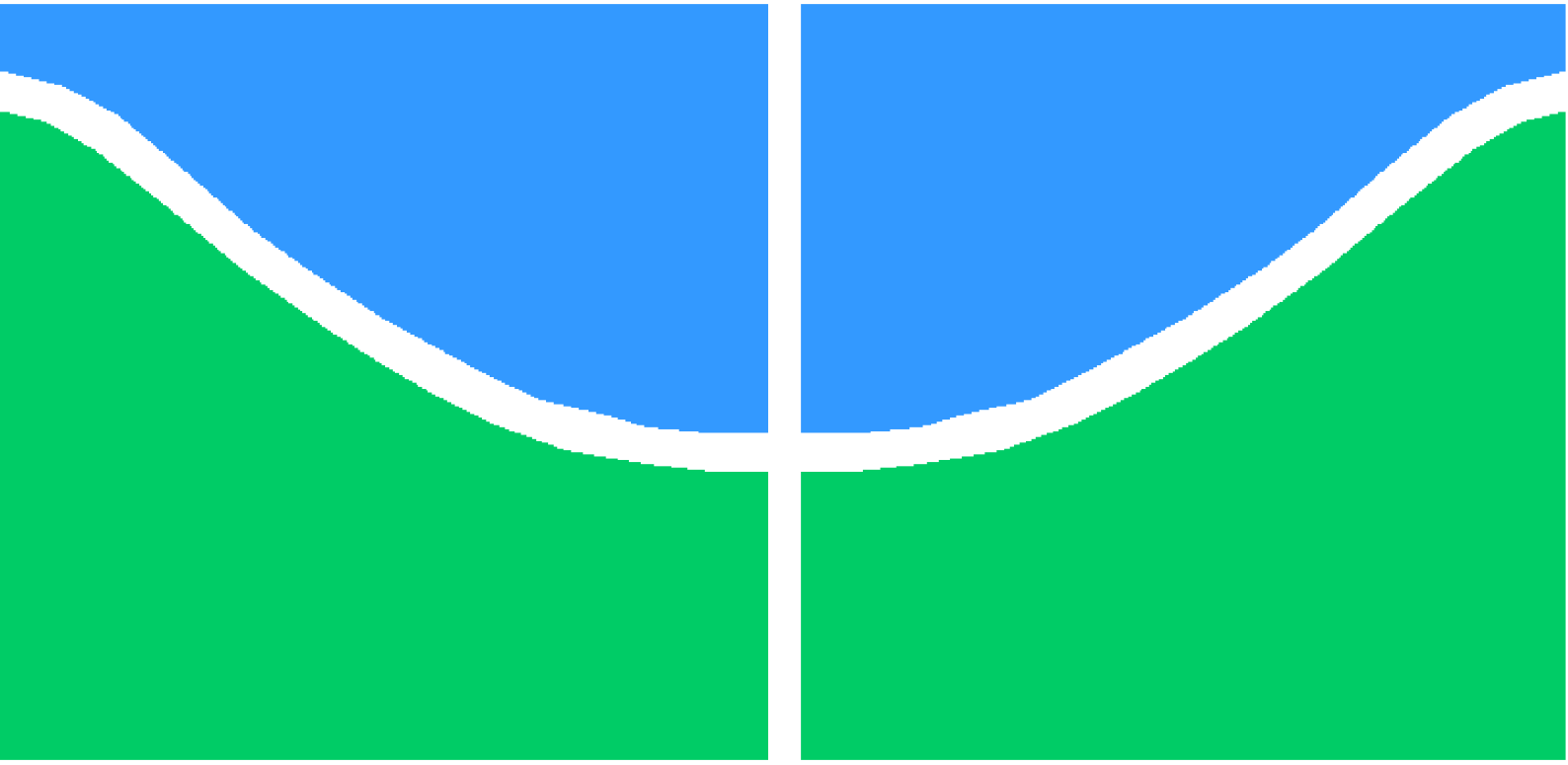


Universidade de Brasília - UnB

Faculdade UnB Gama - FGA

Engenharia de Software

SISTEMA DE DETECÇÃO DE FAIXAS INTEGRADO EM VEÍCULOS DE PEQUENA ESCALA

Autor: João Victor Valadão de Brito

Orientador: Msc. Rafael Rodrigues da Silva

Co-Orientador: Dr. Evandro Leonardo Silva Teixeira

Brasília, DF

2023

João Victor Valadão de Brito

SISTEMA DE DETECÇÃO DE FAIXAS INTEGRADO EM VEÍCULOS DE PEQUENA ESCALA

Monografia submetida ao curso de graduação em Engenharia de Software da Universidade de Brasília, como requisito parcial para obtenção do Título de Bacharel em Engenharia de Software.

Universidade de Brasília - UnB

Faculdade UnB Gama - FGA

Orientador: Msc. Rafael Rodrigues da Silva

Coorientador: Dr. Evandro Leonardo Silva Teixeira

Brasília, DF

2023

João Victor Valadão de Brito
SISTEMA DE DETECÇÃO DE FAIXAS INTEGRADO EM VEÍCULOS DE PEQUENA ESCALA/ João Victor Valadão de Brito. – Brasília, DF, 2023-
81 p. : il. (algumas color.) ; 30 cm.

Orientador: Msc. Rafael Rodrigues da Silva

Co-Orientador: Dr. Evandro Leonardo Silva Teixeira

Trabalho de Conclusão de Curso – Universidade de Brasília - UnB
Faculdade UnB Gama - FGA , 2023.

1. Palavra-chave01. 2. Palavra-chave02. I. Msc. Rafael Rodrigues da Silva. II. Co-Orientador: Dr. Evandro Leonardo Silva Teixeira III. Universidade de Brasília. IV. Faculdade UnB Gama. V. SISTEMA DE DETECÇÃO DE FAIXAS INTEGRADO EM VEÍCULOS DE PEQUENA ESCALA

CDU 02:141:005.6

João Victor Valadão de Brito

SISTEMA DE DETECÇÃO DE FAIXAS INTEGRADO EM VEÍCULOS DE PEQUENA ESCALA

Monografia submetida ao curso de graduação em Engenharia de Software da Universidade de Brasília, como requisito parcial para obtenção do Título de Bacharel em Engenharia de Software.

Brasília, DF, :

Msc. Rafael Rodrigues da Silva
Orientador

Dr. Evandro Leonardo Silva Teixeira
Co-Orientador

Prof. Giovanni Almeida Santos
Convidado 1

PhD. Luciano Emidio Neves da Fonseca
Convidado 2

Brasília, DF
2023

Agradecimentos

Agradeço a Deus, que se mostrou presente ao meu lado durante todo esse projeto. À minha família, que me apoiou nos meus momentos difíceis, e mantiveram sua fé no meu sucesso mesmo quando eu já não acreditava. Aos amigos que conheci nessa jornada, que me acolheram ao projeto e constantemente reservavam um tempo do seu dia para esclarecer minhas dúvidas. E por fim aos meus professores, que me orientaram e auxiliaram durante esta importante etapa da minha vida.

"Meus irmãos, considerem motivo de grande alegria o fato de passarem por diversas provações, pois vocês sabem que a prova da sua fé produz perseverança. E a perseverança deve ter ação completa, a fim de que vocês sejam maduros e íntegros, sem que falte a vocês coisa alguma."

Resumo

Devido as crescentes pesquisas acerca de segurança veicular nas últimas décadas, os automóveis estão sendo equipados com diferentes sistemas que visam auxiliar o condutor. Dentre eles, o Lane Keeping, que visa manter o veículo dentro dos limites da faixa. Neste estudo, são abordados o processo de implementação dos algoritmos de detecção de faixas de rolagem, a partir de técnicas de processamento de imagens. Serão consideradas as principais abordagens de processamento utilizadas em sistemas de Lane Keeping, a partir da captação de imagens em tempo real de uma câmera bifocal. Além de propostas de arquitetura para realizar a comunicação entre os algoritmos e o sistema de direção de um veículo de pequena escala.

Palavras-chaves: Lane Keeping, Processamento de Imagens, Lane Detection, ADAS.

Abstract

Due to the increasing of research on vehicle safety in recent decades, automobiles are being equipped with different systems to assist the driver. Among these functions, there is the Lane Keeping, which aims to keep the vehicle within the lane boundaries. In this study will be presented the implementation process of lane detection algorithms using image processing techniques, capable of communicating with the steering system of a small-scale vehicle. The main processing techniques used in Lane Keeping systems are considered, based on real-time image capture from a bifocal camera. Additionally, architecture proposals are presented to integrate the algorithms with the steering system.

Key-words: Lane Keeping, Image Processing, Lane Detection, ADAS

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

ACC	Adaptive Cruise Control
ACK	Acknowledge
ADAS	Advanced Driver Assistance Systems
AEBS	Advanced Emergency Braking System
AIoI	Artificial Intelligence of Things
ANN-MLP	Artificial Neural Networks - Multilayer Perceptron
CAN	Controller Area Network
CCD	Charge Coupled Devices
CNN	Convolutional Neural Network
CNN-LSTM	Convolutional Neural Network - Long Short-Term Memory
CPU	Central Processing Unit
CUDA	Compute Unified Device Architecture
ECU	Electronic Control Unit
EELA	Extended Edge Linking Algorithm
GPU	Graphics Processing Unit
HT	Hough Transformation
ICTs	Instituições Científicas, Tecnológicas e de Inovação
LDWS	Lane Departure Warning System
LKAS	Lane Keeping Assist System
LCA	Lane Change Assist
MRFs	Markov Random Fields
ONSV	Observatório Nacional de Segurança Viária
PDI	Processamento Digital de Imagem

PIB	Produto Interno Bruto
PI	Controle Proporcional Integral
PID	Controle Proporcional Integral Derivativo
RANSAC	Random Sample Consensus
RGB	Red, Green, Blue
ROI	Region of interest
SDK	Software Development Kit
SVM	Support Vector Machine
UDP	User Datagram Protocol
UnB	Universidade de Brasília
WHO	World Health Organization
WSL	Windows Subsystem for Linux

Lista de símbolos

θr	Ângulo de esterçamento das rodas
L	Entre eixos
t	Bitola
R	Raio de giro
δ_o	Ângulo da roda externa
δ_i	Ângulo da roda interna
X	objeto real
Z	eixo ótico
f	distância focal
x	Imagem
LR	Largura de um veículo real aproximada
dF	Distância de uma faixa de rolamento a 80km/h
LJ	Largura do veículo de pequena escala
dx	Distância entre as faixas para o veículo de pequena escala
Er	Erro, distância que o veículo se encontra do centro da faixa
dc	posição do centro da faixa
pv	posição do veículo

Sumário

1 Introdução

Nos últimos anos, a indústria automotiva brasileira vem se tornando um setor cada vez mais importante para a economia do país, contribuindo com uma parcela significativa do PIB, além de empregar cada vez mais pessoas, direta ou indiretamente. Um aspecto relevante a ser considerado acerca do impacto causado no setor automotivo nacional é o interesse da indústria em estabelecer a produção e desenvolvimento de veículos no território brasileiro. De modo que as empresas automotivas buscam impulsionar o desenvolvimento da mão de obra qualificada necessária para atender às demandas do mercado.

Em 2018, o governo federal criou o programa Rota 2030 a partir da lei nº 13.755, com o objetivo de estabelecer as diretrizes da indústria automotiva brasileira até o ano de 2030. Este programa busca viabilizar os investimentos necessários para potencializar as empresas automotivas nacionais, por meio de uma política a longo prazo, com o intuito de modernizar os veículos atuais a fim de proporcionar aos consumidores automóveis mais seguros e eficientes.

Com isso, o programa Rota 2030 oferece incentivos fiscais às montadoras que realizam pesquisas voltadas à segurança veicular, energias limpas e eficiência energética, que visam reduzir os impactos ambientais e mitigar impactos automotivos. Essa iniciativa busca fortalecer a competitividade da indústria brasileira no mercado internacional, reconhecendo a necessidade de aprimorar a engenharia e o desenvolvimento tecnológico local.

Uma das vertentes do programa é estabelecida por meio de parcerias com representantes da indústria automotiva e entidades governamentais especializadas em pesquisa e desenvolvimento de tecnologias voltadas para a eficiência energética e a segurança veicular. Entre as diferentes frentes adotadas pelo Rota 2030, uma delas está voltada a aproximar os desafios da indústria atual com as Instituições de Ciência e Tecnologia (ICTs). Dessa forma, originou-se uma parceria entre as empresas do setor automotivo e as ICTs, voltada a pesquisas e desenvolvimento de funções de segurança assistida ao condutor. Um dos projetos vinculados ao Rota 2030 é o SegurAuto, o qual possui como principal objetivo a implementação de funções avançadas de assistência ao condutor (*ADAS*) para desenvolvimento de sistemas de auxílio na direção do condutor levando em consideração a malha de trânsito brasileira.

Os acidentes de trânsito ainda acontecem em quantidades elevadas ao redor do mundo, e são um dos principais problemas da sociedade moderna (ORGANIZATION, 2022). Uma estimativa levantada pelo *World Health Organization* (WHO) apontou que cerca de 1,3 milhões de pessoas morrem vitimas de acidentes de trânsito anualmente.

Este estudo aponta como principal fator o erro humano proveniente de decisões por parte do condutor como o excesso de velocidade da via, ultrapassagens proibidas, ingestão de álcool, além dos fatores físicos como as distrações na pista e cansaço. No Brasil, o Observatório Nacional de Segurança Viária (ONSV) divulgou que o fator humano é responsável por cerca de 90% dos acidentes de trânsito. Também foi observado que aproximadamente 5% dos casos levantados são causados por falhas de mal funcionamento do veículo, onde o fator humano ainda está presente, devido a falta de manutenções e cuidados dos veículos. Deste modo, com o objetivo de mitigar os efeitos dos acidentes, os fabricantes e fornecedores têm investido no desenvolvimento dos Sistemas Avançados de Assistência ao Condutor (ADAS), uma vez que segurança e conforto se tornaram um dos principais aspectos considerados pelos consumidores ao adquirirem um veículo.

Os sistemas de assistência ao condutor podem ter acesso a um conjunto adicional de informações sobre o ambiente em que o veículo se encontra e são categorizados em dois grupos distintos:

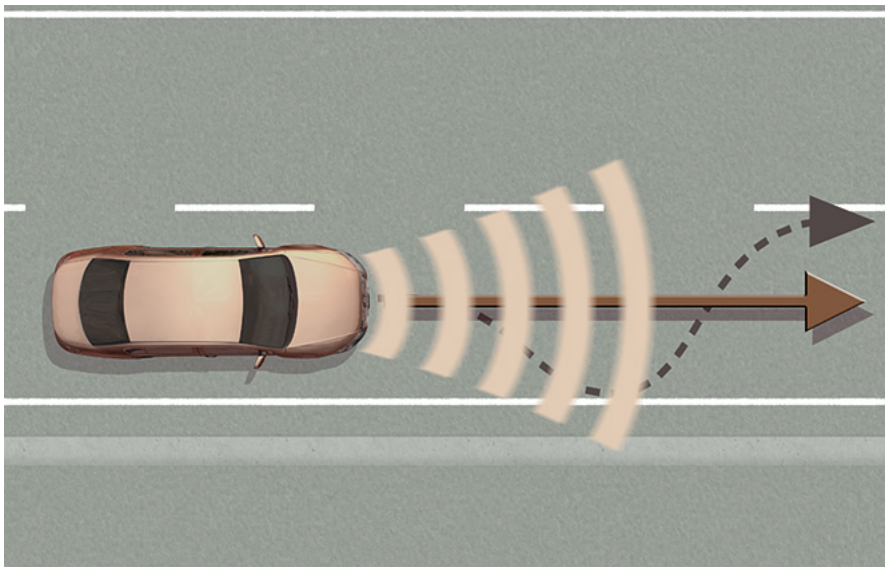
- Longitudinal: Responsáveis pelo controle do veículo no sentido longitudinal, encarregado pelo controle da velocidade do veículo mediante a distância que este se encontra de outros veículos ou objetos, para evitar colisões frontais e mitigar impactos. Alguns desses sistemas são:
  - *Adaptive Cruise Control* (ACC) projetado para manter uma distância segura do veículo a frente ajustando a velocidade atual de acordo com a leitura dos sensores, sem que haja intervenção do motorista;
  - *Advanced Emergency Braking System* (AEBS) é um sistema avançado de frenagem de emergência que opera através da monitoração do tráfego à frente do veículo, garantindo um preciso acionamento dos freios.
- Lateral: Responsáveis pelo controle do veículo no sentido lateral, estes sistemas são responsáveis por auxiliar o condutor em manobras que envolvam a posição do veículo na pista, como o controle da direção em curvas e mudanças de faixas. Alguns desses sistemas são:
  - *Lane Departure Warning System* (LDWS), sistema de detecção de faixas que notifica ao condutor quando está próximo aos limites da sua faixa de rodagem;
  - *Lane Keeping Assist System* (LKAS), utilizado para manter um veículo nos limites da pista, sendo este o foco deste trabalho;
  - Lane Centering, responsáveis por manter o veículo no centro da pista, a partir do controle constante do sistema de direção.

O projeto SegurAuto atua em diferentes frentes, com o objetivo de desenvolver, no âmbito acadêmico, as principais funcionalidades ADAS existentes no mercado. Uma

vez que o projeto está estruturado, torna-se possível dividir os objetos de estudo, no caso, os sistemas de auxílio ao condutor, entre as ICTs. Dessa forma, a Universidade de Brasília (UnB) está responsável pela implementação das funções de *Lane Keeping Assist System* e *Advanced Emergency Braking System* em um veículo de pequena escala. A ênfase deste trabalho está na implementação do LKAS em ambiente laboratorial, considerando o significativo avanço ocorrido nas pesquisas relacionadas às tecnologias de Lane Detection e Road Analysis nas últimas décadas. (HTET; XINXIN, 2015).

Com a crescente no desenvolvimento de tecnologias que atuam na detecção dos ambientes de trânsito, com bons resultados e alta performance, é possível justificar o sucesso dos algoritmos de LKAS. Esses algoritmos atuam a partir da combinação dos dados capturados dos sistemas de *Lane Detection*, responsáveis por identificarem e rastrearem as faixas de rodagem. Esses dados são integrados uma central de tomada de decisão, um sistema de controle responsável por processar as informações e avaliar a melhor decisão durante a condução. Dessa forma, operam em sintonia com o sistema de direção do veículo. O *Lane Detection* está sendo amplamente pesquisado no âmbito dos sistemas ADAS juntamente com *Lane Departure Warning System* (LDWS), que funciona a partir da emissão de alertas ao condutor quando os parâmetros definidos no sistema para manter o veículo entre as faixas não são seguidos.

Figura 1 – LKAS System



Fonte: (OAGANA, 2016)

Dessa forma, este trabalho segue a temática de explorar as propriedades de visão computacional voltada a sistemas de detecção de faixas (*Lane Detection*) e posteriormente avançar para o estágio de manutenção na faixa (*Lane Keeping*). Para isso foi utilizado um veículo em pequena escala, um Jeep Wrangler (Figura 2) para que seja possível verificar

e validar os testes práticos no desenvolvimento dos sistemas. Atualmente, o veículo se locomove através de comandos enviados a partir de um controle remoto, e seus sistemas de aceleração, frenagem e aceleração já foram implementados em Unidades de controle eletrônicos (*ECUs*). Dentre as últimas alterações do veículo, também deve ser incluída a instalação de uma Câmera Zed 2i e uma Zed Box, ambas produzidas pela StereoLabs, e um radar de proximidade que irão nos auxiliar no processo de automação do Jeep, e em seguida, replicar esses resultados em um veículo real.

Figura 2 – Jeep Wrangler

Fonte: O autor

1.1 Objetivos

O objetivo geral desse trabalho é desenvolver o *Lane Keeping*, abordando os conceitos de visão computacional, processamento de imagens e arquitetura de software para garantir a correta implementação do sistema LKA.

Os objetivos específicos do trabalho são:

- Integrar a unidade central de processamento (CPU) ZED Box e a câmera ZED 2i ao veículo de pequena escala;
- Utilizar os recursos computacionais da ZED Box para realizar o processamento de imagens;
- Desenvolver o algoritmo do *Lane Keeping* a partir dos *input* capturados pela câmera ZED 2i;
- Estruturar uma arquitetura geral para a comunicação entre a Zed Box e o sistema de direção;
- Testes e Validações em diferentes cenários

1.2 Organização do trabalho

O trabalho será dividido em capítulos, cada um contendo um tema principal, com objetivos específicos, que serão relembrados no começo de cada seção. Espera-se a seguinte disposição de capítulos para isso:

- Capítulo 2: Neste capítulo será apresentada a fundamentação teórica acerca dos temas deste trabalho. Os temas estudados aqui serão: sistema de aceleração e frenagem, sistemas de direção, sistemas de controle e teoria de controladores, modelagem matemática de motores de corrente contínua e introdução à rede CAN;
- Capítulo 3: Neste capítulo, será apresentada a metodologia utilizada, evidenciando os métodos e as ferramentas;
- Capítulo 4: Neste capítulo, serão apresentados e discutidos os resultados obtidos;
- Capítulo 5: Neste capítulo, será apresentada a conclusão sobre este trabalho.

2 Fundamentação Teórica

Baseado nas pesquisas e estudos relacionados aos temas que abrangem as funções de auxilio aos condutores em veículos de trânsito, este capítulo busca demonstrar trabalhos relacionados ao projeto que alcançaram os resultados esperados no processamento de imagem computacional para a implementação e desenvolvimento de algoritmos LKAS.

2.1 Estado da Arte

A pesquisa Vijitkunsawat e Chantngarm (2020), tem como objetivo comparar os algoritmos de detecção de objetos e faixas de rolagem que estavam sendo utilizados em carros autônomos no ano de 2020. Para isso, o projeto utiliza uma placa Nvidia Jetson Nano como microprocessador das imagens captadas pela câmera IMX-219 em um veículo de pequena escala, para avaliarem o desempenho de cada algoritmo. São selecionados três sistemas, *Support Vector Machine* (SVM), *Artificial Neural Networks - Multilayer Perceptron* (ANN-MLP) e *Convolutional Neural Network - Long Short-Term Memory* (CNN-LSTM) que são implementados ao veículo em diferentes velocidades para analisarem qual apresentará o melhor desempenho em diferentes ambientes. Com isso, os autores concluíram que o algoritmo que apresenta a melhor acurácia é o CNN-LSTM, que se destaca na maioria dos cenários, porém seus resultados mostram que em velocidades reduzidas todos os algoritmos foram capazes de manterem-se dentro dos limites propostos, com suas variações de posicionamento e performance.

Ainda no âmbito de veículos autônomos, Chen, Li e Huang (2020) propõe criar um simulador carregado com o *dataset* de imagens e vídeos gravados a partir de veículos em circulação cotidiano, para que os dados processados pelos algoritmos de LKAS sejam interpretados para interagir com o sistema de direção. Porém, diferente dos demais projetos nessa área, o artigo sugere abandonar a estrutura do controlador tradicional, o qual é responsável por calcular o ângulo de esterção do volante a partir dos dados fornecidos pelo algoritmo LKAS, e propõe otimizar o processo utilizando o *end-to-end learning*. Esse método consiste em treinar um modelo de aprendizagem de máquina, no qual o sistema irá aprender todas as etapas do processo, nesse caso, a partir das imagens capturadas pela câmera acessar e controlar o sistema de direção do veículo. Dessa forma, conseguiram alcançar resultados importantes, uma vez que os testes com o simulador apresentaram ser muito mais efetivos do que apenas utilizando os dados do *dataset*, reduzindo a incidência de falhas de 98 para 9, possibilitando a direção do veículo permanecer autônoma por 98% do tempo.

De forma semelhante, Kuo, Lu e Yang (2019) propõe desenvolver um modelo de

carro autônomo em escala reduzida (1/10), utilizando um algoritmo para detecção de faixas e um controlador de direção. O artigo propõe que a partir de bons algoritmos de LKAS é possível desenvolver sistemas de custos acessíveis, uma vez que o veículo esta equipado apenas com uma *dashcan*, uma câmera de vídeo simples instalada no painel do veículo. Sem utilizar de qualquer outro sensor, como por exemplo RADAR ou LIDAR, o projeto se mostrou capaz de manter o veículo na faixa durante os testes realizados em tempo real. Apesar das dificuldades a pesquisa garante sua premissa inicial, uma vez que apresenta resultados eficazes, onde o algoritmo apresentou resultados melhores que os estudos comparados anteriores quando o veículo é submetidos a testes em retas. Porém, quando são inseridas curvas, o sistema apresenta dificuldades, tanto na parte da detecção quanto no controle de direção, chegando a uma taxa de 5% de erros em curvas.

O estudo de Heidarizadeh (2021) investigou o impacto do desalinhamento da câmera na qualidade do sistema de assistência de permanência em faixa (LKAS) e na experiência e desempenho do usuário em um veículo. Foram realizados experimentos no simulador de direção da Universidade de Leeds com 16 participantes, avaliando diferentes níveis de erro de alinhamento da câmera em cenários de estradas curvas e rodovias. Os resultados mostraram que os erros no LKAS causaram efeitos significativos nas medidas de desempenho de direção, como posição média da faixa, velocidade média e desvio padrão da posição da faixa, em relação aos níveis de erro do sistema de assistência de faixa de rodagem, diminuindo a segurança do sistema. Enquanto as mudanças no erro do LKAS afetaram o desempenho do motorista, elas não prejudicaram a avaliação do participante em relação ao esforço ou ao conforto/confiança ao dirigir.

O artigo Romano et al. (2021) aborda os métodos de pré-processamento para detecção e rastreamento de faixas de trânsito em tempo real. O pré-processamento é crucial para remover ruídos e melhorar a detecção de faixas. Os principais passos de pré-processamento incluem suavização da imagem, seleção da região de interesse, mapeamento de perspectiva inversa e segmentação. Diferentes técnicas, como filtros de média, filtros gaussianos e detecção de bordas são aplicadas durante o pré-processamento. O artigo também discute a extração do ponto de fuga e a transformação da imagem em uma visão aérea. O objetivo é melhorar a precisão e a eficiência dos sistemas de detecção de faixas.

O aumento do número de acidentes rodoviários é um problema sério na sociedade moderna. Date e Gaikwad (2017) apresenta uma comparação entre três técnicas amplamente utilizadas de detecção de faixas de trânsito com base em visão computacional: Transformada de Hough, modelo de estrada e lógica fuzzy. A Transformada de Hough é a técnica mais eficaz para detecção de linhas retas, enquanto os métodos baseados em modelo são mais robustos contra obstruções e dados ausentes. A lógica fuzzy é útil para obter conclusões precisas a partir de dados imprecisos. O artigo descreve as etapas de pré-processamento, seleção da região de interesse, detecção de faixas e medição de afasta-

mento da faixa. As técnicas de detecção de faixas são aplicadas para melhorar a segurança nas estradas e reduzir acidentes.

Dentre as diferentes práticas para manutenção de veículos em faixas abordadas no artigo, os autores concluem que algoritmos que utilizam a transformação de Hough, apresentam melhores resultados, devido ao fato de utilizar menos memória de processamento com a mesma confiabilidade de outros algoritmos. Para que o projeto alcance a total implementação do *Lane Keeping Assist*, é essencial estabelecer a comunicação efetiva entre os algoritmos de *Lane Detection* e o sistema de direção.

2.2 Sistema de *Lane Keeping Assist*

Com o crescimento do número de veículos, ocorreu uma expansão significativa das vias urbanas e estradas para atender às demandas de deslocamentos tanto interurbanos quanto intraurbanos, assim como para promover a conectividade em áreas rurais. Contudo, esse progresso trouxe consigo desafios consideráveis, incluindo questões relacionadas à segurança, poluição e aumento na demanda por energia.

Ao longo dos anos, os veículos evoluíram, tornando-se máquinas tecnológicas sofisticadas que oferecem não apenas mobilidade, mas também lazer, conforto, luxo, esportividade e uma forma de expressar a imagem e personalidade dos indivíduos. Diante disso, foram desenvolvidas diversas leis, regulamentações e padrões para o design, produção e interação dos veículos com o ambiente e o motorista.

Além disso, a interação entre o motorista e o veículo também é uma preocupação importante no design dos automóveis. Atualmente, os veículos contam com sensores e sistemas eletrônicos que contribuem para o controle automático de subsistemas, desde o controle de estabilidade até o suporte ao motorista em planejamento de viagens e seleção de rotas. Os veículos inteligentes visam utilizar totalmente as tecnologias disponíveis para auxiliar os motoristas, aprimorando o controle, a segurança, a eficiência e o conforto durante a condução.

2.2.1 Segurança em veículos conectados

Uma proposta para mitigar o número de fatalidades e acidentes no trânsito envolve a integração de tecnologias de assistência ao condutor nos veículos convencionais. Nesse contexto, surgiram os veículos conectados ou os carros "inteligentes", máquinas que combinam controle humano e computacional. Esses automóveis desempenham funções de direção de maneira autônoma ou fornecem assistência ao condutor para uma condução mais eficiente. Garantindo benefícios em questões de segurança, eficiência e impacto ambiental positivo. A autonomia implica que a máquina pode executar tarefas sem orientação humana, exigindo a percepção da situação, análise e tomada de decisões para ações.

Diferentes sistemas foram desenvolvidos para garantir a segurança do condutor, equipados com uma série de componentes que permitem mapear a área ao redor. Os níveis de complexidade em cada sistema varia desde funções extramente simples como a seleção de rotas, até as funções de prevenção de acidentes. Além disso, a partir do monitoramento das condições do veículo, se tornam mais eficientes energeticamente, reduzindo o consumo de energia por meio de escolhas otimizadas de trajetória e controle da aceleração e velocidade do veículo. Em resumo, os veículos conectados apresentam suporte ao condutor em diferentes níveis, desde a condução autônoma completa até as funções mais básicas de assistência ao motorista.

2.2.2 A necessidade de manter-se na faixa

As saídas de faixa não intencionais podem resultar em acidentes graves, incluindo colisões com objetos fixos, veículos na mesma direção, tráfego contrário, pedestres ou ciclistas. Essas saídas podem ser causadas por distração do motorista, sonolência, desmaios temporários, alta velocidade, visibilidade inadequada das marcações das faixas e outros fatores.

Os sistemas de manutenção de faixa e aviso de saída de faixa têm como objetivo prevenir esses acidentes e reforçar a segurança durante a condução. Além disso, também proporcionam conforto ao reduzir tarefas repetitivas relacionadas à manutenção de faixa, liberam os condutores da monotonia associada a trajetos extensos. Entretanto, as marcações temporárias ou irregulares das faixas e outros sistemas de suporte lateral não são abordados atualmente por esses sistemas.

2.2.3 Diferença entre tecnologias de manutenção em faixa

Existem três categorias principais de sistemas de suporte ao motorista para evitar saídas não intencionais da faixa (ESKANDARIAN, 2012):

- Sistemas de Aviso de Saída de Faixa (LDWS): Esses sistemas emitem avisos ao motorista quando há uma saída não intencional da faixa. Eles não têm atuadores para influenciar a direção do veículo;
- Sistemas de Suporte à Manutenção da Faixa (LKAS): Esses sistemas ajudam o motorista a manter a faixa ativamente, por meio de um torque de assistência na direção do veículo. O motorista ainda é responsável pela manutenção da faixa;
- *Lane Centering*: São responsáveis por manter o veículo no centro da pista, a partir dos sensores e câmeras, esses sistemas apresentam o controle ativo do sistema de direção durante todo o trajeto.

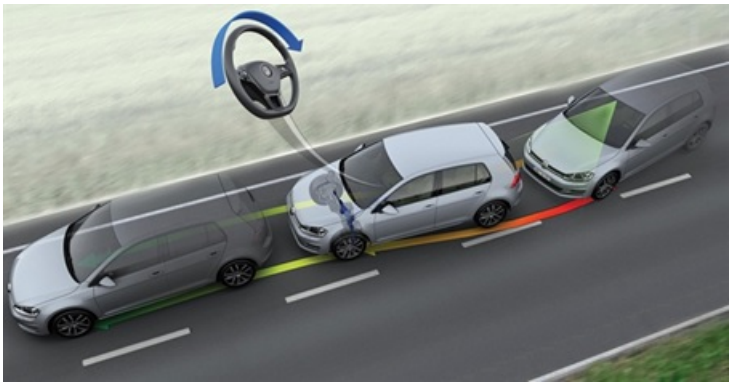
O LDWS utiliza elementos funcionais para detectar a posição da faixa e emitir um aviso ao motorista, solicitando ação corretiva. Ele pode operar com base em uma única faixa visível e usa marcadores de faixa para determinar a lateralidade.

Figura 3 – *Lane Departure Warning System* em veículo Volvo

Fonte: (HERMAWAN, 2017)

Já os LKAS possuem elementos adicionais, como um controlador de manutenção da faixa e um atuador para controlar a direção do veículo. O controlador calcula o torque de assistência adequado com base na posição lateral do veículo e a curvatura da estrada. O atuador converte essa saída do controlador em um torque de direção corretivo.

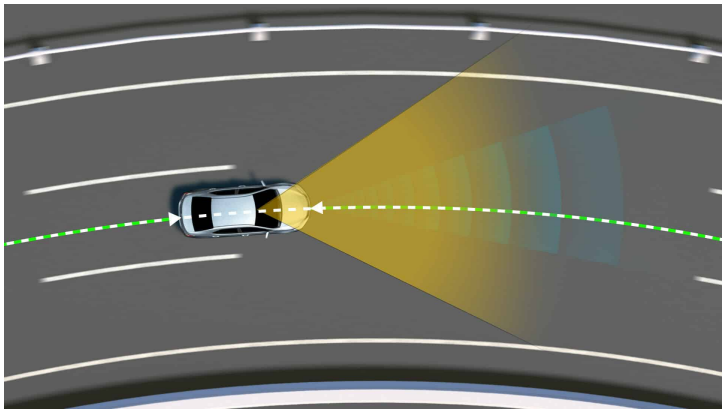
Figura 4 – *Lane Keeping System* em veículo Volvo



Fonte: (HERMAWAN, 2017)

O *Lane Centering*, por sua vez, apresenta um sistema mais avançado. Ao ser ativado, assume a responsabilidade contínua pela direção do veículo. A partir dos dados fornecidos pelas câmeras e sensores, o sistema utiliza de um controlador para calcular o ângulo de esterção do volante durante todo o trajeto. Dessa forma, o sistema é capaz de garantir maior conforto ao condutor em trajetos extensos. apesar dessa autonomia, o motorista deve manter-se atento para intervir quando necessário.

Figura 5 – *Lane Centering* em veículo Toyota



Fonte: (SHAR, 2023)

No livro de (ESKANDARIAN, 2012) é possível encontrar as diferentes variantes de sistemas de auxílio ao condutor no escopo de detecção de faixas, desde aqueles que fornecem um suporte leve ao motorista até sistemas mais rigorosos que mantêm o veículo centralizado na faixa. Alguns sistemas integram o LDWS a recursos adicionais, como frenagem seletiva das rodas ou detecção de pontos cegos. Essas tecnologias estão sendo amplamente adotadas por diversas montadoras em distintas regiões ao redor do mundo.

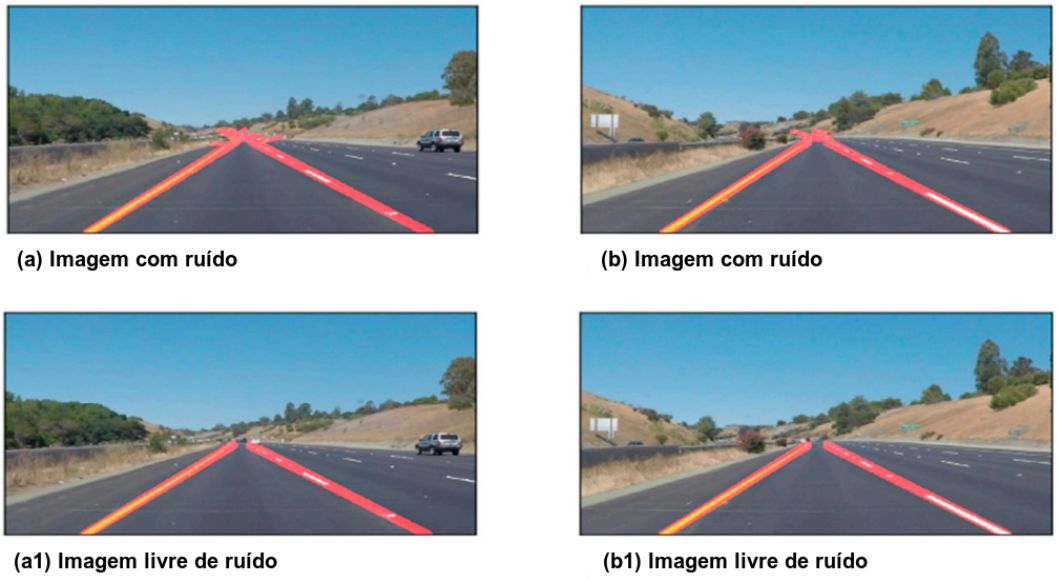
2.2.4 Algoritmo de detecção de faixas

Nos últimos anos, o estudo e desenvolvimento de técnicas de reconhecimento de faixas têm ganhado crescente interesse devido à necessidade de direção autônoma e segura, visando reduzir o número de acidentes. Nesse contexto, a detecção precisa de faixas é essencial para o desenvolvimento de sistemas ADAS em veículos.

Diversos métodos têm sido explorados para a detecção de faixas, incluindo abordagens baseadas em visão computacional, redes neurais convolucionais (CNNs) e aprendizado profundo. Em (JAVEED et al., 2023), os autores mencionam que métodos convencionais, como detecção de bordas, a transformada de Hough (HT), campos aleatórios de Markov (MRFs), o algoritmo de vinculação de bordas estendido (EELA) e o algoritmo RANSAC, também tem sido sido utilizados para encontrar e ajustar a linha das faixas.

A transformada de Hough, desenvolvida no início dos anos 1960, tem sido amplamente utilizada para detectar linhas retas verticais de interesse nas faixas. Essa transformada fornece resultados rápidos e eficazes, abordando vários desafios no processamento de imagens, como correlação e estruturas lineares. Além disso, foram propostos algoritmos que combinam a transformada de Hough com outras técnicas para melhorar a detecção de faixas em condições difíceis, como a presença de sombras, obstruções e variações ambientais.

Figura 6 – Combinação de algoritmos Hough Lines com técnicas de dispersão de ruído

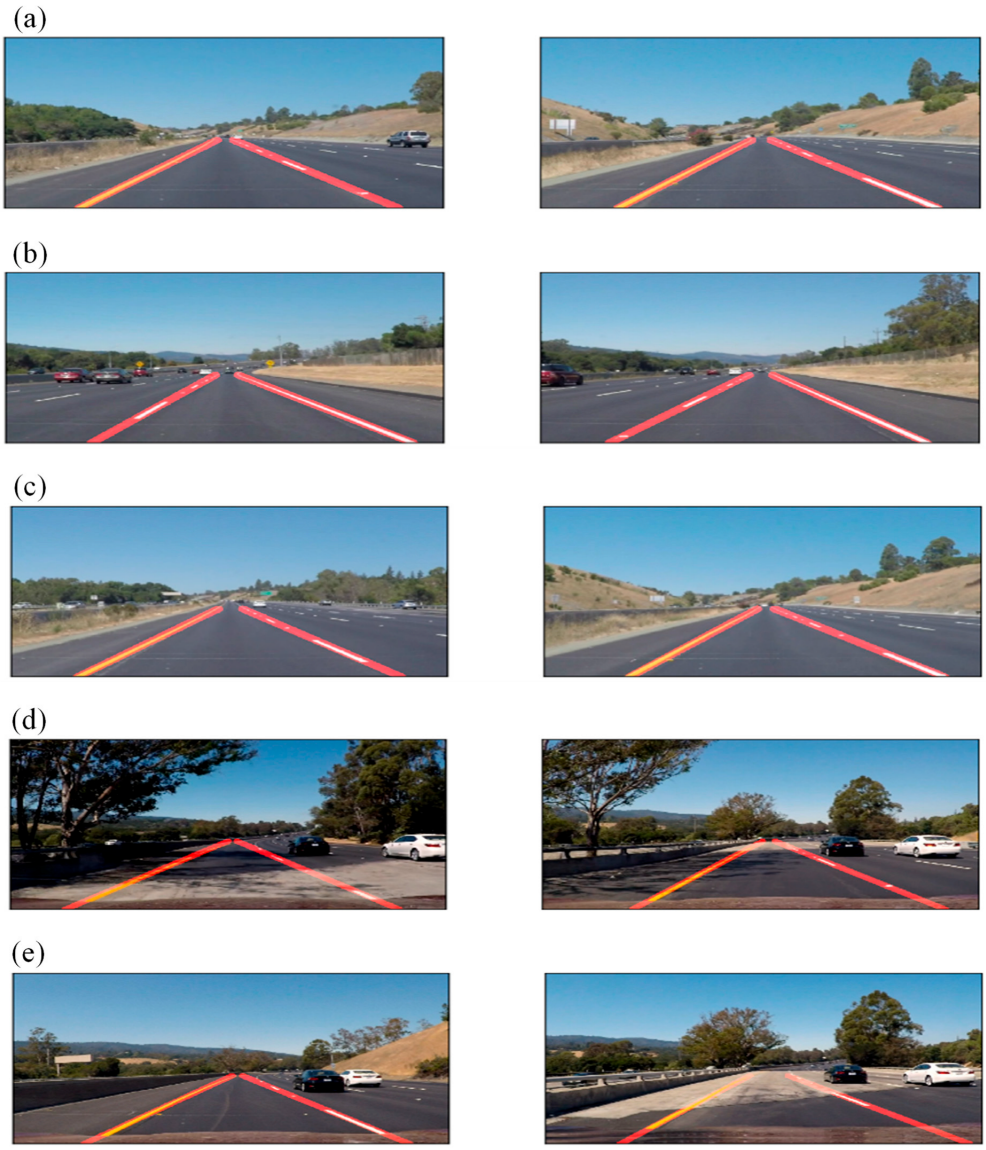


Fonte: (JAVEED et al., 2023)

Nos últimos anos, o avanço do *deep learning*, algoritmos preditivos de redes neurais, tem impulsionado o desenvolvimento teórico e tecnológico nessa área. Com o aumento do poder de processamento, algoritmos baseados em aprendizado profundo têm sido cada vez mais utilizados em sistemas de identificação de faixas. Além disso, técnicas de detecção inteligente de bordas têm mostrado resultados promissores para adaptar-se a condições desafiadoras da estrada e realizar um ajuste preciso da linha das faixas.

Apesar dos avanços, o reconhecimento de faixas ainda apresenta desafios, especialmente em condições reais dinâmicas e complexas. Variações no ambiente da estrada, como chuva, neblina e reflexos de sol, podem afetar a robustez e a confiabilidade dos sistemas de detecção. Portanto, a eficácia dessas abordagens é avaliada com base em seu desempenho em diversas condições complexas, levando em consideração fatores como sombras, obstruções, setas marcadas na superfície da estrada e tempo de execução. A Figura 7 ilustra os resultados obtidos pelo algoritmo de *Hough Lines* em um percurso realizado em altas velocidades, visando avaliar sua eficácia de processamento em pequenos intervalos de tempo. Nessa análise, o algoritmo demonstrou um desempenho satisfatório quanto a sua capacidade de detecção em condições dinâmicas e velozes, alcançando uma acurácia de 96.7% (JAVEED et al., 2023).

Figura 7 – Aplicação dos algoritmos de *Hough Lines* em diferentes percursos a velocidades elevadas



Fonte: (JAVEED et al., 2023)

Desta forma, o reconhecimento de faixas envolve várias etapas, desde a redução de ruído até o cálculo de gradientes, seguido pela aplicação da transformada de Hough para localizar a região de interesse (ROI). A transformada de Hough tem sido amplamente empregada, com diferentes variações propostas para melhorar seu desempenho, levando em consideração a redução do tempo de processamento e das necessidades de armazenamento de dados. A combinação de técnicas convencionais e métodos baseados em aprendizado profundo tem mostrado resultados promissores, buscando alcançar um reconhecimento preciso das faixas em condições reais desafiadoras.

A captação de imagens é parte essencial para o funcionamento dos algoritmos, e está diretamente conectada às técnicas estabelecidas e inovações baseadas em *deep*

learning para a detecção e análise de faixas em tempo real. Nesse contexto, as câmeras desempenham um papel crucial, uma vez que são as responsáveis por fornecer os dados de entrada ao sistema.

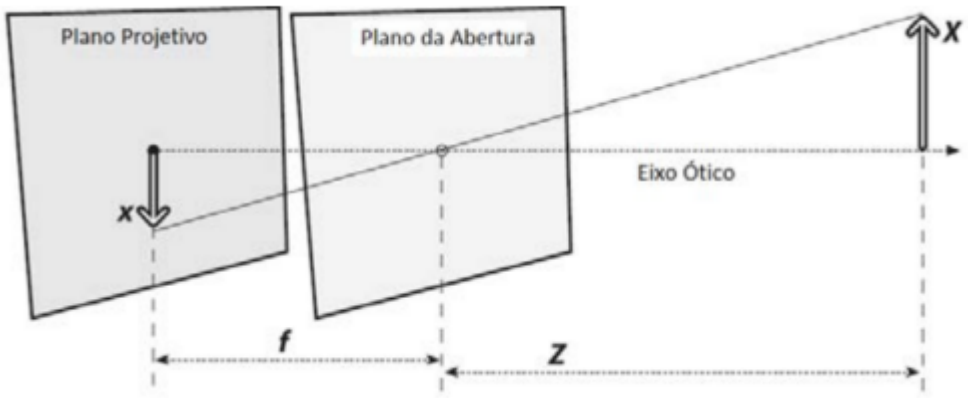
2.3 Câmera

2.3.1 Aquisição de imagem

O processo de aquisição de imagens consiste no dispositivo que será utilizado para a captação dos dados desejados, que posteriormente serão processados no sistema. As câmeras atuam como sensores que respeitam os mesmos princípios do olho humano, a partir das variações na quantidade de luz nos diferentes comprimentos de onda. De acordo com Bradski (2008), a modelagem de qualquer câmera parte do principio do modelo proposto por Aristóteles que ficou conhecida atualmente por *pinhole* ou "buraco de agulha". Representando o modelo mais simples de uma câmera real, essa arquitetura funciona a partir de um pequeno furo em um dispositivo, que permite a passagem da luz para um segundo plano onde é projetada a imagem.

A partir da passagem de luz pelo orifício, a imagem projetada sofre transformações quanto a representação dos objetos tridimensionais, uma vez que serão armazenados no plano projetivo em sua forma 2D. É possível estabelecer uma relação geométrica entre o objeto e a figura, uma vez que durante esse processo a imagem é invertida e reduzida proporcionalmente ao tamanho do objeto e a distância focal, distância entre a imagem e o orifício.

Figura 8 – Câmera *pinhole*



Fonte: Bradski (2008)

Dessa forma, é possível determinar a relação da altura da projeção e a altura do objeto a partir da equação de semelhança de triângulos. Onde X é objeto real, Z é o eixo ótico, f a distância focal e x a imagem:

$$\frac{X}{Z} = \frac{-x}{f} \tag{2.1}$$

O sinal negativo pode ser desconsiderado, pois apenas representa que a imagem projetada é invertida, simplificando a equação para:

$$\frac{X}{Z} = \frac{x}{f} \tag{2.2}$$

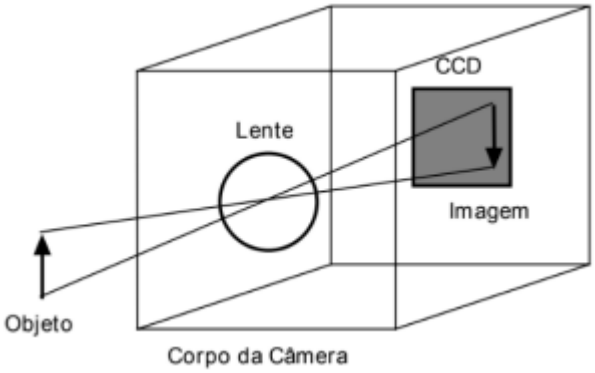
A partir deste modelo, foram desenvolvidos os processos de captação e armazenamento de imagens utilizados nos dias atuais. Entretanto, na prática são adicionados lentes a esse dispositivo para auxiliarem na formação da imagem, responsáveis por redirecionar a luz em um ponto de convergência conhecido. Das diferentes funções que as lentes garantem ao processo de aquisição de imagens, majoritariamente estão relacionadas ao controle da quantidade de luz de entrada no dispositivo a partir da regulagem da abertura da lente. A partir desse controle é possível controlar a quantidade de luz que atinge o fundo do dispositivo, o que garante benefícios quanto as propriedades da imagem projetada.

Uma vez que as lentes conseguem alterar o ponto de convergência, é possível determinar a profundidade de uma imagem, uma vez que atua sobre a distância focal, permitindo que a projeção de um objeto seja representada com as dimensões maiores do que esperadas pela equação dos triângulos retângulos. As lentes também auxiliam na correção das imagens, onde busca garantir que a imagem não apresente problemas de desfoque e distorção, e sejam armazenadas com maior fidelidade.

Para realizar o processo de digitalização, componentes fundamentais foram adicionados as arquiteturas das câmeras, os sensores fotossensíveis. Foram alocados no plano projetivo da câmera, de maneira que irão armazenar a carga elétrica proporcional a quantidade de luz emitida (OGê; HUGO, 2014). Esses sensores funcionam a partir da captação de um espectro eletromagnético da luz, os mais comuns são o verde, o vermelho e o azul. E a partir dos filtros de cores se torna possível combinar esses sensores para uma melhor representação, uma vez que cada sensor contém os valores referentes a um único espectro eletromagnético.

Esses dispositivos meramente analógicos sofreram diversas transformações e adaptações durante as últimas décadas. O processo de digitalização se torna cada vez mais preciso, uma vez que o avanço da tecnologia proporciona algoritmos cada vez mais otimizados. E a partir disso, novos sistemas foram implementados nestes dispositivos denominando um marco na história, uma vez que permite o processamento de imagem em tempo real, nas chamadas câmeras inteligentes.

Figura 9 – Arquitetura de uma câmera monocromática com sensor CCD (*Charge Coupled Devices*) simples



Fonte: Ogê e Hugo (2014)

2.3.2 Câmeras inteligentes

Devido a quantidade massiva de informações disponíveis no cotidiano, as câmeras ganham cada vez mais espaço no mercado e diferentes objetivos além da fotografia convencional. A partir de sensores de alta resolução e processadores de imagem, as câmeras inteligentes funcionam como dispositivos capazes de realizar uma determinada tarefa pré-programada sem que haja a intervenção humana, como por exemplo, a detecção e classificação de objetos.

A partir da captura de imagem e execução de algoritmos em tempo real, torna-se possível vislumbrar diversas novas oportunidades de mercado. Comumente utilizada no setor de segurança, para auxiliar no monitoramento de diferentes locais simultaneamente, uma vez que garante os recursos de visão computacional a partir do processamento em tempo real (CONTRAMANUAL. . . , 2006). As câmeras inteligentes se tornam cada vez mais comuns nas áreas automotivas, vinculados a estrutura dos sistemas de auxílio aos condutores ADAS, permitem ao veículo identificar o ambiente ao seu redor e dessa forma recolher e classificar as informações visuais necessárias para os outros sistemas do veículo.

No contexto de sistema de auxílio ao condutor, a utilização desse tipo de câmera se torna muito mais importante devido a constante necessidade de fornecer aos sistemas de tomada de decisão a maior quantidade de dados processados, para que este analise todos os dados para determinar a resposta do veículo. Para garantir os recursos necessários de processamento de imagem avançados para auxiliar na segmentação de dados, o projeto utiliza uma câmera Zed 2I como entrada para realizar as funções voltadas à visão computacional.

2.4 Visão Computacional

A visão computacional está relacionada à capacidade de computadores de extraírem informações a partir de um frame (foto) ou um conjunto de frames (vídeos). Para Howe (2014), o termo visão computacional pode ser definido como a extração da informação de conteúdo visual, a partir do estudo da análise automática de imagens e vídeos por computadores, de forma a alcançar algum entendimento do mundo. A ideia geral é permitir que uma máquina consiga atingir o aprendizado próximo ao "humano"na captura de dados sobre uma imagem, a partir das técnicas de Processamento Digital de Imagem (PDI).

2.4.1 Imagem Digital

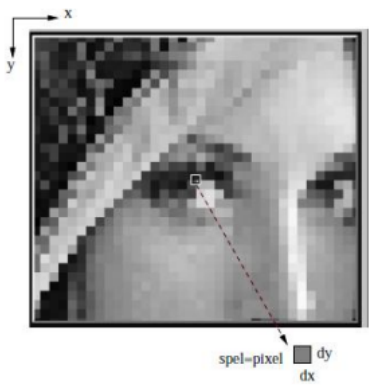
Queiroz e Gomes (2006) utiliza uma definição matemática para caracterizar uma imagem, onde a partir de uma função bidimensional de intensidade luminosa f(x,y) onde x e y representam as coordenadas espaciais, e o valor final da função f retorna a quantidade proporcional de brilho em tais coordenadas. A imagem natural, é caracterizada por ser uma diversidade de tons e cores, que variam quanto à frequência do vermelho ao azul em sua matiz específica. Quando essa imagem é convertida e representada numericamente em um dispositivo, ela é chamada de imagem digital.

No livro de Ogê e Hugo (2014), o conceito de Imagem Digital é definido como uma imagem f(x,y) discretizada tanto espacialmente quanto em amplitude. Ou seja, uma imagem é um conjunto de linhas que formam uma matriz, em que cada ponto, coordenadas x e y, irá identificar uma tonalidade de cinza respectiva a esse ponto. Cada ponto apresenta suas próprias características a partir de uma média do brilho naquele ponto, calculado a partir de uma escala de tonalidades do cinza. Dessa forma, cada ponto na matriz apresentará um valor relacionado ao seu brilho, e foram nomeados *pixels*.

As representações em escala de cinza apresentam 256 níveis de cinza, onde o valor 0 é utilizado para representar a cor preta, ou seja a ausência de luz, enquanto o valor 255 representa o branco. Dessa forma, é possível representar imagens captadas pelo olho humano através de valores que representam a ausência ou presença de luz espalhados em uma matriz de pontos, como ilustra a figura 10.

Com o sucesso da imagem digital em formato binário, surgem novas propostas para encontrar diferentes técnicas de representação de imagens, explorando novas possibilidades, como por exemplo o modelo de cores. Atualmente, grande parte das imagens digitais são coloridas, sendo necessária novas abordagens para serem representadas. Porém, dos diferentes modelos que existem atualmente, a principal diferença entre eles e a imagem digital apresentada (Figura 10), consiste no fato que um pixel estará associado a mais de um valor, e não apenas a um número no intervalo entre 0 e 255. O sistema RGB

Figura 10 – Imagem digital em formato matricial

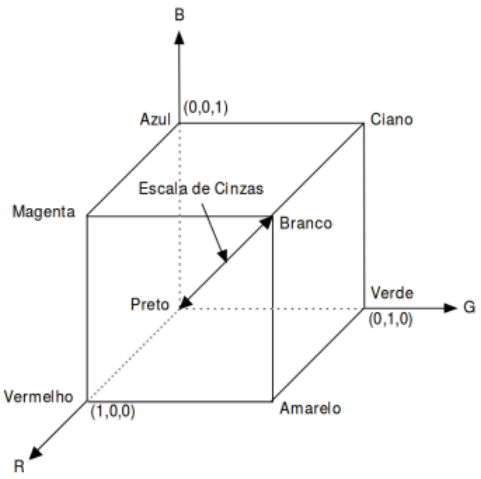


Fonte: (FALCãO; CHIACHIA, 2014)

(Red, Green, Blue) é um dos mais conhecidos, em que consiste na combinação das cores primárias para formar o branco. Dessa forma, a partir da combinação das tonalidades do vermelho, azul e verde é possível representar uma gama de espectros de cores visíveis. Portanto, cada pixel estará associado a um vetor com 3 elementos, indicando a tonalidade das cores do padrão RGB que combinadas irão resultar em uma cor diferente.

Dessa forma, o modelo RGB segue um padrão de matriz tridimensional, ou seja um cubo (figura 11), onde suas arestas representam as três cores primárias e vão de 0 até 1. Em que, para que um ponto possa estar associado a esse padrão deverá manter-se dentro dos limites da matriz, por exemplo, a cor azul pode ser representada a partir do conjunto (0, 0, 1). Uma vez que as cores são combinadas 2 a 2, a partir do seu comprimento de onda, são geradas as cores secundárias ciano, magenta e amarelo. Assim, é possível criar uma relação de oposição entre cores primárias e secundárias a partir das suas pigmentações e representar uma imagem mais próxima a realidade.

Figura 11 – Sistema de cores RGB



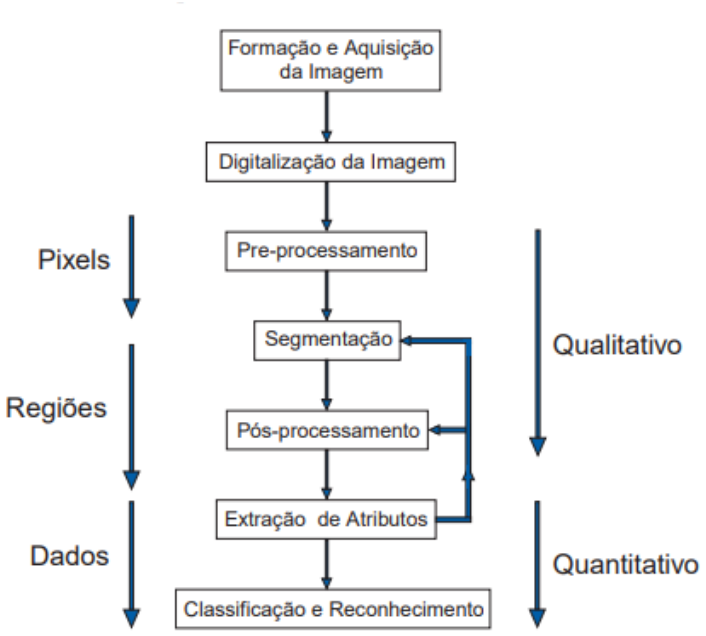
Fonte: (OGê; HUGO, 2014)

Diferentes sistemas para representar as tonalidades do cotidiano surgiram, cada um com sua proposta e seu catálogo de cores possíveis, como os sistemas HSV e HSL com as propostas de assimilarem melhor as cores ao olho humano. Assim, esse sistemas contam com diferentes parâmetros para representar suas cores, coma a matiz, representada pelo H (hue), saturação, representada pelo S (saturation), valor ou luminosidade, representados respectivamente pelo V (value) e L (lightness). Vale notar que a maneira como são armazenados estes valores não mudam, dessa forma, a combinação das coordenadas de um pixel são armazenadas da mesma maneira, porém, representam cores diferentes em modelos diferentes.

2.4.2 Processamento Digital de Imagem

O maior desafio do processamento digital de imagens consiste na percepção das máquinas quanto aos cenários do mundo real. Devido à complexidade e diversidade de elementos provenientes do nosso cotidiano, existem diversos empecilhos que dificultam o processamento de imagens. As condições de iluminação são um dos principais fatores que geram incoerência quando se trata de modelos pré-treinados, algoritmos de aprendizado de máquina treinados a realizar uma tarefa específica, uma vez que a falta de iluminação ou exagerada sobre o conteúdo examinado altera as conclusões do modelo, além da questão da qualidade da imagem recebida que muitas vezes apresentam reflexos e sombras. Para isso, o sistema de processamento de imagens é composto por diferentes etapas responsáveis por guiar o processo da maneira adequada, sendo estas ilustradas abaixo (figura 12).

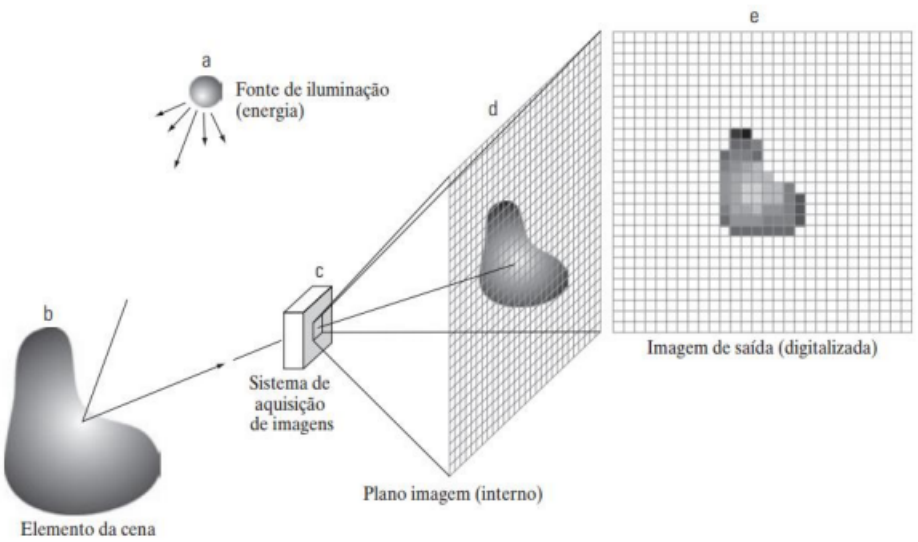
Figura 12 – Etapas do Processamento de Imagens



Fonte: (ANDRADE; ALBUQUERQUE; ALBUQUERQUE, 2003)

O processo se inicia com a etapa de formação e aquisição da imagem, que está relacionada a um dispositivo externo sensível ao espectro de energia eletro-magnético, normalmente uma câmera, que será responsável por converter um cenário tridimensional em uma imagem bidimensional, fornecendo como saída um sinal analógico para ser digitalizado. Em sequencia, ocorre a conversão da imagem em valores digitais a partir dos sinais analógicos captados, onde esses valores serão processados para permanecerem no intervalo disponibilizado de um pixel, e em seguida alocado ao pixel correspondente aquela posição na matriz, como explicado na seção anterior. Utiliza-se como padrão, disponibilizar 1 byte (8 bits) para cada pixel da imagem ser armazenada em uma escala de tons de cinza, e 3 bytes para os *pixels* quando se trata dos demais modelos de imagem coloridas (GONZALEZ; WOODS, 2009).

Figura 13 – Processo de aquisição e digitalização de imagens



Fonte: (GONZALEZ; WOODS, 2009)

A função do pré-processamento é melhorar as condições visuais da imagem para as próximas etapas, dessa maneira existem diferentes técnicas que são adaptadas ao escopo de cada modelo. Dentre as diferentes técnicas de pré-processamento existe a utilização do filtro de mediana comumente utilizado para reduzir o ruído em imagens digitais, classificada como um filtro de suavização. Possui esse nome devido a maneira como aborda a imagem, percorrendo todos os *pixels* e substituindo seu valor pelo cálculo da mediana daquele ponto juntamente com seus vizinhos. A conectividade entre pixels pode apresentar variações para que melhor se adapte ao escopo proposto, entretanto, uma das abordagens é considerar os *pixels* vizinhos como sendo aqueles em contato com o pixel analisado, formando um "cerco"ao seu redor. Dessa maneira, podemos dizer que um pixel P localizado nas coordenadas (x,y) apresentará como *pixels* vizinhos o seguinte vetor de coordenadas:

$$[(x, y-1), (x, y+1), (x-1, y), (x+1, y), (x-1, y-1), (x+1, y-1), (x-1, y+1), (x+1, y+1)] \tag{2.3}$$

A partir dessa abordagem, o filtro consegue preservar melhor as bordas e destacar as características principais da imagem, reduzindo a presença de ruídos como mostra a figura 14. A partir do tamanho da janela de atuação do filtro, ou seja, o número de *pixels* que serão utilizados como vizinhos, irá determinar os aspectos de supressão de ruídos e suavização da imagem. Embora uma janela ampla consiga reduzir maior quantidade de ruídos, alguns detalhes da imagem serão perdidos.

Figura 14 – Aplicação do filtro de Mediana

Fonte: (OGê; HUGO, 2014)

A próxima etapa, consiste no processo de Segmentação, onde a imagem é dividida em segmentos menores baseado em uma característica específica. Esses segmentos são denominados objetos, que consistem em um agrupamento de *pixels* que apresentam algum valor informacional ao sistema, e então são classificadas como zonas de interesse. Assim, como no pré-processamento, existem diferentes técnicas para realizar a segmentação de imagens, não havendo um modelo formal para realizar essa etapa. Desse modo, fica a critério dos responsáveis pelo modelo quais tecnologias ou combinação delas será implementada, uma vez que essa etapa é considerada crítica para o tratamento de informação. Desse modo, se torna um processo adaptativo, onde diferentes abordagens podem ser implementadas afim de garantir uma maior precisão das zonas de interesse.

Dentre as diferentes técnicas de segmentação, a Limiarização tem sido bastante utilizada devido a leva de benefícios que proporciona, uma vez que é fácil e simples de

ser aplicada e apresenta custo computacional relativamente baixo(OGê; HUGO, 2014). A técnica em si apresenta uma premissa muito simples, onde o objetivo é definir os valores associados em cada pixel aos valores extremos representados pelos números 0 e 255 na escala de cinza. Dessa forma, é estipulado um valor limiar dentro do intervalo acima, onde os *pixels* abaixo desse valor receberão o valor 0 enquanto os que apresentam um valor maior a esse limiar receberão o valor 255, permitindo que cada pixel da imagem possa ser classificado de maneira binária (0,1), o que facilita a segmentação da imagem nas áreas de interesse a partir da diferença de intensidade dos *pixels*.

Figura 15 – Aplicação do filtro de limiarização

(a) Imagem em tons de cinza

(b) Imagem após limiarização

Fonte: (OGê; HUGO, 2014)

As técnicas de segmentação são imperfeitas e apresentam defeitos ao final da sua etapa, assim o pós processamento é a etapa responsável por corrigir os defeitos provenientes dessas técnicas. Para isso são utilizadas as operações morfológicas, onde atuam em duas principais áreas, erosão e dilatação. De maneira sucinta, essa etapa busca preencher lacunas vazias e suprimir ruídos provenientes na imagem. Onde a dilatação busca expandir os limites do objeto, a partir da posição dos *pixels* presentes nas extremidades do objeto, onde são comparados com seus vizinhos, onde serão verificadas se existe semelhança com o pixel atual, expandindo a região do objeto.

Enquanto a dilatação atua nas margens do objeto, a erosão busca encontrar defeitos no seu interior, uma vez que essa permite separar objetos que se chocam. Desse modo, a erosão busca por "furos"dentro do objeto que não o pertencem, da mesma maneira que a dilatação, porém quando um pixel de valor divergente é encontrado em relação aos seus vizinhos este recebe um valor menor para ser desconsiderado, de maneira a diminuir o tamanho final do objeto.

A partir desse ponto, é possível realizar a extração de atributos, onde de fato retornará as informações visuais provenientes da imagem. Para isso, utiliza-se a rotulação ou labelização, responsáveis por avaliar os objetos extraídos da etapa anterior e deferir

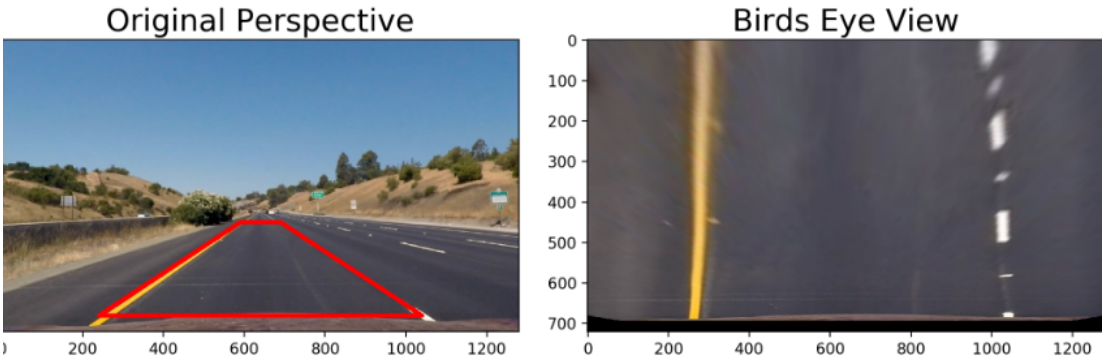
etiquetas ou rótulos, afim de distinguir diferentes objetos dentro da mesma imagem. Após essa etapa, inicia-se o último estágio do processamento responsável pela classificação e reconhecimento. A partir dos pontos semelhantes em objetos diferentes, o sistema deve classificar de maneira automática esses objetos com características em comum em um mesmo grupo. Sendo assim, a etapa de classificação e reconhecimento é essencial, uma vez que a partir do modelo de classificação selecionado afetará diretamente na precisão do sistema de reconhecimento.

2.4.3 Técnica *Bird's Eye*

O objetivo dessa técnica é proporcionar uma vista superior da imagem capturada pela câmera, de modo que o observador tenha a sensação de estar vendo a imagem de cima, como se fosse um pássaro. No contexto do projeto, isso se torna relevante uma vez que facilita a análise de elementos importantes, no caso as faixas de rodagem. Essa técnica permite corrigir aspectos de posicionamento da câmera. Dessa maneira, esta técnica representa as imagens em um formato trapezoidal, expandindo a parte superior da imagem para diminuir a distorção de perspectiva. Para o projeto, o *Bird's Eye* exibe outra funcionalidade essencial, uma vez que a imagem após o processamento irá eliminar a conversão das faixas de rolagem causada pela perspectiva. Assim, a imagem final irá representar 2 retas paralelas, além de apresentar pontos cegos que ficariam ocultos em uma vista frontal.

Além disso, podem ser combinadas com diferentes funções para selecionar uma região de interesse. Dessa maneira se torna possível mapear os limites de *pixels* utilizadas na entrada da função para determinar o tamanho do quadro da saída. O conteúdo capturado pela câmera apresenta uma diversidade de informações que não apresentam nenhum valor para os algoritmos de *Lane Detection*, como carros, árvores e placas. Uma das maneiras de se obter o resultado desejado é escolher 4 pontos na imagem passada como parâmetro, de modo que estes pontos devem estar todos em cima das faixas de rolagem. Cada ponto em uma faixa deve ter um par com a mesma altura na segunda faixa de rolagem, de maneira que quando conectados formem um trapézio, como ilustrado a seguir (figura 16).

Figura 16 – Aplicação do filtro de Mediana



Fonte: (JAVEED et al., 2023)

Em seguida, é possível observar a tabela de coordenadas dos *pixels*:

Origem	Destino
240, 680	240, 680
1040, 680	1040, 680
590, 450	240, 0
690, 450	1040, 0

2.5 Controle do sistema de direção elétrica

Quando se torna viável calcular a distância entre diferentes pontos da imagem por meio dos mapeamentos de *pixels*, é possível, também, determina as variáveis essenciais na implementação do LKA. Dentre elas, destaca-se a capacidade de calcular a distância entre o veículo e a faixa em que ele está posicionado. Na sequência, o processo consiste em interpretar os dados e enviar comandos ao sistema de direção afim de corrigir a trajetória do veículo. Porém, se esses dados brutos forem utilizados diretamente para definir os ângulos de esterçamento do volante no sistema de direção, resultaria em manobras bruscas e imprecisas que podem comprometer a trajetória do veículo.

Dessa forma, um sistema de controle de direção é utilizado para manter o veículo no centro da faixa. Javeed et al. (2023) sugere que os controladores PI e PID convencionais, responsáveis por ajustar o sistema em resposta a um sinal, não são suficientemente eficientes para garantir um funcionamento suave e estável. Para melhorar o desempenho, propõe-se a utilização de um controlador fuzzy PID, ajustando os parâmetros do PID. O controlador fuzzy PID utiliza o erro e a derivada do erro como sinais de entrada, e os parâmetros Kp, Ki e Kd como saídas. Através do uso da biblioteca Scikit-Fuzzy em *Python*, o controlador de ângulo de direção é ajustado de acordo com as regras projetadas,

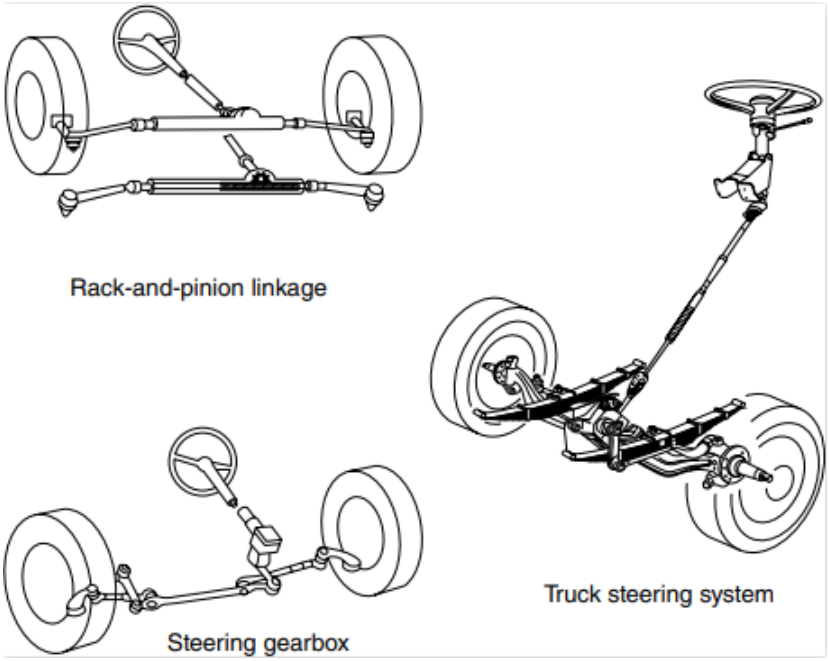
visando obter a melhor resposta dinâmica do veículo com menor sobressinal e erro em regime permanente.

A eficácia dos sistemas propostos é demonstrada por meio de simulações no ambiente Gazebo, utilizando o *software* Gazebo/ROS. Foram realizados experimentos para verificar os algoritmos de detecção de faixa e controle de direção. Os resultados da detecção de faixa e estimativa do ângulo de direção são satisfatórios. O desempenho dos controladores PID e fuzzy-PID é comparado em diferentes velocidades, sendo que o controlador fuzzy-PID apresenta menor sobressinal. Conclui-se que o sistema de assistência proposto proporciona maior precisão e estabilidade na manutenção da faixa.

2.6 Sistema de Direção

Gillespie (1992) define um sistema de direção como sendo capaz de guiar as rodas dianteiras em resposta aos comandos do motorista, com o objetivo de proporcionar o controle direcional do veículo como um todo. A direção do veículo é controlada fundamentalmente pelas rodas dianteiras, de maneira que deva proporcionar segurança e precisão ao motorista ao realizar manobras. Assim, existem diferentes abordagens de controle, que variam quanto a sua complexidade. Dentre os modelos mais conhecidos, se destacam nos veículos de passeio os modelos pinhão-cremalheira (*Rack-and-Pinion*) e o da caixa de redução (*Steering gearbox*), que apresentam arquitetura mais simples quando comparadas aos veículos de carga.

Figura 17 – Modelos de sistemas de direção



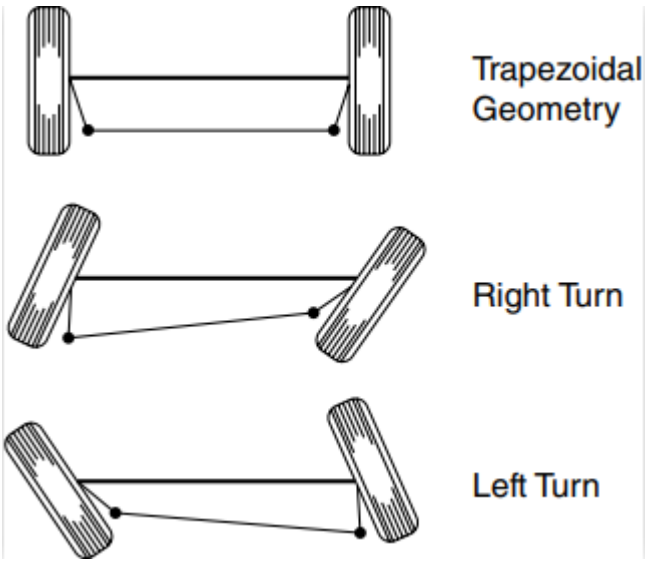
Fonte: Gillespie (1992)

2.6.1 Geometria de Ackerman

A geometria de Ackerman é um conceito fundamental na engenharia automotiva, desempenhando um papel essencial no projeto de sistemas de direção, uma vez que se concentra na configuração das rodas dianteiras de um veículo e sua relação durante curvas. A geometria de Ackerman busca encontrar um ponto de encontro, chamado de "ponto de Ackerman", onde as linhas traçadas a partir das rodas dianteiras se cruzam. Essa geometria é alcançada através da diferença controlada dos ângulos de esterçamento das rodas dianteiras, com o objetivo de proporcionar um movimento mais preciso e suave do veículo durante as curvas.

No início do século XIX, Rudolph Ackerman fez uma importante descoberta relacionada aos ângulos das rodas dianteiras de carruagens. Ele percebeu que ao variar os ângulos das rodas, era possível obter diferentes resultados ao percorrer trajetórias circulares. Observou-se que ao utilizar a geometria de trapézio, na qual a roda interna é direcionada a um ângulo maior do que a roda externa, o veículo fazia curvas mais fechadas. Isso ocorre porque a roda interna não precisa percorrer a mesma distância que a roda externa, uma vez que o raio da curva será menor.

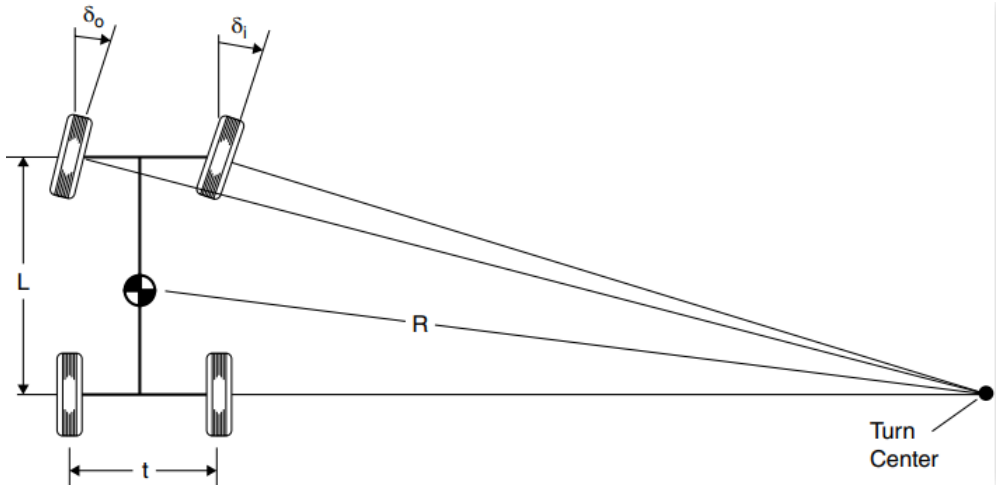
Figura 18 – Arquitetura de direção trapezoidal



Fonte: Gillespie (1992)

A partir da diferença dos ângulos de esterção, é possível diminuir o deslocamento e evitar deslizamento lateral desnecessário em trajetórias circulares. Com a geometria de Ackerman aplicada corretamente, o movimento se torna suave, garantindo ao condutor melhor controle do veículo. Na imagem a seguir (figura 19), são explicitadas as variáveis utilizadas pela geometria para o cálculo dos ângulos de esterçamento das rodas dianteiras a partir do valor da bitola (t), da distância entre os eixos (L) e o raio do giro (R).

Figura 19 – Geometria de direção Ackerman



Fonte: Gillespie (1992)

Calculado a partir da equação:

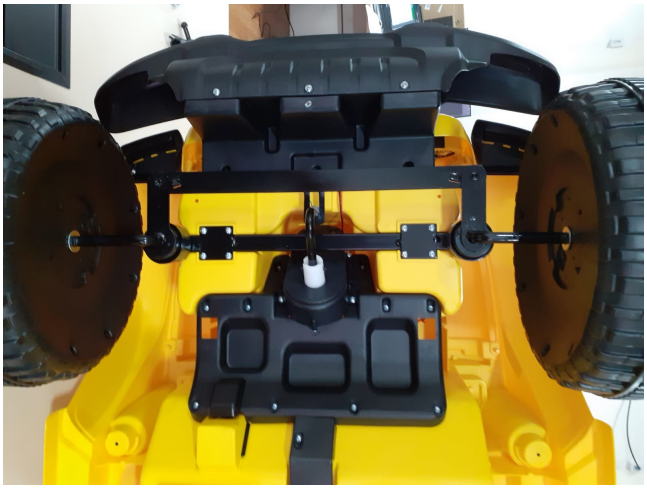
$$\delta_0 = \arctan\left(\frac{L}{R + \frac{t}{2}}\right) \tag{2.4}$$

$$\delta_i = \arctan\left(\frac{L}{R - \frac{t}{2}}\right) \tag{2.5}$$

A busca pelo ponto de Ackerman ideal apresenta desafios complexos, pois diversos sistemas mecânicos do veículo influenciam o comportamento dos ângulos de esterçamento mencionados nas equações anteriores. O valor ideal, situado no centro da curva, permite um movimento fluido e eficiente do veículo. No entanto, inúmeros fatores estruturais e condições do terreno podem limitar ou modificar o valor do Ackerman. Entre esses fatores destacam-se os diferentes modelos de design veicular, a velocidade do veículo, restrições estruturais do sistema e variação na curvatura, que dificultam em definir uma geometria ideal. Em contrapartida, os veículos convencionais de transporte não apresentarão Ackerman igual a 0, uma vez que isso implica na mesma esterção das duas rodas dianteiras, o que ocasiona maior dificuldade em realizar curvas com raios menores.

No entanto, é importante mencionar que alguns modelos mais simples, como o veículo de pequena escala abordado neste documento, ainda possuem uma geometria de Ackerman próxima de zero, ou seja, com um ângulo de Ackerman mínimo. Dessa forma, o veículo de estudo apresenta a geometria das barras combinadas com o modelo de caixa de redução para o controle da direção, composta por barras, alocadas paralelas ao veículo longitudinalmente, conectadas as duas rodas dianteiras, como ilustra a figura 20.

Figura 20 – Geometria das barras do veículo de pequena escala



Fonte: O autor

2.6.2 Modelo da Caixa de Redução

De maneira essencial a direção do veículo, o modelo da caixa de redução atua como o componente responsável por transmitir o movimento de rotação do volante para as rodas dianteiras. (GILLESPIE, 1992) define o funcionamento deste modelo a partir de uma caixa de engrenagens responsáveis por captar a rotação da coluna de direção, e a partir do encaixe das engrenagens, transmitir o movimento para as barras que direcionam o veículo.

A relação entre o ângulo de giro do volante e o ângulo de giro das rodas pode variar de acordo com os objetivos e o design do sistema de direção, buscando garantir um nível mínimo de conforto para o motorista. Geralmente, o ângulo de giro do volante é maior do que o ângulo de giro das rodas, o que permite um melhor controle do veículo com uma quantidade relativamente menor de força aplicada. Essa diferença, garante uma resposta precisa do veículo ao movimento do volante. Quando a rotação da direção passa pela caixa de redução, as sequencias de engrenagens atuam reduzindo esse angulo para converterem em um giro de roda menor.

Em sistemas mais complexos, esses modelos contam com motores elétricos para auxiliarem na rotação do volante e garantir um maior conforto ao condutor. Os motores atuam por meio de um torque adicional na mesma direção do volante, de modo a acompanhar a rotação e reduzir o esforço do condutor. E dessa forma, o sistema de direção se torna mais leve e mais fácil de ser manejado.

A revisão bibliográfica sobre os temas abordados neste trabalho foram conduzidas de maneira abrangente, proporcionando compreender as teorias e desafios associados a essa área. A partir dos objetivos em comum apresentados nos outros trabalhos, este estudo

busca implementar com êxito o controle do sistema de direção a partir de captações de imagem em tempo real. Neste contexto, a seção subsequente discutirá a metodologia adotada, demonstrando os passos e abordagens para alcançar os objetivos propostos a partir dos conhecimentos adquiridos durante a revisão bibliográfica.

3 Metodologia

Este capítulo apresenta a metodologia adotada durante a confecção deste trabalho, descrevendo os principais métodos e ferramentas que estão sendo utilizadas para a implementação das funcionalidades de *Lane Keeping* e *Lane Detection*. Dessa forma, o presente capítulo foi dividido em Estrutura, Processamento de imagens e Arquitetura, onde serão apresentados os componentes fabricados, os algoritmos desenvolvidos e a comunicação dos sistemas.

3.1 Estrutura

O projeto SegurAuto apresenta diferentes componentes fabricados para atender as necessidades do projeto, nessa etapa serão apresentados os componentes mecânicos e estruturais que foram fabricados para instalação dos equipamentos utilizados para o desenvolvimento do trabalho.

3.1.1 Instalação da câmera

Para a captação de imagens, o projeto utiliza uma câmera Zed 2i. Em uma tentativa de replicar a visão humana, esta câmera apresenta 2 "olhos", lentes posicionadas com uma pequena distância uma da outra, permitindo a triangulação do ambiente ao seu redor para fornecer dados tridimensionais do cenário. Além de estar equipada com diversos sensores que disponibilizam dados acerca de posicionamento geográfico, visão estéreo e sensores de movimento. Por meio da API disponibilizada pela SteroLabs, a fabricante, é possível ter acesso a esses dados e permite utilizarmos diferentes configurações dos sensores.

Figura 21 – Câmera Zed 2i - Stereolabs

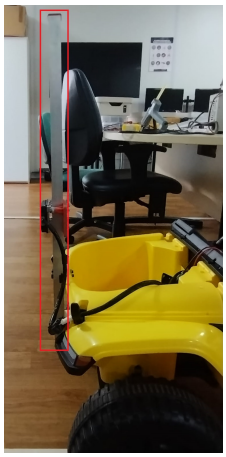
Fonte: Stereolabs Docs

O posicionamento da câmera é essencial para melhorar o desempenho dos algoritmos do projeto, a partir dessa premissa, o projeto necessitava de um suporte para

posicionar a câmera na frente do veículo. Uma vez que a altura e a inclinação ideal da câmera não foram definidos, o suporte deveria ser fabricado de maneira que permitisse realizar alterações nesses parâmetros afim de validar a posição mais compatível com os dados capturados da API.

A estrutura elaborada consiste numa haste de metalon de perfil de 50x25mm e comprimento de 750mm como estrutura de sustentação para os suporte da câmera instalado na frente do veículo, de maneira que permita o controle da posição da câmera na vertical (altura), ou seja, da manipulação da distância que a câmera se encontra do solo.

Figura 22 – Base de Metalon perfil 50x25mm instalada no veículo



Fonte: O autor

Por sua vez o mecanismo para controlar a inclinação da câmera é mais complexo, e dessa forma foram utilizadas diferentes ferramentas para sua fabricação. Foram realizadas as medidas das dimensões da câmera a partir do *software* CATIAV5 para modelagem do suporte. Esse mecanismo conta com 2 peças que se encaixam a partir de um parafuso, a base e o suporte da câmera, e a partir da movimentação dessa peças que permite ajustar a inclinação da câmera. Posteriormente, utilizando uma impressora 3D, foram impressas as peças e instaladas na haste do veículo.

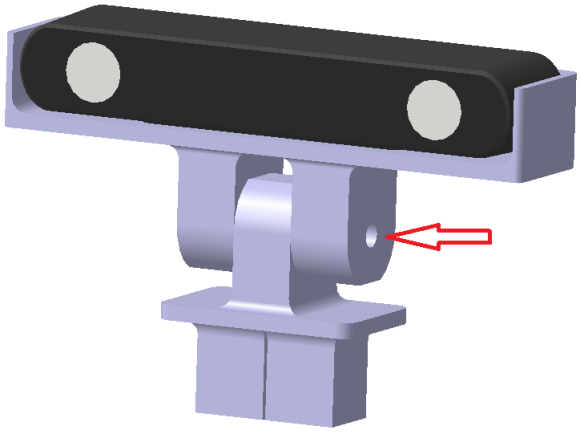
Figura 23 – Design das peças utilizadas para suporte da câmera

(a) (b)

Fonte: O autor

A figura 24 demonstra o encaixe das peças impressas com o parafuso que regula a inclinação, dessa forma a câmera é posicionada na peça de suporte (b) e parafusada. Enquanto isso a primeira peça (a) é fixada no metalon a por meio de 3 parafusos para garantir estabilidade durante a condução do veículo. Dessa forma permite aos integrantes do projeto realizarem alterações sobre o posicionamento da câmera para que os testes de algoritmos possam ser realizados com diferentes campos de visão.

Figura 24 – *Assembly* das peças de suporte da câmera



Fonte: O autor

3.1.2 Instalação do monitor e Zed Box

A câmera Zed 2i, apresenta recursos de processamento já inseridos, porém, para acessá-los é preciso de uma *Graphics Processing Unit* (GPU). Dessa forma, o projeto conta com a Zed Box, uma plataforma compacta, desenvolvida para processar aplicações de *Artificial Intelligence of Things* (AIoI) em ambientes móveis. A escolha desse *hardware* deu-se por diversos motivos, mas principalmente pela integração que possui com a câmera, uma vez que ambas são da mesma fabricante, minimizando os erros de integração.

Em termos estruturais, o fator decisivo para a utilização desse *hardware* no projeto foi devido ao seu design compacto. A arquitetura da Zed Box é bastante simples, representado como o próprio nome diz uma "caixa", formada por uma carcaça de alumínio no formato (109 x 92 x 52 mm). Na superfície superior é possível verificar um design mais sofisticado, pequenas placas inseridas paralelamente umas as outras de 10cm de altura. Uma vez que a Zed Box não apresenta ventoinha para resfriar seus componentes, essas placas funcionam como um grande dissipador de calor, atuando para manter o sistema dentro de uma temperatura aceitável.

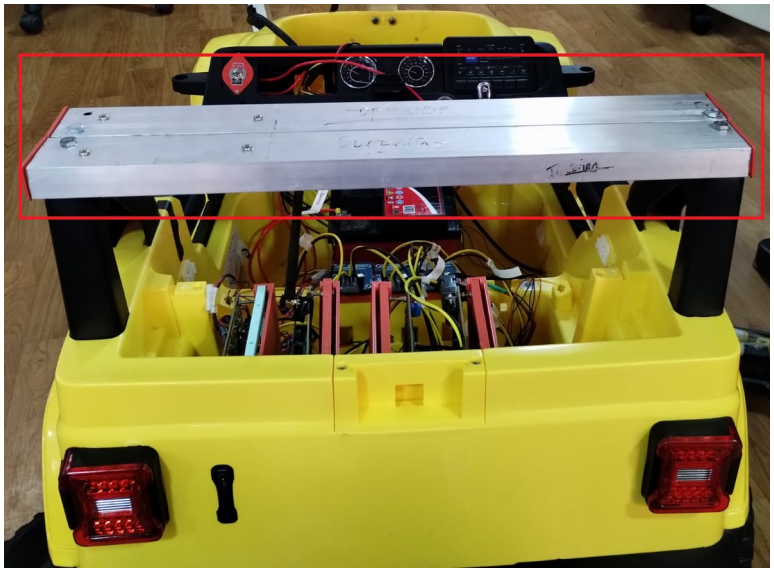
Figura 25 – Zed Box - Stereolabs

Fonte: Stereolabs Docs

Como os algoritmos desenvolvidos são voltados ao processamento de imagens, a presença de um monitor se torna indispensável. Durante todo o processo de desenvolvimento, o monitor funcionará como o dispositivo de interação com o desenvolvedor, permitindo a visualização de inconsistência para que inicie o processo de *debug*, etapa de análise e correção de erros. Além disso, é por meio do monitor que podemos observar quais os dados captados pela câmera e verificar a saída desses dados após o processamento pelos algoritmos. Dessa forma, o monitor se torna indispensável para o desenvolvimento do projeto, uma vez que sua presença é necessária para que sejam realizados tetes.

Tendo em vista que perfurar a base do monitor poderia inutilizar sua utilização em outras atividades que não sejam acopladas ao veículo de pequena escala. Optou-se por desenvolver uma única base para acoplar esses componentes, a Zed Box e o monitor. Produzida a partir da conexão paralela de 2 metalon de perfil 50x25mm parafusadas na traseira do veículo e a impressão de um calço para encaixar nas pontas do metalon por questões de segurança.

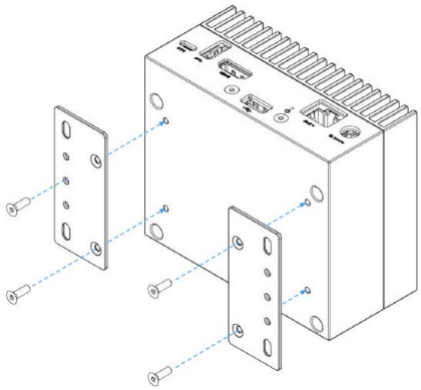
Figura 26 – Base do monitor e da Zed Box



Fonte: O autor

Com as alterações feitas no veículo, foi retirada a base do monitor e marcadas no metalon as devidas medidas de onde ficaria posicionado. Quanto a Zed Box, quando este dispositivo foi comprado, juntamente com o *hardware* está incluído uma pequena peça que se encaixa na parte inferior da Zed e permite parafusá-la. Dessa maneira, após todos as medidas serem demarcados e furadas, se tornou possível realizar testes de campo, uma vez que toda a estrutura energética do veículo já foi desenvolvida anteriormente.

Figura 27 – Instalaçao da Zed Box



Fonte: Stereolabs Docs

3.1.3 Faixas de rolagem

Durante o processo de codificação dos algoritmos surge a necessidade de um ambiente preparado para realizar os testes iniciais, uma vez que o objetivo era explorar a

captação de imagens em tempo real da Zed. Dessa forma, após observado o comportamento de algoritmos de *Lane Detection* em vídeos e imagens gravados, torna-se necessário construir um ambiente no local de trabalho para inicialmente integrar os algoritmos com a captação de imagens da câmera e posteriormente calibrar e configurar os parâmetros que definem as respostas dos sensores.

O veículo apresenta 72 centímetros de distância entre a roda dianteira esquerda até a dianteira direita. Com o intuito de delimitarmos uma "margem"que o veículo deveria permanecer dentro, foi definida uma distância segura de 1.5m entre uma faixa a outra. Esse valor garante que o veículo tenha o espaço necessário para realizar manobras de curva sem que haja a necessidade de sair do trajeto. Como o objetivo do projeto é aplicar os resultados obtidos, a partir de um modelo de pequena escala, em cenários reais, a distância entre as faixas inseridas no laboratório deve ser proporcional ao espaço que um veículo tem disponível em uma pista.

O Código de Transito Brasileiro (CTB) não determina um valor específico para a distância entre as faixas, entretanto, a largura de faixa de rolamento para veículos à 80 km/h estão entre 3.2m e 4.0m. Considerando que a largura de um carro convencional varia a partir de 2.2m, e a largura do Jeep Wrangler é de 0.72m, por meio de proporção simples é possível determinar a distância que as faixas devem permanecer.

- LR: Largura de um veículo real aproximada (utilizado 2.5m)
- dF: Distância de uma faixa de rolamento a 80km/h (utilizado 3.5m)
- LJ: Largura do veículo de pequena escala
- dx: Distância entre as faixas para o veículo de pequena escala

$$\frac{dF}{LR} = \frac{dx}{LJ}\frac{3,5}{2,5} = \frac{dx}{0,72}1,4 = \frac{dx}{0,72}dx = 1,008 \tag{3.1}$$

Aproximando o valor de dx, temos que:

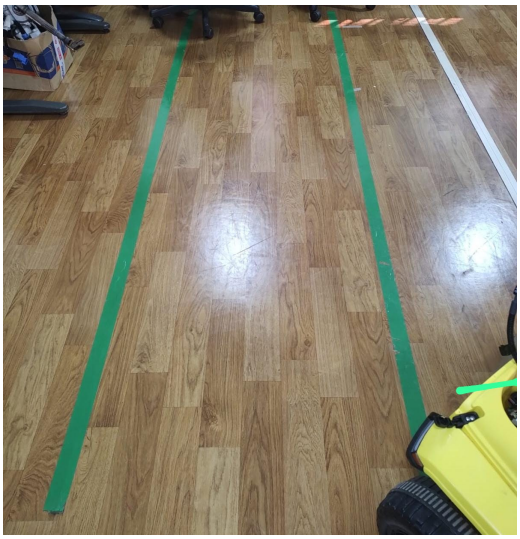
$$dx = 1 \tag{3.2}$$

Uma vez que a distância entra as faixas foi definida, o ambiente pode ser testado. Como explicado no processamento de imagens, as cores possuem papel fundamental durante a detecção de objetos. A primeira cor testada foi o branco, a partir de uma fita adesiva nas marcações do piso. Porém, o branco é facilmente confundido com outras características do piso e influenciado por variações de claridade. Dessa maneira, não era possível obter constância entre os resultados dos testes feitos em sob diferentes níveis de iluminação.

A partir de modificações no reconhecimento das cores no algoritmo, a segunda cor testada foi o amarelo. Do mesmo modo como a primeira, foi adquirido uma fita adesiva para a demarcação. Porém, a fita apresentava baixa "opacidade", ou seja, apresentava um grau de transparência muito alto o que contribuiu para descartarmos essa hipótese, uma vez que o código não conseguia reconhecer as pigmentações de amarelo, e apresentou resultado pior que a fita branca.

Para solucionar a questão da visibilidade, o projeto optou por utilizar fita adesiva de demarcação de solo na cor verde, com dimensões de 48mm x 30m. Essa escolha se baseou no fato de que essa fita possui cores mais intensas e é mais espessa do que outras opções disponíveis, dessa maneira foi possível obter os mesmos resultados no reconhecimento de imagens em diferentes luminosidades e horários do dia.

Figura 28 – Faixas de rodagem instaladas no ambiente de desenvolvimento



Fonte: O autor

3.2 Arquitetura

Essa seção exibe as questões arquiteturais que abrangem o escopo do projeto. O veículo em si já apresenta um modelo de arquitetura próprio, que já foram implementadas anteriormente, como o processo de comunicação das ECUs. Assim, este capítulo irá demonstrar apenas os modelos de estrutura adotados durante o desenvolvimento dos algoritmos de *Lane Keeping* e *Lane Detection*.

3.2.1 Arquitetura Cuda NVIDIA

Por meio da programação paralela, é possível obter melhor desempenho em diversas áreas do desenvolvimento de software, uma vez que permite a execução de diferentes

atividades simultaneamente. A partir dessa premissa, a NVIDIA desenvolveu o *Compute Unified Device Architecture* (CUDA), uma plataforma que permite dividir *tasks* complexas em seus núcleos do processador. Com essa arquitetura, atividades computacionais intensivas podem ser "quebradas"em pedaços menores para serem processadas simultaneamente na GPU.

A Zed Box está equipada com uma placa NVIDIA® Jetson Orin™ Xavier que está estruturada para a arquitetura CUDA. Desenvolvida para suportar as requisições de processamento em aplicações de visão computacional em tempo real, essas placas garantem alto poder de processamento mantendo a eficiência energética. Dessa maneira, a Stereolabs consegue disponibilizar um produto com recursos avançados de visão computacional combinados com suas câmeras para diferentes tipos de aplicações.

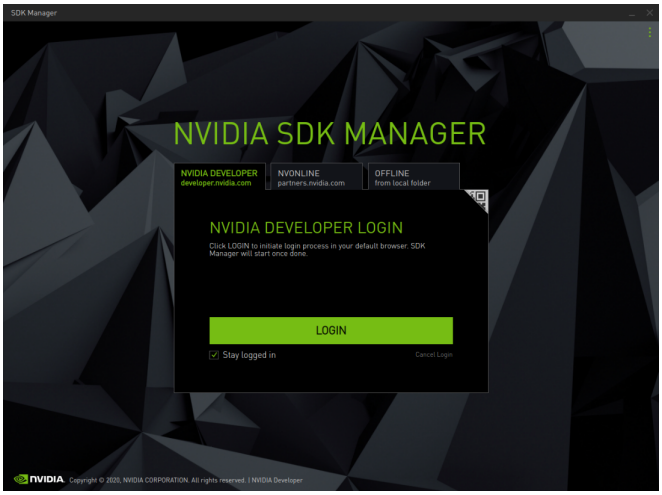
Figura 29 – Integração da NVIDIA® Jetson Orin™ Xavier na Zed Box

Fonte: Stereolabs

Porém, por mais que essa combinação de *hardwares* se assemelhem a um computador convencional, existem algumas diferenças referentes a sua configuração e seu processo de instalação. A principal diferença está na necessidade de um computador ou notebook externo para configurar o ambiente, independente do sistema operacional. Para realizar a primeira configuração da Zed Box, realizar atualizações no sistema operacional ou resetar o equipamento para as configurações originais de fábrica, será necessário a instalação do SDK Manager, disponibilizado pela Stereolabs.

O SDK Manager será a ferramenta a disposição do usuário para acessar as configurações da Zed e deverá estar instalada no computador auxiliar. Outro fator importante de mencionar, é que o sistema operacional que será instalado deve já estar presente no computador antes de iniciar os procedimentos com o SDK. Dessa forma, a partir de um cabo OTG (mini USB) conectado na Zed Box e no computador auxiliar será possível realizar todas as diferentes instalações e configurações de sistemas operacionais.

Figura 30 – Interface SDK Maneger



Fonte: Nvidia Docs

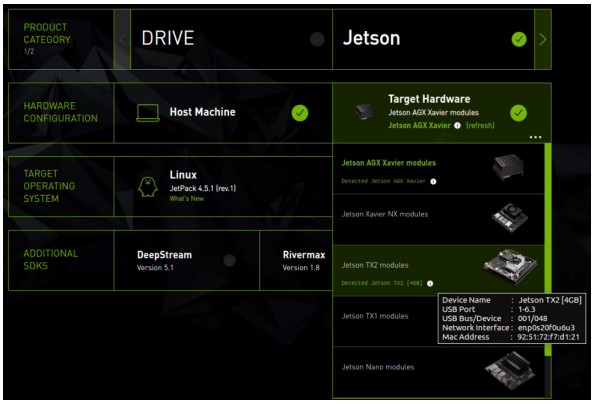
O processo de configuração pode ser trabalhoso, principalmente se o objetivo for instalar um sistema operacional diferente do utilizado no computador auxiliar. A alternativa encontrada para contornar essa questão é utilizar o *Windows Subsystem for Linux* (WSL), uma tecnologia desenvolvida pela Microsoft que permite aos desenvolvedores utilizarem chamadas do ambiente Linux dentro do sistema operacional Windows. Porém, algumas incompatibilidades devem ser corrigidas, uma vez que o WSL não terá acesso as saídas USB do computador.

Uma maneira de garantir que o WSL reconheça os dispositivos conectados por portas USB é atualizando os drivers *Communication* (COM), responsáveis por criar as portas virtuais de comunicação serial entre o computador e dispositivos externos. Esses drivers podem ser atualizados manualmente ou por meio de programas terceiros para verificar e instalar a versão mais compatível. Logo após isso, as conexões do WSL devem ser habilitadas por um prompt de comando Windows, de maneira que ao finalizar os procedimentos corretamente será exibido uma lista das conexões existentes ao digitar o comando:

```
$ lsusb (WSL)
```

A Stereolabs disponibiliza diferentes arquiteturas para cada modelo de placa Nvidia utilizada no sistema embarcado, porém, seus processos de instalação podem ser encontrados em seu site. Cada processo apresenta sua particularidade, que são realizados a partir do SDK Manager. As últimas etapas consistem em carregar as configurações feitas no computador auxiliar para a Zed Box, podendo gastar algumas horas até a instalação ser realizada com sucesso.

Figura 31 – Opções de configuração de diferentes dispositivos pelo SDK Manager



Fonte: Nvidia Docs

3.2.2 Arquitetura de software LKA

Todo projeto conta com uma arquitetura nos estágios do seu desenvolvimento, que devido ao controle das demandas e especificações está suscetível a alterações. Em ambientes de desenvolvimento pequenos, se torna comum criar uma arquitetura provisória para validar as primeiras funções desenvolvidas. Dessa forma, os integrantes podem se concentrar em realizar pesquisas e implementações que permitem vislumbrar pequenos resultados acerca do objetivo da aplicação. De maneira semelhante, serão desenvolvidos os algoritmos de *Lane Detection* partindo de uma arquitetura simplista, voltada a execuções individuais de *scripts* desenvolvidos prioritariamente em *Python* e C++, uma vez que essa linguagens são as recomendadas na documentação de algoritmos da Zed Box.

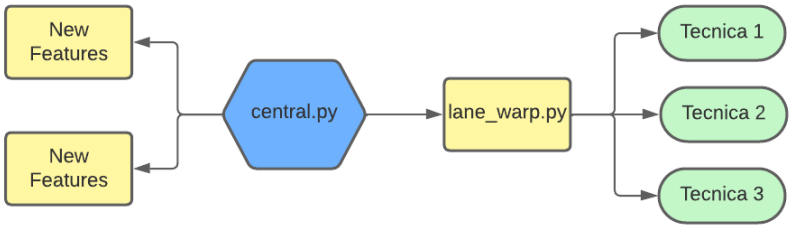
Uma vez que os dados coletados a partir das saídas do algoritmo devem ser repassados para os demais componentes do veículo, torna-se necessário evoluir a arquitetura inicial. Afinal, a arquitetura serve de base para as novas demandas de desenvolvimento no código, e é partir dela que se torna possível garantir heurísticas fundamentais em um sistema de software, como por exemplo a confiabilidade. Além disso, uma arquitetura bem desenvolvida auxilia na manutenibilidade do código, dessa forma, quaisquer alterações ou correções que necessitem ser feitas, poderão ser melhor encontradas e inseridas no projeto.

Outro fator a ser considerado durante a definição da próxima arquitetura, é que o projeto continuará explorando novas áreas, buscando implementar cada vez mais funções ADAS. Assim, a arquitetura não pode ser individualizada para o desenvolvimento dos algoritmos de LKA, deve ser produzida de maneira que novas funções de software que utilizarem o processamento da Zed Box sejam facilmente incluídas na estrutura.

Dessa maneira, a arquitetura será desenvolvida a partir de um *script* central responsável por organizar e controlar as demais funções que serão desenvolvidas no projeto. O arquivo responsável por realizar todas as demandas do *Lane Detection* será definido

como *LaneWarp.py*, onde ficará responsável por configurar os dados capturados pela câmera e organizar como os filtros de processamento serão executados. Dessa forma, torna-se possível garantir a integridade do código e organização dos componentes necessários para alocar no veículo de pequena escala.

Figura 32 – Fluxograma da proposta de arquitetura de software



Fonte: O autor

3.2.3 Arquitetura de componentes

Para a implementação de um sistema de *Lane Keeping* funcional, devemos garantir que os parâmetros de saída das funções sejam incorporadas no sistema de comunicação do veículo, estruturado pelas *Engine Control Units* (ECUs) que comunicam entre si por meio de uma rede *Controller Area Network* (CAN), um protocolo de comunicação serial muito utilizado na troca de mensagens entre componentes em veículos automotivos.

Assim como na utilizado no desenvolvimento do software, a ideia proposta para a arquitetura dos componentes também deriva da premissa de uma arquitetura inicial simples, porém que seja capaz de realizar seu objetivo. A passagem dos dados captados pela câmera até a Zed Box é um processo majoritariamente automático, dessa forma partiremos desse ponto para criar a primeira proposta. Uma vez que a Zed não consegue realizar comunicação CAN com as ECUs será necessário a inserção de um componente capaz de receber os dados da Zed por algum protocolo de comunicação e traduzi-los para serem inseridos na rede CAN.

Como alternativa para o problema exposto, serão utilizado pequenos módulos de processamento embarcados para realizar essa tarefa de "ponte"entre os componentes do veículo, tais como uma Esp-32 ou um Arduíno. O módulo é responsável por receber dados de um protocolo específico e transmiti-los por meio de outro protocolo. Essa tarefa é bastante flexível, permitindo a utilização de diversos componentes para sua realização. Dentre as opções, será utilizado um Arduíno Nano, devido ao seu processamento razoável e a grande disponibilidade dessas peças presentes no laboratório.

Uma vez que a Zed funciona basicamente como um computador convencional, é possível conectar o Arduíno em uma de suas portas USBs para realizar a comunicação se-

rial. Isso permite que dados compactados em formato Json sejam enviados para o Arduíno através de funções que serão chamadas sempre que alguma informação for requisitada na central.

Figura 33 – Fluxograma da proposta de arquitetura de componentes inicial

Fonte: O autor

A arquitetura acima, serve unicamente para validar a chegada dos valores processados pela Zed no sistemas de ECUs do veículo, não exibindo uma função prática para o objetivo da aplicação, uma vez que esses valores logo serão descartados no sistema. Como arquitetura final, o Arduíno será substituído por um controlador PID, este é responsável por receber as variações do erro e suas derivadas para definir a melhor resposta para a direção do veículo. Existem diferentes modelos de controladores, que se aperfeiçoam cada vez mais quanto a esterção do volante durante a condução do veículo, porém, quanto mais elaborado o modelo, mais parâmetros devem ser processados sobre as imagens capturadas.

Diante disso, utilizaremos de início um controlador simples, que requisita como entrada a distância que o veículo se encontra do meio da faixa. Esse erro (Er), pode ser mapeado a partir de uma altura pré definida para 2 pontos sobre a faixa de rolagem. Dessa maneira, o valor médio entre os 2 pontos será o centro (dc), e uma vez que já conhecemos a posição do veículo (pv) será possível verificar a distância em centímetros que este se encontra distante do centro.

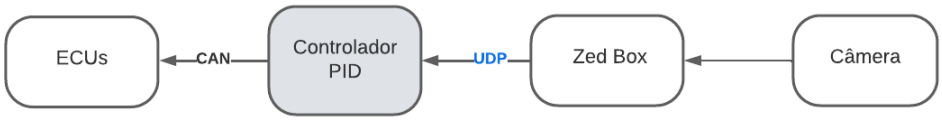
$$Er = pv - dc \tag{3.3}$$

Dessa forma, se o valor do erro for positiva implica que o veículo está mais a direita do centro, enquanto o resultado negativo implica estar mais a esquerda. A partir desses valores fornecidos ao controlador, torna-se possível determinar o ângulo de esterção do volante para contatar a ECU responsável pela direção.

O processo de comunicação também deverá ser alterado, uma vez que não é possível realizar comunicação serial entre o controlador e a Zed Box. Além disso, a comunicação serial não é recomendada para realizar rápidas trocas de mensagens, uma vez que apresentaria baixa performance. Logo, será realizada a comunicação por protocolo *User Datagram*

Protocol (UDP), comunicação por rede que utiliza a camada de transporte sem que necessite estabelecer uma conexão. A partir de um cabo Ethernet conectado aos dispositivos, basta definir o número de IP destino (controlador) e definir uma porta disponível para iniciar o tráfego de dados de maneira muito mais eficiente.

Figura 34 – Fluxograma da proposta de arquitetura de componentes final



Fonte: O autor

O sistema operacional escolhido para o desenvolvimento do projeto será o Linux, a partir da versão 20.04 do Ubuntu, uma vez que este apresenta maior facilidade na instalação de pacotes e dependências do projeto, já que seu ambiente de linha de comando é completo e mais flexível. Além disso, apresenta melhor compatibilidade para utilização do SDK, que é o responsável por disponibilizar aos usuários uma grande quantidade de recursos computacionais.

3.3 Processamento de Imagens

Esta etapa consiste nas técnicas que devem ser aplicadas para o processamento das imagens capturadas pela câmera, como explicado anteriormente, essa etapa consiste na manipulação de imagens digitais. Depois da imagem ser transformada em uma matriz de números, torna-se possível para os computadores interpretarem e manipularem esses dados. Dentre as diversas técnicas diferentes, o processamento de imagens pode ser classificado de acordo com o objetivo de cada técnica, sendo elas: melhoria de imagem, restauração de imagem e análise e compreensão de imagens.

A melhoria de imagem consiste principalmente na utilização de técnicas heurísticas, ou seja, são técnicas que não estão restritas as regras matemáticas de algoritmos mais sofisticados. São definidas a partir da experiência e a prática do usuário para melhorar a qualidade visual da imagem, para que um observador humano consiga extrair as informações desejadas a partir dela. Exemplos de técnicas heurísticas são os ajustes manuais de brilho, contraste e nitidez em uma imagem.

Enquanto isso, a restauração de imagem se preocupa em recuperar informações corrompidas. Borrões, ruídos e desfoque estão entre os diferentes aspectos que atrapalham

a classificação de objetos. Dessa forma, diferentes algoritmos e abordagens são utilizados para garantir uma imagem mais "limpa". A análise e compreensão de imagens são as últimas técnicas do processamento, e é onde ocorre a extração automática das informações desejadas. Estas são as técnicas de Segmentação apresentadas anteriormente.

Para auxiliar no desenvolvimento, a Stereolabs desenvolveu um segundo SDK, porém voltado para as questões de processamento de imagens e que deve ser instalado no ambiente de desenvolvimento. Esse foi nomeado como Zed SDK, projetada para trabalhar em conjunto com a câmera, para auxiliar desenvolvedores na criação das funcionalidades desejadas. Dessa forma, o projeto se torna mais livre para garantir seu objetivo, já que não será necessário desenvolver várias das funções de processamento de imagens, ou buscar bibliotecas online. Assim, os integrantes podem se concentrar em como organizar e configurar essas funcionalidades para realizar a detecção das faixas de rolagem.

Dessa maneira, serão apresentadas neste capítulo a integração do SDK para realizar as principais técnicas de processamento de imagens que serão aplicadas no desenvolvimento do projeto.

3.3.1 Melhoria de imagem

É possível encontrar na documentação oficial da Zed 2i, os arquivos dos procedimentos de instalação dos componentes juntamente com uma série de *scripts* de exemplos, desde simples requisições para imprimir no terminal o número de série da câmera até complicados processamento de imagens. Ainda dentro do escopo de exemplos simples, a partir da correta configuração do ambiente de trabalho e instalação do ZED SDK, será possível executar o primeiro código, onde irá retornar as informações visuais da câmera. Encontrado na aba de *"Tutorials"* no site oficial do componente, estará disponibilizado o arquivo *Camera Control* nas linguagens de C++ ou *Python*.

A partir desse arquivo, torna-se possível verificar como as técnicas heurísticas de melhoria de imagem podem ser implementadas no projeto, uma vez que demonstram como modificar as configurações da câmera para adaptá-la ao objetivo final. Assim, ela permite definir parâmetros como exposição, ganho, contraste, nitidez, brilho e fps para a aquisição da imagem.

Tudo isso, torna-se possível pela declaração de um objeto do tipo *pyzed* no código, onde este ficará responsável por realizar a configuração desses parâmetros. No trecho de código abaixo, definimos um objeto *pyzed* com o nome *sl*, e a partir dele se torna possível acessar os argumentos do processamento de vídeo. No caso abaixo, passamos -1 como entrada para a função, assim ela utilizará os valores definidos como padrão da câmera.

```
import pyzed.sl as sl
```

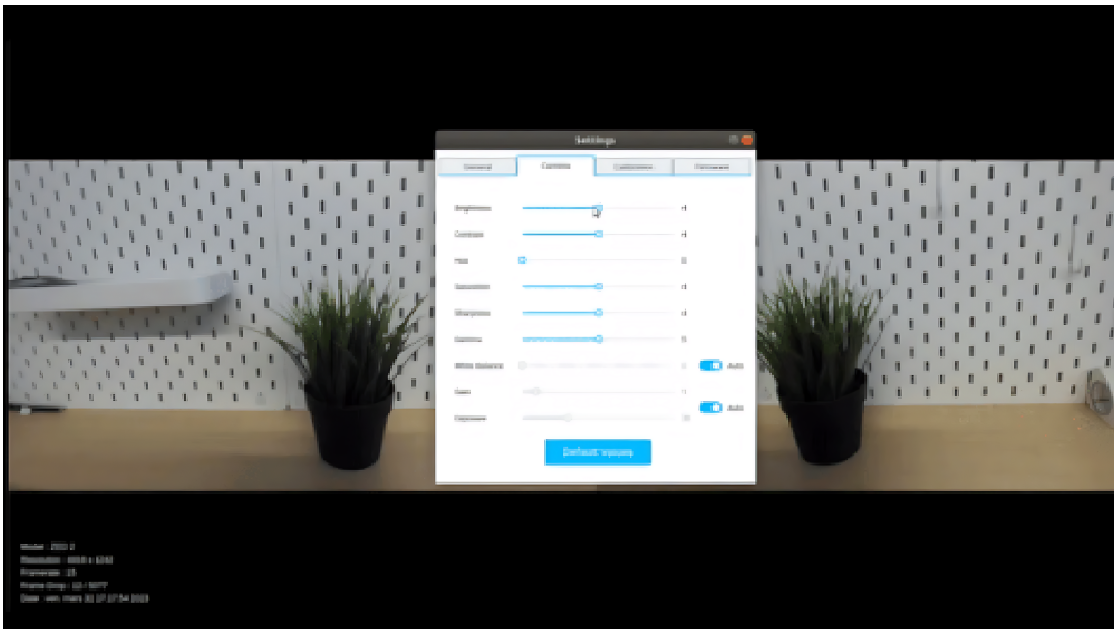
```
camera_settings = sl.VIDEO_SETTINGS.BRIGHTNESS
def main(){

        init = sl.InitParameters()
        cam = sl.Camera()

        cam.set_camera_settings(sl.VIDEO_SETTINGS.BRIGHTNESS, -1)
        cam.set_camera_settings(sl.VIDEO_SETTINGS.CONTRAST, -1)
        cam.set_camera_settings(sl.VIDEO_SETTINGS.HUE, -1)
        cam.set_camera_settings(sl.VIDEO_SETTINGS.SATURATION, -1)
        cam.set_camera_settings(sl.VIDEO_SETTINGS.SHARPNESS, -1)
        cam.set_camera_settings(sl.VIDEO_SETTINGS.GAIN, -1)
        cam.set_camera_settings(sl.VIDEO_SETTINGS.EXPOSURE, -1)
```

Além disso, o código completo demonstra ao usuário como alterar esses parâmetros em tempo de execução. Uma vez que executado, irá ser disponibilizado uma API de interface exibindo a imagem que está sendo capturada pela câmera, e uma barra de filtros com os valores atuais dessas características, garantindo controle dos parâmetros de qualidade de imagem em tempo real ao usuário.

Figura 35 – Interface para alterar parâmetros de melhoria de imagem



Fonte: Stereolabs

3.3.2 Recuperação, Análise e Compreensão de imagens

O desenvolvimento da câmera Zed 2i foi voltado para a captura de imagens em alta resolução. Dessa forma, ela apresenta diversos fatores que garantem a qualidade da imagem. Dentre eles, a combinação de dados entre as duas lentes polarizadas, os sensores de profundidade e os algoritmos de processamento de imagens. Assim, a maioria das técnicas de recuperação são utilizadas dentro do próprio ambiente de processamento da câmera, dessa forma, serão exibidas juntamente com as demais técnicas.

Para as técnicas de Análise e Compreensão, iremos utilizar novamente a ajuda do ZED SDK para manipular os dados carregados pela câmera dentro das GPUs de processamento. A linguagem utilizada para o desenvolvimento do projeto será o *Python* na sua versão 3.7, devido a sua versatilidade de programação. Mesmo que essa linguagem seja relativamente mais lentas que outras linguagens, por exemplo o C++, ela se torna vantajosa ao projeto por disponibilizar diferentes recursos e funções que já estão estruturadas na linguagem. Além disso, a quantidade de materiais de apoio e fóruns online para ajudar na resolução de problemas é muito maior.

Outra vantagem de apresentar uma comunidade ativa, é devido as diversas bibliotecas disponíveis, sendo uma delas o *Open Source Computer Vision Library* (OpenCV) uma biblioteca voltada para o desenvolvimento de softwares nas áreas de visão computacional e aprendizado de máquinas. Desenvolvida pela Apache, esta segue as diretrizes do *Open Source* ou código aberto, que disponibiliza ao público uma diversidade de recursos para serem implementados em seus próprios códigos. Assim, a proposta do projeto, é combinar os algoritmos otimizados disponibilizados pela opencv-python 4.8, biblioteca da OpenCV para a linguagem *Python*, com as funções de captura e manipulação de dados do SDk para desenvolver o primeiro sistema de *Lane Detection*.

Dessa forma, podemos dividir o processo de desenvolvimento nas seguintes etapas:

- Implementar o *Bird's Eye* com os dados capturados pela câmera;
- Converter os *pixels* da imagem resultante em binário e aplicar as técnicas de restauração;
- Utilizar algoritmos de *Hough Lines* para encontrar as faixa de rodagem na imagem;
- Calcular a distância do centro das faixas até a posição do veículo.

Uma vez que essas etapas foram realizadas com sucesso a implementação do *Lane Detection* estará finalizada, onde os valores resultantes do processamento de imagens serão enviados para os outros componentes, como demostrado na arquitetura, para então ser realizado o procedimento de *Lane Keeping*. Porém, apenas com a distância do veículo do centro da pista, não será o suficiente para manter o veículo dentro das faixas de rolagem

diante de um percurso de curvas. Dessa maneira, para implementarmos o sistema de LKA, será necessário desenvolver um algoritmo para detectar uma curva e calcular qual o seu ângulo de curvatura.

A partir do mapeamento dos *pixels* já realizado, torna-se possível converter a distância entre as faixas de rodagem da direita e da esquerda para metros. A partir da visão superior proporcionada pela técnica de *Bird's Eye* será possível visualizar o início de uma trajetória circular, e dessa forma, calcular o raio da curva. Com essa informação sendo repassada juntamente com o erro para o controlador, na segunda proposta de arquitetura, será possível obter um sistema de *Lane Keeping* completamente operacional.

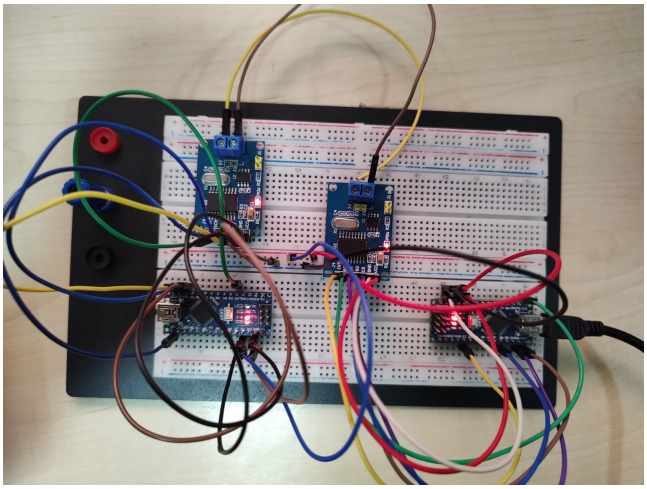
3.4 Comunicação de sistemas

3.4.1 Integração de sistemas por comunicação serial

Para o desenvolvimento dessa etapa utilizou-se um Arduíno secundário configurado com os códigos da ECU de direção, dessa forma se tornou possível realizar os primeiros testes para verificação do tráfego de informações dos diferentes componentes. O primeiro ambiente desenvolvido utiliza um *script* em *Python* a partir de um *Notebook* para verificar se a conexão serial conseguia suportar o envio de dados a 0.2 segundos, velocidade utilizada na rede CAN que as ECUs se comunicam. Dessa forma, o *script* verifica a comunicação serial com o Arduíno e em seguida envia diferentes valores no intervalo determinado.

A comunicação serial atendeu com sucesso à necessidade de transmitir informações neste estágio inicial, assegurando a integridade do envio e recebimento de dados. Como parte do aprimoramento do ambiente de teste, foram incorporados dois dispositivos MCP2515 CAN, permitindo a comunicação CAN, além de um Arduino UNO. Utilizou-se uma *protoboard* para montar o sistema, onde a entrada serial também foi utilizada para energizar todo o sistema, como ilustrado na figura 36.

Figura 36 – Sistema de testes para a comunicação serial

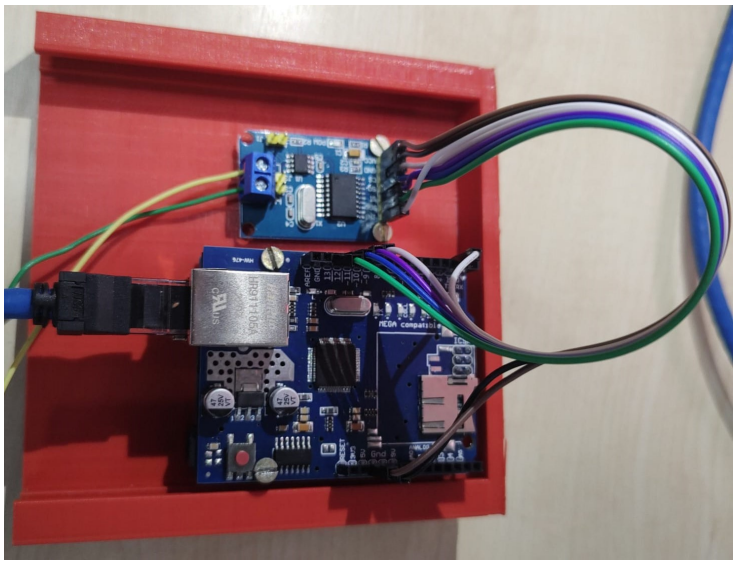


Fonte: O autor

3.4.2 Integração de sistemas por comunicação UDP

Para viabilizar a comunicação dos sistemas desta fase, foi concebida uma ECU de dimensões reduzidas para a acomodação dos componentes. O módulo Ethernet *shield* foi acoplado ao Arduino nos pinos pertinentes e posteriormente fixado à superfície da ECU, adjacente ao módulo MCP2515 CAN. Dessa forma foi possível ter um componente com fácil acesso a rede CAN do sistema e posteriormente foi incorporada ao veículo como uma ECU propriamente.

Figura 37 – ECU de comunicação UDP



Fonte: O autor

A utilização da comunicação por UDP destaca-se pela sua baixa latência, o que

permite que os dados sejam transmitidos e recebidos de maneira rápida e eficiente em uma comunicação de ponta a ponta. Dessa forma, os dados transmitidos pela Zed Box via *socket*, mecanismo para transmissão e recepção de dados em rede, são descompactados dentro do Arduíno sem ultrapassar o intervalo de tempo esperado.

Uma vez que esses dados estão disponíveis para uso no Arduíno, basta alocar os valores em um vetor de 8 *bytes* com um ID em comum para serem enviados a rede do veículo de pequena escala pelo protocolo CAN. Uma vez que a ECU de direção foi configurada para identificar o ID selecionado e dessa forma coordenar o ângulo de exterção do volante. Os fundamentos teóricos e práticos apresentados neste capítulo fornecem a base necessária para a análise e discussão dos resultados, onde serão avaliados os impactos da implementação dessas estratégias no sistema proposto.

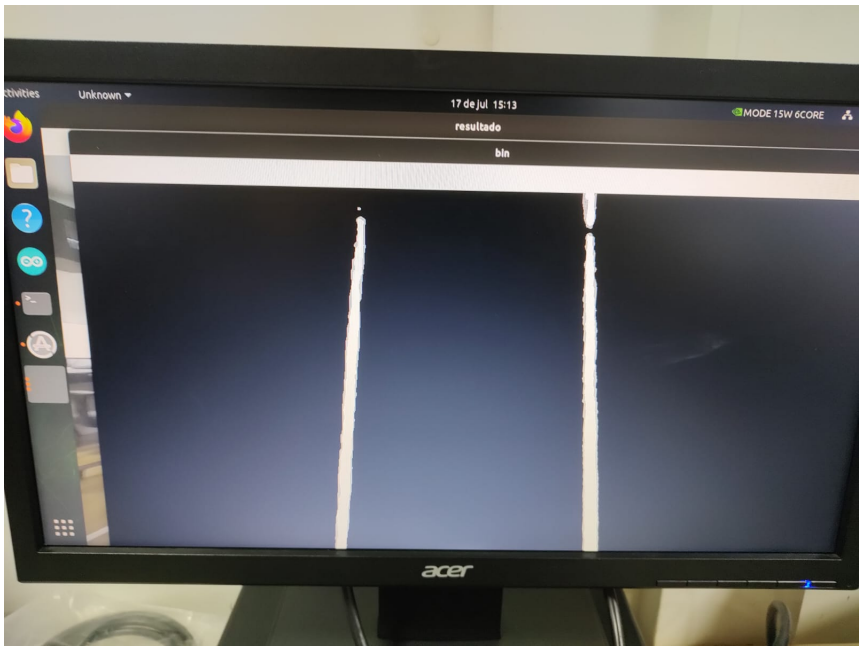
4 Resultados

4.1 Detecção de faixas

A partir das ideias propostas na metodologia, foram possíveis atingir resultados satisfatórios durante essa etapa do projeto, sendo a primeira delas a integração de todos os componentes no veículo de pequena escala. Uma vez que o ambiente de desenvolvimento na Zed Box estava configurado para trabalhar em conjunto com os dados da câmera, avançamos para a instalação no veículo, que permitiu realizar os primeiros testes energéticos acerca da quantidade de corrente que é necessária para manter todo o sistema do Jeep funcionando, já que a própria Zed Box necessita de altas correntes para o processamento.

Já no ambiente de desenvolvimento de software, foi possível combinar as técnicas de processamento de imagem para selecionarmos a região de interesse (ROI) das imagens, e a partir dos algoritmos de detecção de linhas, criar diferentes máscaras, ferramenta para alterar ou manipular dados sem alterar a estrutura original. Desse modo, torna-se possível acompanhar as etapas do processamento de faixas em tempo real, já que a primeira máscara criada tem o objetivo de mostrar as linhas encontradas pelos algoritmos em uma imagem binarizada (figura 38).

Figura 38 – Máscara de faixas detectadas em uma imagem binarizada

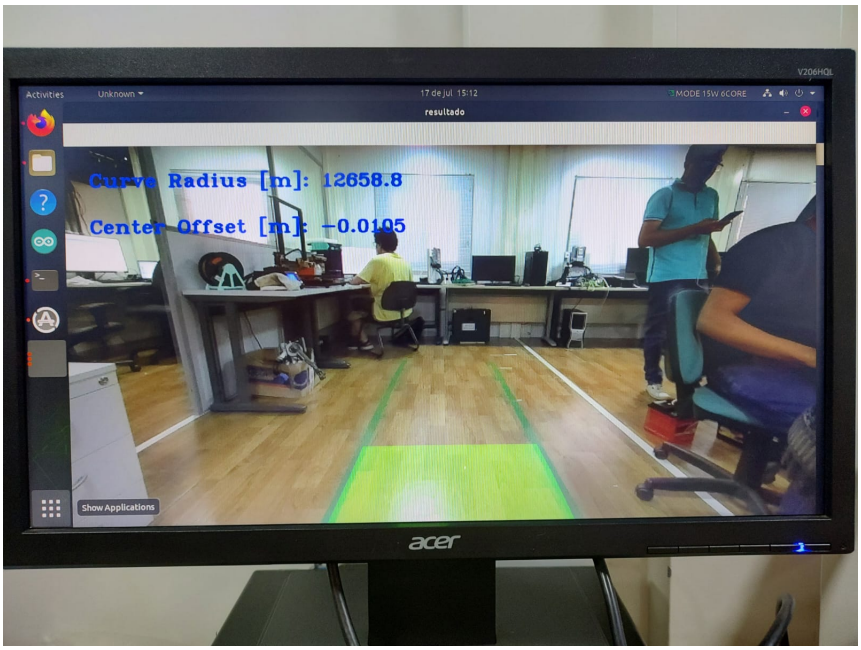


Fonte: O autor

Outra máscara que o projeto utiliza, é a responsável por sobrescrever os dados

originais a partir de uma camada verde que demonstra a região que o veículo deve permanecer. Para isso, utiliza das posições dos *pixels* da primeira máscara (Figura 38) de maneira a criar 2 vetores de coordenadas, relativas a primeira e a segunda faixa. Assim, a partir da altura dos *pixels* mais altos e mais baixos, esse pontos são conectados por novas linhas, formando um trapézio, que será os limites dessa máscara (figura 39).

Figura 39 – Máscara da região de interesse



Fonte: O autor

No estado atual do projeto, os algoritmos de detecção de faixas utilizam um simples cálculo de subtração para verificar a distância que o centro da câmera está da margem direita. Se o valor da variável *centerOffset* for negativo o veículo se encontra a esquerda do centro da pista, e em caso positivo está posicionado a direita. Assim, foi possível realizar testes de campo, para verificar o comportamento do sistema como um todo e mensurar questões energéticas e de posicionamento de componentes.

Figura 40 – Sistema de detecção de faixas instalado no veículo

Fonte: O autor

4.2 Integração de sistemas

4.2.1 Comunicação Serial

A partir de uma base confiável dos valores retornados das função de LKA, se torna possível prosseguir para as seguintes etapas, que envolvem o desenvolvimento de arquiteturas capazes de realizar os diferentes formatos de comunicação, UDP e Serial.

Assim, foi possível realizar um processo de validação eficaz, uma vez que foram verificados os dados recebidos após a inserção na rede CAN e assim, foi possível perceber que a partir de vetores maiores, usados para armazenar os números que foram enviados na comunicação serial, obteve-se erros de integridade e atrasos nas mensagens quando a comunicação fica ativa por alguns minutos.

A imagem abaixo (figura 41) representa as respostas obtidas a partir do cenário descrito da comunicação serial, onde eram enviados valores crescentes no intervalo de 0 até 25, esses valores foram adotados devido ao fato da ECU de direção utilizar o intervalo de 0 à 50 para determinar a angulação do volante. Sendo assim, é possível visualizar o "ERRO", representando a angulação que o volante está do centro (25°). A partir da imagem é possível verificar que alguns valores estão repetidos, isso acontece quando a variável

de armazenamento recebida no Arduíno não é atualizada dentro do tempo esperado e reenviada pela rede CAN, representando o atraso na comunicação serial.

Figura 41 – Dados recebidos na rede CAN pela comunicação Serial

```
ERRO: -19.00 u: -1.28 = P:-0.28 + I:-1.00 + D: 0.00 PWM: 255 t0: 78 ang: 25 set: 7
ERRO: -18.00 u: -1.27 = P:-0.27 + I:-1.00 + D: 0.00 PWM: 255 t0: 77 ang: 25 set: 8
ERRO: -17.00 u: -1.25 = P:-0.25 + I:-1.00 + D: 0.00 PWM: 255 t0: 77 ang: 25 set: 9
ERRO: -16.00 u: -1.24 = P:-0.24 + I:-1.00 + D: 0.00 PWM: 255 t0: 78 ang: 25 set: 9
ERRO: -16.00 u: -1.24 = P:-0.24 + I:-1.00 + D: 0.00 PWM: 255 t0: 77 ang: 25 set: 10
ERRO: -15.00 u: -1.22 = P:-0.22 + I:-1.00 + D: 0.00 PWM: 255 t0: 78 ang: 25 set: 11
ERRO: -14.00 u: -1.21 = P:-0.21 + I:-1.00 + D: 0.00 PWM: 255 t0: 78 ang: 25 set: 12
ERRO: -13.00 u: -1.19 = P:-0.19 + I:-1.00 + D: 0.00 PWM: 255 t0: 78 ang: 25 set: 13
ERRO: -12.00 u: -1.18 = P:-0.18 + I:-1.00 + D: 0.00 PWM: 255 t0: 78 ang: 25 set: 13
ERRO: -12.00 u: -1.18 = P:-0.18 + I:-1.00 + D: 0.00 PWM: 255 t0: 76 ang: 25 set: 14
ERRO: -11.00 u: -1.16 = P:-0.16 + I:-1.00 + D: 0.00 PWM: 255 t0: 77 ang: 25 set: 15
ERRO: -10.00 u: -0.61 = P:-0.10 + I:-0.51 + D: 0.00 PWM: 155 t0: 78 ang: 25 set: 16
ERRO: -9.00 u: -0.60 = P:-0.09 + I:-0.51 + D: 0.00 PWM: 154 t0: 78 ang: 25 set: 17
ERRO: -8.00 u: -0.60 = P:-0.08 + I:-0.52 + D: 0.00 PWM: 153 t0: 78 ang: 25 set: 17
ERRO: -8.00 u: -0.61 = P:-0.08 + I:-0.53 + D: 0.00 PWM: 155 t0: 77 ang: 25 set: 18
ERRO: -7.00 u: -0.60 = P:-0.07 + I:-0.53 + D: 0.00 PWM: 154 t0: 78 ang: 25 set: 19
ERRO: -6.00 u: -0.60 = P:-0.06 + I:-0.54 + D: 0.00 PWM: 152 t0: 78 ang: 25 set: 20
ERRO: -5.00 u: -0.59 = P:-0.05 + I:-0.54 + D: 0.00 PWM: 151 t0: 77 ang: 25 set: 21
ERRO: -4.00 u: -0.58 = P:-0.04 + I:-0.54 + D: 0.00 PWM: 149 t0: 77 ang: 25 set: 21
ERRO: -4.00 u: -0.59 = P:-0.04 + I:-0.55 + D: 0.00 PWM: 150 t0: 77 ang: 25 set: 22
ERRO: -3.00 u: -0.58 = P:-0.03 + I:-0.55 + D: 0.00 PWM: 148 t0: 78 ang: 25 set: 23
ERRO: -2.00 u: -0.57 = P:-0.02 + I:-0.55 + D: 0.00 PWM: 146 t0: 78 ang: 25 set: 24
ERRO: -1.00 u: -0.56 = P:-0.01 + I:-0.55 + D: 0.00 PWM: 143 t0: 78 ang: 25 set: 24
ERRO: -1.00 u: -0.01 = P:-0.01 + I:-0.00 + D: 0.00 PWM: 3 t0: 77 ang: 25 set: 25
```

Fonte: O autor

Embora a comunicação serial não tenha atingido os requisitos essenciais de velocidade e integridade, ela desempenhou um papel fundamental no avanço do projeto. Através dela, foi possível criar as primeiras arquiteturas que serviram para validar o sistema como um todo e permitiu visualizar o percurso completo do tráfego de dados.

4.2.2 Comunicação UDP

Os resultados obtidos por meio da comunicação UDP demonstraram um desempenho notavelmente superior em comparação com a comunicação serial. Utilizando o mesmo código e intervalos direcionados ao sistema de direção, não foram observadas falhas ou duplicações, resultantes de atrasos inerentes ao protocolo de comunicação serial.

Figura 42 – Dados recebidos na rede CAN pela comunicação UDP

```
ERRO: −19.00 u: -1.28 = P:-0.28 + 1:-1.00 + D: 0.00 PWM: 255 tO: 78 ang: 25 set: 7
ERRO: -18.00 u: -1.27 = P:-0.27 + I1-1.00 + D: 0.00 PRM: 255 tO: 77 ang: 25 set: 8
ERRO: -17.00 u: -1.25 = P:-0.25 + I1-1.00 + D: 0.00 PRM: 255 tO: 77 ang: 25 set: 9
ERRO: -16.00 u: -1.24 = P:-0.24 + I:-1.00 + D: 0.00 PRM: 255 tO: 77 ang: 25 set: 10
ERRO: -15.00 u: -1.22 = P:-0.22 + I1-1.00 + D: 0.00 PRM: 255 tO: 78 ang: 25 set: 11
ERRO: -14.00 u: -1.21 = P:-0.21 + I1-1.00 + D: 0.00 PRM: 255 tO: 78 ang: 25 set: 12
ERRO: -13.00 u: -1.19 = P:-0.19 + I:-1.00 + D: 0.00 PRM: 255 tO: 78 ang: 25 set: 13
ERRO: -12.00 u: -1.18 = P:-0.18 + I:-1.00 + D: 0.00 PRM: 255 tO: 76 ang: 25 set: 14
ERRO: -11.00 u: -1.16 = P:-0.16 + I1-1.00 + D: 0.00 PRM: 255 tO: 77 ang: 25 set: 15
ERRO: -10.00 u: -0.61 = P:-0.10 + I:-0.51 + D: 0.00 PRM: 155 tO: 78 ang: 25 set: 16
ERRO: -9.00 u: -O.60 = P:-0.08 + I:-0.52 + Dz 0.00 PRM: 153 tO: 78 ang: 25 set: 17
ERRO: -8.00 u: -0.61 = P:-0.08 + I:-0.53 + Dz 0.00 PRM: 155 tO: 77 ang: 25 set: 18
ERRO: -7.00 u: -0.60 = P:-0.07 + I:-0.53 + Dz 0.00 PRM: 154 tO: 78 ang: 25 set: 19
ERRO: -6.00 u: -O.60 = P:-0.06 + I:-0.54 + Dz 0.00 PRM: 152 tO: 78 ang: 25 set: 20
ERRO: -5.00 u: -0.59 = P:-0.05 + I:-0.54 + Dz 0.00 PRM: 151 tO: 77 ang: 25 set: 21
ERRO: -4.00 u: -0.59 = P:-0.04 + I:-0.55 + Dz 0.00 PRM: 150 tO: 77 ang: 25 set: 22
ERRO: -3.00 u: -O.58 = P:-0.03 + I:-0.55 + Dz 0.00 PRM: 148 tO: 78 ang: 25 set: 23
ERRO: -2.00 u: -O.57 = P:-0.02 + I:-0.55 + Dz 0.00 PRM: 146 tO: 78 ang: 25 set: 24
ERRO: -1.00 u: -0.01 = P:-0.01 + I:-0.00 + Dz 0.00 PRM: 3 tO: 77 ang: 25 set: 25
```

Fonte: O autor

A ECU de direção está configurada para receber mensagens da rede CAN a cada intervalo de 0.2 segundos, valor utilizado no teste realizado acima. Ficou evidenciado que o protocolo UDP é capaz de operar de forma eficaz e confiável dentro desse intervalo, o que o tornou a escolha ideal para garantir a comunicação bem-sucedida para o projeto. Além disso, quando o protocolo de comunicação UDP é observado de forma isolada, sem estar incorporado a rede CAN, fica evidente que ele demonstra excelentes resultados quanto a integridade dos dados em velocidades elevadas. Suas limitações começam a surgir quando o intervalo de transmissão é reduzido para 0.001 segundos, onde se torna possível perceber que a integridade das mensagens foram comprometidas.

4.3 Testes e validação

Por fim, a última etapa consiste nos testes e validação do veículo em diferentes condições para que seja feita a verificação de problemas e avaliação de performance do que este artigo propõe.

4.3.1 Teste de algoritmo em diferentes intensidades luminosas

Dada a influência significativa da luminosidade no desempenho do algoritmo de detecção de faixas, torna-se importante avaliar a sua capacidade em produzir resultados coerentes em diversas condições horárias, cada uma apresentando suas próprias nuances. Com o intuito de abordar essa questão, foram conduzidos testes em distintos períodos do dia, visando validar o comportamento do algoritmo em diferentes contextos de iluminação.

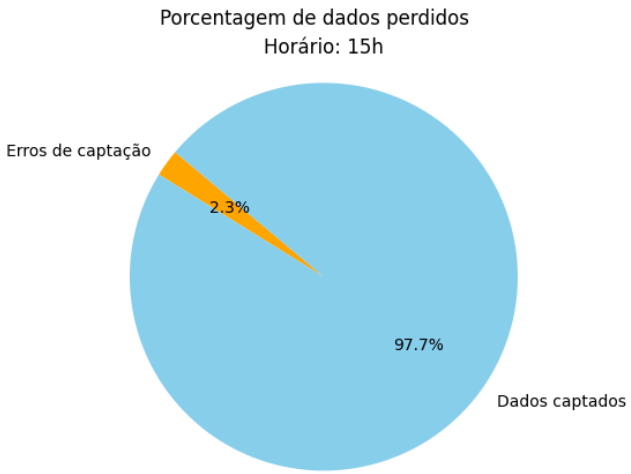
- Horário do teste: 15h

Figura 43 – Cenário teste às 15h

Fonte: O autor

A seleção desse horário específico foi motivada pela elevada intensidade luminosa, uma vez que é o período em que a luz solar atinge seu auge. A partir dessa condições desafiadoras, o cenário foi montado na parte exterior do laboratório e o veículo foi submetido ao circuito em uma trajetória retilínea. Durante os testes, a Zed Box foi equipada com um segundo algoritmo que armazenava os dados enviados ao sistema, e a ECU do Arduíno armazenava os dados que foram recebidos. Para verificarmos o comportamento dos erros, cada vez que o algoritmo encontrava alguma exceção em seu funcionamento era gravado no arquivo de dados armazenados o carácter -". E para verificar a integridade dos dados recebidos, basta comparar o arquivo armazenado na Zed Box com os dados recebidos no Arduíno.

Figura 44 – Resultados obtidos durante o teste às 15h



Fonte: O autor

O gráfico (figura 44) apresenta em azul os dados capturados dentro do intervalo

previsto de funcionamento do algoritmo e que foram capturados no Arduíno. Os dados em laranja representam as falhas do algoritmo durante a captação das faixas, e caso tivessem divergências entre os dados das 2 ECUs seriam apresentadas em vermelho.

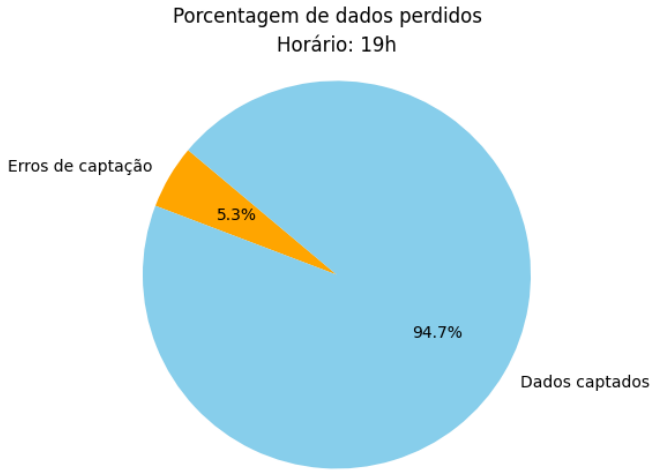
- Horário do teste: 19h

Figura 45 – Cenário teste às 19h

Fonte: O autor

Testar o algoritmo em condições de iluminação mais baixa permite avaliar a robustez do sistema e garantir que ele funcione de maneira confiável em diversas condições de luminosidade. Dessa forma, a partir do mesmo circuito, foram obtidos os resultados abaixo (figura 46).

Figura 46 – Resultados obtidos durante o teste às 19h



Fonte: O autor

Em ambos os testes não foram encontradas diferenças entre a quantidade de dados enviados pela Zed Box e os dados recebidos pela ECU do Arduíno, isso demonstra a eficácia do protocolo UDP para o projeto. Já o algoritmo, encontra espaço para melhoria, uma vez que a quantidade de dados perdidos diante a captação pode ser reduzido nas etapas de processamento de imagens. Para contornar essa limitação, foram implementadas técnicas de tratamento de erros, como a estratégia de armazenar a última posição registrada do veículo na faixa e retransmiti-la ao sistema. Essa abordagem visa mitigar a perda de dados, considerando que o veículo ainda estará posicionado em proximidade da última posição registrada.

4.3.2 Teste de algoritmo em curvas longas

Com a instalação das faixas de rodagem na área externa do laboratório, tornou-se viável a realização de testes de algoritmos em diversos momentos ao longo do dia em um percurso relativamente maior. O que possibilitou ser observado diferentes resultados, onde em horários nos quais o sol se encontra em uma posição mais baixa no céu, o algoritmo de detecção de faixas apresenta desafios adicionais no que diz respeito ao seu desempenho e capacidade de reconhecimento.

Os períodos mais apropriados para avaliar o desempenho do projeto neste cenário específico são das 11:00 às 12:00 e das 13:00 às 14:00, uma vez que o sol se encontra em uma posição elevada no céu, proporcionando uma iluminação ideal da pista. Além disso, notamos que o projeto apresenta um funcionamento consistente durante as horas da manhã. No entanto, em algumas ocasiões, o código pode capturar exceções durante a sua execução, as quais são devidamente tratadas durante o processo de transmissão de dados.

Figura 47 – Cenário teste de algoritmo em curvas longas

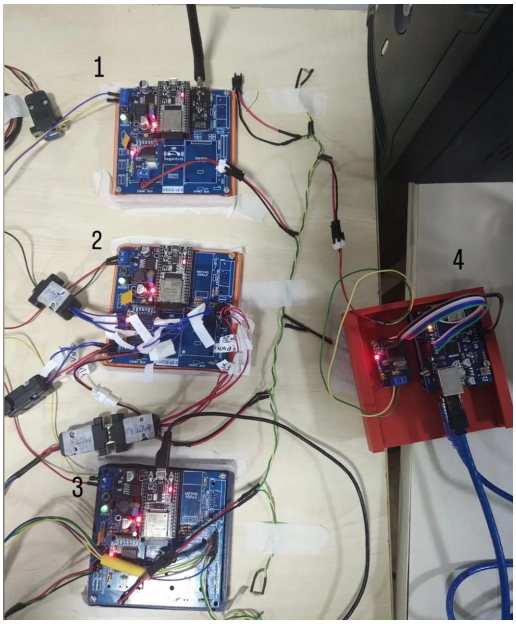
Fonte: O autor

Apesar dos resultados positivos que foram alcançados, o algoritmo enfrentou desafios significativos durante um dos testes conduzidos nesse cenário. Durante a realização do teste no período das 17:00 às 18:00, notou-se que o sistema de detecção de faixas não conseguia fornecer informações confiáveis acerca da localização das margens da pista em determinadas seções do percurso. Após uma análise, foi identificado que a luz solar estava refletindo nas faixas, conferindo-lhes uma característica brilhante. Esse brilho causava confusão nos estágios de processamento de imagens, levando à interpretação errônea da cor verde presente nas faixas, resultando na incapacidade de reconhecê-las de forma precisa.

4.3.3 Teste de LKA em bancada

Durante esse processo, o veículo em pequena escala foi desmontado de maneira que todas as suas ECUs foram retiradas, juntamente com seus motores e demais componentes. Dessa forma foram alocados em uma bancada e inserido junto a rede CAN a ECU responsável por receber os dados do sistema de LKA implementado. A figura 48 demostra as ECUs retiradas do veículo para a realização dessa etapa.

Figura 48 – Cenário teste em bancada do sistema de LKA



Fonte: O autor

As ECUs enumeradas na figura acima representam respectivamente:

- 1 - Transmissão e Recepção: Responsável por receber e enviar dados ao controle;
- 2 - Aceleração: Responsável por controlar a velocidade do veículo;
- 3 - Direção: Responsável pelo controle do ângulo do volante;
- 4 - LKAS: Responsável por repassar os dados do algoritmo LKAS para a rede CAN.

A partir da movimentação da câmera conectada ao sistema em bancada, foi analisada a rotação do motor do sistema de direção. O que permitiu calibragem dos valores enviados no algoritmo e dos valores entregues do potenciômetro no sistema de direção. Além disso, os motores conectados a ECU de direção devem permanecer na mesma velocidade enviada pelo controle para a ECU de recepção. Dessa forma, foram desenvolvidas técnicas para solução de conflitos na rede CAN, para que o sistema atue da melhor maneira possível.

Os resultados obtidos, foram verificados a partir da velocidade que a direção mudava o sentido, a partir da aproximação da câmera das faixas de rolagem instaladas no laboratório, onde era conduzida de uma extremidade da faixa até a outra. Quando os resultados foram considerados satisfatórios, o veículo foi remontado para realizar um teste de campo.

4.3.4 Teste de campo do sistema LKA

Com a configuração do ambiente fora do laboratório, tornou-se viável observar o comportamento do veículo em situações do mundo real. Isso foi conduzido em um pequeno percurso em linha reta, no qual as condições de iluminação e outros desafios visuais não são controlados.

Figura 49 – Teste de campo do sistema LKA

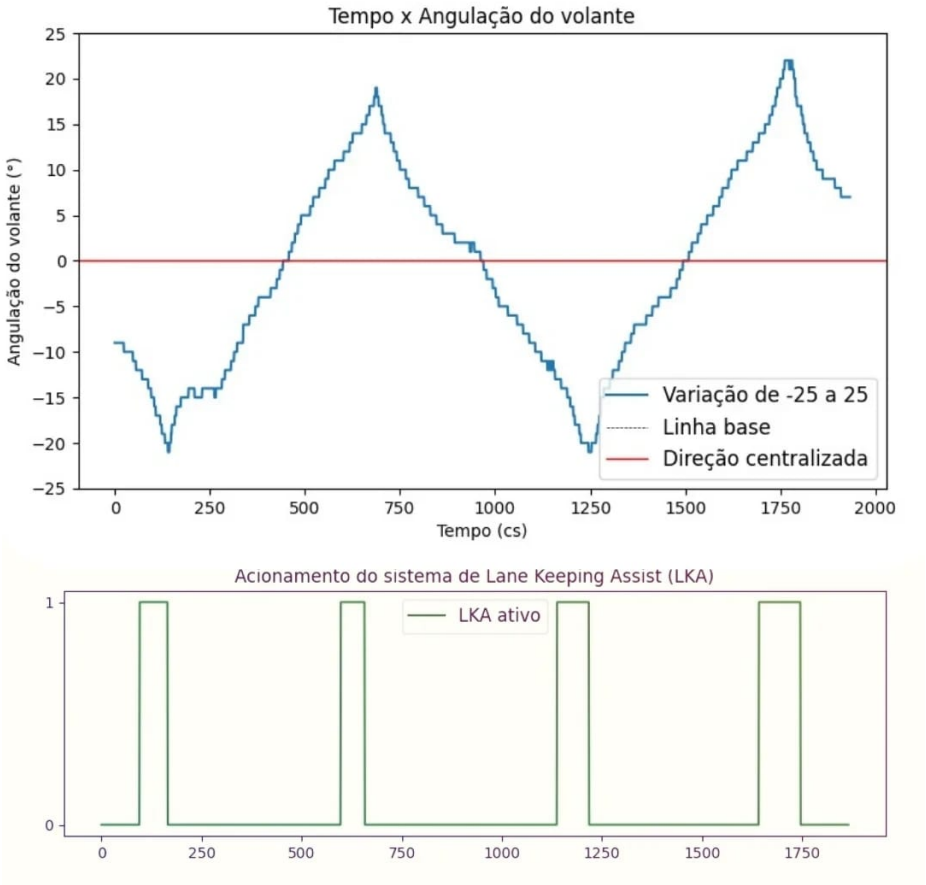
Fonte: O autor

O primeiro passo para a condução do teste foi determinar uma distância mínima para a ativação da função de *Lane Keeping*. O valor escolhido irá representar a distância que o veículo se encontra da faixa mais próxima, assim deve-se evitar que seja muito pequeno, proporcionando pouco tempo para o algoritmo realizar o processamento, e ao mesmo tempo, garantir que não fosse excessivamente grande, o que poderia comprometer a autonomia e o controle do motorista. Para esse teste, foi escolhido a distância de 10 cm.

Uma vez que definido essa etapa, foi possível verificar a transmissão dos dados provenientes do algoritmo para o sistema de direção do veículo. De maneira que, quando o veículo se aproximava da faixa dentro do intervalo definido para redirecioná-lo para os limites da pista.

O gráfico inicial apresenta os valores do ângulo de esterção do volante ao longo do percurso, com uma variação de -25, indicando o valor máximo para a esquerda, até 25, que representa o valor máximo para a direita. Já o gráfico abaixo (figura 50) representa o período de acionamento do LKAS para corrigir a direção do veículo.

Figura 50 – Resultados obtidos durante 1° teste de campo



Fonte: O autor

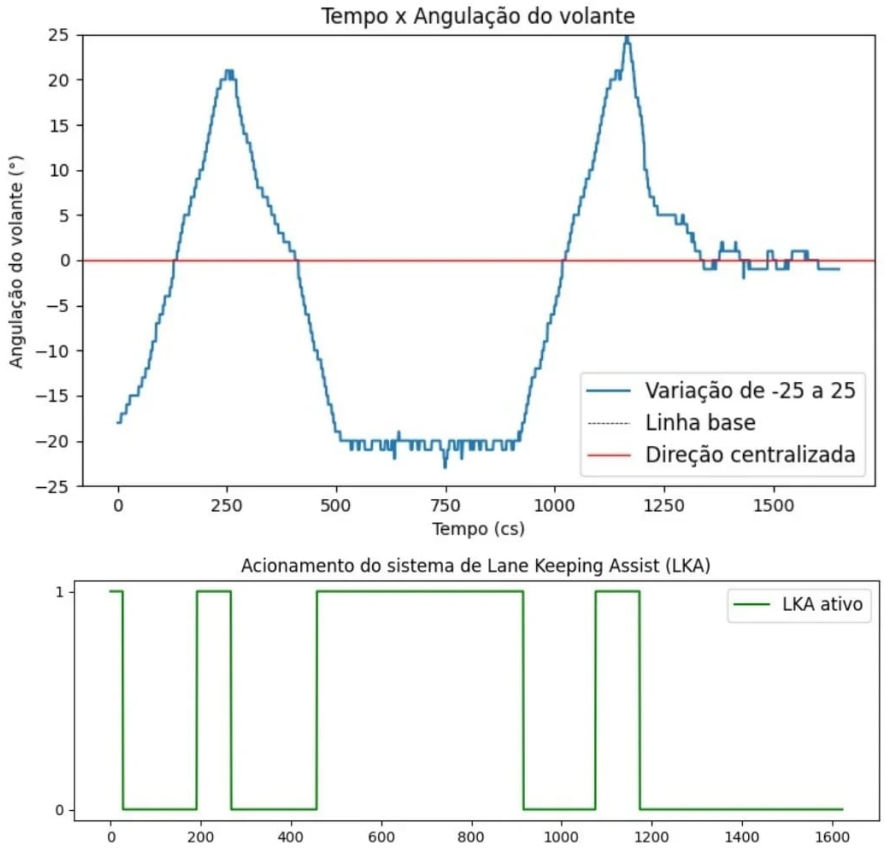
Durante a realização do percurso, observou-se que o veículo manteve-se consistentemente dentro dos limites da faixa de rolagem em todas as iterações, a partir dos 4 acionamentos do sistema de *Lane Keeping*. Esse resultado corrobora com a eficácia do sistema, demonstrando sua capacidade de manter a trajetória do veículo alinhada de acordo com as marcações da pista. Vale ressaltar que o foco deste teste específico estava na avaliação do comportamento do sistema sob condições de velocidade constante.

O gráfico inicial exibe uma variação quase inteiramente linear, invertendo o sentido quando o sistema de *Lane Keeping Assist* é ativado, conforme o veículo se aproxima da faixa. Essa alteração no padrão da variação linear sugere uma resposta dinâmica e imediata do sistema LKA, evidenciando sua capacidade de correção e ajuste da trajetória do veículo. Dessa forma, foi possível analisar o comportamento do veículo ao percorrer a pista de uma extremidade à outra. Onde, ao se aproximar das bordas da faixa de um lado, corrigia a angulação do volante para o outro lado conforme avançava até o término da pista.

O veículo percorreu o mesmo trajeto, porém a uma velocidade dinâmica, a fim de avaliar seu desempenho em relação à influência da aceleração e frenagem no sistema de

Lane Keeping. Essa abordagem permitiu uma análise mais abrangente do comportamento do veículo, considerando as variações na velocidade e os desafios adicionais impostos pelas mudanças no ritmo da condução.

Figura 51 – Resultados obtidos durante 2° teste de campo



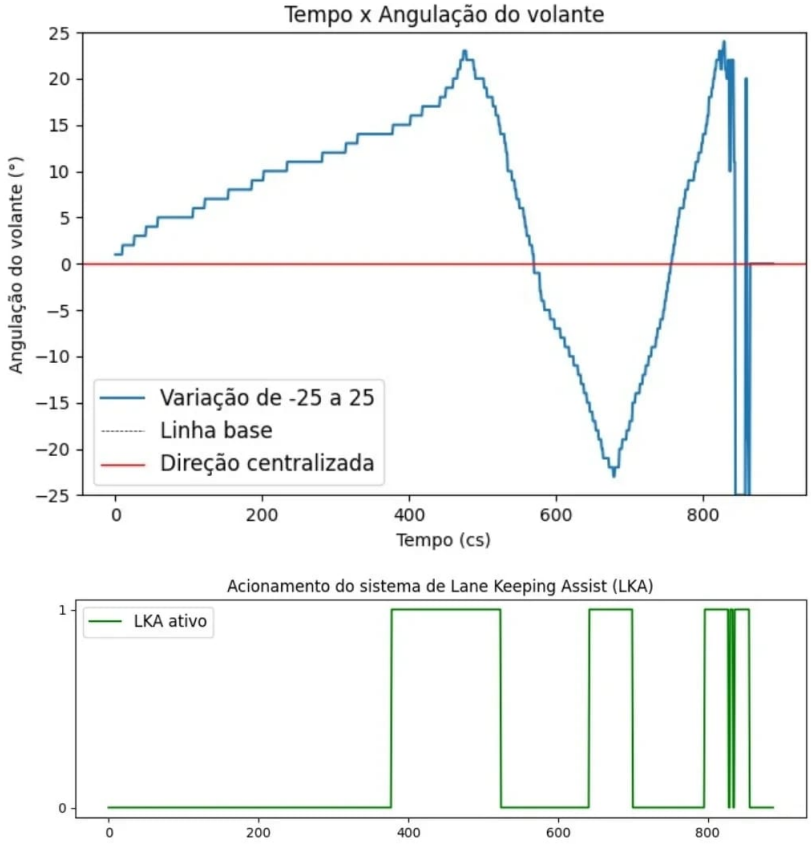
Fonte: O autor

Neste segundo caso, o veículo foi posicionado próximo a uma das faixas de maneira que o percurso iniciou com a ativação do sistema de LKA. Uma vez ativado, o veículo encontrou-se diante de diversas situações de velocidade, resultantes das variações nos recursos de aceleração e frenagem. Essas variações englobaram desde pequenos solavancos em velocidades mais elevadas até paradas completas. Por fim, o veículo se estabiliza no centro da pista com pequenas inclinações nas rodas, resultado de uma parada realizada depois da angulação ser corrigida pelo LKAS e a ECU de direção voltar a interpretar os dados recebidos pelo controle.

A aplicação de uma velocidade invariável proporcionou resultados valiosos sobre a capacidade do sistema em reagir e ajustar a direção do veículo de maneira estável e coerente, evidenciando a confiabilidade do sistema. Porém, foi observado que em velocidades maiores o sistema não se comporta da mesma maneira, já que o LKAS não atua sobre o sistema de aceleração e frenagem, resultando em curvas bruscas que muitas vezes acabam

na saída do veículo da pista.

Figura 52 – Resultados obtidos durante 3° teste de campo



Fonte: O autor

Nesse cenário, utilizou-se uma aceleração constante, direcionando o veículo ao encontro da faixa, de maneira se tornou possível visualizar o funcionamento do sistema em velocidades elevadas. Observou-se que a partir da segunda correção da ECU do *Lane Keeping* o veículo ultrapassa ligeiramente os limites da pista. Entretanto, devido ao movimento abrupto do volante, a direção permanece voltada para o lado oposto antes da próxima correção ser acionada. Essa dinâmica resulta na saída do veículo da pista e em diversas inconsistências no algoritmo, que continua em execução enviando valores inconsistentes para o sistema de direção, como pode ser observado no gráfico a partir do oitavo segundo (figura 52).

Outro ponto que foi possível avaliar durante essas etapas, foi o processo implementado para lidar com o conflito de mensagens que chegam na ECU de direção, provenientes do controle e do sistema LKA. Quando acionada a função de *Lane Keeping* deve ter prioridade quanto aos comandos enviados a direção do veículo, o que pode ser observado durante o desenvolvimento do teste. Assim que o veículo adentra os limites estabelecidos, o sistema de direção só reconhece as mensagens enviadas pela ECU do LKAS, até que a última mensagem enviada contenha a *flag* de finalização do *Lane Keeping*.

5 Conclusão

O desenvolvimento do *Lane Keeping* no veículo de pequena escala é de extrema importância para consolidação das pesquisas direcionadas as funções ADAS, pois, de maneira geral, consolidou o primeiro passo em direção a implementação das tecnologias presentes nos carros autônomos. A partir da base teórica fornecida pela revisão bibliográfica, os objetivos do projeto foram alcançados de maneira satisfatória. Uma vez que os sistemas e arquiteturas desenvolvidos para garantir a manutenção do veículo nos limites da faixa apresentaram resultados condizentes com a proposta do trabalho, a partir de diferentes técnicas de processamento de imagens. Além disso, este trabalho atingiu proporções estruturais maiores que o planejado, desde a criação dos componentes até os circuitos eletrônicos para as ECUs, o que permitiu embarcar os diferentes sistemas no veículo.

Com as diferentes técnicas utilizadas no processamento de imagens, tornou-se possível superar as barreiras impostas pelas condições de luminosidade. Ainda que o algoritmo capture falhas de reconhecimento das faixas durante a execução, os resultados obtidos a partir dos testes demonstram a eficácia das funções do *Lane Detection* quando implementadas no sistema de *Lane Keeping*. Uma vez que esse sistema não permanece "ativo"durante toda a execução do trajeto, apenas quando os limites mínimos são traspassados, tornou-se possível mitigar grande parte das exceções a partir das abordagens para tratamento de erros.

A estrutura inicial do sistema, embora simplificada em um primeiro momento, continua aberta a aprimoramentos, pois mesmo com as mudanças na comunicação entre os componentes, para atingir as demandas de sistemas mais complexos, como o *Lane Centering*, novas técnicas de validação devem ser implementadas. Mesmo assim, todas as etapas da comunicação de componentes funcionam de maneira conjunta com a rede do veículo. Dessa forma, uma nova funcionalidade pode ser facilmente implementada, uma vez que essa arquitetura já está implementada.

Este trabalho demonstrou a possibilidade de desenvolver uma arquitetura de manutenção de veículos nas faixas empregando técnicas de processamento de imagens em tempo real. A partir da combinação de recursos computacionais dos *hardwares* escolhidos com algoritmos de visão computacional oferece uma alternativa viável para a implementação do sistema de *Lane Keeping*.

Embora o trabalho tenha alcançado os resultados esperados, há espaço para aprimoramentos substanciais. Os ângulos de esterção, podem ser refinados por meio da implementação de algoritmos de controle mais complexos e eficientes. Além disso, é possível

estabelecer limites mais precisos para o movimento das rodas, visando atingir a angulação ideal de forma mais assertiva e adaptativa às variadas condições de condução. Diferentes técnicas de processamento de imagens podem ser implementadas para melhorar o desempenho do software em cenários com baixa luminosidade e minimizar as exceções. Essas possíveis melhorias não apenas otimizariam o desempenho do sistema, mas também abririam oportunidades para futuras inovações na implementação do *Lane Keeping Assist*.

Referências

ANDRADE, I. E.; ALBUQUERQUE, M. P.; ALBUQUERQUE, M. P. Processamento digital de imagens. Centro Brasileiro de Pesquisas Físicas, v. 13, 2003. Citado na página 34.

BRADSKI, A. *Learning OpenCV, [Computer Vision with OpenCV Library ; software that sees]*. 1. ed.. ed. [S.l.]: O'Reilly Media, 2008. Gary Bradski and Adrian Kaehler. ISBN 0-596-51613-4. Citado na página 29.

CHEN, Z.; LI, L.; HUANG, X. Building an autonomous lane keeping simulator using real-world data and end-to-end learning. *IEEE Intelligent Transportation Systems Magazine*, v. 12, n. 1, p. 47–59, 2020. Citado na página 21.

CONTRAMANUAL para câmeras inteligentes: vigilância, tecnologia e percepção. Galaxia (São Paulo, Online), 2006. Disponível em: <https://revistas.pucsp.br/index.php/galaxia/article/view/9807/9426>. Citado na página 31.

DATE, P. V.; GAIKWAD, V. Vision based lane detection and departure warning system. In: *2017 International Conference on Signal Processing and Communication (ICSPC)*. [S.l.: s.n.], 2017. p. 240–245. Citado na página 22.

ESKANDARIAN, A. *Handbook of Intelligent Vehicles*. [S.l.]: Springer London, 2012. Citado 2 vezes nas páginas 24 e 26.

FALCãO, A. X.; CHIACHIA, G. Introdução ao processamento de imagem digital. 2014. Citado na página 33.

GILLESPIE, T. D. *Fundamentals of Vehicle Dynamics*. [S.l.]: SAE, 1992. Citado 4 vezes nas páginas 40, 41, 42 e 43.

GONZALEZ, R.; WOODS, R. *Processamento Digital De Imagens*. ADDISON WESLEY BRA, 2009. ISBN 9788576054016. Disponível em: <https://books.google.com.br/books?id=r5f0RgAACAAJ>. Citado na página 35.

HEIDARIZADEH, A. Preprocessing methods of lane detection and tracking for autonomous driving. 04 2021. Citado na página 22.

HERMAWAN, R. Drive safely with precision and accuracy :lane departure warning system. BINUS ASO SCHOOL OF ENGINEERING, 2017. Disponível em: <https://base.binus.ac.id/2017/12/28/drive-safely-with-precision-and-accuracy-lane-departure-warning-system/>. Citado na página 25.

HOWE, K. D. *A Practical Introduction to Computer Vision with OpenCV*. Wiley, 2014. Disponível em: <https://books.google.com.br/books?id=F9MsAwAAQBAJ>. Citado na página 32.

HTET, T. K. K. K. K. K.; XINXIN, D. Comprehensive lane keeping system with mono camera. 2015. Citado na página 18.

JAVEED, M. A. et al. Lane line detection and object scene segmentation using otsu thresholding and the fast hough transform for intelligent vehicles in complex road conditions. *Electronics*, v. 12, n. 5, 2023. ISSN 2079-9292. Disponível em: <https://www.mdpi.com/2079-9292/12/5/1079>. Citado 4 vezes nas páginas 26, 27, 28 e 39.

KUO, C.; LU, Y.; YANG, S. On the image sensor processing for lane detection and control in vehicle lane keeping systems. *Sensors*, v. 19, n. 7, 2019. ISSN 1424-8220. Disponível em: <https://www.mdpi.com/1424-8220/19/7/1665>. Citado na página 21.

OAGANA, A. Lane keeping assist systems explained. AutoEvolution, 2016. Disponível em: <https://www.autoevolution.com/news/lane-keeping-assist-systems-explained-25459.html>. Citado na página 18.

OGê, M. F.; HUGO, V. N. *Processo Digital de Imagens*. Brasport, 2014. Disponível em: <https://www.ogemarques.com/wp-content/uploads/2014/11/pdi99.pdf>. Citado 6 vezes nas páginas 30, 31, 32, 33, 36 e 37.

ORGANIZATION, W. H. *Road traffic injuries*. 2022. https://www.who.int/news-room/fact-sheets/detail/road-traffic-injuries. Citado na página 16.

QUEIROZ, J.; GOMES, H. Introdução ao processamento digital de imagens. *RITA*, v. 13, p. 11–42, 01 2006. Citado na página 32.

ROMANO, R. et al. Impact of lane keeping assist system camera misalignment on driver behavior. *Journal of Intelligent Transportation Systems*, v. 25, n. 2, p. 157–169, 2021. ISSN 1547-2450. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1547245022011331>. Citado na página 22.

SHAR, G. New research from toyota for safe driving, lane-centering, data and more. 2023. Disponível em: <https://www.autoconnectedcar.com/2023/07/new-research-from-toyota-for-safe-driving-lane-centering-data-and-more/>. Citado na página 26.

VIJITKUNSAWAT, W.; CHANTNGARM, P. Comparison of machine learning algorithm's on self-driving car navigation using nvidia jetson nano. p. 201–204, 2020. Citado na página 21.