

UNIVERSIDADE ESTADUAL DO NORTE FLUMINENSE
DARCY RIBEIRO

*Centro de Ciência e Tecnologia
Laboratório de Ciências Matemáticas*

Daniel Terra Gomes

**Simulação de Detecção de Objetos em Tempo
Real para Veículos Autônomos**

Campos dos Goytacazes, RJ

1 de maio de 2025

Daniel Terra Gomes

Simulação de Detecção de Objetos em Tempo Real para Veículos Autônomos

Trabalho de Conclusão de Curso apresentado
ao Curso de Graduação em Ciência da Com-
putação da Universidade Estadual do Norte
Fluminense Darcy Ribeiro como requisito
para a obtenção do título de Bacharel em
Ciência da Computação, sob orientação de
Annabell Del Real Tamariz

Universidade Estadual do Norte Fluminense Darcy Ribeiro

Centro de Ciência e Tecnologia

Laboratório de Ciências Matemáticas

Ciência da Computação

Orientador: Prof. Dr. Annabell Del Real Tamariz

Campos dos Goytacazes, RJ

1 de maio de 2025

Daniel Terra Gomes

Simulação de Detecção de Objetos em Tempo Real para Veículos Autônomos

Trabalho de Conclusão de Curso apresentado
ao Curso de Graduação em Ciência da Com-
putação da Universidade Estadual do Norte
Fluminense Darcy Ribeiro como requisito
para a obtenção do título de Bacharel em
Ciência da Computação, sob orientação de
Annabell Del Real Tamariz

Trabalho aprovado. Campos dos Goytacazes, RJ, data:

Prof. Dr. Annabell Del Real Tamariz
Orientador

Prof. Dr.
Membro da Banca

Campos dos Goytacazes, RJ
1 de maio de 2025

Agradecimentos

*“Behind me lies a farm.
I wonder if there is bread above the hearth
and if I will ever return.”*
(Pantheon, League of Legends)

Resumo

Palavras-chave: Carros autônomos. Detecção de Objetos. Controle Veicular. Simulação. CARLA. YOLO.

Abstract

Keywords: Self-driving car. Objects Detection. Vehicle Control. Simulation. CARLA. YOLO.

Lista de ilustrações

Figura 1 – Método Científico (Cristina; Wazlawick, 2021, p. 4).	21
Figura 2 – Níveis de Automação de condução PT-BR	27
Figura 3 – SAE J3016TM levels of driving automation (SAE, 2021)	28
Figura 4 – Níveis de Automação (Parekh Nishi Poddar, 2022).	31
Figura 5 – Componentes em camadas para alcançar automação na condução (Khan <i>et al.</i> , 2022, p. 4).	32
Figura 6 – Relação dos componentes entre sub tarefas e camadas. (Khan <i>et al.</i> , 2022, p. 4).	33
Figura 7 – Componentes funcionais de um sistema de direção autônomo e arquitetura (Zheng <i>et al.</i> , 2023, p. 2).	35
Figura 8 – O Sistema de Detecção YOLO. O processamento de imagens com YOLO é simples e direto. O sistema (1) redimensiona a imagem de entrada para 448×448 , (2) executa uma única rede convolucional na imagem e (3) limita as detecções resultantes pela confiança do modelo. (Redmon <i>et al.</i> , 2016, p. 1).	36
Figura 9 – Comparações com outros em termos de trade-offs de latência-precisão (esquerda) e tamanho-precisão (direita). Medimos a latência de ponta a ponta usando os modelos oficiais pré-treinados. (Wang <i>et al.</i> , 2024, p. 1).	37
Figura 10 – Componentes de um veículo autônomo (Singh, 2022, p. 15).	38
Figura 11 – Tipos de sensores em VAs e seus posicionamentos (University of Toronto, 2018a, Week 2 - Lesson 1: Sensors and Computing Hardware. 9min00s).	38
Figura 12 – Adequação de sensores para diferentes situações (Khan <i>et al.</i> , 2022, p. 6).	39
Figura 13 – Cobertura geral dos sensores (University of Toronto, 2018a, Week 2 - Lesson 2: Hardware Configuration Design. 9min00s).	41
Figura 14 – Arquitetura de Software (University of Toronto, 2018a, Week 2 - Lesson 3: Software Architecture. 3min49s).	42
Figura 15 – Arquitetura de Software (University of Toronto, 2018a, Week 2 - Lesson 4: Environment Representation. 3min30s).	43
Figura 16 – Arquitetura de Software (University of Toronto, 2018a, Week 2 - Lesson 4: Environment Representation. 2min07s).	44
Figura 17 – Arquitetura de Software (University of Toronto, 2018a, Week 2 - Lesson 4: Environment Representation. 6min36s).	44
Figura 18 – Simulador Carla (CARLA, 2018). Diferentes condições ambientais e Mapas (Capturas de tela pelos autores).	48
Figura 19 – Diagrama de blocos de detecção de sinais de trânsito (Surendra, 2023, p. 2064).	50

Figura 20 – Fluxograma de detecção de sinais de trânsito (Surendra, 2023, p. 2065).	51
Figura 21 – Arquitetura do Software. Elaborado pelo Autor.	53
Figura 22 – Arquitetura de Software da Percepção do Ambiente. Baseado em University of Toronto (2018a).	54
Figura 23 – Arquitetura de Software do Planejador de Movimento. Baseado em University of Toronto (2018a).	54
Figura 24 – Arquitetura de Software do Controlador do Veículo. Baseado em University of Toronto (2018a).	55
Figura 25 – Fluxo da Execução da Solução. Elaborado pelo Autor.	55
Figura 26 – Robô com rodas e uma câmera omnidirecional (Francis <i>et al.</i> , 2016, p. 9).	57
Figura 27 – Modelo de bicicleta (Francis <i>et al.</i> , 2016, p. 13).	57
Figura 28 – Sistemas de coordenadas (Jacobson <i>et al.</i> , 2016, p. 33).	58
Figura 29 – Modelo de bicicleta 2D (modelo de carro simplificado) (University of Toronto, 2018a, Week 4 - Lesson 2: The Kinematic Bicycle Model. 1min13s).	59
Figura 30 – Ponto de referência da roda traseira no modelo de bicicleta 2D (University of Toronto, 2018a, Week 4 - Lesson 2: The Kinematic Bicycle Model. 3min00s).	60
Figura 31 – Ponto de referência na roda traseira no modelo de bicicleta 2D (University of Toronto, 2018a, Week 4 - Lesson 2: The Kinematic Bicycle Model. 3min46s).	61
Figura 32 – Ponto de referência no eixo dianteiro (University of Toronto, 2018a, Week 4 - Lesson 2: The Kinematic Bicycle Model. 4min35s).	62
Figura 33 – Ponto de referência no centro de massa (University of Toronto, 2018a, Week 4 - Lesson 2: The Kinematic Bicycle Model. 5min45s).	62
Figura 34 – Modelagem Dinâmica (University of Toronto, 2018a, Week 4 - Lesson 3: Dynamic Modeling in 2D. 2min18s).	65
Figura 35 – Modelagem Dinâmica (University of Toronto, 2018a, Week 4 - Lesson 3: Dynamic Modeling in 2D. 4min20s).	66
Figura 36 – Modelo Longitudinal de Veículo (University of Toronto, 2018a, Week 4 - Lesson 4: Longitudinal Vehicle Modeling. 1min43s).	68
Figura 37 – Modelo de carro para modelo de bicicleta (University of Toronto, 2018a, Week 4 - Lesson 5: Lateral Dynamics of Bicycle Model. 0min48s).	70
Figura 38 – Sistema Massa-Mola-Amortecedor (University of Toronto, 2018a, Week 5 - Lesson 1: Proportional-Integral-Derivative (PID) Control. 9min00s).	75
Figura 39 – Controlador P (University of Toronto, 2018a, Week 5 - Lesson 1: Proportional-Integral-Derivative (PID) Control. 12min13s).	75
Figura 40 – Controlador PD (University of Toronto, 2018a, Week 5 - Lesson 1: Proportional-Integral-Derivative (PID) Control. 12min13s).	76

Figura 41 – Controlador PI (University of Toronto, 2018a, Week 5 - Lesson 1: Proportional-Integral-Derivative (PID) Control. 12min13s).	76
Figura 42 – Controlador PID (University of Toronto, 2018a, Week 5 - Lesson 1: Proportional-Integral-Derivative (PID) Control. 12min50s).	77
Figura 43 – Segmentos de linha reta (University of Toronto, 2018a, Week 6 - Lesson 1: Introduction to Lateral Vehicle Control. 2min52s).	78
Figura 44 – Pontos de referência (University of Toronto, 2018a, Week 6 - Lesson 1: Introduction to Lateral Vehicle Control. 3min00s).	79
Figura 45 – Curvas parametrizadas (University of Toronto, 2018a, Week 6 - Lesson 1: Introduction to Lateral Vehicle Control. 3min30s).	79
Figura 46 – Trajetória de referência (University of Toronto, 2018a, Week 6 - Lesson 1: Introduction to Lateral Vehicle Control. 6min23s).	80
Figura 47 – Erro de trajetória cruzada (University of Toronto, 2018a, Week 6 - Lesson 1: Introduction to Lateral Vehicle Control. 8min31s).	81
Figura 48 – Rastreamento de caminho geométrico (University of Toronto, 2018a, Week 6 - Lesson 2: Geometric Lateral Control - Pure Pursuit. 2min27s).	82
Figura 49 – Pure pursuit (University of Toronto, 2018a, Week 6 - Lesson 2: Geometric Lateral Control - Pure Pursuit. 3min15s).	83
Figura 50 – Pure pursuit ICR (University of Toronto, 2018a, Week 6 - Lesson 2: Geometric Lateral Control - Pure Pursuit. 3min52s).	84
Figura 51 – Erro Crosstrack (University of Toronto, 2018a, Week 6 - Lesson 2: Geometric Lateral Control - Pure Pursuit. 3min52s).	85
Figura 52 – Planejamento Hierárquico (University of Toronto, 2018b, Module 2 - Lesson 1: Driving Missions, Scenarios, and Behaviour. 10min55s).	87
Figura 53 – Máquinas de Estados Finitos (University of Toronto, 2018b, Module 2 - Lesson 4: Hierarchical Motion Planning. 7min54s).	88
Figura 54 – Lattice Planner (University of Toronto, 2018b, Module 2 - Lesson 4: Hierarchical Motion Planning. 15min30s).	89
Figura 55 – Conformal Lattice Planner (University of Toronto, 2018b, Module 2 - Lesson 4: Hierarchical Motion Planning. 15min30s).	90
Figura 56 – Imagem a partir da Camera: Scene final (CARLA Simulator, 2024).	91
Figura 57 – Modelo. O sistema modela a detecção como um problema de regressão. Dividindo a imagem em uma grade $S \times S$ e para cada célula da grade prevê caixas delimitadoras B , confiança para essas caixas e probabili- dades de classe C . Segundo os autores, essas previsões são codificadas como um tensor $S \times S \times (B \times 5 + C)$. Redmon <i>et al.</i> (2016, p. 2).	93

Figura 58 – Arquitetura. A rede de detecção tem 24 camadas convolucionais seguidas por 2 camadas totalmente conectadas. Camadas convolucionais alternadas 1×1 reduzem o espaço de recursos das camadas precedentes. Nós pré-treinamos as camadas convolucionais na tarefa de classificação do ImageNet na metade da resolução (imagem de entrada 224×224) e então dobrarmos a resolução para detecção (Redmon *et al.*, 2016, p. 3). 94

Figura 59 – Esquema geral de um sistema de detecção e reconhecimento de sinais de trânsito usando o algoritmo de detecção de objetos YOLO. Imagem de (Flores-Calero *et al.*, 2024, p. 2). 95

Lista de tabelas

Tabela 1 – Características dos ganhos P, I e D (University of Toronto, 2018a, Week 5 - Lesson 1: Proportional-Integral-Derivative (PID) Control. 8min11s). 74

Lista de trechos de código

3.1 Exemplo da documentação CARLA para adicionar uma câmera “Scene final” no simulador (CARLA Simulator, 2024)	92
--	----

Lista de abreviaturas e siglas

VA	Veículo Autônomo
VAs	Veículos Autônomos
IA	Inteligência Artificial
SAE	Society of Automotive Engineers
GPS	Global Positioning System
ICR	Instantaneous Center of Rotation
RPM	Rotação Por Minuto
P	Proporcional
PI	Proporcional Integral
PD	Proporcional Derivativo
PID	Proporcional Integral Derivativo
Feedback	Realimentação
Feedforward	Antecipação
2D	Bidimensional
3D	Tridimensional
MPC	Model Predictive Controller
ADAS	Advanced Driver-Assistance System
ADS	Automated Driving System
CNN	Convolutional Neural Networks
RCNN	Region Convolution Neural Network
RNN	Recurrent Neural Network
TCC	Trabalho de Conclusão de Curso
ms	Milissegundo

Sumário

1	INTRODUÇÃO	16
1.1	Problema	18
1.2	Hipótese	18
1.3	Objetivos	19
1.3.1	Objetivos Específicos	19
1.4	Justificativa	20
1.5	Método	20
1.6	Organização do Trabalho	24
2	VEÍCULOS AUTÔNOMOS	26
2.1	Classificação de Níveis de Automação	26
2.2	Tarefa de Condução	31
2.2.1	Tecnologias Essenciais para as Tarefas de Condução	33
2.2.1.1	Percepção e Tarefas Relacionadas	33
2.2.1.2	Detecção e Classificação de Objetos	35
2.2.1.3	Componentes de Hardware	37
2.2.2	Arquitetura da Tarefa de Condução	42
2.3	Simulação de Carros Autônomos	46
2.3.1	Ambientes de simulação de carro autônomo	46
2.4	Trabalhos relacionados	48
3	MODELAGEM CONCEITUAL PARA SIMULAÇÃO DE CARROS AUTÔNOMOS	53
3.1	Controle de Carros Autônomos	56
3.1.1	Modelagem Cinemática em 2D	57
3.1.1.1	Sistemas de Coordenadas, Transformações	58
3.1.1.2	Modelo Cinemático da Bicicleta	59
3.1.2	Modelagem Dinâmica em 2D	64
3.1.2.1	Modelagem Longitudinal	67
3.1.2.2	Modelagem Lateral	69
3.1.3	Controle Longitudinal de Veículos	70
3.1.4	Controle Lateral de Veículos	77
3.1.4.1	Controle Lateral Geométrico - Controlador de Perseguição Pura	82
3.2	Planejamento de Movimento de Carros Autônomos	87
3.3	Percepção Visual de Carros Autônomos	90
3.3.1	Câmeras	91

3.3.2	You Only Look Once - YOLO	92
3.3.2.1	Arquitetura	92
3.3.2.2	Treinamento	94
4	IMPLEMENTAÇÃO	97
4.1	Configurando o Simulador Carla	97
5	RESULTADOS E DISCUSSÕES	99
6	CONSIDERAÇÕES FINAIS	101
7	PERSPECTIVA DE CONTINUIDADE	102
 REFERÊNCIAS		103
 APÊNDICES		109
APÊNDICE A – MATERIAL COMPLETO		110
 ANEXOS		111
ANEXO A – MATERIAL DE RELEVÂNCIA		112

1 Introdução

Veículos Autônomos (VAs) representam um dos avanços tecnológicos mais promissores para transformar a mobilidade urbana e a segurança no transporte (European Commission, Directorate-General for Mobility and Transport, 2021), (Othman, 2021). Seu surgimento acompanha o crescimento do mercado de veículos elétricos (Sebo, 2024), permitindo que empresas como ZOOX®, WAYMO® e LUME ROBOTICS® invistam significativamente em sistemas de condução autônoma. Os quais, devido a tecnologias embarcadas de visão computacional, sensores de proximidade, entre outros (Visto 10), prometem maior eficiência, segurança e conveniência nos sistemas de transporte público e privado (Bratzel; Management, 2022), (Parekh Nishi Poddar, 2022; Dreibelbis, 2023). A partir da incorporação em ampla escala dessa tecnologia será possível notar impactos transformadores na infraestrutura urbana, na configuração das cidades e no comportamento social das pessoas, afetando positivamente a mobilidade cotidiana de diversos grupos populacionais, incluindo crianças, idosos e pessoas com mobilidade reduzida (Lage, 2019; KPMG International, 2020). Desses impactos, um dos mais significativos seria em redução nos acidentes de trânsito. Estudos indicam que erros humanos são responsáveis por mais de 90% dos acidentes de trânsito (National Highway Traffic Safety Administration, 2018), fazendo da tecnologia de VAs uma solução promissora para reduzir significativamente esses acidentes. Visto que esses veículos têm o poder de tomar ações por conta própria, e/ou dar assistência ao motorista em casos de situações perigosas (Okpono *et al.*, 2024). Estimativas sugerem que a adoção em larga escala de VAs poderia evitar até 585.000 mortes ao longo de uma década a partir de 2035 (Lanctot, 2017). Nesse quesito, soluções de detecção de objetos e a análise de cenários em tempo real, podem ser um aliado dos motoristas no seu dia a dia (Federal Highway Administration, 2012). Visto que tais tecnologias embarcadas, podem auxiliar e assistir os condutores durante uma tarefa de condução evitando acidentes e garantindo uma navegação segura (Janai *et al.*, 2020).

No entanto, para que a tecnologia de VAs avance até o ponto de uma direção autônoma nível 5 SAE (SAE, 2021, p. 28), é necessário enfrentar uma série de desafios técnicos, especialmente em relação à visão computacional e sistemas de detecção de objetos, onde se faz necessário identificar e informar rapidamente sobre novas detecções durante uma tarefa de condução. Avanços em estudos de desempenho de soluções de visão computacional em detecção de objetos enfrentam desafios quanto a altos custos na infraestrutura, logística e legislação associados a testes em veículos no mundo real. Devido a isso, pesquisadores estão cada vez mais recorrendo a simulações virtuais para desenvolver e validar essas soluções em sistemas autônomos (Dosovitskiy *et al.*, 2017; Ahire *et al.*, 2024). Ambientes simulados oferecem um cenário controlado e econômico para testar algoritmos de condução

autônoma em diversas situações, sem os riscos imediatos ou custos com recursos dos testes físicos. Dosovitskiy *et al.* (2017) destacam que as simulações democratizam o acesso a ferramentas de pesquisa e desenvolvimento para VAs, reduzindo barreiras financeiras e logísticas para a inovação. Em soluções autônomas, especialmente em VAs, a detecção de objetos é fundamental para a interpretação em tempo real do ambiente, permitindo que os veículos reconheçam e classifiquem obstáculos, placas de trânsito, pedestres e outros objetos essenciais ao seu redor (Neufville; Abdalla; Abbas, 2022; Mozaffari, 2020; Khan *et al.*, 2022). Algoritmos de visão computacional alcançam isso ao utilizar extensos dados de imagem para interpretar pistas visuais, identificar formas, cores, distâncias e objetos, aspectos essenciais para a segurança e a eficiência da condução autônoma. Esses algoritmos são alimentados usando dados oriundos de sensores, por exemplo: câmeras que capturam dados em tempo real, processados pelos VAs para tomar decisões informadas e assistir o condutor durante uma tarefa de condução (Singh, 2022, p. 15), (Khan *et al.*, 2022, p. 6). Para que a tarefa de percepção (Visto 2.2) dos VAs se torne eficaz em ambientes variados, os pesquisadores estão investigando abordagens sofisticadas para simular com precisão a detecção de objetos em tempo real (Dosovitskiy *et al.*, 2017, p. 1). As soluções integram, geralmente, conjuntos de ferramentas de aprendizado profundo com dados de sensores em tempo real para melhorar a precisão na identificação de objetos. Esses algoritmos são treinados com extensos dados de sensores coletados de simulações virtuais, onde são expostos a milhões de quilômetros de cenários de estrada virtuais, garantindo que encontrem várias condições ambientais, obstáculos, casos extremos e situações de alto risco que, de outra forma, seriam raros ou inseguros de replicar em testes no mundo real (Wachenfeld; Winner, 2016). Esse grande volume de dados aumenta a robustez dos modelos de detecção de objetos, garantindo que soluções tomem ações mais precisas, de modo a manter conformidade com a regulação de trânsito, alcançando uma condução segura.

Nos últimos anos, algoritmos baseados em Redes Neurais Convolucionais ou *Convolutional Neural Networks* (CNNs) em inglês e técnicas de aprendizado profundo têm sido amplamente utilizados para tarefas de detecção de objetos em VAs, aproveitando as melhorias no poder computacional (Li; Wang; Wang, 2023a, p. 4). Por exemplo, redes treinadas em conjuntos de dados de placas de trânsito podem ser usadas para detectar e classificar placas de trânsito relacionados à velocidade (Khan *et al.*, 2022, p. 6), e passar essas informações para a camada de Planejamento ou Sistemas Avançados de Assistência ao Condutor em inglês: *Advanced Driver Assistance Systems* (ADAS). Os quais utilizam essas informações para reagir a obstáculos modificando sua rota e movimento e/ou assistir o condutor com informações de trânsito baseadas nos resultados dessas detecções e classificações (Ahire *et al.*, 2024, p. 46), (Okpono *et al.*, 2024), (Khan *et al.*, 2022, p. 4).

Esses recursos de assistência são amplamente implementados em veículos com níveis de automação inferiores ao nível 4, conforme a classificação SAE (SAE, 2021). Por esses níveis demandarem atenção constante ou parcial do condutor durante a tarefa de condução,

esses sistemas de assistência durante a condução, desempenham um papel crucial no suporte aos motoristas, aprimorando a segurança de todos os usuários (Okpono *et al.*, 2024).

1.1 Problema

A falha na detecção e interpretação precisa e oportuna de sinais de trânsito, muitas vezes decorrente de limitações humanas, permanece um desafio significativo, contribuindo para acidentes e comprometendo a segurança.

Estudos apontam que erros humanos são responsáveis por mais de 90% dos acidentes de trânsito, evidenciando a necessidade de soluções tecnológicas capazes de mitigar esses riscos (National Highway Traffic Safety Administration, 2018). Embora os sistemas de assistência avançada ao condutor (ADAS) representem um avanço, sua eficácia depende da precisão dos algoritmos de detecção de objetos, que nem sempre apresentam resultados satisfatórios (Ahire *et al.*, 2024), (Khan *et al.*, 2022).

Além disso, a introdução de tecnologias autônomas, especialmente nos níveis mais altos de automação segundo a classificação da SAE (2021), enfrenta barreiras técnicas e práticas significativas, incluindo desafios relacionados à capacidade de processamento em tempo real, confiabilidade em condições ambientais variadas e integração com outros sistemas do veículo (Dosovitskiy *et al.*, 2017), (Wachenfeld; Winner, 2016). Esses desafios são agravados pelos altos custos associados ao treinamento e teste de algoritmos em ambientes reais, bem como pelas limitações legais e logísticas (Dosovitskiy *et al.*, 2017), (Ahire *et al.*, 2024).

Portanto, é essencial desenvolver e validar soluções que integrem detecção de objetos com precisão elevada e processamento eficiente em tempo real, utilizando plataformas simuladas para contornar as limitações de custo e risco. Tais soluções têm o potencial de aumentar a segurança no trânsito, oferecendo suporte mais robusto a motoristas e possibilitando avanços na implementação de VAs (Janai *et al.*, 2020), (Khan *et al.*, 2022).

1.2 Hipótese

Um sistema de assistência à condução em carros autônomos pode oferecer feedback visual de placas de trânsito em tempo real, tornando a condução mais confiável.

Essa hipótese é fundamentada na viabilidade técnica e nos resultados de trabalhos anteriores, que demonstram a eficácia de soluções baseadas em algoritmos avançados de detecção de objetos aplicados ao reconhecimento de sinais de trânsito. Nesse quesito, Dosovitskiy *et al.* (2017), demonstraram a eficácia do uso de simuladores como o CARLA

para validar sistemas de condução autônoma, reduzindo custos e riscos associados a testes em ambientes reais. Por meio de simulações, [Juanola \(2019\)](#), alcançaram uma precisão de 92% na detecção de placas de limite de velocidade utilizando o algoritmo YOLO, destacando sua capacidade de processar imagens em tempo real e fornecer informações críticas ao motorista. Esse sistema foi testado com tempos de execução rápidos (8 ms em GPU), mostrando ser adequado para aplicações que demandam baixa latência.

Trabalhos como [Li, Wang e Wang \(2023b\)](#) e [Wang e Yu \(2020\)](#) propuseram melhorias em algoritmos de detecção, como YOLOv7 e YOLOv4, aumentando a precisão na identificação de alvos pequenos, como placas de trânsito. Tais aprimoramentos incluem o uso de métricas específicas, como a distância de Wasserstein e redes de atenção (ACmix), que permitem melhor captação de características visuais em ambientes complexos. Essas abordagens comprovam ser possível aprimorar ainda mais a detecção de placas, mesmo em condições adversas ou em escalas reduzidas.

Além disso, [Andrade \(2022\)](#), demonstraram a integração entre detecção de objetos e cálculos de distância, validando soluções robustas em simulações no CARLA. Essas implementações reforçam a possibilidade de desenvolver sistemas que, além de reconhecerem placas de trânsito, forneçam informações detalhadas, como distâncias relativas, para apoio em tarefas de condução.

Portanto, baseando-se nos trabalhos supracitados, é factível concluir que sistemas de assistência à condução, integrados com algoritmos avançados e testados em simulações virtuais, podem oferecer uma solução prática e eficiente para mitigar falhas humanas na interpretação de sinais de trânsito. Esses sistemas podem aprimorar a segurança e confiabilidade da condução, especialmente em VAs e semi-autônomos, conforme apontam estudos anteriores ([Ahire et al., 2024; Khan et al., 2022](#)).

1.3 Objetivos

Desenvolver um sistema de detecção e reconhecimento de placas de trânsito em tempo real para usuários em um carro autônomo, promovendo uma condução assistida mais confiável.

1.3.1 Objetivos Específicos

1. Desenvolver uma solução de carro autônomo capaz de percorrer um trajeto de uma área urbana. Essa abordagem é fundamental porque ambientes urbanos incluem elementos como placas de trânsito e obstáculos que permitem validar o desempenho do sistema de detecção;
2. A solução de carro autônomo deverá lidar com objetos pelo trajeto, e reagir a sinaliza-

ções de trânsito. Visto que a solução precisa reconhecer corretamente sinais relevantes e tomar decisões apropriadas, como reduzir a velocidade ou parar, assegurando a segurança durante a condução;

3. Estabelecer um sistema de interação do carro autônomo e motorista das detecções das placas de trânsito pelo trajeto. Essa interação é importante para dar feedback claro e em tempo real, permitindo que o condutor ou o sistema autônomo entenda as condições da via e reaja de maneira eficiente.

1.4 Justificativa

A elevada taxa de acidentes de trânsito causadas por falhas humanas, muitas vezes ocasionados por falhas na percepção e interpretação de sinalizações de trânsito ([National Highway Traffic Safety Administration, 2018](#)), evidencia a necessidade de soluções tecnológicas que auxiliem os motoristas a identificar e reagir rapidamente às condições do ambiente. Sinalizações visuais como placas de trânsito são essenciais para a segurança, mas podem ser mal interpretadas ou ignoradas devido a fatores como visibilidade reduzida, distração ou cansaço do condutor. A implementação de um sistema de assistência baseado em visão computacional oferece um suporte adicional ao motorista, aumentando a segurança, especialmente em situações de alto risco ou baixa visibilidade ([Okpono et al., 2024](#)). Além disso, avanços em tecnologias automotivas e o maior acesso a veículos com tecnologias embarcadas contribuem para encontrar soluções que usem todos os recursos desses veículos, de modo a promover maior segurança aos seus passageiros.

1.5 Método

De modo a alcançar os objetivos [1.3](#) propostos, este trabalho adota uma abordagem metodológica baseada no método científico (Visto na Figura [1](#)), estruturada em etapas sequenciais e sistemáticas.

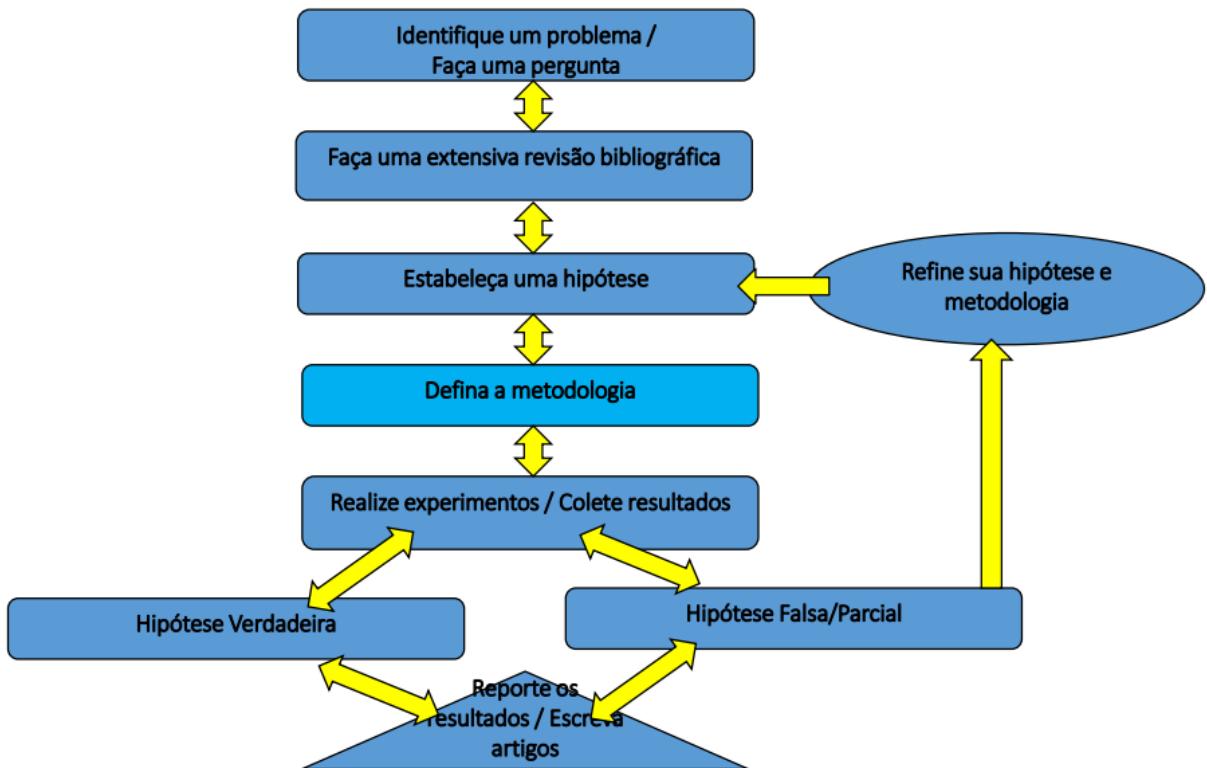


Figura 1 – Método Científico (Cristina; Wazlawick, 2021, p. 4).

Essas etapas visam validar a hipótese de que um sistema de assistência à condução pode oferecer feedback visual de placas de trânsito em tempo real, contribuindo para uma condução mais confiável. A seguir, são descritas as principais etapas do método adotado:

1. Definição do Problema e Planejamento:

- A metodologia inicia com a análise aprofundada do problema identificado: a falha na detecção e interpretação de sinalizações de trânsito, suas implicações na segurança veicular, e a necessidade de sistemas que superem as limitações humanas.
- A partir disso, foram delimitados os objetivos gerais 1.3 e específicos 1.3.1, que orientaram a elaboração de um plano experimental para desenvolver e validar o sistema de detecção de objetos e interação motorista-veículo em simulações.

2. Coleta de Dados e Seleção de Ferramentas:

- Dados para treinamento e validação do modelo de detecção de objetos foram coletados a partir de trabalhos presentes na revisão bibliográfica disponibilizados de maneira pública e dentro da licença MIT. Esse conjunto de dados foi escolhido devido à sua diversidade em tipos de placas, condições ambientais e ângulos de captura. Visto que o conjunto de dados contém imagens do simulador CARLA (2.950 rotuladas e 29.102 não rotuladas) misturando diferentes sessões

em diferentes cenários (Cidade01/Cidade02) e condições climáticas (Ensolarado/Chuvoso/Nublado) ([Juanola, 2019](#)) e outro conjunto de dados rotulado para detecção de objetos no simulador Carla contendo 1028 imagens ([Daniel, 2023](#)).

- As ferramentas e tecnologias escolhidas para o desenvolvimento incluem:
 - **Simulador CARLA:** um ambiente virtual para simulações de condução autônoma, permitindo a realização de experimentos controlados em cenários urbanos variados ([Dosovitskiy *et al.*, 2017](#)).
 - **Algoritmo YOLO (You Only Look Once):** escolhido por sua eficácia na detecção de objetos em tempo real, especialmente em aplicações automotivas ([Bochkovskiy; Wang; Liao, 2020](#); [Wu; Cao, 2022](#)).
 - **Linguagem de programação Python:** escolhido pelo alto número de soluções computacionais encontrados durante a revisão bibliográfica nesta linguagem, e pela maior familiaridade do autor com essa linguagem.
 - **Ambiente de programação:** Visual Studio Code, familiaridade do autor.
 - **Sistema operacional:** Linux e Windows. A edição dos mapas do simulador CARLA, para incluir placas de trânsitos em trajetos, é facilitada a partir de sistema Linux, de mesmo modo o treinamento do Algoritmo YOLO. Devido a isso, essas etapas serão feitas em Linux. O desenvolvimento da solução foi iniciando em Windows e continuará.
 - **Hardware de Processamento:** a execução será realizada preferencialmente em GPU para assegurar baixa latência e maior capacidade computacional, essencial para a avaliação do sistema em condições simuladas.

3. Desenvolvimento da solução:

- Treinamento do Modelo de Detecção:
 - O algoritmo YOLO será treinado com os dados coletados, seguindo a documentação do YOLO [Irie \(2020\)](#), ([Redmon, 2013–2016](#)).
- Integração no Ambiente Simulado:
 - O modelo treinado será incorporado a solução de carro autônomo, onde será responsável por identificar placas de trânsito em cenários urbanos predefinidos no CARLA. Os cenários predefinidos serão àqueles que contém placas de trânsito que possibilitem a validação da nossa solução.
 - O sistema de interação veículo-motorista será implementado para dar feedback visual em tempo real, informando sobre sinalizações detectadas e suas interpretações.
- Carro autônomo:
 - Percepção usando algoritmo YOLO.

- Planejamento de movimento funcional que possa evitar obstáculos estáticos e dinâmicos enquanto rastreia a linha central de uma faixa, ao mesmo tempo, em que lida com placas de trânsito (ex.: placa de parada).
 - * Implementar lógica de planejamento comportamental, bem como verificação de colisão estática, seleção de caminho e geração de perfil de velocidade.
- Controlar o veículo de modo a percorrer o trajeto predeterminado em uma certa velocidade, portanto, será necessário controle longitudinal e lateral.
 - * Implementar Controlador Longitudinal PID (Proporcional-Integral-Derivativo).
 - * Implementar Controle Lateral Geométrico - Controlador de Perseguição Pura.

4. Experimentação e Validação:

- Cenários de Teste:
 - A solução será experimentada em um trajeto predefinido que contenha as placas de trânsitos que possibilitem a validação da solução.
 - As variáveis independentes incluem o tipo de placa, e podem incluir condições ambientais e distância da câmera.
- Coleta de Dados Experimentais:
 - As variáveis dependentes, como a precisão na detecção de placas, o tempo de resposta do sistema e a eficácia do feedback visual, serão medidas.

5. Análise Crítica dos Resultados:

- Os dados coletados serão analisados de modo a avaliar se as detenções condizem com as suas classificações e 90% de confiança.
- A solução de carro autônomo terá seu desempenho avaliado pela conclusão bem sucedida do trajeto predefinido. O sucesso é dado pela não colisão com obstáculos, resposta a placas de trânsitos e conclusão de trajeto.

6. Conclusão e Propostas Futuras:

- Os resultados experimentais serão consolidados para verificar se os objetivos 1.3 foram atingidos, validando ou refutando a hipótese 1.2 inicial.
- Com base nas conclusões, serão sugeridas melhorias e direções para trabalhos futuros, como a implementação em ambientes reais, integração com outros sensores automotivos, uso de outros algoritmos, entre outros.

A partir das etapas apresentadas podemos identificar as Variáveis e Hipóteses Experimentais. Onde as Variáveis Dependentes:

- Taxa de acurácia na detecção de placas de trânsito.
- Tempo de processamento por imagem (latência).
- Eficácia do feedback visual percebido.

e Variáveis Independentes:

- Condições climáticas (ex.: ensolarado, chuva, neblina).
- Tipo de sinalização (ex.: limites de velocidade, placa de pare, etc.).
- Distância do veículo a placa.

As quais quando combinadas podem alterar os resultados dos experimentos. Desse modo, a Hipótese a ser testada se expande para incluir as etapas experimentais: **algoritmos de visão computacional baseados em YOLO, integrados a uma solução de carro autônomo, podem detectar e classificar placas de trânsito com precisão acima de 90% no simulador CARLA e oferecer feedback (assistência) visual em tempo real com baixa latência ao condutor, mesmo em diferentes condições.**

Essa metodologia oferece uma estrutura robusta para alcançar os objetivos 1.3 do trabalho, avaliando capacidades e limitações das soluções propostas e contribuindo para avanços no desenvolvimento de sistemas autônomos.

1.6 Organização do Trabalho

O presente trabalho está estruturado de maneira a guiar o leitor por suas diferentes etapas e contribuições. O Capítulo 1, contextualiza o tema, apresenta o problema, formula a hipótese, define os objetivos (geral e específicos), justifica a relevância do estudo, descreve a metodologia aplicada. Enquanto no Capítulo 2, são explorados os conceitos fundamentais relacionados a VAs, abordando aspectos como implementação, e classificações de automação. Também são discutidas as tecnologias essenciais e a arquitetura desses sistemas, e ainda aborda os ambientes de simulação, com destaque para o simulador CARLA. O Capítulo 3 foca na modelagem conceitual, introduzindo modelos cinemáticos e controles longitudinais e laterais, incluindo o controlador PID e o método de Perseguição Pura, amplamente aplicados em VAs. O Capítulo 4 apresenta a implementação dos conceitos e metodologias discutidos anteriormente, detalhando as etapas realizadas para atingir os objetivos do trabalho. O Capítulo 5 traz os resultados obtidos e as discussões correspondentes, analisando

a eficácia das soluções propostas e seu impacto no campo de VAs. No Capítulo 6, são apresentadas as considerações finais, que sintetizam os principais aprendizados e conclusões do trabalho, destacando as contribuições para a área e as limitações enfrentadas. O Capítulo 7 delineia a perspectiva de continuidade do estudo, sugerindo abordagens para aprofundar os conhecimentos adquiridos e explorar novas frentes de pesquisa. Finalmente, os capítulos (Apêndice A) e (Anexo A) fornecem materiais adicionais e de suporte que complementam as discussões realizadas ao longo do trabalho, garantindo um acesso completo às informações e recursos utilizados.

Adicionalmente, uma lista das principais siglas empregadas neste trabalho pode ser encontrada na Página 13.

2 Veículos Autônomos

Neste capítulo, apresentaremos os principais conceitos, fundamentos teóricos e tecnologias que alicerçam o desenvolvimento de Carros Autônomos. O objetivo é fornecer uma base sólida de conhecimento para embasar os capítulos subsequentes, detalhando os elementos essenciais que possibilitam a condução autônoma, desde a definição de níveis de automação até as técnicas de simulação aplicadas a esse contexto.

Inicialmente, será discutida a Classificação de Níveis de Automação (seção 2.1), explorando os diferentes graus de autonomia existentes e os critérios que os diferenciam. Essa análise é fundamental para compreender a evolução das tecnologias associadas aos VAs, bem como as capacidades e limitações de cada nível, desde sistemas que auxiliam o motorista até veículos que operam sem intervenção humana.

Na sequência, abordaremos a Tarefa de Condução (seção 2.2), um dos aspectos mais desafiadores no desenvolvimento de VAs. Essa seção detalhará os sistemas necessários para a execução da tarefa, incluindo sensores, algoritmos de percepção, e técnicas de detecção e classificação de objetos. Além disso, discutiremos a arquitetura geral que integra esses componentes e possibilita a tomada de decisões autônomas em tempo real.

Posteriormente, a seção dedicada à Simulação de Carros Autônomos (seção 2.3) destacará o papel estratégico dos simuladores na pesquisa e no desenvolvimento de VAs. Serão apresentados diferentes simuladores disponíveis atualmente, suas características e suas contribuições para o avanço da área, permitindo experimentação e validação em ambientes controlados antes da implementação em cenários reais.

Finalmente, este capítulo contará com uma seção sobre Trabalhos Relacionados (seção 2.4), na qual serão analisados estudos prévios que abordam soluções, desafios e avanços no contexto de simulação e detecção de objetos em tempo real para VAs. Essa análise crítica do estado da arte servirá como base para a execução do projeto, evidenciando lacunas, tendências e direções futuras na área.

Com essa estrutura, esperamos que este capítulo atue como um alicerce para o desenvolvimento do trabalho, consolidando os conceitos teóricos e referências práticas necessários para sustentar as soluções propostas ao longo deste estudo.

2.1 Classificação de Níveis de Automação

Esta subseção apresenta considerações importantes para classificar o nível de automação em um sistema de direção, com foco na atenção do motorista, nas ações necessárias e na capacidade do sistema de lidar com diversas tarefas de direção. Sabendo

disso, a *Society of Automotive Engineers* (SAE), traduzido para o português, refere-se a Sociedade de Engenheiros Automotivos, uma das principais associações globais que busca uma classificação. Essa classificação é apresentada na Figura 2, a qual a SAE dividiu os VAs em seis níveis de funcionalidade, que vão desde nenhum recurso de automação (nível 0) sem ODD, até automação completa, sem a necessidade de um condutor humano, (nível 5) e com ODD ilimitado. Usando a terminologia “sistemas de direção autônoma” para se referir a veículos que possuem algum tipo de recurso de direção autônoma (SAE, 2021, p. 30). Nessa classificação, os níveis 1 e 2 incluem alguns recursos, o nível 3 alcança automação limitada, onde o motorista pode abdicar do controle do veículo, desde que esteja disponível para intervir quando solicitado, enquanto no nível 4 essa disponibilidade de motorista só é necessária para evitar paradas de emergências desnecessárias.

As Figuras 2 e 3, apresentam graficamente as diferenças dos níveis de automação dos VAs e suas respectivas classificações (SAE, 2021, p. 28):

	LEVEL 0	LEVEL 1	LEVEL 2	LEVEL 3	LEVEL 4	LEVEL 5
O QUE O HUMANO NO BANCO DO MOTORISTA TEM QUE FAZER?	Você <u>está</u> dirigindo sempre que esses recursos de suporte ao motorista estão ativados - mesmo que seus pés não estejam nos pedais e você não esteja dirigindo	Você deve supervisionar constantemente esses recursos de suporte; você deve dirigir, frear ou acelerar		Você <u>não está</u> dirigindo quando esses recursos de direção automatizada estão ativados - mesmo se você estiver sentado no "banco do motorista"	Quando o recurso solicitar, você deve dirigir	Esses recursos de direção automatizada não exigirão que você assuma a direção
O QUE ESSES RECURSOS FAZEM?	Esses recursos são limitados para fornecer avisos e momentânea assistência	Esses recursos fornecer direção OU freio/ aceleração suporte para o motorista	Esses recursos fornecer direção E freio/ aceleração suporte para o motorista	Esses recursos podem conduzir o veículo sob condições limitadas e será não operar a menos que todos sejam necessários as condições são atendidas		Este recurso pode dirigir o veículo sob todas as condições
EXEMPLO RECURSOS	<ul style="list-style-type: none"> • freio de emergência • aviso de ponto cego • aviso saída de faixa 	<ul style="list-style-type: none"> • centralização da pista OU controle de cruzeiro adaptativo 	<ul style="list-style-type: none"> • centralização da pista E controle de cruzeiro adaptativo ao mesmo tempo 	<ul style="list-style-type: none"> • motorista de engarrafamento 	<ul style="list-style-type: none"> • táxi local sem motorista • pedais/ volante podem ou não estar instalados 	<ul style="list-style-type: none"> • igual ao nível 4, mas o recurso pode dirigir em qualquer lugar em todas as condições

COPYRIGHT © 2021 SAE INTERNATIONAL. THE SUMMARY TABLE WAS COPIED, EDITED, AND TRANSLATED BY DANIEL TERRA GOMES.

ESTES SÃO RECURSOS DE SUPORTE AO MOTORISTA ESTES SÃO RECURSOS DE DIREÇÃO AUTOMATIZADA

Figura 2 – Níveis de Automação de condução PT-BR

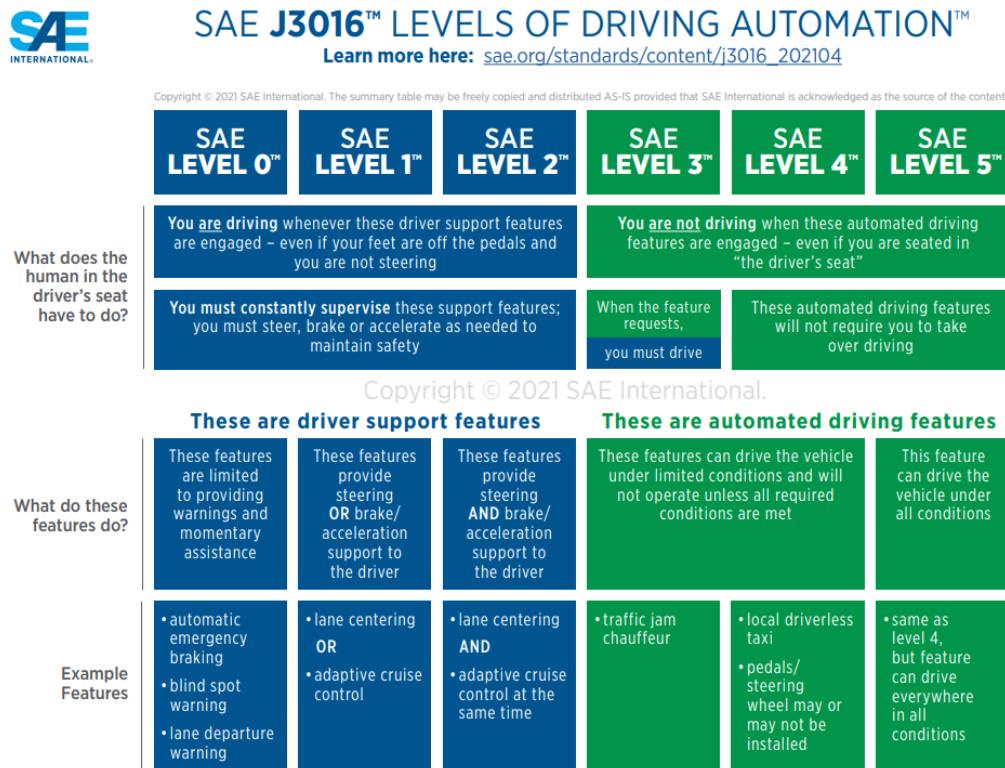


Figura 3 – SAE J3016TM levels of driving automation (SAE, 2021)

Além dessas classificações entre tipos veículos, ainda é possível classificá-los nas seguintes categorias de tipos de automação: Sistema de Assistência Avançada ao Motorista (ADAS) sigla do inglês *advanced driver-assistance system (ADAS)*, ou Condução Automática, Mobilidade Como Serviço (MaaS) sigla do inglês *Mobility as a service (MaaS)*.

Como apresentado, nas Figuras 2 e 3, nos níveis 1, 2 e 3 o condutor precisa estar preparado para caso o sistema precise de algum tipo de ajuda do condutor, esses níveis são vistos como uma espécie de remendo, pois funcionam como um ajudante para suprir uma demanda que os sistemas autônomos, na totalidade, ainda não podem cumprir com maestria (Lage, 2019). Dado que o objetivo com o avanço da tecnologia é que o condutor humano torna-se cada vez menos necessário, e como apresentado no nível 5, visto na Figura 3, é possível até mesmo a retirada do volante do veículo.

Na atualidade, empresas líderes do setor de veículos autônomos, como a Tesla, ainda trabalham com nível 2 de direção autônoma referente ao ADAS para veículos vendidos para a população. (Lage, 2019).

Por outro lado, no início do ano de 2023, a Mercedes-Benz, no seu portal de mídias, afirma ser a primeira empresa a alcançar a marca de direção autônoma SAE nível 3 para o mercado dos Estados Unidos, sendo o estado de Nevada o primeiro a concordar com as regulamentações para a navegação desses tipos de veículos em seu território. A Mercedes

afirma, também, que já em 2024 terá os seus primeiros veículos *DRIVE PILOT*, em português “condutor”, disponíveis para o mercado norte-americano ([Mitropoulos, 2023](#)).

“No mundo moderno, o tempo é um dos bens mais preciosos, e devolver o tempo aos nossos clientes é um elemento central em nossa estratégia de construir os carros mais desejados do mundo. O nosso *DRIVE PILOT* dá um grande passo para o conseguir e coloca-nos na vanguarda da inovação no campo crucialmente importante da condução autônoma. O *DRIVE PILOT* demonstra mais uma vez que nosso pioneirismo faz parte do nosso DNA. A certificação em Nevada marca o início de seu lançamento internacional e, com ela, o início de uma nova era.”

Diz o Markus Schäfer, Membro do Conselho de Administração do Mercedes-Benz Group AG, Diretor de Tecnologia, responsável por Desenvolvimento e Compras ([Mitropoulos, 2023](#)).

Adicionalmente, algumas empresas, como Waymo e Cruise, atualmente operam serviços de carona com veículos com automação de nível 4 nos Estados Unidos da América (EUA) - isso significa que os carros podem operar sem motorista ao volante sob certas condições, como em uma área de serviço designada, portanto, já mapeadas e entendida pelo algoritmo e software de controle dos veículos ([Houser, 2023](#)).

A seguir forneceremos as definições detalhadas para seis níveis de automação de veículos, variando de nenhuma automação de direção (Nível 0) a automação total de direção (Nível 5), no contexto de veículos a motor, definidos pela SAE ([SAE, 2021](#)):

1. **Nível 0** – Sem automação de condução: não existe nenhum tipo de auxílio ao motorista e nenhuma presença/atuação de tecnologia de condução autônoma, ou assistência.
2. **Nível 1** - Assistência ao Motorista: no nível 1, o motorista é assistido apenas em relação à velocidade do veículo, um exemplo prático seria o piloto automático, que mantém a velocidade do veículo constante conforme o gosto do motorista. Neste caso, o condutor deve continuar a frear, acelerar e direcionar o veículo. Um segundo exemplo seria: o recurso de assistência ao estacionamento, executa automaticamente as ações de controle de movimento do veículo necessárias para estacionar um veículo, enquanto o motorista executa as ações de controle de movimento do veículo longitudinal e supervisiona o recurso de estacionamento.
3. **Nível 2** - Automação de Condução parcial: nesta fase, o veículo já consegue realizar ações de forma autônoma, como frear, acelerar e parar o veículo em uma direção, como a tecnologia chamada ACC (Adaptive Cruise Control). No nível 2, o condutor

continua a ser responsável pela condução e exige que o condutor esteja atento à condução e retome a condução em situações de perigo. Um exemplo prático seria, como visto: o recurso de assistência ao estacionamento, mas dessa vez, executa automaticamente as ações de controle de movimento lateral e longitudinal do veículo necessárias para estacionar um veículo sob a supervisão do motorista.

4. **Nível 3 - Automação Condicional de Condução:** consiste em veículos que são capazes de se mover de forma independente, tanto na direção, aceleração e frenagem. Neste nível de condução, o condutor pode realizar outras atividades enquanto o carro segue autonomamente a sua rota, mas por vezes é acionado para assumir a condução por um curto período ou para assumir o controlo total em situações de risco. Nesse cenário, um *automated driving system* (ADS) consegue continuar a executar o *dynamic driving task* (DDT) por pelo menos vários segundos após fornecer ao usuário pronto para *fallback*¹; uma solicitação para intervir. Espera-se então que o usuário pronto para o *fallback* do DDT retome a operação manual do veículo ou alcance uma condição de risco mínimo se ele/ela/elu determinar necessário.
5. **Nível 4 - Alta Automação de Condução:** o veículo controla todas as tarefas que antes eram do condutor, sem necessidade da atenção do mesmo. Desse modo, o veículo fica encargo de executar todo o DDT em uma localidade, ao sofrer uma falha de sistema relevante para o desempenho do DDT. Em resposta, o *ADS-dedicated vehicle* (ADS-DV) realiza o *fallback* do DDT ligando os piscas de emergência, manobrando o veículo para o acostamento e estacionando-o, antes de chamar automaticamente a assistência de emergência. Nesse nível, o ADS consegue atingir automaticamente uma condição de risco mínimo quando necessário.
6. **Nível 5 - Automação de Condução completa:** permite que o veículo elimine a necessidade de um motorista humano, com todos os controles e tarefas de direção realizados por um sistema autônomo. O desempenho do veículo é sustentado, incondicionalmente, por um ADS responsável por todo o DDT e *fallback* do DDT sem qualquer expectativa de que um usuário precise intervir.

A Figura 4 é a representação didática de todos os diferentes níveis (0-5) de automação SAE apresentados na subseção 2.1:

¹ Um plano de backup ou estratégia de contingência; uma alternativa que pode ser usada se algo der errado com o plano principal; um recurso.

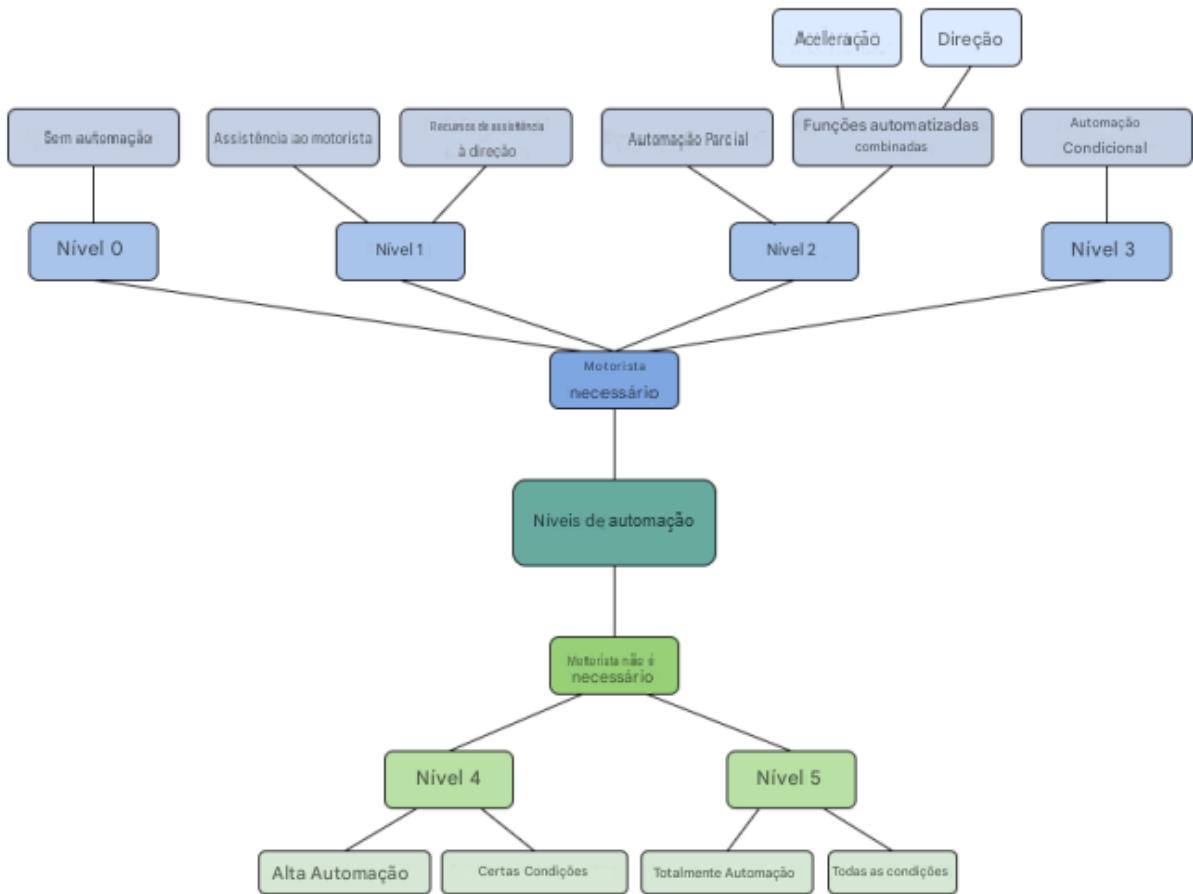


Figura 4 – Níveis de Automação ([Parekh Nishi Poddar, 2022](#)).

Fazendo uma descrição rápida da Figura 4 podemos dizer:

Nível 0 - Sem automação.

Nível 1 - Assistência ao motorista e funcionalidades de assistência.

Nível 2 - Automação parcial e funcionalidades de assistência combinadas.

Nível 3 - Automação condicionada.

Nível 4 - Alta automação em certas condições.

Nível 5 - Totalmente autônomo em todas as condições.

2.2 Tarefa de Condução

Esta subseção aborda as três sub tarefas principais que compõem a atividade de direção: Percepção, Planejamento de Movimento e Controle do Veículo. Na Figura 5, são apresentadas as camadas correspondentes a essas sub tarefas, onde cada uma executa operações e decisões específicas. A interação entre essas camadas em diferentes contextos de uso é fundamental para alcançar automação na direção, como veremos nesta subseção.

A seguir, detalharemos cada sub tarefa, ressaltando a importância do desempenho contínuo ao dirigir um veículo, conforme descrito por (Khan *et al.*, 2022, p. 4):

1. **Percepção:** tem em vista fazer sentido do ambiente em que o veículo se localiza, recorrendo ao uso de recursos presentes no veículo como, sensores (apresentados na subseção 2.2.1.3), dados armazenados, localização a partir de mapas, redes de conexão à internet, etc.
2. **Planejamento:** a sub tarefa de Planejamento visa estabelecer as ações necessárias para que o VA alcançar o objetivo estabelecido de modo a satisfazer o *Operational Design Domain* (ODD)².
3. **Controle:** Esta sub tarefa tem em vista executar o planejamento deferido pela camada anterior, predizendo os controles para execução da tarefa designada, faz o controle dos freios, acelerador e volante, atua na execução da trajetória e controle de toda a execução.



Figura 5 – Componentes em camadas para alcançar automação na condução (Khan *et al.*, 2022, p. 4).

Como podemos identificar na Figura 5 a tarefa de direção depende do cumprimento de cada sub tarefa de direção, de maneira interconectada, já que uma sub tarefa depende

² O *Operational Design Domain* (ODD), traduzida para o português, refere-se ao Domínio de Design Operacional de um VA. É um conceito crítico no domínio dos carros autônomos, visando definir as condições operacionais específicas sob as quais um determinado sistema autônomo é projetado para funcionar de forma confiável. Este conceito serve como um elemento crucial para garantir a segurança e a eficácia dos VAs. Abrangendo características ambientais, temporais e situacionais que influenciam o desempenho de um carro autônomo. O delineamento preciso do ODD é essencial para estabelecer os limites dentro dos quais o sistema de condução autônoma pode operar com segurança e eficácia desejada pelos seus desenvolvedores (SAE, 2021, p. 33).

da execução de maneira satisfatória da tarefa anterior. Na Figura 6, podemos compreender os componentes envolvidos e com quem eles se comunicam em cada sub tarefa.

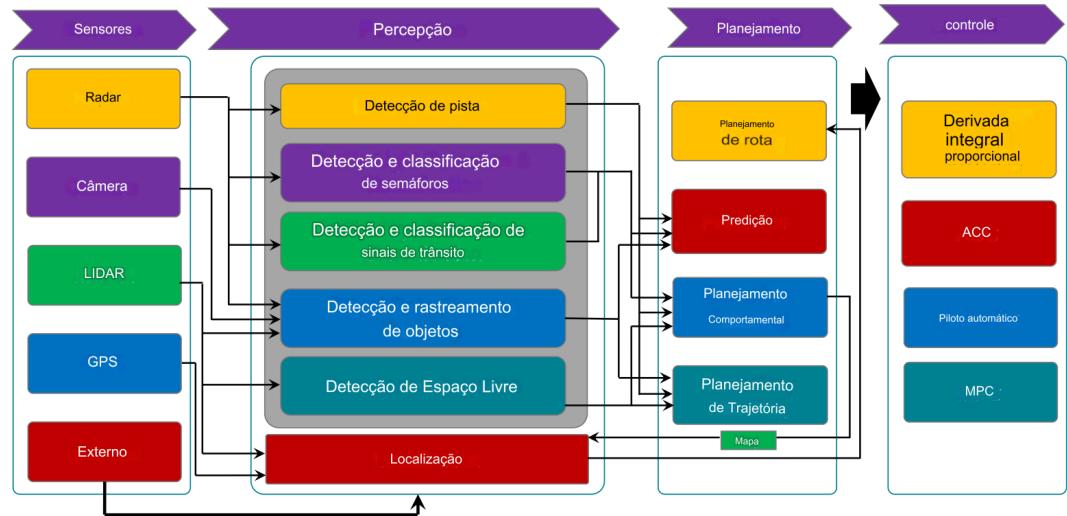


Figura 6 – Relação dos componentes entre sub tarefas e camadas. ([Khan et al., 2022](#), p. 4).

A compreensão que há uma relação entre diferentes componentes no VA, nos ajudará a compreender conceitos relacionados a cada uma das sub tarefas e seus componentes. Dos quais trabalharemos a fundo durante este trabalho.

2.2.1 Tecnologias Essenciais para as Tarefas de Condução

Esta subseção iniciará uma análise de como a tarefa de condução é executada. Concentrando-se, especificamente, na tarefa de percepção, descrevendo os requisitos, componentes, desafios associados, e soluções a este aspecto crucial da tarefa de direção em carros autônomos.

2.2.1.1 Percepção e Tarefas Relacionadas

A tarefa de percepção pode ser conceptualmente dividida em dois componentes fundamentais. Primeiramente, destaca-se a necessidade de compreender o ambiente no entorno do veículo, de modo a compreender o estado atual do ambiente no qual o veículo está localizado. Em segundo lugar, a exigência de tomar decisões informadas durante a tarefa condução.

Desse modo, para um carro autônomo conseguir compreender o seu entorno se faz necessário que o veículo consiga identificar e categorizar as informações que está recebendo do ambiente dos seus sensores (Vistos 11). Nesse processo de identificação, se faz necessário categorizar os elementos identificados, de modo a separá-los em dois principais grupos:

Elementos Estáticos e Dinâmicos. Essa categorização auxiliará no processo de decisão do VA, e podemos descrever esses elementos das seguintes formas (Zheng et al., 2023):

Elementos Estáticos - Os elementos estáticos abrangem características fixas da estrada, como: marcação de faixas de pedestres e sinalizações de trânsito. Além disso, elementos fora da pista, como: meios-fios e placas de sinalização, árvores, etc (Zheng et al., 2023, p. 3). Sendo esse o grupo de elementos foco deste trabalho.

Elementos Dinâmicos - Os elementos dinâmicos envolvem entidades em movimento na via. Por exemplo: veículos, incluindo caminhões, ônibus, carros, motos, etc. Assim como, bicicleta, pedestres, e animais (Zheng et al., 2023, p. 17).

A categorização de elementos é uma atividade dinâmica e constante, pois algumas entidades podem variar de uma classificação para outra a qualquer momento, como os semáforos que podem ter 3 estados diferentes em um objeto. Adicionalmente, a essa necessidade do VA de compreender o seu entorno. Também se faz necessário a compreensão do estado interno do veículo. Devido a isso, temos a tarefa relacionada de localização de ego, que se trata de saber a localização precisa do veículo usando sensores de estado interno. Entretanto, não fazem parte do escopo deste estudo, mas é importante saber sobre a sua existência para possíveis trabalhos futuros. Desse modo, conhecer a posição, o movimento do veículo, e o estado do ambiente é crucial para tomar decisões de condução informadas e seguras. A localização do ego, por exemplo, depende de dados oriundos do *Global Navigation Satellite System* (GNSS), traduzido para o português, refere-se a Sistemas Globais de Navegação por Satélite, das *Inertial Measurement Units* (IMUs), traduzido para o português, referem-se as Unidades de Medição Inercial, e dos sensores de odometria. Quanto ao estado do ambiente são usados sensores como: Câmeras, Radar, Lidar, e Ultra-sônicos. Sendo as Câmeras o sensor foco deste trabalho para a detecção e classificação de objetos. Todavia, é importante ressaltar que os sensores dos VAs alcançam um melhor desempenho em múltiplos cenário e situações na tarefa de entendimento do ambiente quando combinados, tal exemplo de combinação pode ser vista nas Figuras 12 (Zheng et al., 2023, p. 9). Neste trabalho daremos foco nas câmeras, por ser o sensor com melhor pontuação (4) na classificação de objetivos, como visto na Figura 12.

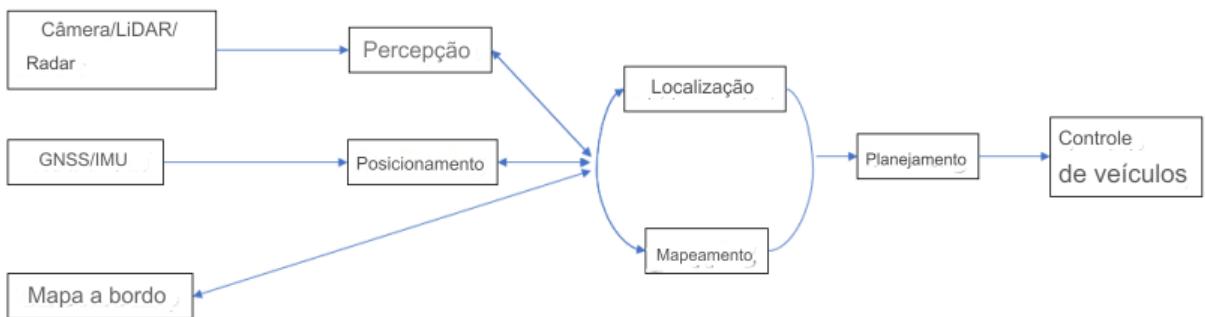


Figura 7 – Componentes funcionais de um sistema de direção autônomo e arquitetura ([Zheng et al., 2023](#), p. 2).

Por fim, outra parte relacionada a navegação dos VAs, visto na Figura 7, são os mapas cadastrados no armazenamento interno (ex.: coordenadas de caminho) e mapas como o *Google Maps*, que fornecem informações adicionais ou totais do setor no qual o VA está operando, possibilitando a navegação em setores totalmente desconhecidos ou uma navegação controlada em setores já totalmente conhecidos pelo sistema ([Zheng et al., 2023](#), p. 2). Esses mapas são usados juntamente com a tarefa de percepção, visto que o veículo deve seguir o caminho estipulado enquanto a percepção faz sua tarefa de identificar e classificar os elementos durante o trajeto. Esses mapas, cadastrados internamente, são usados em soluções autônomas que visam percorrer trajetórias predeterminadas múltiplas vezes, como a nossa solução.

2.2.1.2 Detecção e Classificação de Objetos

A evolução na área de visão computacional trouxe significativos avanços na detecção de objetos, com destaque para a arquitetura You Only Look Once (YOLO), proposta por [Redmon et al. \(2016\)](#). Essa abordagem se diferencia por tratar o processo de detecção e classificação de objetos como um problema único de regressão, unificando a identificação de áreas de interesse e a atribuição de classes. Essa estratégia reduz a complexidade computacional e permite o reconhecimento em tempo real em streaming de vídeo com menos de 25 milissegundos de latência no YOLO ([Redmon et al., 2016](#), p. 1), característica crucial para aplicações como VAs. O YOLO oferece uma solução integrada para detecção, refinamento e classificação que observa a imagem apenas uma vez (You Only Look Once), reduzindo assim a complexidade e os custos computacionais envolvidos no processo ([Redmon et al., 2016](#)).



Figura 8 – O Sistema de Detecção YOLO. O processamento de imagens com YOLO é simples e direto. O sistema (1) redimensiona a imagem de entrada para 448×448 , (2) executa uma única rede convolucional na imagem e (3) limita as detecções resultantes pela confiança do modelo. ([Redmon et al., 2016](#), p. 1).

O Funcionamento da Arquitetura YOLO inicia seu processamento redimensionando as imagens para o tamanho esperado pela rede neural convolucional como visto na Figura 8. Em seguida, a imagem é submetida a uma rede que gera predições para as áreas de interesse, as classes correspondentes e as probabilidades associadas. Por fim, os valores redundantes são eliminados por meio de um filtro que aplica a técnica de Non-Maximum Suppression (NMS), mantendo apenas as predições com maior confiança ([Redmon et al., 2016](#), p. 2).

Ao combinar a identificação de áreas de interesse com a classificação em um único processo, o YOLO se torna um dos algoritmos mais rápidos de sua classe, embora, em algumas circunstâncias, sacrifique um pouco de precisão em relação a métodos mais lentos. A arquitetura do YOLO também se diferencia por considerar os dados globais da imagem, facilitando a distinção entre o plano de fundo e os objetos de interesse, comete menos da metade do número de erros comparação ao Fast Region-based Convolutional Networks (Fast R-CNN) ([Redmon et al., 2016](#), p. 2).

Desde sua primeira versão, o YOLO passou por melhorias significativas. O tornando uma referência em detecção de objetos. Entre essas melhorias, destaca-se o suporte para maior número de classes, detecção aprimorada de objetos pequenos e avanços em precisão e eficiência, como observado nas versões YOLOv2, YOLOv3 e YOLOv4 ([Redmon; Farhadi, 2016](#); [Redmon; Farhadi, 2018](#); [Bochkovskiy; Wang; Liao, 2020](#)).

O progresso da série YOLO não se limitou a apenas detecção de objetos. Ao longo do tempo, a metodologia também passou a influenciar áreas correlatas, como segmentação de instâncias, rastreamento de múltiplos objetos, análise comportamental, reconhecimento facial, entre outras. Isso reflete sua importância como um pilar tecnológico para aplicações em direção autônoma, robótica industrial, autenticação de identidade, saúde inteligente e vigilância visual ([Wang; Liao, 2024](#)).

A capacidade do YOLO de se adaptar a diferentes contextos e sua contínua evolução são características fundamentais que sustentam sua popularidade. Até mesmo as versões mais recentes, como o YOLOv7 e o YOLOv8, continuam a superar desafios em cenários

complexos, como ambientes com objetos pequenos ou parcialmente ocluídos. Além disso, a versão YOLOv10 representa uma convergência entre precisão e eficiência computacional, mostrando que a pesquisa em torno deste framework está longe de atingir seu limite (Wang; Liao, 2024). Para questão de comparativo, a Figura 9 visa ilustrar esse ganho de uma versão de YOLO para outra.

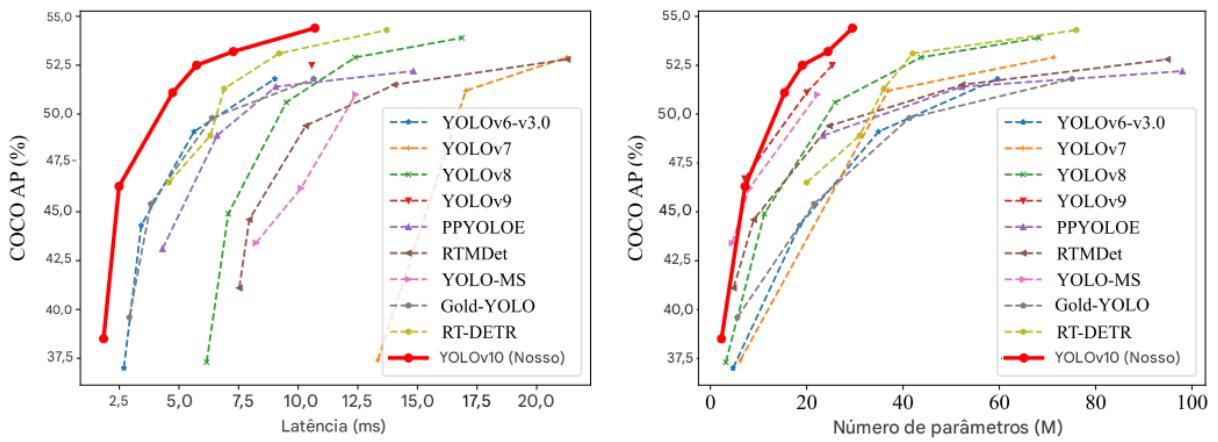


Figura 9 – Comparações com outros em termos de trade-offs de latência-precisão (esquerda) e tamanho-precisão (direita). Medimos a latência de ponta a ponta usando os modelos oficiais pré-treinados. (Wang *et al.*, 2024, p. 1).

2.2.1.3 Componentes de Hardware

Esta subseção é uma visão geral sobre os diferentes sensores (vistos na Figura 11) os quais são peças fundamentais para que os VAs possam compreender o seu entorno. Os sensores apresentados nesta subseção são peças fundamentais dos componentes presentes nos VAs, sendo a porta de entrada de dados para que esses veículos possam fazer sentido das condições do ambiente à sua volta. Os sensores mencionados são organizados conforme mostra a Figura 11. Iniciaremos com a definição de que sensores são dispositivos que medem ou detectam propriedades do ambiente, ou do estado (ego) do veículo, e são comumente classificados em sensores exteroceptivos (gravam propriedades do ambiente) e proprioceptivos (gravam propriedades do estado do VA), definidos a seguir entre Sensores proprioceptivos e Sensores exteroceptivos (Ignatious; Hesham-El-Sayed; Khan, 2022, p. 737):

- Sensores proprioceptivos:** são sensores de estado interno, registram o estado dinâmico de um sistema dinâmico e detectam dados internos como força, taxa angular, pressão da roda, voltagem da bateria e assim por diante. Como exemplo de sensores proprioceptivos, então, a unidades de medição inercial, codificadores, sensores inerciais (giroscópios e magnetômetros) e sensores de localização.

2. Sensores exteroceptivos: são sensores de estado externo, por outro lado, percebem e coletam informações do ambiente do veículo, como medições de distância ou intensidade de luz de objetos. Os sensores externos incluem *câmeras*, detecção e alcance de rádio *Radar*, detecção e alcance de luz *LiDAR* e sensores ultrassônicos, esses sensores podem ser vistos na Figura 10, e seus respectivos posicionamentos na Figura 11.



Figura 10 – Componentes de um veículo autônomo (Singh, 2022, p. 15).

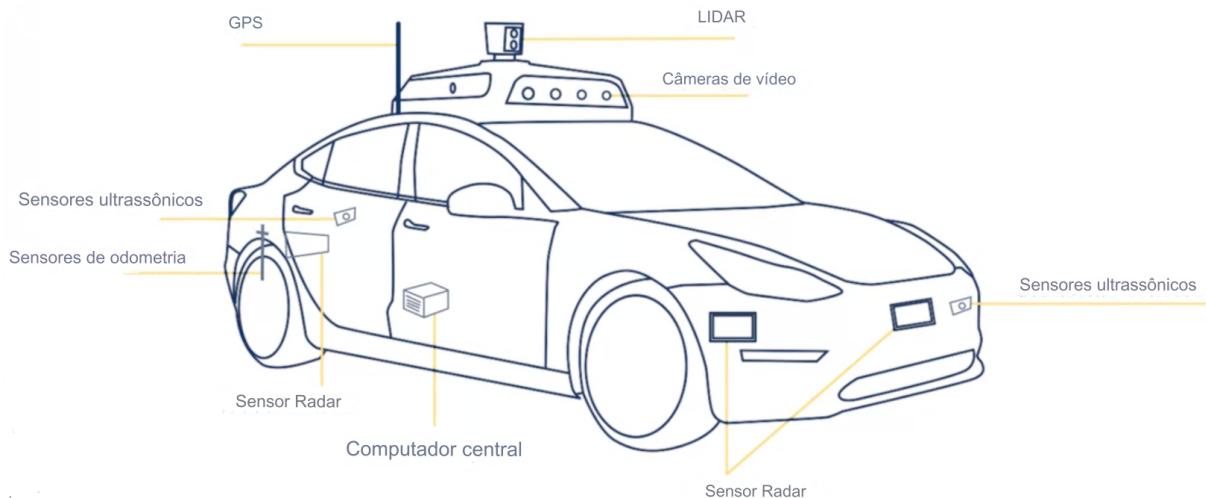


Figura 11 – Tipos de sensores em VAs e seus posicionamentos (University of Toronto, 2018a, Week 2 - Lesson 1: Sensors and Computing Hardware. 9min00s).

A integração dos sensores (conforme demonstrado nas Figuras 11 e 12) é fundamental para viabilizar o desempenho satisfatório dos VAs em uma ampla gama de

cenários que demandam percepção, tais como: condições de chuva em dias ensolarados, terrenos escorregadios com lama, neblina, entre outros. Considerando que a percepção está sujeita a incertezas provenientes dos sensores, como desafios de visibilidade, leituras de *Global Positioning System* (GPS), traduzido para o português, refere-se ao Sistema de Posicionamento Global, corrompidas e retornos ruidosos dos sensores, a combinação destes componentes sensoriais, conforme ilustrado na Figura 12, se destaca como fator crucial para aprimorar o desempenho dos VAs em cenários variados.

Parâmetros	Camera	Radar	LIDAR	Ultra-sonic	Fusão
Campo de visão	3	4	3	2	4
Faixa	3	4	4	1	4
Resolução de velocidade	2	3	4	1	4
Resolução Angular	4	3	2	1	4
Tempo Adverso	2	2	3	2	4
Perturbação de escuridão/luz	3	4	4	4	4
Classificação de objetos	4	3	2	2	4
Detecção de todas as superfícies de objetos	3	3	3	3	4
Esquema de cores: 1 → Pior, 2 → Medioce, 3 → Aceitável, 4 → Bom					

Figura 12 – Adequação de sensores para diferentes situações (Khan *et al.*, 2022, p. 6).

O primeiro componente sensorial que discutiremos são as **Câmeras**. Sendo essas um dos tipos de sensores de estado externo fundamentais, capturando informações detalhadas do ambiente. As câmeras possibilitam a produzir imagens nítidas dos arredores, detectando as luzes emitidas a partir de uma superfície fotossensível (plano de imagem) usando uma lente de câmera (colocada na frente do sensor) (Ignatious; Hesham-El-Sayed; Khan, 2022, p. 738). Os VAs possuem esses sensores de luz visível para fornecer uma visão de 360 graus do ambiente. As câmeras são ótimos na detecção e reconhecimento de objetos, fornecendo detalhes mais ricos e ajudando a entender os objetos sem ou com profundidade, que geralmente não são detectados por outros tipos de sensores (Yao *et al.*, 2024, p. 4), podemos entender essa diferença a partir da Figura 12.

A seguir temos, o **LiDAR**, que corresponde à sigla em inglês para *Light Detection and Ranging*, em português Detecção e Alcance de Luz, foi concebido pela primeira vez na década de 1960 e, desde então, desempenha um papel fundamental no mapeamento de terrenos na esfera aeronáutica e aeroespacial (Ignatious; Hesham-El-Sayed; Khan, 2022, p. 738). No contexto dos VAs, os sensores LiDAR se destacam como instrumentos que empregam a luz como meio para mensurar distâncias. Este processo se dá através do cálculo do tempo necessário para a luz ser refletida no receptor (Mao *et al.*, 2023, p. 1911). Esses sensores são estrategicamente posicionados no VA, conforme ilustrado na Figura 11.

Nosso próximo sensor é datado para antes da Segunda Guerra Mundial, onde surgiu

o *Radio Detection and Ranging*, conhecido como **Radar** e em português *Detecção e alcance de rádio*, cuja essência reside na emissão de ondas eletromagnéticas na região de interesse e na recepção das ondas dispersas (reflexões) provenientes de alvos. Essas reflexões são então processadas para determinar informações relevantes (Ignatious; Hesham-El-Sayed; Khan, 2022, p. 738). O Radar opera de maneira análoga ao Lidar, porém, em vez de luz, utiliza ondas de rádio para medir a distância entre o sensor e os objetos. Além disso, o Radar pode observar a velocidade relativa entre o sensor e o objeto por meio da medição do deslocamento Doppler³, permitindo distinguir entre objetos estacionários e em movimento. Esse recurso também possibilita descartar objetos em movimento durante a construção do mapa. Quando comparado ao Lidar, o Radar apresenta vantagens em termos de custo mais baixo, menor consumo de energia e menor sensibilidade às condições atmosféricas, tornando-o mais adequado para aplicações externas. Entretanto, é importante notar que o Radar possui uma resolução de medição inferior e suas detecções são mais espaçadas do que as do Lidar. Consequentemente, a combinação de dados provenientes do Radar e do Lidar no contexto da associação de dados torna-se mais desafiadora, resultando em um mapeamento Tridimensional (3D) menos preciso (Zheng *et al.*, 2023, p. 8).

Finalizando os sensores exteroceptivos, para nossa introdução, temos os sensores **Sonares/Ultrassônicos**, que empregam a técnica medição de Time-of-Flight (TOF), em português, medição do tempo de voo para determinar a distância até objetos, funcionando por meio do envio e recebimento de ondas sonoras. Similar ao Radar, esse tipo de sensor adquire informações de maneira dispersa, resultando em uma extração imprecisa de recursos e em um tempo de processamento prolongado. Sua aplicabilidade é, portanto, restrita em contextos que envolvem veículos de alta velocidade. Adicionalmente, é importante destacar que os Sensores Sonares/Ultrassônicos possuem um alcance de detecção limitado, podendo ser impactados pelo ruído ambiental e por outras plataformas que utilizam ultrassom com a mesma frequência (Zheng *et al.*, 2023, p. 8).

Iniciando as apresentações dos principais sensores *proprioceptivos*, e seus papéis fundamentais na obtenção de informações cruciais para a navegação veicular. Destacam-se entre esses sensores os GNSS e IMUs responsáveis por mensurar diversos parâmetros, tais como posição, velocidade, direção, taxa de rotação angular e acelerações do veículo ego (Zheng *et al.*, 2023, p. 2). É digno de nota que a fusão de dados provenientes da IMU com sensores como Câmera ou Lidar contribui significativamente para a estimativa de estado, abrangendo elementos como posição, velocidade e atitude do veículo. A utilização da IMU, em particular, torna as atitudes, especialmente o direcionamento, observáveis. Além disso, a integração das medições provenientes da IMU demonstra melhorias substanciais

³ O deslocamento Doppler refere-se à mudança na frequência de ondas (como ondas sonoras, luz ou ondas eletromagnéticas) percebida por um observador em relação à fonte dessas ondas em movimento. Esse fenômeno é nomeado em homenagem ao físico austriaco Christian Doppler, que primeiro descreveu o efeito em 1842 (Fowler, 2009, p. 1).

no desempenho do rastreamento de movimento durante períodos nos quais as observações são interrompidas de maneira momentânea (Zheng *et al.*, 2023, p. 9).

Associados a esses dispositivos, encontram-se, também, os sensores de odometria das rodas, cuja função é rastrear informações provenientes das rodas para estimar tanto a velocidade quanto a taxa de mudança de direção do veículo, elementos cruciais para o eficaz controle veicular.

Em conclusão, analisaremos o posicionamento dos sensores nos VAs, apresentados anteriormente na Figura 11 e como esse posicionamento auxilia o VA para um melhor entendimento da condição do ambiente em que navega. Desse modo, o objetivo é posicionar estratégicamente os sensores para obter uma visão completa do ambiente, atendendo às necessidades específicas de manobras. Nesse ponto as atenções se voltam para os requisitos de cobertura, destacando onde os sensores devem ser colocados para garantir entrada suficiente para as tarefas de direção, essa análise de cobertura destaca a necessidade de sensores de longo alcance com campos de visão angulares mais curtos e sensores de amplo campo angular para percepção onidirecional, a disposição desses sensores é apresentada na Figura 13.

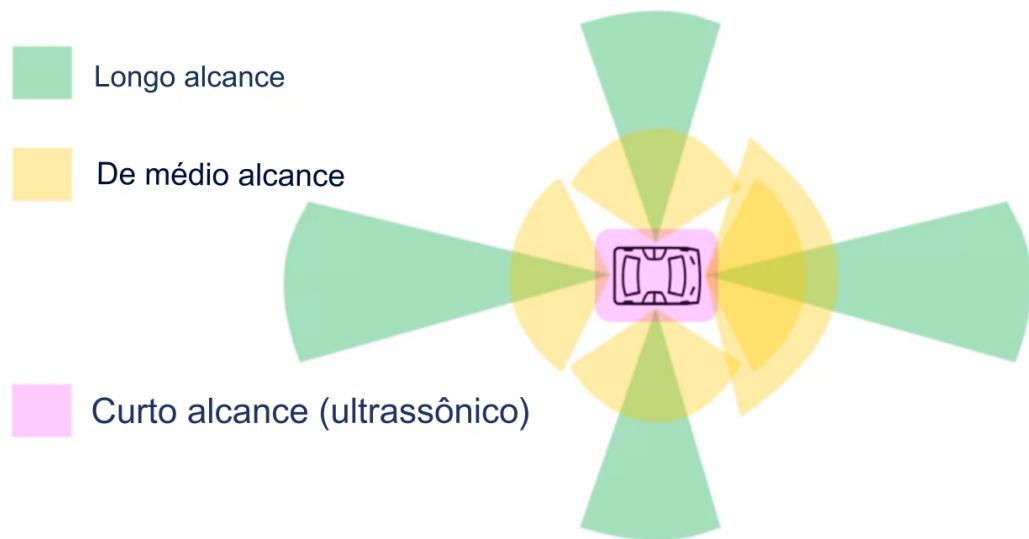


Figura 13 – Cobertura geral dos sensores ([University of Toronto, 2018a](#), Week 2 - Lesson 2: Hardware Configuration Design. 9min00s).

Conforme os cenários se tornam mais complexos, a demanda por cobertura de sensor de 360 graus em curtas distâncias e alcances longitudinais estendidos é evidente. A configuração de sensores na Figura 13 ilustra a necessidade de uma variedade de sensores, incluindo *Ultrassônico & Sonar* necessário para entrada 3D em curto alcance, útil em manobras de estacionamento e detecção de objetos próximos, *LiDAR* essencial para entrada 3D em todas as condições, permitindo a detecção precisa e abrangentes de obstáculos e veículos à frente, *Radar* importante para avaliar e identificar a velocidade, movimentação

e posição de veículos em todos os ambientes, *Câmera* necessária para identificar objetos, veículos, pedestres e sinais de trânsito, tem com campo de visão amplo para monitorar movimentos em todas as direções, especialmente em interseções. O alcance desses sensores é modificado com forme a necessidade do fabricante ([University of Toronto, 2018a](#), Week 2 - Lesson 2: Hardware Configuration Design).

2.2.2 Arquitetura da Tarefa de Condução

Nesta subseção, exploraremos uma arquitetura de software modular representativa para VAs. A arquitetura utiliza informações de componentes de hardware para alcançar o objetivo de direção autônoma. A pilha de software é composta por cinco módulos essenciais: percepção do ambiente, mapeamento do ambiente, planejamento de movimento, controle do veículo e o supervisor do sistema, vistos na Figura 14, e [7](#). Teremos como objetivo fornecer uma decomposição inicial de cada módulo presente na Figura 14, enfatizando as entradas, computações e saídas de cada segmento presente:

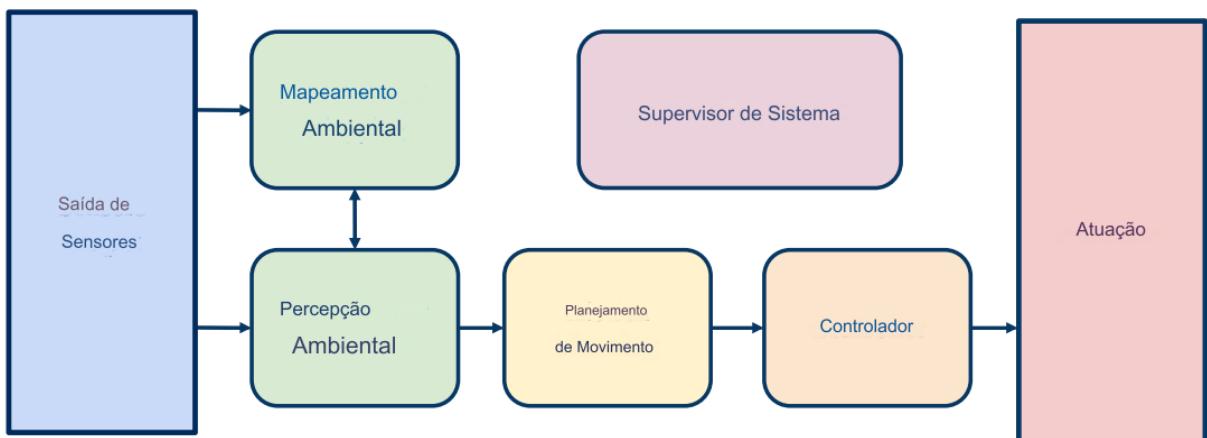


Figura 14 – Arquitetura de Software ([University of Toronto, 2018a](#), Week 2 - Lesson 3: Software Architecture. 3min49s).

- **Saída Sensores:** a implementação da direção autônoma tem como ponto de partida a utilização de sensores, conforme discutido na subseção [2.2.1.3](#) e visualizado na Figura 12. Múltiplos sensores são desejáveis para atingir os mais altos níveis de fidelidade e confiabilidade na execução da condução autônoma foram apresentados, incluindo GPS, IMU, sensor de odometria da roda, LIDAR e câmeras ([Reinholtz et al., 2007](#), p. 2). Esses dispositivos são responsáveis por coletar dados brutos do ambiente ao redor do veículo, os quais são posteriormente analisados e processados nos módulos subsequentes da arquitetura de software dos VAs. Essa abordagem sensorial é fundamental para a tomada de decisões e execução de tarefas autônomas, tornando-se um componente crucial no desenvolvimento de sistemas de direção autônoma.

- **Mapeamento:** o módulo de mapeamento do ambiente gera diferentes tipos de representações do ambiente ao redor do carro autônomo, aprimorando a prevenção de colisões e o planejamento de movimentos. Três tipos principais de mapas são discutidos: mapa de grade de ocupação, mapa de localização e mapa detalhado de estradas ([University of Toronto, 2018a](#), Week 2 - Lesson 3: Software Architecture. 6min24s):

Primeiramente, o Mapa de Grade de Ocupação, construído principalmente a partir de dados LIDAR, o mapa de grade de ocupação, visto na Figura 15, identifica objetos estáticos no ambiente como, por exemplo, árvores e prédios. Para isso, filtros são aplicados aos dados LIDAR para remover pontos não relevantes, e o mapa representa o ambiente como um conjunto de células de grade com probabilidades associadas de ocupação ([Zheng et al., 2023](#), p. 14).

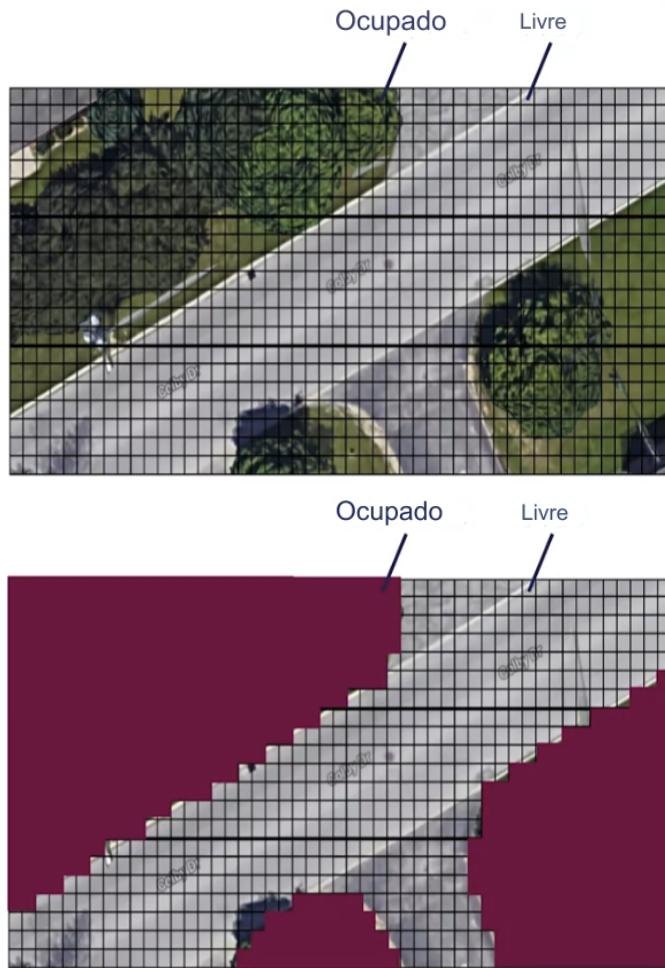


Figura 15 – Arquitetura de Software ([University of Toronto, 2018a](#), Week 2 - Lesson 4: Environment Representation. 3min30s).

A seguir, Mapa de Localização, construído a partir de dados LIDAR ou câmera, combinados para criar uma representação do ambiente em nuvem de pontos, auxiliando na estimativa do estado ego, visto na Figura 16. Os dados do sensor são comparados

ao mapa localização durante a condução para determinar o movimento do veículo em relação ao mapa já cadastrado no VA (Zheng *et al.*, 2023, p. 14).



Figura 16 – Arquitetura de Software ([University of Toronto, 2018a](#), Week 2 - Lesson 4: Environment Representation. 2min07s).

Por fim, Mapa Detalhado de Estradas, visto na Figura 17 fornece informações sobre segmentos de estrada, capturando sinais e marcações de faixa para o planejamento de movimentos, apresentado a seguir. Esse tipo de Mapa De Estrada combina dados pré-gravados e informações em tempo real do ambiente estático coletadas pelo módulo de percepção (Zheng *et al.*, 2023, p. 15).

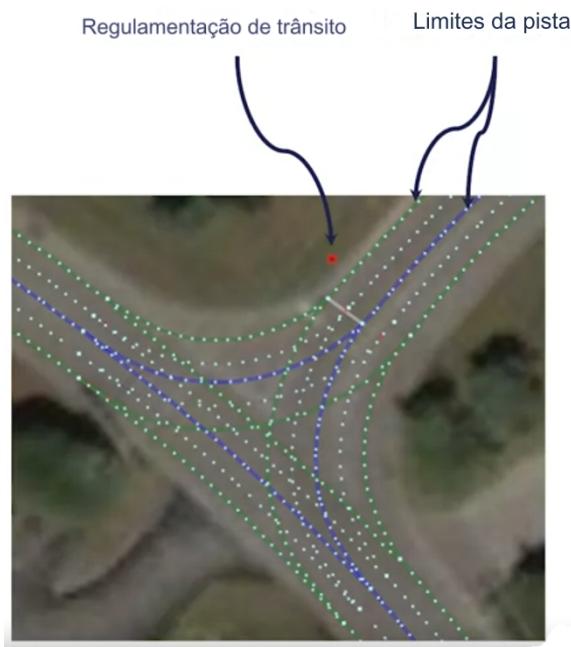


Figura 17 – Arquitetura de Software ([University of Toronto, 2018a](#), Week 2 - Lesson 4: Environment Representation. 6min36s).

- **Percepção:** o módulo de percepção do ambiente desempenha um papel crucial na compreensão do entorno. Ele é composto por dois componentes-chave: sensores

proprioceptivos e exteroceptivos. O proprioceptivo integra múltiplos fluxos de informações, como GPS, IMU e odometria da roda, para determinar com precisão a posição do veículo. Simultaneamente, os sensores exteroceptivos que incluem os módulos de detecção e classificação de objetos dinâmicos e estáticos, descritos na subseção 2.2.1.1, utilizam dados de câmera e LIDAR para identificar e rastrear objetos como carros, bicicletas, pedestres, marcações de estrada e sinais (Zheng *et al.*, 2023, p. 2), (Ignatious; Hesham-El-Sayed; Khan, 2022, p. 737).

- **Planejamento de Movimento:** o planejamento de movimento envolve tomar decisões sobre ações e navegação com base em informações dos módulos de percepção e mapeamento. O processo é decomposto em três camadas: planejamento de missão, planejamento de comportamento e planejamento local (University of Toronto, 2018a, Week 2 - Lesson 3: Software Architecture. 6min24s).

Inicialmente, no Planejamento de Missão se lida com planejamento de longo prazo, determinando a sequência ideal de segmentos de estrada desde a localização atual até o destino (Paden *et al.*, 2016, p. 4). Ao determinar a sequência ideal de segmentos representado como um grafo direcionado com pesos de aresta correspondentes ao custo de percorrer um segmento da estrada, tal rota pode ser formulada como o problema de encontrar um caminho de custo mínimo em um grafo da rede rodoviária. Os gráficos que representam redes rodoviárias podem, no entanto, conter milhões de arestas, tornando impraticáveis os algoritmos clássicos de caminho mais curto, como Dijkstra ou A*. Como apresenta o autor, (Paden *et al.*, 2016, p. 3), esses problemas de planejamento eficiente de rotas em redes de transporte atraiu interesse significativo na comunidade científica de transporte, levando à invenção de uma família de algoritmos que, após uma única etapa de pré-processamento, retornam uma rota ideal em uma rede em escala continental em milissegundos.

Mais adiante, no Planejamento de Comportamento é abordado problemas de planejamento de curto prazo, estabelecendo um conjunto de ações ou manobras seguras a serem executadas ao longo do caminho da missão (Paden *et al.*, 2016, p. 4).

Por fim, no Planejamento Local é realizado o planejamento imediato, definindo um caminho específico e perfil de velocidade para dirigir. O plano local deve ser suave, seguro e eficiente, considerando todas as restrições atuais impostas pelo ambiente e manobra (Paden *et al.*, 2016, p. 3).

- **Controlador:** o módulo de controle executa o caminho planejado gerado pelo módulo de planejamento de movimento. Sendo responsável por separar o problema de controle em controle longitudinal e lateral, regulando aceleração, freios, marchas e ângulo de direção para seguir precisamente o caminho planejado (University of Toronto, 2018a, Week 2 - Lesson 3: Software Architecture. 10min54s).

- **Supervisor de Sistema:** o supervisor do sistema monitora continuamente tanto os componentes de hardware quanto de software para garantir o bom funcionamento do VA. Sendo composto por duas partes: o supervisor de hardware, responsável por verificar falhas de hardware, e o supervisor de software, que valida toda a pilha de software e analisa inconsistências entre as saídas dos módulos ([University of Toronto, 2018a](#), Week 2 - Lesson 3: Software Architecture. 11min37s).
- **Atuação:** por fim, atuação refere-se à transformação de toda a arquitetura em comandos precisos de atuação para o veículo.

2.3 Simulação de Carros Autônomos

Esta seção aborda o aspecto prático do desenvolvimento de carros autônomos por meio da utilização de simulações, fundamentais no processo iterativo e de mitigação de riscos na criação de VA. Ademais, reunimos todos os conceitos discutidos ao longo das seções e subseções anteriores, e aplicamos em um ambiente de simulação. Onde obtivemos um ambiente controlado e dinâmico no qual conceitos, algoritmos e sistemas podem ser rigorosamente testados sem a necessidade de implementação no mundo real. Podemos, então, executar cenários na simulação envolvendo vários veículos e pedestres controlados por inteligência artificial (IA). Onde podemos realizar variações nesses cenários, centenas ou até milhares de vezes, para garantir que nosso veículo tome consistentemente a decisão correta. Este método garante uma avaliação minuciosa das capacidades do veículo, identificando desafios potenciais e aprimorando soluções antes de sua implementação no domínio físico ([Dosovitskiy et al., 2017](#), p. 1). Desse modo, essa seção visa cumprir o objetivo proposto no capítulo 1.3 deste trabalho.

2.3.1 Ambientes de simulação de carro autônomo

Existe uma ampla variedade de simuladores disponíveis, desenvolvidos por equipes tanto da indústria quanto da academia. Segundo o autor ([Dosovitskiy et al., 2017](#), p. 1), esses simuladores são usados desde os primeiros dias de pesquisa em direção autônoma. Onde configurações personalizadas de simulações são utilizadas para treinar e avaliar os sistemas de percepção do veículo. Um exemplo notável em relação ao uso de simuladores, é o uso *ad-hoc*⁴ de jogos digitais para extrair dados de alta fidelidade destinados ao treinamento e avaliação de sistemas de percepção visual. Embora haja um uso *ad-hoc* significativo de simuladores na pesquisa sobre direção autônoma, alguns desses ambientes ou plataformas de simulação existentes apresentam limitações. Por exemplo, segundo o

⁴ O termo “ad hoc” tem sua origem na expressão latina que se traduz literalmente como “para isso” ou “para esta finalidade”. Frequentemente utilizado para descrever uma solução elaborada de forma específica para resolver um problema ou atender a um objetivo preciso, sendo, portanto, não passível de generalização ou aplicação em outros contextos ([adhoc..., 2024](#)).

autor ([Dosovitskiy et al., 2017](#), p. 1), simuladores de corrida de código aberto, como TORCS, não apresentam complexidade da condução urbana: os simuladores de jogos de corrida existentes apresentam limitações significativas quando se trata de replicar a complexidade da condução urbana. Esses simuladores carecem geralmente de elementos essenciais, como pedestres, cruzamentos, tráfego cruzado, regras de trânsito e outras complicações que distinguem a condução urbana das corridas em pistas. Além disso, jogos digitais que simulam ambientes urbanos com alta fidelidade, como o *Grand Theft Auto V*, não oferecem suporte adequado para a avaliação detalhada de políticas de trânsito. Esses jogos têm pouca personalização e controle sobre o ambiente, uma variação de sensores severamente limitada, falta de *feedback* detalhado ao violar regras de trânsito, entre outras limitações devido à sua natureza comercial de código fechado e objetivos fundamentalmente diferentes do seu desenvolvimento.

Devido a isso, neste trabalho utilizaremos o simulador chamado Carla ([CARLA, 2018](#)). Segundo os autores ([University of Toronto, 2018a](#)) e ([Dosovitskiy et al., 2017](#)), Carla é um simulador de código aberto desenvolvido por uma equipe composta por membros do *Computer Vision Center* da Universidade Autônoma de Barcelona, *Intel* e do *Toyota Research Institute*, utilizando o *Unreal Engine 4*⁵. O simulador em questão apresenta ambientes virtuaismeticulosamente elaborados, desenvolvidos integralmente por uma equipe especializada de artistas digitais contratados para esse propósito. Esses ambientes abrangem desde cenários urbanos até uma variedade de modelos de veículos, edifícios, pedestres, placas de rua, entre outros elementos. Diferenciando-se de outros simuladores discutidos, o simulador Carla possibilita a configuração flexível de conjuntos de sensores e fornece sinais que podem ser utilizados para o treinamento de estratégias de direção, tais como coordenadas GPS, velocidade, aceleração e dados detalhados acerca de colisões e outras infrações.

Além disso, destaca-se a capacidade deste simulador em permitir uma ampla gama de condições ambientais, incluindo variações climáticas e diferentes horários do dia. Algumas dessas condições ambientais são exemplificadas na Figura 18, assim como os mapas disponíveis no simulador: *Town1*, *Town2*, *RaceTrack* e *FlatEarth* (“fun-mode”) podem ser vistos, respectivamente.

⁵ Engine e/ou Game Engine um software e/ou um conjunto de bibliotecas projetado para simplificar e abstrair o processo de desenvolvimento de jogos eletrônicos ou outras aplicações que envolvam gráficos em tempo real, destinadas a videogames e/ou computadores ([Lewis; Jacobson, 2002](#)).



Figura 18 – Simulador Carla ([CARLA, 2018](#)). Diferentes condições ambientais e Mapas (Capturas de tela pelos autores).

Toda a simulação pode ser controlada por um cliente externo, que permite enviar comandos ao veículo, registrar dados e executar automaticamente cenários para avaliar o desempenho do carro.

2.4 Trabalhos relacionados

Como iniciado na subseção [2.2.1.2](#), a integração de YOLO com simuladores possibilita o desenvolvimento e a validação de sistemas complexos, como VAs. Trabalhos como o de [Andrade \(2022\)](#) exemplificam essa integração ao utilizar o YOLOv4 para detecção de objetos e estimar a distância no veículo (ego) até outro, e a tese de mestrado ([Juanola, 2019](#)) desenvolve uma solução para detectar placas de velocidade e dar um feedback para o motorista quanto a sua velocidade em relação à placa detectada. Sabendo disso, a solução de deste TCC se diferencia do trabalho [Juanola \(2019\)](#) visto que propomos uma solução de

carro autônomo que percorre um trajeto por conta própria e inclui a detenções de placas de parada e velocidade no simulador CARLA.

Além de enfatizar a relevância do simulador CARLA, o trabalho de [Kim, Jeon e Lim \(2023\)](#) destaca as dificuldades enfrentadas pelos algoritmos de detecção de objetos, como o YOLO, ao lidar com condições adversas, especialmente em ambientes não contemplados no conjunto de dados utilizados para treinamento. Os autores afirmam que tais métodos são altamente dependentes dos dados de treinamento, limitando sua aplicabilidade em novos cenários não representados na base de dados original. Além disso, a coleta de conjuntos de dados apropriados é um desafio significativo, tanto em termos de custos quanto de tempo. Os autores apresentam que modelos treinados com imagens de situações de precipitação de 10mm/h apresentaram baixa acurácia ao serem testados em condições com precipitação acima desse valor, evidenciando o problema de escassez de dados em cenários de chuva intensa. Para mitigar essas limitações, ambientes virtuais como o CARLA têm se consolidado como ferramentas confiáveis para experimentação em condições climáticas adversas, facilitando a coleta de dados para treinamento ([Kim; Jeon; Lim, 2023, p. 2](#)). Isso demonstra a robustez do simulador na criação de cenários adversos para avaliação quantitativa.

[Gao, Tang e Wang \(2021\)](#) também utilizaram o CARLA para conduzir experimentos com YOLOv4, CenterNet e Faster-RCNN, identificando o YOLOv4 como o mais eficiente para detecção de objetos sem otimizações específicas. Este estudo complementa os resultados obtidos por [Kim, Jeon e Lim \(2023\)](#), ao evidenciar o potencial do YOLO em diferentes condições experimentais. Como aborda o estudo de [Ahammed, Hossain e Obermaisser \(2024\)](#) com foco na detecção de obstáculos em zonas de construção, desenvolveram um modelo baseado em YOLO para identificação de obstáculos, contribuindo com uma solução de sistema avançado de assistência ao motorista, atingindo alta precisão de 94% e tempos de inferência rápidos, 1.6ms.

O trabalho de [Surendra \(2023\)](#), apresenta uma abordagem voltada para a detecção de faixas e sinais de trânsito, visando desenvolver sistemas de Assistência Avançada ao Motorista em VAs. Utilizando o simulador CARLA, os autores implementaram e testaram modelos baseados em redes neurais convolucionais (CNN), como o SegNet para segmentação semântica e o YOLO para detecção de objetos. Este estudo é particularmente relevante ao contexto deste trabalho, visto que também utilizamos o simulador CARLA para simulação de detecção em tempo real, com foco em sinais de trânsito e assistência ao condutor.

Especificamente, o trabalho de [Surendra \(2023\)](#) destaca desafios associados à identificação de faixas e sinais de trânsito em cenários urbanos dinâmicos, especialmente sob condições climáticas adversas. Para mitigar essas dificuldades, os autores utilizaram o ambiente virtual para criação do conjunto de dados, de maneira similar ao trabalho de [Kim, Jeon e Lim \(2023\)](#), e validação dos modelos. O modelo de segmentação semântica

SegNet atingiu precisão média (mAP)⁶ de 93,33%, acurácia de 94,80% e uma taxa de erro de 5,20% para detecção de faixas. Já o modelo de detecção de sinais de trânsito baseado no YOLO alcançou uma mAP de 93,67%, acurácia de 95,56% e taxa de erro de 4,44% [Kim, Jeon e Lim \(2023, p. 2067\)](#). Esses resultados reforçam a eficácia do uso de simulações realistas para testar e validar modelos em condições variadas, uma abordagem também adotada neste trabalho em questão.

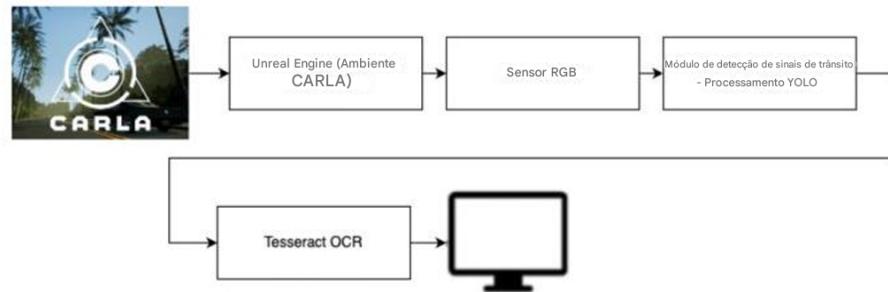


Figura 19 – Diagrama de blocos de detecção de sinais de trânsito ([Surendra, 2023, p. 2064](#)).

Agora no contexto de detecção de sinais de trânsito, o pipeline proposto pelos autores, [Surendra \(2023\)](#), é realizada em duas etapas principais: identificação e reconhecimento. A partir da Figura 19 podemos compreender que, inicialmente, o módulo YOLO analisa as regiões de interesse na cena capturada pela câmera RGB montada no veículo, determinando a localização e o tamanho dos sinais de trânsito. Em seguida, o OCR extrai informações textuais, como limites de velocidade, permitindo uma análise detalhada dos sinais detectados, se diferenciado do trabalho de [Juanola \(2019\)](#) o qual não efetua extração textual.

⁶ mAP é a média do valor médio da precisão em todos os valores de recall. É uma métrica popular para medir a precisão dos algoritmos de detecção de objetos.

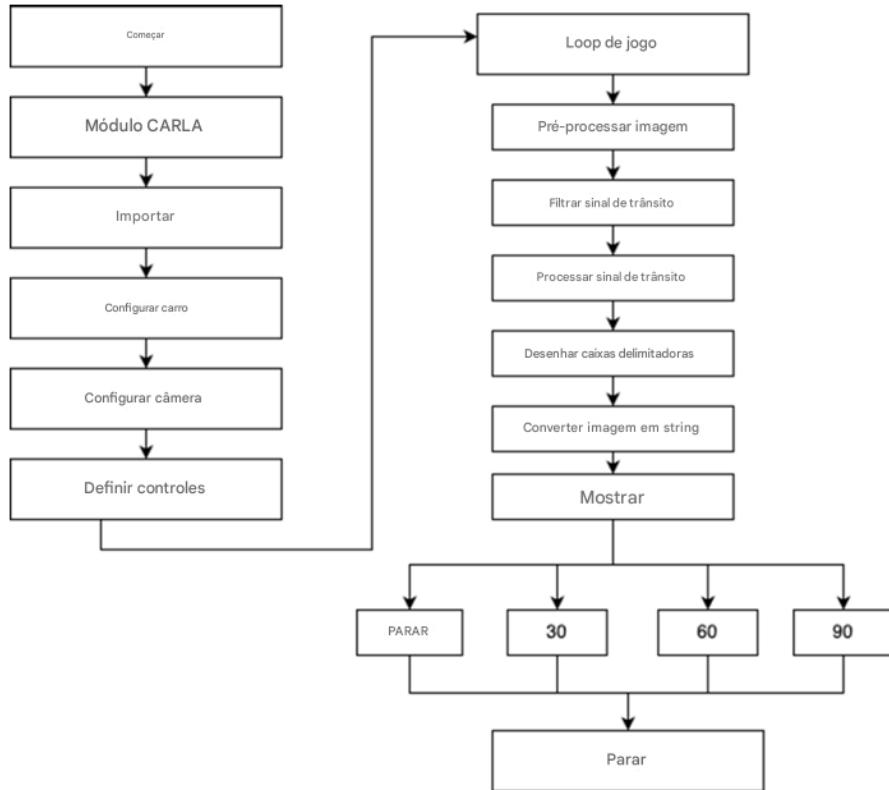


Figura 20 – Fluxograma de detecção de sinais de trânsito ([Surendra, 2023](#), p. 2065).

Os testes foram realizados em conjunto de dados contendo 612 imagens extraídas do CARLA, abrangendo diferentes condições climáticas e cenários urbanos. Conforme podemos identificar no Fluxograma de detecção de sinais de trânsito na Figura 20, as classes de sinais de trânsito incluíram limites de velocidade de 30 km/h, 60 km/h, 90 km/h e sinal de parada.

A relevância deste estudo para o TCC proposto reside na utilização de metodologias similares, como a aplicação do YOLO no simulador CARLA, para a detecção e classificação de sinais de trânsito. Além disso, a análise de desempenho dos modelos em condições variadas, como diferentes climas e densidades de tráfego, proporciona percepções importantes para a validação e aprimoramento de sistemas de percepção e feedback em VAs.

Por fim, diferentemente dos estudos apresentados anteriormente, os trabalhos de [Wu e Cao \(2022\)](#) e [Li, Wang e Wang \(2023b\)](#) propõem modificações na arquitetura YOLO. Onde o trabalho de [Wu e Cao \(2022\)](#) propôs melhorias no YOLOv4 para detecção de sinais de trânsito, utilizando redes mais leves e eficientes como a MobileNetV2 para extração de características. Enquanto, o trabalho de [Li, Wang e Wang \(2023b\)](#) foca na melhoria da detecção de pequenos sinais de trânsito em ambientes complexos, utilizando aprimoramentos no YOLOv7, como camadas específicas para alvos pequenos e métricas de distância otimizadas. Esses avanços evidenciam que o YOLO é um algoritmo em constante

aprimoramento, impulsionado pelos esforços contínuos da comunidade de desenvolvedores e pesquisadores. As modificações propostas nas diferentes versões do YOLO não apenas ampliam sua aplicabilidade em cenários desafiadores, como também demonstram seu potencial para se adaptar às crescentes demandas da percepção em sistemas complexos. Dessa forma, o YOLO consolida-se como uma solução em evolução contínua, capaz de atender às exigências de tarefas críticas, como a detecção de objetos em ambientes dinâmicos e de alta complexidade.

A implementação do YOLO feitas por esses trabalhos é a partir da Darknet, framework Darknet, uma rede neural de código aberto escrita em C e CUDA, que suporta tanto CPU quanto GPU (Redmon, 2013–2016). A escolha pela Darknet vem da sua simplicidade e baixo custo computacional, além da ampla compatibilidade com diferentes ambientes de desenvolvimento (Redmon *et al.*, 2016). Em simulações de VAs, é comum utilizar o YOLOv*-Tiny, uma versão otimizada do YOLOv*, devido à sua menor complexidade, que reduz o tempo de carregamento e processamento, sendo ideal para aplicações em tempo real. Essa versão compacta, no YOLOv3-tiny, contém 16 camadas convolucionais, em comparação com as 75 da versão completa (YOLOv3), proporcionando maior velocidade em troca de uma ligeira redução na precisão (Redmon; Farhadi, 2018).

No contexto do simulador CARLA, o treinamento da rede YOLO para detecção de objetos personalizados, como placas de trânsito, exige a configuração adequada dos parâmetros da rede e o uso de imagens do ambiente simulado, como fizeram os trabalhos apresentados anteriormente (Kim; Jeon; Lim, 2023; Surendra, 2023).

Apesar de suas vantagens, o YOLO apresenta limitações, como a suscetibilidade a ataques direcionados. Estudos como o de Wu *et al.* (2020) demonstraram a viabilidade de criar camuflagens para veículos, de modo que eles não sejam detectados ou sejam erroneamente classificados por redes neurais. Essas vulnerabilidades ressaltam a necessidade de aprimorar os modelos existentes para garantir maior robustez em aplicações críticas, como sistemas autônomos.

De todo modo, a utilização do YOLO, aliado a frameworks como o Darknet e simuladores como o CARLA, permite avanços significativos no desenvolvimento de sistemas de detecção de objetos para carros autônomos. A rápida evolução do algoritmo e sua integração com tecnologias emergentes destacam seu papel fundamental na construção de sistemas mais seguros e eficientes.

3 Modelagem Conceitual para simulação de Carros Autônomos

Neste capítulo, formularemos a modelagem conceitual do artefato que será implementado no Capítulo 4. A fundamentação teórica apresentada no Capítulo 2 oferece a base necessária para a concepção desta arquitetura. Dessa forma, tomamos como referência a Arquitetura da Tarefa de Condução 2.2.2 para elaborar uma solução que atende aos objetivos delineados nos Objetivos 1.3, validando, consequentemente, a Hipótese 1.2 proposta neste trabalho.

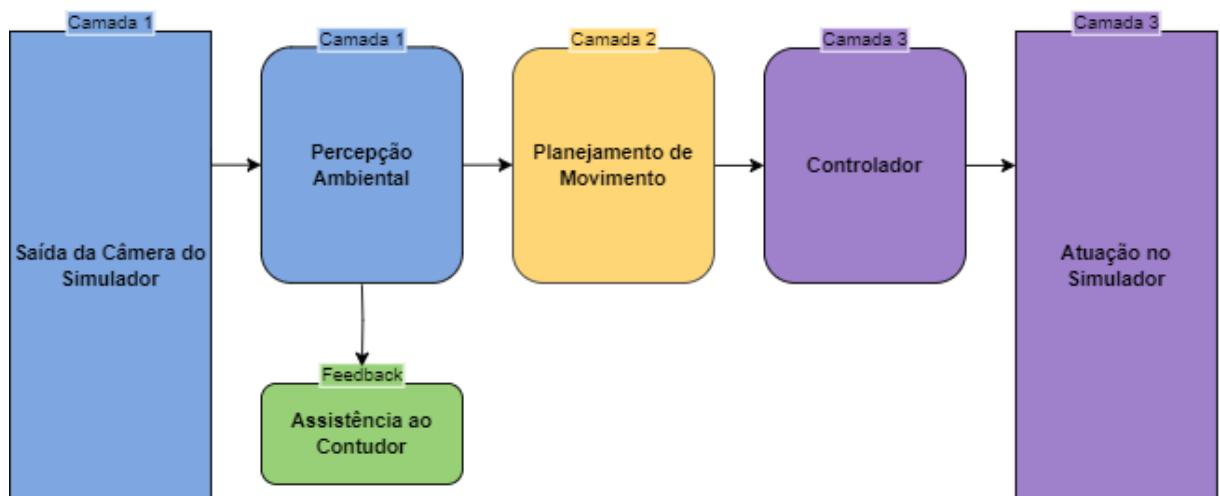


Figura 21 – Arquitetura do Software. Elaborado pelo Autor.

A arquitetura desenvolvida, ilustrada na Figura 21, é composta por múltiplas camadas que refletem os diferentes módulos necessários para a operação de um carro autônomo em um ambiente simulado. O sensor escolhido para a primeira camada é a câmera, devido à sua versatilidade e à riqueza de dados visuais, essenciais para tarefas de percepção ambiental, como elaboraremos na seção 3.3. Essa camada 1 será a responsável por detectar e classificar os objetos e o seu resultado permitirá o feedback ao usuário. Baseado nesses resultados da Camada 1, a Camada 2 (Seção 3.2) ditará os movimentos necessários para a Camada 3 exercer no simulador (Seção 3.1). Como descritas abaixo:

- **Camada 1 - Percepção Ambiental:** A primeira camada processa os dados provenientes da câmera do simulador. O módulo realiza a análise da cena para identificar objetos: placas de velocidade e parada. Este módulo utiliza técnicas de visão computacional, a detecção de objetos baseada no algoritmo YOLO, descrito na seção 3.3.2. Além disso, a partir dos resultados dessa camada podemos assistir o condutor via feedbacks em tempo real, conforme visto na Figura 25, ampliando

a segurança e a confiabilidade na direção. O modelo geral da Camada 1 para a percepção do ambiente pode ser visto na Figura 22.

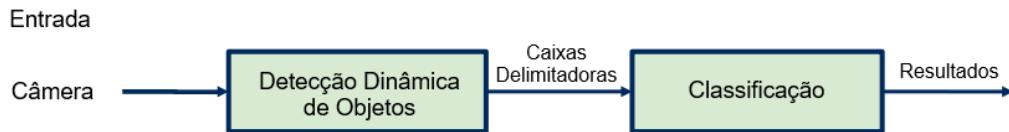


Figura 22 – Arquitetura de Software da Percepção do Ambiente. Baseado em [University of Toronto \(2018a\)](#).

- **Camada 2 - Planejamento de Movimento:** Baseado nas informações fornecidas pela camada de percepção, o módulo de planejamento de movimento gera perfil de velocidade, e calcula trajetórias eficientes para o veículo. Este módulo considera restrições dinâmicas, regras de trânsito e a previsão de movimentos de outros agentes na cena, descrito na seção 3.2. O modelo geral da camada 2 pode ser visto na Figura 23.

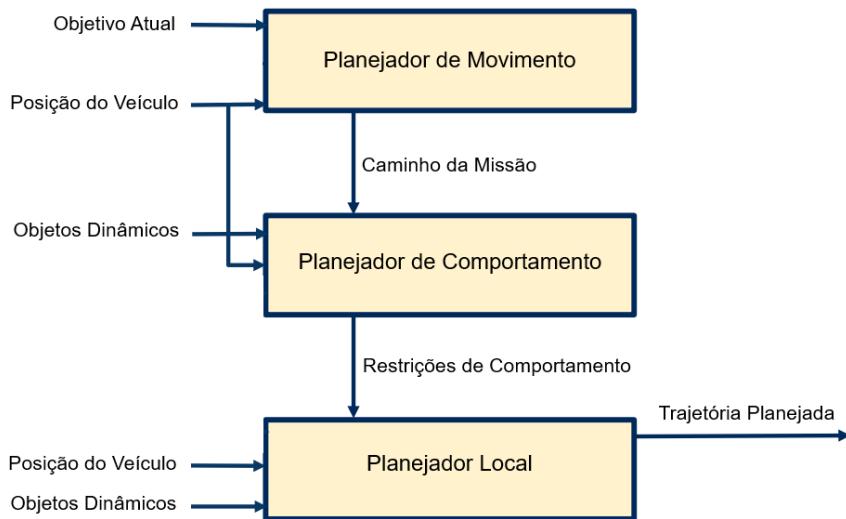


Figura 23 – Arquitetura de Software do Planejador de Movimento. Baseado em [University of Toronto \(2018a\)](#).

- **Camada 3 - Controlador e Atuação:** A terceira camada é responsável por traduzir as trajetórias planejadas em comandos de controle de aceleração e direção, para a nossa solução, que serão aplicados ao simulador. Controladores são utilizados para garantir que o veículo siga a trajetória planejada com precisão, ajustando

continuamente os comandos, como elaborado na seção 3.1. O modelo geral da camada 3 pode ser visto na Figura 24.

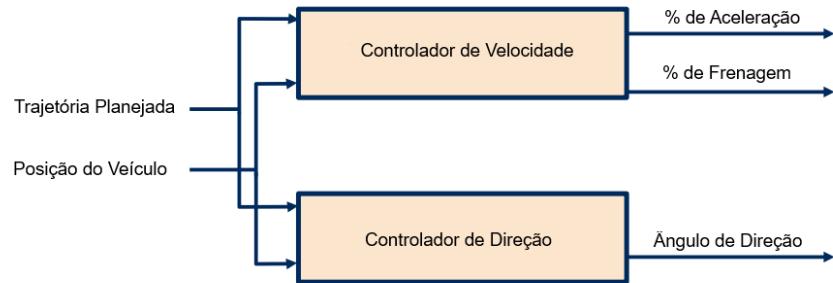


Figura 24 – Arquitetura de Software do Controlador do Veículo. Baseado em University of Toronto (2018a).

Desse modo, o fluxo de dados é iniciado pela captura de informações visuais na saída da câmera do simulador. Em seguida, esses dados percorrem as camadas descritas acima, resultando na atuação final no ambiente simulado. O fluxo também inclui o feedback para assistência ao condutor a partir da camada de percepção, assegurando a atualização contínua das informações percebidas, conforme podemos identificar no fluxograma da Figura 25.

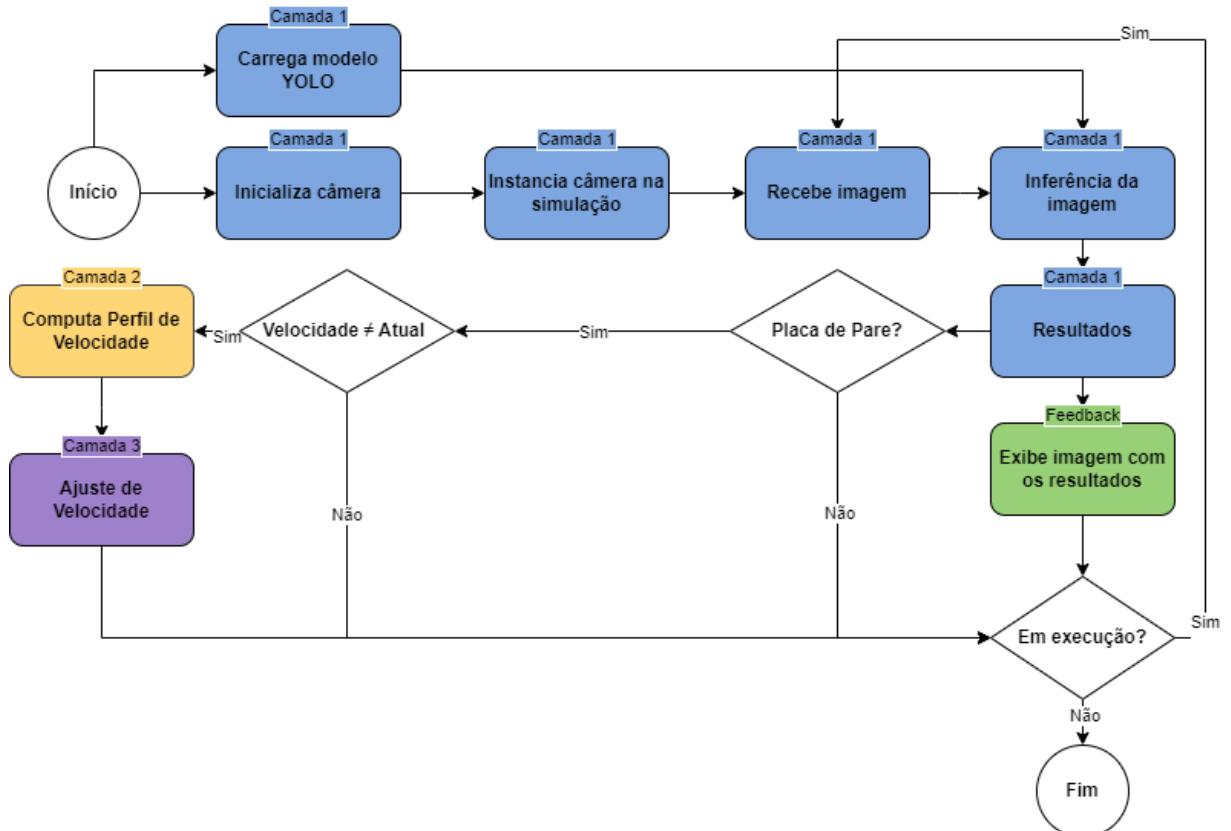


Figura 25 – Fluxo da Execução da Solução. Elaborado pelo Autor.

O fluxograma da Figura 25 apresenta de maneira detalhada o processo completo de percepção, feedback, planejamento e controle, em etapas distribuídas entre as camadas da arquitetura. Como podemos identificar na Figura 25, o processo é iniciado com o carregamento do modelo de detecção YOLO, responsável pela detecção e classificação de objetos na cena. Em seguida, a câmera é inicializada e instanciada na simulação, garantindo a captura contínua de imagens em tempo real. As imagens capturadas são processadas pelo modelo, que realiza inferências para identificar os objetos de interesse. Após a inferência, os resultados são processados e exibidos visualmente, permitindo a validação e interpretação das informações detectadas. Se uma placa de pare for identificada, o sistema retorna os dados para o próximo módulo, possibilitando ajustes dinâmicos. Esse ajuste é permitido pela Camada 2, o perfil de velocidade é computado com base nos dados de detecção e nas condições simuladas. Caso a velocidade atual do veículo não corresponda ao requisito da pare de pare, uma resposta é enviada para a Camada 2 para a mudança do perfil de velocidade e então o fluxo avança para a Camada 3, onde é realizado o ajuste da velocidade, enviando comandos ao simulador para adequação em tempo real.

Esse fluxo segue iterativamente enquanto a solução permanece em execução. Caso o processo seja encerrado, o sistema interrompe a captura de dados e finaliza as operações.

Desse modo, as seções subsequentes 3.1, 3.2 e 3.3 desenvolverão como cada camada será projetada.

3.1 Controle de Carros Autônomos

Agora que adquirimos compreensão sobre os componentes, classificações, terminologias, conceitos e definições dos VAs, iniciaremos nesta seção uma exploração essencial da Modelagem Cinemática e Dinâmica. O propósito ao investigar essas áreas é proporcionar uma compreensão abrangente dos princípios fundamentais que governam o movimento e controle de veículos, conforme estabelecido no Objetivo 1.3.

Ao longo desta seção, serão desenvolvidas duas subseções principais: Modelagem Cinemática e Modelagem Dinâmica. Na subseção de Modelagem Cinemática (Subseção 3.1.1), discutiremos o movimento dos veículos por meio de posições e velocidades, fornecendo percepções sobre coordenadas, transformações e o desenvolvimento de modelos cinemáticos para veículos simples e complexos. Além disso, apresentaremos uma implementação prática desses modelos na subseção 3.1.1.2. Posteriormente, na subseção de Modelagem Dinâmica (Subseção 3.1.2), aprofundaremos as dinâmicas intrincadas do movimento do veículo, considerando as forças e momentos que influenciam carros autônomos.

Ao final desta seção, teremos estabelecido uma base sólida em modelagem cinemática e dinâmica. A compreensão dessas modelagens se torna necessária para entendermos como é concedido o controle de VAs e suas dinâmicas, possibilitando assim a construção dos

fundamentos necessários para compreender as seções subsequentes e atingir os Objetivos 1.3 deste trabalho.

3.1.1 Modelagem Cinemática em 2D

Conforme apontado pelo autor University of Toronto (2018a), o movimento de um veículo pode ser modelado considerando a restrição geométrica que define seu deslocamento ou considerando todas as forças e momentos que atuam sobre ele.

Nesta subseção, abordaremos as diferenças fundamentais entre essas técnicas de modelagem. O primeiro caso é conhecido como Modelagem Cinemática, que se concentra principalmente em descrever o movimento do veículo em termos de posições e velocidades, negligenciando os detalhes intrincados das forças e momentos que atuam sobre ele. Essa abordagem mostra-se particularmente útil em baixas velocidades, onde as acelerações são desprezíveis. Na modelagem cinemática, as restrições geométricas que definem o movimento do veículo têm importância crucial. Isso proporciona um meio simplificado, porém, eficaz, de compreender o movimento do veículo sem adentrar nas complexidades das forças e dinâmicas envolvidas (Paden *et al.*, 2016, p. 5). Exemplos práticos incluem robôs de duas rodas e o modelo de bicicleta, ilustrados nas Figuras 26 e 27.



Figura 26 – Robô com rodas e uma câmera omnidirecional (Francis *et al.*, 2016, p. 9).

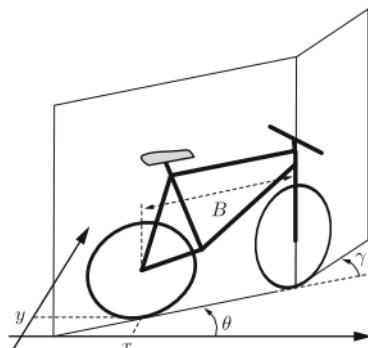


Figura 27 – Modelo de bicicleta (Francis *et al.*, 2016, p. 13).

A modelagem dinâmica, por outro lado, requer uma compreensão mais ampla, ao considerar as forças e eventos que influenciam o movimento de um veículo em uma estrada. Ao considerar esses fatores, os modelos dinâmicos oferecem uma representação mais precisa do comportamento do veículo em uma ampla gama de condições operacionais. Entretanto, o desenvolvimento de modelos dinâmicos demanda uma maior complexidade e intensidade computacional, uma vez que visa representar fielmente a realidade, em contraste com os modelos cinemáticos (Jacobson *et al.*, 2020, p. 9).

A distinção entre essas abordagens destaca o equilíbrio entre simplicidade e precisão na modelagem do movimento de veículos. Enquanto a modelagem cinemática é suficiente

para capturar o movimento em baixas velocidades, a modelagem dinâmica torna-se indispensável à medida que acelerações e forças externas se tornam significativas. Compreender essa dicotomia é crucial para selecionar a técnica de modelagem apropriada com base nos requisitos específicos da aplicação, seja para o desenvolvimento de sistemas de controle, simulação ou estimativa de estado para VAs, os quais serão abordados na subseção 3.1.1.2 seguinte.

3.1.1.1 Sistemas de Coordenadas, Transformações

Os sistemas de coordenadas e as transformações desempenham um papel fundamental na compreensão da dinâmica espacial de VAs. Essa subseção inicia elucidando a importância dos sistemas de coordenadas como sistemas de referência, destacando seu papel na descrição precisa da posição e orientação de objetos. A seguir trataremos de três tipos principais de sistemas de coordenadas: global (inercial), corpo e sensor, vistos na Figura 28. O sistema global, ou inercial, serve como uma referência fixa ligada à Terra, geralmente representada em coordenadas East North Up (ENU), traduzido para o português, refere-se a Leste Norte para Cima ou coordenadas Earth-Centered Earth Fixed (ECEF), traduzido para o português, refere-se a Terra-Centrada Terra-Fixa. Em contraste, o sistema de corpo está fixado à estrutura do veículo, como seu centro de gravidade ou eixo traseiro, e sofre movimento e rotação em relação ao sistema inercial à medida que o veículo se move (Jacobson *et al.*, 2020, p. 61). O sistema de sensor, por outro lado, está associado a cada sensor, visto na Figura 11, a bordo do veículo, definindo as coordenadas para as saídas do sensor.

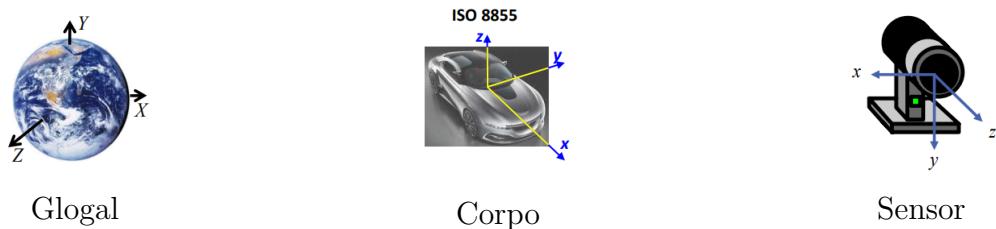


Figura 28 – Sistemas de coordenadas (Jacobson *et al.*, 2016, p. 33).

A partir do conteúdo já apresentado, podemos inferir que em diversas aplicações da robótica, é necessário associar diversas coordenadas a um sistema em movimento e também representar elementos desses sistemas no referencial inercial. Desse modo, para realizar essa tarefa, é imprescindível efetuar a transformação de variáveis de um referencial de coordenadas para outro, como, por exemplo, do referencial do corpo para o referencial inercial. Por exemplo, mesmo em um robô de duas rodas com um único sensor apresenta três desses referenciais a serem considerados, como visto na Figura 28, enquanto um VA pode envolver dezenas deles (University of Toronto, 2018a, Week 4 - Lesson 1: Kinematic Modeling in 2D. 3min44s).

Para manter uma representação consistente dos dados do sensor (vistos na subseção 2.2.1.3) para percepção (visto na subseção 2.2.1.1), precisamos conseguir transformar informações entre quadros de coordenadas.

3.1.1.2 Modelo Cinemático da Bicicleta

Nesta subseção, abordaremos os princípios da modelagem cinemática e das suas restrições, apresentando conceitos fundamentais necessários para compreender o movimento de veículos. O objetivo é esclarecer o conceito de ICR ou *Instantaneous Center of Rotation* em inglês, fornecendo as bases para o desenvolvimento do modelo cinemático de bicicleta, como mostra o autor (Francis *et al.*, 2016, p. 13). O modelo cinemático mais simples de uma bicicleta; pode ser visto na Figura 27, no qual o quadro da bicicleta é perpendicular ao solo e o eixo de direção passa pelo centro da roda dianteira. Denominados por (x, y) as coordenadas do ponto de contato da roda traseira com o solo. Definidos por θ como o ângulo que o quadro faz com o eixo x , e y como o ângulo de direção, conforme ilustrado na Figura 27. Apesar de não ser uma réplica fiel de uma bicicleta, esse modelo é bastante útil, ao captar as características fundamentais de um veículo com quatro rodas, sendo apenas as duas dianteiras direcionáveis.

Desse modo, iniciaremos a modelagem de uma bicicleta que nos auxiliará no entendimento das dinâmicas envolvidas em um VA. O exemplo que trabalharemos é proposto e fundado pelos seguintes autores University of Toronto (2018a), Rajamani (2011), Francis *et al.* (2016). Antes de darmos início a modelagem, definiremos algumas variáveis adicionais além daquelas que usamos para o robô de duas rodas apresentado na subseção 3.1.1.1. Segundo o autor, University of Toronto (2018a) o modelo de bicicleta a ser desenvolvido é chamado de modelo de direção da roda dianteira, pois a orientação da roda dianteira pode ser controlada em relação à direção do veículo.

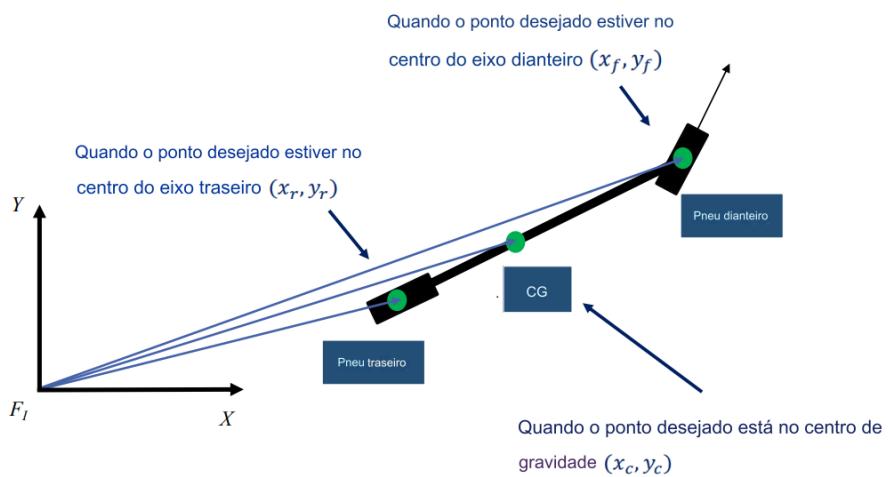


Figura 29 – Modelo de bicicleta 2D (modelo de carro simplificado) (University of Toronto, 2018a, Week 4 - Lesson 2: The Kinematic Bicycle Model. 1min13s).

Para isso, mais uma vez, assumimos que o veículo opera em um plano 2D denotado pelo referencial inercial F_I , como visto na Figura 29. A fim de esclarecimento, no modelo de bicicleta proposto, a roda dianteira representa as rodas dianteiras direita e esquerda do VA, e a roda traseira representa as rodas traseiras direita e esquerda do VA. O autor [University of Toronto \(2018a\)](#) apresenta que para analisar a cinemática do modelo de bicicleta, devemos selecionar um ponto de referência X, Y no veículo que pode ser colocado no centro do eixo traseiro, no centro do eixo dianteiro, ou no centro de gravidade, ou CG na Figura 29. É importante notar que a seleção do ponto de referência interfere nas equações cinemáticas resultantes, que, por sua vez, interferem nos desenvolvimentos do controlador que utilizaremos. Sabendo nisso, iniciaremos com o modelo do ponto de referência no eixo traseiro.

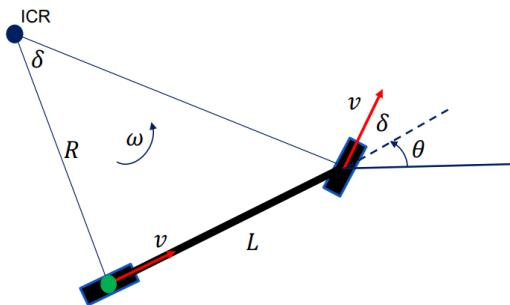


Figura 30 – Ponto de referência da roda traseira no modelo de bicicleta 2D ([University of Toronto, 2018a](#), Week 4 - Lesson 2: The Kinematic Bicycle Model. 3min00s).

Denotaremos a localização do ponto de referência do eixo traseiro como x_r, y_r , e o rumo da bicicleta como θ . Assim sendo, utilizaremos a variável L para representar o comprimento da bicicleta, medida entre os eixos das duas rodas, conforme ilustrado na Figura 30. Destarte, como no modelo do robô de duas rodas, essas são nossas principais variáveis de estado do modelo. Os parâmetros de entrada para o modelo de bicicleta são ligeiramente diferentes daqueles para o robô de duas rodas, uma vez que agora precisamos definir um ângulo de direção para a roda dianteira. Denominamos esse ângulo de direção como δ , e ele é medido em relação à direção frontal da bicicleta. A velocidade é representada por v e aponta na mesma direção que cada roda. O autor [University of Toronto \(2018a\)](#) apresenta que isso é assumido conforme a condição de não escorregamento, que requer que nossa roda não possa se movimentar lateralmente ou escorregar longitudinalmente. Essa é a mesma suposição que nos permite calcular a velocidade frontal do robô de duas rodas com base nas taxas de rotação de suas rodas.

Portanto, devido a essa condição de não deslizamento, temos mais uma vez que ω , a taxa de rotação da bicicleta, é igual à velocidade (v) sobre o ICR, raio R , portanto temos a equação 3.1:

$$\dot{\theta} = \omega = \frac{v}{R} \quad (3.1)$$

O autor University of Toronto (2018a) apresenta que a partir dos triângulos semelhantes formados por L e R , e v e δ , podemos identificar que o tan de δ é igual à distância entre eixos L sobre o raio de giro instantâneo R . Assim, podemos chegar à equação 3.2:

$$\tan \delta = \frac{L}{R} \quad (3.2)$$

Combinando as equações 3.1 e 3.2, podemos encontrar a relação entre a taxa de rotação de ω do veículo e o δ do ângulo de direção, já que ω é igual a v ($\tan \delta$) sobre L . Dessa forma, chegamos à equação 3.3:

$$\dot{\theta} = \omega = \frac{v}{R} = \frac{v \tan \delta}{L} \quad (3.3)$$

Podemos agora formular o modelo cinemático completo da bicicleta para o ponto de referência do eixo traseiro, como visto na Figura 31:

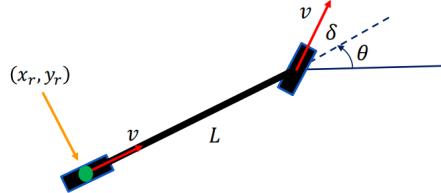


Figura 31 – Ponto de referência na roda traseira no modelo de bicicleta 2D (University of Toronto, 2018a, Week 4 - Lesson 2: The Kinematic Bicycle Model. 3min46s).

Nesta configuração do modelo, os componentes da velocidade do ponto de referência na direção x e y são iguais à velocidade direta v vezes $\cos(\theta)$ e $\sin(\theta)$, temos respectivamente as equações 3.4 e 3.5:

$$\dot{x}_r = v \cos \theta \quad (3.4)$$

$$\dot{y}_r = v \sin \theta \quad (3.5)$$

As equações 3.4 e 3.5 são combinadas com a equação da taxa de rotação derivada anteriormente para formar o modelo da bicicleta com eixo traseiro. A partir dessa combinação chegamos a equação 3.6:

$$\dot{\theta} = \frac{v \tan \delta}{L} \quad (3.6)$$

Como apresentado no início desta subseção, o modelo cinemático da bicicleta pode ser reformulado para outros pontos de referência, como, por exemplo: quando o centro do eixo dianteiro é tomado como ponto de referência x, y .

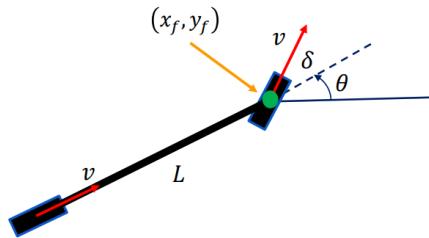


Figura 32 – Ponto de referência no eixo dianteiro ([University of Toronto, 2018a](#), Week 4 - Lesson 2: The Kinematic Bicycle Model. 4min35s).

Desse modo, podemos chegar em modelos como mostrado na Figura 32, que a velocidade (v) aponta desta vez na direção da roda dianteira, definida pela soma de δ e θ . O autor ([University of Toronto, 2018a](#)) introduz que a partir do desenvolvimento de derivações podemos chegar ao seguinte modelo cinemático para o veículo com seu eixo dianteiro como referencial 3.7:

$$\begin{aligned}\dot{x}_f &= v \cos(\theta + \delta) \\ \dot{y}_f &= v \sin(\theta + \delta) \\ \dot{\theta} &= \frac{v \sin \delta}{L}\end{aligned}\quad (3.7)$$

Finalmente, o último cenário para o nosso modelo da bicicleta é quando o ponto desejado é posicionado no centro de gravidade ou centro de massa, conforme mostrado na Figura 33 como cg .

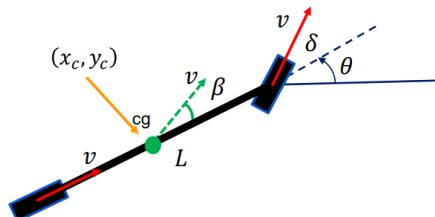


Figura 33 – Ponto de referência no centro de massa ([University of Toronto, 2018a](#), Week 4 - Lesson 2: The Kinematic Bicycle Model. 5min45s).

O autor ([University of Toronto, 2018a](#)) salienta que devido às restrições de não deslizamento que impomos nos pneus dianteiros e traseiros, a direção do movimento no

centro de gravidade é ligeiramente diferente da direção da velocidade para frente em cada roda e da orientação da bicicleta. Desse modo, chamaremos essa diferença de ângulo de escorregamento ou ângulo de escorregamento lateral, que denotaremos de β , sendo medido como a diferença angular entre a velocidade no centro de gravidade e a orientação da bicicleta. Essa definição de ângulo de escorregamento lateral também se aplicará quando passarmos para a modelagem dinâmica de veículos, onde pode se tornar mais pronunciada. O modelo cinemático com o ponto de referência no centro de gravidade pode ser derivado de maneira semelhante aos modelos com o ponto de referência nos eixos traseiro e dianteiro, como relevado pelo autor ([University of Toronto, 2018a](#)). Portanto, podemos chegar à seguinte formulação vista na equação 3.8, que, também, utilizaremos como base para a modelagem da dinâmica de veículos.

$$\begin{aligned}\dot{x}_c &= v \cos(\theta + \beta) \\ \dot{y}_c &= v \sin(\theta + \beta) \\ \dot{\theta} &= \frac{v \cos \beta \tan \delta}{L}\end{aligned}\tag{3.8}$$

Por último, devido à condição de não escorregamento, podemos calcular o ângulo de escorregamento a partir da geometria do nosso modelo de bicicleta. Dado LR , a distância da roda traseira até cg , o ângulo de escorregamento é dado pela equação 3.9:

$$\beta = \tan^{-1} \left(\frac{l_r \tan \delta}{L} \right)\tag{3.9}$$

Por fim, o autor ([University of Toronto, 2018a](#)) ressalta que não é comum ser possível alterar instantaneamente o ângulo de direção de um veículo de um extremo ao outro de sua faixa, como é atualmente possível com nosso modelo cinemático. Visto uma vez que δ é uma entrada que seria selecionada por um controlador, não há restrição quanto à rapidez com que pode ser alterada, o que é um tanto irreal. Em vez disso, nossos modelos cinemáticos podem ser formulados com quatro estados: $[x, y, \theta, \delta]^T$. Se assumirmos que só podemos controlar a taxa de mudança do ângulo de direção φ , podemos simplesmente estender nosso modelo para incluir δ como um estado e usar a taxa de direção φ como nossa entrada modificada, dessa forma chegamos nas entradas: $[v, \varphi]^T$.

Dessa forma, nosso modelo cinemático de bicicleta está completo, utilizando como entradas a velocidade (v) e a taxa de mudança do ângulo de direção φ . O estado do sistema, que inclui as posições XC, YC , a orientação θ e o ângulo de direção δ , evolui conforme as equações cinemáticas do modelo, que satisfazem a condição de ausência de deslizamento.

$$\begin{aligned}\dot{x}_c &= v \cos(\theta + \beta) \\ \dot{y}_c &= v \sin(\theta + \beta) \\ \dot{\theta} &= \frac{v \cos \beta \tan \delta}{L} \\ \dot{\delta} &= \varphi\end{aligned}\tag{3.10}$$

Agora podemos utilizar o modelo 3.10 para implementar controladores cinemáticos de direção.

3.1.2 Modelagem Dinâmica em 2D

A modelagem dinâmica desempenha um papel crucial na compreensão do comportamento de veículos, especialmente no desenvolvimento de VAs. Abrangendo a complexa interação de forças e eventos que afetam o movimento do veículo, permitindo uma compreensão mais aprofundada, superando as limitações dos modelos cinemáticos, apresentados nas subseções 3.1.1.2 e 3.1.1 anteriores, especialmente em cenários que envolvem altas velocidades, estradas escorregadias. Além disso, a modelagem dinâmica auxilia na compreensão dos comportamentos de subsistemas, onde as restrições cinemáticas podem ser insuficientes. Por exemplo, ao considerar as forças que causam o deslizamento dos pneus e considerar as restrições dinâmicas, aprimorando assim a precisão preditiva (*Jacobson et al., 2020*, p. 89).

Nesta subseção, será explorada a importância da modelagem dinâmica em sistemas de veículos, juntamente com seus componentes-chave e aplicações, desde a configuração de quadros de coordenadas até a aplicação da segunda lei de Newton. Além disso, serão discutidas as implicações da modelagem dinâmica na estimativa de estado, no desenvolvimento de controladores e no aprimoramento da segurança em sistemas de VAs.

Como introduzido, no contexto da modelagem veicular, quando o veículo está em movimento e realizando curvas em velocidades elevadas, ou quando a pista está escorregadia, as premissas estabelecidas pela condição de não deslizamento podem não mais ser válidas. Isso ocorre à medida que as forças aplicadas ao veículo fazem com que os pneus deslizem sobre a estrada. Devido a isso, modelar o equilíbrio de forças durante condições de deslizamento pode ampliar o conjunto de condições de direção para as quais nosso modelo permanece em uma previsão precisa do movimento. Além disso, muitos subsistemas do veículo apresentam condições semelhantes, nos quais uma restrição cinemática rígida não captura adequadamente a evolução desse subsistema, sendo necessários modelos dinâmicos nesses casos (*Jacobson et al., 2020*, p. 89). Um exemplo seria o sistema de transmissão, para o qual o equilíbrio de torques é necessário para capturar a relação entre a posição do acelerador e o torque nas rodas, por meio dos sistemas do motor e da transmissão ([University of Toronto, 2018a](#), Week 4 - Lesson 3: Dynamic Modeling in 2D. 1min55s). Ao

modelar explicitamente o equilíbrio de forças e momentos atuando no veículo, podemos determinar as acelerações que o veículo está experimentando e utilizá-las para modelar o seu movimento.

Portanto, para construir um modelo dinâmico típico para um veículo, segundo o autor ([University of Toronto, 2018a](#)), podemos seguir os seguintes passos vistos na Figura 34, e descritos a seguir 3.1.2:

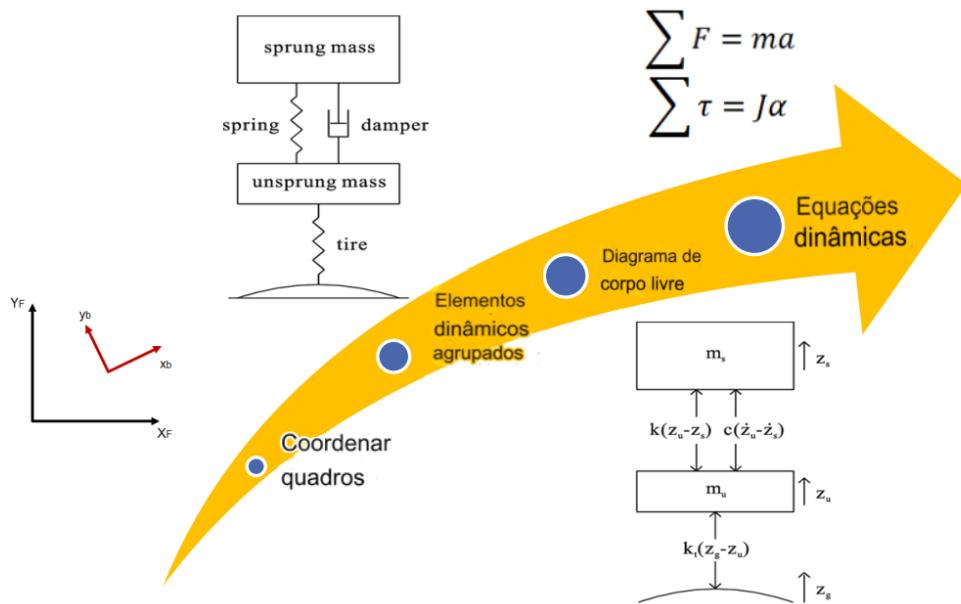


Figura 34 – Modelagem Dinâmica ([University of Toronto, 2018a](#), Week 4 - Lesson 3: Dynamic Modeling in 2D. 2min18s).

1. Inicialmente, é necessário estabelecer os referenciais de coordenadas a serem utilizados no modelo, como o referencial do corpo e o referencial inercial discutidos na subseção 3.1.1.2.
2. Em seguida, é essencial desmembrar o sistema dinâmico em elementos dinâmicos concentrados. Esses elementos combinam aspectos potencialmente distribuídos do sistema em elementos discretos ou concentrados. No caso de um sistema massa-mola-amortecedor, os elementos concentrados seriam a mola, a massa e o amortecedor. Além disso, é necessário definir um modelo para cada elemento concentrado. Por exemplo, a mola linear fornece uma força proporcional ao seu deslocamento a partir da posição de repouso.
3. Posteriormente, é útil esboçar o diagrama de corpo livre para cada corpo rígido na lista de elementos, nomeando e modelando adequadamente todas as forças e momentos atuantes sobre o corpo.

4. Finalmente, é indispensável utilizar a segunda lei de Newton, elaborando as equações matemáticas que definem nosso modelo dinâmico, somando todas as forças ao longo de cada eixo para a dinâmica translacional e todos os momentos ao redor de cada eixo de rotação para a dinâmica rotacional. O resultado é uma equação diferencial ordinária que descreve o movimento de corpo rígido, constituindo assim o modelo dinâmico.

Usando os passos descritos anteriormente podemos construir um modelo dinâmico para um carrinho de correr, visto na Figura 35, cuja posição nos interessa.

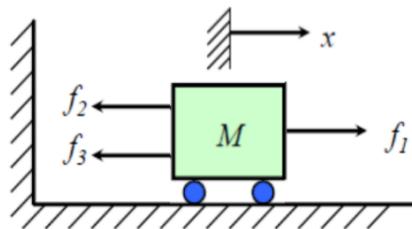


Figura 35 – Modelagem Dinâmica ([University of Toronto, 2018a](#), Week 4 - Lesson 3: Dynamic Modeling in 2D. 4min20s).

Daremos partida, definindo o sistema de coordenadas, apresentado na subseção 3.1.1.1, para a posição do carrinho, indicado por x , como feito no passo a passo da modelagem dinâmica na Figura 34. Em seguida, identificamos o corpo rígido, que, neste caso, é o carrinho de massa m . Posteriormente, elaboramos um diagrama de corpo livre e definimos todas as forças atuantes no carrinho. Neste caso específico, observamos três forças atuando no carrinho: f_1 , que puxa o carrinho para a direita, e f_2 e f_3 , que puxam o carrinho para a esquerda. Finalmente, a lei de Newton é aplicada na direção x para formar o modelo dinâmico. A equação dinâmica resultante que descreve o movimento do carrinho é apresentada a seguir 3.11:

$$\begin{aligned} Ma &= \sum F \\ M\ddot{x} &= f_1 - f_2 - f_3 \end{aligned} \tag{3.11}$$

Esse mesmo passo a passo de modelagem dinâmica podem ser aplicados para outros exemplos, como o de um amortecedor. Segundo o autor [University of Toronto \(2018a\)](#) a modelagem ajuda os engenheiros a projetar e ajustar a suspensão do veículo para melhor conforto e dirigibilidade do veículo. Esse exemplo de modelagem, incluindo o de uma roda, pode ser encontrado a partir dos autores ([University of Toronto, 2018a](#), Week 4 - Lesson 3: Dynamic Modeling in 2D. 4min41s), e muitos outros exemplos a partir do autor ([Jacobson et al., 2020](#)). Como esses exemplos de aplicação do passo a passo de modelagem se tornam repetitivos e mais de um se torna desnecessário para este trabalho, decidimos por não incluí-los.

Como base no conteúdo visto, podemos compreender que os modelos dinâmicos de veículos são de grande utilidade para diversas aplicações. Segundo o autor ([University of Toronto, 2018a](#)), podem, até mesmo, ser empregados para aprimorar os métodos de estimação de estado ao integrar dados de sensores para rastrear o movimento, e auxiliar no projeto de controladores visando seguir uma trajetória ou caminho desejado. Auxiliando engenheiros de VAs a definir os limites do desempenho do veículo, evitando o planejamento de trajetórias inseguras que um VA não consiga performar de maneira adequada. Esses modelos mais robustos consideram todos os componentes e forças que atuam no veículo, sendo conhecidos como modelos 3D completos. Entretanto, esses modelos são extremamente complexos. Devido a isso, iniciaremos nossos entendimentos a partir do modelo 2D, considerando duas partes principais dessa modelagem: modelo dinâmico longitudinal e lateral. Dos quais desenvolveremos nas subseções [3.1.2.1](#) e [3.1.2.2](#) a seguir.

3.1.2.1 Modelagem Longitudinal

Esta subseção apresenta os conceitos fundamentais sobre a dinâmica longitudinal de veículos e dos modelos de componentes indispensáveis, essenciais para gerar torques nos pneus. A discussão gira em torno dos princípios fundamentais necessários para compreender e modelar as complexas interações que influenciam o movimento de um veículo ao longo do seu eixo longitudinal.

Iniciaremos com uma elucidação sobre o equilíbrio das forças dinâmicas que atuam em um veículo na direção longitudinal. Isso inclui uma análise de forças como arrasto aerodinâmico, resistência ao rolamento, força gravitacional e forças nos pneus. Posteriormente, delinearemos modelos típicos de componentes do trem de força necessários para compreender o movimento longitudinal. Destacando a importância desses modelos no desenvolvimento de controladores longitudinais essenciais para o controle do veículo e otimização de desempenho.

Assim, começaremos a compreender a dinâmica longitudinal do veículo, a qual é baseada apenas na dinâmica do veículo que gera movimento para frente. Conforme apresenta o autor [University of Toronto \(2018a\)](#), a partir da Figura [36](#) podemos compreender um movimento longitudinal típico de um veículo em uma estrada inclinada.

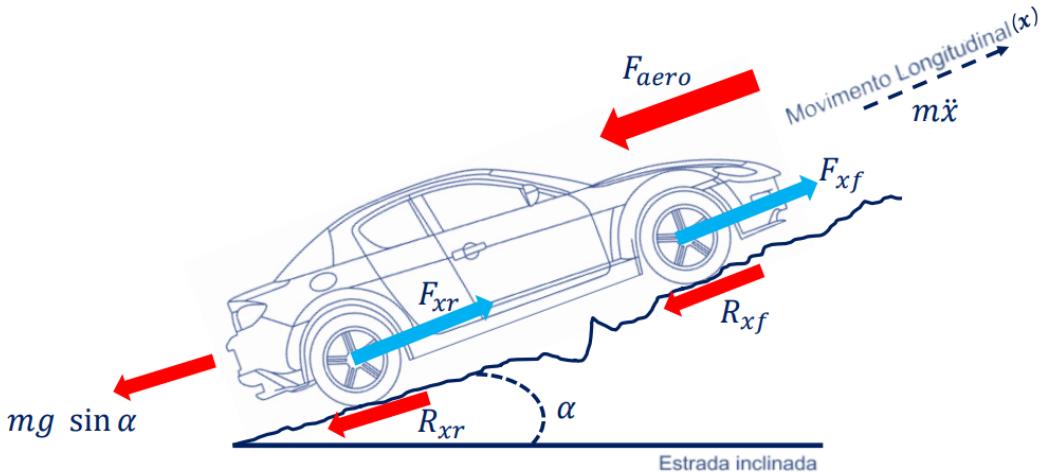


Figura 36 – Modelo Longitudinal de Veículo ([University of Toronto, 2018a](#), Week 4 - Lesson 4: Longitudinal Vehicle Modeling. 1min43s).

Como podemos identificar na Figura 36, as forças que atuam no veículo incluem as forças nos pneus dianteiros e traseiros, representadas por F_{xf} e F_{xr} , respectivamente. Adicionalmente, consideram-se a força de arrasto aerodinâmico F_{aero} e as resistências ao rolamento R_{xf} e R_{xr} . Há ainda a força decorrente da gravidade, representada por $mg \sin \alpha$, onde α é a inclinação local da estrada. O autor ([University of Toronto, 2018a](#)) apresenta que com base na segunda lei de Newton, as forças longitudinais nos pneus dianteiros e traseiros, F_{xf} e F_{xr} , provenientes do conjunto propulsor do veículo, devem superar as forças de resistência, tais como a força aerodinâmica F_{aero} , a força gravitacional $mg \sin \alpha$ e a resistência ao rolamento dos pneus dianteiros e traseiros, R_{xf} e R_{xr} . O desequilíbrio entre essas forças define a aceleração do veículo na direção longitudinal, representada por \ddot{x} . Dessa forma, chegamos na equação 3.12:

$$m\ddot{x} = F_{xf} + F_{xr} - F_{aero} - R_{xf} - R_{xr} - mg \sin \alpha \quad (3.12)$$

- Onde: $m\ddot{x}$ = Aceleração do veículo;
 F_{xf} = Força do pneu dianteiro;
 F_{xr} = Força do pneu traseiro;
 F_{aero} = Forças aerodinâmicas atuando contra o movimento do veículo;
 R_{xf} = Resistência ao rolamento devido aos pneus dianteiros;
 R_{xr} = Resistência ao rolamento devido aos pneus traseiros;
 $mg \sin \alpha$ = Força gravitacional devido à inclinação da estrada (componente do peso paralela ao plano inclinado).

As equações 3.12 da dinâmica longitudinal podem ser simplificadas agrupando as forças nas rodas dianteiras e traseiras como: $F_x = F_{xf} + F_{xr}$ e as forças de resistência ao rolamento dianteira e traseira como: $R_x = R_{xf} + R_{xr}$. Além disso, podemos assumir

inclinações moderadas da estrada, permitindo a aplicação da aproximação de um pequeno ângulo. Assim, $\sin \alpha = \alpha$. Dessa forma, nosso modelo dinâmico simplificado para o movimento longitudinal de um carro é apresentado na equação 3.13 a seguir:

$$m\ddot{x} = F_x - F_{aero} - R_x - mg\alpha \quad (3.13)$$

Onde: $m\ddot{x}$ = Termo Inercial;

F_x = Força de Tração;

F_{aero} = Arrasto Aerodinâmico;

R_x = Resistência ao Rolamento;

$mg\alpha$ = Componente da Força Gravitacional.

Na equação 3.13, $m\ddot{x}$ é o termo inercial que define a aceleração longitudinal do veículo. F_x é a força de tração gerada pelo trem de força, e F_{aero} , R_x e $mg\alpha$ compõem as forças totais de resistência atuando no veículo, que chamaremos de F_{load} .

Agora com a equação 3.13 formulada, precisamos desenvolver modelos para cada uma das forças presentes e definir como elas se conectam às entradas de aceleração e freio que nosso sistema autônomo aplicará.

3.1.2.2 Modelagem Lateral

Na subseção 3.1.2.1 anterior, foi discutida a dinâmica longitudinal do veículo. Agora, o foco nesta subseção, é direcionado para a modelagem dinâmica de um carro de quatro rodas, adentrando especificamente nas complexidades da dinâmica lateral. O objetivo é ampliar o modelo cinemático de bicicleta discutido na subseção 3.1.1.2, transitando de uma representação simplificada para um modelo dinâmico mais detalhado e realista.

Para isso, a nossa exploração inicia amenizando certas restrições, como a condição de não deslizamento, e incorporando forças que influenciam o movimento do veículo. O principal objetivo é fornecer uma compreensão robusta de como a dinâmica lateral pode ser modelada com precisão para um carro de quatro rodas, considerando fatores como forças nos pneus e rigidez em curva.

Neste contexto, utilizaremos as contribuições do autor ([University of Toronto, 2018a](#)) como base para a expansão do nosso modelo cinemático de bicicleta, conforme apresentado na subseção 3.1.1.2, para um modelo dinâmico. Essa ampliação ocorrerá ao flexibilizarmos a condição previamente imposta de ausência de deslizamento e ao considerarmos forças que não foram contempladas no modelo cinemático inicial. É importante destacar que, no modelo dinâmico abrangente da bicicleta, serão preservados dois componentes do movimento: o primeiro na direção longitudinal e o segundo na direção lateral. Especificamente para o modelo lateral do veículo, nosso foco estará na modelagem da taxa

de rotação do veículo com base nos momentos que o influenciam durante o deslocamento. Alcançando assim compressão das dinâmicas envolvidas no controle de VAs.

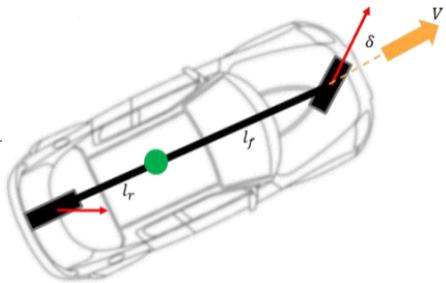


Figura 37 – Modelo de carro para modelo de bicicleta ([University of Toronto, 2018a](#), Week 4 - Lesson 5: Lateral Dynamics of Bicycle Model. 0min48s).

Desse modo, para iniciar a modelagem da dinâmica lateral do modelo de bicicleta, serão adotadas as seguintes premissas: primeiramente, considera-se que a velocidade longitudinal para frente é constante. Segundo o autor ([University of Toronto, 2018a](#)) essa escolha visa desacoplar os modelos dinâmicos lateral e longitudinal, simplificando significativamente a tarefa de modelagem. Em segundo lugar, assim como no modelo cinemático de bicicleta, as rodas esquerda e direita, tanto para os eixos dianteiro quanto traseiro, são agrupadas em uma única roda cada, conforme podemos identificar na Figura 37. Dessa forma, essa premissa converte as quatro rodas para o modelo de bicicleta com duas rodas visto anteriormente. Por fim, assume-se que outros efeitos não lineares, como movimentos de suspensão, inclinação da estrada e forças aerodinâmicas vistos na subseção 3.1.2.1 anterior, são desprezíveis. Na prática, esses efeitos podem ter um impacto significativo nas forças exercidas nos pneus e no VA, e podem ser vistos de maneira aprofundada a partir dos autores ([University of Toronto, 2018a](#)) e ([Jacobson et al., 2020](#)). Portanto, essa é mais uma vez uma premissa limitante em alguns casos, mas é suficiente para os objetivos 1.3 deste trabalho.

3.1.3 Controle Longitudinal de Veículos

No âmbito da condução autônoma, o controle efetivo da dinâmica longitudinal de um veículo é crucial. Esta seção explora os fundamentos do controle longitudinal de veículos, com foco específico no controlador PID (Proporcional-Integral-Derivativo). O objetivo é proporcionar uma compreensão abrangente de como os VAs regulam sua velocidade, um aspecto crítico que influencia o desempenho geral do veículo e a segurança nas estradas.

Será apresentado um guia sobre sistemas de controle lineares clássicos e invariantes no tempo, abrangendo conceitos vitais como funções de transferência e análise no domínio de Laplace. O objetivo é garantir um entendimento sólido dos fundamentos teóricos antes

de adentrar na aplicação prática dos controladores PID, vistos na seção 2.3. Devido a isso, veremos sobre os conceitos centrais dos fundamentos dos controladores. Isso inclui uma exploração de circuitos de realimentação (*feedback* no inglês) e como os controladores interpretam dados sensoriais para gerar sinais de atuação. A representação de sistemas por meio de funções de transferência será discutida em detalhes, esclarecendo as nuances de sistemas lineares e não lineares. A importância de zeros e pólos em funções de transferência será elucidada para fornecer uma compreensão abrangente do comportamento do sistema.

O foco será então direcionado para o controlador PID, um pilar na teoria de controle. Faremos a formulação matemática do controlador PID, compreendendo termos proporcionais, integrais e derivativos. Os papéis dos ganhos proporcional, integral e derivativo (K_p , K_i e K_d) serão explicados, preparando o terreno para uma discussão aprofundada sobre métodos de ajuste e otimização para alcançar os objetivos 1.3.

Para ilustrar a aplicação prática dos controladores PID, será utilizado um modelo de amortecedor de massa-mola de segunda ordem como estudo de caso, elaborado pelo autor (University of Toronto, 2018a). Será realizada uma análise do sistema, incluindo a derivação da função de transferência do sistema. Através da análise da resposta ao degrau, será demonstrada a influência do controle PID no comportamento do sistema, abrangendo parâmetros como tempo de subida, *overshoot* ou “sobressinal ou sobressalto” e tempo de estabilização.

Portanto, utilizaremos o referencial teórico incorporado neste trabalho referente a modelagem de veículos na seção 3.1 para relacionar o desenvolvimento de modelos dinâmico e cinemático para um veículo com base no modelo de bicicleta 3.1.1.2. Esses modelos visam capturar como o sistema dinâmico reage a comandos de entrada do motorista, como direção, aceleração e freio, e como reage a perturbações, como vento, superfície da estrada e diferentes cargas no veículo. Os efeitos dos comandos de entrada e das perturbações sobre os estados, como velocidade e taxa de rotação do veículo, são definidos pelos modelos cinemáticos e dinâmicos como visto na seção 3.1. Sabendo disso, o papel do controlador, então, é regular alguns desses estados do veículo, detectando as variáveis de estado atuais e gerando sinais de atuadores para atender aos comandos fornecidos (University of Toronto, 2018a, Week 5 - Lesson 1: Proportional-Integral-Derivative (PID) Control). Neste caso, no controle longitudinal, o controlador detecta a velocidade do veículo e ajusta os comandos do acelerador e do freio para corresponder à velocidade desejada definida pelo sistema autônomo de planejamento de movimento visto na subseção 2.2 (University of Toronto, 2018a, Week 2 - Lesson 3: Software Architecture. 6min24s).

Dessa forma, segundo o autor University of Toronto (2018a), em um típico ciclo de controle de *feedback*, o modelo de processo recebe os sinais do atuador como entrada e gera as variáveis de saída ou de estado do sistema. Estas saídas são medidas por sensores, e estimadores são utilizados para fundir as medições em estimativas precisas de saída.

As estimativas de saída são então comparadas com as variáveis de saída desejadas ou de referência, e a diferença ou erro é encaminhada para o controlador. O controlador pode ser entendido como um algoritmo matemático que gera sinais para o atuador para minimizar o sinal de erro, aproximando as variáveis de estado do modelo de processo das variáveis de estado desejadas. O autor University of Toronto (2018a), salienta que o modelo de processo, seja ele linear ou não linear, pode ser representado de várias maneiras. Duas das formas mais comuns são a forma de espaço de estados, que acompanha a evolução de um estado interno para conectar a entrada à saída, e a forma de função de transferência, onde o sistema deve ser linear e invariante no tempo, que modela diretamente a relação entre entrada e saída. Segundo o autor University of Toronto (2018a), a função de transferência é uma expressão que relaciona as entradas (U) e as saídas (Y) de um sistema. Essa relação é definida no domínio de Laplace como uma função de S , uma variável complexa utilizada para transitar do domínio do tempo para o domínio de S . Isso permite uma análise mais simplificada da relação entrada-saída, como demonstrado nas equações 3.14 e 3.15 a seguir:

$$Y(s) = G(s)U(s) \quad (3.14)$$

Onde: $Y(s)$ = A saída no domínio de Laplace;

$G(s)$ = A função de transferência;

$U(s)$ = A entrada no domínio de Laplace.

$$s = \sigma + j\omega \quad (3.15)$$

Onde: s = Variável de frequência complexa;

σ = Parte real da frequência complexa;

j = Unidade imaginária;

ω = Frequência angular.

Conforme o autor University of Toronto (2018a) apresenta, no contexto da análise de funções de transferência, as raízes do numerador e do denominador oferecem uma compreensão significativa sobre a resposta de um sistema às funções de entrada. Os zeros de um sistema correspondem às raízes do numerador, enquanto os pólos do sistema são as raízes do denominador, e podemos identificar isso na equação 3.16 a seguir:

$$Y(s) = G(s)U(s) = \frac{N(s)}{D(s)}U(s) \quad (3.16)$$

Onde: $Y(s)$ = Resposta de saída no domínio de Laplace;

$G(s)$ = Função de transferência;

$U(s)$ = Sinal de entrada no domínio de Laplace;

$N(s)$ = Numerador da função de transferência, relacionado aos zeros;

$D(s)$ = Denominador da função de transferência, relacionado aos pólos.

Esses sistemas de resposta para entradas e saídas, segundo o autor ([University of Toronto, 2018a](#)) podem apresentar variações, desde abordagens simples, como multiplicação por ganho constante, tabelas de consulta e equações lineares, até métodos mais detalhados fundamentados em funções não lineares e otimização ao longo de horizontes de previsão finitos. Entre os controladores básicos e clássicos, destacam-se os controladores “*lead-lag*” e o controlador Proporcional Integral Derivativo, conhecidos como PID. O controle PID é expresso matematicamente pela adição de três termos dependentes da função de erro (K_p , K_i e K_d). Um termo proporcional diretamente relacionado ao erro, um termo integral proporcional à integral do erro e um termo derivativo proporcional à derivada do erro, conforme a equação [3.17](#) do controlador PID que é denotado $u(t)$.

$$u(t) = K_p e(t) + K_I \int_0^t e(t) dt + K_d \dot{e}(t) \quad (3.17)$$

Onde: $u(t)$ = A saída do controlador;
 K_p = Ganho proporcional;
 $e(t)$ = Erro, a diferença entre a entrada e a saída desejada;
 K_i = Ganho integral;
 $\int_0^t e(t) dt$ = Integral do erro ao longo do tempo;
 K_d = Ganho derivativo;
 $\dot{e}(t)$ = Taxa de variação do erro.

Segundo o autor [University of Toronto \(2018a\)](#), ao realizar a transformada de Laplace do controle PID [3.17](#), obtém-se a função de transferência $G_c(s)$. A multiplicação por s no domínio de Laplace é equivalente a realizar uma derivada no domínio do tempo, enquanto a divisão por s é equivalente a realizar uma integral. Ao somar esses três termos do controlador PID, obtemos uma única função de transferência para o controle PID, conforme a equação [3.18](#).

$$U(s) = G_c(s)E(s) = \left(K_p + \frac{K_I}{s} + K_D s \right) E(s) = \frac{(K_D s^2 + K_p s + K_I)}{s} E(s) \quad (3.18)$$

Onde: $U(s)$ = A função de transferência do controlador;
 $G_c(s)$ = Ganho do controlador;
 $E(s)$ = Erro no domínio de Laplace;
 K_p = Ganho proporcional;
 K_I = Ganho integral;
 K_D = Ganho derivativo;
 s = Variável complexa no domínio de Laplace.

Ainda segundo o autor [University of Toronto \(2018a\)](#), a função de transferência do controlador PID apresenta um pólo simples na origem proveniente do termo integral. Além disso, ela inclui um numerador de segunda ordem com dois zeros que podem ser posicionados em qualquer lugar do plano complexo ao selecionar valores apropriados para os ganhos. Assim, o projeto de controle PID resume-se na escolha das posições dos zeros para alcançar a saída desejada ou desempenho com base no modelo, conforme podemos observar na equação 3.19.

$$G_c(s) = \frac{K_d s^2 + K_p s + K_i}{s} \quad (3.19)$$

Onde: $G_c(s)$ = Função de transferência do controlador;

$K_d s^2$ = Termo proporcional com dois zeros adicionados;

$K_p s$ = Termo integral;

K_i = Termo derivativo;

s = Variável de Laplace com um polo adicionado.

Os efeitos de cada ação Proporcional (P), Integral (I) e Derivativa (D) estão resumidos na Tabela 1.

Resposta Fechada	Tempo de Subida	Sobres-salto	Tempo de Acomodação	Erro de Estado Estacionário
Aumentar K_p	Diminuir	Aumentar	Peq. mudança	Diminuir
Aumentar K_i	Diminuir	Aumentar	Aumentar	Eliminar
Aumentar K_d	Peq. mudança	Diminuir	Diminuir	Pequena mudança

Tabela 1 – Características dos ganhos P, I e D ([University of Toronto, 2018a](#), Week 5 - Lesson 1: Proportional-Integral-Derivative (PID) Control. 8min11s).

Em última instância, os ganhos PID devem ser selecionados com conhecimento da interação de seus efeitos, a fim de ajustar a resposta do sistema para obter o desempenho adequado. Os conhecimentos relacionados ao PID serão necessários com o avanço deste trabalho para a implantação do controlador longitudinal, de modo a alcançar os objetivos 1.3.

Dessa forma, apresentaremos um exemplo proposto pelo autor [University of Toronto \(2018a\)](#) para podermos compreender como esse controlador funciona, onde primeiro a função de transferência do sistema dinâmico proposto é revisada e então um controlador PID o modelo de amortecedor de massa de mola é projetado, visto na Figura 38.

Após implementação o autor [University of Toronto \(2018a\)](#) apresenta os gráficos resultantes da sua solução, conforme visto nas figuras 39, 40, 41, e 42.

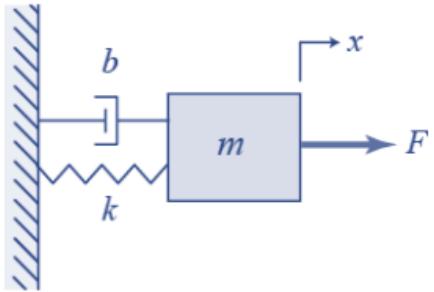


Figura 38 – Sistema Massa-Mola-Amortecedor ([University of Toronto, 2018a](#), Week 5 - Lesson 1: Proportional-Integral-Derivative (PID) Control. 9min00s).

A resposta ao degrau de alguns controladores PID diferentes, disponibilizados pelo autor ([University of Toronto, 2018a](#)). Nos gráficos, a linha horizontal tracejada representa a referência ou saída desejada, e o objetivo dos controladores é manter a saída real próxima a essa referência, conforme visto na Figura 39.

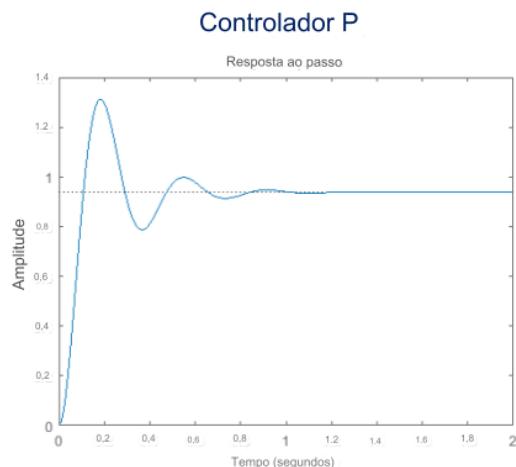


Figura 39 – Controlador P ([University of Toronto, 2018a](#), Week 5 - Lesson 1: Proportional-Integral-Derivative (PID) Control. 12min13s).

No primeiro exemplo ($G_P(s) = K_P$), visto no gráfico da Figura 39, são apresentadas as respostas ao degrau para o controle proporcional puro do sistema massa-mola-amortecedor. Na resposta do controlador P, observamos um tempo de subida rápido, um sobressalto significativo e uma oscilação prolongada, resultando em um longo tempo de acomodação.

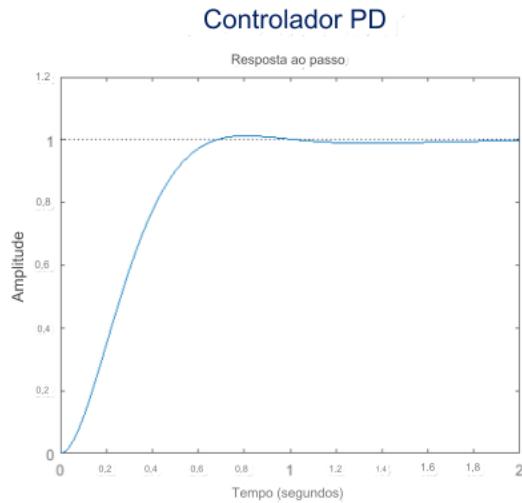


Figura 40 – Controlador PD ([University of Toronto, 2018a](#), Week 5 - Lesson 1: Proportional-Integral-Derivative (PID) Control. 12min13s).

A inclusão do controle derivativo ($G_{PD}(s) = K_P + sK_D$), visto no gráfico da Figura 40, melhora a resposta ao degrau em termos de sobressalto e tempo de acomodação, mas diminui o tempo de subida.

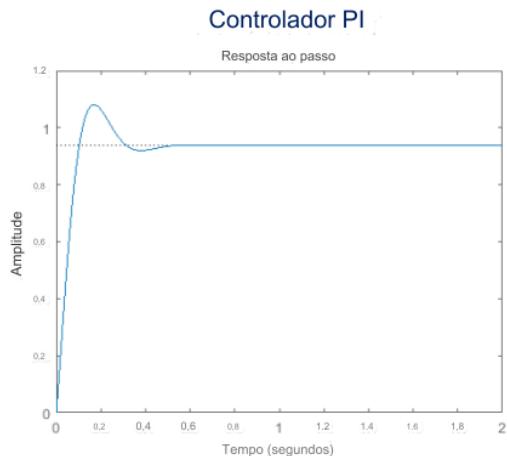


Figura 41 – Controlador PI ([University of Toronto, 2018a](#), Week 5 - Lesson 1: Proportional-Integral-Derivative (PID) Control. 12min13s).

A adição do termo integral ($G_{PI}(s) = K_P + \frac{K_I}{s}$), visto no gráfico da Figura 41, por outro lado, mantém um tempo de subida curto e consegue reduzir oscilações e sobressalto, resultando em um tempo de acomodação rápido também.

Incorporar todos os três termos PID no controlador proporciona ainda mais flexibilidade na criação da resposta ao degrau. Através da afinação cuidadosa dos ganhos do controlador, é possível aproveitar os benefícios de todos os três termos para eliminar sobressaltos e ainda manter tempos de subida e de estabilização muito curtos.

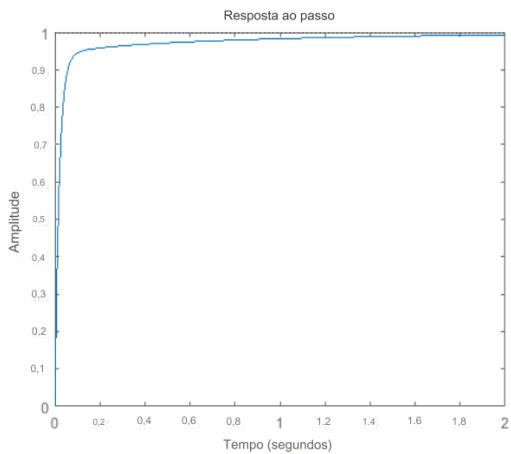


Figura 42 – Controlador PID ([University of Toronto, 2018a](#), Week 5 - Lesson 1: Proportional-Integral-Derivative (PID) Control. 12min50s).

Conforme evidenciado no gráfico da Figura 42, o sistema se aproxima rapidamente da referência, sem apresentar sobressalto, quando submetido ao controle PID.

3.1.4 Controle Lateral de Veículos

Esta seção aborda os aspectos essenciais do controle lateral para o desenvolvimento de VAs. Na seção 3.1.3 anterior, foram explorados conceitos fundamentais de controle longitudinal no contexto do desenvolvimento de carros autônomos. A partir dessa base, o foco agora se volta para o controle lateral, um elemento crucial para garantir um seguimento preciso do caminho por parte dos VAs.

As discussões a seguir, visam fornecer uma compreensão abrangente do controle lateral. A exploração começa com uma visão geral do controle lateral do veículo em um exemplo proposto pelo autor ([University of Toronto, 2018a](#)). Nesse sentido estudamos sobre estratégias de controle geométrico de seguimento de caminho, baseadas na suposição de aderência perfeita do modelo cinemático ([Jacobson et al., 2020](#)), e apresentamos sobre o Controlador de Perseguição Pura que usaremos no desenvolvimento da nossa implementação da seção 2.3. É importante salientar que ainda há controladores como o Controlador Stanley e o Controlador Preditivo de Modelo, ou MPC (do inglês: *Model Predictive Controller*), uma estratégia avançada frequentemente empregada em VAs, segundo o autor [University of Toronto \(2018a\)](#). Entretanto, não trataremos desses controladores, visto que não farão parte da solução deste trabalho.

Sabendo disso, o principal objetivo de um controlador lateral é garantir que o veículo consiga seguir precisamente uma trajetória predefinida, executando o plano de movimento ([Zheng et al., 2023](#), p. 2). Selecionando o ângulo de direção necessário para corrigir quaisquer erros que se acumulem e acompanhar as mudanças na direção da trajetória

conforme surgem. Segundo o autor ([University of Toronto, 2018a](#)), de modo a projetar um controlador lateral, é necessário definir os seguintes requisitos:

1. O erro entre a posição do veículo e as coordenadas desejadas adequadas da trajetória;
2. Selecionar uma estratégia de projeto de controle que leve os erros a zero, enquanto ainda atende as limites de ângulo de direção;
3. Considerar as limitações dinâmicas do veículo e as características desejadas do percurso, como aceleração lateral máxima e mínimo solavanco.

No geral, o comando de controle deve estar ciente das forças disponíveis nos pneus e não exceder as capacidades do veículo ao corrigir erros de trajetória.

O autor ([University of Toronto, 2018a](#)), evidencia que a trajetória do veículo seguirá um caminho de referência ou seguimento de trajetória com o sistema de planejamento no controlador lateral e pode ser definido das seguintes maneiras:

1. **Segmentos de linha reta:** sendo essa a abordagem mais simples, sendo definida por um segmento de linha reta, exigindo uma sequência de vértices conectados linearmente, conforme identificável na Figura 43. É salientando que essa definição de caminho pode ser muito compacta e fácil de construir, supondo que os pontos estejam espaçados adequadamente e se o ambiente permita principalmente movimento em linha reta, como em uma grade de vias urbanas de Manhattan. No entanto, o caminho inclui descontinuidades de direção, tornando o rastreamento preciso de uma trajetória desafiadora.

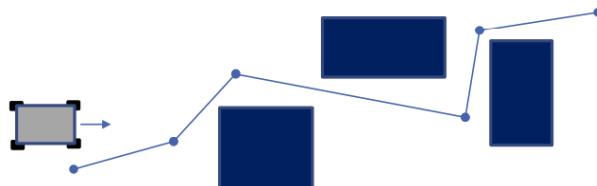


Figura 43 – Segmentos de linha reta ([University of Toronto, 2018a](#), Week 6 - Lesson 1: Introduction to Lateral Vehicle Control. 2min52s).

2. **Pontos de referência:** sendo esse uma versão melhorada do segmento de linha reta, onde o espaçamento entre os pontos é geralmente fixado em termos de distância ou tempo de deslocamento, conforme podemos identificar na Figura 44. Neste caso, as posições relativas dos pontos de passagem podem ser reduzidas para atender a uma restrição de curvatura. Segundo o autor ([University of Toronto, 2018a](#)), os caminhos pontilhados são bastante úteis devido à sua simplicidade de uso e à possibilidade de serem elaborados diretamente a partir de estimativas de estado ou pontos de passagem de GPS obtidos em operações anteriores de uma rota específica.

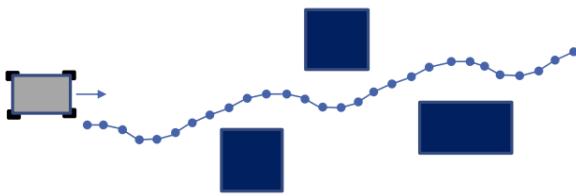


Figura 44 – Pontos de referência ([University of Toronto, 2018a](#), Week 6 - Lesson 1: Introduction to Lateral Vehicle Control. 3min00s).

3. Curvas parametrizadas: definido por um caminho utilizando uma sequência de curvas parametrizadas contínuas, as quais podem ser desenhadas a partir de um conjunto fixo de primitivas de movimento ou podem ser identificadas por meio de otimização durante o planejamento, podemos entender esse processo a partir da Figura 45. Onde oferecem a vantagem de um movimento continuamente variável e podem ser construídas de modo a possuir derivadas suaves, contribuindo para a consistência nos cálculos de erro e taxa de erro.

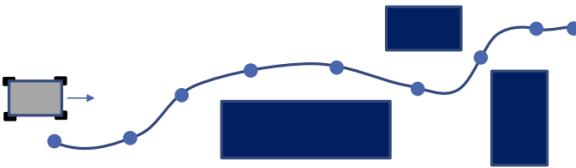


Figura 45 – Curvas parametrizadas ([University of Toronto, 2018a](#), Week 6 - Lesson 1: Introduction to Lateral Vehicle Control. 3min30s).

O autor saliente ([University of Toronto, 2018a](#)) que em todos esses casos de seguimento de trajetória, o controlador busca eliminar o desvio do veículo em relação à trajetória desejada e alinhar a orientação do veículo com a orientação da trajetória.

Desse modo, iniciaremos o entendimento de como se é dada a trajetória desejada no projeto de controle lateral, a qual, segundo o autor ([University of Toronto, 2018a](#)), é dividida em duas categorias principais. A primeira categoria de controladores são os controladores geométricos, que dependem da geometria e das coordenadas do caminho desejado, bem como dos modelos cinemáticos do veículo. Sendo esses, o controlador de perseguição pura “perseguidor de cenoura” e o controlador Stanley. A outra categoria de controladores é denominada controladores dinâmicos. O controlador avançado mais popular e comumente utilizado nessa categoria é o MPC, que realiza uma otimização de horizonte finito para identificar o comando de controle a ser aplicado ([University of Toronto, 2018a](#)).

De modo a compreendermos o funcionamento dos controladores laterais usaremos o modelo cinemático de bicicleta (elaborado na subseção 3.1.1.2), como sugerido pelo autor [University of Toronto \(2018a\)](#), como as bases para a elaboração desta discussão, para isso nos baseamos nos autores ([University of Toronto, 2018a; Rajamani, 2011; Francis et al.,](#)

2016; Snider *et al.*, 2009). Desse modo, para esta discussão, é utilizado um segmento de reta como nossa trajetória de referência, representada por uma linha preta contínua, vista na Figura 46.

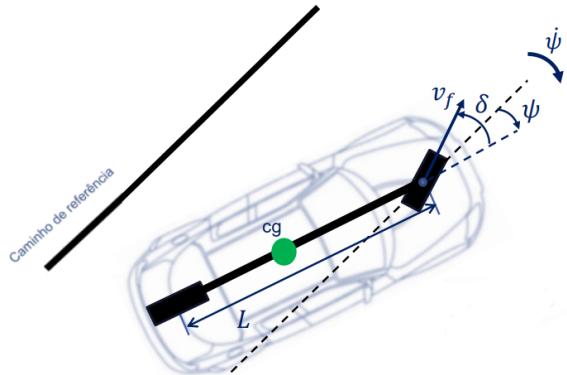


Figura 46 – Trajetória de referência ([University of Toronto, 2018a](#), Week 6 - Lesson 1: Introduction to Lateral Vehicle Control. 6min23s).

Nesse cenário, podemos ter 2 tipos de erros: erro de rumo e erro de trajetória cruzada, segundo o autor [University of Toronto \(2018a\)](#). O erro de rumo é igual à diferença entre a orientação da trajetória e a orientação do veículo no ponto de referência ao longo do caminho. Trata-se de uma medida fundamental que avalia quão bem o veículo está alinhado e movendo-se na direção do caminho desejado. A taxa de erro de rumo ($\dot{\psi}$) auxilia na compreensão de como o erro de rumo evolui ao longo do tempo e pode ser calculada a partir das equações do modelo cinemático de bicicleta ([Snider *et al.*, 2009](#), p. 18), conforme visto na equação 3.20 resolvida em relação ao eixo dianteiro.

$$\dot{\psi}_{\text{des}}(t) - \dot{\psi}(t) = \frac{v_f(t) \sin \delta(t)}{L} \quad (3.20)$$

Onde: $\dot{\psi}_{\text{des}}(t)$ = Taxa de erro de rumo no tempo t

$\dot{\psi}(t)$ = Rumo atual no tempo t

$v_f(t)$ = Velocidade para frente no tempo t

$\sin \delta(t)$ = Seno do ângulo de direção no tempo t

L = Comprimento da distância entre eixos.

Para segmentos de reta, a taxa desejada de mudança de direção é zero, e pode ser desconsiderada da equação 3.20. Segundo o autor [University of Toronto \(2018a\)](#), isso ocorre porque a direção de referência não varia com o tempo para uma reta, uma vez que redefinimos nossa direção em relação à direção atual do caminho, resultando na equação 3.21.

$$\dot{\psi}(t) = \frac{-v_f(t) \sin \delta(t)}{L} \quad (3.21)$$

Como mencionado, o outro tipo de erro é um erro de desvio chamado de erro de trajetória cruzada, também conhecido como “*Crosstrack Error*” no inglês. O autor University of Toronto (2018a), frisa que o erro de trajetória cruzada é a distância entre o ponto de referência no veículo e o ponto mais próximo na trajetória de caminho desejada. A qual é a principal medida de quão próxima à posição do veículo está da posição desejada ao longo do caminho de referência. Obs.: Tanto o erro de rumo quanto o erro de trajetória cruzada devem convergir para zero para o veículo rastrear adequadamente o caminho desejado (University of Toronto, 2018a).

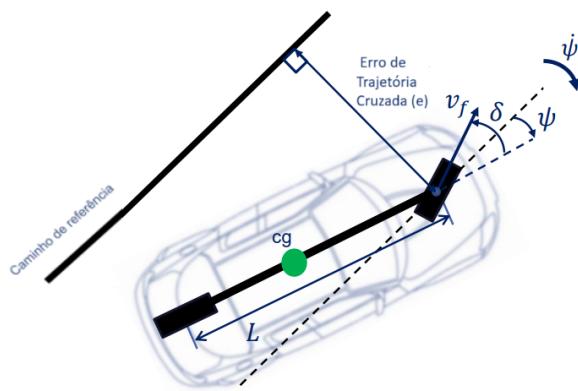


Figura 47 – Erro de trajetória cruzada (University of Toronto, 2018a, Week 6 - Lesson 1: Introduction to Lateral Vehicle Control. 8min31s).

Desse modo, a taxa de variação do erro de trajetória cruzada, visto na Figura 47, pode ser calculada extraindo o componente lateral da velocidade de avanço, conforme apresentado na equação 3.22.

$$\dot{e}(t) = v_f(t) \sin(\psi(t) - \delta(t)) \quad (3.22)$$

Onde:	$\dot{e}(t)$	= Taxa de variação do erro de trajetória cruzada no tempo t
	$v_f(t)$	= Velocidade para frente no tempo t
	$\sin(\psi(t) - \delta(t))$	= Seno da diferença entre o ângulo do curso e o ângulo de direção no tempo t
	$\psi(t)$	= ângulo do curso no tempo t
	$\delta(t)$	= ângulo de direção no tempo t

A equação 3.22 revela que, à medida que a velocidade aumenta, o erro de trajetória cruzada varia mais rapidamente. Conforme indicado pelo autor (University of Toronto, 2018a), esse comportamento implica na necessidade de aplicar ângulos de direção menores para corrigir erros de trajetória cruzada de magnitude semelhante. No entanto, não nos aprofundaremos nesta correção neste trabalho, uma vez que já estudamos detalhadamente os temas relevantes para alcançar os Objetivos 1.3 deste trabalho, como caminho de

referência, controladores existentes: MPC, Stanley, e na subseção subsequente Pure Pursuit 3.1.4.1, erro de rumo, e erro de trajetória cruzada.

3.1.4.1 Controle Lateral Geométrico - Controlador de Perseguição Pura

A seção 3.1.4 anterior proporcionou uma definição dos conceitos essenciais para o controle lateral de veículos. Nesta subseção, o foco inicial será na introdução do conceito de um controlador geométrico de trajetória, que utiliza o modelo cinemático do veículo para selecionar comandos de direção. Em seguida, será desenvolvido um controlador de perseguição pura (no inglês: *Pure Pursuit*) para VAs, que visa seguir uma trajetória de referência no ambiente.

Segundo o autor ([University of Toronto, 2018a](#)), o controlador geométrico de trajetória é, genericamente, qualquer controlador que acompanha uma trajetória de referência usando apenas a geometria da cinemática do veículo e da trajetória de referência. Especificamente para VAs, esse controlador é uma forma de controle lateral que desconsidera as forças dinâmicas no veículo, assumindo que a condição sem deslizamento se mantém nos pneus. Este controlador se baseia em um modelo cinemático de bicicleta e em medidas de erro definidas na seção 3.1.4 para construir uma regra de comando de direção que alcance a perseguição de trajetória.

Em seguida, apresentaremos o controlador de perseguição pura, onde um ponto de referência é colocado em uma trajetória a uma distância fixa à frente do veículo, e os comandos de direção necessários para interceptar esse ponto usando um ângulo de direção constante são calculados. Este método utiliza o ponto central do eixo traseiro como ponto de referência no veículo e define a linha que conecta o centro do eixo traseiro ao ponto de referência alvo como uma linha de distância fixa.

Com base nesse contexto, começaremos apresentando os controladores geométricos de seguimento de trajetória, os quais se fundamentam em um ponto de referência ao longo da trajetória desejada. Esse ponto pode coincidir com o utilizado para calcular os erros de orientação e desvio lateral, ou, conforme discutiremos nesta subseção, pode ser um ponto de antecipação posicionado a uma determinada distância à frente do veículo ao longo da trajetória, como ilustrado na Figura 48 como um ponto vermelho.

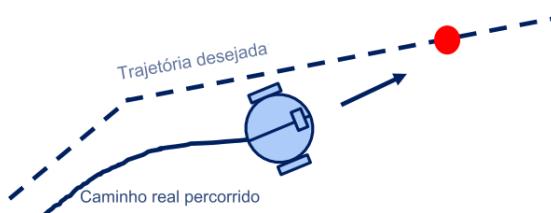


Figura 48 – Rastreamento de caminho geométrico ([University of Toronto, 2018a](#), Week 6 - Lesson 2: Geometric Lateral Control - Pure Pursuit. 2min27s).

No método de perseguição pura usaremos esse ponto vermelho da Figura 48, onde a sua ideia central, segundo os autores University of Toronto (2018a), e Snider *et al.* (2009, p. 9): é que um ponto de referência pode ser colocado na trajetória a uma distância fixa à frente do veículo, e os comandos de direção necessários podem ser calculados para interseccionar esse ponto usando um ângulo de direção constante. Desse modo, à medida que o veículo vira em direção à trajetória para seguir essa curva, o ponto continua a se movimentar para frente, reduzindo o ângulo de direção e levando suavemente o veículo em direção à trajetória.

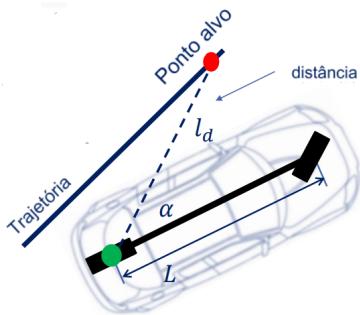


Figura 49 – Pure pursuit (University of Toronto, 2018a, Week 6 - Lesson 2: Geometric Lateral Control - Pure Pursuit. 3min15s).

O ponto de referência no veículo corresponde ao centro do eixo traseiro, conforme destacado na Figura 49. Além disso, observamos haver uma linha que conecta o centro do eixo traseiro ao ponto de referência desejado, estabelecendo uma distância fixa l_d conhecida como distância de antecipação. Essa distância é representada pela linha tracejada que se estende até o ponto vermelho. O ângulo formado entre a direção da carroceria do veículo ao ponto vermelho é denominado α .

Segundo o autor University of Toronto (2018a), para elaborar a lei do controlador de perseguição pura, precisamos dos conceitos sobre centro instantâneo de rotação. Nesse caso, o ponto de destino na trajetória (ponto vermelho), o centro do eixo traseiro e o centro instantâneo de rotação formam um triângulo com dois lados de comprimento R e um de comprimento l_d , conforme podemos identificar na Figura 50.

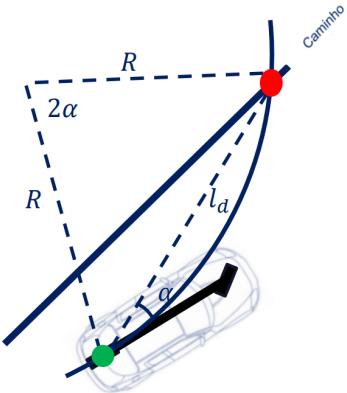


Figura 50 – Pure pursuit ICR ([University of Toronto, 2018a](#), Week 6 - Lesson 2: Geometric Lateral Control - Pure Pursuit. 3min52s).

Segundo o autor [University of Toronto \(2018a\)](#), precisamos definir o arco que leva o ponto de referência do veículo ao ponto de destino na trajetória. Este arco é a parte do círculo do ICR que abrange o ângulo de 2α e pode ser derivado usando identidades trigonométricas padrão, conforme a equação 3.23 escrita baseada na lei dos senos ([Snider et al., 2009](#), p. 9):

$$\frac{L_d}{\sin 2\alpha} = \frac{R}{\sin \left(\frac{\pi}{2} - \alpha \right)} \quad (3.23)$$

Então, usando mais algumas identidades trigonométricas, podemos chegar na equação 3.24 simplifica.

$$\frac{l_d}{2 \sin \alpha \cos \alpha} = \frac{R}{\cos(\alpha)} \quad (3.24)$$

O que leva à expressão compacta da equação 3.25.

$$\frac{l_d}{\sin \alpha} = 2R \quad (3.25)$$

Finalmente, a curvatura κ , a qual é a inversa do raio de arco R , pode ser expressa conforme na equação 3.26.

$$\kappa = \frac{1}{R} = \frac{2 \sin \alpha}{l_d} \quad (3.26)$$

Para calcular o ângulo de direção precisamos rastrear esse arco, e isso é alcançado a partir do modelo da bicicleta, conforme podemos identificar na Figura 50. Nesse cenário, temos que o ângulo de direção define o raio da curva ([Snider et al., 2009](#), p. 10), e podemos estabelecer a equação 3.27.

$$\delta = \tan^{-1} k L \quad (3.27)$$

Segundo o autor [University of Toronto \(2018a\)](#), combinando esta expressão (δ) da equação 3.27 com a equação 3.26 (κ), podemos expressar o ângulo de direção necessário para seguir a curva, conforme a equação 3.28.

$$\delta = \tan^{-1} \left(\frac{2L \sin \alpha}{l_d} \right) \quad (3.28)$$

De modo a compreender essa implementação para direção, precisamos nos aprofundar em como os valores de erro evoluem em circuito fechado, para o controlador de perseguição pura que estamos trabalhando nesta subseção.

Desse modo, podemos definir o erro de trajetória cruzada (e) como a distância entre o vetor de orientação e o ponto alvo (círculo em vermelho), conforme podemos identificar na Figura 51, e a expressão do erro na equação 3.29.

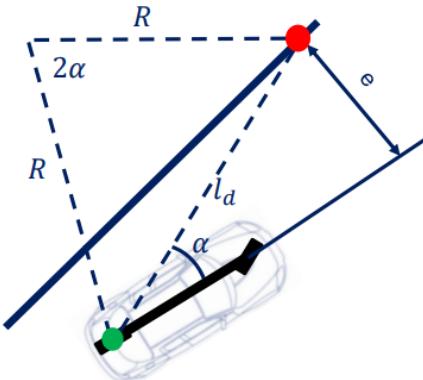


Figura 51 – Erro Crosstrack ([University of Toronto, 2018a](#), Week 6 - Lesson 2: Geometric Lateral Control - Pure Pursuit. 3min52s).

$$\sin \alpha = \frac{e}{l_d} \quad (3.29)$$

Ao combinar a equação 3.29 com a expressão da equação 3.26, o autor [University of Toronto \(2018a\)](#) salienta que podemos observar que a curvatura da trajetória gerada pelo controlador de perseguição pura é proporcional ao erro transversal no ponto de referência de busca à frente (ponto vermelho). O resultado dessa combinação pode ser observado na equação 3.30.

$$\kappa = \frac{2}{l_d^2} e \quad (3.30)$$

O autor [University of Toronto \(2018a\)](#) reforça que a medida que o erro aumenta, a curvatura também se intensifica, reposicionando o veículo mais assertivamente na trajetória. A equação 3.30 evidencia que o controlador de perseguição pura opera de maneira semelhante ao controle proporcional, visto em subseções anteriores, para corrigir o erro de desvio lateral, utilizando a curvatura da trajetória como saída do controlador.

Entretanto, será necessário incluir uma modificação no controle de curvatura para considerar diferentes valores de velocidade do veículo. Desse modo, podemos ajustar a distância de antecipação (l_d) com base na velocidade do veículo ($K_{dd}v_f$).

Substituindo este ajuste nas equações 3.28 e 3.30, chegamos ao controlador completo de perseguição pura ([Snider et al., 2009](#), p. 10), conforme podemos identificar na equação 3.31.

$$\delta = \tan^{-1} \left(\frac{2L \sin \alpha}{K_{dd} v_f} \right) \quad (3.31)$$

Onde: δ = Ângulo de direção;

$2L \sin \alpha$ = Duas vezes o produto do comprimento entre os eixos do veículo e o seno do ângulo do alvo;

$K_{dd} v_f$ = Produto do ganho proporcional pela velocidade para frente.

O controlador seleciona o ângulo de direção que formará um arco até o ponto de referência antecipado e ajusta este ponto de referência para ficar mais distante à medida que o veículo está se deslocando mais rapidamente. Este projeto resulta em comandos de direção e taxas de curva que são alcançáveis dadas as forças disponíveis nos pneus, embora seja necessário ajustá-lo para atingir esse objetivo, segundo o autor [University of Toronto \(2018a\)](#). Sendo esses ajustes tratados a fundo pelo autor [Snider et al. \(2009, p. 11\)](#).

Dessa forma, com o desenvolvido nesta subseção, temos o conhecimento necessário para construir controladores laterais geométricos de perseguição pura para nossa solução de carro autônomo.

3.2 Planejamento de Movimento de Carros Autônomos

O planejamento hierárquico é uma abordagem estruturada para resolver o problema de planejamento de movimento em VAs, dividindo o problema em uma sequência de problemas de otimização organizados em níveis de abstração. Essa hierarquia permite lidar com a complexidade do problema de maneira modular e eficiente, facilitando a execução em tempo real.

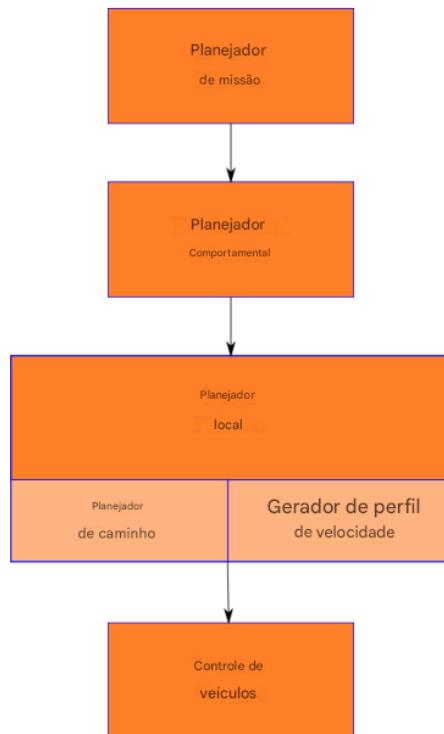


Figura 52 – Planejamento Hierárquico ([University of Toronto, 2018b](#), Module 2 - Lesson 1: Driving Missions, Scenarios, and Behaviour. 10min55s).

No topo da hierarquia da Figura 52 encontra-se o planejamento de missão, responsável por definir a rota global que o veículo deve seguir para navegar do ponto atual ao destino, considerando mapas, conexões viárias e o fluxo de tráfego. Essa etapa abstrai variáveis de baixo nível, como obstáculos e outros agentes, focando na eficiência do trajeto em termos de tempo ou distância.

O segundo nível corresponde ao planejamento de comportamento, que determina as ações do veículo em função do cenário de direção atual. Essas ações incluem manutenção de faixa, mudanças de faixa, curvas em interseções e manobras como retornos, considerando elementos regulatórios, como semáforos e placas de sinalização.

Enquanto, o planejamento local opera em menor escala, gerando trajetórias livres de colisões e perfis de velocidade detalhados para alcançar o estado final desejado. Essa etapa pode ser dividida em duas subetapas: planejamento de caminho, para calcular a

trajetória espacial, e geração de perfil de velocidade, para assegurar uma movimentação suave e segura, o qual buscamos para nossa solução quando uma placa de velocidade for identificada pela camada de percepção.

Por fim, chega a camada de Controle de Veículos a qual desenvolvemos na seção [3.1](#).

O planejamento hierárquico considera diversos cenários de condução, que variam em complexidade e dependem de fatores como a estrutura da via e a presença de obstáculos estáticos ou dinâmicos ([University of Toronto, 2018b](#)).

Quanto a objetos estáticos temos, por exemplo, placas de parada. Neste contexto, as Máquinas de Estados Finitos (FSMs) surgem como uma abordagem robusta para lidar com o problema do planejamento comportamental. As FSMs são baseadas em dois componentes principais: estados e transições. Os estados representam manobras específicas que o VA pode executar, como "acelerar", "frear" ou "parar em um sinal". As transições, por sua vez, determinam como o sistema evolui de um estado para outro, dependendo das entradas recebidas, como a posição de veículos próximos, a mudança de um sinal de trânsito ou a presença de pedestres ([University of Toronto, 2018b](#)).

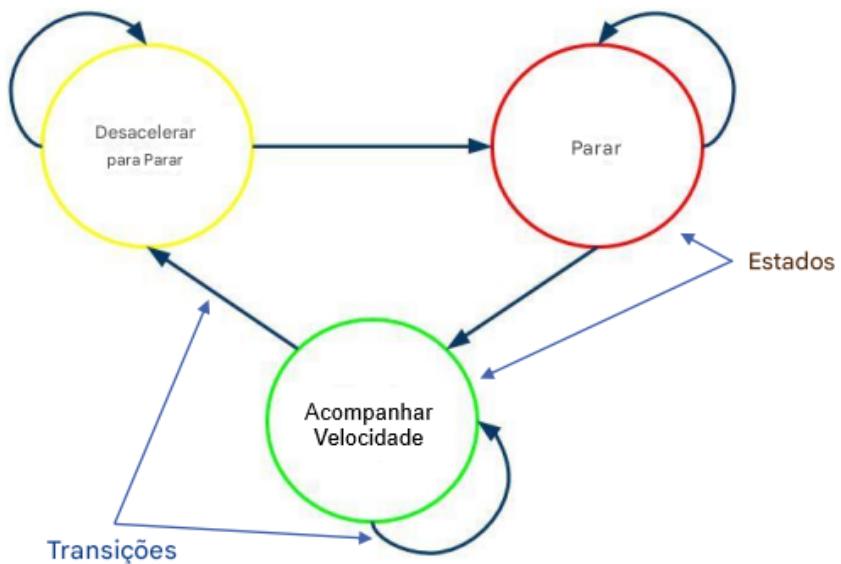


Figura 53 – Máquinas de Estados Finitos ([University of Toronto, 2018b](#), Module 2 - Lesson 4: Hierarchical Motion Planning. 7min54s).

Conforme a Figura 53, ao encontrar um sinal de "Pare", o FSM poderia transitar entre os seguintes estados:

- 1. Desacelerar para Parar:** o planejador sinaliza a necessidade de reduzir a velocidade.

2. **Parar:** após atingir a posição correta, o veículo mantém-se parado por um tempo determinado.
3. **Acompanhar Velocidade:** ao liberar o cruzamento, o FSM ordena o prosseguimento seguro.

O planejador ainda precisa considerar o planejamento local, que exige a geração de trajetórias cinemática e dinamicamente viáveis, livres de colisões que garantam o conforto e segurança dos ocupantes. Nesse cenário, os Lattice planners, visto na Figura 54, têm se destacado como uma abordagem eficiente e robusta para o planejamento de movimento em sistemas autônomos, abordando restrições diferenciais e limitações do espaço de estados. Esses planejadores reduzem a complexidade computacional do problema ao restringir o espaço de busca a um conjunto limitado de ações que o veículo pode executar em qualquer ponto do ambiente. Esse conjunto, denominado conjunto de controle, é emparelhado com uma discretização do espaço de trabalho, formando implicitamente um grafo que pode ser explorado por algoritmos de busca como Dijkstra ou A*. Esse processo permite não apenas a busca eficiente por trajetórias, mas também a verificação de colisões, atribuindo custos infinitos às arestas que cruzam obstáculos (University of Toronto, 2018b).

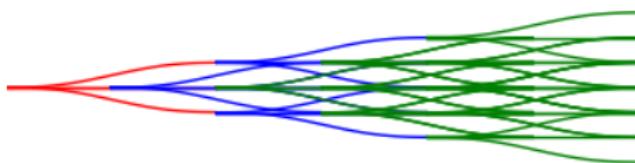


Figura 54 – Lattice Planner (University of Toronto, 2018b, Module 2 - Lesson 4: Hierarchical Motion Planning. 15min30s).

A discretização proposta em um estado lattice é uma representação regular do espaço de estados, onde cada vértice corresponde a um estado alcançável e cada aresta representa um movimento exato e factível. Essa estrutura regular permite a reutilização de um conjunto canônico de arestas, mantendo a propriedade de conectividade exata entre vértices vizinhos. O conjunto de controle, nesse contexto, é definido pelo fator de ramificação do grafo, isto é, o número de arestas conectando cada vértice. Isso resulta em um grafo estruturado que facilita a aplicação de algoritmos de busca para encontrar movimentos viáveis que satisfaçam as restrições ambientais e diferenciais, preservando a regularidade do espaço de busca (Pivtoraiko; Knepper; Kelly, 2009).

Segundo o autor University of Toronto (2018b), uma variação comumente utilizada dos planejadores de lattice é o conformal lattice planner, no qual pontos-alvo são definidos em posições à frente do veículo, lateralmente deslocados em relação à direção da via, conforme podemos identificar na Figura 55.

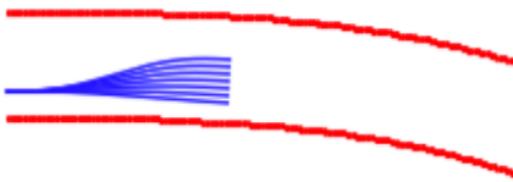


Figura 55 – Conformal Lattice Planner ([University of Toronto, 2018b](#), Module 2 - Lesson 4: Hierarchical Motion Planning. 15min30s).

Para cada ponto, uma trajetória é otimizada, e a trajetória selecionada é aquela que melhor satisfaz os objetivos definidos, como minimização de energia de curvatura, enquanto permanece livre de colisões. Adicionalmente, a geração de perfis de velocidade é tratada como um problema de otimização restrita, integrando objetivos como minimizar variações bruscas de aceleração e respeitar limites de conforto dinâmico. Métodos para aproximações convexas são frequentemente empregados para evitar mínimos locais durante a otimização ([University of Toronto, 2018b](#)).

Os planejadores de lattice, ao integrarem restrições diferenciais diretamente na estrutura do grafo, conseguem se destacar por oferecerem soluções determinísticas e eficientes. Eles não apenas abordam desafios relacionados à viabilidade dos movimentos, mas também otimizam trajetórias nos limites impostos pelas dinâmicas do sistema. Isso os torna ferramentas essenciais para o planejamento de movimento em sistemas autônomos, equilibrando desempenho computacional e qualidade das trajetórias ([Pivtoraiko; Knepper; Kelly, 2009](#)).

3.3 Percepção Visual de Carros Autônomos

Essa seção aborda os principais aspectos relacionados à percepção visual em carros autônomos, destacando o uso de câmeras e algoritmos avançados de detecção de objetos, como o YOLO (*You Only Look Once*). Inicialmente, exploraremos o papel das câmeras, descritas na subseção 3.3.1, suas características e limitações, bem como sua aplicação no contexto do simulador CARLA. Em seguida, na subseção 3.3.2, apresentaremos o algoritmo YOLO, suas vantagens em relação a outros métodos, e sua arquitetura, detalhada na subseção 3.3.2.1. Por fim, discutiremos o treinamento do YOLO, os conjuntos de dados utilizados e como esses elementos serão integrados à nossa solução para detecção e classificação de objetos em tempo real no ambiente simulado.

3.3.1 Câmeras

As Câmeras são uma das tecnologias mais utilizadas para observar o ambiente, e é o melhor sensor para classificação de objeto (Khan *et al.*, 2022, p. 6). Uma câmera produz imagens nítidas dos arredores detectando as luzes emitidas de uma superfície fotossensível (plano de imagem) usando uma lente de câmera (colocada na frente do sensor) (Ignatious; Hesham-El-Sayed; Khan, 2022). Os VAs possuem esses sensores de luz visível para fornecer uma visão de 360 graus do ambiente. As câmeras são ótimos na detecção e reconhecimento de objetos, fornecendo detalhes mais ricos e ajudando a entender os objetos sem ou com profundidade, que geralmente não são detectados por outros tipos de sensores. Dentre esses objetos estão: Sinais de trânsito (limite de velocidade, sinais de parada, sinais de ultrapassagem), semáforos, pedestres, animais são alguns exemplos de tais objetos sem ou com profundidade (Khan *et al.*, 2022). Esses dados coletados com as câmeras são enviados para os algoritmos baseados em IA para uso posterior, e criação de uma imagem 2D (Singh, 2022). No entanto, esses sensores são imprecisos em ambientes escuros e geram uma abundância de dados para processar. Outros tipos de câmeras como as infravermelhas, também, são utilizadas para melhor desempenho em condições de baixa visibilidade (Parekh Nishi Poddar, 2022). As câmeras são dispostas nos VAs como mostrado na Figura 11, e para detalhes específicos dos diferentes modelos de câmeras.

No cenário de simulação, câmeras e outros sensores podem ser adicionados ao veículo definindo as configurações enviadas pelo cliente. Há quatro sensores de câmera diferentes disponíveis no simulador CARLA: "Scene final", "Depth map", "Semantic segmentation", e "Ray-cast based lidar" (CARLA Simulator, 2024). Sendo a câmera "Scene final" a escolhida para a solução de detecção e classificação de objetos, por trazer imagens mais próximas ao mundo real. A câmera "Scene final" fornece uma visão da cena após aplicar alguns efeitos de pós-processamento para criar uma sensação mais realista como visto na Figura 56.

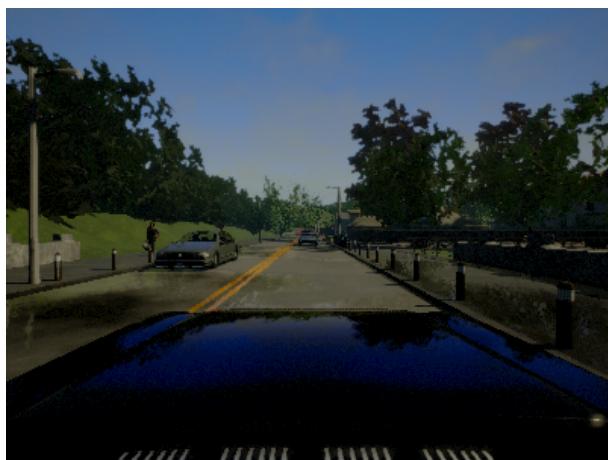


Figura 56 – Imagem a partir da Camera: Scene final (CARLA Simulator, 2024).

No trecho de código 3.1 podemos ver um exemplo dessa câmera adicionado ao

cliente do simulador Carla.

```

1 # Cria uma câmera no simulador CARLA com pós-processamento Scene Final
2 camera = carla.sensor.Camera('MyCamera', PostProcessing='SceneFinal')
3 # Define o campo de visão da câmera
4 camera.set(FOV=90.0)
5 # Define o tamanho da imagem gerada pela câmera
6 camera.set_image_size(800, 600)
7 # Configura a posição da câmera em relação ao veículo
8 camera.set_position(x=0.30, y=0, z=1.30)
9 # Define a rotação da câmera
10 camera.set_rotation(pitch=0, yaw=0, roll=0)
11 # Adiciona a câmera configurada ao cenário
12 carla_settings.add_sensor(camera)

```

Lista de código 3.1 – Exemplo da documentação CARLA para adicionar uma câmera “Scene final” no simulador ([CARLA Simulator, 2024](#)).

3.3.2 You Only Look Once - YOLO

Apresentado na subseção 2.2.1.2 juntamente com trabalhos relacionados. Essa subseção visa apresentar como o YOLO foi projetado e como o usaremos para nossa solução de detecção e classificação de imagens em tempo real no simulador. Assim como a arquitetura unificada do YOLO e sua capacidade de operar em velocidades de tempo real, sem comprometer a precisão média, tendo como um dos principais benefícios a utilização do contexto global da imagem durante as previsões, ao contrário de métodos baseados em classificadores que analisam regiões isoladas. Além disso, o YOLO realiza todas as previsões com uma única avaliação da rede, diferentemente de sistemas como o R-CNN, que requer milhares de avaliações para processar uma única imagem ([Redmon, 2018](#)). Essa característica confere ao YOLO uma velocidade excepcional, características essenciais para lidar com os desafios em sistemas autônomos, sendo mais de 1.000 vezes mais rápido que o R-CNN e 100 vezes mais rápido que o Fast R-CNN, conforme detalhado em trabalhos anteriores sobre o sistema [Redmon e Farhadi \(2018\)](#).

3.3.2.1 Arquitetura

O YOLO adota uma abordagem unificada para a detecção de objetos, integrando componentes tradicionalmente separados em uma única rede neural. Essa rede processa a imagem inteira para prever caixas delimitadoras (*bounding boxes*) e classes simultaneamente, raciocinando globalmente sobre o conteúdo completo da imagem ([Redmon et al., 2016](#), p. 1).

A entrada da rede é dividida em uma grade $S \times S$, onde cada célula da grade é responsável por detectar objetos cujo centro está contido na célula. Para cada célula, a

rede prevê B caixas delimitadoras e uma pontuação de confiança associada, refletindo a probabilidade de presença de um objeto e a precisão da previsão, com base no índice de interseção sobre união (*Intersection over Union - IOU*¹) entre a previsão e o valor real (Redmon *et al.*, 2016, p. 1).

Cada caixa delimitadora contém cinco valores: x , y , w , h e a confiança. As probabilidades condicionais de classe são multiplicadas pelas pontuações de confiança das caixas. Esse processo pode ser melhor entendido a partir da Figura 57 (Redmon *et al.*, 2016, p. 1).

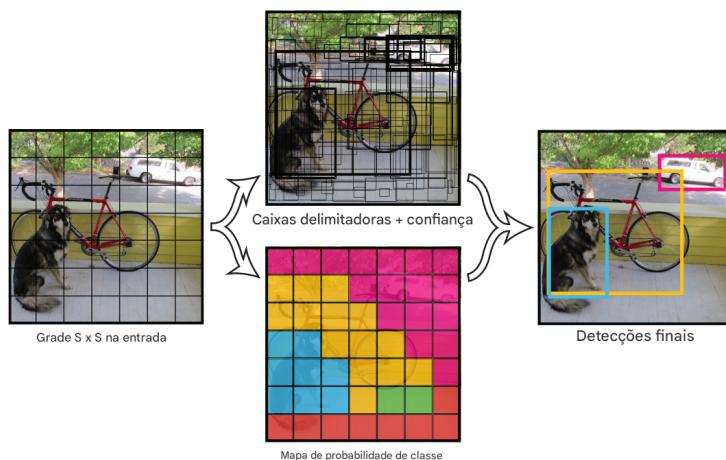


Figura 57 – Modelo. O sistema modela a detecção como um problema de regressão. Dividindo a imagem em uma grade $S \times S$ e para cada célula da grade prevê caixas delimitadoras B , confiança para essas caixas e probabilidades de classe C . Segundo os autores, essas previsões são codificadas como um tensor $S \times S \times (B \times 5 + C)$. Redmon *et al.* (2016, p. 2).

A arquitetura da rede é baseada no modelo GoogLeNet, combinando 24 camadas convolucionais seguidas por 2 camadas totalmente conectadas. As camadas convolucionais iniciais extraem características da imagem, enquanto as camadas totalmente conectadas preveem as probabilidades de saída e as coordenadas. Reduções 1×1 e convoluções 3×3 alternadas são usadas para reduzir a dimensionalidade e manter a robustez do modelo (Redmon *et al.*, 2016, p. 2).

¹ IOU é uma métrica de avaliação de detectores de objetos que mede a taxa de sobreposição entre a verdade básica e os limites previstos.

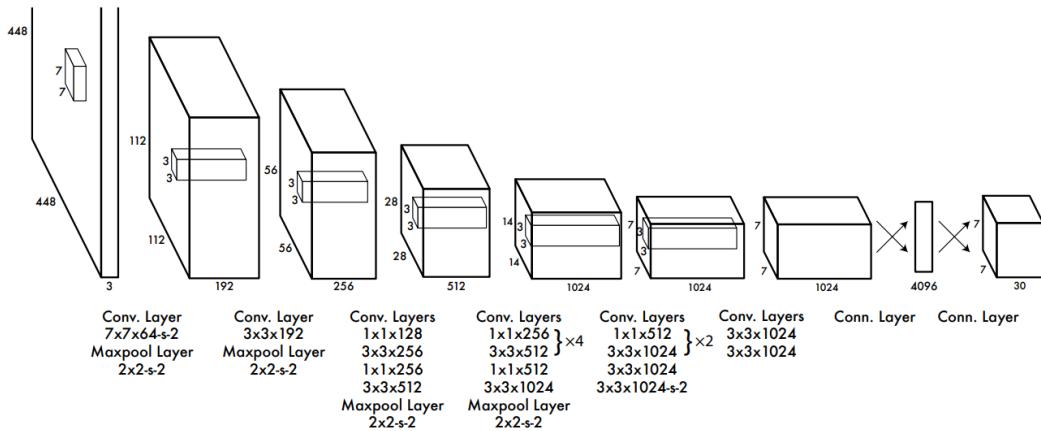


Figura 58 – Arquitetura. A rede de detecção tem 24 camadas convolucionais seguidas por 2 camadas totalmente conectadas. Camadas convolucionais alternadas 1×1 reduzem o espaço de recursos das camadas precedentes. Nós pré-treinamos as camadas convolucionais na tarefa de classificação do ImageNet na metade da resolução (imagem de entrada 224×224) e então dobramos a resolução para detecção (Redmon *et al.*, 2016, p. 3).

A Figura 58 apresenta o design da arquitetura, que inclui camadas convolucionais pré-treinadas no conjunto de dados ImageNet para classificação e adaptadas posteriormente para detecção com resolução ampliada.

3.3.2.2 Treinamento

O treinamento do YOLO segue uma abordagem cuidadosamente estruturada para otimizar sua eficiência e precisão. Segundo Redmon *et al.* (2016), a rede neural é inicialmente pré-treinada em um conjunto de dados amplo, como o ImageNet, que contém 1.000 classes, permitindo à rede aprender características visuais genéricas. Esse pré-treinamento utiliza as primeiras 20 camadas convolucionais, seguidas por uma camada de *pooling* média e uma camada totalmente conectada, alcançando uma precisão *top-5* de 88% na validação do ImageNet. Após essa fase, a rede é adaptada para detecção de objetos, adicionando camadas convolucionais e totalmente conectadas com pesos inicializados aleatoriamente, além de aumentar a resolução de entrada de 224×224 para 448×448 , garantindo maior detalhamento visual, conforme podemos identificar na Figura 58 e 59 que mostra o esquema global desse tipo de sistema para detecção. Uma imagem de entrada capturada com uma câmera é alimentada para o algoritmo de detecção de objetos YOLO, e por meio de uma rede neural convolucional profunda 58 ele detecta objetos e emite sinais de trânsito isolados quando apropriado. No contexto da nossa solução teremos a caixa delimitadora da detecção.

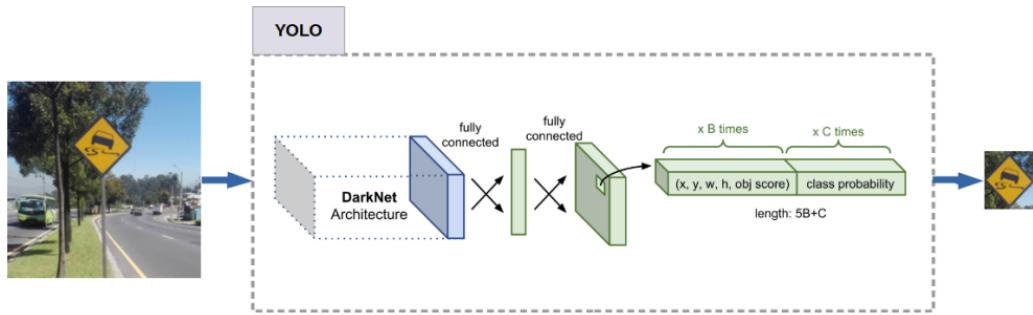


Figura 59 – Esquema geral de um sistema de detecção e reconhecimento de sinais de trânsito usando o algoritmo de detecção de objetos YOLO. Imagem de (Flores-Calero *et al.*, 2024, p. 2).

Entrada: Uma imagem é fornecida como entrada (neste caso, uma estrada com uma placa de trânsito).

Arquitetura Darknet: A imagem passa pela rede convolucional Darknet para extração de características.

Camadas totalmente conectadas: As características extraídas são processadas por camadas totalmente conectadas para gerar as previsões.

Predição de caixas delimitadoras: Para cada âncora (B), são calculados:

- x, y : Coordenadas do centro da caixa delimitadora.
- w, h : Largura e altura da caixa.
- obj : Confiança de que a caixa contém um objeto.

Probabilidade de classe: Para cada classe (C), é calculada a probabilidade de que o objeto na caixa pertence a essa classe.

Saída final: A posição da placa de trânsito e sua categoria são destacadas como a previsão final.

Segundo o autor Redmon *et al.* (2016), a função de perda do YOLO é projetada para equilibrar diferentes aspectos da detecção, como localização, confiança e classificação, empregando pesos distintos para as contribuições individuais ao erro. Durante o treinamento, a perda relacionada às coordenadas das caixas delimitadoras é ampliada para aumentar a precisão de localização, enquanto a perda associada a caixas sem objetos é reduzida, evitando a predominância de gradientes irrelevantes. O uso do índice de IOU, apresentado inicialmente 3.3.2.1, orienta a atribuição de responsabilidade às caixas preditoras, promovendo especialização e melhorando a capacidade da rede em prever objetos de tamanhos e proporções variados.

A regularização e a generalização são asseguradas por técnicas como *dropout* e aumento de dados. Aplicado após a primeira camada totalmente conectada, prevenindo sobreajuste/*overfitting* entre camadas.

Nesta trabalho, utilizaremos o conjunto disponibilizado por ([Juanola, 2019](#)) e ([Daniel, 2023](#)), que contém imagens extraídas do simulador CARLA. Este conjunto foi mencionado previamente na seção de métodos ([1.5](#)). O treinamento será realizado seguindo as etapas da documentação do YOLO ([Redmon, 2018](#)) e teremos como base o procedimento de treinamento realizado pelo autor [Juanola \(2019\)](#).

4 Implementação

4.1 Configurando o Simulador Carla

Nesta subseção, apresentaremos como o simulador Carla foi configurado para desenvolver a solução.

Primeiramente, precisamos dos seguintes pré-requisitos para usarmos o simulador ([University of Toronto, 2018a](#)):

Pré-requisitos

Especificações de Hardware recomendadas:

- Processador Intel ou AMD quad-core, 2.5 GHz ou mais rápido
- NVIDIA GeForce 470 GTX ou AMD Radeon 6870 HD series ou superior
- 8 GB de RAM
- Aproximadamente 10GB de espaço em disco para a configuração do simulador

Nota: Computadores com especificações inferiores, incluindo sistemas com gráficos integrados, também podem executar o simulador CARLA, embora com desempenho reduzido.

Especificações de Software:

Windows/Ubuntu: CARLA requer Windows 7 64-bit ou Ubuntu 16.04 (ou superior). **Firewall:** Foi necessário habilitar a rede e permitir o acesso do firewall ao carregador do CARLA e, por padrão, às portas 2000, 2001 e 2002 (TCP e UDP).

Drivers da Placa Gráfica: Atualizamos os drivers mais recentes para evitar problemas gráficos. No caso, OpenGL 3.3 ou superior e DirectX 10 como recomendado.

Python:

- Foi instalado Python 3.6.0 com *pip*. O cliente Python do CARLA funciona com Python 3.5.x ou Python 3.6.x.
- Segundo o autor, Python 3.7 atualmente não é compatível.

Preparando o Simulador CARLA

Baixando e extraindo o Simulador CARLA

1. Baixamos o simulador CARLA (`CarlaUE4Windows.zip`), o qual pode ser encontrado no capítulo [A](#) (apêndice) deste trabalho.
2. Extraímos o conteúdo do `CarlaUE4Windows.zip`. A extração criou uma pasta chamada `CarlaSimulator` no diretório de trabalho, que hospeda os arquivos do servidor e cliente CARLA necessários para os projetos. O guia disponibilizado pelo autor ([University of Toronto, 2018a](#)) assume que o simulador é extraído para `C:\Coursera\CarlaSimulator`.

Instalação de Dependências Python para o Cliente As dependências adicionais necessárias para os arquivos do cliente do Simulador CARLA estão detalhadas no arquivo: `C:\Coursera\CarlaSimulator\requirements.txt`.

Foi executado os seguintes comandos para instalar essas dependências para o usuário atual:

```
> python -m pip install -r C:\Coursera\CarlaSimulator\requirements.txt  
--user [^2^] [2]
```

De modo a solucionar os erros que ocorreram durante essa instalação, recorremos ao material do capítulo [A](#).

Para mais detalhes e instruções completas para a instalação, ou se houver problemas ao instalar essas dependências do simulador CARLA, consulte o material do capítulo [A](#). Salientamos que os binários do CARLA utilizados aqui são uma versão modificada pelo autor ([University of Toronto, 2018a](#)) do CARLA original, com mapas adicionais incluídos.

5 Resultados e Discussões

delete below

Outros Aspectos Relevantes para Discussão

Além dos resultados acima, este capítulo pode abordar outros aspectos essenciais da pesquisa desenvolvida:

- **Avaliação Temporal do Desempenho do Sistema:** Apresentar os tempos médios de processamento por frame, latência de detecção e tempo de resposta do veículo a eventos visuais. Isso é fundamental para atestar a viabilidade do sistema em tempo real.
- **Taxa de Acerto e Métricas de Desempenho:** Discutir métricas como precision, recall, F1-score e mAP (mean Average Precision), calculadas a partir dos resultados das detecções em diferentes cenários dentro do CARLA.
- **Cenários de Teste e Robustez do Sistema:** Analisar o desempenho da solução em diferentes condições ambientais e de tráfego simuladas (ex: chuva, neblina, tráfego intenso, cruzamentos), avaliando a robustez do sistema.
- **Estimação de Distância:** Caso tenha implementado essa funcionalidade, mostrar como o sistema estima a distância dos objetos detectados e discutir sua precisão e utilidade para decisões de planejamento de trajetória.
- **Integração com o Planejamento de Movimento:** Discutir como as detecções foram utilizadas no processo de tomada de decisão (por exemplo, o veículo freou corretamente ao detectar a placa de "Stop"?). A sinergia entre percepção e controle é vital na avaliação de soluções autônomas.
- **Limitações Identificadas:** Listar os principais gargalos ou falhas observadas durante a execução do sistema, sejam elas relacionadas a performance computacional, falhas de detecção ou de atuação do veículo.
- **Comparações com Trabalhos Relacionados:** Discutir brevemente como os resultados obtidos se comparam com experimentos similares na literatura ou em outros projetos baseados no CARLA.

Essas análises ampliam a profundidade da discussão e demonstram o rigor científico necessário para um trabalho de conclusão de curso de qualidade, além de servirem como ponte para recomendações futuras de melhoria e extensão do sistema.

delete above

6 Considerações Finais

7 Perspectiva de continuidade

Referências

- ADHOC - Quick search results. 2024. <https://www.oed.com/search/dictionary/?scope=Entries&q=adhoc>. Acessado: 05-03-2024. Citado na página 46.
- AHAMMED, A. S.; HOSSAIN, M. S. A.; OBERMAISER, R. **A Computer Vision Approach for Autonomous Cars to Drive Safe at Construction Zone**. 2024. Preprint. Disponível em: <https://arxiv.org/abs/2409.15809>. Citado na página 49.
- AHIRE, P. *et al.* Simulating vehicle driving using carla. **Journal of Electrical Systems**, Engineering and Scientific Research Groups, v. 20, n. 10s, p. 44–52, 2024. Citado 4 vezes nas páginas 16, 17, 18 e 19.
- ANDRADE, G. G. d. Trabalho de Conclusão de Curso (Graduação), **Uma Proposta para Detecção de Objetos e Estimação de Distância em um Simulador de Veículos Autônomos**. 2022. Orientador: Prof. Giovanni Almeida Santos. Disponível em: https://github.com/guilherme1guy/carla_darknet_integration. Acesso em: 29 abr. 2025. Citado 2 vezes nas páginas 19 e 48.
- BOCHKOVSKIY, A.; WANG, C.-Y.; LIAO, H.-Y. M. **YOLOv4: Optimal Speed and Accuracy of Object Detection**. 2020. Disponível em: <https://arxiv.org/abs/2004.10934>. Citado 2 vezes nas páginas 22 e 36.
- BRATZEL, S.; MANAGEMENT, C. of A. **Die Zukunft der Mobilität – Die Zukunftstrends in den Bereichen Elektromobilität, Connected Car und Mobilitätsdienstleistungen**. 2022. Center of Automotive Management (CAM). Disponível em: <https://www.bnpparibas.de/de/studie-zu-aktuellem-umbruch-der-mobilitaetsbranche/>. Acesso em: 09 mar. 2024. Citado na página 16.
- BUSSEMAKER, K. Sensing requirements for an automated vehicle for highway and rural environments. Delft University of Technology, 2014. Citado na página 112.
- CARLA. [S.l.: s.n.]: CARLA Team, 2018. <https://carla.org/>. Acessado: 26-03-2024. Citado 3 vezes nas páginas 7, 47 e 48.
- CARLA Simulator. **CARLA Simulator: Cameras and Sensors**. [S.l.], 2024. Available in the CARLA Simulator documentation, Quick Start section. Acessado: 28-11-2024. Citado 4 vezes nas páginas 9, 12, 91 e 92.
- CRISTINA, P. M.; WAZLAWICK, P. R. S. **Metodologia de Pesquisa para Ciência da Computação**. 2021. https://edisciplinas.usp.br/pluginfile.php/4409810/mod_resource/content/2/2-MetodologiaDePesquisa-EstilosDePesquisa_cap2.pdf. Acessado: 26-11-2024. Citado 2 vezes nas páginas 7 e 21.
- Daniel. **Carla-Object-Detection-Dataset: Labeled Dataset for Object Detection in Carla Simulator**. 2023. Acessado: 26-11-2024. Disponível em: <https://github.com/DanielHfnr/Carla-Object-Detection-Dataset/tree/master>. Citado 2 vezes nas páginas 22 e 96.

- DOSOVITSKIY, A. *et al.* Carla: An open urban driving simulator. In: PMLR. **Proceedings of the Conference on Robot Learning**. 2017. p. 1–16. Disponível em: <https://proceedings.mlr.press/v78/dosovitskiy17a.html>. Acesso em: 29 abr. 2025. Citado 6 vezes nas páginas 16, 17, 18, 22, 46 e 47.
- DREIBELBIS, E. **Global fleet of autonomous vehicles may emit more carbon than Argentina**. 2023. <https://www.pcmag.com/news/global-fleet-of-autonomous-vehicles-may-emit-more-carbon-than-argentina>. Acessado: 2023-3-7. Citado na página 16.
- European Commission, Directorate-General for Mobility and Transport. **Safer roads for all: EU Road Safety Policy Framework 2021-2030**. 2021. Acessado: 11-07-2024. Disponível em: https://road-safety.transport.ec.europa.eu/system/files/2021-07/safer_roads4all.pdf. Citado na página 16.
- Federal Highway Administration. **The Relation Between Speed and Crashes**. 2012. Acessado: 11-07-2024. Disponível em: https://safety.fhwa.dot.gov/speedmgt/ref_mats/fhwasa1304/Resources3/08%20-%20The%20Relation%20Between%20Speed%20and%20Crashes.pdf. Citado na página 16.
- FLORES-CALERO, M. *et al.* Traffic sign detection and recognition using yolo object detection algorithm: A systematic review. **Mathematics**, v. 12, n. 2, 2024. ISSN 2227-7390. Disponível em: <https://www.mdpi.com/2227-7390/12/2/297>. Citado 2 vezes nas páginas 10 e 95.
- FOWLER, M. **The Doppler Effect**. [S.l.: s.n.]: University of Virginia, 2009. Citado na página 40.
- FRANCIS, B. A. *et al.* Models of mobile robots in the plane. **Flocking and rendezvous in distributed robotics**, Springer, p. 7–23, 2016. Citado 5 vezes nas páginas 8, 57, 59, 79 e 80.
- GAO, W.; TANG, J.; WANG, T. An object detection research method based on carla simulation. **Journal of Physics: Conference Series**, IOP Publishing, v. 1948, n. 1, p. 012163, jun 2021. Disponível em: <https://dx.doi.org/10.1088/1742-6596/1948/1/012163>. Acesso em: 29 abr. 2025. Citado na página 49.
- HOUSER, K. **Mercedes-Benz wins race to bring Level 3 autonomous cars to US**. 2023. <https://www.freethink.com/hard-tech/drive-pilot>. Acessado: 2023-3-29. Citado na página 29.
- IGNATIOUS, H. A.; HESHAM-EL-SAYED; KHAN, M. An overview of sensors in autonomous vehicles. **sciencedirect**, Elsevier, p. 1–6, 2022. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1877050921025540>. Citado 5 vezes nas páginas 37, 39, 40, 45 e 91.
- IRIE, K. **Yolov4_darknet: YOLOv4 - Neural Networks for Object Detection (Windows and Linux version of darknet)**. 2020. Citado na página 22.
- JACOBSON, B. *et al.* Vehicle dynamics. **Chalmers University of Technology**, Chalmers, 2016. Citado 2 vezes nas páginas 8 e 58.

- JACOBSON, B. *et al.* Vehicle dynamics compendium. **Chalmers University of Technology**, Chalmers, 2020. Disponível em: <https://research.chalmers.se/en/publication/520229>. Citado 7 vezes nas páginas 57, 58, 64, 66, 70, 77 e 112.
- JANAI, J. *et al.* Computer Vision for Autonomous Vehicles: Problems, Datasets and State of the Art. **Foundations and Trends in Computer Graphics and Vision**, v. 12, n. 1–3, p. 1–308, 2020. ISSN 1572-2740, 1572-2759. Citado 2 vezes nas páginas 16 e 18.
- JUANOLA, M. S. **Speed Traffic Sign Detection on the CARLA Simulator Using YOLO**. jul 2019. Dissertação (Dissertação (Mestrado em Sistemas Inteligentes Interativos)) — Universitat Pompeu Fabra, jul 2019. Treball fi de màster. Idioma: inglês. Disponível em: <http://hdl.handle.net/10230/42548>. Acesso em: 29 abr. 2025. Citado 5 vezes nas páginas 19, 22, 48, 50 e 96.
- KHAN, M. A. *et al.* Level-5 autonomous driving—are we there yet? a review of research literature. **ACM Computing Surveys (CSUR)**, ACM New York, NY, p. 1–38, 2022. Disponível em: https://www.researchgate.net/publication/358040996_Level-5_Autonomous_Driving_Are_We_There_Yet_A_Review_of_Research_Literature. Citado 8 vezes nas páginas 7, 17, 18, 19, 32, 33, 39 e 91.
- KIM, T.; JEON, H.; LIM, Y. Challenges of yolo series for object detection in extremely heavy rain: Calra simulator based synthetic evaluation data set. **arXiv preprint**, 2023. Disponível em: <https://arxiv.org/abs/2312.07976>. Acesso em: 29 abr. 2025. Citado 3 vezes nas páginas 49, 50 e 52.
- KPMG International. **2020 Autonomous Vehicles Readiness Index**. 2020. Disponível em: <https://assets.kpmg.com/content/dam/kpmg/xx/pdf/2020/07/2020-autonomous-vehicles-readiness-index.pdf>. Acesso em: 22 fev. 2023. Citado na página 16.
- LAGE, C. A. Quatro cenários para os veículos autônomos no mundo ocidental. **UNIVERSIDADE DE BRASÍLIA**, 2019. Citado 2 vezes nas páginas 16 e 28.
- LANCTOT, R. **Accelerating the Future: The Economic Impact of the Emerging Passenger Economy**. Strategy Analytics, 2017. Acessado: 11-07-2024. Disponível em: <https://www.intel.com/content/dam/www/public/us/en/documents/pdf/passenger-economy-report-autonomous-driving.pdf>. Citado na página 16.
- LEWIS, M.; JACOBSON, J. Game engines. **Communications of the ACM**, v. 45, n. 1, p. 27, 2002. Citado na página 47.
- LI, S.; WANG, S.; WANG, P. A Small Object Detection Algorithm for Traffic Signs Based on Improved YOLOv7. **Sensors**, MDPI, v. 23, p. 7145, 2023. Citado na página 17.
- LI, S.; WANG, S.; WANG, P. A small object detection algorithm for traffic signs based on improved yolov7. **Sensors**, v. 23, n. 16, 2023. ISSN 1424-8220. Disponível em: <https://www.mdpi.com/1424-8220/23/16/7145>. Citado 2 vezes nas páginas 19 e 51.
- MAO, J. *et al.* 3d object detection for autonomous driving: A comprehensive survey. **International Journal of Computer Vision**, Springer, p. 1–55, 2023. Citado na página 39.

- MITROPOULOS, A. Mercedes-benz erhält als weltweit erstes automobilunternehmen zertifizierung für sae level 3-system für us-markt. Mercedes-Benz Group AG, 2023. Disponível em: <https://group-media.mercedes-benz.com/marsMediaSite/de/instance/ko.xhtml?oid=55116818>. Citado na página 29.
- MOZAFFARI, O. Y. S. Deep learning-based vehicle behaviour prediction for autonomous driving applications: a review. *elsevier*, v. 2, p. 1–15, 2020. Citado na página 17.
- National Highway Traffic Safety Administration. **Critical Reasons for Crashes Investigated in the National Motor Vehicle Crash Causation Survey: DOT HS 812 506 A Brief Statistical Summary**. 2018. Disponível em: <https://crashstats.nhtsa.dot.gov/Api/Public/Publication/812506>. Acesso em: 11 jul. 2024. Citado 3 vezes nas páginas 16, 18 e 20.
- NEUFVILLE, R.; ABDALLA, H.; ABBAS, A. Potential of connected fully autonomous vehicles in reducing congestion and associated carbon emissions. *Sustainability*, v. 14, n. 11, 2022. ISSN 2071-1050. Disponível em: <https://www.mdpi.com/2071-1050/14/11/6910>. Acesso em: 29 abr. 2025. Citado na página 17.
- OKPONO, J. *et al.* Advanced driver assistance systems road accident data insights: Uncovering trends and risk factors. *The International Journal of Engineering Research*, v. 11, n. 9, p. a141–a152, 2024. Disponível em: <https://tijer.org/tijer/papers/TIJER2409017.pdf>. Acesso em: 29 abr. 2025. Citado 4 vezes nas páginas 16, 17, 18 e 20.
- OTHMAN, K. Multidimension analysis of autonomous vehicles: The future of mobility. *Civil Engineering Journal*, v. 7, n. 7, p. 71–93, 2021. Disponível em: <https://www.civilejournal.org/index.php/cej/article/view/3181>. Acesso em: 29 abr. 2025. Citado na página 16.
- PADEN, B. *et al.* A survey of motion planning and control techniques for self-driving urban vehicles. *IEEE Transactions on Intelligent Vehicles*, v. 1, n. 1, p. 33–55, 2016. Disponível em: <https://ieeexplore.ieee.org/document/7490340>. Acesso em: 29 abr. 2025. Citado 2 vezes nas páginas 45 e 57.
- PAREKH NISHI PODDAR, M. C. D. A review on autonomous vehicles: Progress, methods and challenges. *Electronics*, 2022. ISSN 2525-8761. Citado 4 vezes nas páginas 7, 16, 31 e 91.
- PIVTORAIKO, M.; KNEPPER, R. A.; KELLY, A. Differentially constrained mobile robot motion planning in state lattices. *Journal of Field Robotics*, Wiley Online Library, v. 26, n. 3, p. 308–333, 2009. Citado 2 vezes nas páginas 89 e 90.
- RAJAMANI, R. **Vehicle dynamics and control**. [S.l.: s.n.]: Springer Science & Business Media, 2011. Citado 3 vezes nas páginas 59, 79 e 80.
- REDMON, J. **Darknet: Open Source Neural Networks in C**. 2013–2016. <http://pjreddie.com/darknet/>. Acessado: 28-11-2024. Citado 2 vezes nas páginas 22 e 52.
- REDMON, J. **You Only Look Once (YOLO) - Darknet**. 2018. Disponível em: <https://pjreddie.com/darknet/yolo/>. Acesso em: 28 nov. 2024. Citado 2 vezes nas páginas 92 e 96.

- REDMON, J. *et al.* **You Only Look Once: Unified, Real-Time Object Detection**. 2016. Disponível em: <https://arxiv.org/abs/1506.02640>. Acesso em: 29 abr. 2025. Citado 10 vezes nas páginas 7, 9, 10, 35, 36, 52, 92, 93, 94 e 95.
- REDMON, J.; FARHADI, A. **YOLO9000: Better, Faster, Stronger**. 2016. Disponível em: <https://arxiv.org/abs/1612.08242>. Citado na página 36.
- REDMON, J.; FARHADI, A. **YOLOv3: An Incremental Improvement**. 2018. Disponível em: <https://arxiv.org/abs/1804.02767>. Citado 3 vezes nas páginas 36, 52 e 92.
- REINHOLTZ, C. *et al.* Darpa urban challenge technical paper. Citeseer, 2007. Disponível em: https://ptolemy.berkeley.edu/projects/chess/pubs/379/DARPA_Urban_Challenge_Sydney_Berkeley_B161-TechnicalPaper.pdf. Acesso em: 29 abr. 2025. Citado na página 42.
- SAE. Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles. **SAE Mobilius**, SAE International, p. 1–41, 2021. Disponível em: https://www.sae.org/standards/content/j3016_202104. Citado 9 vezes nas páginas 7, 16, 17, 18, 27, 28, 29, 32 e 112.
- SEBO, D. Impact of electric vehicle market growth on automotive industry transformation: Trends, potentials, and challenges analysis. In: **Economic and Social Development (Book of Proceedings), 106th International Scientific Conference on Economic and Social**. [S.l.: s.n.], 2024. p. 73. Disponível em: <https://www.crooris.hr/crosbi/publikacija/prilog-skup/820618>. Acesso em: 29 abr. 2025. Citado na página 16.
- SINGH, G. B. . K. B. . R. K. Autonomous vehicles and intelligent automation: Applications, challenges, and opportunities. Hindawi, 2022. Citado 4 vezes nas páginas 7, 17, 38 e 91.
- SNIDER, J. M. *et al.* Automatic steering methods for autonomous automobile path tracking. **Robotics Institute, Pittsburgh, PA, Tech. Rep. CMU-RITR-09-08**, 2009. Citado 6 vezes nas páginas 79, 80, 83, 84, 86 e 112.
- SURENDRA, E. al. H. Lane detection and traffic sign detection using deep learning and computer vision for autonomous driving research using carla simulator. **Miscellaneous**, v. 11, n. 10, p. 2062, 2023. Disponível em: <https://doi.org/10.17762/ijritcc.v11i10.8891>. Acesso em: 29 abr. 2025. Citado 6 vezes nas páginas 7, 8, 49, 50, 51 e 52.
- University of Toronto. **Introduction to Self-Driving Cars**. Coursera Inc, 2018. Taught by Steven Waslander and Jonathan Kelly, Associate Professors in Aerospace Studies. Part of the Self-Driving Cars Specialization. Advanced level. Includes a project using the CARLA simulation environment. Online. Disponível em: <https://www.coursera.org/learn/intro-self-driving-cars?specialization=self-driving-cars>. Acesso em: 01 mar. 2024. Citado 46 vezes nas páginas 7, 8, 9, 11, 38, 41, 42, 43, 44, 45, 46, 47, 54, 55, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 97 e 98.
- University of Toronto. **Motion Planning for Self-Driving Cars**. Coursera Inc, 2018. Taught by Steven Waslander and Jonathan Kelly, Associate Professors in Aerospace Studies. Part of the Self-Driving Cars Specialization. Advanced level. Online. Disponível

- em: <https://www.coursera.org/learn/motion-planning-self-driving-cars>. Acesso em: 24 nov. 2024. Citado 5 vezes nas páginas 9, 87, 88, 89 e 90.
- WACHENFELD, W.; WINNER, H. The Release of Autonomous Vehicles. In: MAURER, M. et al. (ed.). **Autonomous Driving: Technical, Legal and Social Aspects**. Berlin, Heidelberg: Springer, 2016. p. 425–449. ISBN 978-3-662-48847-8. Citado 2 vezes nas páginas 17 e 18.
- WANG, A. et al. **YOLOv10: Real-Time End-to-End Object Detection**. 2024. Disponível em: <https://arxiv.org/abs/2405.14458>. Citado 2 vezes nas páginas 7 e 37.
- WANG, C.-Y.; LIAO, H.-Y. M. YOLOv1 to YOLOv10: The Fastest and Most Accurate Real-Time Object Detection Systems. **arXiv:2408.09332v1 [cs.CV]**, ago. 2024. Disponível em: <https://arxiv.org/html/2408.09332v1>. Acesso em: 29 abr. 2025. Citado 2 vezes nas páginas 36 e 37.
- WANG, H.; YU, H. Traffic sign detection algorithm based on improved yolov4. In: **2020 IEEE 9th Joint International Information Technology and Artificial Intelligence Conference (ITAIC)**. [S.l.: s.n.], 2020. v. 9, p. 1946–1950. Citado na página 19.
- WU, T. et al. Physical Adversarial Attack on Vehicle Detector in the Carla Simulator. **arXiv:2007.16118 [cs]**, ago. 2020. Disponível em: <<http://arxiv.org/abs/2007.16118>>. Disponível em: <http://arxiv.org/abs/2007.16118>. Citado na página 52.
- WU, X.; CAO, H. Traffic sign detection algorithm based on improved yolov4. **Journal of Physics: Conference Series**, IOP Publishing, v. 2258, n. 1, p. 012009, apr 2022. Disponível em: <https://dx.doi.org/10.1088/1742-6596/2258/1/012009>. Citado 2 vezes nas páginas 22 e 51.
- YAO, S. et al. Radar-camera fusion for object detection and semantic segmentation in autonomous driving: A comprehensive review. **IEEE Transactions on Intelligent Vehicles**, v. 9, n. 1, p. 2094–2128, 2024. Citado na página 39.
- ZHENG, S. et al. Simultaneous localization and mapping (slam) for autonomous driving: Concept and analysis. **Remote Sensing**, v. 15, n. 4, 2023. ISSN 2072-4292. Disponível em: <https://www.mdpi.com/2072-4292/15/4/1156>. Citado 9 vezes nas páginas 7, 34, 35, 40, 41, 43, 44, 45 e 77.

Apêndices

APÊNDICE A – Material Completo

Neste apêndice, disponibilizamos o link para o material completo desenvolvido ao longo do presente trabalho. O conteúdo abrange todos os códigos, dados, gráficos e informações detalhadas referentes ao tema em questão. O acesso ao material completo proporciona uma compreensão mais aprofundada do trabalho realizado, permitindo uma análise mais minuciosa dos resultados obtidos.

Para acessar o material completo sobre a subseção **Configurando o Simulador Carla 4.1**, clique no seguinte link: https://github.com/ARRETdaniel/Self-Driving_Cars_Specialization/tree/main/CARLA%20Installation%20Guide

Recomendamos a exploração deste recurso para uma apreciação abrangente das etapas apresentadas ao longo do trabalho. O material está disponível online para facilitar o acesso e a referência contínua. Se houver problema ao tentar consultar qualquer um desses materiais, não hesite em nos contactar via e-mail: danielterra@pq.uenf.br.

Agradecemos a atenção e interesse na pesquisa apresentada, esperamos que o material disponibilizado enriqueça ainda mais a compreensão sobre o assunto abordado.

Anexos

ANEXO A – Material de Relevância

Neste anexo, fornecemos uma descrição dos materiais relevantes para aprofundamento relacionados a esta iniciação científica, e a sua contribuição.

Iniciamos com o artigo *Vehicle Dynamics COMPENDIUM* de 2020, publicado pela Chalmers University of Technology ([Jacobson et al., 2020](#)). Este trabalho é essencial para compreender todos os aspectos relacionados à modelagem abordada nesta pesquisa. Ele oferece a base necessária para a compreensão da modelagem lateral, longitudinal e dos subsistemas dos VAs, incluindo aspectos avançados não abordados neste estudo.

Adicionalmente, temos a dissertação de mestrado *Sensing requirements for an automated vehicle for highway and rural environments*, de 2014, que aborda todos os sensores e métricas associados aos VAs, bem como análises desses sensores em diferentes contextos de aplicação ([Bussemaker, 2014](#)).

Além disso, mencionamos o documento SAE *International J3016* de 2021, elaborado para descrever sistemas autônomos ([SAE, 2021](#)). Ele engloba todas as discussões e definições relevantes para caracterizar e definir os níveis de condução autônoma.

Ademais, o artigo *Automatic Steering Methods for Autonomous Automobile Path Tracking* de 2009 contribuiu de maneira significativa, auxiliando-nos na compreensão de algumas das equações presentes nos modelos apresentados neste trabalho de iniciação científica ([Snider et al., 2009](#)).

Por fim, desenvolvemos de maneira paralela a este trabalho sobre o **Modelo Cinemático da Bicicleta** os códigos e materiais referentes a implementação podem ser encontrados no seguinte link: https://github.com/ARRETdaniel/Self-Driving_Cars_Specialization/blob/main/Introduction%20to%20Self-Driving%20Cars/week4/module_4/Course_1_Module_4/Kinematic_Bicycle_Model.ipynb De mesma forma, implementamos sobre **Modelo Longitudinal de Veículo**, acesso à implementação a partir do link: https://github.com/ARRETdaniel/Self-Driving_Cars_Specialization/blob/main/Introduction%20to%20Self-Driving%20Cars/week4/module_4/Course_1_Module_4/Longitudinal_Vehicle_Model.ipynb