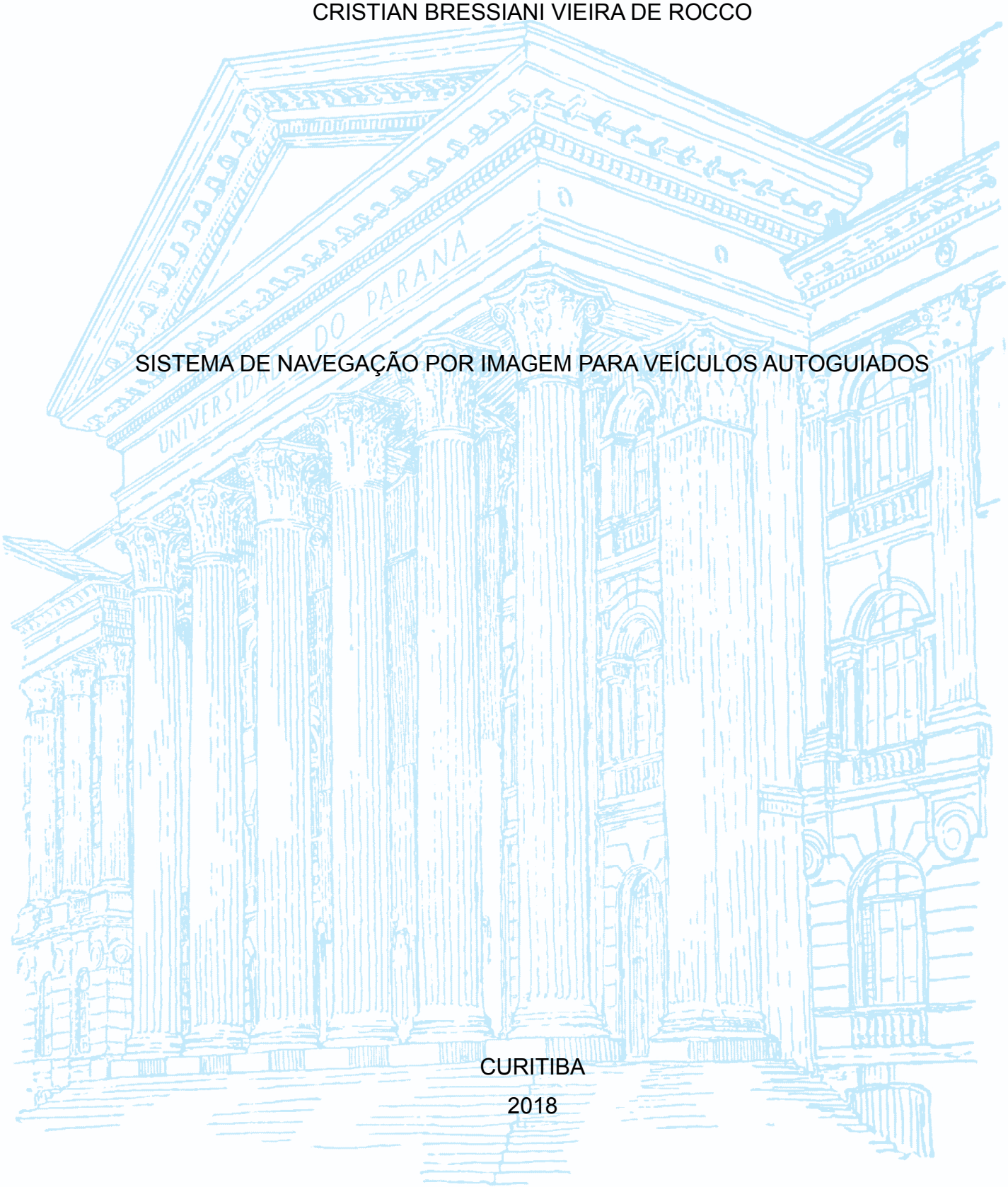
UNIVERSIDADE FEDERAL DO PARANÁ

CRISTIAN BRESSIANI VIEIRA DE ROCCO



SISTEMA DE NAVEGAÇÃO POR IMAGEM PARA VEÍCULOS AUTOGUIADOS

CURITIBA

2018

CRISTIAN BRESSIANI VIEIRA DE ROCCO

SISTEMA DE NAVEGAÇÃO POR IMAGEM PARA VEÍCULOS AUTOGUIADOS

Trabalho De Conclusão De Curso apresentado ao Curso de Graduação em Engenharia Elétrica, Setor de Tecnologia, da Universidade Federal do Paraná, como requisito parcial à obtenção do título de Engenheiro Eletricista ênfase em eletrônica.

Orientador: Prof. Dr. EDUARDO PARENTE RIBEIRO.

CURITIBA

2018

Dedico esse trabalho ao povo Brasileiro, o qual, guiado por Deus, possam sempre fazer do Brasil uma grande nação.

AGRADECIMENTOS

Existem muitas pessoas que gostaria de agradecer pela minha formação pessoal, espiritual e profissional, por terem contribuído para a elaboração desse presente trabalho com motivação, persistência e atenção.

Agradeço inicialmente a Deus por tudo que me ofereceu, oportunidades e desafios para o meu crescimento pessoal.

A minha Família por ter sempre me motivado e ajudado nos momentos desafiadores.

A todos os meus amigos, dentre os quais destaco Jonathan Rocha, Lucas Hashimoto, Vinicius Scuzziatto e Luisa Dias que participaram de forma mais próxima ao TCC com críticas e sugestões.

Aos colegas de trabalho da empresa AGVS por permitir uma oportunidade de trabalhar com Veículos Autoguiados e aperfeiçoar meu conhecimento na área de robótica.

Aos meus professores pela minha formação. Dentre eles destaco, Alexandre Aoki e meu orientador Eduardo Parente.

RESUMO

O presente trabalho tem como objetivo desenvolver e implementar em um protótipo um sistema de navegação baseado em processamento de imagem. A navegação é conduzida por uma faixa abaixo do protótipo e este possui a capacidade de identificar cones de trânsito.

O protótipo elaborado atingiu uma navegação autônoma de 32 centímetros por segundo de forma estável. O código elaborado foi capaz de identificar o centro da faixa de navegação em 13 quadros por segundo e identificar o cone em 9 quadros por segundo, sendo a taxa de acerto de 72 %.

Ao longo do projeto realizou-se um levantamento de trabalhos semelhantes assim como algumas das características mais importantes para veículos autoguiados. Utilizou-se a biblioteca OpenCV de forma extensiva e o Hardware de processamento adotado foi o Raspberry Pi 3B+. A comunicação se deu por meio do sistema VNC assim como do protocolo SSH. O acionamento dos atuadores foram por meio de PWM. A câmera utilizada ao longo do projeto foi o módulo da Raspberry Rev1.3.

Palavras-chave: OpenCV. Processamento de imagem. Raspberry Pi. Cone de trânsito. Veículo autoguiado.

ABSTRACT

The current final year project had the objective to develop and implement a prototype guided by image processing. The navigation path was indicated by lanes under de robot, which also was capable of detecting traffic cones.

The prototype elaborated reach a speed of 32 centimeters per second with a stable autonomous navigation. The code elaborated was capable of identifying the center of the lane under 13 frames per second and the cones under 9 frames per second with a successful rate of 72 %.

During the project it was realized an evaluation of similar projects and some important characteristics of autonomous vehicles. For the project it was used extensively the library OpenCV and the hardware Raspberry Pi 3B+. The communication system was by VNC and also based on the protocol SSH. The actuation of the motors was by PWM. The camera used was de Raspberry module Rev 1,3.

Keywords: OpenCV. Image processing. Raspberry Pi. Transit Cones. Autonomous Vehicle.

LISTA DE FIGURAS

FIGURA 1 - TALAS DE EMENTA PARA ELETROCALHAS.....34
FIGURA 2 - PROTÓTIPO UTILIZADO.....34
FIGURA 3 - RODÍZIO PARA SUPORTE TRASEIRO.....35
FIGURA 4 - REFLEXO OBSERVADO NO PISO.....39
FIGURA 5 - PISTA UTILIZADA PELO PROTÓTIPO.....40
FIGURA 6 - FLUXOGRAMA GERAL.....41
FIGURA 7 - FILTRO AZUL POR ESPAÇO DE COR.....44
FIGURA 8 - FILTRO LARANJA POR ESPAÇO DE COR NO RASPBERRY.....44
FIGURA 9 - MUDANÇA DO ESPAÇO DE COR PARA HSV.....44
FIGURA 10 - MASCARA PELO FILTRO DE COR E SOBREPOSIÇÃO.....44
FIGURA 11 - FILTRO CANNY - CONCRETO.....45
FIGURA 12 - DILATAÇÃO MORFOLÓGICA - CONCRETO.....46
FIGURA 13 - EROSÃO MORFOLÓGICA - CONCRETO.....46
FIGURA 14 - EROSÃO MORFOLÓGICA - CONE.....47
FIGURA 15 - DILATAÇÃO MORFOLÓGICA - CONE.....47
FIGURA 16 - DETECÇÃO DOS POLÍGONOS MAIORES - CONCRETO.....48
FIGURA 17 - DETECÇÃO DO POLÍGONO MAIOR - AZULEJO.....48
FIGURA 18 - DETECÇÃO DO POLÍGONO MAIOR - OBSTÁCULO.....49
FIGURA 19 - DETECÇÃO REALIZADA POR AMBOS OS MÉTODOS.....50
FIGURA 20 - DETECÇÃO SATISFATÓRIA DO CONE – CELULAR.....59
FIGURA 21 - DETECÇÃO SATISFATÓRIA DO CONE – CELULAR.....59
FIGURA 22 - DETECÇÃO INSATISFATÓRIA DO CONE - CELULAR.....60
FIGURA 23 - DETECÇÃO INSATISFATÓRIA DO CONE - CELULAR.....60
FIGURA 24 - DETECÇÃO SATISFATÓRIA DO CONE – MODULO RASPBERRY…60
FIGURA 25 - DETECÇÃO DO CONE – MODULO RASPBERRY.....61

LISTA DE GRÁFICOS

GRÁFICO 1 - TEMPORIZAÇÃO DO CÓDIGO PADRÃO COM EXIBIÇÃO DE IMAGEM, COMUNICAÇÃO VNC E USO DA IDE.....54
GRÁFICO 2 - TEMPORIZAÇÃO DO CÓDIGO PADRÃO SEM EXIBIÇÃO DE IMAGEM, COMUNICAÇÃO VNC E USO DA IDE.....54
GRÁFICO 3 - TEMPORIZAÇÃO DO CÓDIGO PADRÃO COM EXIBIÇÃO DE IMAGEM, COMUNICAÇÃO VNC E USO DO TERMINAL.....55
GRÁFICO 4 - TEMPORIZAÇÃO DO CÓDIGO PADRÃO SEM EXIBIÇÃO DE IMAGEM E COMUNICAÇÃO SSH.....55
GRÁFICO 5 - GRÁFICO COMPARATIVO DO CÓDIGO PADRÃO NA IDE.....56
GRÁFICO 6 - GRÁFICO COMPARATIVO SEM EXIBIÇÃO DE IMAGEM NA IDE….56
GRÁFICO 7 - TEMPORIZAÇÃO UTILIZANDO O PROTOCOLO SSH.....57
GRÁFICO 8 - ERRO COLETADO AO LONGO DE 25 METROS.....58

LISTA DE QUADROS

QUADRO 1 – COMPARAÇÃO ENTRE TIPOS DE SENSORES.....22

LISTA DE TABELAS

TABELA 1 - COMPARAÇÃO ENTRE MOTORES ELÉTRICOS.....32
TABELA 2 - RELAÇÃO ENTRE PWM E FREQUÊNCIA DE ROTAÇÃO.....36
TABELA 3 - RELAÇÃO ENTRE PWM E CORRENTE DEMANDADA.....37

LISTA DE ABREVIATURAS OU SIGLAS

CRIM - Motor de Indução com Rotor em Gaiola (*Cage Rotor Induction Motor*)

DC - Corrente Contínua (*Direct Current*)

FPGA - Arranjo de Portas Programáveis em Campo *(Field Programmable Gate Array)*

FPS - Quadros por segundo (*Frames Per Second)*

GNSS - Sistema de Navegação Global por Satélite (*Global Navigation Satellite System)*

GPIO - General Purpose Input/Output

HSV - Matriz, Saturação e Valor (*Hue, Saturation and Value*)

IDE - Ambiente de Desenvolvimento Integrado (*Integrated Development Environment)*

LIDAR - Detecção e Telemetria por Laser (*Light Detection And Ranging*)

Li-Ion - Íon de Lítio

Li-Po - Polímero de Lítio

PID - Proporcional, Integral e Derivativo

PMSM - Motor Síncrono de Imã Permanente (*Permanent Magnet Synchronous Motor)*

PWM - Modulação de Largura de Pulso *(Pulse Width Modulation)*

RADAR - Detecção e Telemetria por Rádio (*Radio Detection And Ranging*)

RFB - *Remote Frame Buffer*

SSH - *Secure Shell*

TCC - Trabalho de Conclusão de Curso

TTL - Lógica Transistor-Transistor (*Transistor-Transistor Logic*)

UFPR - Universidade Federal do Paraná

VNC - *Virtual Network Computing*

2D - Espaço Bidimensional

3D - Espaço Tridimensional

LISTA DE SÍMBOLOS

A	- Ampère
cm	- Centímetros
cm/s	- Centímetros por segundo
J	- Joule
kg	- Quilograma
m	- metro
N	- Newton
s	- Segundo
us	- Microssegundo
V	- Volt
W	- Watt

SUMÁRIO

1 INTRODUÇÃO.....15
1.1 OBJETIVOS.....16
1.2 METODOLOGIA.....17
2 REVISÃO DE LITERATURA.....19
2.1 ROBÓTICA.....19
2.2 VEÍCULOS AUTÔNOMOS.....19
2.3 TIPOS DE SENSORES.....20
2.4 SENSORES PARA SEGUIR LINHAS.....22
2.5 PROCESSAMENTO DE IMAGEM.....23
2.6 OPENCV.....24
2.7 LINGUAGEM PYTHON.....24
2.8 MICROPROCESSADOR.....25
2.9 MICROCONTROLADOR.....26
2.10 DRIVER.....26
2.11 COMUNICAÇÃO.....27
2.12 BATERIAS.....29
2.13 MOTORES E LOCOMOÇÃO.....31
3 MATERIAIS E MÉTODOS.....33
3.1 CARENAGEM.....33
3.2 NÚMERO DE RODAS.....35
3.3 POSICIONAMENTO DAS RODAS.....35
3.4 RODÍZIO E RODAS.....35
3.5 MOTOR.....36
3.6 BATERIAS.....37
3.7 CONVERSOR DE TENSÃO.....38
3.8 POSICIONAMENTO DA CÂMERA.....38
3.9 MONTAGEM DA PISTA.....39
3.10 DESENVOLVIMENTO DO ALGORITMO.....40
3.11 DESAFIOS ENCONTRADOS.....51
3.12 MATERIAL UTILIZADO.....51

4 RESULTADOS.....52
4.1 TEMPORIZAÇÃO.....52
4.2 MOVIMENTAÇÃO DO PROTÓTIPO.....57
4.3 IDENTIFICAÇÃO DO OBSTÁCULO.....58
5 CONCLUSÃO.....63
5.1 RECOMENDAÇÕES PARA TRABALHOS FUTUROS.....63
REFERÊNCIAS.....65
APÊNDICE 1 – FLUXOGRAMA INICIALIZAÇÃO.....68
APÊNDICE 2 – FLUXOGRAMA OBTER IMAGEM.....69
APÊNDICE 3 – FLUXOGRAMA DETECTAR CONE.....70
APÊNDICE 4 – FLUXOGRAMA DETECTAR REFERÊNCIA DA FAIXA.....71
APÊNDICE 5 – FLUXOGRAMA NAVEGAÇÃO.....72
APÊNDICE 6 – FLUXOGRAMA FINALIZAÇÃO.....73

1 INTRODUÇÃO

A detecção de objetos e a identificação de padrões por meio de processamento de imagens está cada vez mais ágil, robusta e confiável, de modo que é um dos setores da engenharia eletrônica que mais tem impactado o ser humano nas últimas décadas. O número de aplicações nessa área vem aumentando, adentrando em novos setores como esportes, medicina e na automação industrial.

Com o propósito de se fazer uso desta tecnologia e de se aproveitar da constante tendência no desenvolvimento de eletrônicos, em especial microprocessadores mais rápidos, menores e com baixo consumo energético, observou-se uma oportunidade de se desenvolver um sistema de navegação por meio de processamento de imagem.

Atualmente existem diversos sistemas de navegação comercialmente disponíveis, como Sensores por Guias Magnéticas (ROBOTEQ, 2018), por Scanners a Laser (SICK, 2018), por Fototransistores como o TCRT5000 (GÖTTING, 2018) e por câmeras. O sistema de navegação por câmeras possui variações quanto a direção da câmera – posicionamento à frente, direcionada para o lado, para cima ou para baixo, entre outras posições, sendo que a simples variação da direção da câmera pode levar a criação de novos paradigmas e dificuldades na implementação do sistema de navegação.

O uso de câmeras para navegação de veículos quer sejam eles industriais ou urbanos é largamente utilizada (THRUN et al., 2007), e dessa forma observa-se uma grande possibilidade de crescimento científico e pessoal nessa área. Um dos principais fatores para a elaboração de um trabalho de final de curso que envolvam processamento de imagem e mecatrônica são: a grande demanda industrial por pessoas que trabalhem com mecatrônica e o grande desenvolvimento atual, quer seja em medicina (BANKMAN, 2008), veículos autônomos (COMMA.AI, 2018) e drones (CHUNG, 2018) que demandem o uso de processamento de imagem.

Com a elaboração do trabalho foi possível desenvolver o conhecimento referente ao funcionamento de microprocessadores, drivers, sistema operacional,

bibliotecas do Python e principalmente processamento de imagens. Todas essas atividades são constantemente demandadas em empresas para o desenvolvimento de software e veículos autônomos (OLIVEIRA, 2011).

1.1 OBJETIVOS

Este trabalho de Conclusão de Curso tem como objetivo o desenvolvimento e implementação de um sistema de navegação por meio do processamento de imagem. A navegação fará uso de faixas ou linhas como referências e será implementada em um protótipo em miniatura de um veículo autoguiado o qual será desenvolvido ao longo do período. Também será implementado um algoritmo capaz de identificar um objeto encontrado em pátios industriais.

1.1.1 Objetivo geral

Espera-se que ao término do projeto obtenha-se um protótipo em miniatura de um veículo guiado por imagens, este protótipo terá uma velocidade média ou superior a 15 cm por segundo, o posicionamento da câmera será direcionado para frente e, espera-se que possua uma taxa de processamento de no mínimo 2 quadros por segundo (FPS) quando utilizado um algoritmo para a detecção de algum tipo de objeto e 12 quadros por segundo sem a detecção de objetos.

1.1.2 Objetivos específicos

Este trabalho será desenvolvido em um período de 4 meses. Estando divido em etapas, sendo estas:

- Desenvolvimento de um protótipo de veículo autoguiado.
    - Desenvolvimento do protótipo em miniatura de um veículo.
    - Escolha do microprocessador, microcontrolador e câmera.
    - Escolha e justificativa da linguagem de programação a ser utilizada.

- Elaboração e aperfeiçoamento do código a ser implementado para o sistema de navegação.
- Implementação do código de navegação.
- Ajuste dos parâmetros de controle.
- Análise da navegação do protótipo de veículo autoguiado.
- Implementação e possível desenvolvimento de um algoritmo capaz de identificar um objeto encontrado em pátios industriais.

- Elaboração do Relatório de Trabalho de Conclusão de Curso (TCC).

1.2 METODOLOGIA

O andamento do projeto e a metodologia adotada se dará por meio das 5 etapas a seguir. Levantamento bibliográfico, desenvolvimento do algoritmo, elaboração do protótipo, implementação do código no microprocessador e no computador e finalmente a análise e síntese dos resultados.

O levantamento bibliográfico teve como propósito não apenas identificar as principais referências teóricas, mas também as principais referências práticas, isto é, de projetos semelhantes. Ao todo se observou 2 trabalhos de TCCs (ANDRADE, 2016; RIBEIRO, 2017), que apresentaram propostas semelhantes ao seguimento de linhas por processamento de imagem, ainda que apresentem diferenças substanciais.

Observou-se no levantamento bibliográfico as definições dos principais conceitos, as diferenças entre os modelos de baterias, microprocessadores utilizados, bibliotecas, linguagem de programação, sensores e motores. Dessa forma foi possível dar uma base teórica para escolhas ao longo do Trabalho de Conclusão de Curso.

O desenvolvimento do código é constituída como uma das partes principais do projeto. Após definido a linguagem Python e o uso da biblioteca OpenCV, focou-se na instalação destas linguagens e bibliotecas. Dessa forma, iniciou-se o processo para o desenvolvimento do código. O desenvolvimento do código tomou como base

observações de outros trabalhos semelhantes e desta forma um cenário final funcional foi atingido.

Foram necessárias diversas funções para o correto funcionamento do código, por exemplo, o processamento de imagem para a detecção da rota e para a detecção de cones de trânsito (laranja/branco) fizeram uso do filtro Canny (CANNY, 1986), operações Morfológicas (FISHER et al., 1996), encontro de contornos (SUZUKI et al., 1985) e mudança de espaço de cor (BAGGIO, 2012; GONZALES; WOODS, 2007; SZELISKI, 2010, p. 90)

A elaboração do protótipo ocorreu de forma quase que paralela ao desenvolvimento do código, visto que não é necessário o código para definir as movimentações básicas do veículo, nem o protótipo para se elaborar partes do código.

Para a elaboração do protótipo fez-se uso de conectores de eletrocalhas, uma vez que se observou que após parafusados, os suportes podem ser entortados e moldados da forma desejada. Assim sendo, elaborou-se um protótipo possível de ser customizado.

Após a elaboração do protótipo realizou-se os testes do código desenvolvido. A ocorrência de falhas e limitações tanto de software quando mecânicas, como o posicionamento da câmera em caso da captação de um reflexo excessivo do piso, por exemplo, levaram a constante alteração e aperfeiçoamento do protótipo.

Após realizados os testes, gráficos foram elaborados e analisados, como pode ser observado na presente monografia.

2 REVISÃO DE LITERATURA

Em conjunto com o desenvolvimento do setor de automação, várias pessoas, empresas e universidades criaram e distribuíram uma vasta gama de informações de forma gratuita. Algumas das principais informações utilizadas foram obtidas destes acessos gratuitos: Trabalhos de Conclusão de Curso, a biblioteca OpenCV e projetos como “Donkey Car” (2017).

2.1 ROBÓTICA

A robótica é uma das áreas que mais influenciam a espécie humana e é definida como a área que estuda o comportamento de uma máquina que realiza movimentos complexos[1]. A atividade executada pela máquina é controlada por um microprocessador uma das únicas criações humanas capazes de processar cenários complexos.

Os robôs fazem uso de dispositivos eletromecânicos e se destacam muito em confiabilidade e facilidade de desenvolvimento com o aperfeiçoar da eletrônica quando comparados com robôs puramente mecânicos.

2.2 VEÍCULOS AUTÔNOMOS

Com o desenvolvimento da eletrônica, a implementação da robótica em diversos setores da sociedade tem se tornando mais evidente e dessa forma, muitos processos que antes eram executados de forma manual, como por exemplo: o ato de dirigir, tendem a se tornar cada vez mais autônomos e assim mais integrados no ramo da robótica.

A implementação de veículos autônomos já é crucial para o uso em fábricas (MILLER; SUBRIN, 1987) de forma a aumentar a produtividade das empresas e se evitar encargos trabalhistas. As etapas para veículos autônomos foram, em ordem cronológica, função Cruise, estacionamento automático e, atualmente com os carros

1 “robot | Definição de robô em inglês pelos Dicionários Oxford” (2018)

da Tesla uma navegação autônoma em autoestradas, isto é: recorrendo aos níveis de automação (NHTSA, 2017).

Embora as empresas estejam tentando lançar um veículo totalmente autônomo, ainda faltam alguns anos para que esse cenário se torne realidade, isso se deve a grande complexidade para o desenvolvimento do veículo e incorporação na sociedade.

A dificuldade do desenvolvimento de um sistema de comunicação robusto entre os veículos e o desenvolvimento de um embasamento legal, por exemplo, são fatores atualmente contrários ao desenvolvimento do veículo autoguiado.

Também existem empresas que um dos principais desafios é o desenvolvimento de baterias mais baratas e com maior densidade energética para assim desenvolver um carro totalmente elétrico, sustentável e com menor custo (TESLA, 2018).

2.3 TIPOS DE SENSORES

Existem diversos tipos de sensores que vêm sido utilizados no mundo da robótica, em específico, ao desenvolvimento de carros autônomos. Os principais tipos de sensores para os carros autônomos são (THRUN et al., 2007; ROBOTEQ, 2018):

- Câmera Convencional
- Câmera Stereo
- LIDAR
- RADAR Automotivo
- RADAR de navegação
- Posição global (Sistema de Navegação Global por Satélite – GNSS)
- Bússola eletrônica
- Encoder
- Sensor de Faixa magnético
- Sensor por Indução (Antena)
- Fototransistor

Todos os sensores apresentados possuem suas vantagens e desvantagens quando aplicados em um veículo autônomo. Dentre todos os tipos de sensores apresentados, os mais utilizados em fábricas são os quatro últimos, sendo os três últimos (sensor de faixa magnético, sensor por indução e sensor óptico) utilizados para o seguimento de linhas.

Uma das vantagens dos sensores de linha quando comparados com os outros apresentados para a navegação, é a complexidade reduzida, de forma que raramente apresentam um mal funcionamento.

Sensores de mapeamento bidimensional (2D) ou tridimensional (3D) como é o caso do LIDAR e RADAR dependem de uma elevada capacidade de processamento, situação essa que frequentemente torna o processo complexo e caro.

Uma abordagem muito utilizada é o uso de inteligência-artificial, que embora normalmente apresente resultados mais satisfatórios para cenários complexos (DARPA, 2018), acaba modificando de forma própria muitos parâmetros, tornando assim difícil de se realizar uma auditoria, verificação do código ou fornecer um embasamento teórico (COHEN, 1991).

Sensores como o LIDAR e RADAR apresentam custo de milhares de reais quando se inclui *hardware* para processamento. Enquanto que os sensores magnético, indução ou óptico, embora possuam um custo menor, não possuem a capacidade de detectar objetos a distância, o que é necessário em ambientes industriais e urbanos.

Dessa forma, para o projeto desta monografia, adotou-se uma câmera como sensor, uma vez que embora outros tipos de sensores pudessem ser utilizados, existem questões relacionadas ao custo, tamanho, demandas e complexidade do projeto. Buscou-se um sensor relativamente econômico e pequeno quando comparado com o LIDAR e RADAR, porém ao mesmo tempo flexível em sua aplicação quando comparado com seguidores tradicionais de linha, que são incapazes, por exemplo, de detectar obstáculos.

2.4 SENSORES PARA SEGUIR LINHAS

Os principais sensores utilizados para seguir linhas são:

- Magnético
- Indução
- Óptico
- Câmera

A seguir é apresentado uma tabela comparativa entre os variados tipos de sensores.

QUADRO 1 – COMPARAÇÃO ENTRE TIPOS DE SENSORES

	Magnético	Indução	Óptico/Fototransistor	Câmera
Tipo do Trajeto	Passivo	Ativo	Passivo	Passivo*
Material do Trajeto	Fita	Condutor	Faixa*	Faixa*
Implementação do Trajeto	Fácil	Difícil	Fácil | Médio	Fácil | Médio
Imune a sujeira	Sim	Sim	Não	Não*
Sensíveis a condição de Luz	Não	Não	Sim	Sim*
Marcadores	Sim	Não	Não	Sim
Detecção de Objetos	Não	Não	Não	Sim

FONTE: ROBOTEQ (2018b) Adaptado.

É importante destacar que o uso de câmera para navegações em linha apresenta como principais vantagens uma alta resolução, a possibilidade de se detectar objetos e a fácil instalação, embora, problemas como sensibilidade a sujeira e principalmente a variações na iluminação sejam os principais aspectos negativos quando comparados com outros sensores como magnético ou indução.

2.5 PROCESSAMENTO DE IMAGEM

Um dos principais sensores que é utilizado para veículos autônomos é a câmera e isto se deve ao fato de que se trata de uma abordagem que até o presente momento é funcional, uma vez que a humanidade faz uso de imagens como principal mecanismo de navegação de veículos.

Iniciado em torno dos anos 60 nos Estados Unidos da América, o processamento de imagem era principalmente utilizado em imagens de satélite, médicas e reconhecimento de caracteres, embora naquela época a capacidade de processamento era reduzida quando comparada com as décadas seguintes (ROSENFELD, 1969).

Com o passar do tempo e desenvolvimento dos computadores, a área de processamento de imagem deixou de ser uma atividade com uso acadêmico e militar (ROSENFELD, 1969) para também ser de uso comercial, e assim o número de aplicações cresceu muito até os dias atuais.

Atualmente, o processamento de imagem evoluiu para o que se chama de visão computacional, isto é, a capacidade de se processar imagens em alto nível e assim realizar operações que envolvem inteligência artificial como identificação de objetos, criação de modelos tridimensionais entre outros.

A principal mudança entre processamento de imagem e visão computacional está na mudança de conceitos, isto é, em processamento de imagem cuida-se mais dos dados brutos, enquanto em visão computacional considera a imagem como uma visão tal qual o ser humano considera, possibilitando assim classificações de objetos, ou seja, a compreensão do que é captado pelo sensor (KLETTE, 2014).

O desenvolvimento de um código pode ocorrer de diversas maneiras, com diversas linguagens e fazendo uso de diversas bibliotecas. Por questões de direitos autorais e constante uso pela comunidade acadêmica e empresarial a biblioteca de código aberto (*Open Source*) OpenCV, é uma das mais utilizadas no mundo no que se refere ao processamento de imagem (BRADSKI; KAEHLER, 2000).

2.6 OPENCV

O OpenCV é uma biblioteca constantemente atualizada, de código aberto e disponível para as 3 linguagens mais utilizadas em programação, isto é Python, C++ e Java. O OpenCV é uma biblioteca robusta e confiável pela constante revisão dos códigos realizada pela comunidade acadêmica.

As principais funções utilizadas ao longo do projeto foram: as funções morfológicas, detector de bordas Canny, Identificação de Contornos, Localização dos centros e momentos, Comparador de Amostras (*matchTemplate*), filtragem por espaço de cor Matriz, Saturação e Valor (*Hue, Saturation and* Value – HSV) e envolveram para a correta movimentação do protótipo: Modulação Por Largura de Pulso (*Pulse Width Modulation* – PWM), *General Purpose Input/Output* (GPIO) (PAJANKAR; KAKKAR, 2016) e teclado (HOWSE, 2013, p. 33).

2.7 LINGUAGEM PYTHON

Observou-se ao longo dos projetos três tipos principais de linguagem de programação quando aplicada para processamento de imagem. Como cada linguagem possui características tanto positivas quanto negativas se buscou a escolha da linguagem mais adequada à biblioteca OpenCV e ao microprocessador escolhido, considerando o tempo de processamento e facilidade de implementação (PRECHELT, 2000). As três principais linguagens são:

Elas são:

- C++
- Java
- **Python**

Dentre as linguagens apresentadas, optou-se pelo Python pelos seguintes motivos:

- O programador já havia feito uso dessa linguagem anteriormente
- Quando comparado com o C++, a linguagem Python é consideravelmente mais rápida de ser implementada, ainda que o tempo de processamento é

consideravelmente mais elevado caso não sejam utilizadas bibliotecas otimizadas como o Numpy.

- A linguagem Python diferente das outras duas não necessita da declaração da variável, tornando dessa forma a implementação mais ágil.

2.8 MICROPROCESSADOR

Microprocessador é um processador em escala reduzida que possui a capacidade de executar funções de cálculos. O microprocessador é o maior desenvolvimento da engenharia eletrônica desde o seu início, e com o desenvolvimento deste tipo de componente a velocidade de processamento, o consumo energético e a confiabilidade têm aumentado consideravelmente desde a primeira versão desenvolvida em 1971 pela Intel (FERREIRA, 1998).

O uso de microprocessadores é essencial para o setor de processamento de imagens e mais especificamente a visão computacional. Isso se deve a necessidade de cálculos de alta complexidade e o grande número de dados envolvidos.

Com o desenvolvimento dos microprocessadores várias empresas e universidades iniciaram o desenvolvimento de microprocessadores mais baratos e com maior acesso para fins educacionais ou de pesquisa. O projeto Raspberry é um exemplo disso.

Sendo uma placa de desenvolvimento muito utilizada e com uma enorme comunidade ativa, o Raspberry Pi tem uma abordagem código livre e de baixo custo, de modo que é utilizado largamente em meios universitários, desde o nível de técnico ao nível de mestrado. As principais características da versão utilizada (Raspberry Pi 3B+) são:

- Modulo de Câmera facilmente conectável e disponível
- Baixo consumo energético
- Wi-fi, *Universal Serial Bus* (USB) e Bluetooth já acoplados
- Baixo custo (próximo de R$ 300 nas lojas oficiais)
- Grande bibliografia disponível
- Dimensões reduzidas

2.9 MICROCONTROLADOR

Semelhante a estrutura de um microprocessador, o microcontrolador possui a capacidade de realizar cálculos rápidos com um pequeno consumo energético. Porém diferente de um microprocessador, o microcontrolador normalmente apresenta uma capacidade de processamento mais reduzidas e normalmente a capacidade de operar diretamente em atuadores e sensores.

O microcontrolador para uso em veículos autoguiados é interessante, uma vez que, muitos sinais após adquiridos, são processados por este, e assim utilizados para o acionamento dos atuadores que levam a realização de operações normalmente mecânicas. Muitas vezes esse tipo de procedimento reduz o número de dados a serem processados pelo microprocessador. No projeto em questão, o microcontrolador não foi utilizado pois o microprocessador adotado é capaz de se comunicar diretamente com o Driver H L298 em um tempo aceitável para o projeto, além de o número de atuadores ser reduzido (em número de 2).

2.10 DRIVER

O Driver possui como objetivo traduzir sinais elétricos providos normalmente por uma unidade de processamento, operadas em tensões e correntes reduzidas, para acionar dispositivos que exijam uma grande potência, corrente ou tensão. Sendo a estrutura de um driver constituída de diodos e transistores.

O Driver H L298 é um circuito integrado com 15 terminais que suporta tensões e correntes elevadas para a eletrônica. O driver é constituído de uma ponte H que aceita comandos lógicos, tecnologia Lógica Transistor-Transistor (TTL), e possui uma tensão mínima de 2.3 V para o acionamento da ponte, a tensão a ser destinada para o atuador pode possuir valores entre 2.5 e 46 V, a corrente máxima aceita é de 3 A e a capacidade máxima de dissipação é de 25 W [ST, 201?]

Escolheu-se o driver em questão por ser um modelo que aceita uma grande flexibilidade na tensão de alimentação para ser utilizada nos motores, e por aceitar uma tensão de acionamento mínima de 2.3 V, isto é um valor aceitável para o microprocessador utilizado, Raspberry Pi, que faz uso de 3.3 V em suas terminações

(GPIO) conforme RaspberryPi [201?]. A questão de se encontrar uma ponte H já pronta e que faz uso de um driver robusto para o projeto em questão também foi outro fator fundamental para essa escolha.

2.11 COMUNICAÇÃO

Observou-se ao longo do projeto diversas formas de se realizar a comunicação entre o microprocessador e o computador utilizado para visualizar as imagens e o código. Essa comunicação é necessária e **vantajosa** pelos seguintes motivos:

- Facilidade de debugar o código
- Possibilidade de observar o processamento em tempo real e identificar eventuais falhas
- Possibilidade de comunicar com o microprocessador utilizado comunicação sem fio (*Wireless*).
- Possibilidade de se utilizar teclados, mouses e monitores do computador / notebook.

As **desvantagens** de se utilizar esse tipo de abordagem são:

- Maior complexidade do que programar diretamente sobre o microprocessador,
- Possível necessidade de cabos USB ou roteadores
- Perda em processamento por parte do microprocessador

Dessa forma, como se considerou a comunicação sem fio mais vantajosa, adotou-se uma comunicação por Wi-Fi realizando-se a alteração do código por meio do protocolo *Secure Shell* (SSH) e *Virtual Network Computing* (VNC).

2.11.1 Wireless

O sistema de comunicação utilizado foi sem fio, e sendo mais específico a tecnologia IEEE 802.11, ou vulgarmente conhecida como Wi-Fi. Esse meio de comunicação foi utilizado por ser compatível tanto pelo microprocessador quando pelo computador utilizado.

Uma alternativa de comunicação poderia vir a ser o Bluetooth embora essa escolha seja não recomendada para a transmissão de imagens uma vez que a taxa de transferência é mais reduzida. Desta forma utilizou-se por meio da tecnologia IEEE 802.11 os protocolos VNC e SSH.

2.11.1.1 SSH

O protocolo Secure Shell é um protocolo desenvolvido para o uso seguro de login e outros serviços de rede operando sobre uma rede insegura. Possui uma arquitetura Cliente-Servidor. O protocolo SSH faz uso de uma criptografia de chave pública (YLONEN; LONVICK, 2005).

O uso do protocolo SSH foi essencial ao longo do projeto uma vez que se considerou a possibilidade de que transmissão de dados como posição do centro da faixa se dá de forma ágil o suficiente para não interferir de forma negativa no funcionamento do protótipo. Embora seja um protocolo que possibilite uma rápida taxa de transmissão de dados e uma necessidade de processamento baixo, ele não possibilita a visualização de imagens, ainda que seja possível fazer uso do terminal do microprocessador.

2.11.1.2 VNC

O *Virtual Network Computing* (VNC) é um sistema de compartilhamento de desktop gráfico, que faz uso do protocolo *Remote Frame Buffer* (RFB) de forma que é possível operar e visualizar o desktop de um outro computador por meio de uma rede de computador. É possível por meio da rede VNC transferir movimentos do Mouse, Teclado e algumas vezes até arquivos (RICHARDSON et al., 1998).

O uso desse sistema de compartilhamento se observou necessário ao longo do projeto uma vez que o protocolo SSH por não possibilitar a transmissão de imagens não possibilitava observar o funcionamento do protótipo em tempo real. O sistema VNC por tornar possível esse tipo de abordagem faz com que a identificação de falhas em tempo real se torne possível, ainda que por se tratar de um processo mais complexo, existe uma redução na performance do protótipo.

2.12 BATERIAS

Com o desenvolvimento da eletrônica e a redução das dimensões dos equipamentos, diversas empresas e governos observaram a possibilidade de se utilizar dos equipamentos eletrônicos de forma portátil. Porém para que esses equipamentos possam serem utilizados é necessário uma fonte de energia para a alimentação desses equipamentos. Dessa forma destacam-se duas opções para aplicações em robótica:

- Geração da energia de forma local (normalmente painéis solares)
- Armazenamento da energia para uso local (baterias químicas como Chumbo-Ácido, Íon de Lítio e Polímero de Lítio são as baterias mais utilizadas pelo seu preço e densidade energética embora tenha outras formas de se armazenar energia).

No protótipo em questão adotou-se baterias de Íon-Lítio, devido ao custo baixo e densidade energética razoável quando comparados com outros tipos de baterias.

O uso de painel solar no caso é inadequado, uma vez que para a geração de 15 W médio ao longo de um dia seria necessário uma área de 0,75 m^2 (MacKay, 2008), isso se utilizado ao ar aberto, situação essa inadequada tanto para o protótipo devido ao tamanho, quanto para a maioria dos veículos autoguiados em fábricas, dessa forma as baterias foram carregadas por meio de conexões na rede de 127 V. Por questões ecológicas e de custo, foram analisadas apenas baterias recarregáveis.

2.12.1 Chumbo-Ácido

A bateria chumbo ácido é uma das mais antigas do mundo, foi inventada em 1859 e por ser antiga e apresentar um funcionamento relativamente simples é uma das baterias que já atingiram seu estágio de maturação. Isto é, não se espera grandes aperfeiçoamentos na tecnologia (DELL; RAND, 2001).

Embora antiga, a bateria de chumbo ácido é uma das mais utilizadas no mundo. Quase 10% do armazenamento de energia elétrica no mundo para sistemas de distribuição utiliza esse tipo de bateria, estando atrás apenas da bateria de íon-lítio (ZHANG et al., 2017).

As principais desvantagens referente a bateria de chumbo ácido são:

- Grande impacto ambiental se descartadas erroneamente,
- Baixa densidade de energia
- Efeito Memória,
- Baixo número de ciclos.

Embora o custo desse tipo de bateria seja um dos mais reduzidos, a massa é elevada para muitas das aplicações, incluindo para o protótipo em questão.

2.12.2 Bateria de Íon de Lítio

As baterias de Íon de Lítio (Li-Oon) possuem uma grande aplicação em diversos setores partindo desde o uso em armazenamento de energia para redes de distribuição e transmissão ao uso em carros elétricos (ZHANG et al., 2017). É a bateria mais utilizada no mundo, tanto para o sistema elétrico, quanto para dispositivos eletrônicos. Possui um preço módico quando comparado entre as de Polímero de Lítio e a de Chumbo-Ácido, e uma boa densidade de energia com um número de ciclos relativamente elevado.

2.12.3 Bateria de Polímero de Lítio

As baterias de Polímero de Lítio (Li-Po) possuem uma grande aplicação no que se refere a rôbos de competição tanto de batalha quanto de corrida por serem baterias mais resistentes tanto ao aumento de temperatura, quanto ao impacto. A principal desvantagem desse tipo de bateria atualmente é o custo mais elevado quando comparado com a bateria Li-Ion. A Bateria de Li-Po apresenta uma densidade energética semelhante ao de Íon-Lítio (BECK, 2018).

2.13 MOTORES E LOCOMOÇÃO

A locomoção é uma das funções essenciais para um veículo que tem como propósito se mover. Dessa forma atentou-se consideravelmente para os detalhes dos possíveis motores a serem adotados. Não apenas para o protótipo em questão mas também para possíveis veículos em escala real, como veículos autônomos.

2.13.1 Motores

Existe uma grande variedade de motores, eles podendo ser principalmente de combustão ou elétricos, pelo fato de frequentemente os motores de combustão serem maiores descarta-se dessa forma esse tipo de motor para o protótipo e dessa forma faz-se o uso de um motor elétrico DC pela facilidade e preço de obtenção do mesmo.

Ao longo do trabalho focou-se no comparativo entre motores elétricos, embora uma análise mais realista e completa devem abordar todos os modelos de motor tanto elétricos quanto a combustão.

Normalmente são utilizados 3 tipos de motores com fins de locomoção na robótica eles são: motor de indução com rotor em gaiola (CRIM), motor síncrono de imã permanente (PMSM) e o motor de corrente contínua (DC). Cada motor possui sua característica conforme a TABELA 1.

Muito embora o motor PMSM seja o mais recomendado para a aplicação em questão, as variáveis envolvido para a elaboração da tabela se referem a motores de grandes dimensões e potências, isto é, aplicado para veículos autônomos. Uma vez que o protótipo em questão apresenta dimensões reduzidas, 22 cm por 27 cm, o motor com menor custo é o DC Motor e por questões financeiras e a desnecessidade do uso de um conversor, o Motor DC foi o adotado ao longo do projeto.

TABELA 1 - COMPARAÇÃO ENTRE MOTORES ELÉTRICOS

	Motor DC	CRIM	PMSM
Densidade de Potência	2.5	3.5	5
Eficiência	2.5	3.5	5
Controlabilidade	5	4	4
Confiabilidade	3	5	4
Custo	4	5	3

FONTE: CHAN (2002)

2.13.2 Locomoção

Na robótica existem diversas formas que possibilitam a movimentação dos robôs em solo. Elas são deslize, rolamento, pulo, caminhada, corrida e rastejo. O presente trabalho adotará o uso de rolamento, por meio de rodas pelos seguintes motivos:

- Maior simplicidade na montagem,
- Maior confiabilidade por possuir menos partes móveis,
- Maior controle,
- Baixo custo,
- Boa locomoção em terrenos regulares,
- Pequeno gasto energético.

Para aplicações industriais normalmente se adota a locomoção por rolamento também, muito embora para terrenos irregulares ou instáveis como areia fofa, outras possibilidades como caminhada também possa ser utilizado (SIEGWART et al., 2004).

3 MATERIAIS E MÉTODOS

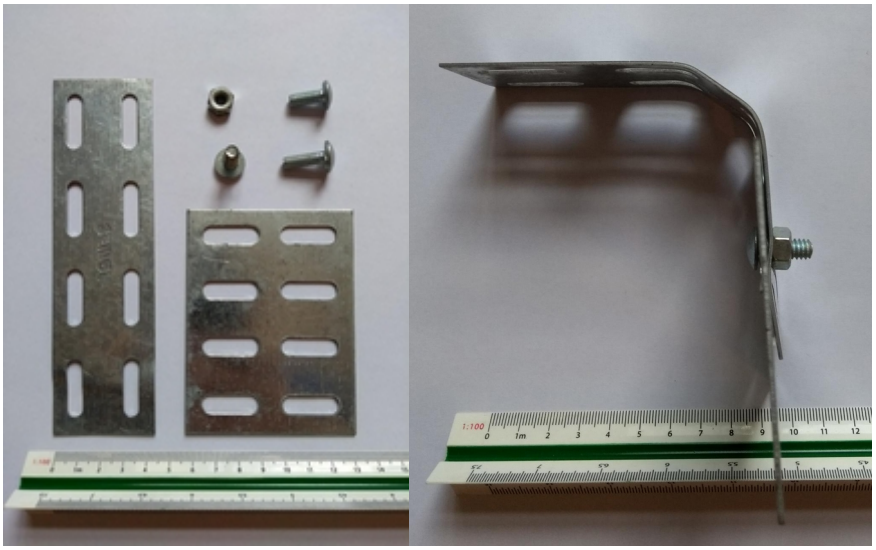
Esse capítulo tem como objetivo apresentar e descrever os principais acessórios, equipamentos e procedimentos utilizados para a elaboração e desenvolvimento do Trabalho de Conclusão de Curso, assim como do protótipo de veículo autoguiado correspondente. O protótipo em questão tem como objetivo apresentar de forma simples a movimentação de um veículo autônomo utilizado em fábricas, ainda que possa servir como base para projetos que envolvam mobilidade urbana. O protótipo elaborado possui as principais características de um projeto em grande escala, isto é o uso e existência de um motor, driver, carenagem, rodas, baterias, câmera e principalmente microprocessador.

Como o protótipo em si já envolve as principais características de um veículo de porte maior, decidiu-se deixá-lo da forma mais simples e efetiva possível, isto é com duas rodas com tração independente frontal, uma roda traseira sem tração. A carenagem adotada foi tal de forma que a modelagem e modificação do projeto se dê de forma rápida e efetiva. O código utilizado no protótipo foi elaborado de forma que pudesse ser o mais rápido e preciso possível no tempo disponível. A seguir serão apresentados com maior detalhe a montagem do protótipo.

3.1 CARENAGEM

De modo a possibilitar uma montagem e desmontagem rápida assim como aberto a modificações, decidiu-se realizar a montagem do protótipo por meio de talas de ementa para eletrocalhas. A fixação entre as talas se deu por meio de parafusos franceses e roscas M6, FIGURA 1.

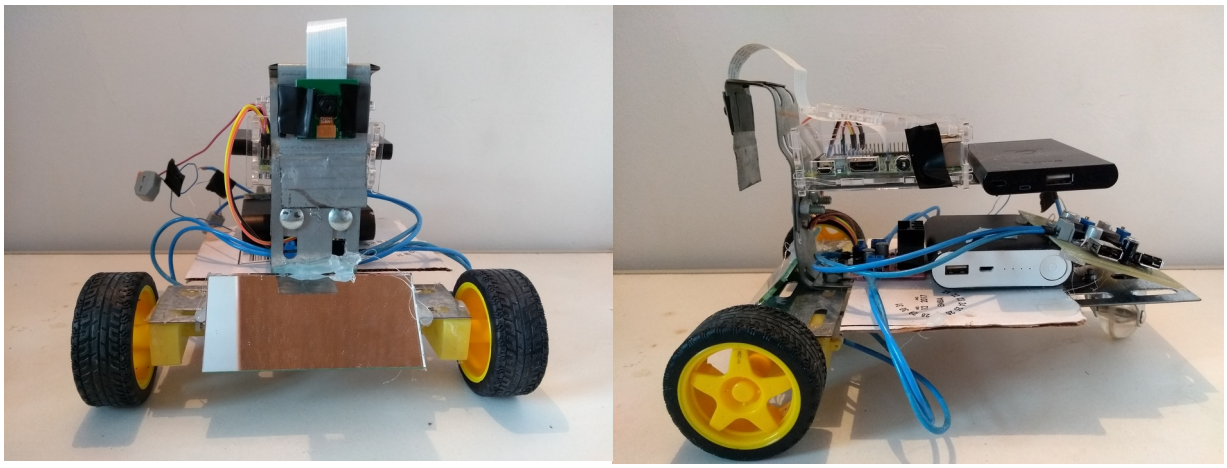
FIGURA 1 - TALAS DE EMENTA PARA ELETROCALHAS



FONTE: O autor (2018).

Por meio do uso das talas de eletroduto foi possível montar a estrutura, FIGURA 2, que é utilizada como suporte a todos os eletrônicos utilizados, assim como as rodas e ao rodízio. É interessante destacar que a câmera está posicionada a uma altura de 15 centímetros do chão.

FIGURA 2 - PROTÓTIPO UTILIZADO



FONTE: O autor (2018).

3.2 NÚMERO DE RODAS

O número de rodas adotado ao longo do projeto foi de três por possibilitar uma maior mobilidade, menor massa, um custo mais reduzido e uma montagem mais fácil. Normalmente as indústrias automotivas fazem uso de quatro rodas para se aumentar a estabilidade do veículo e a distribuição do peso, situação essa que pode ser desconsiderada no protótipo em questão por possuir uma massa de apenas 1,18 kg.

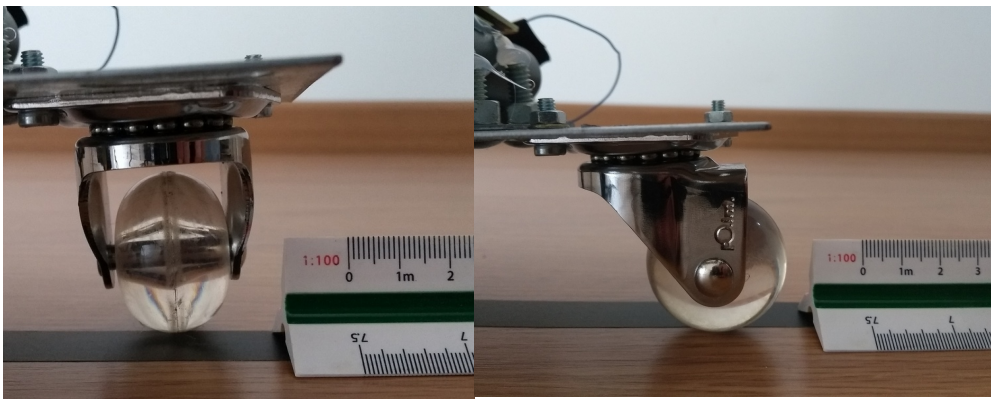
3.3 POSICIONAMENTO DAS RODAS

Embora seja muito comum realizar o posicionamento das duas rodas de tração na parte traseira do protótipo, realizar esse tipo de abordagem reduz consideravelmente a mobilidade do veículo, isto porque posicionando as rodas de tração na frente é possível realizar uma rotação completa de forma que o piloto ou a câmera se mantenha no eixo de rotação.

3.4 RODÍZIO E RODAS

Embora existam diferentes tipos de rodas e abordagens para que a região traseira do protótipo apresente uma mobilidade adequada, adotou-se o uso de um rodízio, FIGURA 3, pela facilidade e custo para a obtenção.

FIGURA 3 - RODÍZIO PARA SUPORTE TRASEIRO



FONTE: O autor (2018).

O tamanho do rodízio é muito importante para o correto funcionamento do protótipo, isto se deve ao fato de que se observou que, quanto maior o rodízio menos vibrações são transmitidas para a câmera do protótipo. O excesso de vibrações leva ao surgimento de borrões nas imagens.

Essa situação também pode ser resolvida por meio de processamento da imagem, porém, não é o recomendado uma vez que se trata de um processamento pesado e o custo para a aquisição de um processador mais rápido é mais elevado do que a simples mudança estrutural do protótipo.

As rodas de tração adotadas ao longo do projeto possuem um raio de 3,25 cm e largura de 2,5 cm.

3.5 MOTOR

Assim como apresentado na revisão bibliográfica, por questão de custo e facilidade de obtenção, optou-se por adotar um motor DC com caixa de redução de 48 para 1. Observa-se que a tensão de alimentação máxima recomendada é de 6 V.

Dessa forma foi elaborada uma tabela entre a relação de PWM, a frequência de rotação e a velocidade linear (roda de 3,25 cm de raio) para uma tensão de alimentação de 5.95 V, tabela 2. Os dados coletados correspondem a roda frontal esquerda.

TABELA 2 - RELAÇÃO ENTRE PWM E FREQUÊNCIA DE ROTAÇÃO

PWM	Frequência (Hz)	Velocidade (cm/s)
0%	-	-
10%	-	-
20%	0,516	10,5
30%	0,959	19,6
40%	1,363	27,8
50%	1,682	34,4
60%	1,931	39,4
70%	2,142	43,7
80%	2,353	48,0
90%	2,489	50,8
100%	2,872	58,6

FONTE: O autor (2018).

Observa-se também por meio da tabela 3 o consumo energético do Drive para uma tensão de alimentação de 5 V para cada um dos PWMs apresentados, assim como o consumo do drive sem o acionamento dos motores, isto é no PWM 0% um consumo de 40 mA.

TABELA 3 - RELAÇÃO ENTRE PWM E CORRENTE DEMANDADA

PWM	Corrente (mA)
0%	40
10%	100
20%	120
30%	150
40%	170
50%	190
60%	200
70%	210
80%	220
90%	230
100%	240

FONTE: O autor (2018).

3.6 BATERIAS

Inicialmente adotou-se no projeto uma bateria da Inova, Íon-Lítio, que possui capacidade de 10400 mAh para uma tensão de 3.6 V ou 37.44 Wh. Observou-se que durante o uso de apenas uma bateria era possível operar o protótipo satisfatoriamente por 40 min, após esta duração o microprocessador desligava de forma aleatória.

Observou-se também que dos dois elevadores de tensão XL6009 utilizados, um apresentava um sobreaquecimento inesperado fazendo com que a corrente de entrada variasse entre 1A a 3A, isso sem o funcionamento do microprocessador. Dessa forma para se aumentar a confiabilidade da fonte para o microprocessador, implementou-se uma bateria da Sony, Lítio-Polímero, de 5000 mAh para uma tensão de 5 V, isto é 25 Wh e realizou-se a mudança do cabo micro-usb. Após essas modificações foi possível realizar testes no protótipo por períodos em torno de 3 horas.

3.7 CONVERSOR DE TENSÃO

Ao longo do projeto, inicialmente decidiu-se pela escolha de dois elevadores de tensão XL6009, isto porque a tensão fornecida no USB pela bateria da Inova é de 5 V porém com uma corrente de 2 A, devido a um conversor interno a bateria, o que era insuficiente para atender as demandas do projeto. Dessa forma decidiu-se fazer-se uso das baterias sem o uso do elevador interno, e sim por meio de um elevador externo que suporta uma maior capacidade de corrente, no caso o XL6009 [XLSEMI, 201?].

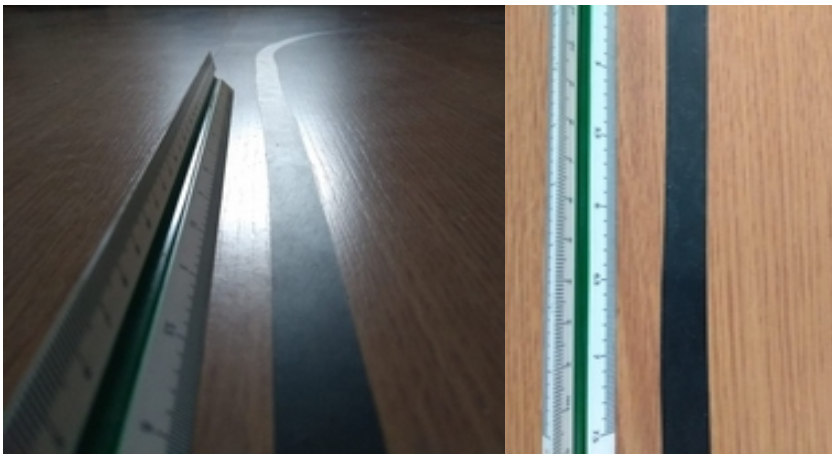
Dessa forma um conversor era utilizado para a alimentação do driver em 6 V, enquanto o outro para a alimentação do microprocessador em 5 V. O conversor utilizado pelo microprocessador apresentou falhas nas últimas semanas de testes e foi substituído por uma bateria capaz de fornecer a corrente necessária.

3.8 POSICIONAMENTO DA CÂMERA

O posicionamento da câmera foi um dos detalhes cruciais ao longo do projeto. As principais dificuldades encontradas para o posicionamento da câmera foram: o tamanho do cabo disponível para a conexão da câmera ao microprocessador, a altura da câmera devido ao reflexo encontrado no chão e a inclinação da câmera. Duas alternativas foram consideradas, apontar a câmera para frente ou apontar a câmera para baixo.

Observou-se que a câmera apontando para frente, dependendo do piso, está muito suscetível ao brilho, a esquerda da FIGURA 4. Porém, apontar a câmera para frente torna possível a previsão do trajeto assim como a detecção dos obstáculos, muito embora não identifique a posição atual do protótipo.

FIGURA 4 - REFLEXO OBSERVADO NO PISO



FONTE: O autor (2018).

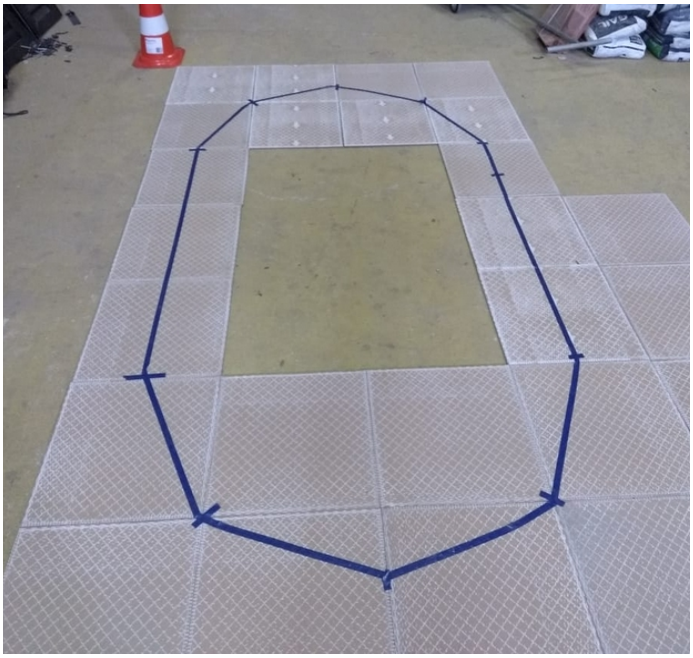
Em contra partida, com a câmera apontando verticalmente para baixo é detectado pouco reflexo, a direita da FIGURA 4, embora leve ao aumento do borrão na imagem quando em movimento e requer uma resposta mais rápida por parte do microprocessador para que o sistema de controle não se torne instável.

Para o protótipo em questão posicionou-se a câmera direcionada para frente com uma leve inclinação (menor do que 10º) para baixo, uma vez que foi possível reduzir o brilho por meio do uso de filtros digitais e mudança do piso da pista por meio do uso do azulejo invertido, isto é, a região vitrificada encostada no piso.

3.9 MONTAGEM DA PISTA

Observou-se que após a montagem de diversas pistas existem duas situações inadequadas para o correto funcionamento do protótipo, elas são: piso muito irregular ou um piso muito liso. A situação do piso irregular é indesejada devido à dificuldade de movimentação por parte do protótipo, além de dificultar o funcionamento da câmera caso ocorram vibrações. Por sua vez, um piso muito liso, normalmente leva uma situação onde o reflexo é elevado e assim torna difícil a identificação da faixa por parte do microprocessador. Dessa forma utilizou-se o lado de baixo de um revestimento cerâmico por possuir baixa reflexão e ser mais regular, FIGURA 5.

FIGURA 5 - PISTA UTILIZADA PELO PROTÓTIPO

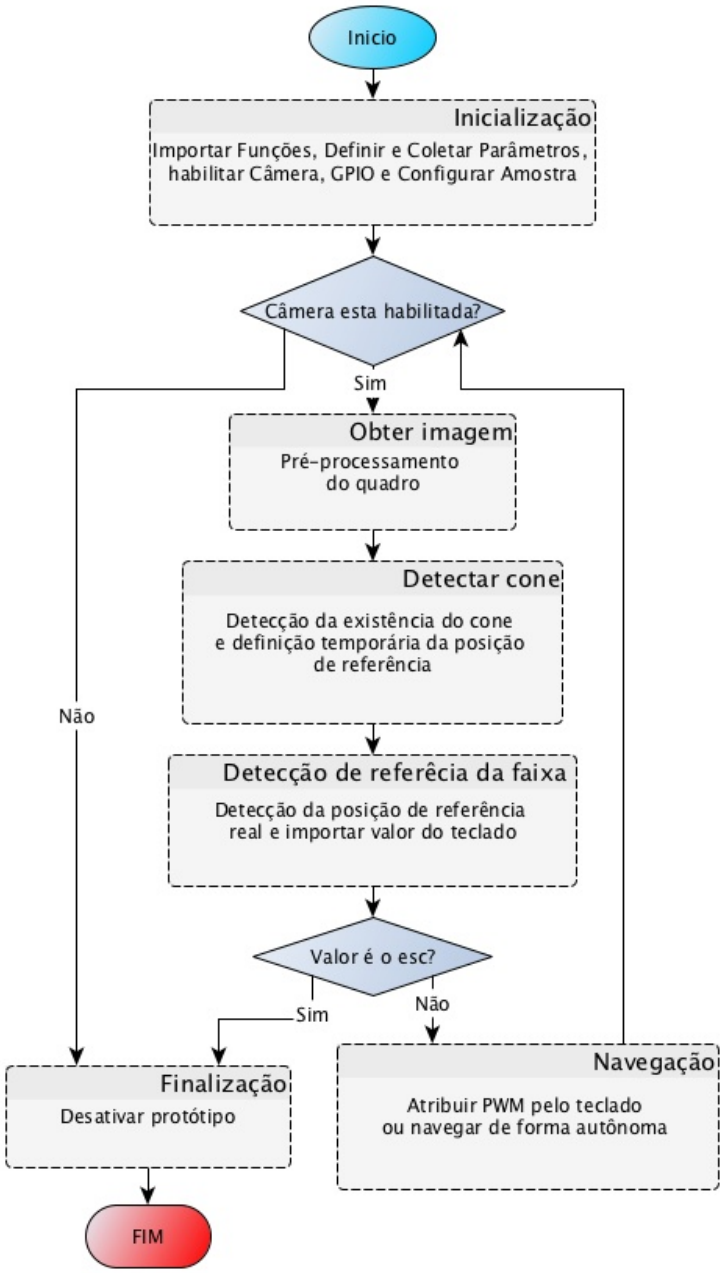


FONTE: O autor (2018).

3.10 DESENVOLVIMENTO DO ALGORITMO

Após apresentadas o funcionamento das funções, é possível agora apresentar os resultados obtidos e as etapas adotadas no código para que apresente os resultados desejáveis. De uma forma em geral por ser tratar de um código relativamente complexo o número de procedimentos adotados é elevado, isto é em torno de 80. Dessa forma decidiu-se organizar o código em agrupamentos, que totalizam 11. Por questões de objetividade é apresentado no texto principal apenas o Fluxograma Geral na FIGURA 6. As principais subdivisões estão do apêndice 1 ao 6.

FIGURA 6 - FLUXOGRAMA GERAL



FONTE: O autor (2018).

É possível observar no fluxograma geral que a principal parte do código é constituído de 6 agrupamentos, eles são para a inicialização do código, obtenção de imagem, detecção do obstáculo, no caso o cone, detecção da faixa, navegação e a finalização. Dentre todos os agrupamentos utilizados foram implementadas funções e procedimentos principais que será descrito a seguir.

3.10.1 Importação das bibliotecas

Utilizado logo no início do algoritmo (APÊNDICE 1), as bibliotecas utilizadas foram principalmente OpenCV e Numpy, visto que são as mais utilizadas para o processamento de imagem, como também são uma das bibliotecas com maior revisão bibliográfica científica (CULJAK et al., 2012). A biblioteca OpenCV é a responsável pelas principais funções relacionadas ao processamento de imagem, enquanto a Numpy foi largamente utilizada para operações em matrizes, como ordenar e encontrar números específicos.

3.10.2 Configuração da câmera

O principal motivo para a escolha da câmera do módulo do Raspberry Pi é a grande flexibilidade nas configurações. Dentre todas as configurações adotadas as principais características adotadas foram: definição do número de quadros por segundo, saturação da imagem, brilho e sensitividade ISO (JONES, 2018). Essas escolhas foram interessantes para que a câmera pudesse apresentar uma melhor performance nas condições de teste, ainda que, o algoritmo desenvolvido apresente um funcionamento adequado sem o uso dessas configurações quando o ambiente está bem iluminado.

A câmera adotada corresponde a câmera do próprio módulo do Raspberry Pi de forma que torna possível a modificação das configurações apresentadas. As configurações foram utilizadas no início do algorítimo, vide no fluxograma o agrupamento inicialização, APÊNDICE 1

3.10.3 Redimensionamento

Uma etapa muito importante é o redimensionamento da imagem, esta etapa de processamento é utilizada em 2 grupos, na inicialização APÊNDICE 1, e obtenção de imagem, APÊNDICE 2, para que posteriormente seja aplicado para a detecção da referência da faixa, APÊNDICE 4.

O redimensionamento da imagem é necessário, visto que, embora seja possível importar a imagem com uma resolução já razoavelmente reduzida, isto é 320 por 240 é possível otimizar o código ainda mais por meio da redução da imagem para 180 por 135 cuja detecção da faixa ainda é aceitável. O aspecto negativo de se realizar o redimensionamento é a perda da precisão da faixa, isto é, no caso em questão, quando utilizada uma resolução de 320 por 240 é possível obter uma precisão de 1 mm, enquanto que realizando um redimensionamento da imagem para 180 por 135, perde-se essa resolução para um valor próximo de 1.8 mm, porém com um tempo de processamento 3 vezes menor.

O uso do redimensionamento não foi utilizado para a detecção do Cone, visto que se viu desnecessário para a tarefa adotada, deve-se considerar também que para a detecção de objetos a perda de resolução reduz consideravelmente a confiabilidade e a identificação de objetos a uma distância mais elevada, gerando assim mais falsos positivos, ou falsos negativos.

A questão do tempo de processamento da imagem para a detecção do cone não é crucial, visto que capturar 2 FPS leva ao protótipo a percorrer no máximo 50 cm entre as detecções, situação esta a qual não leva a uma mudança brusca do cenário.

3.10.4 Espaço de cor HSV

A mudança do espaço de cor torna possível o uso de filtros por tons em imagens, isto é filtrar se a imagem é vermelha, laranja, azul entre outras cores. A mudança de espaço de cor foi utilizada tanto para a detecção da faixa azul, FIGURA 7, quanto para o cone por meio de um filtro laranja, FIGURA 8, 9 e 10.

Esse processo foi utilizado na inicialização APÊNDICE 1, detecção do cone, APÊNDICE 3, e detecção da referência da faixa, APÊNDICE 4.

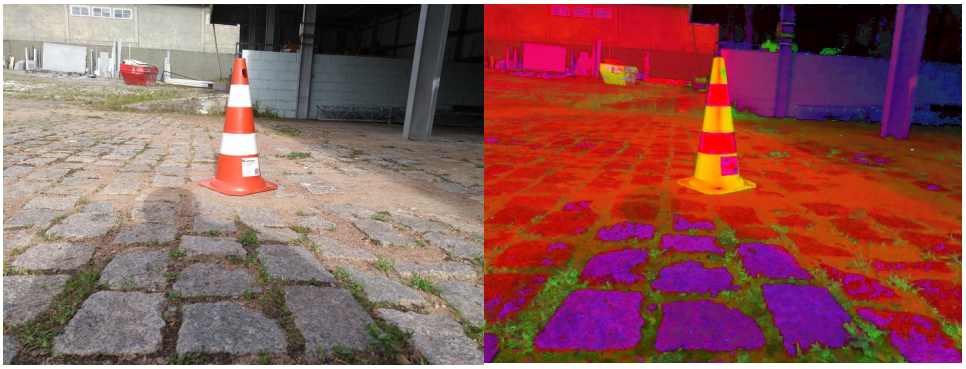
FIGURA 7 - FILTRO AZUL POR ESPAÇO DE COR

FONTE: O autor (2018).

FIGURA 8 - FILTRO LARANJA POR ESPAÇO DE COR NO RASPBERRY

FONTE: O autor (2018).

FIGURA 9 - MUDANÇA DO ESPAÇO DE COR PARA HSV



FONTE: O autor (2018).

FIGURA 10 - MASCARA PELO FILTRO DE COR E SOBREPOSIÇÃO

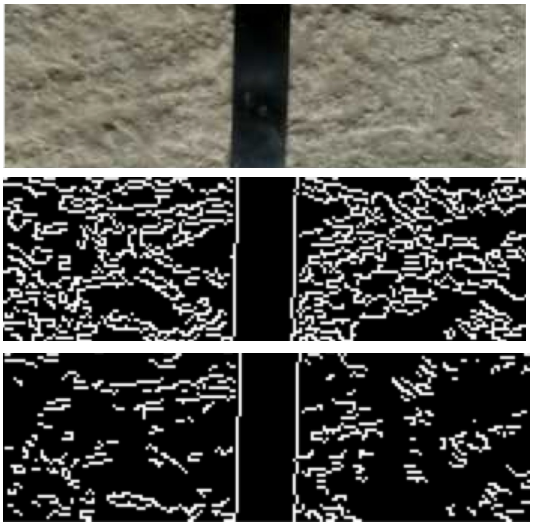
FONTE: O autor (2018).

3.10.5 Filtro Canny

O detector de bordas Canny assim como apresentado na revisão bibliográfica, página 24, é um dos detectores de borda mais utilizados no mundo. O uso desse detector de bordas foi utilizado para duas principais tarefas no algoritmo, isto é encontrar as bordas da faixa utilizada no chão e a detecção do cone.

O uso de um detector de bordas Canny para faixas filtra ainda mais as imagens obtidas, pois torna possível considerar apenas as bordas das faixas, mudando-se os parâmetros de *threshold*, FIGURA 11.

FIGURA 11 - FILTRO CANNY - CONCRETO



FONTE: O autor (2018).

A filtragem por meio apenas da cor embora muitas vezes seja satisfatório é ineficaz caso a cor do plano de fundo seja semelhante ou apresente ruídos da mesma cor da faixa.

Foi crucial o uso do filtro Canny para identificar a borda das faixas e posteriormente a detecção da referência da faixa, APÊNDICE 4,

3.10.6 Funções Morfológicas

As funções morfológicas, que foram utilizadas para a detecção da faixa, APÊNDICE 4, e do obstáculo, APÊNDICE 3, e tornou possível que o algoritmo

despreze detalhes indesejados na imagem como sujeiras e faixas que estão em posições horizontais, assim como tornou possível a união de contornos de interesse.

No caso da detecção do centro foi necessário que se realizasse uma dilatação de 1 por 11 (largura por altura) nas bordas detectadas e uma erosão 5 por 3, dessa forma como a erosão vertical é maior do que a horizontal detalhes na imagem que não possuam um vizinho na região superior ou inferior acabam sendo removidos conforme é possível observar nas figuras 12 e, 13.

A dilatação horizontal é realizada de forma que o posterior procedimento de erosão remova características de pontos e pequenas faixas horizontais brancas.

FIGURA 12 - DILATAÇÃO MORFOLÓGICA - CONCRETO

FONTE: O autor (2018).

FIGURA 13 - EROSÃO MORFOLÓGICA - CONCRETO

FONTE: O autor (2018).

No caso da detecção do cone é possível observar que houve ao todo três processos morfológicos, que são na sequência de processamento: erosão, direita da FIGURA 14, dilatação, esquerda da figura 15, e erosão novamente, direita da figura 15.

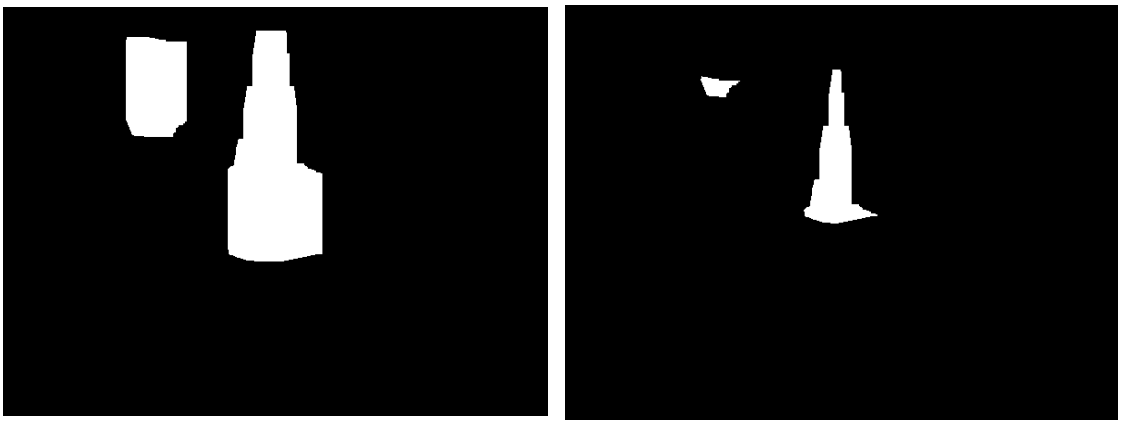
A primeira erosão é realizada de modo que se reduza regiões laranjas das quais não se tem interesse. A dilatação seguinte é utilizada de modo a se unir as

regiões não capturadas pelo filtro de cor, que é o caso da região branca presente no cone. A erosão final é utilizada para se retornar as dimensões originais dos cones.

FIGURA 14 - EROSÃO MORFOLÓGICA - CONE

FONTE: O autor (2018).

FIGURA 15 - DILATAÇÃO MORFOLÓGICA - CONE



FONTE: O autor (2018).

3.10.7 Encontrar Contornos

A detecção dos contornos foi utilizada tanto para a detecção da faixa, APÊNDICE 4, quanto para a detecção do obstáculo, APÊNDICE 3, no caso o cone.

No caso da detecção da faixa após realizado a dilatação das bordas da imagem e realizado uma filtragem de detalhes horizontais é possível observar a existência de polígonos os quais normalmente estão posicionados na mesma posição que as bordas das faixas detectados. Como se espera que os filtros utilizados removeram ou reduziram o tamanho de polígonos e detalhes não

desejados é possível selecionar os maiores polígonos os quais normalmente correspondem as faixas. Dessa forma selecionando os dois maiores ou o maior, dependendo da distância da câmera a faixa[2], é possível que normalmente se encontre os dois lados da faixa. E se não forem os dois seriam ao menos um dos lados que, embora não ideal, é satisfatório para a navegação do protótipo.

A ordenação por tamanho também é feita para a detecção do obstáculo, dessa forma é possível identificar o maior polígono encontrado na imagem que corresponda a cor laranja, o que normalmente é o cone.

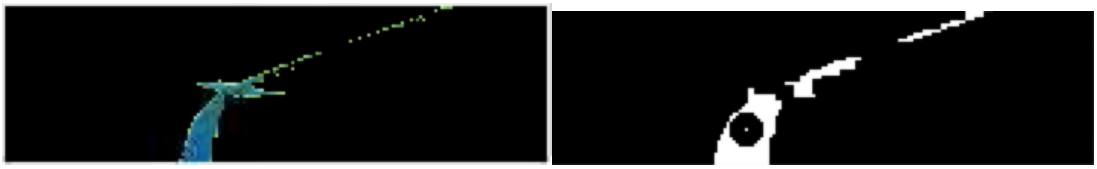
3.10.8 Momento e Centroide

Após ordenado de forma decrescente ao tamanho das áreas dos polígonos e selecionado os dois maiores ou o maior, é necessário a detecção do centro desses. Dessa forma utilizou-se uma função que detecta o centro dos polígonos por meio de um calculo do momento. Dessa forma obtêm-se o centro e a posição dos polígonos desejados da imagem, de forma mais específica o centro das bordas das faixas, APÊNDICE 4.

FIGURA 16 - DETECÇÃO DOS POLÍGONOS MAIORES - CONCRETO

FONTE: O autor (2018).

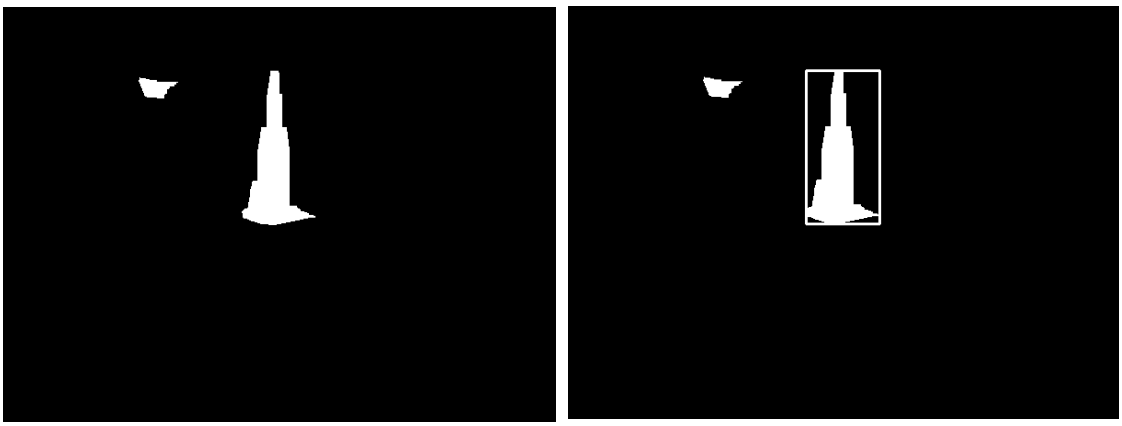
FIGURA 17 - DETECÇÃO DO POLÍGONO MAIOR - AZULEJO



FONTE: O autor (2018).

2 Esse detalhe deve ser configurado manualmente no algorítimo proposto.

FIGURA 18 - DETECÇÃO DO POLÍGONO MAIOR - OBSTÁCULO



FONTE: O autor (2018).

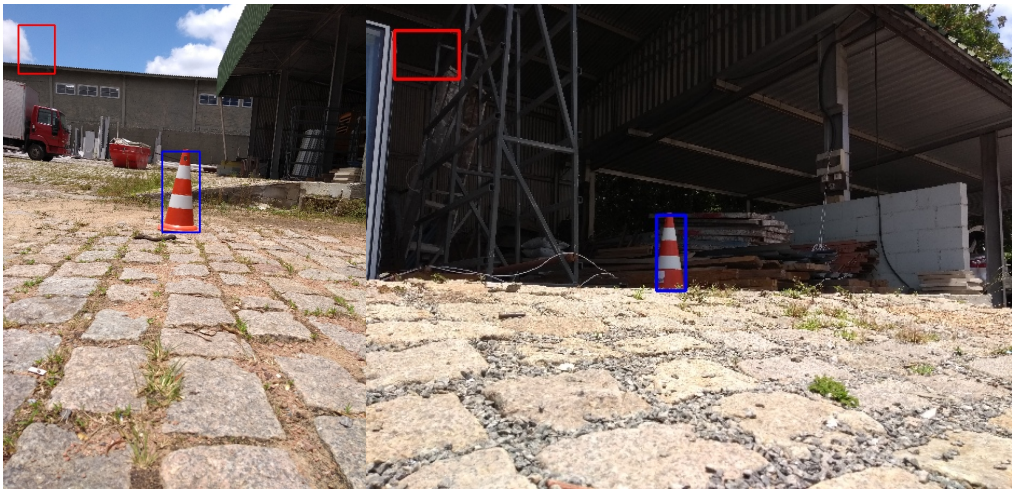
3.10.9 Média Aritmética dos Centros

No caso da detecção do centro da faixa para os casos em que a câmera está próxima ao chão é realizado uma média aritmética simples, visto que o centro da faixa é dado como a distância média entre os centros dos dois maiores polígonos, APÊNDICE 4.

3.10.10 Detecção do Cone

Embora inicialmente tenha se tentado adotar a detecção do cone por meio do método de Comparação de Modelos (*Match Template* – CV_TM_CCOEFF), [OPENCV, 201?]**.** Detectando-se a região que apresenta o maior pico de energia, observou-se ineficaz para a detecção de cones, das quais destacam-se 2 principais motivos: as imagens com grandes dimensões apresentam um tempo de processamento muito elevado e com resultados com um número elevado de falsos positivos e o modelo adotado possui poucas caraterísticas, isto é não apresenta nenhuma irregularidade senão a região das bordas, tornando o método ineficaz.

FIGURA 19 - DETECÇÃO REALIZADA POR AMBOS OS MÉTODOS



FONTE: O autor (2018).

O problema da incompatibilidade do tamanho da imagem do modelo e da imagem de detecção foi resolvido por meio da mudança de escala da imagem modelo, ainda que sem resultado positivo, retângulo em vermelho na FIGURA 19.

Dessa forma adotou-se outra abordagem para a solução do problema de detecção de cones, isto é, por meio da mundana de escalas de cores para HSV e posterior filtro para as cores laranjas, retângulo em azul na FIGURA 19.

3.10.11 Função PWM e Sistema de controle

Após processado e identificado a posição do centro da faixa e do Cone é possível fazer-se uso de um sistema de controle do protótipo. A comunicação do microprocessador ao driver se dá por PWM, de forma que 100% do PWM corresponde a tensão máxima, no caso 6 V e 0% corresponde a tensão mínima de 0 V. Ao todo se utilizou 4 PWMs, dois para a movimentação frontal, e outros dois PWMs correspondem a movimentação de ré. A movimentação de ré só foi aplicada para o controle do protótipo pelo operador por meio do teclado. Essa movimentação foi utilizada por questões de praticidade e a possibilidade de se observar a movimentação do veículo em situações específicas.

O sistema de controle adotado foi obtido por meio de tentativa e erro, porém sempre analisando as imagens geradas e o trajeto percorrido pelo protótipo de

forma que se observou que um sistema de controle Proporcional, Integral e Derivativo (PID) poderia ser uma alternativa viável. Dessa forma, foi possível obter um controle aceitável para que o protótipo percorresse a pista de forma satisfatória, APÊNDICE 5 para o fluxograma da navegação e APÊNDICE 6 para a finalização da navegação/código.

3.11 DESAFIOS ENCONTRADOS

Um dos principais desafios encontrados correspondeu a dificuldade do posicionamento satisfatório da câmera e uma forma de se reduzir o reflexo e o brilho causado pelo sol, ou lâmpada no chão, parede ou diretamente à câmera.

Outro fator considerável se deu devido a detecção da câmera sempre a frente do protótipo de forma que o robô acaba tomando sempre decisões antecipadas, uma vez que a curva que seria para ser realizada quando o protótipo está sobre a curva, acaba sendo realizada 20 centímetros antes de chegar a ela.

3.12 MATERIAL UTILIZADO

Houve muitos softwares e equipamentos utilizados ao longo do projeto, dentre eles se destaca o grande uso de softwares *Open Source.* Os principais recursos utilizados foram:

- **Hardware**:
  - Módulo de câmera (Raspberry Rev 1.3),
  - Notebook (2.7 GHz Intel Core i5, 8 GB memória DDR3),
  - Microprocessador/PC (Raspberry Pi 3B+),
  - Estrutura do veículo.
- **Software**:
  - LibreOffice, Zotero e yEd,
  - Ambiente de Desenvolvimento Integrado (IDE) PyCharm e Thonny,
  - GNU nano,
  - VNC Viewer, FileZilla,
  - Biblioteca Numpy e OpenCV,

4 RESULTADOS

Os principais resultados obtidos se resumem a performance do código e movimentação do protótipo. As principais características destacadas são a qualidade da detecção, tempo de processamento, velocidade do veículo e estabilidade para seguir a linha. Dentre todas as características abordadas foi possível atingir as demandas tanto do protótipo em termos de velocidade, quanto em termos de processamento.

4.1 TEMPORIZAÇÃO

Após realizado toda a elaboração do código, os ajustes e as otimizações que se observaram relevantes ao longo do projeto, decidiu-se temporizar as principais etapas encontradas para que dessa forma fosse possível observar o tempo de processamento do código não apenas para melhorias futuras, mas também para a demonstração dos resultados em relação a proposta do TCC. Dessa forma obteve-se uma amostra do tempo de processamento para um ciclo de 1000 quadros.

A realização da temporização se deu nas seguintes situações: uma através da exibição das imagens por meio do VNC e utilizando a IDE Thonny, nativa do microprocessador adotado, uma sem a exibição de imagens e utilizando a IDE, uma com a exibição de imagens por meio do terminal e outra por meio do código SSH sem a exibição das imagens, sendo que para a última a função gráfica estava desativada no Raspberry Pi 3B+. O código padrão corresponde ao código em que a cada 10 quadros é realizada a detecção do cone em questão.

É interessante observar no GRÁFICO 5 que dentre todas as situações utilizadas, o tempo para adquirir a imagem é um dos tempos mais elevados, isto é sempre entre 50000 e 70000 us por quadro ou entre 20 e 14 FPS, soluções para esse tipo de limitação podem ser implementadas por meio do uso de processos paralelos e *multithread* que no caso não foram implementados por se focar principalmente no processamento dos obstáculos e faixas. A exibição das imagens também foi um fator negativo para o tempo de importação, isto porque ao exibir a

imagem parte do processamento deixa de ser ocupado para a importação e passa a ser dedicado a exibição, reduzindo o tempo em torno de 20000 us.

Observa-se por sua vez que o tempo médio do código é próximo de 100000 us em situações em que se exibe a imagem, GRÁFICOS 1 e 3, e de apenas 80000 us quando não são exibidas, GRÁFICOS 2 e 4. Embora exista essa diferença de tempo, por questões de praticidade realizou-se os testes com a exibição das imagens.

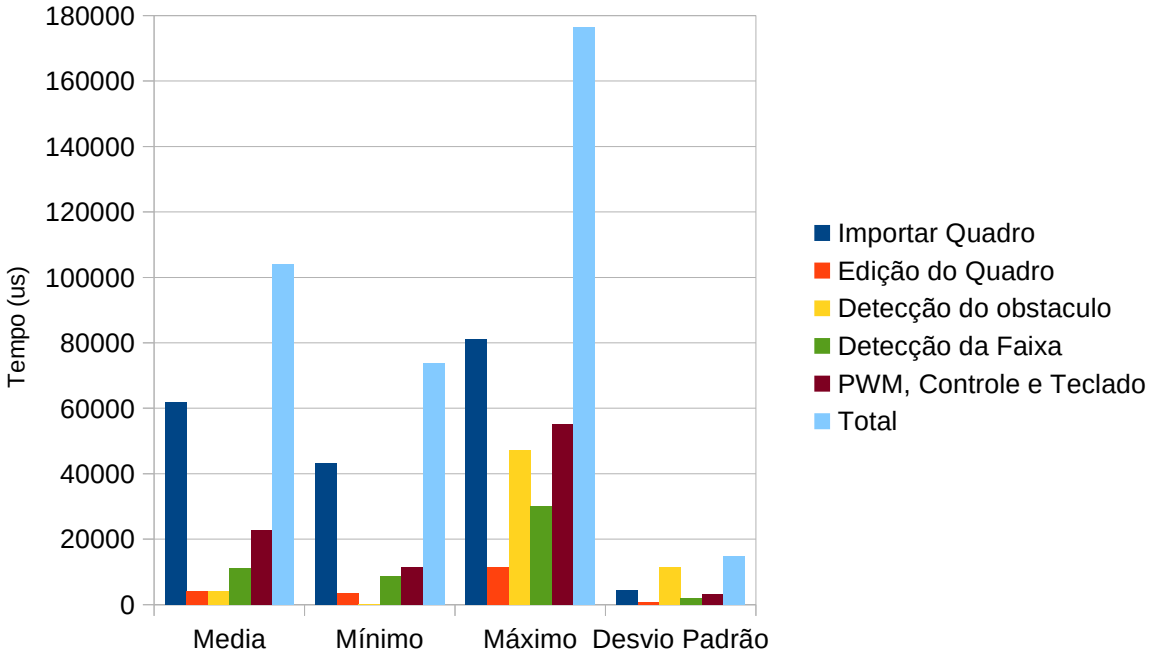
Outro fator importante de ser destacado é que o tempo de processamento para a edição do quadro e detecção do centro ou a detecção do obstáculo são consideravelmente reduzidas, isto é atingindo no máximo 41491 us ou 24 FPS para a identificação da faixa e 58704 us ou 17 FPS para a detecção do obstáculo, GRÁFICO 1, este longo tempo de processamento se da principalmente pelo fato de muitos processos de exibições estarem sendo considerados nas etapas de detecção do centro e do cone.

O processo mais rápido por sua vez, que se refere a não exibição de imagens GRÁFICO 4, foi possível obter tempos muito reduzidos, isto é, mesmo para as situações mais demoradas um tempo para a edição do quadro e detecção do centro ou a detecção do obstáculo de 22283 us ou 45 FPS, e 45090 us ou 22 FPS respectivamente.

O uso da IDE por sua vez não afetou consideravelmente a velocidade do tempo médio no código de modo que a diferença foi de apenas 1 % no tempo total nos processos em que se exibia a imagem. O tempo médio total consumido fazendo-se uso do terminal foi 103311 us e fazendo-se uso da IDE foi de 104105 us.

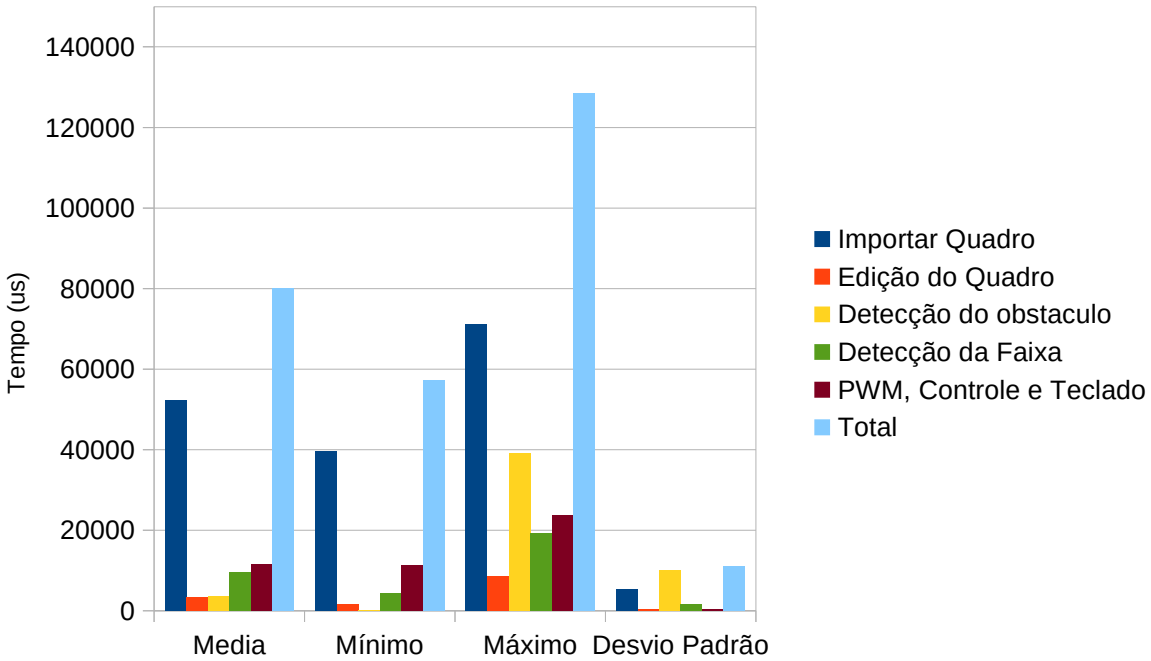
Observou-se também, por meio do GRÁFICO 6 que caso a detecção de obstáculos seja implementada a cada quadro o tempo de processamento se torna em torno de 40000 us mais lento o que inviabiliza uma navegação autônoma devido ao sistema de controle se tornar instável, dessa forma decidiu-se realizar a detecção do obstáculo a cada 10 quadros, assim se pode demonstrar uma correta navegação e funcionamento da detecção do cone.

GRÁFICO 1 - TEMPORIZAÇÃO DO CÓDIGO PADRÃO COM EXIBIÇÃO DE IMAGEM, COMUNICAÇÃO VNC E USO DA IDE



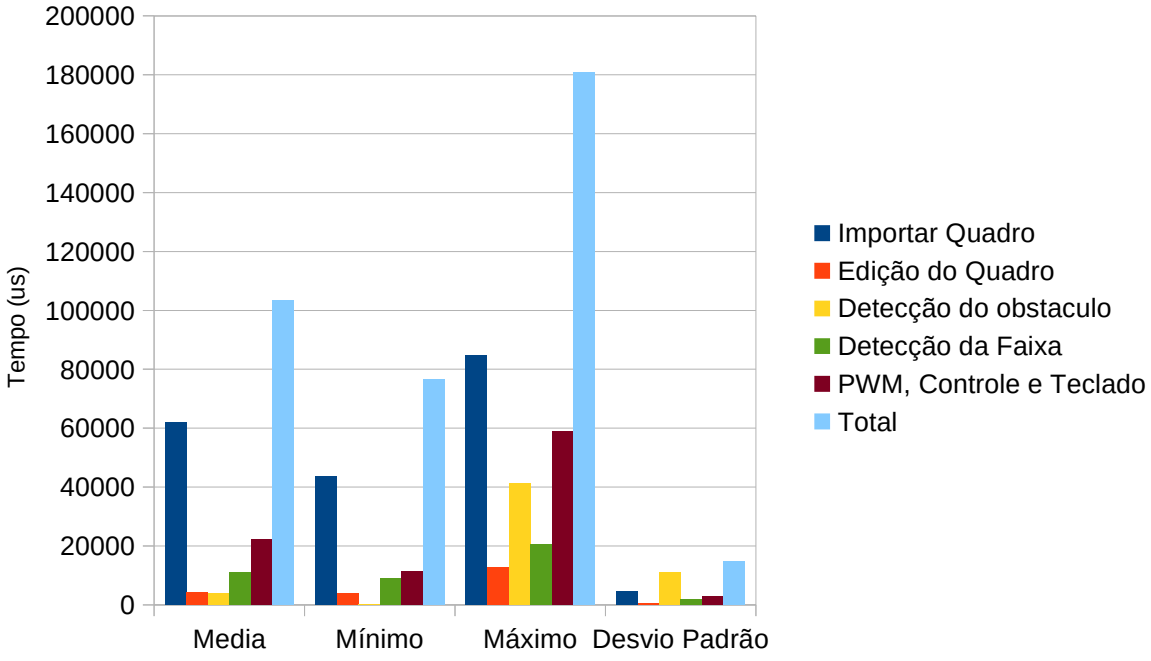
FONTE: O autor (2018).

GRÁFICO 2 - TEMPORIZAÇÃO DO CÓDIGO PADRÃO SEM EXIBIÇÃO DE IMAGEM, COMUNICAÇÃO VNC E USO DA IDE



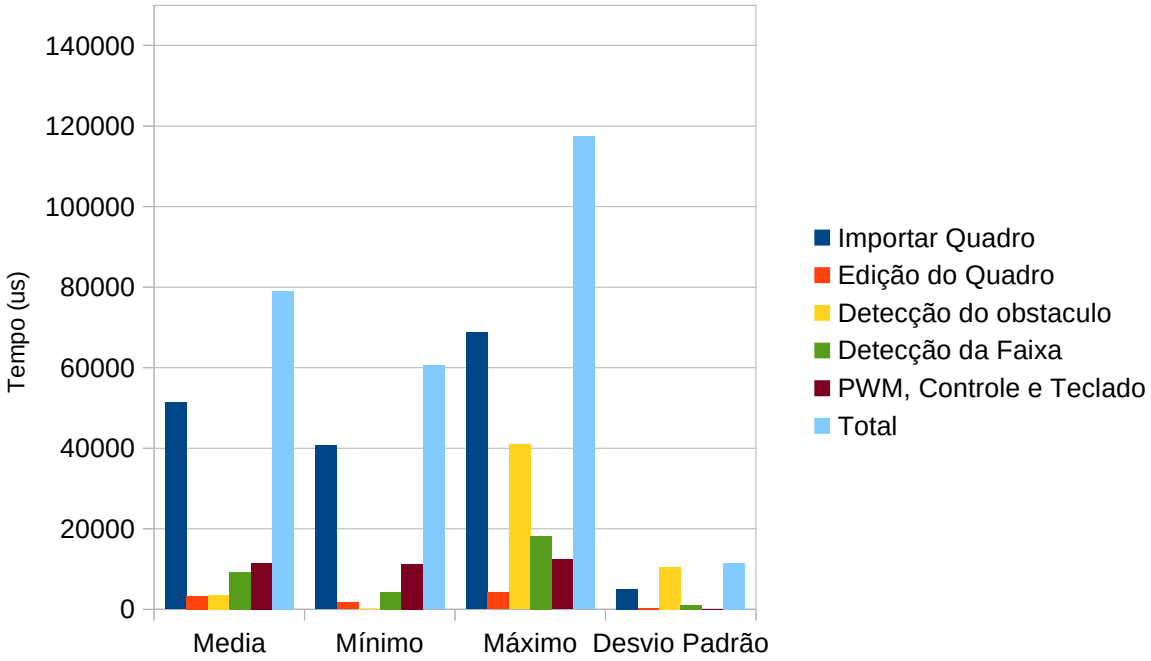
FONTE: O autor (2018).

GRÁFICO 3 - TEMPORIZAÇÃO DO CÓDIGO PADRÃO COM EXIBIÇÃO DE IMAGEM, COMUNICAÇÃO VNC E USO DO TERMINAL



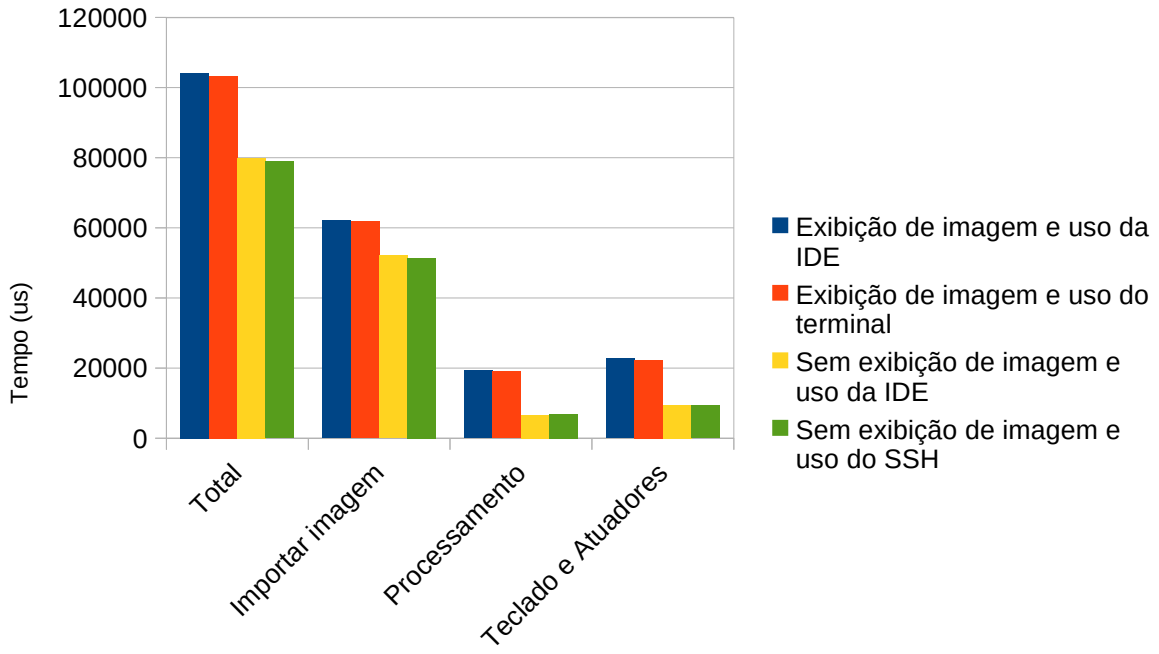
FONTE: O autor (2018).

GRÁFICO 4 - TEMPORIZAÇÃO DO CÓDIGO PADRÃO SEM EXIBIÇÃO DE IMAGEM E COMUNICAÇÃO SSH



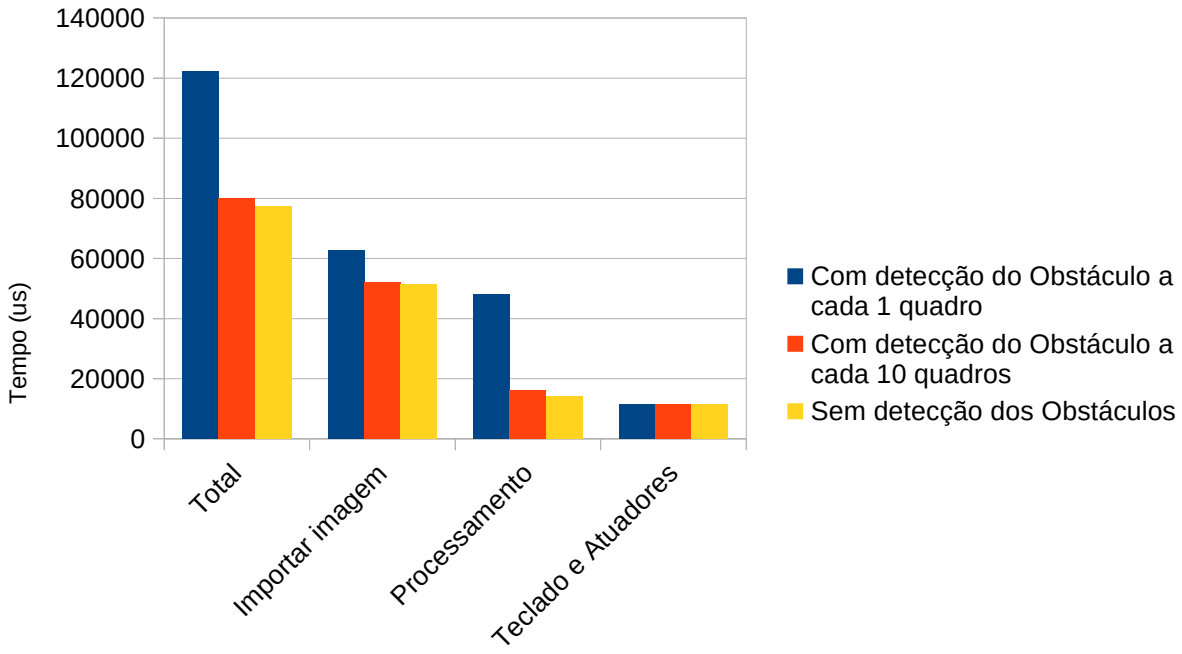
FONTE: O autor (2018).

GRÁFICO 5 - GRÁFICO COMPARATIVO DO CÓDIGO PADRÃO NA IDE



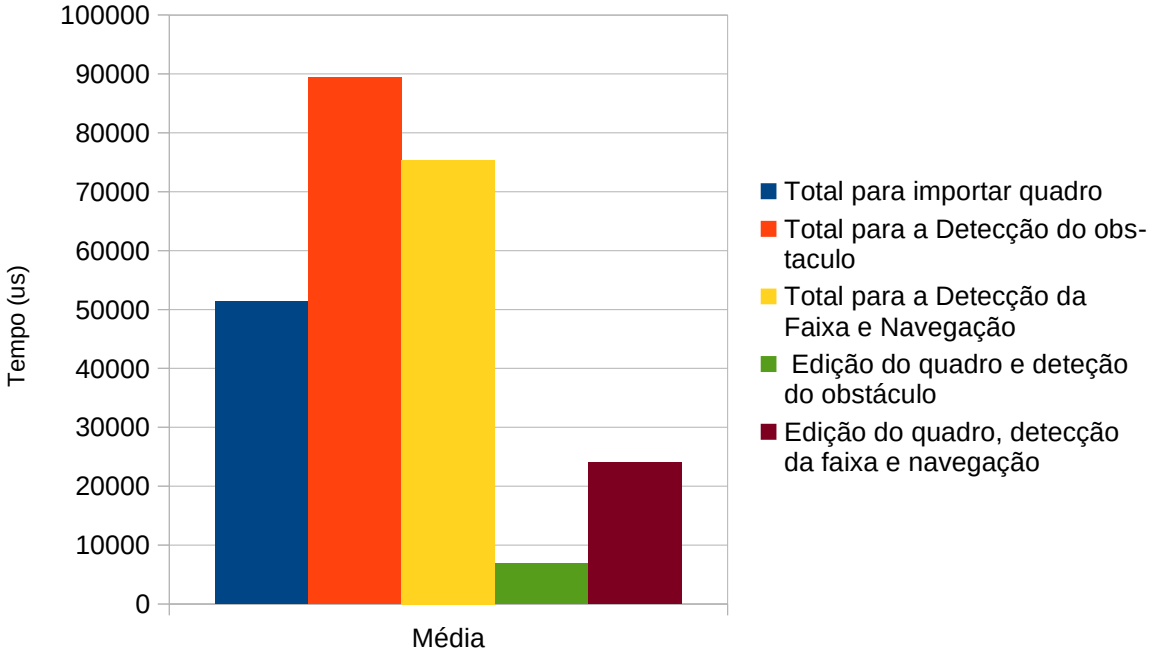
FONTE: O autor (2018).

GRÁFICO 6 - GRÁFICO COMPARATIVO SEM EXIBIÇÃO DE IMAGEM NA IDE



FONTE: O autor (2018).

GRÁFICO 7 - TEMPORIZAÇÃO UTILIZANDO O PROTOCOLO SSH



FONTE: O autor (2018).

Como foi proposto uma velocidade de processamento superior a 12 FPS para a detecção da faixa, e se obteve 45 FPS apenas para o processamento da faixa e 13 FPS incluindo a importação de imagem e, para a detecção do cone que se propôs 2 FPS se obteve 22 FPS sem incluir a importação e 11 FPS incluído a importação da imagem, GRÁFICO 7. Conclui-se que o tempo de processamento quando utilizado o protocolo SSH atinge o proposto.

4.2 MOVIMENTAÇÃO DO PROTÓTIPO

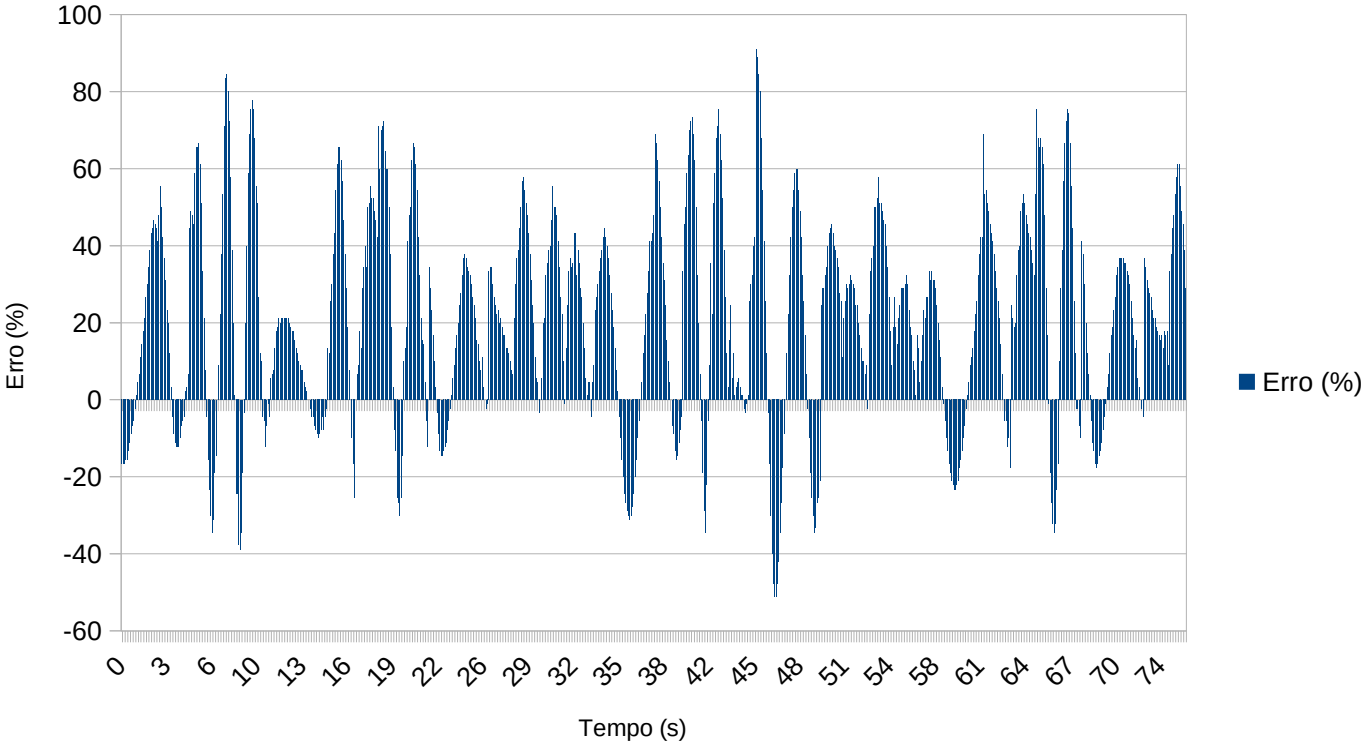
Após realizada a montagem do protótipo e o desenvolvimento satisfatório do algorítimo é necessário o teste em campo do protótipo. Para a realização desse teste realizou-se uma montagem de uma pequena pista. O comprimento da pista é de 8,2 metros.

Por meio dos testes foi possível observar que para 5.96 V de alimentação o PWM médio foi de 70,6% para a roda esquerda e 34,2% para a roda direita. Ao todo,

para a pista que corresponde a 8,2 metros, o protótipo acabou levando 232 segundos para realizar 9 voltas, 73,8 m, e dessa forma a velocidade obtida é de 32 centímetros por segundo, isto é 110 % superior ao proposto nos objetivos.

Ao longo de uma das navegações, GRÁFICO 8, foi possível observar que o erro médio em relação ao centro da faixa foi de 27,7%, que corresponde em torno de 4,5 cm, observou-se também um desvio padrão de 26,9 e valores máximos de 91% e um erro mínimo de -51%.

GRÁFICO 8 - ERRO COLETADO AO LONGO DE 25 METROS



FONTE: O autor (2018).

4.3 IDENTIFICAÇÃO DO OBSTÁCULO

Após implementado o método para a identificação dos cones, foi possível aplicar o método para centenas de fotos e observar um resultado satisfatório para a maioria das imagens e cenários testados, tanto em imagens de alta resoluções fotografadas por meio do celular quanto nas imagens capturadas pela câmera do

protótipo. Todas as imagens captadas pelo celular apresentavam a existência do cone, embora nem sempre tenha sido detectado.

O número de imagens utilizadas para a correção dos parâmetros foi em torno de 50 sendo que após os ajustes em 36 (72%) imagens o objeto foi adequadamente detectado, o número de falsos positivos foi de 4 (8%) e o número de falso negativo foi de 10 (20%).

Posteriormente decidiu-se fazer-se o teste em 78 imagens adicionais as quais não foram utilizadas nenhuma vez anteriormente de forma a se evitar alguma manipulação inadequada do resultado. Dentre elas 57 (73%) captaram adequadamente o cone, o número de falsos positivos por sua vez foi de 11 (14%) e o de falsos negativos 10 (13%).

FIGURA 20 - DETECÇÃO SATISFATÓRIA DO CONE – CELULAR

FONTE: O autor (2018).

FIGURA 21 - DETECÇÃO SATISFATÓRIA DO CONE – CELULAR

FONTE: O autor (2018).

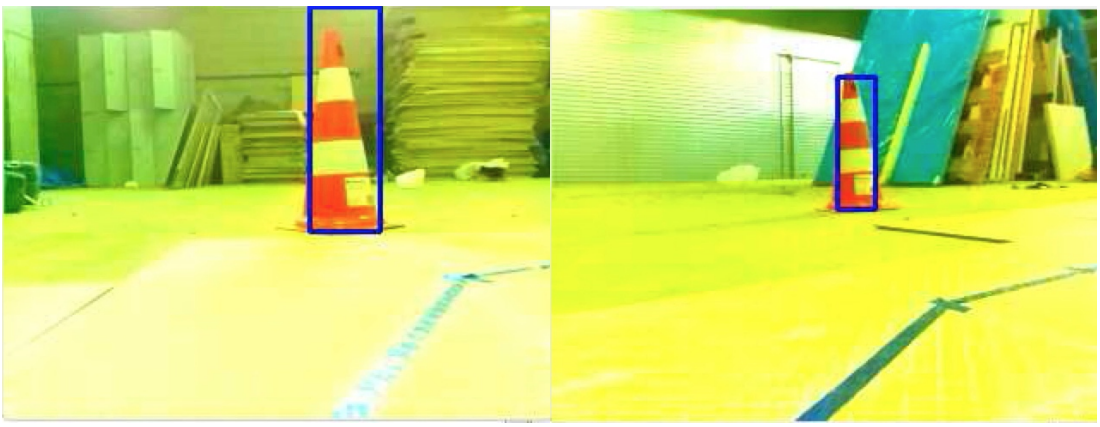
FIGURA 22 - DETECÇÃO INSATISFATÓRIA DO CONE - CELULAR

FONTE: O autor (2018).

FIGURA 23 - DETECÇÃO INSATISFATÓRIA DO CONE - CELULAR

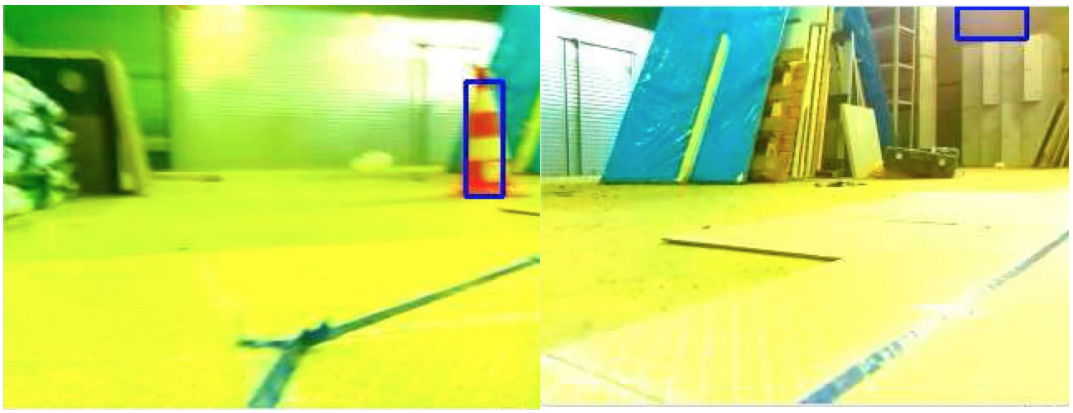
FONTE: O autor (2018).

FIGURA 24 - DETECÇÃO SATISFATÓRIA DO CONE – MODULO RASPBERRY



FONTE: O autor (2018).

FIGURA 25 - DETECÇÃO DO CONE – MODULO RASPBERRY



FONTE: O autor (2018).

É interessante destacar algumas limitações observadas pelo método adotado. As principais limitações e imperfeiçoes da detecção são atualmente: o número de cones detectados na imagem, verificação do formato da região detectada (direita da FIGURA 21), incapacidade de distinguir cores muito similares (FIGURA 22, direita da FIGURA 25), dificuldade de identificar cones em ambientes com menor iluminação (esquerda da FIGURA 23) ou imagens a qual obstáculo apresente um tamanho reduzido (direita da FIGURA 23). Essas limitações não foram solucionadas ou pelo tempo despendido para a solução, ou por necessidade de se aprimorar ainda mais o método, por meio de normalização da imagem, no caso da iluminação e Redes Neurais para a detecção do formato do cone.

O número máximo de cones identificado para cada imagem é de apenas um por se considerar apenas o contorno de maior área da imagem. O formato da região identificada não foi adotado como critério devido à necessidade de se implementar outros métodos mais complexos. A dificuldade de se distinguir cores similares se dá devido à necessidade da câmera se adequar ao tipo de iluminação adotado, e dessa forma foi necessário utilizar um filtro de cor mais tolerante. As imagens que apresentam um cone em tamanho reduzido não apresentam uma identificação adequada pela necessidade de se remover ruídos de pequena dimensão e dessa forma o cone acaba sendo confundido por um ruído.

Referente as imagens adequadamente identificadas é interessante destacar que dificilmente apresentavam um cenário com as limitações descritas

anteriormente. Exemplos de imagens com boa taxa de sucesso são: FIGURA 20, esquerda da FIGURA 21, FIGURA 24 e esquerda da FIGURA 25, ainda que a última apresente um leve borrão de movimento.

Referente a detecção por meio do formado do cone, tentou-se utilizar o método do CV_TM_CCOEFF e CV_TM_CCOEFF_NORM na região já destacada pela mudança de espaço de cor, porém não se viu resultado satisfatório provavelmente pela ausência de características e irregularidades do obstáculo escolhido, FIGURA 19 em vermelho.

5 CONCLUSÃO

O projeto atendeu aos objetivos apresentados uma vez que navegou de forma autônoma com velocidade superior aos 15 centímetros por segundo, atingindo 32 centímetros por segundo. Apresentou uma taxa de processamento superior a 12 FPS para a identificação da faixa, no caso obteve-se 13 FPS, e durante detecção do obstáculo em vez de 2 FPS, atingiu-se 11 FPS.

É interessante destacar também a capacidade de detecção dos cones na maioria das imagens que foi em torno de 73 % de resultados satisfatórios, assim como é importante observar que o sistema de controle do protótipo, embora possa ser aperfeiçoado, atendeu uma estabilidade aceitável, atingindo erros médios na faixa de 4,5 centímetros em relação ao centro da faixa.

O uso de câmeras em ambientes industriais vêm crescendo de forma exponencial, partindo como sensores simples de esteiras, para uma navegação autônoma de veículos. As principais vantagens do uso de uma câmera quando comparada com outros sensores é a grande versatilidade, podendo agir na detecção de linhas, em navegação autônoma com detecção de objetos, criação de ambientes tridimensionais, detecção do menor percurso, entre outros.

Espera-se que este trabalho possa contribuir no aperfeiçoamento e sofisticação do setor industrial, quer seja com exemplos ou até mesmo na forma da abordagem adotada.

5.1 RECOMENDAÇÕES PARA TRABALHOS FUTUROS

Algumas sugestões para o aperfeiçoamento desse trabalho ou para a elaboração de outros projetos similares porém mais aperfeiçoado ou com outro enfoque são:

- Desenvolvimento de um sistema de navegação em grupos de robôs, (seria interessante observar a abordagem implementada em drones como referência),
- Implementar uma criação de ambientes tridimensionais computacionais, ou por câmera estéreo ou por câmeras simples,

- Elaboração de uma carenagem robusta para um projeto coletivo de modo que possa ser utilizado semestralmente por diferentes projetos de TCC e assim sempre levando ao aperfeiçoamento do projeto,
- Implementar a capacidade de se desviar de objetos,
- Identificação de trajetos ótimos, um bom documento como base é apresentado por THRUN et al. (2007),
- Navegação autônoma tendo como fonte principal a energia solar,
- Desenvolvimento e simplificação de protótipos como o “Donkey Car” (2017) de forma a atrair mais o público infanto juvenil para a área da eletrônica,
- Implementação de ultrassom de forma a se obter um mapa bidimensional do local,
- Fazer uso de Arranjo de Portas Programáveis em Campo (FPGA) em conjunto com microprocessadores de modo a tornar o processamento mais rápido,
- Aperfeiçoamento do sistema de locomoção, por meio do incremento da velocidade de locomoção, e melhoria do software de forma a tornar o protótipo mais robusto, mesmo em casos de vibração da câmera,
- Possibilitar o uso de múltiplos protótipos de modo a estender o alcance de comandos do operador, isto é retransmitindo-os de robô em robô,
- Implementar *thresholding* e processamento em paralelo.

REFERÊNCIAS

AMADEU, M. S. U. DOS S.; OTHERS; OTHERS; OTHERS. Manual de normalização de documentos científicos: de acordo com as normas da ABNT. , 2015.

ANDRADE, P. H. M. D. **ROBÔ MÓVEL COM VISÃO COMPUTACIONAL USANDO A BIBLIOTECA OPENCV PARA SEGUIMENTO DE LINHA**. 2016.

BAGGIO, D. L. **Mastering OpenCV with practical computer vision projects**. Packt Publishing Ltd, 2012.

BANKMAN, I. **Handbook of medical image processing and analysis**. Elsevier, 2008.

BECK, R. F. BATERIAS DE LÍTIO-ÍON (LiB) - PERSPECTIVAS E MERCADO. , 2018. Fundação CPqD.

BRADSKI, G.; KAEHLER, A. OpenCV. **Dr. Dobb's journal of software tools**, v. 3, 2000.

CANNY, J. A computational approach to edge detection. **IEEE Transactions on pattern analysis and machine intelligence**, , n. 6, p. 679–698, 1986.

CHAN, C. The state of the art of electric and hybrid vehicles. **Proceedings of the IEEE**, v. 90, n. 2, p. 247–275, 2002.

CHUNG, T. DARPA Subterranean (SubT) Challenge. Disponível em: <https://www.darpa.mil/program/darpa-subterranean-challenge>. Acesso em: 17/11/2018.

COHEN, P. R. A survey of the Eighth National Conference on Artificial Intelligence: Pulling together or pulling apart? **AI magazine**, v. 12, n. 1, p. 16, 1991.

COMMA.AI. comma.ai. Disponível em: <https://comma.ai/>. Acesso em: 17/11/2018.

CULJAK, I.; ABRAM, D.; PRIBANIC, T.; DZAPO, H.; CIFREK, M. A brief introduction to OpenCV. MIPRO, 2012 proceedings of the 35th international convention. **Anais...** . p.1725–1730, 2012. IEEE.

DARPA. DARPA Perspective on AI. Disponível em: <https://www.darpa.mil/about-us/darpa-perspective-on-ai>. Acesso em: 18/11/2018.

DELL, R. M.; RAND, D. A. J. **Understanding batteries**. Royal Society of Chemistry, 2001.

Donkey Car. .Disponível em: <http://www.donkeycar.com/>. Acesso em: 17/11/2018.

FERREIRA, J. M. M. **Introdução ao projecto com sistemas digitais e microcontroladores**. FEUP Edições, 1998.

FISHER, R.; PERKINS, S.; WALKER, A.; WOLFART, E. Hypermedia image processing reference. **England: John Wiley & Sons Ltd**, 1996.

GONZALES, R.; WOODS, E. **Digital Image Processing, 3, d edition**. Prentice-Hall, 2007.

GÖTTING. HG G-73840 | Götting KG. Disponível em: <https://www.goetting-agv.com/components/73840>. Acesso em: 3/8/2018.

HOWSE, J. **OpenCV computer vision with python**. Packt Publishing Ltd, 2013.

JONES, D. 8. API - The PiCamera Class — Picamera 1.12 documentation. Disponível em: <https://picamera.readthedocs.io/en/release-1.12/api_camera.html>. Acesso em: 21/11/2018.

KLETTE, R. **Concise computer vision**. Springer, 2014.

MACKAY, D. **Sustainable Energy-without the hot air**. UIT Cambridge, 2008.

MILLER, R. K.; SUBRIN, R. **Automated guided vehicles and automated manufacturing**. Society of manufacturing engineers, 1987.

NHTSA. Automated Vehicles for Safety. Disponível em: <https://www.nhtsa.gov/technology-innovation/automated-vehicles-safety>. Acesso em: 18/11/2018.

OLIVEIRA, J. A. DE. **DESENVOLVIMENTO DE UM PROTÓTIPO DE VEÍCULO AUTO GUIADO PARA CÉLULA PRODUTIVA**. INSTITUTO DE TECNOLOGIA PARA O DESENVOLVIMENTO, 2011.

OPENCV. OpenCV: Template Matching. Disponível em: <https://docs.opencv.org/3.4/d4/dc6/tutorial_py_template_matching.html>. Acesso em: 4/8/2018.

PAJANKAR, A.; KAKKAR, A. **Raspberry Pi By Example**. Packt Publishing Ltd, 2016.

PRECHELT, L. An empirical comparison of seven programming languages. **Computer**, , n. 10, p. 23–29, 2000.

RASPBERRYPI. GPIO - Raspberry Pi Documentation. Disponível em: <https://www.raspberrypi.org/documentation/usage/gpio/>. Acesso em: 26/11/2018.

RIBEIRO, D. V. **ROSELI: ROBÔ SEGUIDOR DE LINHA PARA MAPEAMENTO DE AMBIENTES INTERNOS.** 2017.

RICHARDSON, T.; STAFFORD-FRASER, Q.; WOOD, K. R.; HOPPER, A. Virtual network computing. **IEEE Internet Computing**, v. 2, n. 1, p. 33–38, 1998.

robot | Definition of robot in English by Oxford Dictionaries. .Disponível em: <https://en.oxforddictionaries.com/definition/robot>. Acesso em: 17/11/2018.

ROBOTEQ. Magnetic Guide Sensors | Roboteq. Disponível em: <https://www.roboteq.com/index.php/roboteq-products-and-services/magnetic-guide-sensors>. Acesso em: 3/8/2018a.

ROBOTEQ. Datasheet - MGS1600GY Magnetic Sensor Datasheet. , 2018b. Roboteq.

ROSENFELD, A. Picture processing by computer. **ACM Computing Surveys (CSUR)**, v. 1, n. 3, p. 147–176, 1969.

SICK. Sensores 2D-LiDAR | SICK. Disponível em: <https://www.sick.com/br/pt/solucoes-de-medicao-e-deteccao/sensores-2d-lidar/c/g91900>. Acesso em: 3/8/2018.

SIEGWART, R.; NOURBAKHSH, I. R.; SCARAMUZZA, D. Autonomous mobile robots. **Massachusetts Institute of Technology**, 2004.

ST. Datasheet - L298 DUAL FULL-BRIDGE DRIVER. , 2018.

SUZUKI, S.; OTHERS; OTHER; OTHER. Topological structural analysis of digitized binary images by border following. **Computer vision, graphics, and image processing**, v. 30, n. 1, p. 32–46, 1985.

SZELISKI, R. **Computer vision: algorithms and applications**. Springer Science & Business Media, 2010.

TESLA. Electric Cars, Solar Panels & Clean Energy Storage | Tesla. Disponível em: <https://www.tesla.com/>. Acesso em: 18/11/2018.

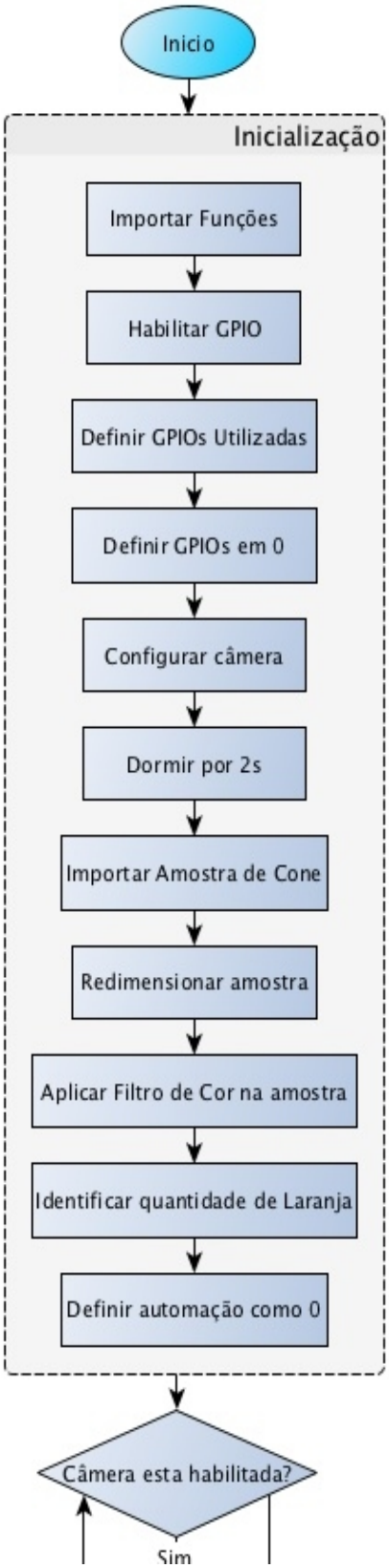
THRUN, S.; MONTEMERLO, M.; DAHLKAMP, H.; et al. Stanley: The robot that won the darpa grand challenge. **The 2005 DARPA grand challenge**. p.1–43, 2007. Springer.

XLSEMI. Datasheet - XL6009 400KHz 42V 5A Switching Current Boost DC/DC Converter. , 201DC. XLSEMI.

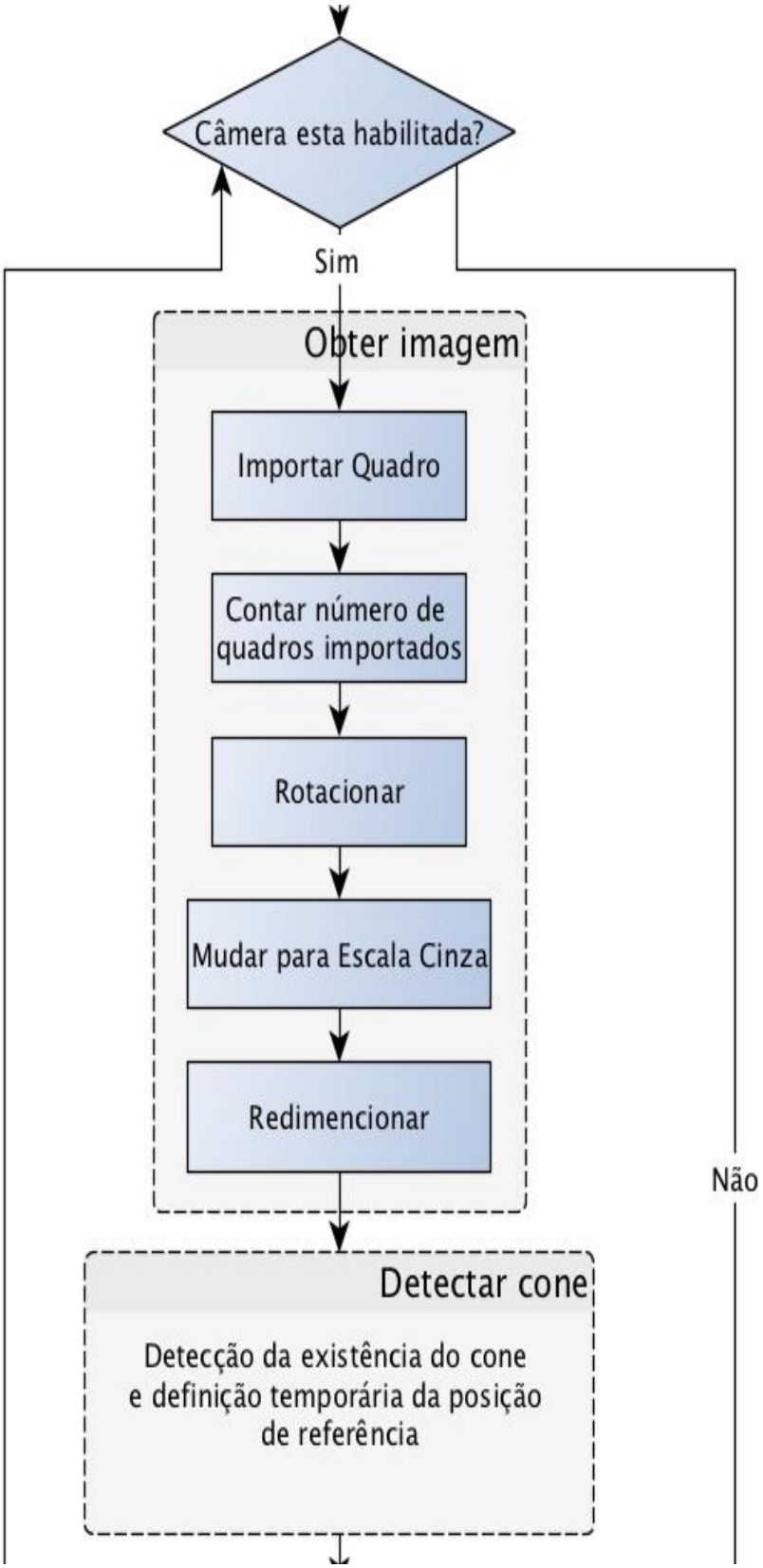
YLONEN, T.; LONVICK, C. **The secure shell (SSH) protocol architecture**. 2005.

ZHANG, C.; WEI, Y.-L.; CAO, P.-F.; LIN, M.-C. Energy storage system: Current studies on batteries and power condition system. **Renewable and Sustainable Energy Reviews**, 2017.

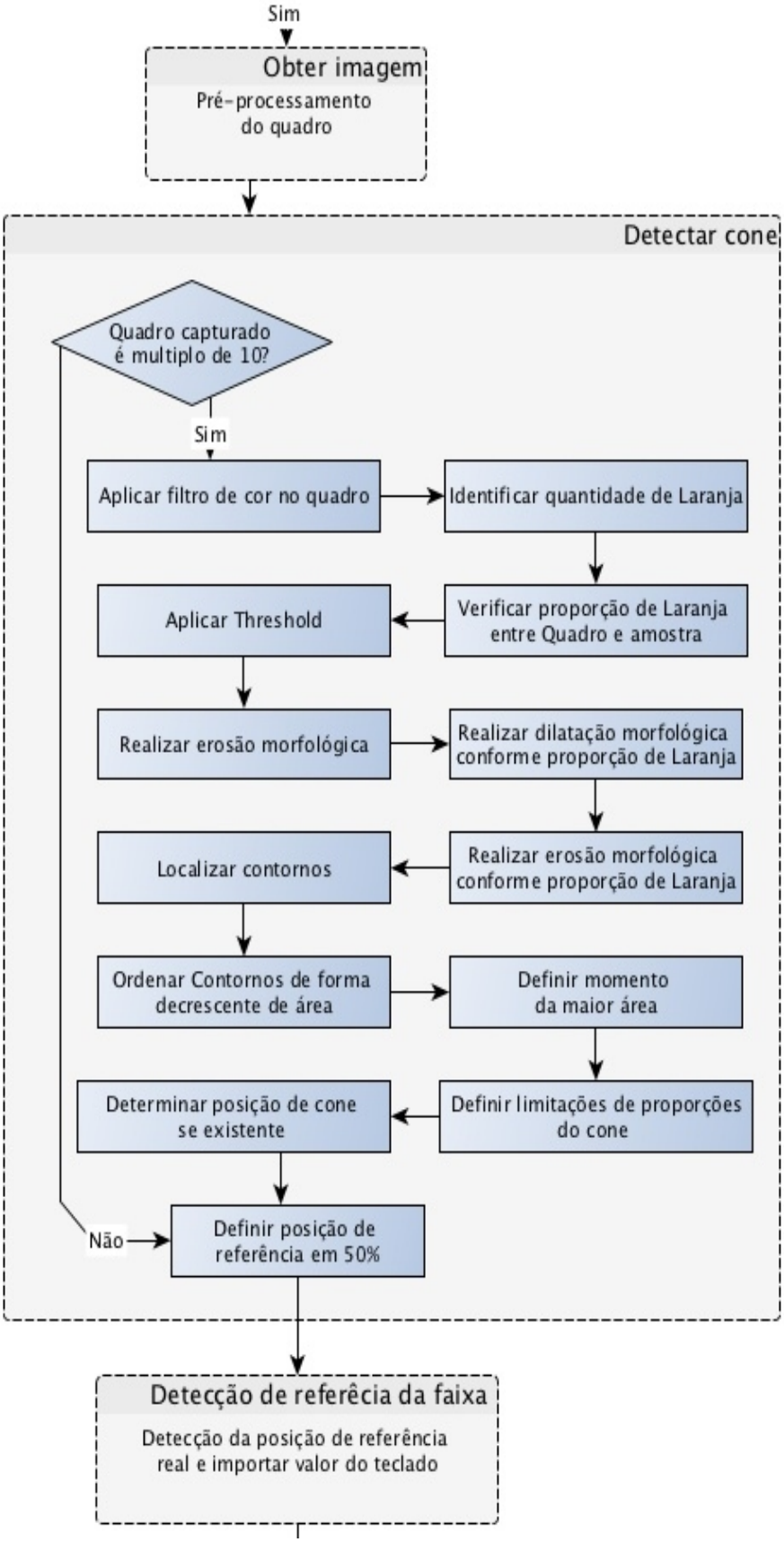
APÊNDICE 1 – FLUXOGRAMA INICIALIZAÇÃO



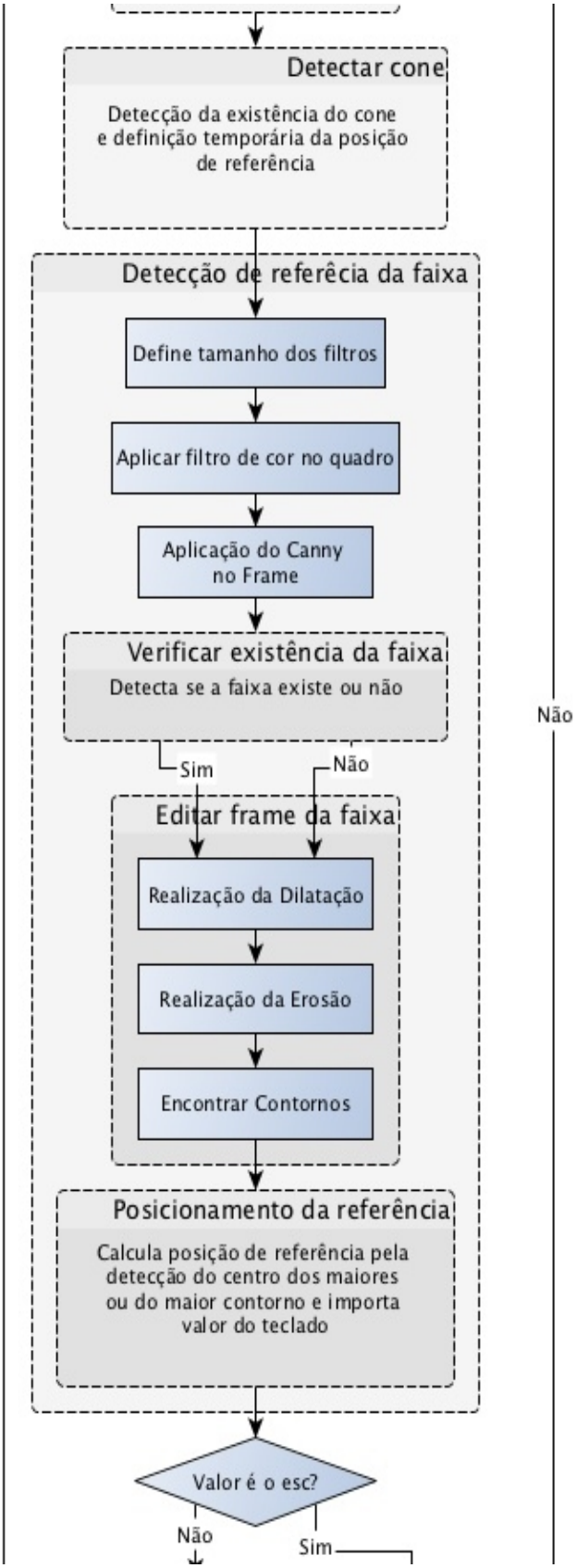
APÊNDICE 2 – FLUXOGRAMA OBTER IMAGEM



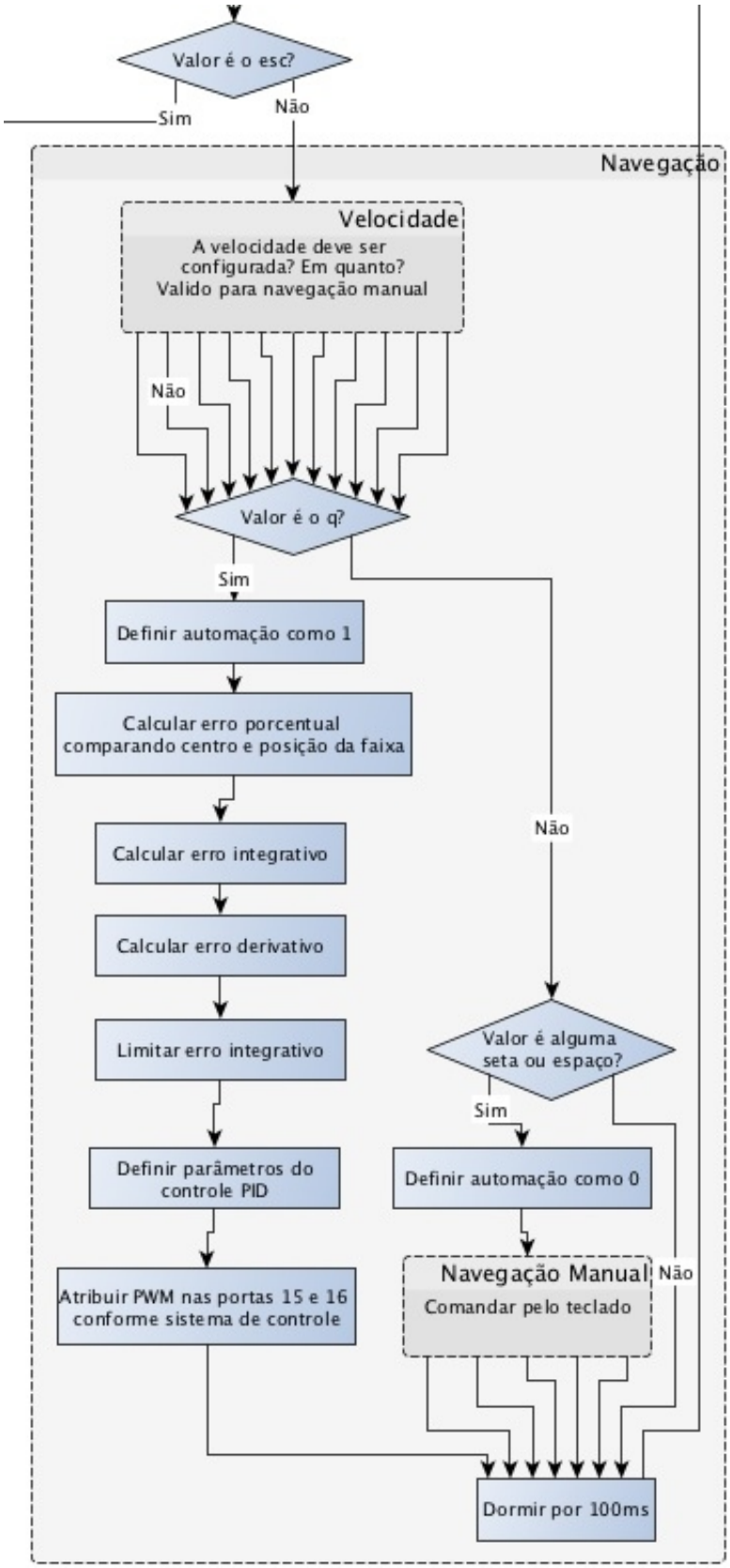
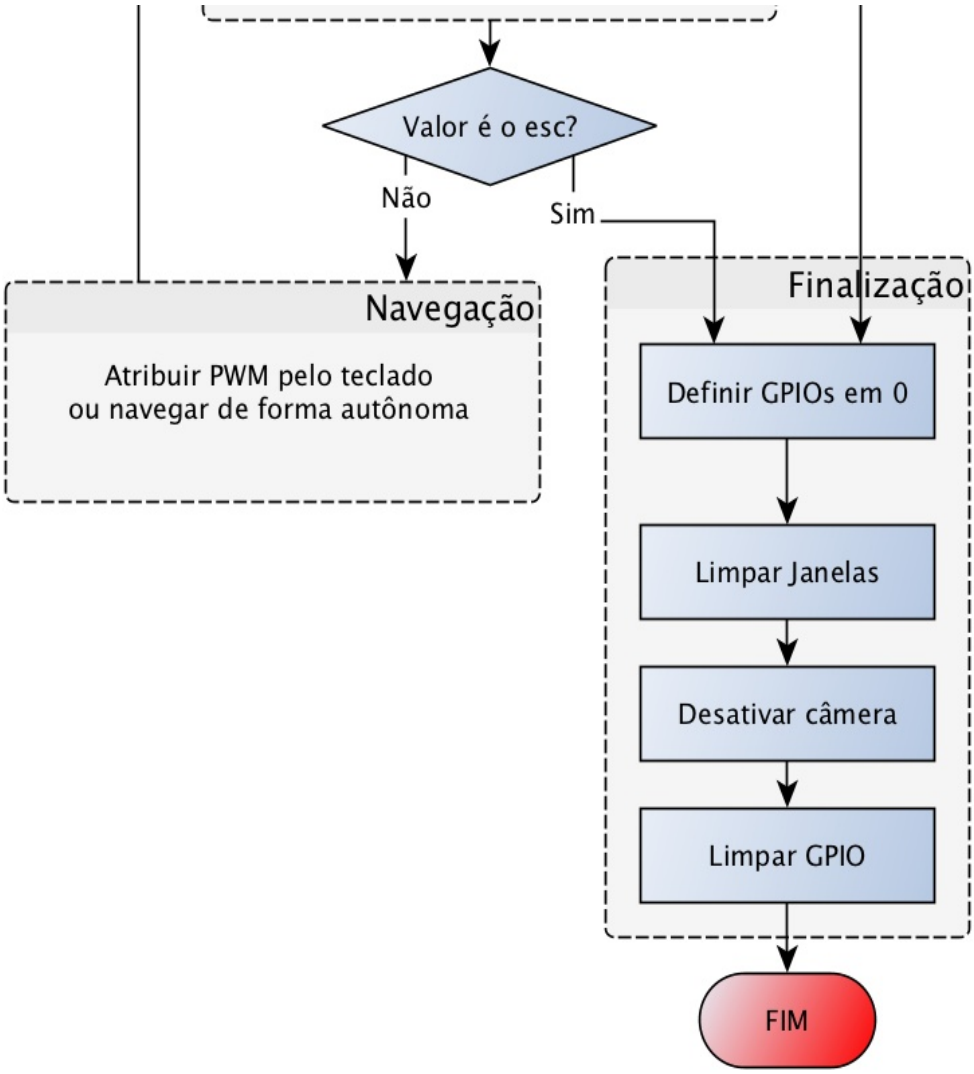
APÊNDICE 3 – FLUXOGRAMA DETECTAR CONE



APÊNDICE 4 – FLUXOGRAMA DETECTAR REFERÊNCIA DA FAIXA



APÊNDICE 5 – FLUXOGRAMA NAVEGAÇÃO



APÊNDICE 6 – FLUXOGRAMA FINALIZAÇÃO