ports/raspberry-pi/> Acesso em: 03 de julho.